STAT 339: HOMEWORK 5 (MCMC, GRAPHICAL MODELS, AND HMMS)

UPDATED: DUE ON BLACKBOARD BY CLASS TIME MONDAY 5/1)

Instructions. Turn in your writeup and code to Blackboard as an archive file (e.g., `.zip`, `.tar`, `.gz`) by the start of class on Monday 4/10. **Note: To make grading smoother, please include a main writeup file in your archive in pdf form with a file name like `cdawson.pdf` (sub your Obie ID). All plots and results should be included and described here, with references as appropriate to implementation files.**

As always, you may use any language you like for the progamming components of this assignment — the tasks are stated in a language-neutral way. You may also use any typesetting software to prepare your writeup, but the final document should be a PDF. LaTeXis encouraged; a reproducible research format in which code is embedded into the document (e.g., knitr, RMarkdown, Jupyter or IPython Notebook) is even more encouraged.

All data is available at `http://colindawson.net/data/<name>.csv`.

Date: Last Revised: April 16, 2017.

1. **Gibbs Sampling with a Gaussian Mixture Model.** Consider the Gaussian mixture model we used for clustering on the last assignment. The distribution of each feature vector $\mathbf{y}_n$ is modeled as a weighted average of K different D-dimensional Gaussians (where D is the dimension of the feature space:

$$p(\mathbf{y}_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{y}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

On the last assignment you implemented the EM algorithm to find a local maximum of the likelihood function over the full set of parameters $\boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$. For the same model consider putting conjugate priors on these parameters: a Dirichlet prior distribution on $\boldsymbol{\pi}$, a D-dimensional Normal on each $\boldsymbol{\mu}_k$, and D independent Gamma distributions on the diagonal entries of each $\boldsymbol{\Sigma}_k^{-1}$; or if you prefer, inverse-Gamma distributions on the diagonal entries of $\boldsymbol{\Sigma}_k$.

   (a) Implement a Gibbs sampler to generate samples from the posterior distribution of $z_1, \ldots, z_N, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ given $\mathbf{y}_1, \ldots, \mathbf{y}_N$, where z_n is the cluster indicator taking a value between 1 and K for each n. Run your sampler for a few values of K on a couple of datasets that you clustered on HW4. To assess convergence, plot a few of the sampled variables as a function of iteration (e.g., the coordinates of the means of the clusters). You should run your sampler long enough to get 1000 or so sampled values of each parameter *after* convergence — i.e., when none of the means are no longer systematically drifting in a particular direction. You will probably want to save your sampled parameter values at each iteration to disk, so that you do not have to re-run your sampler every time you want to compute something new with the samples.

   (b) Using only the iterations following an assessment that the chain has more or less converged, estimate the "pairwise co-clustering matrix": that is, compute the matrix whose n, m entry represents the probability that $\mathbf{y}_n$ and $\mathbf{y}_m$ are in the same cluster. Remember that since you have samples of the z_n sequences, you can estimate this probability by simply counting the number of (post-convergence) iterations for which $z_n = z_m$ and dividing by the number of (post-convergence) iterations. Render this co-clustering matrix as a heatmap image (you can use the same method for this that you used to visualize the digit data in HW1), where brighter colors represent higher co-clustering probabilities. The diagonal should of course consist of all 1s.

   (c) Using only the iterations following an assessment that the chain has more or less converged, estimate the log of the marginal likelihood of the (training) data by averaging over the likelihoods computed for each sampled parameter

set. I.e., estimate

$$\log \mathbb{E}_{p(\boldsymbol{\theta} \mid \mathbf{y})}\left[p(\mathbf{y} \mid \boldsymbol{\theta})\right] \approx \log\left\{\frac{1}{S - S_{min} + 1}\sum_{s=S_{min}}^{S} p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)})\right\}$$

where S_{min} is chosen to be an iteration that is large enough that the sampler can be said to have converged by that point.

For numerical stability, you will want to make use of the following identity

$$\begin{aligned}
\log\left\{\sum_{s=1}^{S} p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)})\right\} &= \log\left\{\sum_{s=1}^{S} \exp\{\log(p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)}))\}\right\} \\
&= \log\left\{\sum_{s=1}^{S} \exp\{\log(p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)}) + C)\}\exp\{-C\}\right\} \\
&= \log\left\{\sum_{s=1}^{S} \exp\{\log(p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)}) + C)\}\exp\{-C\}\right\} \\
&= \log\left\{\sum_{s=1}^{S} \exp\{\log(p(\mathbf{y} \mid \boldsymbol{\theta}^{(s)}) + C)\}\right\} - C
\end{aligned}$$

where C is chosen to make the terms being exponentiated small in magnitude.

(d) Compare and plot the results from part (b) for different K for a few datasets. Does this metric tend to select a good value of K?

(e) Using the same computation as in (b), for some two-dimensional data of your choice (you might want to try it with more than one), plot contours of the marginal density function by computing it at each point in a grid. How does the marginal density function compare to the density for a specific choice of parameters?

2. Consider the Bayes net depicted in Fig. 1, which comes from the BRML book. Each variable is binary.

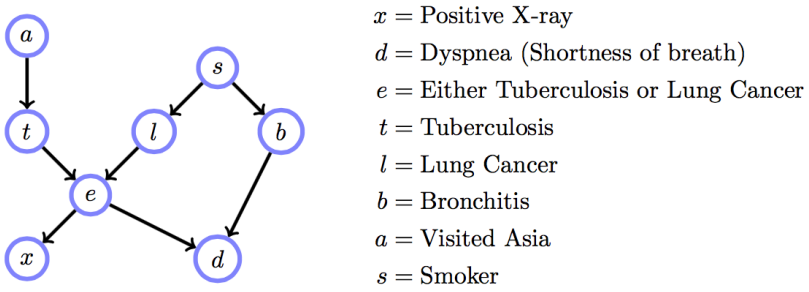

Figure 3.15: Belief network structure for the Chest Clinic example.

FIGURE 1. Bayes Net for diagnosis of lung disease at a chest clinic

(a) Write down the factorization of the joint distribution that is implied by the graph.

(b) According to the model, can you predict whether someone has visited Asia based on whether or not they are a smoker? That is, are s and a independent?

(c) Does knowing that someone is a smoker help you predict whether they visited Asia if you also have a chest x-ray? That is, are s and a conditionally independent given x? Explain the intuition behind these two results.

3. **HMMs for Typo Correction.** This exercise is based on Example 23.5 in BRML. Consider using an HMM as a model of typed sequences, in which the intended key is usually pressed, but with some probability a neighboring key is pressed instead. We will consider sequences consisting exclusively of lower case letters and spaces, so there are 27 total states and 27 total observable symbols. For simplicity, encode symbols as integers: $a = 1, b = 2, \ldots, z = 26, <\text{space}> = 27$. The "neighboring key" model defines an emission matrix, $\mathbf{B}$, where $\mathbf{B}_{kj} = p(x_t = j \mid z_t = k)$, where $\mathbf{B}_{kj}$ is large when $j = k$, moderate when the keys indexed by j and k are nearby on the (QWERTY layout) keyboard, and small when they are distant. The transition matrix, $\mathbf{A}$ has entries $\mathbf{A}_{k'k} = p(z_t = k \mid z_{t-1} = k')$, and is constructed using the statistics of English words. Heatmap images of these two matrices are shown in Figs. 2 and 3, respectively, where white is 1.0 and black is 0.0. The matrices, $\mathbf{A}$ and $\mathbf{B}$ themselves are available in the files `typing_transition_matrix.csv` and `typing_emission_matrix.csv`, respectively. Assume for simplicity that the first letter intended is uniformly chosen from the 26 non-space keys.

   (a) Implement a forward-filtering and backward-sampling algorithm to obtain samples from the posterior distribution of the latent state sequence $\mathbf{z}$ given the observed sequence $\mathbf{x}$. Recall (or anticipate, if you are reading this before we go over this in class) that forward filtering iteratively computes the posterior distribution for z_t given the observations $x_1, \ldots, x_{t-1}$ (which we abbreviate as $x_{1:t-1}$). This is obtained using the recursion relation:

$$p(z_t = k, x_{1:t}) = \sum_{k'=1}^{K} p(z_{t-1} = k', x_{1:t-1}) p(z_t = k \mid z_{t-1} = k') p(x_t \mid z_t = k)$$

   If $\mathbf{m}_t$ is defined as the forward "message" vector whose kth entry is $p(z_t = k \mid x_{1:t})$, and $\mathbf{A}$ and $\mathbf{B}$ are the transition and emission distributions as described above, then we have

$$\mathbf{m}_t = \mathbf{A}^\mathsf{T} \mathbf{m}_{t-1} \odot \mathbf{B}_{\cdot, x_t}$$

   where $\odot$ is the elementwise product, and $\mathbf{B}_{\cdot,j}$ is the jth column of $\mathbf{B}$. The initial message vector $\mathbf{m}_1$ is obtained by multiplying each initial probability $p(z_1 = k)$ by the likelihood $p(x_1 \mid z_1 = k)$.

   Note that although the equations above yield joint probabilities, the probabilities obtained will be increasingly small as we accumulate terms that we are multiplying together. For numerical stability (to avoid underflow), it is a good idea to rescale $\mathbf{m}_t$ by a constant at each iteration. Since we need to normalize eventually to obtain distributions over just the z_t values so we can sample them, this rescaling does not affect the posterior probabilities.

Once we have computed $\mathbf{m}_t$ for each $t = 1, \ldots, T$, then we can sample a sequence iteratively going backwards from $T, \ldots, 1$. First sample z_T from the distribution obtained by normalizing $\mathbf{m}_T$, and then, conditioning on what has already been sampled, compute

$$\begin{aligned} p(z_t = k, z_{t+1} = k, x_{1:T}) &= p(z_t = k, x_{1:t}) p(z_{t+1} = k' \mid z_t = k) p(x_{t+1:T} \mid z_{t+1} = k') \\ &\propto m_{t,k} \mathbf{A}_{kk'} \end{aligned}$$

where the last term in the first line can be dropped since it is constant with respect to z_t. As a vector, the distribution we are sampling from is proportional to

$$\mathbf{m}_t \odot \mathbf{A}_{\cdot k}$$

(where again, $\mathbf{A}_{\cdot,k}$ represents the kth column of $\mathbf{A}$.

(b) Apply your algorithm to sample a few thousand possible intended sequences given the observed sequence `kezrninh`. You will likely find it convenient to precompute a matrix of likelihoods: let $B^*_{tk} = p(x_t \mid z_t = k)$, since the x_t are fixed. Note that this is not MCMC as we have fixed the transition and emission matrices, and so we are sampling directly from the posterior distribution, and can do this as often as we like.

(c) Many of the possible sequences will consist of letter combinations that do not form real English words. The file `brit-a-z.txt` is a dictionary of British English (this was obtained from Barber, who works at a British university). I have provided a Python module, `text_utils.py` (in the same location as the data) that defines four functions: `dict_from_file()` creates a Python dictionary object out of a text file with one entry per line, where the keys are the words and the values are simply 1s. Note that all entries are lowercase; entries with capital letters will not be found. `decode_int_list_to_string()` converts a list of integers, each between 0 and 26 and converts `0` to `a`, `1` to `b`, etc., and `26` to a space, and returns the corresponding string. `encode_string_to_int_list()` does the opposite. Finally, `check_validity()` takes a string and a dictionary as returned by the first function, and returns `True` or `False` according to whether all space separated substrings appear in the dictionary. Use this code to discard sampled sequences that contain unknown or nonsense words. Note that if your code returns integer sequences, you will need to convert these into strings first. What is the most likely intended sequence for the observed sequence `kezrninh`, as measured by the sequence most often sampled by your algorithm that passes the validity check? (The correct answer should be `learning`)

(d) Suppose you do not have access to the transition and emission matrices. Can a typo model be learned from data? Implement a Gibbs sampler that alternates between sampling $\mathbf{z}$ sequences conditioned on a current guess for $\mathbf{A}$ and $\mathbf{B}$ (you can use the function you wrote in part (a) for this), and sampling $\mathbf{A}$ and $\mathbf{B}$ conditioned on the current guess for $\mathbf{z}$. Use independent symmetric Dirichlet priors for each row of the transition matrix, $\mathbf{A}$ (with prior weight $\alpha_{transition}/27$ attached to each entry), and use "diagonal-based" Dirichlet priors for each row of the emission matrix $\mathbf{B}$ — that is, use a weight of $\alpha_{emission}/27$s in each cell, except that in row k, the kth entry gets extra weight κ. This corresponds to the assumption that the most likely outcome is that the intended letter is produced. The smaller the extra weight added, the "noisier" we are assuming our data is. Using these priors, the posterior for row k of the transition matrix is simply

$$\mathsf{Dirichlet}(\alpha_{transition} + n_{k1}, \ldots, \alpha_{transition} + n_{kK})$$

where $n_{kk'}$ is the number of transitions from state k to k' in the current guess for $\mathbf{z}$. The posterior for row k of the emission matrix is similar, except for the extra mass of κ on entry k. Remember that you can sample from a Dirichlet using a pre-implemented library function, such as `numpy.random.dirichlet()`.

(e) Generate three long-ish (thousands of letters each) random sequences using the ground truth $\mathbf{A}$ and $\mathbf{B}$. One will be your training set, one your validation set and one your test set. Calculate the log marginal probability of the validation set using the $\mathbf{A}$ and $\mathbf{B}$ matrices inferred from the training set, varying the α and κ hyperparameters. The marginal probability of the data can be computed using the message vectors that we are already computing, by noting that

$$p(x_{1:T}) = \sum_{k=1}^{K} p(z_T = K, x_{1:T}) = \sum_{k=1}^{K} m_{Tk}$$

where m_{Tk} is the kth entry of $\mathbf{m}_t$ as defined above.

Using the best choices of hyperparameters, infer $\mathbf{A}$ and $\mathbf{B}$ for the test set, averaging over post-burn-in iterations, and compare the results to the ground truth matrices.

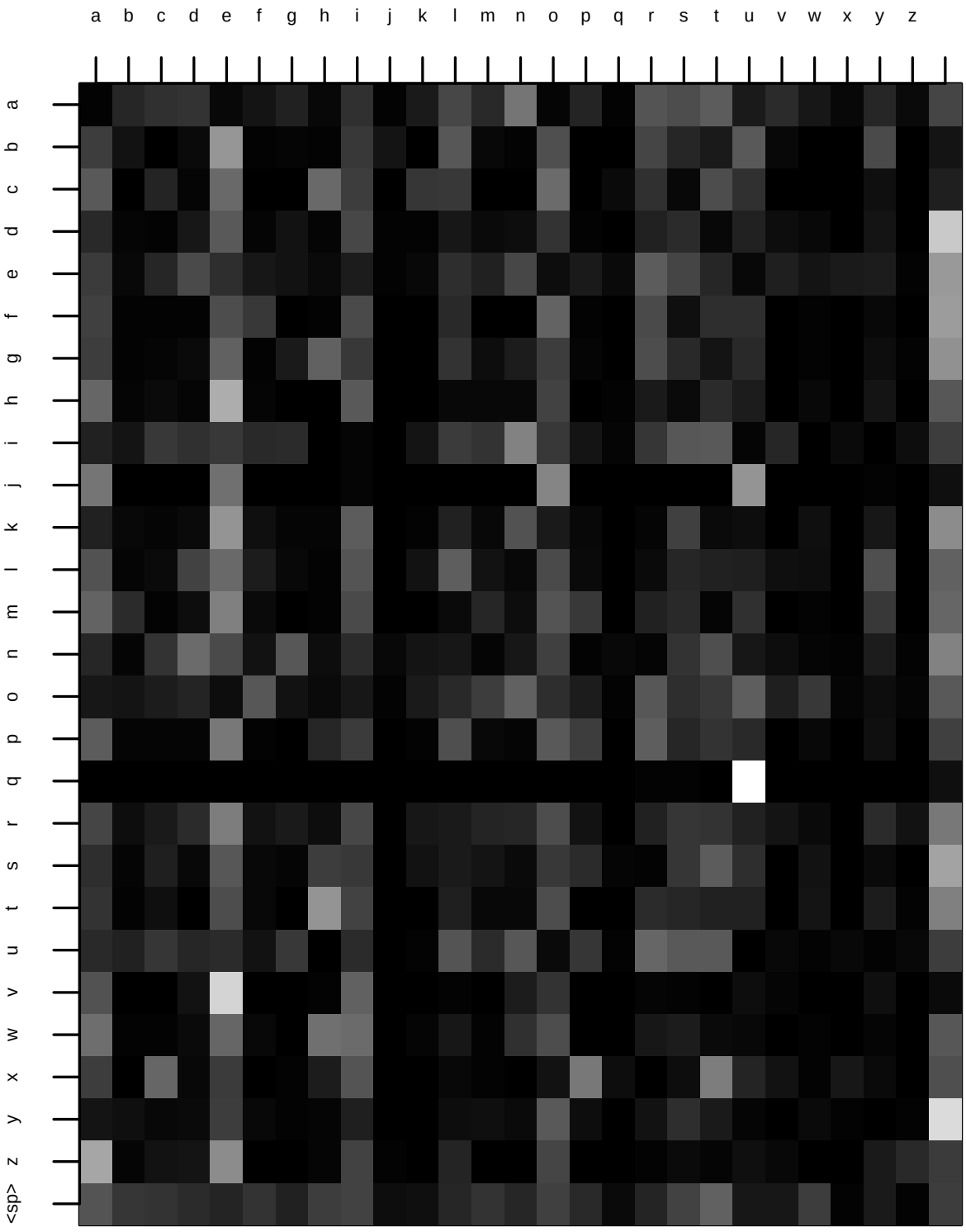

FIGURE 2. Transition matrix **A** for the typo HMM model

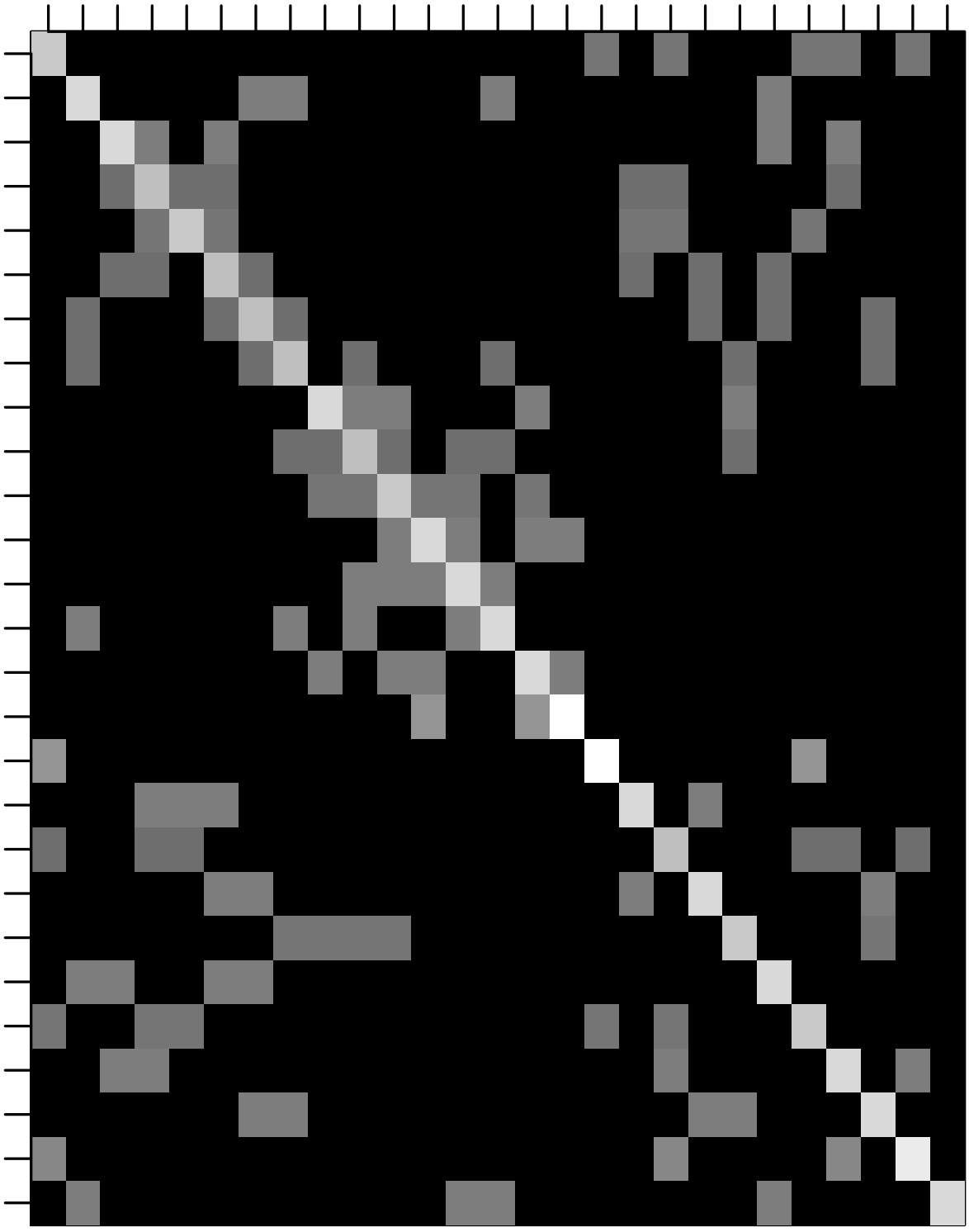

FIGURE 3. Emission matrix $\mathbf{B}$ for the typo HMM model