Neural networks
Training CRFs - pseudolikelihood
**GENERAL CRF**

**Topics:** CRFs in general

- Gradients in general CRFs always take the form:

\[
\frac{\partial - \log p(y^{(t)} | X^{(t)})}{\partial \theta} = - \left( \sum_f \frac{\partial}{\partial \theta} \log \Psi_f(y^{(t)}, X^{(t)}) \right)
\]

- The expectation over \( y \) will often need to be approximated, using loopy belief propagation
  
  - it will often involve only a few of the \( y_k \) variables
Topics: pseudolikelihood

• Why not just change the loss function to a tractable one

\[- \sum_{k=1}^{K} \log p(y_k | y_1, \ldots, y_{k-1}, y_{k+1}, \ldots, y_K, X)\]

› predict, in turn, each \( y_k \) not just from \( X \), but also all the other elements of \( y \)
› can compute the exact gradients
  - the probabilities only require normalizing \( y_k \) individually, like in a regular softmax
  - each conditional often only depend on few variables (local Markov property)
› however, often tends to work less well
› we still need to compute \( p(y_k | X) \) to do predictions anyways