

This set is due 2:30pm, March 1<sup>st</sup>, via Moodle. Note that you get two weeks for this set. You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus.

## 1 SVD and PCA

**Question A:** Let  $X$  be an  $N \times d$  matrix and  $Y = X^T$ . If the SVD of  $Y = U\Sigma V^T$ , then show that the columns of  $V$  are the PCA of  $X$ .

**Question B:** Prove that the SVD is the

i. Best rank at-most- $k$  approximation in the Frobenius norm. That is, for  $A = U\Sigma V^T$ , show  $A_k = \sum_{j=1}^k \sigma_j u_j v_j^T$  satisfies

$$A_k = \arg \min_{\text{rank}(B) \leq k} \|A - B\|_F$$

ii. The complexity equivalent to the trace norm. That is,

$$\|Y\|_* = \min_{Y=AB^T} \frac{1}{2} (\|A\|_F^2 + \|B\|_F^2)$$

## 2 Matrix Factorization

In the setting of collaborative filtering, we derive the coefficients of the matrices  $U_{K \times M}$  and  $V_{K \times N}$  by minimizing the regularized square error:

$$\arg \min_{U,V} \frac{\lambda}{2} (\|U\|_{\text{Fro}}^2 + \|V\|_{\text{Fro}}^2) + \sum_{i,j} (y_{ij} - u_i^T v_j)^2$$

where  $u_i$  and  $v_j$  are the  $i^{\text{th}}$  and  $j^{\text{th}}$  column of  $U$  and  $V$ , respectively. Then  $Y_{M \times N} \approx U^T V$ .

**Question A:** Specify  $\partial_{u_i}$  and  $\partial_{v_j}$  for stochastic gradient descent formula below, where  $\eta$  is the learning rate:

$$\begin{aligned} u_i &= u_i - \eta \partial_{u_i} \\ v_j &= v_j - \eta \partial_{v_j} \end{aligned}$$

**Question B:** Another method to minimize the regularized squared error is alternating least squares (ALS). ALS solves the problem by fixing one of the  $U$  and  $V$ , and solving the optimal condition for the other. Then keep rotating this process until it converges. Derive the closed form for the optimal conditions of  $u_i$  and  $v_j$ .

**Question C:** Implement matrix factorization with stochastic gradient descent, using your answer from part A. Assume your input data is in the form of three vectors: a vector of  $is$ ,  $js$ , and  $y_{ij}$ s.

**Question D:** We have posted the MovieLens data set on Moodle. Download both data sets (training and testing), and train your model on the training data. The format of the data is user, movie, rating, where rating is the rating that the user gave the movie. Set  $\lambda = 0$  (in other words, do not regularize), and vary the

number of latent factors ( $k$ ). Set  $k = 10, 20, 30, 50, 100$ , and plot your  $E_{in}, E_{out}$  against  $k$ . What trends do you notice in the plot? Can you explain them?

**Question E:** Now, do the same thing, but this time with the regularization term. Choose an appropriate  $\lambda$ , and use the same range of values for  $k$  as you did in the previous part. Note that you may have to use different values of  $\lambda$  for different values of  $k$ . What trends do you notice in the graph? Can you explain them in the context of your plots for the previous part?

### 3 Word2Vec

The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or document. Given a sequence of training words  $w_1, w_2, \dots, w_W$ , the objective of the Skip-gram model is to maximize the average log probability


$$\frac{1}{W} \sum_{i=1}^W \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t) \quad (1)$$


where  $c$  is the size of the training context. Larger  $c$  results in more training examples and higher accuracy, at the expense of training time.

#### Softmax

**Question A:** The basic Skip-gram formulation defines

$$p(w_O | w_I) = \frac{\exp(v'_{w_O}^T v_{w_I})}{\sum_{w=1}^W \exp(v'_w{}^T v_{w_I})} \quad (2)$$

where  $v_w$  and  $v'_w$  are the “input” and “output” vector representations of  $w$ , and  $W$  is the number of words in the vocabulary. How does computing  $\nabla \log p(w_O | w_I)$  scale with  $W$ ? 

**Question B:** Each embedding vector  $v \in \mathbb{R}^D$  lives in a  $D$ -dimensional space. In principle, one could use any  $D$ . What do you expect to happen to the training objective as  $D$  increases? Why do you think one might not want to use very large  $D$ ? 

#### Hierarchical Softmax

A computationally **efficient** approximation of softmax is the hierarchical softmax. The hierarchical softmax uses a **binary tree representation** of the output layer with the  $W$  words as its leaves and, for each node, explicitly represents the relative probabilities of its child nodes.

Unlike the standard formulation of the Skip-gram, which assigns two representations,  $v_w$  and  $v'_w$  to each word  $w$ , the hierarchical softmax formulation has one representation for each word  $w$  and one representation  $v'_n$  for every inner node  $n$  of the binary tree. Figure 1 shows the structure of the hierarchical softmax tree.

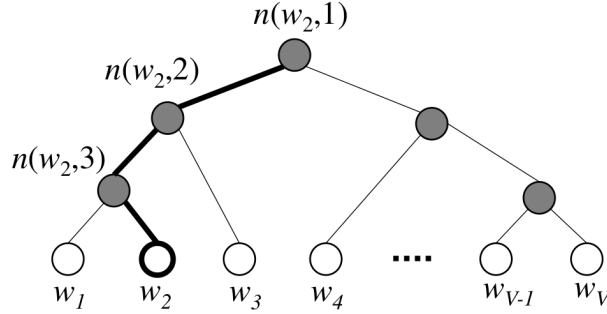




Figure 1: Hierarchical Softmax Tree

**Question C:** The main advantage of the hierarchical softmax is that only  $\log_2(W)$  nodes are required to obtain the probability distribution. Does this improve upon the result from Question A? 

**Question D:**

i. If  $W$  words are stored in a binary tree, how many internal nodes does the tree have? 

ii. How many vectors  $v_w$  and  $v'_w$  does the hierarchical softmax model represent? How does this **compare** to the number of vectors  $v_w$  and  $v'_w$  the basic softmax model represents? Thus, the hierarchical softmax model does not require representing prohibitively many more vectors than the basic softmax model.

**Question E:**

i. Each word  $w$  can be reached by an appropriate path from the root of the tree. Let  $n(w, j)$  be the  $j$ -th node on the path from the root to  $w$ , and let  $L(w)$  be the length of this path, so  $n(w, 1) = \text{root}$  and  $n(w, L(w)) = w$ . In addition, for any inner node  $n$ , let  $\text{ch}(n)$  be an arbitrary fixed child of  $n$  and let  $[[x]] = \begin{cases} 1 & x \text{ is true} \\ -1 & x \text{ is not true} \end{cases}$ . For simplicity, you may assume  $\text{ch}(n)$  is always the left child of  $n$ . Then the hierarchical softmax defines  $p(w_O|w_I)$  as follows:

$$p(w|w_I) = \prod_{j=1}^{L(w)-1} \sigma([n(w, j+1) = \text{ch}(n(w, j))]) \cdot v'^T_{n(w, j)} v_{w_I} \quad (3)$$

where  $\sigma(x) = 1/(1 + \exp(-x))$ . **Verify that the hierarchical softmax defines a distribution.** That is, verify  $\sum_{w=1}^W p(w|w_I) = 1$  (Hint: Show  $\sigma(x) + \sigma(-x) = 1$ )

This implies that the cost of computing  $\log p(w_O|w_I)$  and  $\nabla \log p(w_O|w_I)$  is proportional to  $L(w_O)$ , which on average is no greater than  $\log W$ .

ii. Draw a hierarchical softmax tree for  $|W| = 4$ , and label each node with all applicable values of  $n(w_i, j)$  or  $w_i$ . Note, the labels  $w_i$  should be increasing from left to right, as in Figure 1. Given the following table of dot products, compute  $p(w_1|w_3)$  using the hierarchical softmax formulation 3.

Vector	Vector	Dot
$v_{w_1}$	$v_{n(w_1,1)}$	$\frac{1}{2}$
$v_{w_1}$	$v_{n(w_4,2)}$	$\frac{1}{3}$
$v_{w_1}$	$v_{n(w_2,2)}$	$\frac{1}{4}$
$v_{w_1}$	$v_{w_2}$	$\frac{1}{5}$
$v_{w_1}$	$v_{w_3}$	$\frac{1}{6}$
$v_{w_1}$	$v_{w_4}$	$\frac{1}{7}$

**Question F:** Suppose you want to train a model using gradient descent to minimize 1 using either the softmax 2 or hierarchical softmax 3 formulation. The feature vectors cannot be initialized to all zeros. What is the problem in this case?

## Subsampling

**Question G:** In very large corpora, the most frequent word can easily occur hundreds of millions of times (such as “the”, “in”, “a”). Such words usually provide less information value than rare words. One way counter the imbalance between the rare and frequent words is by using subsampling: each word  $w_i$  in the training set is discarded with probability

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

where  $f(w_i)$  is the frequency of the word, and  $t$  is a chosen threshold, typically around  $10^{-5}$ .

Show the ranking of the frequencies is preserved in expectation by this subsampling policy. That is, if  $f(w_i) > f(w_j)$  before the subsampling, show  $\mathbb{E}[f(w_i)] > \mathbb{E}[f(w_j)]$  after the subsampling.

## Empirical Results

You will now have the chance to observe the power of Word2Vec yourself. Download the implementation from <http://word2vec.googlecode.com/svn/trunk> (with svn, you can use “svn checkout <http://word2vec.googlecode.com/svn/trunk>”). Then run ./demo-word.sh (note, this may take a while to load). If this takes too long to load, you may use a subset of test8 to train.

### Question H:

- Spend a few minutes exploring the similarities of different words. What is the highest-similarity pair of words you found? What is their similarity? Can you reason about why this pair of words is similar?
- Suppose a high similarity between words denoting family members indicates caring between those family members. What are some interesting relations between family members this model suggests? (Consider the ordering of words such as “father”, “mother”, “child”, “daughter”, “son”, etc.)
- If your name is in the dictionary, what words are your name closest to? Does this result make intuitive sense?