A  Gibbs Sampling for bi-conv-PGDS

It is a non-trivial task to develop Gibbs sampling update equations for the bi-conv-PGDS model, mainly due to the difficult to sample the gamma shape parameters from their conditional posteriors. By exploiting related variable augmentation and marginalization techniques of Zhou et al. [11] and their generalizations into the inference for gamma Markov chains [43,51,60], we propose a bidirectional Gibbs sampler to make it simple to compute the conditional posterior of the model parameters. We repeatedly exploit the following three properties, as summarized in [43], in order to do the inference.

A.1  The notation table for our generative model

<table>
<thead>
<tr>
<th>symbol</th>
<th>meaning</th>
<th>symbol</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_j$</td>
<td>$j$th input document</td>
<td>$X_{jt}$</td>
<td>$t$th sentence of $D_j$</td>
</tr>
<tr>
<td>$W_{jt}$</td>
<td>convolutional representation of $X_{jt}$</td>
<td>$w_{jtk}$</td>
<td>$k$th convolutional representation of $W_{jt}$</td>
</tr>
<tr>
<td>$\Pi$</td>
<td>Transition matrix</td>
<td>$\bar{w}<em>{jtk}, \bar{w}</em>{jtk}$</td>
<td>forward, backward convolutional representation</td>
</tr>
<tr>
<td>$\delta_{jt}$</td>
<td>sentence-specific scaling factor</td>
<td>$v$</td>
<td>factor weight</td>
</tr>
<tr>
<td>$D_k$</td>
<td>$k$th convolutional filter</td>
<td>$M_{jt}$</td>
<td>thresholding count matrix of $X_{jt}$</td>
</tr>
<tr>
<td>$\bar{\Pi}, \bar{\Pi}$</td>
<td>forward, backward Transition matrices</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

A.2  Properties

Property 1 (P1): If $x = \sum_{n=1}^{N} x_n$, where $x_n \sim \text{Poisson}(\theta_n)$ are independent Poisson-distributed random variables, then $(x_1, \ldots, x_N) \sim \text{Multinomial}(x, \theta_1, \ldots, \theta_N / \sum_{n=1}^{N} \theta_n)$ and $x \sim \text{Poisson}\left(\sum_{n=1}^{N} \theta_n\right)$ [7,61].

Property 2 (P2): If $x \sim \text{Poisson}(c\theta)$, where $c$ is a constant, and $\theta \sim \text{Gamma}(a, 1/b)$, then $x \sim \text{NB}\left(a, 1 - \frac{1}{1 + bc}\right)$ is a negative binomial (NB) distributed random variable. We can equivalently parameterize it as $x \sim \text{NB}(a, g(\zeta))$, where $g(z) = 1 - \exp(-z)$ is the Bernoulli-Poisson link and $\zeta = \ln(1 + bc)$.

Property 3 (P3): If $x \sim \text{NB}(a, g(\zeta))$ and $l \sim \text{CRT}(x, a)$ is a Chinese restaurant table (CRT) distributed random variable, then $x$ and $l$ are equivalently jointly distributed as $x \sim \text{SumLog}(l, g(\zeta))$ and $l \sim \text{Poisson}(a\zeta)$ [11]. The sum logarithmic (SumLog) distribution is further defined as the sum of $l$ independent and identically logarithmic-distributed random variables, i.e., $x = \sum_{i=1}^{l} y_i$ and $y_i \sim \text{Logarithmic}(g(\zeta))$.

A.3  Inference

Similar to Wang et al. [20], to avoid directly process sparse document matrix, which will bring unnecessary burden in computation and storage, we apply variable augmentation under the Poisson likelihood [7,13] to upward propagate latent count matrices $M_{jt}$ as

$$
(M_{j1t}, \ldots, M_{jKt} | -) \sim \text{Multi}\left(M_{jt}: \frac{D_1 \ast w_{j1}}{\sum_{k=1}^{K} D_k \ast w_{jkt}}, \ldots, \frac{D_K \ast w_{jK}}{\sum_{k=1}^{K} D_k \ast w_{jkt}}\right)
$$

where $w_{jtk} := \bar{w}_{jtk} + \bar{w}_{jtk}$. Note only nonzero elements of $M_{jtk} \in Z^{V \times L_{jt}}$ need to be focused on. By expanding the convolutional operation along the dimension of $D_k$, we rewrite the likelihood function of $w_{jtk}$ as

$$
m_{jtkvl} \sim \text{Pois}\left(\delta_{jt} \sum_{f=1}^{F} d_{kvlf} w_{jtk(l-f+1)}\right)
$$
where $w_{jtk(l-f+1)} := 0$ if $l - f + 1 \notin \{1, 2, \ldots, S_{jt}\}$. Thus each nonzero element $m_{jtkv}$ could be augment as
\[(d_{jtkv1}, \ldots, d_{jtkvF} | -) \sim \text{Multi}(m_{jtkv}; \frac{d_{kv1}w_{jtk(l-1+1)}}{\sum_{f=1}^{F}d_{kv1}w_{jtk(l-1+1)}}, \ldots, \frac{d_{kvF}w_{jtk(l-F+1)}}{\sum_{f=1}^{F}d_{kvF}w_{jtk(l-F+1)}})\]

Assigning $d_{jtkv} = (d_{jtkv1}, \ldots, d_{jtkvF}) \in \mathbb{Z}^F$. Via P1, we have
\[
((d'_{jtkv1}, \ldots, d'_{jtkvF})' | m_{jtkv}) \sim \text{Multi} (m_{jtkv}; D_k(\cdot))
\]
where $\cdot$ denotes summing over the corresponding index and $d_{j-kv} = \sum_{f=1}^{F} d_{jtkv}$, $m_{j-kv} = \sum_{t=1}^{T_j} \sum_{v=1}^{V} m_{jtkv}$. **Sampling the convolutional filter:** With the Dirichlet-multinomial conjugacy, we have
\[(D_k(\cdot) | -) \sim \text{Dir}((d'_{j-kv1}, \ldots, d'_{j-kvF})' + \eta \mathbf{1}_F)\]

Similarly, we can expand the convolution along the dimension of $w_{jtk}$ as $m_{jtkv} \sim \text{Pois}(\sum_{s=1}^{S_j} w_{jtk}s d_{kv(l-s+1)})$, where $d_{kv(l-s+1)} := 0$ if $l - s + 1 \notin \{1, 2, \ldots, F\}$ and augment $m_{jtkv}$ as
\[(m_{jtkv1}, \ldots, m_{jtkvS_j} | -) \sim \text{Multi}\left(m_{jtkv}; \frac{w_{jtk1}d_{kv(l-1+1)}}{\sum_{s=1}^{S_j} w_{jtk1}d_{kv(l-1+1)}}, \ldots, \frac{w_{jtkS_j}d_{kv(l-S_j)+1)}}{\sum_{s=1}^{S_j} w_{jtkS_j}d_{kv(l-S_j)+1}} \right)\]

Assigning $m_{jtkv} = (m_{jtkv1}, \ldots, m_{jtkvS_j}) \in \mathbb{Z}^{S_j}$. By marginalizing out $D_k$, we can use P1 to decouple $D_k * w_{jtk}$ and get
\[m_{jtk} \sim \text{Pois}(\delta_{jt} w_{jtk})\]

Assigning $A_{jtk} = \sum_{v=1}^{V} \sum_{s=1}^{S_j} m_{jtkv}$, As $w_{jtk} = \bar{w}_{jtk} + \tilde{w}_{jtk}$, via P1, we can separate $\bar{w}_{jtk}$ and $\tilde{w}_{jtk}$ by augmenting $A_{jtk}$ as $A_{jtk} = \bar{A}_{jtk} + \tilde{A}_{jtk}$, where
\[\bar{A}_{jtk} \sim \text{Pois}(\delta_{jt} \bar{w}_{jtk}), \tilde{A}_{jtk} \sim \text{Pois}(\delta_{jt} \tilde{w}_{jtk})\]
This is the unique property of bi-conv-PGDS because of the bidirectional dynamic structure. We denote the convolutional representation of sentence $t$ in document $j$ as $\bar{W}_{jt} = \bar{W}_{jt} + \tilde{W}_{jt} = (w_{jtk1}, \ldots, w_{jtkK}) \in \mathbb{R}^{K \times S_j}$. Inspired by related development on inference for gamma Markov chains [43][51][60], we develop a backward-forward Gibbs sampling for forward feature representation $\bar{W}_{jt}$, while a forward-backward Gibbs sampling for backward feature representation $\tilde{W}_{jt}$. Take $\bar{W}_{jt}$ as an example, we should start with $\bar{W}_{jt0}$, because none of the other time-step depend on it in their priors. Via P2, we can marginalize over $\bar{W}_{jt}$ and obtain the following equation
\[\bar{A}_{jT_j:k_sT_{j-1}} \sim \text{NB}(\tau_0 \bar{\Pi}_k; \sum_{s=1}^{S_{j,T_{j-1}}} \bar{w}_{j,T_{j-1}:s}/S_{j,T_{j-1}}, g(\tilde{\zeta}_{jT_j}), \tilde{\zeta}_{jT_j} = \ln(1 + \frac{\delta_{jt}}{\tau_0})\]

where $g(\tilde{\zeta}) = 1 - \exp(-\tilde{\zeta})$. In order to marginalize over $\bar{W}_{j,T_{j}-1}$, we introduce an auxiliary variable [13]
\[(\bar{x}'_{jT_j:k_sT_{j-1}} | -) \sim \text{CRT}(\bar{A}_{jT_j:k_sT_{j-1}}; \tau_0 \bar{\Pi}_k; \sum_{s=1}^{S_{j,T_{j-1}}} \bar{w}_{j,T_{j-1}:s}/S_{j,T_{j-1}}).\]

With P3, the joint distribution over $\bar{A}_{jT_j:k_sT_{j}}$ and $\bar{x}'_{jT_j:k_sT_{j}}$ can be repressed as:
\[\bar{A}_{jT_j:k_sT_{j}} \sim \text{SumLog}(\bar{x}'_{jT_j:k_sT_{j}}, g(\tilde{\zeta}_{jT_j})),\]
\[\bar{x}'_{jT_j:k_sT_{j}} \sim \text{Pois}(\tilde{\zeta}_{jT_j} \tau_0 \bar{\Pi}_k; \sum_{s=1}^{S_{j,T_{j-1}}} \bar{w}_{j,T_{j-1}:s}/S_{j,T_{j-1}}).\]
To marginalize over $\tilde{W}_{jT_j}$, again via $P1$, we augment $\tilde{x}^{(2)}_{jT_j,k,s,T_j}$ as

$$
\tilde{x}^{(2)}_{jT_j,k,s,T_j} = \sum_{k=1}^{K} \mathcal{G}_{jT_j,k,s,T_j} \sim \text{Pois}(\tilde{\zeta}_{jT_j} \tau_0 \Pi_{kk_1} \sum_{s=1}^{S_j,T_j-1} \tilde{w}_{j,T_j-1,k,s}/S_j,T_j-1) \tag{19}
$$

Since the transition weights $\sum_{k=1}^{K} \Pi_{kk_1} = 1$, we have

$$
\tilde{Z}_{jT_j,k,s,T_j} \sim \text{Pois}(\tilde{\zeta}_{jT_j} \tau_0 \sum_{s=1}^{S_j,T_j-1} \tilde{w}_{j,T_j-1,k,s}/S_j,T_j-1) \tag{20}
$$

To get the likelihood for $\tilde{u}_{j,T_j-1,k,s,T_j-1}$, we further augment $\tilde{Z}_{jT_j,k,s,T_j}$ as

$$
\tilde{Z}_{jT_j,k,s,T_j} \sim \text{Pois}(\tilde{\zeta}_{jT_j} \tau_0 \tilde{w}_{j,T_j-1,k,s,T_j-1}/S_j,T_j-1) \tag{21}
$$

Next, we summarize the information about the data at time-steps $T$ and $T-1$ as $\tilde{u}_{j,T_j-1,k,s,T_j} = \tilde{Z}_{jT_j,k,s,T_j} + \tilde{A}_{j,T_j-1,k,s,T_j-1}$, via $P1$, get the likelihood

$$
\tilde{u}_{j,T_j-1,k,s,T_j} \sim \text{Pois}((\tilde{\zeta}_{jT_j} \tau_0 S_j,T_j/S_j,T_j-1 + \delta_{j,T_j-1}) \tilde{w}_{j,T_j-1,k,s,T_j-1}) \tag{22}
$$

With $\tilde{u}_{j,T_j-1,k,s,T_j-1}$, we can sample auxiliary $\tilde{x}^{(2)}_{jT_j-1,k,s,T_j-1}$ via CRT distribution and re-express it in the same way with $\mathcal{G}^{(16)}(\mathcal{G}^{(17)},\mathcal{G}^{(18)})$

$$
\tilde{x}^{(2)}_{jT_j-1,k,s,T_j-1} = \text{Pois}(\tilde{\zeta}_{j,T_j-1} \tau_0 \Pi_{kk_1} : \sum_{s=1}^{S_j,T_j-2} \tilde{w}_{j,T_j-2:s}/S_j,T_j-2) \tag{23}
$$

Repeating the process all the way back to $t = 1$, it is able to marginalize over all latent feature representations $\{\tilde{w}_{jtk}\}_{j=1,t=1,k=1}$.

**Sample forward feature representation $\tilde{w}_{jtk}$**: After sampling the auxiliary variable above, we sample $\tilde{w}_{jtk}$ forward from $t = 1$ to $T$ using the gamma-Poisson conjugacy

$$
\begin{align*}
(\tilde{w}_{j1k} | -) &\sim \text{Gam}(\tilde{A}_{j1k} + \tilde{Z}_{j2k} + \tau_0 \tilde{v}_0, 1/((\tilde{\delta}_{1j} + \tilde{\zeta}_{1,T_j} \tau_0 + \tau_0)) \\
(\tilde{w}_{jtk} | -) &\sim \text{Gam}(\tilde{A}_{jtk} + \tilde{Z}_{j,t+1,k} : + \tau_0 \Pi_{kk_1} : \sum_{s=1}^{S_j,t-1} \tilde{w}_{j,t-1:s}/S_j,t-1; 1/((\tilde{\delta}_{jt} + \tilde{\zeta}_{j,t+1,T_j} + \tau_0))
\end{align*} \tag{24}
$$

where $\tilde{Z}_{j,T_j+1,k} = 0$, $\tilde{\zeta}_{j,T_j+1} = 0$, $\tilde{\zeta}_{jt} = \ln(1 + \frac{\tilde{\delta}_{jt} + \tilde{\zeta}_{j,t+1,T_j} + \tau_0}{\tau_0})$.

**Sample forward transition matrix $\tilde{\Pi}$**: Given $\mathcal{G}$ for each sentence and since $\sum_{k=1}^{K} \tilde{\pi}_{kk} = 1$, we marginalize over $\tilde{w}_{jt}$ and get

$$
(\tilde{\tilde{Z}}_{j1k}, \ldots, \tilde{\tilde{Z}}_{jKk} | -) \sim \text{Multi}(\tilde{\tilde{Z}}_{j1k}, \ldots, \tilde{\tilde{Z}}_{KKk}). \tag{25}
$$

With the Dirichlet-multinomial conjugacy, we have

$$
(\tilde{\pi}_{k} | -) \sim \text{Dir}(\tilde{\tilde{v}}_1 \tilde{v}_k + \tilde{\tilde{Z}}_{1,k}, \ldots, \tilde{\tilde{v}}_K \tilde{v}_k + \tilde{\tilde{Z}}_{K,k}) \tag{26}
$$

For $\tilde{\delta}_{jt}$, $\tilde{\tilde{z}}$, $\{\tilde{v}_k\}_{k=1}^K$ and $\tilde{\zeta}$, we sample them in the same way with $\mathcal{G}^{[43]}$.

In parallel with backward-forward sampling for $\tilde{W}_{jt}$, we can sample $\tilde{W}_{jt}$ in the same way with $\tilde{W}_{jt}$, but towards the opposite direction.
B Scalable inference via stochastic gradient MCMC

While the computation of the Gibbs sampler can be accelerated inside each iteration, it requires processing all documents in each iteration and hence has limited scalability. For scalable inference, we update our global parameters by generalizing TLASGR-MCMC [17][18], a stochastic gradient MCMC (SG-MCMC) algorithm that is proposed to sample simplex-constrained global parameters in a mini-batch learning setting and improve its sampling efficiency by preconditioning its gradient via the Fisher information matrix (FIM). More Specifically, after sampling auxiliary latent counts using parallel augmentable techniques as in Appendix A, the update of $D$ with TLASGR-MCMC can be described as

$$\begin{align}
(D_{k}(::))_{i+1} &= \left\{ (D_{k}(::))_{i} + \frac{\varepsilon_{i}}{M_{k}} \left[ (\rho(d'_{k1}, \ldots, d'_{kN})' + \eta) 
- (\rho d_{k} + \eta V F) (D_{k}(::))_{j} \right] 
+ \mathcal{N} \left( 0, \frac{2\varepsilon_{i}}{M_{k}} \text{diag}((D_{k}(::))_{i}) \right) \right\} \land 
\end{align}$$

where $[\cdot] \land$ denotes the simplex constraint and $M_{k}$ is calculated using the estimated FIM. $d'_{k1}$ and $d'_{kN}$ come from (10) and (11). In addition, $\pi_{k}$, the $k$th column of the forward transition matrix $\Pi$, can be efficiently sampled as

$$\begin{align}
(\pi_{k})_{i+1} &= \left[ (\pi_{k})_{i} + \frac{\varepsilon_{i}}{M_{k}} \left[ (\rho \pi_{k} + v) - (\rho \pi_{k} + \bar{v}) \right] \right] 
+ \mathcal{N} \left( 0, \frac{2\varepsilon_{i}}{M_{k}} \text{diag}(\pi_{k})_{i} - (\pi_{k})_{i} (\pi_{k})_{i}^{T} \right) \land 
\end{align}$$

where $\pi_{k}$, and $\bar{v}$, come from the augmented latent counts $\tilde{Z}$ in (25) and $\varepsilon_{i}$ denotes the prior of $\pi_{k}$. The backward transition matrix $\Pi$ can be parallely sampled in the same way. More details of TLASGR-MCMC can be found in Cong et al. [17] and Guo et al. [51]. The developed SG-MCMC algorithm for bi-conv-PGDS is described in Algorithm 1.

Algorithm 1 stochastic-gradient MCMC for bi-conv-PGDS

Input: Data mini-batches; Output: Global parameters of bi-conv-PGDS.

for $iter = 1, 2, \ldots$ do

\*
Collect local information

Parallel bidirectional Gibbs sampling on the $ith$ mini-batch for $\{d_{jtv}\}_{j,t,v}$, $\{\tilde{Z}_{jtk1k}\}_{j,t,k1,k}$, $\{\tilde{Z}_{jtk1k}\}_{j,t,k1,k}$, $\{\tilde{u}_{jtk}\}_{j,t,k}$, $\{\tilde{v}_{jtk}\}_{j,t,k}$ with (10) (11) (25) (22);

Calculating for the $\tilde{\zeta}_{jt}$, $\tilde{\zeta}_{jt}$ with (25);

Parallel bidirectional Gibbs sampling for the $\{\tilde{W}_{j}, \tilde{W}_{j}\}_{j,t}$ with (24);

\*
Update global parameters

for $k = 1, 2, \ldots K$ do

Update $M_{k}$ according to (20); then $\{D_{k}\}_{i}$ with (27); Update $M_{k}$ and $\tilde{M}_{k}$ according to (20); then $\{\pi_{k}\}_{i}$ and $\{\tilde{\pi}_{k}\}_{i}$ with (28);

end for

Update $\tilde{\xi}$, $\tilde{\xi}$, $\{\tilde{v}_{k}\}$, $\{\tilde{u}_{k}\}$, $\tilde{\beta}$, $\tilde{\beta}$ according to (43)

end for

C Hybrid SG-MCMC/auto-encoding variational inference

To allow for scalable inference and fast testing, we develop a hybrid SG-MCMC/autoencoding variational inference [47][48] algorithm by combining TLASGR-MCMC [17][18] and our proposed convolutional-recurrent variational inference network. In other words, in minibatch based each
More specifically, after sampling auxiliary latent counts using parallel augmentable techniques, we update the global parameters $D, \tilde{\Pi}$, and $\tilde{\Pi}$ with TLASGR-MCMC as in Appendix B. Given the global parameters, the task here is to optimize the parameters of the convolutional-recurrent variational inference network. As the usual strategy of variational inference, we achieve this optimization via SGD by minimizing the negative evidence lower bound (ELBO), which can be expressed as

$$L = \sum_{j=1}^{M} \sum_{t=1}^{T} \mathbb{E}_{q(W_{jt}, \tilde{W}_{jt})} \left[ \ln p(x_{jt}| D, \tilde{W}_{jt}, \tilde{W}_{jt}) \right]$$

$$- \text{KL} \left( q(W_{jt}, \tilde{W}_{jt}) \| p(W_{jt}| \tilde{\Pi}, \tilde{W}_{jt, t-1})p(\tilde{W}_{jt}| \tilde{\Pi}, \tilde{W}_{jt, t+1}) \right)$$

(29)

KL$(q(\cdot)||p(\cdot)) = \mathbb{E}_{q(\cdot)} \log (q(\cdot)/p(\cdot))$ denotes the Kullback–Leibler (KL) Divergence from distribution $p(\cdot)$ to $q(\cdot)$. The proposed hybrid stochastic-gradient MCMC and autoencoding variational inference algorithm is described in Algorithm 2, which is implemented in PyTorch [62], combined with pyCUDA [63] for more efficient computation.

**Algorithm 2** Hybrid stochastic-gradient MCMC and autoencoding variational inference for bi-conv-PGDS

Set mini-batch size as $M$, the number of convolutional filters $K$ and hyperparameters:

- Initialize inference model parameters $\Omega$ and generative model parameters $\{D_{k}\}_{k=1}^{K}$, $\tilde{\Pi}$, $\tilde{\Pi}$;

- for $\text{iter} = 1, 2, \ldots$ do

- Randomly select a mini-batch of $M$ documents consisting of $T$ sentences to form a subset $\{x_{i,1:T}\}_{i=1}^{M}$;

- Draw random noise $\{\epsilon_{i,t}\}_{i=1,t=1}^{M,T}$ from uniform distribution for sampling latent states;

- Calculate $\nabla_{\Omega}L \left( \Omega, D, \tilde{\Pi}, \tilde{\Pi}; X, \epsilon_{i,t} \right)$ according to (29), and update $\Omega$;

- Sample $\{\tilde{W}_{jt}, \tilde{W}_{jt}\}_{j=1,t=1}^{M,T}$ from equation (4) in our paper via $\Omega$, update $D$ according to [20], update $\tilde{\Pi}$ and $\tilde{\Pi}$ according to (28) parallelly.

end for

### D Datasets

Five large scale document datasets are used in this paper, including Reuters, ELEC, IMDB-2, IMDB-10 and yelp 2013. ELEC dataset [64] consists of electronic product reviews, which is part of a large Amazon review dataset. IMDB dataset is a benchmark dataset for sentiment analysis, where the task is to determine whether a movie review is positive or negative. Yelp is a restaurant review dataset. Reuters is an English news dataset and a multilabel-dataset. We use the standard ModApte splits [65] to transform Reuters to a single label dataset. Following [20][57], for ELEC and IMDB-2, we use 50% of the data for training and the remaining 50% for testing. For other datasets, we use 80% for training and the remaining 20% for testing. The statistics of these five datasets are summarized in Table 5.

### E Semi-supervised classification

In many practical scenarios, unlabeled data are abundant, however, there are not many practical cases where the potential of such unlabeled data is fully realized. Base on this condition, it is imperative to seek to complement scarcer but more valuable labeled data, to improve the generalization ability of supervised models. By ingesting unlabeled data, the model can learn to abstract latent representations that capture the semantic meaning of all available sentences irrespective of whether or not they are labeled. This can be done prior to the supervised model training, as a two-step process. Recently, many methods exploiting this idea have been wildly utilized and have achieved state-of-art performance in many tasks [55][66][68].
Table 5: Data statistics: #s denotes the number of sentences (average and maximum per document), #w denotes the number of words (average and maximum per document), V_{pre} denotes the number of words in the set of pre-trained word vectors.

<table>
<thead>
<tr>
<th>Data set</th>
<th>classes</th>
<th>documents</th>
<th>average #s</th>
<th>max #s</th>
<th>average #w</th>
<th>max #w</th>
<th>vocabulary</th>
<th>V_{pre}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reuters</td>
<td>90</td>
<td>10789</td>
<td>6.6</td>
<td>248</td>
<td>144.3</td>
<td>2244</td>
<td>34397</td>
<td>30000</td>
</tr>
<tr>
<td>ELEC</td>
<td>2</td>
<td>50000</td>
<td>2.8</td>
<td>201</td>
<td>124</td>
<td>6000</td>
<td>52248</td>
<td>30000</td>
</tr>
<tr>
<td>IMDB-2</td>
<td>2</td>
<td>50000</td>
<td>13.2</td>
<td>148</td>
<td>265</td>
<td>2800</td>
<td>95212</td>
<td>30000</td>
</tr>
<tr>
<td>IMDB-10</td>
<td>10</td>
<td>135669</td>
<td>14.0</td>
<td>148</td>
<td>325.6</td>
<td>2802</td>
<td>115831</td>
<td>30000</td>
</tr>
<tr>
<td>Yelp 2014</td>
<td>5</td>
<td>1125457</td>
<td>9.2</td>
<td>151</td>
<td>159.9</td>
<td>1199</td>
<td>476191</td>
<td>30000</td>
</tr>
</tbody>
</table>

Figure 7: Example phrase-level topics from ELEC learned by conv-PGDS.

As a complex pre-trained methods, BERT is pre-trained on large scale data and achieved state-of-the-art performance in many tasks. In the semi-supervised experiment of this paper, we first pre-train BERT with unlabeled data using masked ML (MLM) tasks, which is designed by masking some percentage of the input tokens at random, and then predict those masked tokens. Then, we train it on labeled data for document classification.

Our proposed convolutional-recurrent autoencoding framework can also be applied to semi-supervised learning from document. Instead of pre-training a fully supervised model as BERT, we simultaneously train a sequence autoencoder and a supervised model. In principle, by using the joint training strategy as in Section 3 in our paper, the learned document embedding vector will preserve both reconstruction and classification ability. Specifically, we consider the following objective

$$L_{semi} = \sum_{d \in (\mathcal{D}_l + \mathcal{D}_u)} L_g (d) + \xi \sum_{d \in \mathcal{D}_l} L_s (d)$$  \hspace{1cm} (30)$$

which is a modification of our supervised loss, where \(\mathcal{D}_l\) and \(\mathcal{D}_u\) denote the set of labeled and unlabeled data, respectively.

F Additional experimental results

We list the top 15 phrase-level topics from ELEC learned by conv-PGDS in Fig. 7, which are related to the corresponding transition matrix in Fig. 3 in our paper. It is clearly that the words in different convolutional filters can be combined into different phrase-level topics.

In addition to conv-PGDS, we also visualize the inferred convolutional filters and transition matrices of bi-conv-PGDS. Fig. 8 shows the top 20 topics learned by bi-conv-PGDS on ELEC, whose
As introduced before, the reason why we develop a parallelized Gibbs sampler as well as a hybrid SG-MCMC/VI is that both of them have their own advantages. We compare the accuracy of text classification and the corresponding testing time in Table 6. As we can see, the use of encoder makes our model fast in testing time, but leads to a tradeoff in accuracy.
Figure 10: Two example forward transition relations between topics from ELEC learned by bi-conv-PGDS (better understood together with Fig.9 (a)).

Figure 11: Two example backward transition relations between topics from ELEC learned by bi-conv-PGDS (better understood together with Fig.9 (b)).

attn-CPFA GT:4 Prediction: 5

*Normally I love chipotle and I would rate it 5 stars.*

The ingredients are scrumptious and the experience is sublime.

However, at this restaurant not all of the staff was on their game.

attn-bi-conv-PGDS GT:4 Prediction: 4

*Normally I love chipotle and I would rate it 5 stars.*

The ingredients are scrumptious and the experience is sublime.

However, at this restaurant not all of the staff was on their game.

attn-CPFA GT:5 Prediction: 3

What I hate about buffets in general is that the food is usually cooked poorly when cooked in large quantities

But that is totally not the case here.

All the meat and fish I sampled were cooked perfectly.

attn-bi-conv-PGDS GT:5 Prediction: 5

What I hate about buffets in general is that the food is usually cooked poorly when cooked in large quantities

But that is totally not the case here.

All the meat and fish I sampled were cooked perfectly.

Figure 12: Examples of attention for Yelp’14 learned by attn-CPFA and attn-bi-conv-PGDS.

attn-CPFA GT:0 Prediction: 1

*I really enjoyed using these headphones.*

They are comfortable, lightweight and they sound great.

However, they are not made well at all.

If you use these daily, the left ear will short out after about a month of use.

attn-bi-conv-PGDS GT:0 Prediction: 0

*I really enjoyed using these headphones.*

They are comfortable, lightweight and they sound great.

However, they are not made well at all.

If you use these daily, the left ear will short out after about a month of use.

Figure 13: Examples of attention for ELEC learned by attn-CPFA and attn-bi-conv-PGDS.

Table 6: Comparison of the testing times (seconds) with batch-size 25.

<table>
<thead>
<tr>
<th>Methods</th>
<th></th>
<th>Accuracy</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Testing time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reuters</td>
<td>IMDB-3</td>
<td>IMDB-10</td>
<td>Reuters</td>
<td>IMDB-3</td>
<td>IMDB-10</td>
</tr>
<tr>
<td>bi-conv-PGDS (TLASGR-MCMC)</td>
<td>78.0 ± 0.7</td>
<td>84.5 ± 0.8</td>
<td>84.0 ± 0.8</td>
<td>37.9 ± 0.4</td>
<td>17.35</td>
<td>15.07</td>
<td>23.61</td>
</tr>
<tr>
<td>bi-conv-PGDS (hybrid SG-MCMC and VI)</td>
<td>76.8 ± 1.0</td>
<td>83.7 ± 1.2</td>
<td>82.6 ± 1.1</td>
<td>36.9 ± 0.7</td>
<td>0.15</td>
<td>0.14</td>
<td>0.17</td>
</tr>
</tbody>
</table>

21