Generativ odels

Hidden Markov Models (HMM): the generative story

The generative story: first, the tags are drawn from a prior distribution; next, the tokens are drawn from a cond likelihood.

$y_0 \leftarrow \Diamond$, $m \leftarrow$

repeat

  $y_m \sim Categorical(\lambda_{y_{m-1}})$ e current tag

  $w_m \sim Categorical(\phi_{y_m})$ urrent word

until $y_m = \blacklozenge$ $\triangleright$ terminate when the stop symbol is generated

The independence assumptions of HMM

In addition to the usual in yes, two additional independence assumptions are needed for HMM:

- The probability of each word token only depends on its tag, not on any other element in the sequence:

m

- Each tag y_m depends only on its pred

$$P(\boldsymbol{y}) = \prod_{m=1}^{M} P(y_m | y_{m-1})$$

when $y_m = \Diamond$ in all cases.

Parameter estimation of HMMs

The hidden Markov model has two groups of parameters:

- **Emission probabilities** ϕ. The probability $P_e(w_m|y_m;\phi)$ is the emission probability
- **Transition probabilities** λ. The prob $|y_{m-1};\lambda)$

Both of these groups relative frequency e thed probabilities are,

$$\phi_{k,i} \triangleq Pr(W_m = i | Y_m = k) = \frac{count(W_m = i, Y_m = k)}{count(Y_m = k)}$$

$$\lambda_{k,k'} \triangleq Pr(Y_m = k' | Y_{m=1} = k) = \frac{count(Y_m = k', Y_{m-1} = k)}{count(Y_{m-1} = k)}$$

Inference for HMMs

- The goal of the inference in the hidden Markov model is to find the highest probability sequence:

$$\hat{\boldsymbol{y}} = \operatorname{argmax} \log$$

- As $P(\boldsymbol{y}, \boldsymbol{w})$ e inference problem can be reformulated as finding t f $\boldsymbol{w}$ and $\boldsymbol{y}$:

$$\hat{\boldsymbol{y}} = \operatorname*{argmax}_{\boldsymbol{y}} \log P(\boldsymbol{y}, \boldsymbol{w})$$

Inference for HMMs

- Applying the inde

$$
\begin{aligned}
\log P(\boldsymbol{y}, \boldsymbol{w}) &= \log P(\boldsymbol{y}) + \log P(\boldsymbol{w} \mid \boldsymbol{y}) \\
&= \sum_{m=1}^{M+1} \log P_Y(y_m \mid y \quad (w_m \mid y_m) \\
& \qquad w_m \\
&= \sum_{m=1}^{M+1} s_m(y_m, y_m
\end{aligned}
$$

- This allows us to apply a variant of the Viterbi algorithm, where the only parameters are the transition probabilities and emission probabilities, which correspond to two features at each position in the sequence. This limitation explains the performance disadvantage of HMMs.

Discriminative alternatives to HMMs

As with Naïve Bayes, o al features (e.g., suffixes) cannot be applied without violating the independent assumptions. Discriminative models do not have this problem and additional features can be included:

$$
\begin{aligned}
&\boldsymbol{f}(\boldsymbol{w} = \text{the man who whistles tunes pianos}, \boldsymbol{y} = \text{DT NN WP VBZ VBZ NNS}) \\
&= f(w_0 = \mathit{the}, y_0 \\
&\quad + f(w_0 = \mathit{man}, \\
&\quad + f(w_0 = \mathit{who}, y_0 = \mathit{WP}) + f(y_0 = \mathit{WP} \\
&\quad + f(w_0 = \mathit{whistles}, y_0 = \mathit{VBZ}) + f(y_0 = \quad + f(\mathit{suffix}_0 = \mathit{es}) \\
&\quad + f(w_0 = \mathit{tunes}, y_0 = \mathit{VBZ}) + f(y_0 = \mathit{VBZ}, y_{-1} = \mathit{VBZ}) + f(\mathit{suffix}_0 = \mathit{es}) \\
&\quad + f(w_0 = \mathit{pianos}, y_0 = \mathit{NNS}) + f(y_0 = \mathit{NNS}, y_{-1} = \mathit{VBZ}) \\
&\quad + f(y_0 = \blacklozenge, y_{0-1} = \mathit{NNS})
\end{aligned}
$$

Note that you do not need to add the same morphological feature for each word token.

Structur ling

Structured Perceptron

- Each tagging seq

 model

$$\Psi(\boldsymbol{w}, \boldsymbol{y}) = \sum_{m=1}^{M+1} \psi(\boldsymbol{w}, y_m, y_{m-1}, m)$$

$$\sum^{M+1} \qquad {}_{1}, m)$$

$$= \boldsymbol{\theta} \cdot \sum_{m=1} \qquad m)$$

$$= \boldsymbol{\theta} \cdot \boldsymbol{f}^{(global)}(\boldsymbol{w}, \boldsymbol{y}_{1:M})$$

  where $y_{M+1} = \blacklozenge$ and $y_0 = \lozenge$ by construction.
- The best tagging sequence can be found efficiently with the Viterbi algorithm.
- As a discriminative model, Perceptron can handle an arbitrary number of features at eash position.

Parameter estimation for structured perceptron

- In the training phase, the sentences in the training set are decoded one a time (with the Viterbi Algorith
- If the highest scoring tagging sequence is not t correct tag se

$$\hat{y} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{w})} \theta \cdot \boldsymbol{f}($$

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} + \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{w}, \hat{\boldsymbol{y}})$$

The averaged perceptr[illegible]

procedure AVE_PERCEPTRON([illegible] , y [illegible])
    $t \leftarrow 0, \boldsymbol{\theta}^{(0)} \leftarrow \mathbf{0}, \boldsymbol{m} \leftarrow \mathbf{0}$
    repeat
        $t \leftarrow t + 1$
        Select a sequence i [illegible] line training
        $\hat{\boldsymbol{y}} \leftarrow$ [illegible] ecoding by Viterbi
        if $\hat{\boldsymbol{y}} \neq$ [illegible]
            $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)} + \boldsymbol{f}^{(global)}$ [illegible] $^{(i)}$ [illegible] $^{(i)}$ [illegible] $^{(global)}(\boldsymbol{w}^{(i)}, \hat{\boldsymbol{y}})$
        else [illegible] $\triangleright$ Fe[illegible] e sequence
            $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)}$
        $\boldsymbol{m} \leftarrow \boldsymbol{m} + \boldsymbol{\theta}^{(t)}$
    until tired
    $\bar{\boldsymbol{\theta}} = \frac{1}{t}\boldsymbol{m}$
    return $\bar{\boldsymbol{\theta}}$

Parameter Update example

- Correct tag sequence:

  - The_DT man_NN who_WP whistles_VBZ tunes_VBZ pianos_NNS

- Highest scoring sequence under the current
  - The_D _ _ _ es_NNS pianos_

- Which features need to be updated?

$$\theta_{(tunes,VBZ)} \leftarrow \theta_{(tunes,VBZ)} + 1 \quad , \quad \theta_{(tunes,NNS)} - 1$$

$$\theta_{(VBZ,VBZ)} \leftarrow \theta_{(VBZ,VBZ)} + 1 \quad \theta_{(VBZ,NNS)} \leftarrow \theta_{(VBZ,NNS)} - 1$$

$$\theta_{(VBZ,NNS)} \leftarrow \theta_{(VBZ,NNS)} + 1 \quad \theta_{(NNS,NNS)} \leftarrow \theta_{(NNS,NNS)} - 1$$

Stru es

Structured Support Vector Machines

- Classification with SVMs enforces a large-margin constraint that requires that there is a margin of at least 1 bet score of the correct label and all incorrect label this means

$$/ \quad y) \geq 1$$

- This can be minimally extended to seque

$$\forall \boldsymbol{y} \neq \boldsymbol{y}^{(i)}, \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{y}^{(i)}) - \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{y}) \geq 1$$

Extending SVMs to sequences

- A "structured" Support Vector Machine (SVM) outputs a structured object such as a sequence.

- When extending SVMs to sequence labelin to be addressed.

  - Some errors are more serious than others. T have the s ve differen . e woould e former

  - Having a fixed margin of 1 would sugges enumerate all possible sequences, which is infeasible as the number of sequences is exponential to the length of the sequence.
- The solution requires an adjustment of how the margin is computed.

Extending SVMs to sequences

- Instead of using a fix
reflects how the se

$$\forall \boldsymbol{y} \neq \boldsymbol{y}^{(i)}, \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}^{(i)}) - \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}) \geq c(\boldsymbol{y}, \boldsymbol{y}^{(i)})$$

- Next, instead of using a delta function $\delta(\boldsymbol{y}, \boldsymbol{y}^{(i)})$ to compute the error for the entire sequence, w **Hamming cost**, which counts the number of errors in $\boldsymbol{y}$:

- Instead of training against all labelings ve a margin that satisfies the above constraint, we focus on the prediction that **maximally** violates the margin constraint. We can identify this prediction by solving:

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y} \neq \boldsymbol{y}^{(i)}}{\operatorname{argmax}} \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}) - \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}^{(i)}) + c(\boldsymbol{y}, \boldsymbol{y}^{(i)})$$

$$= \underset{\boldsymbol{y} \neq \boldsymbol{y}^{(i)}}{\operatorname{argmax}} \boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}) + c(\boldsymbol{y}, \boldsymbol{y}^{(i)})$$

Extending SVMs to sequence labeling

- Reformulating t

$$\boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}^{(i)}) - \max_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{w})} \left(\boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{w}^{(i)}, \boldsymbol{y}) + c(\boldsymbol{y}, \boldsymbol{y}^{(i)}) \right) \geq 0$$

- This means that the constraint will be met (and be complete) $^{(i)}$ $^{(i)}$) is equal or greater than t l alternatives.
- In the training process, we identify predic (scores highly according to the model) an large cost according to the truth), and red these predictions by adjusting weights.
- Note that Hamming cost can be reduced to local parts by adding a feature $f_m(y_m) = \delta(y_m \neq y_m^{(i)})$, and can be incorporated into the Viterbi algorithm for purposes of the identifying the prediction to train against.

A comparison between Structured Perceptron and Structured SVM

- In the training process, the perceptron algori the prediction that has the highest score acco model to train against, while Structured SV the predictio can be identifi
- No cost needs to (or can be) computed duri both models
- In practice, with a large training set, the perceptron algorithm works pretty well, obviating the need for more complicated SVM models.