Organizing Web Information (CS 728) | **Midterm Exam**
Computer Science and Engineering | **2018-02-24 Saturday**
Indian Institute of Technology Bombay | **18:30–20:30 CC103**

NAME ______________________________ ROLL ____________________

This exam has 8 printed page/s. Write your name and roll number on **EVERY SIDE (and not just sheet)**, because we may take apart your answer book and/or xerox it for correction. Write your answer clearly within the spaces provided and on any last blank page. Do not write inside the rectangles to be used for grading. **If you need more space than is provided, you probably made a mistake in interpreting the question.** Start with rough work elsewhere, but you need not attach rough work. Use the marks alongside each question for time management. **Illogical or incoherent answers are worse than wrong answers or even *no* answer, and may fetch negative credit.** You may not use any computing or communication device during the exam. You may use textbooks, class notes written by you, approved material downloaded **prior to the exam** from the course Web page, course news group, or the Internet, or notes made available by me for xeroxing. If you use class notes from other student/s, you must obtain them **prior to the exam** and **write down his/her/their name/s and roll number/s** here.

1. We are designing a standard linear chain CRF. Input and prediction are sequences $\boldsymbol{x} = (x_1, \ldots, x_T)$ and $\boldsymbol{y} = (y_1, \ldots, y_T) \in \{1, \ldots, M\}^T$.

1.a Suppose, from domain knowledge, we know that no valid label sequence $\boldsymbol{y}$ can have both states a and b occur in it. Label sequences that have neither or one of them are allowed. Design (with explanation) a suitable dynamic programming table $V(t, m, k)$ for $k \in [\varnothing, a, b]$ and fill in the expressions to complete the table and indicate how to extract the best predicted sequence $\boldsymbol{y}$ subject to this constraint.

		5

$V(t, m, \varnothing)$ records paths that have neither a nor b in them. $V(t, m, a)$ records paths where a occurs but b does not. $V(t, m, b)$ records paths where b occurs but a does not. The optimal objective is

$$\max\Big\{\max_m V(T, m, \varnothing), \max_m V(T, m, a), \max_m V(T, m, b)\Big\}.$$

We initialize $V(0,\star,\varnothing)=0, V(0,\star,a)=-\infty, V(0,\star,b)=-\infty$, and, for $t>0$,

$$V(t,m,\varnothing)=\begin{cases}\max_{m'\in[M]} V(t-1,m',\varnothing)+w\cdot\varphi(x_t,m',m), & m\notin\{a,b\}\\ -\infty, & \text{otherwise}\end{cases}$$

$$V(t,a,a)=\max\begin{cases}\max_{m'\in[M]} V(t-1,m',a)+w\cdot\varphi(x_t,m',a)\\ \max_{m'\in[M]} V(t-1,m',\varnothing)+w\cdot\varphi(x_t,m',a)\end{cases}$$

$$V(t,m,a)=\max_{m'\in[M]} V(t-1,m',a)+w\cdot\varphi(x_t,m',m),\qquad m\notin\{a,b\}$$

$$V(t,b,a)=-\infty$$

$$V(t,b,b)=\max\begin{cases}\max_{m'\in[M]} V(t-1,m',b)+w\cdot\varphi(x_t,m',b)\\ \max_{m'\in[M]} V(t-1,m',\varnothing)+w\cdot\varphi(x_t,m',b)\end{cases}$$

$$V(t,m,b)=\max_{m'\in[M]} V(t-1,m',b)+w\cdot\varphi(x_t,m',m),\qquad m\notin\{a,b\}$$

$$V(t,a,b)=-\infty$$

1.b Suppose, from domain knowledge, we know that at least one of two designated states a and b has to appear at least once in a valid label sequence $\boldsymbol{y}$. Design a suitable dynamic programming table $V(t,m,k)$ for a suitable index space k (clearly define it) and fill in the expressions to complete the table and indicate how to extract the best predicted sequence $\boldsymbol{y}$ subject to this constraint.

		5

The dynamic programming table has two layers: $V(t,m,N)$ (seen neither a nor b up through time t) and $V(t,m,E)$ (seen either a or b or both at least once through time t).

$$V(0,\star,N)=0$$
$$V(\star,a,N)=-\infty$$
$$V(\star,b,N)=-\infty$$
$$V(t,m,N)=\max_{m'} V(t-1,m',N)+w\cdot\varphi(x_t,m',m)\qquad m\notin\{a,b\},t\in[1,T]$$
$$V(0,\star,E)=-\infty$$
$$V(t,m,E)=\max\begin{Bmatrix}\max_{m'} V(t-1,m',N)\\ \max_{m'} V(t-1,m',E)\end{Bmatrix}+w\cdot\varphi(x_t,m',m)\qquad m\in\{a,b\},t\in[1,T]$$
$$V(t,m,E)=\max_{m'} V(t-1,m',E)+w\cdot\varphi(x_t,m',m)\qquad m\notin\{a,b\},t\in[1,T]$$

And finally we pick $\max_m V(T,m,E)$.

2. We will develop a fine type formulation based on jointly embedding mention context and type labels, with a few embellishments. Let m be a mention, whose raw feature vector is collected from text as $\boldsymbol{x}_m\in\mathbb{R}^M$. Let $\boldsymbol{U}\in\mathbb{R}^{D\times M}$ project these raw feature vectors to $\mathbb{R}^D$.

Suppose there are K type labels. Type k is represented by a vector $\boldsymbol{y}_k \in \{0,1\}^K$ where $y_k(k) = 1$ and all $y_k(\neq k) = 0$. Type vectors $\boldsymbol{y}$ are projected by matrix $\boldsymbol{V} \in \mathbb{R}^{D\times K}$ into D-dimensional space.

2.a The score of type k, given mention m, is modeled as the dot product between the two projections. Write down the score and call it $f_m(k)$.

		1

$f_m(k) = (\boldsymbol{U}\boldsymbol{x}_m) \cdot (\boldsymbol{V}\boldsymbol{y}_k)$

2.b Given training instance (m_i, k_i), the hinge loss would usually be defined as $\ell_i(k) = \max\{0, 1 + f_{m_i}(k) - f_{m_i}(k_i)\}$. But typical type systems have redundant and overlapping types. If k is "very similar" to k_i in terms of the entities they contain, the above hinge loss may be unfair. Suppose E_k is the set of entities contained in type k. Suggest and justify a margin $\Delta_k(k')$ that addresses this problem. There may be many acceptable solutions.

		2

Here is one proposal:

$$w_{kk'} = \tfrac{1}{2}\left(\frac{|E_k \cap E_{k'}|}{|E_k|} + \frac{|E_k \cap E_{k'}|}{|E_{k'}|}\right)$$
$$\Delta_k(k') = \frac{1}{\spadesuit + w_{kk'}}$$

where $\spadesuit > 0$ is some tuned constant. If E_k and $E_{k'}$ overlap a lot, $\Delta_k(k')$ should be small, and vice versa.

2.c In reality more than one type labels may be active at a mention. Suppose training data comes in the form of certified present and absent types (m_i, K_i^+, K_i^-). Complete the following loss function for the ith instance:

$$\ell_i = \sum_{k\in K_i^+} \sum_{k'\in K_i^-} \underbrace{\qquad\qquad\qquad\qquad}_{\clubsuit}.$$

		1

$$\ell_i = \sum_{k\in K_i^+} \sum_{k'\in K_i^-} \max\{0, \Delta_k(k') + f_{m_i}(k') - f_{m_i}(k)\} \tag{1}$$

2.d Express the rank $R_{m_i}(k^*)$ of a correct type label k^* as a (possibly discontinuous) function of all the label scores $f_{m_i}(k) : k = 1, \ldots, K$. The label with largest score has rank 0. The loss can then be made rank-sensitive by writing

$$\ell_i = \sum_{k\in K_i^+} \sum_{k'\in K_i^-} \clubsuit\, R_{m_i}(k).$$

Approximate the rank using a smooth function that can facilitate learning by gradient descent.

		1

Actually, for every mention and every correct label k^*, we should be counting how many *incorrect* labels defeat it in terms of scores. We should not be comparing two correct labels. Let the sets of correct and incorrect labels be K_i^+, K_i^-.

$$R_{m_i}(k^*) = \sum_{k' \in K_i^-} \begin{cases} 1, & \Delta_{k^*}(k') + f_{m_i}(k') > f_{m_i}(k^*), \\ 0, & \text{otherwise} \end{cases}$$

The smooth version can be obtained via a sigmoid:

$$R_{m_i}(k^*) = \sum_{k' \in K_i^-} \sigma\big(\blacklozenge(\Delta_{k^*}(k') + f_{m_i}(k') - f_{m_i}(k^*)) + \heartsuit\big),$$

where $\blacklozenge > 0$ and $\heartsuit$ are hyperparameters. These sums-of-sigmoids are generally hard to train, though.

2.e We need to be robust to noisy training data. If a corpus is annotated with mentions m of entities e, and from the KG we know a set of types K_e to which e belongs, we should not use K_e as K^+ for all mentions of e. To tackle this problem, we divide mention instances into two kinds: *clean* mentions $\mathcal{M}_c$ where the KG connects e to exactly one path in the type hierarchy, and *noisy* mentions where more than one paths are connected to the entity, but only some of them may be active in a specific mention. We will write the overall loss objective as

$$\min_{\boldsymbol{U},\boldsymbol{V}} \left[\sum_{(m_i, K_i^+, K_i^-) \in \mathcal{M}_c} \boxed{\text{clean mention loss}} + \sum_{(m_i, K_i^+, K_i^-) \in \mathcal{M}_n} \boxed{\text{noisy mention loss}} \right]$$

For the clean mentions we will use the above loss function. For noisy mentions, we will insist only that the *largest* score from among K_i^+ should beat all scores from K_i^-. Write out in detail the noisy loss function.

		2

We need to change the symmetric sum-sum form in (1) to a sum-max form: For each bad type, *some* good type should beat it.

$$\ell_i = \sum_{k' \in K_i^-} \max \left\{ 0, f_{m_i}(k') - \max_{k^* \in K_i^+} \left[f_{m_i}(k^*) - \Delta_{k^*}(k') \right] \right\}$$

This is not the only reasonable proposal.

3. We are given a document with N entity mentions with contexts. Our goal is to infer entity labels $\boldsymbol{y} = y_1, \ldots, y_N$. In the star model, we choose

$$\hat{y}_i = \operatorname*{argmax}_{y_i} \left[\phi_i(y_i) + \sum_{j \neq i} \max_{y_j} \psi_{ij}(y_i, y_j) \right].$$

3.a Complete the following pseudocode to compute $\hat{\boldsymbol{y}}$.

for each $i = 1, \dots, N$ **do**
  $bestiLabel \leftarrow$ NULL, $bestiScore \leftarrow -\infty$
  for each possible label y_i **do**
    $yiScore \leftarrow$ ______
    for $j = 1, \dots, N$, $j \neq i$ **do**
      $bestjiSupport \leftarrow$ ______
      for each possible label y_j **do**
        if $bestjiSupport <$ ______ **then**
          $bestjiSupport \leftarrow$ ______
      $yiScore \leftarrow yiScore +$ ______
    if $bestiScore <$ ______ **then**
      $bestiScore \leftarrow$ ______
      $bestiLabel \leftarrow$ ______
  set $\hat{y}_i \leftarrow bestiLabel$
return $\hat{\boldsymbol{y}}$

		4

for each $i = 1, \dots, N$ **do**
  $bestiLabel \leftarrow$ NULL, $bestiScore \leftarrow -\infty$
  for each possible label y_i **do**
    $yiScore \leftarrow \phi_i(y_i)$
    for $j = 1, \dots, N$, $j \neq i$ **do**
      $bestjiSupport \leftarrow -\infty$
      for each possible label y_j **do**
        if $bestjiSupport < \psi_{ij}(y_i, y_j)$ **then**
          $bestjiSupport \leftarrow \psi_{ij}(y_i, y_j)$
      $yiScore \leftarrow yiScore + bestjiSupport$
    if $bestiScore < yiScore$ **then**
      $bestiScore \leftarrow yiScore$
      $bestiLabel \leftarrow y_i$
  set $\hat{y}_i \leftarrow bestiLabel$
return $\hat{\boldsymbol{y}}$

3.b A potential problem with this formulation is that there is no guarantee that $\hat{y}_j$ will be the best supporting label for $\hat{y}_i$. We will solve N separate problems, centered on mentions $i = 1, \dots, N$, to infer $\boldsymbol{y}^{(i)} = (y_1^{(i)}, \dots, y_N^{(i)})$, and try to make these solutions consistent with a single global solution $\boldsymbol{y}^{(0)}$. For simplicity assume each $y_i \in \{1, \dots, M\}$. We will change the signature of y_i from an integer in $[1, M]$ to a 1-hot vector $Y_i \in \{0,1\}^M$. Let $\phi_{im} = \phi_i(m)$ be the local potential of node i. What is the local score at node i as a function of $\boldsymbol{\phi}_i \in \mathbb{R}^M$ and Y_i?

		1

The local score (log node potential) at mention (node) i is $\boldsymbol{\phi}_i \cdot Y_i$.

3.c If ψ_{ij} is represented by a real $M \times M$ matrix, write down the potential for edge (i, j) with the nodes having 1-hot labels Y_i and Y_j.

		2

The log edge potential between nodes i and j is $Y_i^\top \psi_{ij} Y_j$.

3.d Relax Y_i from 1-hot to the unit simplex Δ_M in M dimensions, i.e., $Y_{im} \in \mathbb{R}$, $Y_{im} \geq 0$ and $\sum_m Y_{im} = 1$. Complete the objective for the local optimization for the ith problem:

$$\max_{Y_i \in \Delta_M} \left[\phi_i \underline{\hspace{3cm}} + \sum_{j \neq i} \max_{\underline{\hspace{1cm}} \in \Delta_M} \underline{\hspace{3cm}} \right]$$

What solver can you use to maximize this objective? (Extra credit: Is the optimization convex? If not, can you make it convex by introducing additional variables and constraints?)

		2

$$\max_{Y_i \in \Delta_M} \left[\phi_i \cdot Y_i + \sum_{j \neq i} \max_{Y_j \in \Delta_M} Y_i^\top \psi_{ij} Y_j \right]$$

The optimization is convex if all feasible regions are convex (which holds) and the objective to maximize is concave (which does not hold in general because of the $Y_i^\top \psi_{ij} Y_j$ terms). The standard technique to turn the slave optimizations convex (in fact, linear) is to introduce auxiliary variables $Z_{i,m;i',m'} \in \{0, 1\}$ (or in $[0, 1]$ when relaxed) such that $Y_{i,m} = \sum_{i',m'} Z_{i,m;i',m'}$ etc., and then write the objective as a linear function of Y and Z. This results in additional (in)equality constraints coupling Y, Z but the whole slave optimization can be solved by an LP solver.

3.e Following through with our plan above, we will solve N problems. In the ith problem, the relaxed label variables will be $\boldsymbol{Y}^{(i)} = (Y_1^{(i)}, \ldots, Y_N^{(i)}) \in \Delta_M^N$. We will then create a penalty if these solutions deviate from a global solution $\boldsymbol{Y}^{(0)} = (Y_1^{(0)}, \ldots, Y_N^{(0)}) \in \Delta_M^N$. Complete the following global objective:

$$\min_{\boldsymbol{\lambda} \in \Delta_N} \max_{\substack{\boldsymbol{Y}^{(0)} \in \Delta_M^N \\ \{\boldsymbol{Y}^{(i)} \in \Delta_M^N : i = 1, \ldots, N\}}} \sum_{i=1}^{N} \left(\underline{\hspace{4cm}} + \sum_{j \neq i} \underline{\hspace{5cm}} \right)$$

$$- \sum_{i=1}^{N} \lambda_i \, \| Y^{(i)} - \underline{\hspace{2.5cm}} \|_1.$$

Here $\|C\|_1$ is the L1 norm of C. (This may not be the way dual decomposition is usually set up. If you prefer you may set up the global objective in a different way, as long as it satisfies our goals.)

		1

$$\min_{\boldsymbol{\lambda}\in\Delta_N} \max_{\substack{\boldsymbol{Y}^{(0)}\in\Delta_M^N \\ \{\boldsymbol{Y}^{(i)}\in\Delta_M^N : i=1,\ldots,N\}}} \sum_{i=1}^{N} \left(\phi_i \cdot \underset{\sim}{Y_i^{(i)}} + \sum_{j\neq i} \underset{\sim\sim\sim}{\max_{Y_j^{(i)}\in\Delta_M} Y_i^{(i)\top} \psi_{ij} Y_j^{(i)}} \right) - \clubsuit \sum_{i=1}^{N} \lambda_i \, \|\boldsymbol{Y}^{(i)} - \underset{\sim}{\boldsymbol{Y}^{(0)}}\|_1.$$

Note that the penalty has to be *subtracted*, and it might help to multiply by some balancing factor ♣.

3.f Given the high-level pseudocode below:

choose initial $\boldsymbol{\lambda}, \boldsymbol{Y}^{(0)}$
for some number of iterations **do**
  for each problem indexed i **do**
    fix current $\boldsymbol{\lambda}, \boldsymbol{Y}^{(0)}$ and solve for next $\boldsymbol{Y}^{(i)}$
  update $\boldsymbol{Y}^{(0)}$
  update $\boldsymbol{\lambda}$

write down how you would use standard optimizers (or simple update expressions) to solve the three key steps.

		3

Solving for $\boldsymbol{Y}^{(i)}$: In the pseudocode for choosing $\hat{\boldsymbol{y}}$, we could search through all possible y_i, and let the current choice of y_i force the best possible discrete y_js. It is not necessarily optimal to follow a similar recipe, but it may behave reasonably in practice:

- First optimize $\operatorname{argmax}_{Y_i^{(i)}\in\Delta_M} \phi_i Y_i^{(i)} - \lambda_i \|Y_i^{(i)} - Y_i^{(0)}\|_1$ using an LP solver. Note that λ_i and $Y_i^{(0)}$ are frozen at the moment.
- Now for each $j\neq i$, choose $\operatorname{argmax}_{Y_j^{(i)}\in\Delta_M} \boxed{Y_i^{(i)\top}\psi_{ij}} Y_j^{(i)}$. Now that $Y_i^{(i)}$ is also fixed momentarily, the boxed part is a constant matrix. So this problem is also easily solved via LP.

Projected gradient descent may also work.

Updating $\boldsymbol{Y}^{(0)}$: This is just finding a weighted medoid, easily solved via LP.

Updating $\boldsymbol{\lambda}$: At this point we have constants $a_i = \|\boldsymbol{Y}^{(i)} - \underset{\sim}{\boldsymbol{Y}^{(0)}}\|_1 \geq 0$, and must find $\operatorname{argmin}_{\boldsymbol{\lambda}\in\Delta_N} \sum_i a_i \lambda_i$. Naturally we would set $\lambda_{i^*} = 1$ for $i^* = \operatorname{argmin}_i a_i$ and $\lambda_i = 0$ for all $i \neq i^*$. In other words, every outer iteration, $\boldsymbol{\lambda}$ would get set to a 1-hot vector, which may not be a good thing. Encouraging some entropy to $\boldsymbol{\lambda}$ may help stability in practice.

Dual decomposition differs from the above in various ways. We would explicitly introduce decision variables Z_{ij} for every edge, leading to a linear objective $\psi_{ij} \odot Z_{ij}$. Z and Y would be coupled by marginal inequalities. Also, we would use $\boldsymbol{\lambda} \in \mathbb{R}^{N\times N\times M}$, not take norm errors, but instead write something like

$\sum_{n=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\lambda_{nij}(Y_{ij}^{(n)} - Y_{ij}^{(0)})$. But this would let us solve all local problems exactly via LP.

Total: 30