HOMEWORK 9: LEARNING PARADIGMS *

10-301 / 10-601 INTRODUCTION TO MACHINE LEARNING (FALL 2026)
https://www.mlcourse.org/10601

OUT: Monday, November 23
DUE: Tuesday, December 1
TAs: Muskan, Harry, Jerry, Alice, Markov, Neural

Summary This is the final homework assignment. This assignment covers **Ensemble Methods**, **Autoencoders**, **PCA**, and **Recommender Systems**.

START HERE: Instructions

- **Collaboration Policy**: Please read the collaboration policy here: https://www.mlcourse.org/10601/syllabus
- **Late Submission Policy:** See the late submission policy here: https://www.mlcourse.org/10601/syllabus
- **Submitting your work:** You will use PhiGrade to submit answers to all questions.
  - **Written:** For written problems such as short answer, multiple choice, derivations, proofs, or plots, please use the provided template. Submissions can be handwritten onto the template, but should be labeled and clearly legible. If your writing is not legible, you will not be awarded marks. Alternatively, submissions can be written in LaTeX. Each derivation/proof should be completed in the boxes provided. You are responsible for ensuring that your submission contains exactly the same number of pages and the same alignment as our PDF template. If you do not follow the template, your assignment may not be graded correctly by our AI assisted grader and there will be a **2% penalty** (e.g., if the homework is out of 100 points, 2 points will be deducted from your final score).

*Compiled on 2026-11-24 at 16:13:49

Instructions for Specific Problem Types

For "Select One" questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

- ● Matt Gormley
- ○ Henry Chai
- ○ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

- ● Matt Gormley
- ○ Henry Chai
- ⊗ Noam Chomsky

For "Select all that apply" questions, please fill in all appropriate squares completely:

Select all that apply: Which are instructors for this course?

- ■ Matt Gormley
- ■ Bryan Wilder
- □ Henry Chai
- □ I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are the instructors for this course?

- ■ Matt Gormley
- ■ Bryan Wilder
- ⊠ Henry Chai
- ⊠ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

10-601	10-~~6~~301

1 Written Questions (46 points)

1 LaTeX Point and Template Alignment (1 points)

1.1. (1 point) **Select one:** Did you use LaTeX for the entire written portion of this homework?

- ◯ Yes
- ◯ No

1.2. (0 points) **Select one:** I have ensured that my final submission is aligned with the original template given to me in the handout file and that I haven't deleted or resized any items or made any other modifications which will result in a misaligned template. I understand that incorrectly responding yes to this question will result in a penalty equivalent to 2% of the points on this assignment.
Note: Failing to answer this question will not exempt you from the 2% misalignment penalty.

- ◯ Yes

1.3. (0 points) **Select one:** Did you fill out the Exit Poll for the previous HW? Completing the exit poll will count towards your participation grade.

- ◯ Yes

2 PCA (6 points)

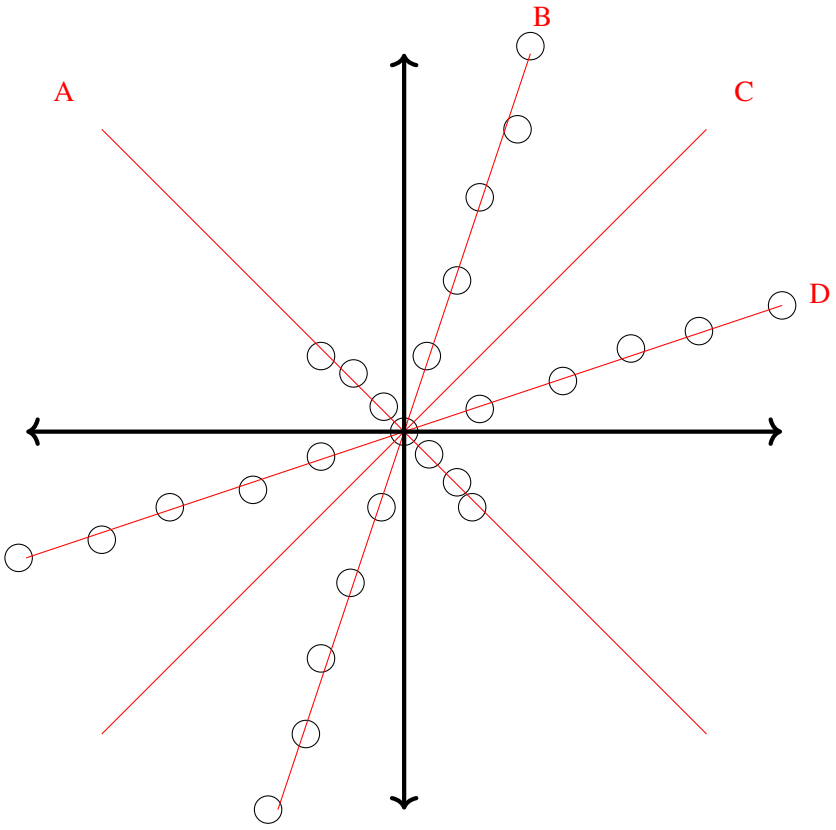
Some PCA Theory

2.1. (1 point) **Select one:** Assume we apply PCA to a matrix $\mathbf{X} \in \mathbb{R}^{n \times 2}$ and obtain two sets of PCA feature scores, $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n$, where $\mathbf{z}_1$ corresponds to the first principal component and $\mathbf{z}_2$ corresponds to the second principal component. Which is more common in the training data:

- ◯ a point with large feature values in $\mathbf{z}_1$ and small feature values in $\mathbf{z}_2$
- ◯ a point with small feature values in $\mathbf{z}_1$ and large feature values in $\mathbf{z}_2$

2.2. (2 points) Based on the dataset shown below, list only the principal components in order from first to last. You might not list all four principle components. Be sure to consider all data points in the graph, not just those that lie on the indicated lines.

Your Answer



PCA in Practice

For this section, refer to the PCA demo linked here. In this demonstration, we have performed PCA for you on a simple four-feature dataset. Run the code in the notebook, then answer the questions below based on the results. The questions have also been copied to the relevant code cells in the Colab notebook for ease.

2.3. (1 point) **Select one:** Do you see any special relationships between any of the features? In particular, take a look at the `petal_length` feature. How would you describe its association with each of the **other features**? Select the correct statement *with appropriate justification.*

- ◯ The features are highly correlated: we observe linearly proportional relationships where increases in `petal_length` often correspond to increases in another feature
- ◯ The features are highly correlated: we observe that the color classes can be separated with decision boundaries along the `petal_length` axis.
- ◯ The features are uncorrelated: we observe random noise as if the features were generated from independent distributions
- ◯ The features are uncorrelated: we observe the "default $y = x$" relationship between features

2.4. (1 point) If we wanted to find k principal components such that we preserve **at least** 95% of the variance in the data, what would be the value of k? Hint: it is helpful here to look at the cumulative variance in the first k components, which we have calculated for you.



2.5. (1 point) See the visualization below of the first two principal components for the Fisher Iris Dataset (i.e. the projection of the Fisher Iris Dataset down to two dimensions), where the points are colored by label. From this graph, do you think PCA is an appropriate dimensionality reduction method for this dataset? Why or why not? Explain in your own words.

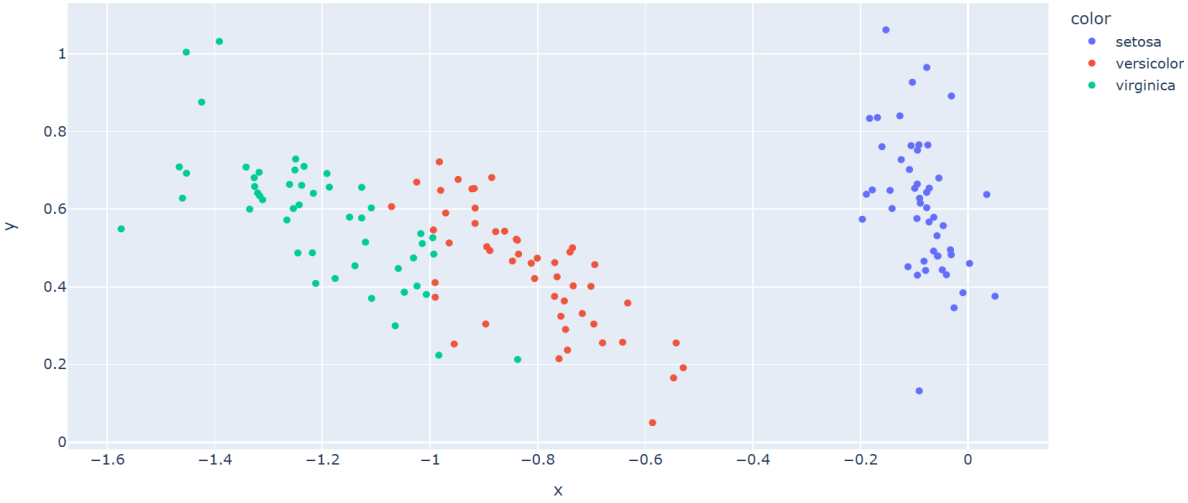


Figure 1: Fisher Iris Dataset in 2D

Explanation

3 Autoencoders (11 points)

3.1. (1 point) **Select all that apply:** An autoencoder is trained to reconstruct its input $\mathbf{x}^{(i)} \in \mathbb{R}^d$ from itself. Which of the following design choices prevent the autoencoder from achieving zero training loss by simply learning the identity map?

- ☐ A continuous hidden layer $\mathbf{u}^{(i)} \in \mathbb{R}^k$ with $k < d$.
- ☐ Corrupting each input with random noise and training the network to reconstruct the clean input.
- ☐ A hidden layer with $k > d$ units, together with an L_1 penalty $\lambda\|\mathbf{u}^{(i)}\|_1$ on the hidden activations.
- ☐ A linear autoencoder with $k = d$ hidden units and no regularizer.
- ☐ Adding more hidden layers to the encoder and decoder, keeping every layer's width at least d.

3.2. Traditional autoencoders typically use a low-dimensional bottleneck ($k < d$) to compress inputs, while sparse autoencoders instead encourage sparse hidden activations through an L1 penalty.

Consider a sparse autoencoder with input $\mathbf{x} \in \mathbb{R}^2$ and $k = 3$ hidden units (so $k > d$):

$$\mathbf{u} = \mathrm{ReLU}(W^\top \mathbf{x} + \mathbf{c}), \qquad \hat{\mathbf{x}} = V\mathbf{u} + \mathbf{b},$$

trained with loss $\ell(\mathbf{x}) = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda\|\mathbf{u}\|_1$, where $\lambda = 0.1$ and

$$W^\top = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} 0 \\ 0 \\ -2 \end{bmatrix}, \quad V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.5 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Parts (a) through (d) refer to the single input $\mathbf{x} = (2, 1)$.

3.2.a. (1 point) What is the hidden activation $\mathbf{u}$?

Answer

3.2.b. (1 point) What is the reconstruction $\hat{\mathbf{x}}$?

Answer

3.2.c. (1 point) What is the loss $\ell(\mathbf{x})$?

Answer

3.2.d. (1 point) **Select one:** Suppose we increase λ substantially and retrain. Which outcome is most likely?

- ◯ Fewer hidden units are nonzero per input, and reconstruction error increases.
- ◯ More hidden units are nonzero per input, and reconstruction error decreases.
- ◯ The number of nonzero units is unchanged, but each unit's activation grows larger.

3.2.e. (1 point) Now consider a VAE whose encoder outputs $\boldsymbol{\mu}_\phi(\mathbf{x}) = (1, -2)$ and $\boldsymbol{\sigma}_\phi(\mathbf{x}) = (0.5, 2)$. Using the reparameterization trick with noise sample $\boldsymbol{\varepsilon} = (2, -1)$, what is $\mathbf{z}$?

Answer

3.3. Recall that a linear autoencoder (without biases) with k hidden units solves

$$\min_{V,W \in \mathbb{R}^{d \times k}} \sum_{i=1}^{N} \|VW^\top \mathbf{x}^{(i)} - \mathbf{x}^{(i)}\|_2^2.$$

Instead of holding the hidden dimension k fixed, your friend Neural tries to also minimize the objective over k, solving

$$\operatorname*{argmin}_k \min_{V,W \in \mathbb{R}^{d \times k}} \sum_{i=1}^{N} \|VW^\top \mathbf{x}^{(i)} - \mathbf{x}^{(i)}\|_2^2.$$

You found this idea to be a bad one.

3.3.a. (1 point) What is the minimum possible value of the objective function when minimizing over k?

Answer

3.3.b. (1 point) **Select one:** Why is allowing $k \geq d$ generally undesirable when training a linear autoencoder for dimensionality reduction?

- ◯ Because linear autoencoders cannot reconstruct inputs when $k \geq d$.
- ◯ Because the autoencoder can learn the identity mapping without learning a lower-dimensional representation.
- ◯ Because increasing k always increases reconstruction error.
- ◯ Because the encoder and decoder must have identical weights.

3.4. (1 point) **Select one:** A linear autoencoder with k hidden units (no nonlinearities) is trained to convergence on centered data with squared reconstruction error. Which statement is true?

- ◯ Its decoder columns are always exactly the top k principal components, in order.
- ◯ It always achieves lower reconstruction error than PCA with the same k.
- ◯ Its decoder columns span the same subspace as the top k principal components, though they need not be orthonormal or equal to the PCs themselves.

3.5. (1 point) **Select one:** To interpret a trained LLM, we record activations $\mathbf{h}^{(i)} \in \mathbb{R}^d$ at one layer and train a sparse autoencoder on them with $k \gg d$ hidden units. Which of the following best explains why we choose $k \gg d$?

- ◯ The model likely represents many more concepts than it has dimensions, so a wide, sparse dictionary lets each hidden unit correspond to a single concept.
- ◯ A wide hidden layer is needed so that the autoencoder can copy $\mathbf{h}^{(i)}$ to its output exactly.
- ◯ With $k \gg d$, the L_1 penalty has no effect, so the hidden units are free to take any value.

3.6. (1 point) **Select one:** In a variational autoencoder, the encoder outputs $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\boldsymbol{\sigma}_\phi(\mathbf{x})$, and we sample $\mathbf{z}$ from $\mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \operatorname{diag} \boldsymbol{\sigma}_\phi(\mathbf{x})^2)$. What does the reparameterization trick accomplish?

- ◯ It makes the ELBO exactly equal to the log-likelihood $\ln P(\mathbf{x} \mid \theta)$, so no bound is needed.
- ◯ It writes $\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, I)$, so $\mathbf{z}$ is a differentiable function of ϕ and we can backpropagate into the encoder.
- ◯ It removes the need for an encoder, since $\mathbf{z}$ can be sampled directly from the prior $\mathcal{N}(\mathbf{0}, I)$ during training.

4 Ensemble Methods (14 points)

4.1. (1 point) **True or False:** In a random forest, it is generally better for the trees to be highly correlated, as this reduces ensemble variance.

◯ True

◯ False

4.2. (2 points) **Select all that apply:** Which of the following is true about OOB error?

□ OOB error is calculated on a held-out dataset separate from the dataset used to generate bootstrap samples

□ OOB error is the aggregated value of the errors of subsets of the ensemble on samples those subsets were not trained on

□ Cross-validation error is the same as OOB error

□ OOB error can be used instead of validation error when hyperparameter tuning a random forest model

□ None of the above

4.3. (2 points) **Select all that apply:** Which of the following are hyperparameters that can be tuned in a random forest?

□ Number of trees trained

□ Number of points used to train each decision tree

□ Size of feature subsets used to train each decision tree

□ Which features are used for splits in each decision tree

□ None of the above

4.4. In this question, we will consider the behavior of an error bound for random forests in the case of **binary classification**. Given a random forest of B trees $\{h_i(x)\}_{i=1}^{B}$ and a sample (x, y) drawn from some data distribution $\mathcal{D}$, define the classification margin as:

$$m(x,y) = \frac{1}{B}\left(\sum_{i=1}^{B}\mathbb{I}[h_i(x) = y] - \sum_{i=1}^{B}\mathbb{I}[h_i(x) \neq y]\right)$$

In words, the margin $m(x, y)$ is the difference between the average vote for the correct label and the average vote for the incorrect label.

4.4.a. (1 point) **Fill in the blank:** For any example (x, y), the example is classified incorrectly if and only if $m(x, y) \leq$ ____. Assume majority vote ties are classified incorrectly.

Answer

4.4.b. (1 point) **Select one:** Using Chebyshev's inequality, it is possible to bound the generalization error of a random forest as $P(m(x,y) < c) \leq \frac{\text{Var}(m(x,y))}{s^2}$, where c is your answer to part (a) and s is the *strength* of the ensemble

$$s = \mathbb{E}_{(x,y)\sim\mathcal{D}}[m(x,y)].$$

Through some additional manipulation, it is further possible to show that $\text{Var}(m(x,y)) \leq \overline{\rho}(1-s^2)$, where $\overline{\rho}$ is the mean correlation between trees in the ensemble. Substitute this into the given bound. Which of the following describes how the error bound is affected by s and $\overline{\rho}$?

- ◯ The error bound gets smaller as $\overline{\rho}$ increases and $|s|$ increases.
- ◯ The error bound gets smaller as $\overline{\rho}$ increases and $|s|$ decreases.
- ◯ The error bound gets smaller as $\overline{\rho}$ decreases and $|s|$ increases.
- ◯ The error bound gets smaller as $\overline{\rho}$ decreases and $|s|$ decreases.

4.5. (1 point) **True or False:** Consider some training point $(x^{(i)}, y^{(i)})$ to the AdaBoost algorithm. If for all t, the weak learner h_t learned during training at time t correctly classifies $h_t(x^{(i)}) = y^{(i)}$, there will eventually be a finite time t such that the weight assigned to $x^{(i)}$ in the training distribution $\mathcal{D}_t$ reaches exactly 0.

◯ True

◯ False

4.6. Assume we use a deterministic training procedure for weak learners. Suppose for some iteration t' of AdaBoost we find that the weak classifier learned by the algorithm at time t' has error $\epsilon_{t'} = 0.5$ of the weak learner $h_{t'}$ on the training distribution weighted by $\mathcal{D}_{t'}$.

4.6.a. (1 point) What weight $\alpha_{t'}$ will AdaBoost assign to the classifier $h_{t'}$ from above?

$\alpha_{t'}$

4.6.b. (1 point) **Select all that apply:** Which of the following scenarios are possible?

□ $\mathcal{D}_{t'+1}(i) > \mathcal{D}_{t'}(i)$ and $h_{t'}$ correctly classifies $x^{(i)}$

□ $\mathcal{D}_{t'+1}(i) > \mathcal{D}_{t'}(i)$ and $h_{t'}$ incorrectly classifies $x^{(i)}$

□ $\mathcal{D}_{t'+1}(i) < \mathcal{D}_{t'}(i)$ and $h_{t'}$ correctly classifies $x^{(i)}$

□ $\mathcal{D}_{t'+1}(i) < \mathcal{D}_{t'}(i)$ and $h_{t'}$ incorrectly classifies $x^{(i)}$

□ None of the above.

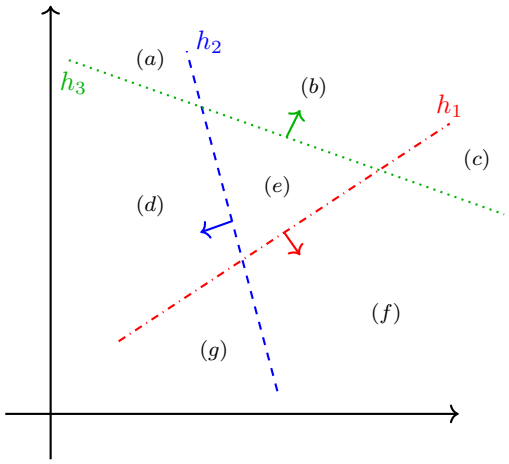
4.6.c. (1 point) **Select all that apply:** Which of the following are true about the next iteration of the AdaBoost algorithm?

□ The errors $\epsilon_{t'+1}$ and $\epsilon_{t'}$ are equivalent

□ The weak learners $h_{t'+1}$ and $h_{t'}$ will be equivalent (i.e., they will have the same output for every input)

□ None of the above

4.7. The figure below shows three binary classifiers h_1, h_2, h_3 in a 2D space. We intend to use AdaBoost to combine them into a strong classifier. The data set consists of 7 points (a) - (g), and the weight initializations for these points are uniform (i.e. weights are equal and add up to 1). The true label of (g) is negative. Use this information to answer the following:



4.7.a. (1 point) **Select all that apply:** Using $\alpha_1 = \alpha_2 = \alpha_3 = 1$ as weights of each classifier, which data points among (a), (b), (c), (d), (e), (f), and (g) would be classified as +1 by weighted majority vote?

- ☐ (a)
- ☐ (b)
- ☐ (c)
- ☐ (d)
- ☐ (e)
- ☐ (f)
- ☐ (g)
- ☐ None of the above

4.7.b. (1 point) **Fill in the blank:** Use $\alpha_1 = 0, \alpha_2 = 1, \alpha_3 = 0$ as weights of each classifier, and assume that h_1, h_2, and h_3 update the sample distribution in this order. Fill in the weight of the datapoint (g) after each weak classifier is applied. No weight normalization is needed.

w_1	w_2	w_3

4.7.c. (1 point) **True or False:** Neural argues that no classifier can ever receive weight $\alpha = 0$, because the dataset contains 7 (odd number) datapoints, the weighted error ϵ_t of any weak classifier can never be exactly 0.5. **Select one and briefly justify your answer.**

◯ True

◯ False

5 Recommender Systems (14 points)

5.1. (2 points) **Select all that apply:** In which of the following situations will a collaborative filtering system be a more appropriate learning algorithm than a linear or logistic regression model?

- □ You manage an online bookstore, and you have book ratings and sales data from many users. For each user, you want to recommend other books she will enjoy, based on her own ratings and the ratings of other users.
- □ You manage an online bookstore, and you have book ratings and sales data from many users. You want to learn to predict the expected sales volume (number of books sold) as a function of the average rating of a book.
- □ You run an online news aggregator, and for every user, you know some subset of articles that the user likes and some different subset that the user dislikes. You want to use this to find other articles that a given user likes.
- □ You've written a piece of software that downloads news articles from many news websites. In your system, you also keep track of which articles you personally like and which ones you dislike, and the system also stores away features of these articles (e.g., word counts, name of author). Using this information, you want to build a system to try to find additional new articles that you personally will like.
- □ None of the above

5.2. Neural the Narwhal decides to set up a friend-recommendation system for all the students in 10-301/601, of which there are $N = 10,301,601$. Ideally, Neural would store the full $N \times N$ matrix M, where M_{ij} is 1 if student i and j are friends, 1 if $i = j$, 0 if student i and j are nemeses, or `null` if students i and j have never met. Assume that these are the only possible relationships between 2 people and that all relationships are symmetric (so it cannot be the case that student i thinks student j is their friend while j thinks i is their nemesis). Unfortunately, storing M in its entirety would take over 10 TB of storage, which Neural cannot afford on a TA salary. Neural instead uses the following procedure to approximate M as UU^T for some $N \times d$ matrix U, where each row of U corresponds to a student.

Given learning rate η, ground truth relationships M
Randomly initialize user embedding matrix $U \in \mathbb{R}^{N \times d}$
while not converged **do**
  Sample $i \sim$ Uniform(1, N), $j \sim$ Uniform(1, N)
  $\hat{M}_{ij} \leftarrow \sigma(\vec{u}_i^T \vec{u}_j)$ // σ is sigmoid function
  $\mathcal{L}(\vec{u}_i, \vec{u}_j) \leftarrow -M_{ij} \log(\hat{M}_{ij}) - (1 - M_{ij}) \log(1 - \hat{M}_{ij})$ // Compute logistic loss
  $\vec{g}_i \leftarrow \nabla_{\vec{u}_i} \mathcal{L}(\vec{u}_i, \vec{u}_j)$ // Compute and perform gradient updates
  $\vec{g}_j \leftarrow \nabla_{\vec{u}_j} \mathcal{L}(\vec{u}_i, \vec{u}_j)$
  $\vec{u}_i \leftarrow \vec{u}_i$ - $\eta \cdot \vec{g}_i$
  $\vec{u}_j \leftarrow \vec{u}_j$ - $\eta \cdot \vec{g}_j$

5.2.a. (1 point) Explain why line 4 of the algorithm is incorrect. You do not need to provide a line of code unless you feel that it would help you in your explanation.

Explanation

5.2.b. (1 point) **Select one:** Based on the loss function given on line 6, derive an expression for $\nabla_{\vec{u}_j}\mathcal{L}(\vec{u}_i, \vec{u}_j)$ in terms of $\vec{u}_i$, $\vec{u}_j$, and M_{ij}. Note that σ denotes the sigmoid function and $\log$ is natural log.

- ◯ $\vec{u}_i M_{ij}\sigma(\vec{u}_i^T\vec{u}_j)$
- ◯ $\vec{u}_i(M_{ij} + \sigma(\vec{u}_i^T\vec{u}_j))$
- ◯ $\vec{u}_i(M_{ij} - \sigma(\vec{u}_i^T\vec{u}_j))$
- ◯ $\vec{u}_i(-M_{ij} + \sigma(\vec{u}_i^T\vec{u}_j))$

5.2.c. (2 points) **Select all that apply:** Why is it appropriate here to factorize M as the product of a single matrix by itself UU^T rather than as the product of two distinct matrices VW^T?

- □ UU^T enforces that our approximation is symmetric, while VW^T is not necessarily symmetric.
- □ SGD is guaranteed to converge faster and to a better optimum for U because we have fewer parameters to learn.
- □ We can parallelize training for U, whereas we could not parallelize training for V and W.
- □ We wish to model relationships between objects of a single type, not between objects of two different types.
- □ None of the above.

5.3. Neural the Narwhal is building a generative recommender system for an online bookstore. Each book is represented by a two-dimensional content embedding $\vec{x}$. Neural assigns each book a Semantic ID using two levels of *residual quantization*:

1. At level 1, choose the code whose vector is closest to $\vec{x}$ in Euclidean distance.
2. Compute the residual by subtracting the selected level-1 vector from $\vec{x}$.
3. At level 2, choose the code whose vector is closest to the residual in Euclidean distance.

A Semantic ID is a sequence of codes derived from an item's content. It represents the item in a form that a generative recommender can predict. In this example, the Semantic ID is the ordered pair of selected codes.

For one book, suppose

$$\vec{x} = \begin{bmatrix} 3 \\ 2 \end{bmatrix},$$

and Neural has the following codebooks:

Level 1 Code	Vector	Level 2 Code	Vector
A	$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$	P	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
B	$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$	Q	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$
C	$\begin{bmatrix} 0 \\ 2 \end{bmatrix}$	R	$\begin{bmatrix} -1 \\ 0 \end{bmatrix}$

5.3.a. (2 points) What is the book's Semantic ID? Show the selected level-1 code, the residual, and the selected level-2 code.

Solution

5.3.b. (1 point) **Select one:** A reader bought picture books ten years ago and science nonfiction in the past month. Neural uses a two-tower recommender that computes a user embedding from the embeddings of the books the reader bought. Neural knows the purchase dates and wants each recent purchase to receive more weight than each older purchase, while still giving every purchase some weight.

Which change would accomplish this?

- ◯ Use only books bought in the past month to compute the user embedding.
- ◯ Average the embeddings of all purchased books with equal weights.
- ◯ Average the embeddings using positive weights that decrease with the time since each purchase.
- ◯ Sort the books from newest to oldest, then average their embeddings with equal weights.

5.4. Neural the Narwhal now works at *Narwhal Tunes*, a music streaming service. A *query* $\vec{x}$ consists of the song currently playing, the listener's recent history, and the time of day. An *item* y is a song, described by its ID, artist, genre and audio features. The team trains a two-tower model with relevance score

$$s(\vec{x}, y) = \vec{u}(\vec{x}; \boldsymbol{\theta})^T \vec{v}(y; \boldsymbol{\theta})$$

from logs of pairs $(\vec{x}_i, y_i)$, where y_i is the song the listener chose next.

5.4.a. (1 point) A music streaming service has $M = 40{,}000{,}000$ songs. Ana proposes using a single neural network $f(\mathbf{x}, y)$ to score each query-song pair. Ben proposes using a two-tower model, where song embeddings can be precomputed.

How many neural network forward passes does each approach require per request to score the entire catalog?

Ana

Ben

5.4.b. (1 point) In one training batch with $B = 3$, the scores for query $\vec{x}_2$ are

	y_1	y_2	y_3
$s(\vec{x}_2, \cdot)$	$\ln 2$	$\ln 3$	0

Here y_2 is the song listened to after $\vec{x}_2$. Compute the in-batch softmax loss $-\ln p_B(y_2 \mid \vec{x}_2)$.

Solution

5.4.c. (1 point) In the batch from the previous part, songs appear in the training stream with frequencies $p_1 = 0.5$, $p_2 = 0.25$ and $p_3 = 0.25$. Song y_1 is a chart-topping single. The team now replaces each score with $s(\vec{x}_2, y_j) - \ln p_j$. Recompute the loss.

Solution

5.4.d. (1 point) In one or two sentences, explain what behavior the change in the previous part is meant to fix, and why a song like y_1 is the one affected.

Explanation

5.4.e. (1 point) **Select one:** At 3 a.m., an indie artist uploads a new song that nobody has played yet. The song has a title, artist, genre tags and audio features. Narwhal Tunes also runs a matrix factorization model, with one learned vector per user and per song stored in lookup tables. Which statement is correct?

- ◯ Neither model can score the song until it has been played enough times to accumulate listening data.
- ◯ Both models can score it immediately, because each one produces a vector for every song in the catalog.
- ◯ The matrix factorization model can score it, because it only needs the song's ID, but the two-tower model cannot, because it needs the song's listening history.
- ◯ The two-tower model can score it, because its item embedding is computed from the song's features, but the matrix factorization model has no learned row for it.

6 Collaboration Questions

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found here.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details.
2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details.
3. Did you find or come across code that implements any part of this assignment? If so, include full details.

Your Answer