Discriminative Sequence Models and Conditional Random Fields

Mark Gales

Lent 2014

Machine Learning for Language Processing: Lecture 6

MPhil in Advanced Computer Science

Sequence Models

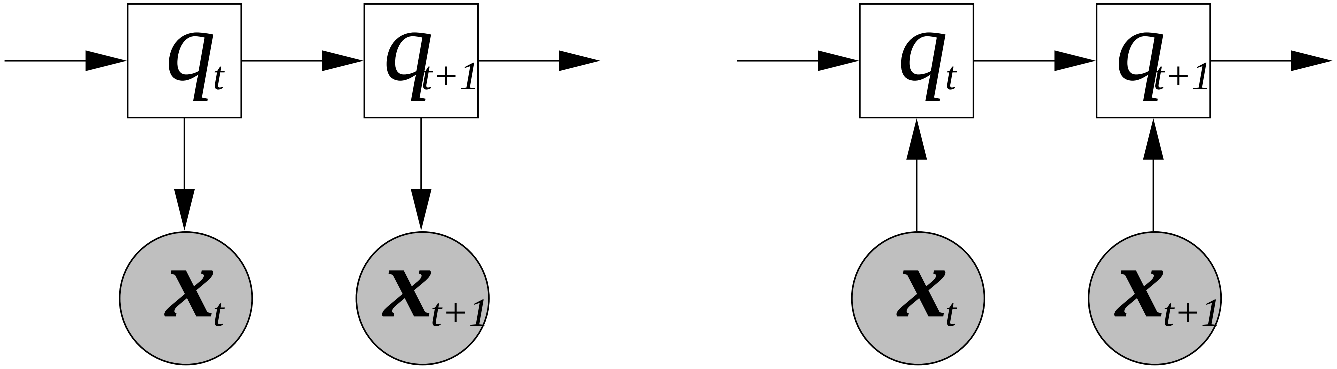
- So far examined the hidden Markov model (HMM) as a sequence model
  - generative model of the data sequence, $P(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T | q_0, \ldots, q_{T+1})$,
  - use Bayes' rule to yield "class sequence" posteriors $P(\boldsymbol{y}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$
  - here $\boldsymbol{y} = \{y_0, \ldots, y_{T+1}\}$ (the states are associated with classes)
- HMM parameters usually trained using maximum likelihood
  - possible to also use discriminative training criteria to estimate parameters $\boldsymbol{\lambda}$
  - conditional maximum likelihood, maximise label posterior, $P(\boldsymbol{y}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$

$$\hat{\boldsymbol{\lambda}} = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \left\{ \sum_{r=1}^{R} \log \left(\frac{P(\boldsymbol{y}^{(r)}) P(\boldsymbol{x}_1^{(r)}, \ldots, \boldsymbol{x}_{T_r}^{(r)} | \boldsymbol{y}^{(r)}, \boldsymbol{\lambda})}{\sum_{\boldsymbol{q} \in \boldsymbol{Q}_{T_r}} P(\boldsymbol{q}) P(\boldsymbol{x}_1^{(r)}, \ldots, \boldsymbol{x}_{T_r}^{(r)} | \boldsymbol{q}, \boldsymbol{\lambda})} \right) \right\}$$

  - R sequences, labels $\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(R)}$
  - sequence r is of length T_r, with observations $\boldsymbol{x}_1^{(r)}, \ldots, \boldsymbol{x}_{T_r}^{(r)}$

What about discriminative sequence models?

Discriminative Sequence Models



Generative Model Discriminative Model

- Simple generative model (left) and discriminative model (right)
  - right BN a maximum entropy Markov model (q_{T+1} dropped for simplicity)

$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \prod_{t=1}^{T} P(q_t | q_{t-1}, \boldsymbol{x}_t)$$

  state posterior probability given by (Z_t normalisation term at time t)

$$P(q_t | q_{t-1}, \boldsymbol{x}_t) = \frac{1}{Z_t} \exp\left(\sum_{i=1}^{D} \lambda_i f_i(q_t, q_{t-1}, \boldsymbol{x}_t) \right)$$

Sequence Maximum Entropy Models

- State posteriors modelled in the Maximum Entropy Markov model
  - could extend to the complete sequence

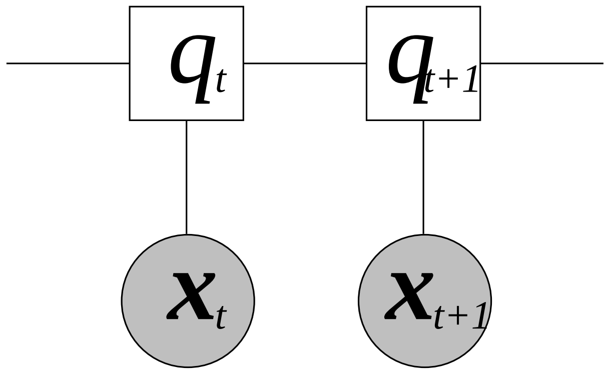
$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \frac{1}{Z} \exp\left(\sum_{i=1}^{D} \lambda_i f_i(q_0, \ldots, q_T, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)\right)$$

- Problem is that there are a vast number of possible features

What features to extract from the state/observation sequence?

- need to be able to handle variations in length of the sequence
- keep the number of model parameters $\boldsymbol{\lambda}$ reasonable

(Simple) Linear Chain Conditional Random Fields



- Extract features based on undirected graph
  - conditional independence assumptions similar to HMM (though undirected)

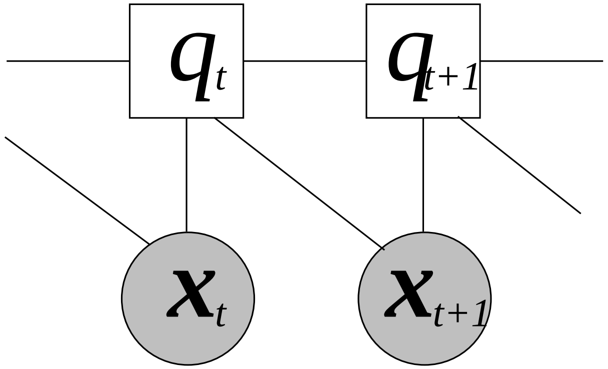
- Posterior model becomes

$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T} \left(\sum_{i=1}^{D_{\mathsf{t}}} \lambda_i^{\mathsf{t}} f_i(q_t, q_{t-1}) + \sum_{i=1}^{D_{\mathsf{a}}} \lambda_i^{\mathsf{a}} f_i(q_t, \boldsymbol{x}_t) \right) \right)$$

  - D_{t} number of transition style features with parameters $\boldsymbol{\lambda}^{\mathsf{t}}$
  - D_{a} number of acoustic style features with parameters $\boldsymbol{\lambda}^{\mathsf{a}}$

- This has some relationships to HMMs for particular forms of features (though training different)

Linear Chain Conditional Random Fields



- Extract features based on undirected graph
  - conditional independence assumptions extended to previous state

- Posterior model becomes

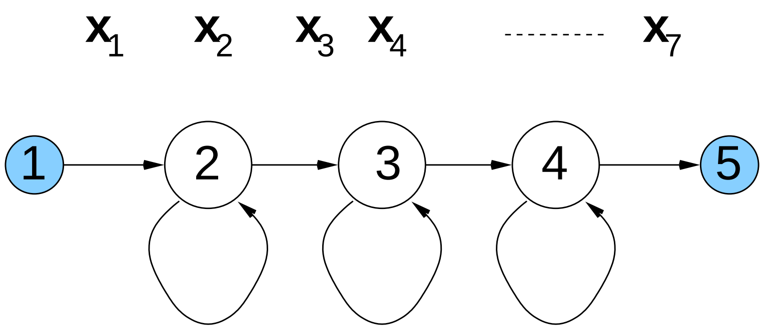
$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T}\left(\sum_{i=1}^{D} \lambda_i f_i(q_t, q_{t-1}, \boldsymbol{x}_t)\right)\right)$$

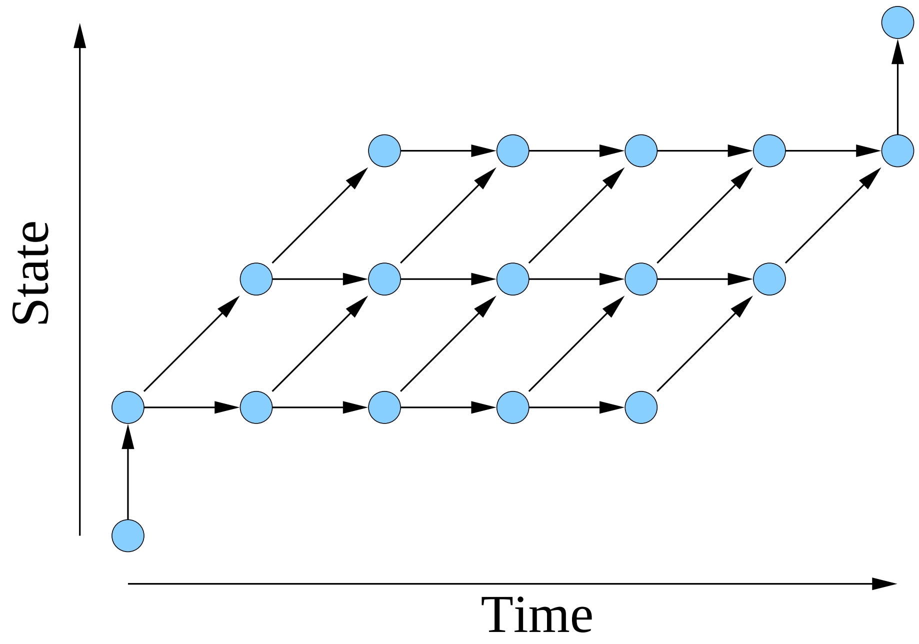
- More interesting than HMM-like features
  - features the same as MaxEnt Markov model
  - BUT normalised globally not locally

Normalisation term

- Need to be able to compute the normalisation term efficiently
  - initially consider the simple linear chain case

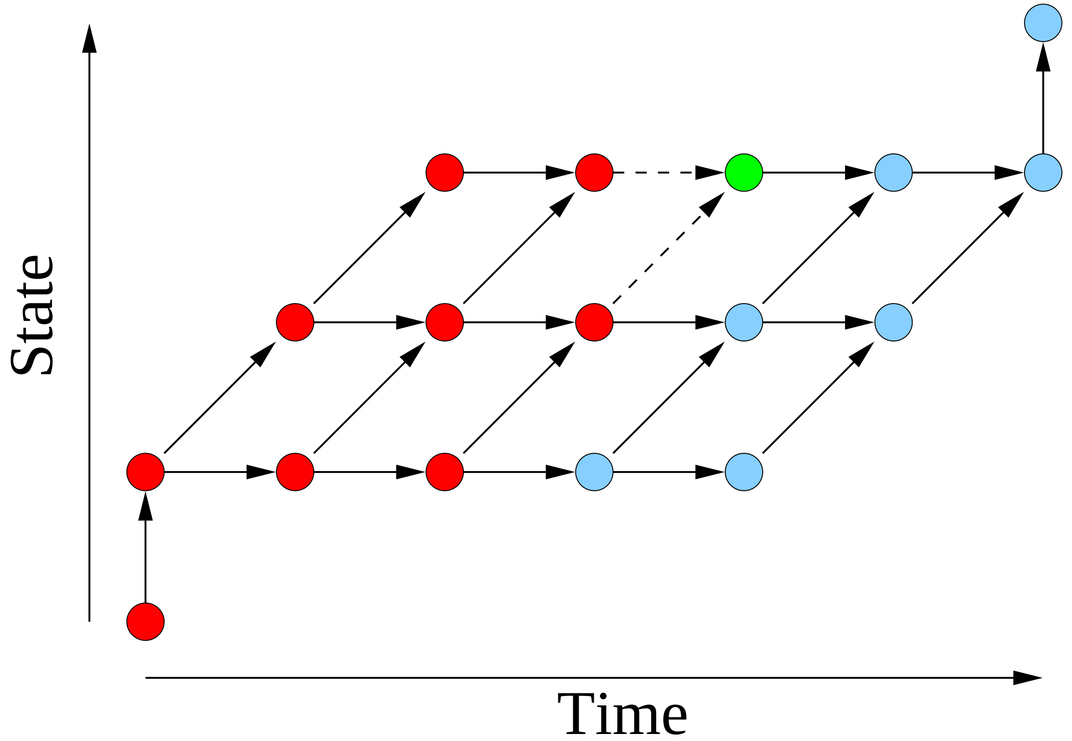
$$Z = \sum_{\boldsymbol{q} \in \boldsymbol{Q}_T} \exp\left(\sum_{t=1}^{T}\left(\sum_{i=1}^{D_{\mathtt{t}}} \lambda_i^{\mathtt{t}} f_i(q_t, q_{t-1}) + \sum_{i=1}^{D_{\mathtt{a}}} \lambda_i^{\mathtt{a}} f_i(q_t, \boldsymbol{x}_t)\right)\right)$$





- Consider same topology and observation sequence $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_7$ as the HMM

Total Path Cost to a State/Time



- Red possible partial paths
- Green state of interest

$$\text{LAdd}(a,b) = \log\left(\exp(a)+\exp(b)\right)$$
$$\exp(\text{LAdd}(a,b)) = \exp(a)+\exp(b)$$

- Total path cost to state $\mathtt{s}_i$ at time t is $\alpha_i(t)$
  - total path cost to state $\mathtt{s}_4$ at time 5 given by (compare to Viterbi)

$$\alpha_4(5) = \text{LAdd}\left(\alpha_3(4) + \sum_{i=1}^{D_{\mathtt{t}}} \lambda_i^{\mathtt{t}} f_i(\mathtt{s}_4,\mathtt{s}_3), \alpha_4(4) + \sum_{i=1}^{D_{\mathtt{t}}} \lambda_i^{\mathtt{t}} f_i(\mathtt{s}_4,\mathtt{s}_4)\right) + \sum_{i=1}^{D_{\mathtt{a}}} \lambda_i^{\mathtt{a}} f_i(\mathtt{s}_4,\boldsymbol{x}_5)$$

Forward-Backward Algorithm

- α is related to the forward-"probability" that is used to train HMMs
  - recursion for this form of model can be expressed as

$$\alpha_j(t) = \log\left(\sum_{k=1}^{N}\exp\left(\alpha_k(t-1) + \sum_{i=1}^{D_{\mathtt{t}}}\lambda_i^{\mathtt{t}} f_i(\mathbf{s}_j, \mathbf{s}_k)\right)\right) + \sum_{i=1}^{D_{\mathtt{a}}}\lambda_i^{\mathtt{a}} f_i(\mathbf{s}_j, \boldsymbol{x}_t)$$

  - normalisation term can then be expressed as $Z = \exp(\alpha_N(T))$
- There's also a term related to the backward-"probability"
  - consider observation at time t given state $\mathbf{s}_j$, $\beta_j(t)$

$$\beta_j(t) = \log\left(\sum_{k=1}^{N}\exp\left(\beta_k(t+1) + \sum_{i=1}^{D_{\mathtt{t}}}\lambda_i^{\mathtt{t}} f_i(\mathbf{s}_k, \mathbf{s}_j) + \sum_{i=1}^{D_{\mathtt{a}}}\lambda_i^{\mathtt{a}} f_i(\mathbf{s}_k, \boldsymbol{x}_{t+1})\right)\right)$$

  - designed so that $Z = \sum_{i=1}^{N}\exp\left(\alpha_i(t) + \beta_i(t)\right)$

(Aside) HMM-Training using EM

- The forward-backward algorithm used in EM training of HMMs
  - enables latent variable posteriors $P(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\lambda})$ to be computed
  - similar form to simple linear chain CRF

$$\sum_{i=1}^{D_{\mathsf{t}}} \lambda_i^{\mathsf{t}} f_i(q_t = \mathbf{s}_j, q_{t-1} = \mathbf{s}_i): \quad \log(P(q_t = \mathbf{s}_j, q_{t-1} = \mathbf{s}_i)) = \log(a_{ij})$$

$$\sum_{i=1}^{D_{\mathsf{a}}} \lambda_i^{\mathsf{a}} f_i(q_t = \mathbf{s}_j, \boldsymbol{x}_t): \quad \log(p(\boldsymbol{x}_t|q_t = \mathbf{s}_j)) = \log(b_j(\boldsymbol{x}_t))$$

- (Log) forward $\alpha_j(t)$ and (log) backward probabilities, $\beta_j(t)$:

$$\alpha_j(t) = \log(p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, q_t = \mathbf{s}_j)) = \log\left(\sum_{k=1}^{N} a_{kj} \exp\left(\alpha_k(t-1)\right)\right) + \log(b_j(\boldsymbol{x}_t))$$

$$\beta_j(t) = \log(p(\boldsymbol{x}_{t+1}, \ldots, \boldsymbol{x}_T|q_t = \mathbf{s}_j)) = \log\left(\sum_{k=1}^{N} a_{jk} b_k(\boldsymbol{x}_{t+1}) \exp\left(\beta_k(t+1)\right)\right)$$

(Aside) HMM-Update Formulae

- Forward and backward probabilities can be used to derive posteriors
  - at iteration l

$$\gamma_j^{[l]}(t) = P(q_t = \mathbf{s}_j | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, \boldsymbol{\lambda}^{[l]}) = \exp\left(\alpha_j^{[l]}(t) + \beta_j^{[l]}(t) - \alpha_N^{[l]}(T)\right)$$

- Update formulae with Gaussian state output distribution $b_j(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$

$$\boldsymbol{\mu}_j^{[l+1]} = \frac{\sum_{t=1}^{T} \gamma_j^{[l]}(t)\boldsymbol{x}_t}{\sum_{t=1}^{T} \gamma_j^{[l]}(t)}$$

$$\boldsymbol{\Sigma}_j^{[l+1]} = \frac{\sum_{t=1}^{T} \gamma_j^{[l]}(t)\boldsymbol{x}_t\boldsymbol{x}_t^{\mathsf{T}}}{\sum_{t=1}^{T} \gamma_j^{[l]}(t)} - \boldsymbol{\mu}_j^{[l+1]}\boldsymbol{\mu}_j^{[l+1]\mathsf{T}}$$

General Sequence CRFs

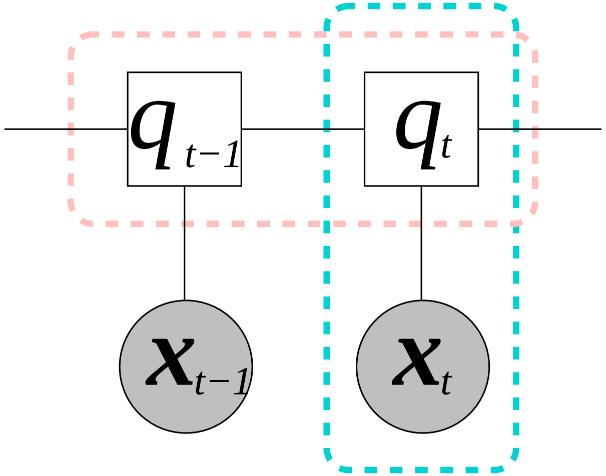
- The general form of CRF uses an undirected graphical model to define features
  - need to be able to handle sequence data - dynamic CRF
  - undirected graph repeated each time instance - set of cliques is $\boldsymbol{C}$

- The posterior probability for this form of model is

$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T} \sum_{\mathcal{C} \in \boldsymbol{C}} \boldsymbol{\lambda}_{\mathcal{C}}^{\mathsf{T}} \mathbf{f}(\boldsymbol{q}_{\mathcal{C}t}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, t) \right)$$

  - $\boldsymbol{\lambda}_{\mathcal{C}}^{\mathsf{T}}$ time-independent parameters associated with clique $\mathcal{C}$
  - $\mathbf{f}(\boldsymbol{q}_{\mathcal{C}t}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, t)$ time-dependent features extracted from clique $\mathcal{C}$ with time-dependent label sequence $\boldsymbol{q}_{\mathcal{C}t}$

Example of a Sequence CRF



- Cliques associated with linear CRF

$$\boldsymbol{C} = \{\mathcal{C}_1, \mathcal{C}_2\}$$

1. transitions: $\mathcal{C}_1 = \{q_t, q_{t-1}\}$
2. acoustics: $\mathcal{C}_2 = \{q_t, \boldsymbol{x}_t\}$

- Posterior model for the simple linear chain CRF

$$P(q_0, \ldots, q_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T} \sum_{\mathcal{C} \in \boldsymbol{C}} \boldsymbol{\lambda}_{\mathcal{C}}^{\mathsf{T}} \mathbf{f}(\boldsymbol{q}_{\mathcal{C}t}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, t) \right)$$

$$= \frac{1}{Z} \exp\left(\sum_{t=1}^{T} \left(\boldsymbol{\lambda}^{\mathsf{t}\mathsf{T}} \mathbf{f}(q_t, q_{t-1}) + \boldsymbol{\lambda}^{\mathsf{a}\mathsf{T}} \mathbf{f}(q_t, \boldsymbol{x}_t) \right) \right)$$

Training CRFs

- Training for CRFs is normally fully observed

training observation sequence	$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$
training label sequence	$y_1, \ldots, y_T$

  - where $y_\tau \in \{\omega_1, \ldots, \omega_K\}$
- No need to use EM (or related approaches)
  - extension to CRFs includes additional latent variables hidden CRFs
  - training data for HCRFs only partially observed
- Need to find the model parameters $\boldsymbol{\lambda}$ so that

$$\begin{aligned}
\hat{\boldsymbol{\lambda}} &= \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \left\{ P(y_1, \ldots, y_T | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, \boldsymbol{\lambda}) \right\} \\
&= \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \left\{ \frac{1}{Z} \exp\left(\sum_{i=1}^{D} \lambda_i f_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, y_1, \ldots, y_T) \right) \right\}
\end{aligned}$$

Generalised Iterative Scaling for CRFs

- CRF (also MaxEnt model) training is a convex optimisation problem
  - one solution to train parameters is generalised iterative scaling

$$\lambda_i^{[k+1]} = \lambda_i^{[k]} + \frac{1}{C}\log\left(\frac{f_i(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T,y_1,\ldots,y_T)}{\sum_{\boldsymbol{q}\in\boldsymbol{Q}_T} P(\boldsymbol{q}|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T,\boldsymbol{\lambda}^{[k]})f_i(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T,\boldsymbol{q})}\right)$$

  - iterative approach (parameters at iteration k are $\boldsymbol{\lambda}^{[k]}$)

- (strictly) requires that the features add up to a constant

$$\sum_{i=1}^{D} f_i(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T,\boldsymbol{q}) = C, \quad \forall\boldsymbol{q}\in\boldsymbol{Q}_T$$

  - extensions relaxes this requirements, e.g. improved iterative scaling

Inference with CRFs

- Recognition with CRFs involves finding the most probable label sequence $\hat{\boldsymbol{q}}$

$$\begin{aligned}\hat{\boldsymbol{q}} &= \underset{\boldsymbol{q}\in\boldsymbol{Q}_T}{\operatorname{argmax}} \{P(\boldsymbol{q}|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T)\} \\ &= \underset{\boldsymbol{q}\in\boldsymbol{Q}_T}{\operatorname{argmax}} \left\{\sum_{i=1}^{D} \lambda_i f_i(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T,\boldsymbol{q})\right\}\end{aligned}$$

  – normalisation term Z not used as it is the same for all label sequences

- The Viterbi algorithm is often used to perform recognition
  – for the simple linear chain CRF relationship to HMM Viterbi clear:

$$\hat{\boldsymbol{q}} = \underset{\boldsymbol{q}\in\boldsymbol{Q}_T}{\operatorname{argmax}} \left\{\sum_{t=1}^{T}\left(\sum_{i=1}^{D_{\mathtt{t}}} \lambda_i^{\mathtt{t}} f_i(q_t,q_{t-1}) + \sum_{i=1}^{D_{\mathtt{a}}} \lambda_i^{\mathtt{a}} f_i(q_t,\boldsymbol{x}_t)\right)\right\}$$