Organizing Web Information (CS 728) **Final exam**
Computer Science and Engineering **2017-04-19 Wednesday**
Indian Institute of Technology Bombay **09:30–12:30 SIC201**

NAME _____ ROLL _____

This exam has 12 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. Consider a learning problem where the system has to fit a model for $\Pr(y|x)$. In the classic conditional model, given labeled instance (x_i, y_i), the goal is to maximize $\Pr(y_i|x_i; w)$ over model parameters w. However, this does not directly handle general cost functions for misclassification, e.g., $\Delta(y, y') \in \mathbb{R}_+$.

1.a One common approach to incorporate Δ while training is to minimize the expected loss over all training instances (complete):

$$\operatorname*{argmin}_w \sum_i \sum_y \Delta(\_\_\_\_, \_\_\_\_) \Pr(\_\_\_\_|\_\_\_\_; w)$$

		1

$$\operatorname*{argmin}_w \sum_i \sum_y \Delta(y, y_i) \Pr(y|x; w)$$

No plausible partial credit; in particular, $\sum_i \sum_y \Delta(y, y_i) \Pr(y_i|x; w)$ is very wrong.

1.b If w^* is the result of the above training optimization, and a test instance x provided, the test/inference objective consistent with the above training procedure is (complete)

$$\operatorname{argmax} \_\_\_\_\_$$

Does cost Δ play a role during the above inference? (Obviously, the gold label is not known to the system for test instances.)

		2

Inference amounts to $\text{argmax}_y \Pr(y|x; w^*)$ as in the case without costs. Δ has served its purpose in the training objective and does not have a role during inference.

1.c In the maxent framework to fit $\Pr(y|x)$, a feature map $f(x, y) \in \mathbb{R}^K$ is designed; $f_k(x, y)$ is the k-th feature. For a specific instance (x_i, y_i), let $A_i = f(x_i, y_i)$. Then the traditional maximum entropy constraints are:

$$\sum_y f(x_i, y) \Pr(y|x) = A_i.$$

Because $\sum_y \Pr(y|x_i) = 1$, we can write the above as

$$\sum_y \Pr(y|x_i) \Big[\underbrace{f(x_i, y) - f(x_i, y_i)}_{\text{discrepancy}}\Big] = 0.$$

Now consider cost-sensitive classification, given Δ as above. One way to involve misclassification costs into the maxent formulation is to scale the discrepancy with the cost:

$$\sum_y \Pr(y|x_i)\, \Delta(y, y_i) \Big[\underbrace{f(x_i, y) - f(x_i, y_i)}_{\text{discrepancy}}\Big] = 0. \quad (1)$$

I.e., we are more unhappy with the discrepancy if the y has large cost. The maxent goal is to maximize the entropy

$$H(\Pr(y|x)) = -\sum_y \Pr(y|x) \log \Pr(y|x)$$

subject to the above constraints. Using $w \in \mathbb{R}^K$ as the dual variables corresponding to the constraints (1) and a standard Lagrangian analysis, complete the dual objective, showing key steps:

$$\min_{w \in \mathbb{R}^K} \log \sum_y \exp \Big[\underbrace{\qquad\qquad}_{} w \cdot \Big(\underbrace{\qquad\qquad}_{} - \underbrace{\qquad\qquad}_{}\Big)\Big] \quad (2)$$

		4

The primal objective $f(\Pr(y|x_i))$ is to maximize $H(\Pr(y|x_i))$ subject to conditions (1) and that $\Pr(y|x_i) \geq 0$ and $\sum_y \Pr(y|x_i) = 1$ for all x. Let the corresponding dual variables be w and λ and write the Lagrangian objective as

$$L(\Pr(y|x_i); \lambda, w) = H(\Pr(y|x_i)) + \lambda \left[\sum_y \Pr(y|x_i) - 1\right] + w \cdot \sum_y \Pr(y|x_i) \Delta(y, y_i) \big(f(x_i, y) - f(x_i, y_i)\big)$$

It turns out we need not bother about λ, and, differentiating wrt $\Pr(y|x_i)$ above, we can write:

$$\Pr^*(y|x_i) = \frac{1}{Z_i} \exp\Big[\Delta(y, y_i)\ w \cdot \big(f(x_i, y) - f(x_i, y_i)\big)\Big]$$

(Full credit for this step, although, technically, we need to verify this gives supremum.) With λ now out of the way, we can write, using weak duality,

$$\begin{aligned}
f^*(\Pr(y|x_i)) &\leq \sup_{\Pr(y|x_i)} L(\Pr(y|x_i), w), \quad \text{(which we seek to minimize)} \\
L(\Pr^*(y|x_i), w) &= -\sum_y \Pr^*(y|x_i) \underline{\log \Pr^*(y|x_i)} \\
&= -\sum_y \Pr^*(y|x_i) \Big[\Delta(y, y_i)\ w \cdot \big(f(x_i, y) - f(x_i, y_i)\big) - \log Z_i\Big] \\
&= -w \cdot \sum_y \Pr^*(y|x_i) \Delta(y, y_i) \big(f(x_i, y) - f(x_i, y_i)\big) \\
&\quad + (\log Z_i) \sum_y \Pr^*(y|x_i) = -w \cdot \vec{0} + \log Z_i = \log Z_i
\end{aligned}$$

Given we wanted to maximize the primal objective, we should minimize the dual objective, which leads us to complete (2) as

$$\min_w \log \sum_y \exp\Big[\underbrace{\Delta(y, y_i)}\ w \cdot \big(\underbrace{f(x_i, y)} - \underbrace{f(x_i, y_i)}\big)\Big] = \min_w \log Z_i.$$

1.d To train w using gradient descent on the above objective, first find the gradient of (2), and show that the gradient is the expectation of $\Delta(y', y_i)\big(f(x_i, y') - f(x_i, y_i)\big)$ where y' follows the distribution $Q_3(y)$ given by (complete)

$$Q_3(y) \propto \exp\Big[\underbrace{\qquad\qquad} w \cdot \underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}\Big].$$

		4

The gradient of the dual objective is

$$\begin{aligned}
\frac{1}{Z_i}\frac{dZ_i}{dw} &= \sum_{y'} \Delta(y', y_i)\big(f(x_i, y') - f(x_i, y_i)\big) \frac{\exp\Big[\Delta(y', y_i)\ w \cdot \big(f(x_i, y') - f(x_i, y_i)\big)\Big]}{Z_i} \\
&= \sum_{y'} \Delta(y', y_i)\big(f(x_i, y') - f(x_i, y_i)\big) Q_3(y'), \quad \text{where} \\
Q_3(y) &\propto \exp\Big[\underbrace{\Delta(y, y_i)}\ w \cdot \big(\underbrace{f(x_i, y) - f(x_i, y_i)}\big)\Big]
\end{aligned}$$

1.e After w is trained, to be consistent with the above constraints, given a test instance x, we should infer its label as (complete and explain briefly):

$$\operatorname*{argmin}_{\hat{y}} \sum_y \exp\Big[\Delta(\hat{y}, y) \underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}\Big]$$

		2

We want to find $\hat{y}$ which, if it was the correct label, minimizes the discrepancy with all ys:

$$\operatorname*{argmin}_{\hat{y}} \sum_{y} \exp\Big[\Delta(\hat{y}, y) w \cdot \big(f(x_i, y) - f(x_i, \hat{y})\big)\Big]$$

2. It seems suboptimal to invest the same D dimensions for all words in `word2vec` or `GloVe`. Fixed size vectors cannot represent diverse semantic complexities of different words. *Race* is a richly polysemous word, whereas *soccer* has a specific invariant meaning. Given focus word i and context word k, we define their cooccurrence probability as

$$\Pr(i, k, m) = \frac{1}{Z} \exp\big(-E(w_i, c_k, m)\big), \tag{3}$$

where $w_i, c_k \in \mathbb{R}^\infty$ and $m \geq 1$ is a random integer representing the largest index up to which the dot product between w_i and c_k should be computed, ignoring the rest:

$$E(w_i, c_k, m) = \underbrace{m \log a}_{①} - \sum_{j=1}^{m} \Big[w_{i,j} c_{k,j} \underbrace{- \lambda w_{i,j}^2 - \lambda c_{k,j}^2}_{②} \Big],$$

with $a > 1$ and $\lambda > 0$.

2.a Justify informally why devices ① and ② make $\Pr(i, k, m)$ a reasonable distribution. Specifically, argue why, with these devices in place, and all w_i, c_k initialized to a finite number of nonzero elements, at termination, there will be a finite M such that $w_{i,m}, c_{k,m} = 0$ for all $m > M$.

		2

The $m \log a$ part in the exponent makes $\Pr(\cdot, \cdot, m)$ decrease as a^{-m} so that m has finite mean. In each gradient update step, the optimizer can write only a finite number of elements in w_i and c_k. Therefore, if the optimizer terminates in a finite number of steps, w_i and c_k cannot have nonzeros beyond a finite index M.

2.b In (3), Z is the partition function, given by

$$Z = \sum_{i,k,m} \exp\big(-E(w_i, c_k, m)\big).$$

The sum over i, k is finite with the number of terms being the square of the vocabulary size. But the sum over m is apparently infinite. Fix some i, k and consider the potentially problematic infinite sum $\sum_{1 \leq m} \exp\big(-E(w, c, m)\big)$. By splitting this into

$$\sum_{m=1}^{M} e^{-E(w,c,m)} + \sum_{m>M} e^{-E(w,c,m)},$$

complete and prove that

$$\sum_{1<m} \exp\big(-E(w,c,m)\big) = \sum_{m=1}^{M} e^{-E(w,c,m)} + \underbrace{\qquad a \qquad}\ \exp\big(-E(w,c,\ \underbrace{\qquad\qquad}\ )\big),$$

where the last blank should be filled with either M or $M+1$. (Show all key steps.)

		3

$$\sum_{1\le m} e^{-E(w,c,m)} = \sum_{m=1}^{M} e^{-E(w,c,m)} + \sum_{m>M} e^{-E(w,c,m)}.$$

Focusing on the second term,

$$\sum_{m>M} e^{-E(w,c,m)} = e^{-E(w,c,M+1)} \sum_{m>M} \exp\Big[-(m-M-1)\log a + \underbrace{\sum_{j=M+1}^{m} w_j c_j - w_j^2 - c_j^2}_{=0}\Big].$$

Choose M large enough so that the sum inside is zero, so that the above

$$= e^{-E(w,c,M+1)} \sum_{m>M} \exp\big[-(m-M-1)\log a\big],$$

which, changing variable to $n = m - M - 1$,

$$= e^{-E(w,c,M+1)} \sum_{n\ge 0} a^{-n} = \frac{a}{a-1} e^{-E(w,c,M)}.$$

This shows that if we can bound M, we can compute the partition function efficiently.

2.c We will express the log-likelihood objective (to maximize) of skip-gram as marginalized over all possible m:

$$\begin{aligned} \log \Pr(k|i) &= \log \sum_{m\ge 1} \Pr(k,m|i) = \log \sum_{m\ge 1} \Pr(k|i,m)\Pr(m|i) \\ &\ge \sum_{m\ge 1} \Pr(m|i)\ \log \Pr(k|i,m) \end{aligned} \tag{4}$$

Why does the inequality in the last line hold?

		1

By the curvature of the log function, i.e., Jensen's inequality, if $\{p_m : m \ge 1\}$ is a distribution with $\sum_m p_m = 1$, and $\{a_m : m \ge 1\}$ is any real sequence, then $\log \sum_m p_m a_m \ge \sum_m p_m \log a_m$. Therefore, to maximize the learning objective, we can instead maximize the lower bound on the rhs.

2.d To compute the last expression (4), we will need to estimate $\Pr(m|i)$. Sketch how we can compute $\Pr(m|i,k)$ and therefrom, $\Pr(m|i)$.

		2

By Bayes rule, $\Pr(m|i,k) = \dfrac{\Pr(i,k,m)}{\sum_{m'} \Pr(i,k,m')} = \dfrac{e^{-E(i,k,m)}}{\sum_{m'} e^{-E(i,k,m')}}$

We already know how to compute the denominator in a finite number of steps.

2.e Sketch visually how you would expect $\Pr(m|i)$ (a histogram against m) to look for word i equal to *break* and *violent*.

		2

The word *break* is highly polysemous. We expect the support of the $\Pr(m|\textit{break})$ distribution to spread to large m, and perhaps have multiple maxima in this range. The word *violent* is essentially unambiguous, so we expect its support to be much smaller.

2.f Suppose variable-dimensional vectors are trained for each word. Recall that `word2vec` retains only w_i, and computes word-pair similarity as $\text{sim}(i,j) = \cos(w_i, w_j)$. Why is that a bad idea with variable-dimensional embeddings?

		1

The whole point of variable dimensionality embeddings is that semantically richer words will get more dimensions and therefore generally larger norms. Cosine destroys the information in the length of the vector (and therefore, likely some of the information in m too).

2.g Propose a better expression for $\text{sim}(i,j)$ that marginalizes over the uncertainty of m.

		2

$$\text{sim}(i,j) = \sum_{M \geq 1} \Pr(M|i,j) \left[\sum_{1 \leq m \leq M} w_{im} w_{jm} \right]$$

3. In TransE, the measure of disbelief in triple (s,r,o) where s,o are subject and object entities and r is the relation, is $d(s,r,o) = \|e_s + v_r - e_o\|_2^2$. Suppose the optimizer always ensures $\|e_s\| = \|e_o\| = 1$.

3.a Complete the following expression for $d(s,r,o)$ via basic algebraic manipulation:

$$d(s,r,o) = \|e_s\|_2^2 + \|e_o\|_2^2 + \|v_r\|_2^2 - 2\big(e_s \cdot \underbrace{\quad} + v_r \cdot (\underbrace{\quad} - \underbrace{\quad})\big).$$

		3

$$d(s,r,o) = \|e_s\|_2^2 + \|e_o\|_2^2 + \|v_r\|_2^2 - 2\big(e_s \cdot \underset{\sim}{e_o} + v_r \cdot (\underset{\sim}{e_o} - \underset{\sim}{e_s})\big).$$

3.b If (s,r,o) is a known valid triple and (s',r,o') is an invalid, suitably perturbed triple, such that the current embeddings do not satisfy the constraint $d(s,r,o) + \text{margin} \leq d(s',r,o')$, the learning algorithm should try to _____ (increase/decrease) $d(s,r,o)$ and _____ (decrease/increase) $d(s',r,o')$ (complete). | | | 2 |

Decrease, increase.

3.c In trying to do so, to e_s will be added a vector (obtained from the gradient of a suitable objective) times some stepsize. Write down this vector. (Hint: it is a simple linear combination of two of the vectors above.) | | | 2 |

$\partial d(s,r,o)/\partial e_s \propto e_s - (e_o - v_r)$. Therefore, the update to e_s will look like

$$e_s \leftarrow e_s - \eta\big(e_s - (e_o - v_r)\big) = (1-\eta)e_s + \eta(e_o - v_r)$$

(followed by scaling back to unit norm). Here η is a stepsize. In words, we scale e_s slightly, and add a vector in the direction of $e_o - v_r$, because ideally the model wants $e_s + v_r$ to be e_o.

3.d Similarly, to e_o will be added a slightly different vector. Write it down, ignoring constants and stepsize. (Hint: it is a simple linear combination of two of the vectors above.) | | | 2 |

$\partial d(s,r,o)/\partial e_o \propto e_o - (e_s + v_r)$. Therefore, the update to e_o will look like

$$e_o \leftarrow e_o - \eta\big(e_o - (e_s + v_r)\big) = (1-\eta)e_o + \eta(e_s + v_r)$$

(followed by scaling back to unit norm). Here η is a step size. In words, we scale e_o slightly, then add a vector in the direction of $e_s + v_r$, because ideally the model wants e_o to equal $e_s + v_r$.

3.e Can TransE support symmetric relations (e.g., sibling-of)? | | | 1 |

Wanting $e_s + v_r = e_o$ and $e_o + v_r = e_s$ leads to $v_r = \vec{0}$ which isn't useful. Therefore, no, TransE is not expected to be good at modeling symmetric relations that are not reflexive.

4. Suppose a knowledge graph (KG) has entity set $\mathcal{E}$ and R relations. Unless otherwise specified, we will assume real vectors are in $\mathbb{R}^D$, and real matrices are $D \times D$, some of which may be diagonal.

4.a TransE models the evidence score of tuple (s, r, o) as inversely related to $\|e_s + v_r - e_o\|$. (In this part there is no matrix associated with each relation, only a vector v_r.) Globally how many real scalar parameters (i.e. total number of elements in all vectors) have to be trained by TransE?

		1

$D\,|\mathcal{E}| + D\,R$

4.b TransR projects each entity with a relation-specific matrix before checking for translation. Its (inverse) evidence score is $\|M_r e_s + v_r - M_r e_o\|$. What is the number of parameters trained by TransR?

		1

$\underbrace{D^2R + DR}_{\text{relations}} + \underbrace{D|\mathcal{E}|}_{\text{entities}}$

4.c A slightly different approach "ProjE" is to

- Project e_s and v_r first using two global diagonal matrices A, B respectively, add a global vector $b_1 \in \mathbb{R}^D$, and apply a nonlinearity σ_1.
- Dot-product with the candidate e_o, add a global scalar offset b_2, and apply a nonlinearity σ_2 to get the probability $\Pr(s, r, o)$ that tuple (s, r, o) is valid.

Write down the expression for $\Pr(s, r, o)$.

		2

$\Pr(s, r, o) = \sigma_2\big(e_o \cdot \sigma_1(Ae_s + Bv_r + b_1) + b_2\big)$

4.d From among standard options, choose and justify the form of nonlinearity you will use for σ_1 and σ_2.

		2

σ_2 has to output a probability estimate so it should be a sigmoid. σ_1 can be tanh or sigmoid or ReLU; b_2 will shift as needed.

4.e Globally, how many parameters have to be trained in ProjE?

		1

$\underbrace{D + D + D + 1}_{\text{global}} + \underbrace{D\,R}_{\text{relations}} + \underbrace{D\,|\mathcal{E}|}_{\text{entities}}$

4.f Can ProjE support learning asymmetric relations (e.g., Twitter-follows), where $(s, r, o) \nRightarrow (o, r, s)$?

		1

Yes, given e_s, e_o are involved in asymmetric positions in the expression for $\Pr(s, r, o)$, it is certainly possible to fit embeddings such that $\Pr(s, r, o) \neq \Pr(o, r, s)$. Whether these can be fitted consistently with many other relations involving the same entities is hard to say in general.

4.g Extra credit (answer on separate sheet): Can ProjE support learning antisymmetric relations (e.g., parent-of), where $(s, r, o) \Rightarrow \neg(o, r, s)$?

5. Consider KG-driven embeddings for entities and relations toward knowledge base completion. Embedding vectors are in $\mathbb{R}^n$, with elements indexed $0, \ldots, n-1$. The *circular convolution* of embeddings $x, y \in \mathbb{R}^n$ is defined as

$$(x * y)[j] = \sum_{0 \le k < n} x[(j-k) \bmod n]\; y[k]$$

whereas the *circular correlation* is defined as

$$(x \star y)[j] = \sum_{0 \le k < n} x[(j+k) \bmod n]\; y[k]$$

Note the different symbols used, $*$ and $\star$ and the corresponding different indices used to access x.

5.a Is $x * y = y * x$? Prove or provide a counterexample. | | | 2 |

	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Rows (columns) are indices into x (y). The value of $(x * y)[j]$ is the sum of all cells numbered j. Since this pattern is symmetric around the diagonal, $x * y = y * x$.

5.b Is $x \star y = y \star x$? Prove or provide a counterexample. | | | 2 |

This time the pattern is not symmetric:

	0	1	2	3
0	0	3	2	1
1	1	0	3	2
2	2	1	0	3
3	3	2	1	0

so in general $x \star y \neq y \star x$.

5.c If $\text{flip}(x) = (x_{n-1}, \ldots, x_0)^\top$, then $x \;\_\_\_\; y = \;\_\_\_\; \text{flip}(x) \;\_\_\_\; y$ (complete with the two operators and a short proof). | | | 2 |

$x \mathbin{\underset{\sim}{\star}} y = \text{flip}(\text{flip}(x) \mathbin{\underset{\sim}{*}} y)$. The index pattern for $\text{flip}(x) * y$ is given by flipping the rows of the table for $x * y$, after which we flip the contents of the table:

	0	1	2	3
0	3	0	1	2
1	2	3	0	1
2	1	2	3	0
3	0	1	2	3

$\longrightarrow$

	0	1	2	3
0	0	3	2	1
1	1	0	3	2
2	2	1	0	3
3	3	2	1	0

5.d For vectors $a, b \in \mathbb{R}^n$, what is $a \star (a * b)$?

		1

This is a bit tedious; for $n = 3$ the expression turns out to be:

$$\begin{bmatrix} b_0(a_0^2 + a_1^2 + a_2^2) + b_1a_0a_2 + b_2a_0a_1 + b_1a_0a_1 \\ \qquad\qquad + b_2a_1a_2 + b_1a_1a_2 + b_1a_0a_2 \\ b_1(a_0^2 + a_1^2 + a_2^2) + b_0a_0a_1 + b_2a_0a_2 + b_0a_0a_2 \\ \qquad\qquad + b_2a_1a_2 + b_0a_1a_2 + b_1a_0a_1 \\ b_2(a_0^2 + a_1^2 + a_2^2) + b_0a_0a_1 + b_1a_1a_2 + b_0a_1a_2 \\ \qquad\qquad + b_1a_0a_2 + b_0a_0a_2 + b_1a_0a_1 \end{bmatrix}$$

The above expression will get full credit, although no particular insight can be drawn from it, unless a is sampled in a specific way. E.g., each a_i, b_i can be drawn from a normal distribution $\mathcal{N}(0, 1/n)$. Then the above vector will look like

$$\begin{bmatrix} b_0(1 + \xi) + \eta_0 \\ b_1(1 + \xi) + \eta_1 \\ b_2(1 + \xi) + \eta_2 \end{bmatrix} \approx (1 + \xi)b + \eta,$$

where ξ, η are zero-mean noise terms. In other words, b will be approximately recovered. Now in reality a will be fitted by backprop, not sampled, but the central limit theorem means if n is large enough (as against 3 above), approximate recovery of b should still happen.

5.e In *holographic embeddings*, each entity t and relation r are associated with vectors $e_t \in \mathbb{R}^n$ and $w_r \in \mathbb{R}^n$. The score supporting tuple (s, r, o) is given by:

$$f_H(s, r, o) = w_r \cdot (e_s \star e_o)$$

Here $\cdot$ is the standard dot product. What is a potential benefit of holographic embeddings over DistMult, where

$$f_D(s, r, o) = w_r \cdot (e_s \odot e_o),$$

where $\odot$ is elementwise product? (Hint: compare $f_D(s, r, o)$ with $f_D(o, r, s)$.)

		1

Because $\star$ is not symmetric, holographic embeddings can support asymmetric relations, unlike DistMult.

5.f Let (dotless) $\imath = \sqrt{-1}$ be the imaginary unit and $\mathbb{C}$ be complex numbers. The complex conjugate of $z = a + \imath b$ is denoted $\bar{z} = a - \imath b$. Conjugation applies elementwise to vectors and matrices. The complex dot product between $c, d \in \mathbb{C}^n$ is

defined as $c \cdot d = \sum_i \bar{c}_i d_i$. Is $c \cdot d = d \cdot c$ in general? Provide proof or counterexample.

		1

Let $n = 1$. For two complex numbers a and b, is $\bar{a}b = \bar{b}a$ always? Let $a = 1 + 2\boldsymbol{\imath}$ and $b = 2 - 1\boldsymbol{\imath}$.

5.g For $c, d \in \mathbb{C}^n$, compare $c \cdot d$ and $\overline{d \cdot c}$.

		1

$\overline{d \cdot c} = \overline{\sum_i \bar{d}_i c_i} = \sum_i \overline{\bar{d}_i c_i} = \sum_i \bar{c}_i d_i = c \cdot d$

5.h In *complex embeddings*, entity and relation vectors are in $\mathbb{C}^n$ rather than $\mathbb{R}^n$. We define the score of tuple (s, r, o) as (complex multiplication is associative):

$$f_C(s, r, o) = \Re\left[\sum_{0 \le j < n} w_r[j]\ e_s[j]\ \overline{e_o[j]}\right],$$

where each term in the sum multiplies three complex numbers, and after the sum we retain only the real part. Can complex embeddings support learning antisymmetric relations (like parent-of), i.e., $(s, r, o) \implies \neg(o, r, s)$? If so, how?

		2

We are looking for a form of (a complex vector) w_r such that if $f_C(s, r, o)$ is large then $f_C(o, r, s)$ is small, and vice versa. Let $n = 1$, i.e., consider s, r, o each as modeled by a single complex number. Then $f_C(s, r, o) = w_r e_s \bar{e}_o$. For notational simplicity, shorthand this as

$$\begin{aligned}
\Re(rs\bar{o}) &= \Re\big((r_1 + r_2\boldsymbol{\imath})(s_1 + s_2\boldsymbol{\imath})(o_1 - o_2\boldsymbol{\imath})\big) \\
&= \Re\Big((r_1 + r_2\boldsymbol{\imath})\big((s_1 o_1 + s_2 o_2) + (s_2 o_1 - s_1 o_2)\boldsymbol{\imath}\big)\Big) \\
&= r_1 s_1 o_1 + r_1 s_2 o_2 - r_2 s_2 o_1 + r_2 s_1 o_2
\end{aligned}$$

If $r_1 = 0, r_2 = 1$, then $\Re(rs\bar{o}) = s_1 o_2 - s_2 o_1$. Conversely, $\Re(ro\bar{s}) = o_1 s_2 - o_2 s_1 = -\Re(rs\bar{o})$, thus fulfilling the antisymmetry requirement. Conversely, if $r_1 = 1, r_2 = 0$, then $\Re(rs\bar{o}) = s_1 o_1 + s_2 o_2 = \Re(ro\bar{s})$. Thus, by using complex arithmetic, r can represent both symmetric and antisymmetric relations.

5.i Complete the following derivation with brief explanation for each step:

$$\begin{aligned}
\sum_{0 \le j < n} w_r[j]\ e_s[j]\ \overline{e_o[j]} &= \overline{\Big(\underbrace{\quad} \odot \overline{\underbrace{\quad}}\Big)} \cdot \underbrace{\quad} \\
&= \overline{\underbrace{\quad} \cdot \Big(\overline{\underbrace{\quad}} \odot \underbrace{\quad}\Big)}
\end{aligned}$$

and thus we can write

$$f_C(s, r, o) = \Re\left(w_r \cdot (\overline{\underbrace{\quad}} \odot \underbrace{\quad})\right).$$

Can you see any significant similarity with an earlier formula?

		4

$$\sum_{0\le j<n} w_r[j]\; e_s[j]\; \overline{e_o[j]} = (e_s \odot \overline{e_o})^\top w_r \qquad \text{by definition}$$

$$= \overline{\left(\underset{\sim}{e_s} \odot \underset{\sim}{\overline{e_o}}\right)} \cdot \underset{\sim}{w_r} \qquad \because a^\top b = \bar{a} \cdot b$$

$$= \overline{\underset{\sim}{w_r} \cdot \overline{\left(\underset{\sim}{e_s} \odot \underset{\sim}{\overline{e_o}}\right)}} \qquad \because a \cdot b = \overline{b \cdot a}$$

$$= \overline{\underset{\sim}{w_r} \cdot \left(\underset{\sim}{\overline{e_s}} \odot \underset{\sim}{e_o}\right)} \qquad \because \overline{a\bar{b}} = \bar{a}b$$

$$\therefore \quad f_C(s,r,o) = \Re\left(\overline{\underset{\sim}{w_r} \cdot \left(\underset{\sim}{\overline{e_s}} \odot \underset{\sim}{e_o}\right)}\right) = \Re\left(w_r \cdot (\underset{\sim}{\overline{e_s}} \odot \underset{\sim}{e_o})\right) \qquad \because \Re(\bar{a}) = \Re(a).$$

$$\text{Recall} \quad f_H(s,r,o) = w_r \cdot (e_s \star e_o) = w_r \cdot (\text{flip}(e_s) * e_o).$$

The similarity is highlighted.

5.j Extra credit (answer on separate sheet): If you know your way around discrete Fourier transforms and complex algebra, you can formalize your answer above.

Total: 60