Problem 1 (10 marks)

(Eigenchangers!) Suppose we wish to do PCA for an $N \times D$ matrix $\mathbf{X}$ and assume $D > N$. The traditional way to do PCA is to compute the eigenvectors of the covariance matrix $\mathbf{S} = \frac{1}{N}\mathbf{X}^\top\mathbf{X}$ (assuming centered data). Show that, if someone instead gives you an eigenvector $\boldsymbol{v} \in \mathbb{R}^N$ of the matrix $\frac{1}{N}\mathbf{X}\mathbf{X}^\top$, you can use it to get an eigenvector $\boldsymbol{u} \in \mathbb{R}^D$ of $\mathbf{S}$. What is the advantage of this way of obtaining the eigenvectors of $\mathbf{S}$?

Problem 2 (10 marks)

(A General Activation Function) Consider the following activation function: $h(x) = x\sigma(\beta x)$ where σ denotes the sigmoid function $\sigma(z) = \frac{1}{1+\exp(-z)}$. Show that, for appropriately chosen values of β, this activation function can approximate (1) the identity activation function, and (2) the ReLU activation function.

Problem 3 (15 marks)

(Mixtures meet Neural Nets!) Consider modeling some data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \{0, 1\}$, using a mixture of logistic regression models, where we model each binary label y_n by first picking one of the K logistic regression models, based on the value of a latent variable $z_n \sim$ multinoulli$(\pi_1, \ldots, \pi_K)$, and then generating y_n *conditioned* on z_n as $y_n \sim$ Bernoulli$[\sigma(\boldsymbol{w}_{z_n}^\top \boldsymbol{x}_n)]$.

Now consider the *marginal* probability of the label $y_n = 1$, given $\boldsymbol{x}_n$, i.e., $p(y_n = 1|\boldsymbol{x}_n)$, and show that this can be quantity can also be thought of as the output of a neural network. Clearly specify what is the input layer, hidden layer(s), activations, the output layer, and the connection weights of this neural network.

Problem 4 (30 marks)

(Probabilistix Formulation of Matrix Factorization with Side Information) Consider an $N \times M$ rating matrix $\mathbf{X}$, where the rows represents the N users and the columns represent the M items. We are also given some side information: for each user n, a feature vector $\boldsymbol{a}_n \in \mathbb{R}^{D_U}$, and for each item m, a feature vector $\boldsymbol{b}_m \in \mathbb{R}^{D_I}$. Let's model each entry of $\mathbf{X}$ as $p(X_{nm}|\boldsymbol{u}_n, \boldsymbol{v}_m, \theta_n, \phi_m) = \mathcal{N}(X_{nm}|\theta_n + \phi_m + \boldsymbol{u}_n^\top \boldsymbol{v}_m, \lambda_x^{-1})$.

In this model, $\boldsymbol{u}_n \in \mathbb{R}^K$ and $\boldsymbol{v}_m \in \mathbb{R}^K$ represent the user n and item m latent factors, respectively. In addition, for each user n, we have a user-specific bias $\theta_n \in \mathbb{R}$ (bias regardless of the item being rated), and for each item m, we have an item-specific bias $\phi_m \in \mathbb{R}$ ("popularity" of the item, regardless is who has rated this item).

Assume Gaussian priors on the latent factors $\boldsymbol{u}_n \in \mathbb{R}^K$ and $\boldsymbol{v}_m \in \mathbb{R}^K$: $p(\boldsymbol{u}_n) = \mathcal{N}(\boldsymbol{u}_n|\mathbf{W}_u\boldsymbol{a}_n, \lambda_u^{-1}\mathbf{I}_K)$ and $p(\boldsymbol{v}_m) = \mathcal{N}(\boldsymbol{v}_m|\mathbf{W}_v\boldsymbol{b}_m, \lambda_v^{-1}\mathbf{I}_K)$. Note that the *mean* of the priors of the latent factors $\boldsymbol{u}_n$ and $\boldsymbol{v}_m$ depends on the given user and item features ($\boldsymbol{a}_n$ and $\boldsymbol{b}_m$, respectively) via a linear model with regression parameters matrices $\mathbf{W}_u \in \mathbb{R}^{K \times D_U}$ and $\mathbf{W}_v \in \mathbb{R}^{K \times D_I}$, respectively. You don't need to assume any priors on the bias parameters θ_n, ϕ_m, and the regression parameters $\mathbf{W}_u$, $\mathbf{W}_v$.

Assume $\Omega = \{(n, m)\}$ to denote the set of indices of the observed entries of $\mathbf{X}$, Ω_{r_n} to be the set of items rated by user n, and Ω_{c_m} to be the set of users who rated item m (similar notation that we saw in class).

Your goal is to estimate $\{\boldsymbol{u}_n, \theta_n\}_{n=1}^N$, $\{\boldsymbol{v}_n, \phi_m\}_{m=1}^M$, $\mathbf{W}_u$, and $\mathbf{W}_v$. Assume all other parameters to be known.

Write down the loss function for this model (this will be the negative of the MAP objective for the model) and use the ALT-OPT algorithm to derive the update equations for all the unknowns. The expressions must be in closed form (it is possible for this model).