APPENDIX

A. Further Related Works

Some researchers studied Bayesian decompositions with other priors. [10] modeled country-to-country interactions using a four-mode tensor to represent an action between two countries in a given month. They applied a Bayesian Poisson Tucker decomposition to group countries, actions, and time steps. [40] computed the Tucker decomposition of a tensor using a normal prior and a variational expectation maximization posterior algorithm. [41] developed a CP Alternating Poisson Regression and fit their model using a log-likelihood score.

Hierarchical decompositions were studied in a non-Bayesian context. [42] defined the hierarchical Tucker format according to a hierarchy of vector subspaces. [43] computed hierarchical Tucker decompositions using hierarchical singular value decomposition. [23] defined a recursive tensor decomposition algorithm to estimate a latent tree graphical hierarchy. Their model depicts the joint probability of a set of observed variables as nodes dependent on their hidden parents. [24] adjusted the resolution or size of the core tensor to provide different resolution decompositions of the same data. Unlike our model, none of these models used Bayesian inference nor expressed the hierarchy of latent topics in each mode. Instead, these models depicted a hierarchy of vector spaces, a hierarchy of hidden variables, or different resolutions. [44] developed a hierarchical Dirichlet process to cluster grouped data.

B. Generative Process with Vectorization

In truth, we need to vectorize the topics in Algorithm 1, yielding the more complete Algorithm 6. For ease of notation, we define $K = \prod_{i=1}^{p} K_i$ and map vec : $\mathcal{K} \mapsto \{1, \cdots, K\}$ as a one-to-one mapping from a set of topics to a single topic index. We denote vec^{-1} as the inverse map. Our model does not depend on the choice of such a map.[3] Let $\mathcal{S}_d = \left\{ \boldsymbol{v} \in \mathbb{R}_+^d \;\middle|\; \sum_{i=1}^{d} v_i = 1 \right\}$ be the d-dimensional probability simplex

Algorithm 6: Generative Process with Vectorization

for $x = 1, \cdots, d_0$ **do**
  Draw core tensor $\tilde{\boldsymbol{\phi}}_x \sim \text{Dir}(\boldsymbol{\alpha}) \in \mathcal{S}_K$
  for $\boldsymbol{k} \in \mathcal{K}$ **do**
    $\phi_{x\boldsymbol{k}} = \tilde{\phi}_{x\text{vec}(\boldsymbol{k})}$
for $j = 1, \cdots, p$ **do**
  for $k = 1, \cdots, K_j$ **do**
    Draw auxiliary matrices $\boldsymbol{\psi}_k^{(j)} \sim \text{Dir}\left(\boldsymbol{\beta}^{(j)}\right) \in \mathcal{S}_{d_j}$
for $x = 1, \cdots, d_0$ **do**
  for $i = 1, \cdots, \lambda_x$ **do**
    Draw latent topics $\varepsilon \sim \text{Mult}\left(\{1, \cdots, K\}, \tilde{\boldsymbol{\phi}}_x\right)$
    $\boldsymbol{z}_i^{(x)} = \text{vec}^{-1}(\varepsilon)$
    for $j = 1, \cdots, p$ **do**
      Draw features $y_{ij}^{(x)} \sim \text{Mult}\left(\{1, \cdots, d_j\}, \psi_{z_{ij}^{(x)}}^{(j)}\right)$

Algorithm 7: Non-Collapsed Bayesian Tucker Decomposition Gibbs Sampler

for $x = 1, \cdots, d_0$ **do**
  Draw core tensor $\tilde{\boldsymbol{\phi}}_x \sim \text{Dir}(\boldsymbol{\alpha} + \boldsymbol{n}_x) \in \mathcal{S}_K$
  for $\boldsymbol{k} \in \mathcal{K}$ **do**
    $\phi_{x\boldsymbol{k}} = \tilde{\phi}_{x\text{vec}(\boldsymbol{k})}$
for $j = 1, \cdots, p$ **do**
  for $k = 1, \cdots, K_j$ **do**
    Draw auxiliary matrices $\boldsymbol{\psi}_k^{(j)} \sim \text{Dir}\left(\boldsymbol{\beta}^{(j)} + \boldsymbol{m}_h^{(j)}\right) \in \mathcal{S}_{d_j}$
for $x = 1, \cdots, d_0$ **do**
  for $i = 1, \cdots, \lambda_x$ **do**
    Compute topic probabilities $P\left(\boldsymbol{z}_i^{(x)} = \boldsymbol{k} | -\right) \propto \phi_{x\boldsymbol{k}} \psi_{ky_i}^{(j)}$
    Draw $\boldsymbol{z}_i^{(x)}$ from $P\left(\boldsymbol{z}_i^{(x)} = \boldsymbol{k} | -\right)$

C. Non-Collapsed Algorithms

See Algorithm 7.

D. Collapsed Gibbs Derivation

Here, we derive equations for collapsed Gibbs sampling of a conditional Bayesian Tucker decomposition. This derivation is similar to that of LDA. We begin with the total probability of our model and integrate out $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$:

$$
\begin{aligned}
P(\boldsymbol{Y}, \boldsymbol{Z} | \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \int_{\boldsymbol{\phi}} \int_{\boldsymbol{\psi}} P(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{\phi}, \boldsymbol{\psi} | \boldsymbol{\alpha}, \boldsymbol{\beta}) \; d\boldsymbol{\psi} \; d\boldsymbol{\phi} \\
&= \int_{\boldsymbol{\psi}} \prod_{j=1}^{p} \prod_{h=1}^{K_j} P\left(\boldsymbol{\psi}_h^{(j)} | \boldsymbol{\beta}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{y}_{xi} | \boldsymbol{\psi}_{\boldsymbol{z}_{xi}}\right) d\boldsymbol{\psi} \\
&\quad \times \int_{\boldsymbol{\phi}} \prod_{x=1}^{d_0} P\left(\boldsymbol{\phi}_x | \boldsymbol{\alpha}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi}.
\end{aligned}
$$

[3] An example mapping is $\text{vec}(\boldsymbol{k}) = k_1 + (k_2 - 1)K_1 + \cdots + (k_p - 1)\prod_{i=1}^{p-1} K_i$. This is a generalization of the column-major-order map.

Algorithm 8: Non-Collapsed PAM-Based Algorithm

for $\ell = 1, \cdots, L$ **do**
  for $j = 1, \cdots, p$ **do**
    if $\ell \neq 1$ *or* j *is not a root mode* **then**
      for $\boldsymbol{i} \in \{$*possible parent topics*$\}$ **do**
        $\boldsymbol{P}_{\boldsymbol{i}}^{(\ell,j)} \sim \text{Dir}\left(\boldsymbol{\gamma}_{\boldsymbol{i}}^{(\ell,j)} + \boldsymbol{n}_{\boldsymbol{i}}^{(\ell,j)}\right)$
      for $x = 1, \cdots, d_0$ **do**
        $P\left(\boldsymbol{Y}_x^{(j)} | \boldsymbol{Y}_{-x}^{(j)}, \boldsymbol{c}^{(j)}, \boldsymbol{Z}^{(j)}\right) = \frac{\Gamma\left(m_{k,(\cdot)}^{(j),-x} + d_j\beta^{(j)}\right)}{\prod_y \Gamma\left(m_{k,y}^{(j),-x} + \beta^{(j)}\right)} \frac{\prod_y \Gamma\left(m_{k,y}^{(j)} + \beta^{(j)}\right)}{\Gamma\left(m_{k,(\cdot)}^{(j)} + d_j\beta^{(j)}\right)}$
        $P\left(c_{x,\ell}^{(j)} = k | \boldsymbol{Y}^{(j)}, \boldsymbol{c}_{-x,\ell}^{(j)}, \boldsymbol{Z}^{(j)}\right) \propto P\left(\boldsymbol{Y}_x^{(j)} | \boldsymbol{Y}_{-x}^{(j)}, \boldsymbol{c}_\ell^{(j)}, \boldsymbol{Z}^{(j)}\right) P_{ik}^{(\ell,j)}$

All $\boldsymbol{\psi}$'s and $\boldsymbol{\phi}$'s are independent of each other and thus can be treated separately. We first examine the $\boldsymbol{\phi}$'s:

$$\begin{aligned}
&\int_{\boldsymbol{\phi}} \prod_{x=1}^{d_0} P\left(\boldsymbol{\phi}_x | \boldsymbol{\alpha}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi} \\
&= \prod_{x=1}^{d_0} \int_{\boldsymbol{\phi}_x} P\left(\boldsymbol{\phi}_x | \boldsymbol{\alpha}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi}_x.
\end{aligned}$$

Now, we look at a single $\boldsymbol{\phi}$:

$$\begin{aligned}
&\int_{\boldsymbol{\phi}_x} P\left(\boldsymbol{\phi}_x | \boldsymbol{\alpha}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi}_x \\
&= \int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{\alpha_k - 1} \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi}_x.
\end{aligned}$$

Letting n_x^k denote the count of topic(s) k given independent variable x, we can express

$$\prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) = \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k}.$$

Thus, the $\boldsymbol{\phi}_x$ integral can be rewritten as

$$\begin{aligned}
&\int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{\alpha_k - 1} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k} d\boldsymbol{\phi}_x \\
&= \int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k + \alpha_k - 1} d\boldsymbol{\phi}_x.
\end{aligned}$$

According to the functional expression of the Dirichlet distribution,

$$\int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} n_x^k + \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma\left(n_x^k + \alpha_k\right)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k + \alpha_k - 1} d\boldsymbol{\phi}_x = 1.$$

We apply this equation to get rid of the integral, resulting in a fraction made up of products of Gamma functions,

$$\begin{aligned}
&\int_{\boldsymbol{\phi}_x} P\left(\boldsymbol{\phi}_x | \boldsymbol{\alpha}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{z}_{xi} | \boldsymbol{\phi}_x\right) d\boldsymbol{\phi}_x \\
&= \int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k + \alpha_k - 1} d\boldsymbol{\phi}_x \\
&= \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \frac{\prod_{k=1}^{K_1 \cdots K_j} \Gamma\left(n_x^k + \alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} n_x^k + \alpha_k\right)} \\
&\quad \times \int_{\boldsymbol{\phi}_x} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} n_x^k + \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(n_x^k + \alpha_k)} \prod_{k=1}^{K_1 \cdots K_j} \phi_{xk}^{n_x^k + \alpha_k - 1} d\boldsymbol{\phi}_x \\
&= \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \frac{\prod_{k=1}^{K_1 \cdots K_j} \Gamma\left(n_x^k + \alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} n_x^k + \alpha_k\right)}.
\end{aligned}$$

Similarly, we derive the $\boldsymbol{\psi}$ part, letting $m_{hy}^{(j)}$ denote the count of dependent variable y in the j^{th} mode given topic h:

$$\begin{aligned}
&\int_{\boldsymbol{\psi}} \prod_{j=1}^{p} \prod_{h=1}^{K_j} P\left(\boldsymbol{\psi}_h^{(j)} | \boldsymbol{\beta}\right) \prod_{i=1}^{\lambda_x} P\left(\boldsymbol{y}_{xi} | \boldsymbol{\psi}_{\boldsymbol{z}_{xi}}\right) d\boldsymbol{\psi} \\
&= \prod_{j=1}^{p} \prod_{h=1}^{K_j} \int_{\boldsymbol{\psi}_h^{(j)}} P\left(\boldsymbol{\psi}_h^{(j)} | \boldsymbol{\beta}\right) \prod_{i=1}^{\lambda_x} P\left(y_{xi}^{(j)} | \boldsymbol{\psi}_{z_{xi}^{(j)}}^{(j)}\right) d\boldsymbol{\psi}_h^{(j)} \\
&= \prod_{j=1}^{p} \prod_{h=1}^{K_j} \int_{\boldsymbol{\psi}_h^{(j)}} \frac{\Gamma\left(\sum_{y=1}^{d_j} \beta_y^{(j)}\right)}{\prod_{y=1}^{d_j} \Gamma(\beta_y^{(j)})} \prod_{y=1}^{d_j} \left(\psi_{z_{xi}^{(j)} y}^{(j)}\right)^{\beta_y^{(j)} - 1} \\
&\quad \times \prod_{y=1}^{d_j} \left(\psi_{z_{xi}^{(j)} y}^{(j)}\right)^{m_{hy}^{(j)}} d\boldsymbol{\psi}_h^{(j)} \\
&= \prod_{j=1}^{p} \prod_{h=1}^{K_j} \int_{\boldsymbol{\psi}_h^{(j)}} \frac{\Gamma\left(\sum_{y=1}^{d_j} \beta_y^{(j)}\right)}{\prod_{y=1}^{d_j} \Gamma(\beta_y^{(j)})} \prod_{y=1}^{d_j} \left(\psi_{z_{xi}^{(j)} y}^{(j)}\right)^{m_{hy}^{(j)} + \beta_y^{(j)} - 1} d\boldsymbol{\psi}_h^{(j)} \\
&= \prod_{j=1}^{p} \prod_{h=1}^{K_j} \frac{\Gamma\left(\sum_{y=1}^{d_j} \beta_y^{(j)}\right)}{\prod_{y=1}^{d_j} \Gamma(\beta_y^{(j)})} \frac{\prod_{y=1}^{d_j} \Gamma\left(m_{hy}^{(j)} + \beta_y^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j)} + \beta_y^{(j)}\right)}.
\end{aligned}$$

By combining the expressions from the $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ parts, we obtain,

$$\begin{aligned}
P(\boldsymbol{Y}, \boldsymbol{Z} | \boldsymbol{\alpha}, \boldsymbol{\beta}) = &\prod_{x=1}^{d_0} \frac{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} \alpha_k\right)}{\prod_{k=1}^{K_1 \cdots K_j} \Gamma(\alpha_k)} \frac{\prod_{k=1}^{K_1 \cdots K_j} \Gamma\left(n_x^k + \alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1 \cdots K_j} n_x^k + \alpha_k\right)} \\
&\times \prod_{j=1}^{p} \prod_{h=1}^{K_j} \frac{\Gamma\left(\sum_{y=1}^{d_j} \beta_y^{(j)}\right)}{\prod_{y=1}^{d_j} \Gamma(\beta_y)} \frac{\prod_{y=1}^{d_j} \Gamma\left(m_{hy}^{(j)} + \beta_y\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j)} + \beta_y^{(j)}\right)}.
\end{aligned}$$

Next, we need to derive an expression for the probability

distribution of $\boldsymbol{z}_i^{(c)}$, which denotes the hidden variable(s) for the i^{th} count in $x = c$, where $\boldsymbol{y} = \boldsymbol{v}$. Let a superscript $-ci$ denote the count, excluding the i^{th} count in $x = c$. By Bayes' Theorem,

$$P(\boldsymbol{z}_i^{(c)}|\boldsymbol{Z}^{-ci}, \boldsymbol{Y}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{P(\boldsymbol{z}_i^{(x)}, \boldsymbol{Z}^{-ci}, \boldsymbol{Y}|\boldsymbol{\alpha}, \boldsymbol{\beta})}{P(\boldsymbol{Z}^{-ci}, \boldsymbol{Y}|\boldsymbol{\alpha}, \boldsymbol{\beta})}.$$

By applying this and dropping the denominator, we can express the probability distribution of $\boldsymbol{z}_i^{(c)}$ as proportional to the expression we derived above,

$$\begin{aligned}
&P(\boldsymbol{z}_i^{(c)} = \boldsymbol{k}|\boldsymbol{Z}^{-ci}, \boldsymbol{Y}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto P(\boldsymbol{z}_i^{(c)} = \boldsymbol{k}, \boldsymbol{Z}^{-ci}, \boldsymbol{Y}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&\propto \left(\frac{\Gamma\left(\sum_{k=1}^{K_1\cdots K_j}\alpha_k\right)}{\prod_{k=1}^{K_1\cdots K_j}\Gamma(\alpha_k)}\right)^{d_0} \prod_{x\neq c}\frac{\prod_{k=1}^{K_1\cdots K_j}\Gamma\left(n_x^k+\alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1\cdots K_j} n_x^k+\alpha_k\right)} \\
&\times \prod_{j=1}^{p}\left(\frac{\Gamma\left(\sum_{y=1}^{d_j}\beta_y^{(j)}\right)}{\prod_{y=1}^{d_j}\Gamma(\beta_y^{(j)})}\right)^{K_j} \prod_{h=1}^{K_j}\prod_{y\neq v_j}\Gamma\left(m_{hy}^{(j)}+\beta_y^{(j)}\right) \\
&\times \frac{\prod_{k=1}^{K_1\cdots K_j}\Gamma\left(n_c^k+\alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1\cdots K_j} n_c^k+\alpha_k\right)} \prod_{j=1}^{p}\prod_{h=1}^{K_j}\frac{\Gamma\left(m_{hv_j}^{(j)}+\beta_{v_j}^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j)}+\beta_y^{(j)}\right)}.
\end{aligned}$$

We simplify this expression by dropping multiplicative constants,

$$\begin{aligned}
&\propto \frac{\prod_{k=1}^{K_1\cdots K_j}\Gamma\left(n_c^k+\alpha_k\right)}{\Gamma\left(\sum_{k=1}^{K_1\cdots K_j} n_c^k+\alpha_k\right)} \prod_{j=1}^{p}\prod_{h=1}^{K_j}\frac{\Gamma\left(m_{hv_j}^{(j)}+\beta_{v_j}^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j)}+\beta_y^{(j)}\right)} \\
&\propto \prod_{h=1}^{K_1\cdots K_j}\Gamma\left(n_c^h+\alpha_h\right)\prod_{j=1}^{p}\prod_{h=1}^{K_j}\frac{\Gamma\left(m_{hv_j}^{(j)}+\beta_{v_j}^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j)}+\beta_y^{(j)}\right)}.
\end{aligned}$$

We now split this expression to obtain an h-independent summation, which can be dropped,

$$\begin{aligned}
&\propto \prod_{h\neq\boldsymbol{k}}\Gamma\left(n_c^{h,-ci}+\alpha_h\right)\prod_{j=1}^{p}\prod_{h\neq k_j}\frac{\Gamma\left(m_{hv_j}^{(j),-ci}+\beta_{v_j}^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{hy}^{(j),-ci}+\beta_y^{(j)}\right)} \\
&\quad\times\Gamma\left(n_c^{\boldsymbol{k},-ci}+\alpha_{\boldsymbol{k}}+1\right)\prod_{j=1}^{p}\frac{\Gamma\left(m_{k_jv_j}^{(j),-ci}+\beta_{v_j}^{(j)}+1\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{k_jy}^{(j),-ci}+\beta_y^{(j)}+1\right)} \\
&\propto\Gamma\left(n_c^{\boldsymbol{k},-ci}+\alpha_{\boldsymbol{k}}\right)\prod_{j=1}^{p}\frac{\Gamma\left(m_{k_jv_j}^{(j),-ci}+\beta_{v_j}^{(j)}\right)}{\Gamma\left(\sum_{y=1}^{d_j} m_{k_jy}^{(j),-ci}+\beta_y^{(j)}\right)} \\
&\quad\times\left(n_c^{\boldsymbol{k},-ci}+\alpha_{\boldsymbol{k}}\right)\prod_{j=1}^{p}\frac{m_{k_jv_j}^{(j),-ci}+\beta_{v_j}^{(j)}}{\sum_{y=1}^{d_j} m_{k_jy}^{(j),-ci}+\beta_y^{(j)}} \\
&\propto\left(n_c^{\boldsymbol{k},-ci}+\alpha_{\boldsymbol{k}}\right)\prod_{j=1}^{p}\frac{m_{k_jv_j}^{(j),-ci}+\beta_{v_j}^{(j)}}{\sum_{y=1}^{d_j} m_{k_jy}^{(j),-ci}+\beta_y^{(j)}}.
\end{aligned}$$

E. Generalizations to $p \geq 3$

For the independent topic model, we generalize to p independent hierarchical models (such as trees), where topic tuples $\boldsymbol{T}_x$ are all possible combinations of $\boldsymbol{T}_x^{(1)}, \cdots$, and $\boldsymbol{T}_x^{(p)}$. For the hierarchical model, with general p, it is possible to have more complex dependence relations between modes. For example, Figure 6(a) shows a model where modes 2 and 3 depend on mode 1 but do not directly depend on each other. To this end, we assume we are given a DAG representing the dependency structure of the p modes. At each level, each customer moves through the DAG and selects a topic in each mode (ordered topologically) based on probability $\boldsymbol{P}_{\boldsymbol{i}}^{(\ell,m)}$, where ℓ is the level, m is the mode, and $\boldsymbol{i}$ represents the topics of all parent modes. Similarly to the $p = 2$ case, the tuples of topics visited at each level of the DAG or all possible tuples of elements in the topic lists for each mode correspond to the topic list for each x (those topics with non-zero prevalence). Furthermore, it is possible to have a mixture of independent and hierarchical topic models; a specific mode or set of modes could be independent of the other modes. For example, Figure 6(b) presents a model where mode 1 is independent of modes 2 and 3, while mode 3 depends on mode 2.

(a) Hierarchical Model (b) Mixed Model

Fig. 6. General p Topic Model Examples ($p = 3$)

F. Derivatives of Log-Likelihood

We define Log-Likelihood LL as the log of our model probability, found in Section III-C, (6).

$$\begin{aligned}
LL &= \log P(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{\phi}, \boldsymbol{\psi}, \boldsymbol{T}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\
&= \log P(\boldsymbol{\psi}|\boldsymbol{\beta}) + \log P(\boldsymbol{\phi}, \boldsymbol{T}|\boldsymbol{\alpha}, \boldsymbol{\gamma}) + \log P(\boldsymbol{Y}, \boldsymbol{Z}|\boldsymbol{\phi}, \boldsymbol{\psi})
\end{aligned}$$

First, we derive the derivative with respect to α_i:

$$\begin{aligned}
\frac{\partial LL}{\partial\alpha_i} &= \frac{\partial}{\partial\alpha_i}\log P(\boldsymbol{\phi}, \boldsymbol{T}|\boldsymbol{\alpha}, \boldsymbol{\gamma}) \\
&= \frac{\partial}{\partial\alpha_i}\log\prod_{x=1}^{d_0} P\left(\boldsymbol{\phi}_x|\boldsymbol{\alpha}, \boldsymbol{T}_x\right) P\left(\boldsymbol{T}_x|\boldsymbol{\gamma}\right) \\
&= \frac{\partial}{\partial\alpha_i}\sum_{x=1}^{d_0}\log P\left(\boldsymbol{\phi}_x|\boldsymbol{\alpha}, \boldsymbol{T}_x\right) \\
&= \sum_{x=1}^{d_0}\frac{1}{P\left(\boldsymbol{\phi}_x|\boldsymbol{\alpha}, \boldsymbol{T}_x\right)}\frac{\partial P\left(\boldsymbol{\phi}_x|\boldsymbol{\alpha}, \boldsymbol{T}_x\right)}{\partial\alpha_i} \\
&= \sum_{x=1}^{d_0}\frac{B(\boldsymbol{\alpha})}{\prod_{j=1}^{\boldsymbol{K}}\phi_{xj}^{\alpha_j-1}}\frac{\partial}{\partial\alpha_i}\left[\frac{1}{B(\boldsymbol{\alpha})}\prod_{j=1}^{\boldsymbol{K}}\phi_{xj}^{\alpha_j-1}\right] \\
&= \sum_{x=1}^{d_0}\left[B(\boldsymbol{\alpha})\frac{\partial}{\partial\alpha_i}\frac{1}{B(\boldsymbol{\alpha})} + \frac{\alpha_i-1}{\phi_{xi}}\right] \\
&= \sum_{x=1}^{d_0}\left[\Psi^{(0)}\left(\sum_{j=1}^{\boldsymbol{K}}\alpha_j\right) - \Psi^{(0)}(\alpha_i) + \frac{\alpha_i-1}{\phi_{xi}}\right],
\end{aligned}$$

where B is the Beta function and $\Psi^{(0)}$ is the Polygamma function.

Next, we similarly derive the derivative with respect to $\beta_i^{(j)}$:

$$\frac{\partial LL}{\partial \beta_i^{(j)}} = \frac{\partial}{\partial \beta_i^{(j)}} \log P(\boldsymbol{\psi}|\boldsymbol{\beta}) = \sum_{h_j=1}^{K_j} \log P\left(\boldsymbol{\psi}_{h_j}^{(j)}|\boldsymbol{\beta}^{(j)}\right)$$
$$= \sum_{h_j=1}^{K_j} \left[\Psi^{(0)}\left(\sum_{y=1}^{d_j} \beta_y^{(j)}\right) - \Psi^{(0)}(\beta_i^{(j)}) + \frac{\beta_i^{(j)} - 1}{\psi_{h_j i}^{(j)}}\right]$$

G. Conditions

Now, we determine under which conditions the loose and strong partition properties hold in multi-modal hierarchical models. While we sketch our proofs here, detailed proofs are in Appendix H. We begin the case where the loose partition property holds in the PAM-based model ($p = 2$).

Theorem A.1. *The loose partition property holds in the PAM model ($p = 2$) if and only if the Dirichlet distributions have symmetric priors.*

In the proof, we consider a single Dirichlet distribution and show that a uniform prior is sufficient for the loose partition property to hold. Then, we use a non-uniform example to show that uniformity is necessary for this property to hold. Lastly, we argue that the theorem holds for the entire PAM model if and only if it holds for every node in the model.

Next, we prove that there does not exist a hierarchical topic model ($p = 2$) for which all three properties apply. First, we show that the probability of a new customer assigning to topic in each mode has a functional form.

Lemma A.1. *If there exists a hierarchical topic model ($p = 2$) where the rich-get-richer, strong partition, and exchangeability properties hold, then $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$ are of the form: $\xi_i \propto \begin{cases} f\left(\rho_{i(\cdot)}\right), & \rho_{i(\cdot)} > 0 \\ \gamma_0(K_1), & \rho_{i(\cdot)} = 0 \end{cases}$ and $\theta_{ij} \propto \begin{cases} g_i(\rho_{ij}), & \rho_{ij} > 0 \\ \gamma_i(K_2), & \rho_{ij} = 0 \end{cases}$, for some functions f and g_i ($\forall\ i$).*

We show that this formulation is sufficient to satisfy the strong partition property and necessary to satisfy the rich-get-richer and exchangeability properties.

Next, we show that these functions must be linear.

Lemma A.2. *Given the assumptions and results of Lemma A.1, f and g_i (for all i) are linear.*

We use the formula from Lemma A.1 to compute the probabilities for two different orderings. Given the exchangeability property, these formulations must be equal. From this we conclude that f must be linear.

Finally, we use the linear form to show that such a model does not exist.

Theorem A.2. *There does not exist a hierarchical topic model ($p = 2$) where the rich-get-richer, strong partition, and exchangeability properties all hold.*

We use an example in Appendix H to show that the formulation in Lemma A.2 violates the strict partition property.

However, the independent trees model allows all three properties. This follows from the properties of each independent CRP.

Theorem A.3. *The exchangeability, loose partition, and rich-get-richer properties hold for the independent trees model.*

This conclusion rationalizes our choice of model with the loose partition property holding rather than the strong version.

H. Conditions Proofs

First, we prove Theorem A.1, showing that the parameters of the Dirichlet distributions in our PAM model are symmetric.

Proof. First, we look at a single Dirichlet distribution, i.e., a single-node PAM model, and show that the partition property applies if and only if the parameters are symmetric. Without loss of generality, assume the parameters of the Dirichlet distribution $\boldsymbol{\gamma} = (\gamma_1, \cdots, \gamma_K)$ are ordered such that $\gamma_1 \leq \cdots \leq \gamma_K$, where K is the number of topics. Also, by PAM $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_K) \sim \text{Dir}(\boldsymbol{\gamma})$.

First, we show sufficiency. To this end, assume $\gamma_p = \gamma_q := \gamma$ for any $p,\ q$. Here, both the probability distribution and expectation of all θ's are equal (this is clear from examining the probability distribution function of the Dirichlet distribution). We denote the probability of m_p people to topic p and m_q people to topic q, along with our topics assignment, as $P(m_p, m_q, -)$. Thus this probability,

$$P(m_p, m_q, -) \propto \int_0^1 \int_0^1 P(m_p, m_q|\theta_p, \theta_q) P(\theta_p, \theta_q) d\theta_p d\theta_q$$
$$\propto \int_0^1 \int_0^1 \theta_p^{m_p+\gamma-1} \theta_q^{m_q+\gamma-1} d\theta_p d\theta_q$$
$$= \int_0^1 \theta^{m_p+\gamma-1} d\theta \int_0^1 \theta^{m_q+\gamma-1} d\theta,$$

is equal to the probability of assigning m_q people to topic p and m_p people to topic q,

$$P(m_q, m_p, -) \propto \int_0^1 \theta^{m_p+\gamma-1} d\theta \int_0^1 \theta^{m_q+\gamma-1} d\theta.$$

Similarly, since the probability distributions over all θ's are the same, the probability of assigning m people to topics 1 through K is equal to the probability of assigning any permutation of m people to topics 1 through k. Thus the partition property holds.

Next, we show necessity. To this end, assume $\gamma_p < \gamma_q$ (for some p and q). The probability of assigning m_p people to topic p and m_q people to topic q,

$$P(m_p, m_q, -) \propto \int_0^1 \int_0^1 \theta_p^{m_p+\gamma_p-1} \theta_q^{m_q+\gamma_q-1} d\theta_p d\theta_q$$
$$= \frac{1}{(m_p + \gamma_p)(m_q + \gamma_q)},$$

is not equal to the probability of assigning m_q people to topic p and m_q people to topic q,

$$P(m_q, m_p, -) \propto \frac{1}{(m_p + \gamma_q)(m_q + \gamma_p)},$$

for all m_p and m_q. If $m_p < m_q$, then elementary algebra shows $(m_p + \gamma_p)(m_q + \gamma_q) < (m_p + \gamma_q)(m_q + \gamma_p)$ and if $m_p > m_q$, then $(m_p + \gamma_p)(m_q + \gamma_q) > (m_p + \gamma_q)(m_q + \gamma_p)$. Thus, the partition property does not hold.

If and only if the partition property holds for single nodes of the PAM model, it is possible to re-arrange topics in both modes (i.e., the loose partition property applies). □

Next, we prove Lemma A.1, showing that for the given properties to hold, then $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$ must be of the given form.

Proof. For such a model, $\xi_i = \xi_j$ if and only if $\rho_{i(\cdot)} = \rho_{j(\cdot)}$ and $\theta_{ki} = \theta_{kj}$ if and only if $\rho_{ki} = \rho_{kj}$. This is sufficient because of the strong partition property and necessary because of the rich-get-richer property. By the strong partition property and the chain rule, $\xi_i \theta_{ij} = \xi_k \theta_{kl}$ if $\rho_{ij} = \rho_{kl}$. Also, because of the rich-get-richer and exchangeability properties, we can express $\xi_i \propto \begin{cases} f\left(\rho_{i(\cdot)}\right), & \rho_{i(\cdot)} > 0 \\ \gamma_0(K_1), & \rho_{i(\cdot)} = 0 \end{cases}$ and $\theta_{ij} \propto \begin{cases} g_i(\rho_{ij}), & \rho_{ij} > 0 \\ \gamma_i(K_2), & \rho_{ij} = 0 \end{cases}$. □

Then, we prove Lemma A.2, showing that the functions must be linear.

Proof. Suppose we want to assign x people to topic 1 and one person to topic 2. One way (case one) to do this would be to assign all x people to topic 1, then one person to topic 2. Another way (case two) to do this would be to assign $x-1$ people to topic 1, then one person to topic 2, then one more customer to topic 1. The probabilities of these cases can be expressed as:

$$P(\text{case one}) = \frac{f(1)}{\gamma_0(1) + f(1)} \cdots \frac{f(x-2)}{\gamma_0(1) + f(x-2)} \frac{f(x-1)}{\gamma_0(1) + f(x-1)} \frac{\gamma_0(1)}{\gamma_0(1) + f(x)}$$

$$P(\text{case two}) = \frac{f(1)}{\gamma_0(1) + f(1)} \cdots \frac{f(x-2)}{\gamma_0(1) + f(x-2)} \frac{\gamma_0(1)}{\gamma_0(1) + f(x-1)} \frac{f(x-1)}{\gamma_0(2) + f(x-1) + f(1)}$$

Thus, the differences in the probabilities are that the first case has $\gamma_0(1) + f(x)$ in the last denominator while the second case has $\gamma_0(2) + f(x-1) + f(1)$ in the last denominator. If the exchangeability property applies, we have $P(\text{case one}) = P(\text{case two})$ and in turn $\gamma_0(1) + f(x) = \gamma_0(2) + f(x-1) + f(1)$. Since this must apply for all x, by induction we have $f(x) = xf(1) + \gamma_0(2) - \gamma_0(1)$. Thus, f is linear.

In general, $\xi_i \propto \begin{cases} \rho_{i(\cdot)} - \gamma_{02}, & \rho_{i(\cdot)} > 0 \\ \gamma_{01} + \gamma_{02} K_1, & \rho_{i(\cdot)} = 0 \end{cases}$ and $\theta_{ij} \propto \begin{cases} \rho_{ij} - \gamma_{i2}, & \rho_{ij} > 0 \\ \gamma_{i1} + \gamma_{i2} K_2, & \rho_{ij} = 0 \end{cases}$. Note that this is a generalized nCRP. □

Finally, we prove Theorem A.2, showing that such a model does not exist.

Proof. Given the forms of $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$ from Lemma A.2 and suppose we swap two elements $(i,j) \neq (m,n),\ i \neq m$ within $\boldsymbol{\rho}$. Then the ratio ν of the original probability with the swapped probability is:

$$\omega(\rho_{ij}, \rho_{mn}) := \frac{\Gamma(\rho_{m(\cdot)} - \gamma_{02})}{\Gamma(\rho_{m(\cdot)} - \rho_{mn} + \rho_{ij} - \gamma_{02})} \times \frac{\Gamma(\rho_{i(\cdot)} - \gamma_{02})}{\Gamma(\rho_{i(\cdot)} - \rho_{ij} + \rho_{mn} - \gamma_{02})}$$

$$\nu = \omega(\rho_{ij}, \rho_{mn}) \frac{\Gamma(\rho_{ij} - \gamma_{i2}) \Gamma(\rho_{mn} - \gamma_{m2})}{\Gamma(\rho_{ij} - \gamma_{m2}) \Gamma(\rho_{mn} - \gamma_{i2})}.$$

If the strict partition property is satisfied, then $\nu = 1$ for all $\rho_{ij},\ \rho_{mn},\ \rho_{i(\cdot)}$, and $\rho_{m(\cdot)}$. Note that $\frac{\Gamma(\rho_{ij} - \gamma_{i2}) \Gamma(\rho_{mn} - \gamma_{m2})}{\Gamma(\rho_{ij} - \gamma_{m2}) \Gamma(\rho_{mn} - \gamma_{i2})} = 1$ for all ρ_{ij} and ρ_{mn} if and only if $\gamma_{i2} = \gamma_{m2}$, however $\omega(\rho_{ij}, \rho_{mn}) \neq 1$ for all $\rho_{ij},\ \rho_{mn},\ \rho_{i(\cdot)}$, and $\rho_{m(\cdot)}$. For example, if $\rho_{ij} = 1,\ \rho_{mn} = 2,\ \rho_{i(\cdot)} = 3,\ \rho_{m(\cdot)} = 3$, and $\gamma_{02} = 0$, then $\omega(\rho_{ij}, \rho_{mn}) = \frac{2!2!}{1!3!} = \frac{2}{3} \neq 1$. This shows that $\nu = 1$ is not always possible. □

I. Model Variations

In our efforts to boost the performance of our models, we implemented and developed several variations and modifications to our original methodology, including:

- **Different seeds:** Train the model using ten different random number generator seeds. Pick the model with the highest log-likelihood or coherence.
- **Keep best:** Check the log-likelihood or coherence during training every ten iterates. Save the best model rather than the final one. Another variation of this method is resetting to the current best model after each check.
- **MAP estimate:** When using the collapsed sampler, use the maximum a posteriori (MAP) estimate of the auxiliary matrices and core tensors rather than performing one non-collapsed iterate.
- **Adjust the number of counts:** For example, double the counts or set all non-zero counts to one.
- **Sparse cutoff:** Set a cutoff value below which all proportional probabilities are set to zero.
- **Initialization:** Initialize the auxiliary matrices with those trained using a different method.
- **Ratio of topic model iterates to Bayesian Tucker:** Do two Bayesian Tucker sample iterations for every draw from the hierarchical topic model, or vice-versa.

- **Set a topics goal:** Pick an ideal number of topics. Adjust γ during training to reach that goal. We used the formula:

$$\gamma_{\text{new}} = \gamma_{\text{old}} \max\left\{\min\left\{\left(\frac{\text{topics goal}}{\text{\# of topics}}\right)^{\prod_{i=1}^{p}\frac{1}{L_i-1}}, 2\right\}, \frac{1}{2}\right\}.$$

- **Exponential weighting:** Apply an exponential weight to the relative probabilities in the Collapsed Gibbs Sampler (Algorithm 3) as such:

$$P\left(\boldsymbol{z}_i^{(x)} = \boldsymbol{k}|-\right) \propto \left(n_x^{\boldsymbol{k},-xi} + \alpha_{\boldsymbol{k}}\right) \times \prod_{j=1}^{p}\left[\frac{m_{k_j y_j}^{(j),-xi} + \beta_{y_j}^{(j)}}{\sum_{y=1}^{d_j} m_{k_j y}^{(j),-xi} + \beta_y^{(j)}}\right]^{w_p},$$

  where w_p is the weight for mode p. This allows us to adjust each mode's relative variance.

For our coherence measure experiments (Section VI-B and Appendix J4), we used different seeds, keep best, MAP estimate, topics goals, and exponential weighting. We used keep best for our classification models (Appendix J5). While we tried the other methods mentioned above, they did not perform as well. We did not use the above modifications with the likelihood models (Appendix J7).

J. Experiments

The decomposition algorithm was mainly written in MATLAB, using Tensor Toolbox [45]. Sampling functions were written in C to improve run time. The preprocessing for the genetic data was done in R. The text data was processed in Python, using Gensim [46] and the Natural Language Toolkit (NLTK) [47]. The non-hierarchical CP decompositions were also done in Python, using TensorLy [48]. Our experiments were run on a 3.1 GHz Dual-Core Intel Core i5 processor with 16 GB 2133 MHz LPDDR3 memory.

TABLE II
CANCER TYPES AND TCGA DESIGNATION

Cancer Type	TCGA Designation
Breast	Breast Invasive Carcinoma (BRCA)
Lung	Lung Squamous Cell Carcinoma (LUSC) or Lung Adenocarcinoma (LUAD)
Prostate	Prostate Adenocarcinoma (PRAD)
Colorectal	Colon Adenocarcinoma (COAD) or Rectum Adenocarcinoma (READ)

1) Data Sets: The cancer data contains 3,037 patients. The Cancer Genome Atlas (TCGA) designations we considered for each type of cancer are given in Table II. This data set contained 1,044 patients with breast cancer, 1,066 patients with lung cancer, 494 patients with prostate cancer, and 433 patients with colorectal cancer. After linking with the Reactome pathways, we were left with counts of variants on 7,846 genes and 1,678 pathways. This includes single-nucleotide polymorphisms and other mutations. We do not consider gene expression data.

Once we similarly linked the ASD genetic variants to the pathways, our data set contains 3,408 patients (half diagnosed with ASD), 7,211 genetic variants, and 1,413 pathways.

The Reuters data contains 5,501 articles, 8,820 phrases, and 6,837 unique words.

We split each data set into a 30% held-out test set and performed 10-fold cross-validation (CV) on the remaining training/validation data.

2) Coherence Measure: To utilize this coherence measure in our context, we made some modifications: 1) our metric is intrinsic and does not utilize an external corpus to determine the gene or pathway probabilities and co-occurrence probabilities; and 2) we determined co-occurrence as having a variant on a pair of genes or pathways, not accounting for sequences as in many NLP examples.

3) Coherence Models: To pre-process our data, we removed genes that appeared in fewer than 200 or more than 2,000 patients and words that appeared in fewer than 200 or more than 2,000 documents, eliminating rare and common genes or words. We also removed phrases that were present in fewer than 10 articles. Words without an assigned phrase were then assigned to a single-word phrase corresponding to the given word. We modified [49]'s implementation to process the R8 data into the article, phrase, and word tensor structure. The CP TensorLy model was trained on the entire training data set and then split into folds; while the other models were split into folds before fitting the decompositions. Our coherence measures examined the top 5 genes, pathways, sentences, or words in each topic.

Bayesian cancer and ASD models had a topic goal of 500 and R8 models had a topic goal of 50 (see Appendix I), while the CP TensorLy model had 200 topics for all data sets. The hierarchical models used three levels.

The HBT models used the independent trees hierarchy. We trained PAM-based models on the Cancer data and found the models slightly under-performed the independent trees models. The best PAM-based model had ten topics per level, used Genes as a dominant mode and Cartesian topics, and had a PMI gene coherence of 16.35 and PMI pathway coherence of 9.55.

The hLDA models used a modified version of the independent trees HBT model with only two modes. The CP tree model also used a modified independent trees HBT model, with a single hierarchical tree and restricting the core tensor to be diagonal. The CP TensorLy model uses the alternating-least squares method to decompose the tensor [33].

For the R8 models with bad phrases removed, we removed words and phrases that showed up infrequently in topics.

Experimenting with several HBT models showed that the model log-likelihood increased for the first 30-40 samples, then leveled off (with some random fluctuation). Based on this, we chose to run each model for 100 iterations, checking every 10 iterations and keeping the best model. We also used 10 different random seeds.

4) Reuters Experiments: We also applied a similar structure from our genetic models to NLP. Incorporating phrases as

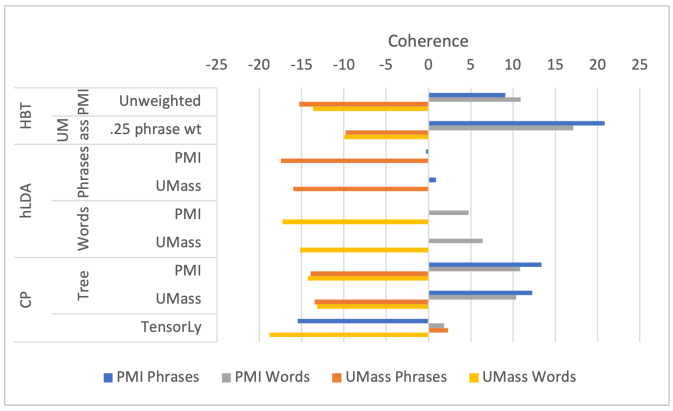

Fig. 7. Independent Trees HBT models are more coherent than comparable models on R8 data. Our best HBT models outperformed the hLDA, CP tree, and CP TensorLy baselines on three-of-four coherence measures (using the mean over ten cross-validation folds).

TABLE III
MEAN VALIDATION COHERENCE OF R8 MODELS WITH BAD PHRASES REMOVED

Model	Phrases		Words	
	PMI	UMass	PMI	UMass
HBT (PMI)	11.80	-12.91	12.38	-12.37
HBT (UMass)	**12.61**	-11.19	**13.39**	**-10.79**
CP (TensorLy)	-15.47	**-6.64**	-4.00	-20.04

another mode may improve the grouping of articles. The hierarchical structure here would be: words make up phrases, which make up articles. We looked at the eight largest classes (earnings, acquisitions, money - foreign exchange, grain, crude, trade, interest, and shipping) in the Reuters-21578 data set (denoted R8) [50]. We used SpaCy [51] to group words into phrases (noun chunks). This gives us a count of words in each phrase in each article.

While our best HBT model outperformed the baseline models on three-of-four coherence measures on the R8 data, it under-performed on UMass phrase coherence. Figure 7 shows that the HBT model using UMass coherence and .25 phrase weighting outperformed the CP model using PMI coherence, the best baseline model, on PMI phrase and word coherence by 56.07% and 57.84% respectively. This HBT model also outperformed the CP model using UMass coherence by 24.54% on UMass word coherence. However, while the CP TensorLy model performed worst on these three coherence measures, it outperformed our best model on UMass phrase coherence (2.35 to -9.79). After examining this data set further, we removed words and phrases that frequently showed up in topics and had few co-occurrences with other words in the topic. While we tried various word sets and were able to make improvements, due to the Bayesian nature of the algorithms, words with fewer co-occurrences are more likely to be placed in the same topic than in a deterministic model. Table III shows that while these changes narrowed the gap in performance on UMass phrase coherence, the CP TensorLy model still outperformed our best HBT model on this measure.

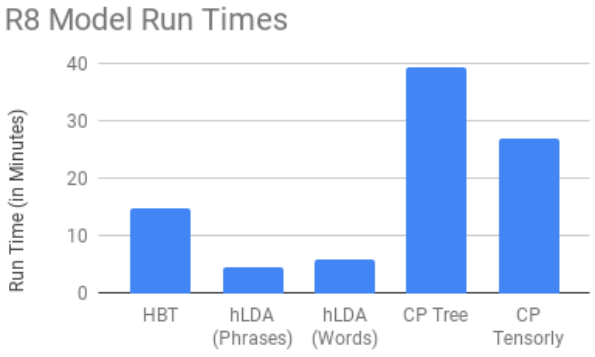

Fig. 8. Time to train a single decomposition model on R8 data.

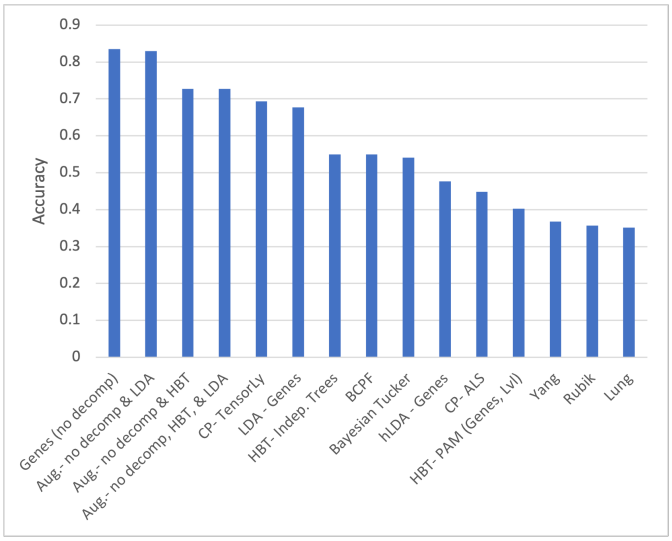

Fig. 9. Genes data (without decomposition) is more accurate than decomposition and augmented models on cancer data (using the mean over ten cross-validation folds).

Figure 8 shows that while our algorithm takes longer to train on the R8 data than the ASD and Cancer data sets, it still significantly outperforms the CP models, supporting our conclusions in Section VI-D.

5) Classification: We trained logistic regression models using patient groups from the decomposition models to predict each patient's cancer diagnosis. Figure 9 shows the accuracy of these models, and Appendix J6 details our implementation. Unfortunately, none of the models we trained, including the baselines, outperformed classifying using the genetic variant counts (i.e., not using a decomposition model), which had a mean validation accuracy of 83.49%. Augmenting gene data with our decomposition models did not improve the accuracy of our predictions. Of the decomposition models, CP TensorLy performed best (69.33% accuracy). The other CP models, [52]'s Bayesian CP Factorization (BCPF) (54.94% accuracy) and [45]'s alternating least squares (ALS) CP (44.78% accuracy) performed significantly worse. The second-best decom-

position model was LDA on genes (67.74% accuracy). On the other hand, our genes-based hLDA model had a 47.68% accuracy. Our independent trees HBT model and our non-hierarchical Bayesian Tucker model performed about the same, with accuracies of 54.98% and 54.14%, respectively. Our PAM-based HBT model, using genes as the dominant mode and the level method, performed worse (40.31% accuracy). [7]'s Bayesian Conditional Tensor factorization (36.74% accuracy) and [53]'s Rubik model (35.70% accuracy) only slightly outperformed diagnosing all patients with Lung cancer, the most prevalent class, at 35.10% prevalency.

6) Classification Models: For the CP TensorLy model, we removed genes that appeared in fewer than 200 or more than 2,000 patients and used 200 topics. For the LDA model, we modified [8]'s variational Bayes implementation using 40 topics. For our independent trees HBT model, we removed genes that appeared in fewer than 400 or more than 1,000 patients and used three levels, $\gamma = 0.1$, and our "keep best" methodology. For our Bayesian Tucker model, we removed genes that appeared in fewer than 200 or more than 2,000 patients and used 10 topics on each mode and our "keep best" methodology. For our hLDA model, we used two levels and $\gamma = 0.1$. For our CP ALS model, we used 25 topics. For our PAM-based HBT, we removed genes that appeared in fewer than 400 or more than 1,000 patients and used three levels and 10 topics per level. For the Rubik model, we used 5 topics.

7) Likelihood: One issue with comparing likelihoods between models is that the probabilities in Section III are not comparable due to the different hierarchical model structures. To compute the held-out likelihood, we would need to sum over or integrate out our hierarchical model variables, which do not have a closed-form solution.

To solve this problem, we use a non-parametric likelihood estimate, similar to [21] and based on empirical likelihood [54]. First, we randomly generate one thousand patients using the trained generative process. Then, we compute the probabilities of a held-out test or validation patients as a mixture of the generated patients. Unlike other likelihood measures, this method is stable, easy to compute, and yields values that are comparable between models.

We trained HBT decomposition models using various hierarchical models and computed the mean validation log-likelihood (over the ten CV folds, using the above methodology). Each hierarchical model was trained with varying levels $L \in \{2, 3, 4, 5\}$. Additionally, we trained the independent trees model with three CRP hyperparameters $\gamma \in \{0.5, 1, 2\}$. For the PAM-based model, we compared each choice of dominant mode (genes or pathways) and topic set composition (Cartesian or level set method; see Section III-C for definitions). We also used varying topics per level $\tau \in \{10, 25, 50\}$.

We plotted the results in Figure 10. This figure depicts the mean validation log-likelihood (over the ten CV folds) for hierarchical models trained on the cancer data set, comparing it to the total number of topics (product of the number of gene and pathway topics) for the model. In the case of the independent trees model, the number of topics is an average

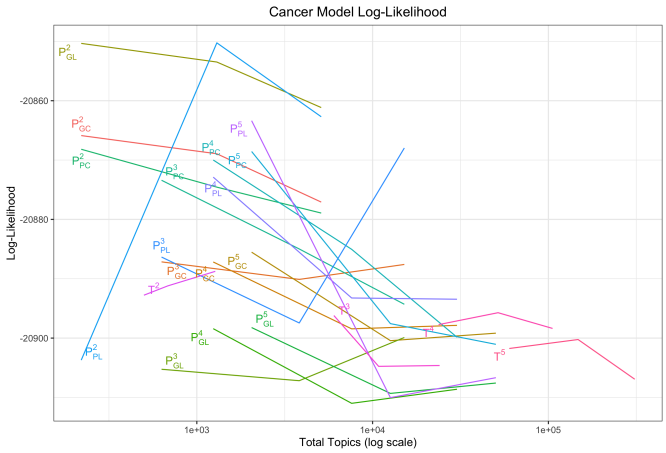

Fig. 10. Each line has a coded label: The large letter indicates if the point is from the independent trees model ("T") or the PAM-based model ("P"). The superscript indicates the number of levels in the model. The subscript (for the PAM-based models) indicates the dominant mode, genes ("G") or pathways ("P"), followed by the topic set composition method, Cartesian ("C") or level set ("L"). Every label (combination of model type, number of levels, dominant mode, and topic set composition, if applicable) has the same color.

over the CV folds. Although the CRP hyperparameter (γ) and the number of topics per level (τ) are not displayed on the plot, they dictate the number of total topics.

Figure 10 shows that the log-likelihood for models with fewer topics is often larger than those with more topics (14 of 20 lines peak at the fewest topics), indicating that some models are overfitted. We observed that some PAM-based models outperformed independent trees models, which we hoped would be the case given the inherent hierarchical structure between genes and pathways. Still, there was no clear reason for which models were better than others (other than the total number of topics). The PAM-based models with level-set topic composition are the most sensitive to changes in the other parameters (the standard deviation of the log-likelihoods of all such models with genes as the dominant mode was 23.15 and 19.86 for pathway-dominant models). The independent trees models were the least sensitive (with a standard deviation of 5.68). The PAM-based models with Cartesian topic composition were in between (with a standard deviation of 11.52 for gene-dominant models and 12.56 for pathway-dominant models). However, the differences between these models are well within the margin of error. The standard deviation in log-likelihood over the CV folds for each model is about 2,090 (or 10% the log-likelihood).

Appendix References

[40] Z. Xu, F. Yan, and A. Qi, "Infinite tucker decomposition: Nonparametric Bayesian models for multiway data analysis," in *Proceedings of the 29th International Conference on Machine Learning*, ser. ICML '07, J. Langford and J. Pineau, Eds. New York, NY, USA: ACM, 2012, pp. 1023–1030.

[41] E. C. Chi and T. G. Kolda, "On tensors, sparsity, and nonnegative factorizations," *SIAM Journal on Matrix Analysis and Applications*, vol. 33, no. 4, pp. 1272–1299, 2012.

[42] W. Hackbusch and S. Kühn, "A new scheme for the tensor representation," *Journal of Fourier Analysis and Applications*, vol. 15, no. 5, pp. 706–722, Oct 2009.

[43] L. Grasedyck, "Hierarchical singular value decomposition of tensors," *Society for Industrial and Applied Mathematics Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2029–2054, 2010.

[44] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Hierarchical Dirichlet processes," *Journal of the american statistical association*, vol. 101, no. 476, pp. 1566–1581, 2006.

[45] B. W. Bader and T. G. Kolda, "Tensor toolbox for MATLAB, version 3.2.1," Apr 2021. [Online]. Available: http://www.tensortoolbox.org/

[46] R. Rehurek and P. Sojka, "Gensim–python framework for vector space modelling," *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, vol. 3, no. 2, 2011.

[47] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. "O'Reilly Media, Inc.", 2009.

[48] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic, "Tensorly: Tensor learning in python," *Journal of Machine Learning Research (JMLR)*, vol. 20, no. 26, 2019.

[49] K. Ding, J. Wang, J. Li, D. Li, and H. Liu, "Be more with less: Hypergraph attention networks for inductive text classification," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4927–4936.

[50] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[51] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength natural language processing in python," 2020.

[52] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian CP factorization of incomplete tensors with automatic rank determination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1751–1763, 2015.

[53] Y. Wang, R. Chen, J. Ghosh, J. C. Denny, A. Kho, Y. Chen, B. A. Malin, and J. Sun, "Rubik: Knowledge guided tensor factorization and completion for health data analytics," *KDD : proceedings. International Conference on Knowledge Discovery & Data Mining*, vol. 2015, pp. 1265–1274, Aug 2015. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/31452969

[54] P. J. Diggle and R. J. Gratton, "Monte Carlo methods of inference for implicit statistical models," *Journal of the Royal Statistical Society, Series B: Methodological*, vol. 46, pp. 193–227, 1984.