Abstract

Oftentimes documents are linked to one another in a network structure, e.g., academic papers cite other papers, Web pages link to other pages. In this paper we propose a holistic topic model to learn meaningful and unified low-dimensional representations for networked documents that seek to preserve both textual content and network structure. On the basis of reconstructing not only the input document but also its adjacent neighbors, we develop two neural encoder architectures. **Adjacent-Encoder**, or **AdjEnc**, induces competition among documents for topic propagation, and reconstruction among neighbors for semantic capture. **Adjacent-Encoder-X**, or **AdjEnc-X**, extends this to also encode the network structure in addition to document content. We evaluate our models on real-world document networks quantitatively and qualitatively, outperforming comparable baselines comprehensively.

Introduction

Text corpora constitute an important class of data, covering academic papers, Web pages, product descriptions, etc. To better make sense of the meaning within text documents, we seek to learn a lower-dimensional representation. One such representation is based on the notion of topics. Essentially, a document is associated with a set of topics, and in turn a topic is associated with pertinent words. Classically, many topic models are based on graphical models, such as LDA (Blei, Ng, and Jordan 2003). More recently, topic models are based on neural approaches including Auto-Encoders and its variants, such as KATE (Chen and Zaki 2017).

In this work, we investigate neural topic models not for plain-text documents per se, but for networked documents. In addition to textual content, oftentimes documents link to one another in a network structure. For example, academic papers form a citation network, Web pages form a hyperlink network. Many previous works on topic modeling focus on textual content of documents; some do incorporate the network structure to jointly learn representations, such as RTM (Chang and Blei 2009). To this end, novel approaches to unsupervised topic modeling for document networks are ger-

### Problem

Let \( G = (D, \mathcal{E}) \) be a given document network. \( D = \{d_1, d_2, ..., d_N\} \) is a set of documents, where each document \( d \in \mathbb{R}^{[V]} \) is a vector in the vocabulary space. In turn, the adjacency matrix \( \mathcal{E} \in \mathbb{R}^{N \times N} \) is a 0-1 matrix where \( \mathcal{E}_{ij} = 1 \) indicates document \( d_i \) links to \( d_j \), and \( \mathcal{E}_{ij} = 0 \) otherwise. Here we model an undirected network, i.e., \( \mathcal{E}_{ij} = \mathcal{E}_{ji} \) and \( \mathcal{E} = \mathcal{E}^T \), though the proposed models could generalize to directed networks as well. We would use edge and link interchangeably. For a document \( d \), its neighbors are those directly linked to \( d \). For simplicity, we use \( \mathcal{N}(d) = \{d_1, d_2, ..., d_k\} \) to denote \( d \)'s neighbors. The definition of neighborhood here is reflexive, i.e., we also regard \( d \) as its own neighbor, \( d \in \mathcal{N}(d) \) and \( \varepsilon_{ii} = 1 \).

Given \( G \) as input, our aim is to embed documents in \( G \) within a low-rank topic space. Recent neural topic models are based on the traditional **Auto-Encoder** family, which naturally embodies the notion of a topic model, by learning the association between documents and topics (hidden neurons), as well as topics and words. However, in seeking to reconstruct the input document, it would model each document independently and disregard the network structure in \( G \).

### Proposed Approach

To deal with networked documents, we propose an approach called **Adjacent-Encoder** or **AdjEnc**, whose key distinction is to also reconstruct the neighbors of the input document, in addition to the document itself. Hypothetically, this allows documents in a network to collaboratively learn from one another, such that close neighbors would have similar representations in the topic space. The realization of this principle leads to novel structures within the **AdjEnc** architecture, i.e.,

- **Neighbor Competition**: Neighbors contribute information differentially. In the encoding phase, we evaluate attentions between the target document and its neighbors. In turn, neighbors propagate topics to the target document.

- **Neighbor Reconstruction**: In the decoding phase, the target document reconstructs the contents of its adjacent neighbors. This increases the robustness and invariance of topic representations with respect to output documents, while also incorporating the neighborhood structure without additional parameters over those of Auto-Encoders.

Beyond reconstructing the content of neighbors, it is fea-
sible to reconstruct their neighborhood structure as well. This factors in higher-order proximities by modeling the adjacency matrix explicitly. We realize this in an extension Adjacent-Encoder-X or AdjEnc-X, which jointly embeds content and network structure in a unified manner.

**Contributions.** Our contributions are as follows. First, we propose two novel architectures, Adjacent-Encoder and Adjacent-Encoder-X, as unsupervised topic models for document networks. Second, we systematically incorporate network structure in two ways, neighbor competition for topic propagation and neighbor reconstruction for semantic capture. Moreover, Adjacent-Encoder-X also investigates reconstruction of textual content and network structure. Third, we compare our models quantitatively and qualitatively against baselines of neural and graphical varieties on several evaluation metrics. Fourth, beyond showing improvements over comparable baselines, we investigate the complementarity and improved effectiveness of neighbor competition and reconstruction when combined with other architectural extensions such as denoising, contractive, and sparsity.

**Related Work**

There are architectural variants to Auto-Encoder that have been shown to improve the performance. Denoising Auto-Encoder (DAE) (Vincent et al. 2010) adds random noise to the input document and reconstructs its original content to learn useful patterns while avoiding overfitting. Contractive Auto-Encoder (CAE) (Rifai et al. 2011) introduces the Frobenius norm of Jacobian matrix to the loss function for regularization. Variational Auto-Encoder (VAE) (Kingma and Welling 2014) makes use of variational inference to learn topics in a generative approach. K-Sparse Auto-Encoder (KSAE) (Makhzani and Frey 2013) and K-Competitive Auto-Encoder (KATE) (Chen and Zaki 2017) force topics to be sparse by keeping the values of only $k$ hidden neurons and zeroing others. ProdLDA (Srivastava and Sutton 2017) uses product of experts to generate words in an unsupervised manner. Some models learn representations for vertices based on attributed graph, e.g., Graph Neural Networks (Scarselli et al. 2009; Rumelhart, Hinton, and McClelland 1986), Graph Convolutional Network (Kipf and Welling 2017), Graph Attention Networks (Velickovic et al. 2018), Variational Graph Auto-Encoders (Kipf and Welling 2016). Others propagate vertex embeddings over networks, such as Embedding Propagation (EP) (García-Durán and Niepert 2017). Yet others consider networks only (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016; Tang et al. 2015; Cao, Lu, and Xu 2015).

**Model Architecture and Analysis**

In this section, we describe the technical details of our proposed models, Adjacent-Encoder and Adjacent-Encoder-X.

We briefly review Auto-Encoder to make our contrast clear. With activation function $f$, we learn hidden representation $h_d$ for the input document $d$ at the encoder: $h_d = f(Wd + b)$. The decoder reconstructs the original content of the input document by $\hat{d} = f'(W'hd + c)$. Here $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^{|V|}$ are biases, $W \in \mathbb{R}^{m \times |V|}$ and $W' \in \mathbb{R}^{|V| \times m}$ are encoder and decoder parameters. Typically we use weight tying ($W' = W^T$) as regularization. $V$ is the vocabulary, and $m$ is the number of hidden neurons. By minimizing the reconstruction error, we obtain $h_d$ as topic representations.

**Adjacent-Encoder**

Fig. 1 contrasts our proposed models Adjacent-Encoder (Fig. 1b) and Adjacent-Encoder-X (Fig. 1c) with tradi-
Adjacency-Encoder (Fig. 1a). Here we describe Adjacent-Encoder by highlighting its constituent structures, and defer the discussion on Adjacent-Encoder-X to the next section.

As a running example, we assume a toy network of 5 documents \{A, B, C, D, E\} as in Fig. 1b. The key principle behind Adjacent-Encoder is to have a target document, say A, reconstruct itself and its adjacent neighbors, say B and D. This manifests via the mechanisms illustrated in Fig. 2.

**Neighborhood Competition.** The first is to allow competition among documents to assess relative importance among neighbors. As in (Chen and Zaki 2017), we represent each input document as a log-normalized word count vector \(d \in \mathbb{R}^{|V|}\), i.e., each dimension is \(d_i = \frac{\log(1+n_i)}{\max_{i \in V} \log(1+n_i)}\) where \(n_i\) is the count of word \(i\) in \(d\). For a target document \(d\), we learn its hidden vector \(h_d\) at the feedforward phase by \(h_d = \tanh(W_d + b)\). The attention coefficients between \(d\)'s neighbors and itself are \(a_{d,d'}\) as shown in Fig. 2b.

\[
\begin{align}
\epsilon_{d,d'} &= h_d^T h_d' ,
\quad a_{d,d'} = \frac{\exp(\epsilon_{d,d'})}{\sum_{d' \in \mathcal{N}(d)} \exp(\epsilon_{d,d'})} \\
\end{align}
\]

where \(d' \in \mathcal{N}(d)\) is a neighbor of \(d\). The attention coefficients measure relative importance among \(d\)'s neighbors.

Links among documents indicate a shared relationship. Thus we propagate the topics of neighbors to the target document \(d\), which, in turn, is also a neighbor of other documents, thereby propagating topics even further. We allow topics to flow through neighbors across the network, so that documents collaboratively learn from one another. This procedure is driven by the following transformation, which is also illustrated by Fig. 2c.

\[
\hat{h}_d = \sum_{d' \in \mathcal{N}(d)} a_{d,d'} h_{d'} .
\]

**Neighbor Reconstruction.** The content of a document is the observed reflection of its internal topics. Since we know linked documents are likely to share similar topics, we could use topic representation \(h_d\) of a target document to reconstruct the contents of its adjacent neighbors in a “1-to-N” reconstruction manner. We adopt sigmoid as the output activation function, and weight tying is used for regularization.

\[
\hat{d} = \text{sigmoid}(W^T \hat{h}_d + c) 
\]

where \(\hat{d}\) is the reconstructed document. We use binary cross-entropy as the loss function.

\[
l(d', \hat{d}) = -\sum_{i \in V} [d'_i \log(\hat{d}_i) + (1 - d'_i) \log(1 - \hat{d}_i)] .
\]

Again, \(d' \in \mathcal{N}(d)\) is one of the adjacent neighbors. We have each target document \(d\) reconstruct each of its neighbors, as illustrated in Fig. 2d. Each document in the network takes turn as the target document. We repeat the learning process above for unsupervised training until convergence.

Reconstructing adjacent neighbors is somewhat related to Denoising Auto-Encoder (DAE), which reconstructs a document from a noisy version of itself. Instead of noise at the input layer, our models have a target document reconstruct adjacent neighbors, which in a way serves as “noise” at the output layer. In our case, the “noise” is naturally introduced by the network, instead of being a random and artificial addition to documents. The reconstruction of neighbors allows our models to capture the case where two documents are different in observed content, but consistent in terms of the internal topics, thus the learned topics can preserve document semantics well. Furthermore, it also increases the robustness and invariance of the learned topics w.r.t. output documents.

**Inference.** Once our models have been trained, we simply encode each testing document by \(h_d = \tanh(W_d + b)\). Here \(h_d\) is the topic representation of the testing document, which preserves information of both text and network structure.

**Adjacent-Encoder-X**

The previously described Adjacent-Encoder models network structure implicitly. In this section we propose an improved framework, Adjacent-Encoder-X, which models network structure explicitly. The distinction of these two is illustrated in Fig. 1. The name is inspired by the ‘X’ structure of dual observations of textual content and adjacency vector.

**Neighbor Competition.** Adjacency matrix \(E\) represents the network structure. The \(i^{th}\) row (or column) \(e_i\) represents the neighborhood relationship of \(i^{th}\) document. If two documents have many common neighbors, their corresponding adjacency vectors are similar. Intuitively, the more common neighbors two documents have, the more likely they share similar topics. Two academic papers may share similar topics if both cite many of the same papers. Web pages may be of the same category if they link to common Web pages.

Hence, we treat the adjacency vector \(e\) as another input in addition to the textual content \(d\). The hidden vector of the feedforward phase can be computed by \(h_d = \tanh(W_d + b)\). The attention coefficients between \(d\)'s neighbors and itself are \(a_{d,d'}\) as shown in Fig. 2b. The other steps follow the same logic as before.
The complexity of each target document is $O(m \deg_{\text{max}})$. For neighbor reconstruction, we reconstruct all adjacent neighbors, thus we have $O(m F \deg_{\text{max}})$. Putting all three components together, for each target document, we obtain $O(m F + m \deg_{\text{max}} + m F \deg_{\text{max}})$ for the overall model.

**Experiments**

Our experimental objective is to validate the quality of topics learned by our models on evaluative tasks such as document classification, document clustering, link prediction, etc.

**Setup**

**Datasets.** Cora\(^1\) is a public collection of papers and their citations (McCallum et al. 2000). Each document is an abstract. Two documents are linked by an undirected edge if one cites the other. Following (Zhu et al. 2007), we extract four independent datasets: Data Structure (DS), Hardware and Architecture (HA), Machine Learning (ML), and Programming Language (PL). Each dataset is organized into categories, which we treat as class labels (not used in learning, only evaluation). Table 2 presents their statistics. All data and code are available for reproducibility\(^2\).

**Baselines.** We compare our models with several categories of baseline models as listed below.

- **Auto-Encoders:** Since our models are encoders, the most appropriate baselines are of the Auto-Encoder family, i.e., AE, DAE (Vincent et al. 2010), CAE (Rifai et al. 2011), VAE (Kingma and Welling 2014), KSAE (Makhzani and Frey 2013), and the state-of-the-art topic model KATE (Chen and Zaki 2017). As they encode only the document content, through this comparison we validate the efficacy of jointly learning content and network structure.

- **Generative topic models:** Another family of topic models are based on the generative approach. We compare to those that incorporate document content and network structure concurrently, such as RTM (Chang and Blei 2009), PLANE (Le and Lauw 2014), and the recent NRTM (Bai et al. 2018). We also include ProdLDA (Srivastava and Sutton 2017), a recent topic model that still encodes each document independently.

- **Graph embedding:** Recently there are some models making use of Auto-Encoder for unsupervised graph representation learning. Strictly speaking, they are not topic models, nor baseline. For completeness, we include a comparison to VGAE (Kipf and Welling 2016).

**Training Details.** Following (Chen and Zaki 2017; Bai et al. 2018), the activation functions for AE, DAE, CAE, KSAE, and NRTM are $\text{sigmoid}$, while those for VAE and KATE are $\text{tanh}$ (hidden) and $\text{sigmoid}$ (output) respectively. We use validation set to choose the best hyperparameters. DAE with Gaussian noise of 0.25 std.dev. outperforms other kinds of noise. We choose 2 and 0.01 as Dirichlet hyperparameters.

\(^1\)http://people.cs.umass.edu/mccallum/data/cora-classify.tar.gz

\(^2\)https://github.com/cezhang01/Adjacent-Encoder
the number of nonzero hidden neurons, \(k\), to 4, 8, 16, 32, and 52 when the number of topics is 16, 32, 64, 128, and 256, respectively. Each result is an average of 10 independent runs.

**Transductive vs. Inductive Learning.** There are two scenarios in which we can apply the models. In the transductive setting, the objective is to derive topic representations of the documents already in the corpus. In this case, all documents in the corpus are present during training. Conversely, in the inductive setting, the objective is to generalize beyond the training corpus to unseen data, which we simulate by keeping a random subset of 80% documents for training (out of which we further randomly split 10% documents for validation) and the remaining 20% for testing. As both are feasible scenarios, we discuss our experiments under each setting.

**Transductive Learning**

For validating the derived document representations, we rely on three evaluative tasks. The first two are document classification and clustering, evaluated via class labels (these are never part of any learning). The last is link prediction.

**Document Classification.** Intuitively, topic representations may align with categorizations of documents, i.e., documents within a class may share similar topics. Since our goal is high-quality topic representations, we use simple K-Nearest Neighbors as the classifier at the testing phase. For each document, we hide its actual label, and predict its label as the majority label of its k-nearest neighbors based on the Euclidean distance in the low-dimensional topic space. Classification accuracy is used as the metric. The accuracies at \(m = 64\) and \(K = 10\) are summarized in Table 3 (left).

Indeed, our models outperform the baselines significantly across all four datasets. Except for ML dataset, Adjacent-Encoder-X generally achieves higher results than Adjacent-Encoder, because the former captures higher-order proximity in having common neighbors in addition to being direct neighbors. Among the baselines, DAE, VAE, KATE, and VGAE tend to be better, but none achieves a consistent outperformance over others. The top panel of Fig. 3 and 4 presents the results when varying the number of topics \(m\) and neighbors \(K\), respectively. Our models still outper-
form baseline models most of the time. The only exception is HA, on which our models are competitive with KATE and VGAE. However, the best result of our models as well as baselines is achieved at $K = 10$ where both our models outperform all the baselines significantly.

**Document Clustering.** We can also use the representations for clustering documents (we use $K$-means), investigating if documents in a cluster tend to share the same class.
Class labels are used only for investigating normalized mutual information (NMI) for evaluation. The clustering result at \( m = 64 \) is shown in Table 3 (middle columns). Overall, our models outperform all the baselines significantly. Except for DS, Adjacent-Encoder-X achieves better clustering than Adjacent-Encoder. Among the baselines, ProdLDA, PLANE, and VGAE tend to perform better than others.

**Link Prediction.** Given two documents, we could use their topics to predict the link between them. Following (Bishop 1995), the link generation probability is given by 

\[
P(\varepsilon_{ij} = 1 | \mathbf{h}_d, \mathbf{h}_d') \propto \exp(-||\mathbf{h}_d - \mathbf{h}_d'||^2).
\]

One measure is to examine whether models provide a high generation probability to actual links. As in (Agibetov and Samwald 2018), we use Mean Average Precision (MAP) as evaluation metric. Following (Le and Lauw 2014), we randomly hide one link for those documents with at least three neighbors (excluding itself) and keep the remaining network connected. The remaining network is present for training. After convergence, we use the topics to predict the held-out links.

Table 3 (rightmost) presents the results for \( m = 64 \). Adjacent-Encoder and Adjacent-Encoder-X outperform the baselines significantly across four datasets. By comparing our models with AE-based models, we see that considering network structure helps to embed neighbors more closely, thereby achieving a high MAP. Our models rank links higher than others that factor in the network structure (RTM, PLANE, NRTM, VGAE), supporting our outperformance on jointly learning content and network structure.

**Inductive Learning**

For the inductive setting, we evaluate model performance for out-of-sample documents. Thus, we take care not to involve the testing documents during training our encoder models. For training, we observe links only within the training set. For testing, we observe links connecting one testing and one training document, but not links with two testing documents. Once the models are trained, we use their parameters to derive the document representations for test documents and apply the same three evaluative tasks as before.

Table 4 shows the results for the inductive setting. The effects of number of topics and neighbors on inductive document classification is shown by Fig. 3 and 4 (bottom panel). Evidently, similar conclusion as with transductive learning can be drawn that our models are consistently better than baselines. For classification and clustering, PLANE presents satisfying results on transductive, but deteriorates on inductive learning. Among baselines, DAE, VAE, and VGAE tend to outperform others. For link prediction, Adjacent-Encoder ranks links higher on DS and HA, while Adjacent-Encoder-X performs better on ML and PL. The highest MAP is 0.462, achieved on HA, meaning that Adjacent-Encoder generally places true links on top 2 among candidate links.

**Topic Analysis**

For better understanding of topic-word association learnt by a topic model\(^1\), we conduct experiments on topic analysis.

---

\(^1\)VGAE is not a topic model, and is not included in this analysis.

**Table 5:** Topic Coherence PMI when \( m = 64 \).

| Model               | PMI |
|---------------------|-----|
| Adjacent-Encoder    |     |
| Adjacent-Encoder-X  | 2.360 | 2.054 | 2.180 | 2.499 | 2.321 |
| AE                  | 0.294 | 0.446 | 0.665 | 0.969 |
| DAE                 | 1.170 | 1.125 | 1.203 | 1.553 |
| CAE                 | 0.348 | 0.558 | 0.526 | 0.684 |
| VAE                 | 0.685 | 0.793 | 1.831 | 1.379 |
| KSAE                | 0.547 | 0.285 | 0.770 | 0.759 |
| KATE                | 1.312 | 1.755 | 1.619 | 2.003 |
| ProdLDA             | 1.638 | 1.315 | 1.837 | 2.088 |
| RTM                 | 1.279 | 1.678 | 1.199 | 1.615 |
| PLANE               | 1.585 | 1.847 | 1.756 | 2.099 |
| NRTM                | 1.533 | 2.041 | 1.328 | 1.632 |

**Topic Coherence.** Our topic-word association is given by \( W \) for Adjacent-Encoder, and \( W_1 \) for Adjacent-Encoder-X. We use PMI (Bouma 2009), defined as 

\[
\text{PMI}(w_i, w_j) = \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)},
\]

To evaluate the coherence of predicted words. Using Google Web IT 5-gram Version 1 (Evert 2010), \( p(w_i) \) is evaluated from 1-corpus, and \( p(w_i, w_j) \) from 5-gram corpus. For each topic, we average the pairwise PMI of its top 10 words. For each model, we average PMI of its topics.

Table 5 shows that network-based models tend to perform well, benefitting from document relatedness in addition to text content. Adjacent-Encoder has higher topic coherence than Adjacent-Encoder-X except for ML, presumably in modeling the adjacency vector explicitly the latter may reduce the reconstruction precision of text content. Nevertheless, our models still outperform baselines in most cases.

**Topic Interpretability.** To gain a semantic sense of topics, we qualitatively present top 10 words of 5 randomly selected topics (Table 6). Adjacent-Encoder’s topic 2 seems to discuss K-nearest neighbor. Topic 3 discusses reinforcement learning. For Adjacent-Encoder-X, topic 1 captures Markov decision problem, while topic 2 seems decision trees.

Each column of topic-word matrix corresponds to a word representation over topics. Thus we check whether similar words are embedded closely. Table 7 presents 5 nearest neighbors for each of 5 query words in the word representation space. Both of our models can find relevant words. For example, Adjacent-Encoder provides “nearest” and “k-nearest” for the query word neighbor. Adjacent-Encoder-X presents “protein” and “promoter” for the query word dna.

**Visualization**

As exploratory analysis, visualization provides an intuitive sense of how topic models embed documents. One may expect a good model to embed documents of a category closely. We apply t-SNE (van der Maaten and Hinton 2008) to project 64-dimensional topic space into 2-dimensional visualization space. As a sampler, Fig. 5 shows four of the methods on ML dataset. Adjacent-Encoder and Adjacent-Encoder-X produce good separation between categories.
Figure 5: t-SNE visualization on ML dataset. (best seen in color)

| Topic          | Adjacent-Encoder                                                                 |
|----------------|----------------------------------------------------------------------------------|
|                | maze, markov, mdp, observable, minute, intractable, severe, markovian, pomdp, analog |
| 1              | move, 0-1, image, nearest, promoter, neighbor, grid, k-nearest, analogy, sketch   |
| 2              | reward, influence, recurrent, credit, exploratory, max, reinforcement, net, reactive, policy |
| 3              | inference, hmm, graphical, practitioner, translation, defense, methodological, causal, probable, assist |
| 4              | pair, net, coordination, backpropagation, stronger, broad, network, classic, pendulum, multiclass |

| Topic          | Adjacent-Encoder-X                                                               |
|----------------|----------------------------------------------------------------------------------|
|                | mdp, policy, clarify, identical, pomdp, observable, tradeoff, consequence, noisy, larger |
| 1              | cart, selective, exploratory, estimator, phoneme, categorization, stability, multiclass, terminate, axis-parallel |
| 2              | parent, graph, substructure, overlap, graphical, load, emulate, integration, fashion, generalisation |
| 3              | ica, toronto, detector, blind, maximization, derivation, facial, pca, nonparametric, expansion |
| 4              | sigmoidal, shift, logistic, treatment, loop, testing, net, quantify, razor, adversarial |

Table 6: Top 10 words of 5 randomly selected topics.

Table 7: Top 5 words of 5 randomly selected query words.

| Query Word | Adjacent-Encoder                                                                 |
|------------|----------------------------------------------------------------------------------|
| solution   | call, application, problem, solve, provide                                       |
| neighbor   | general, framework, provide, model, knowledge                                    |
| modeling   | learning, task, computational, include, general                                   |
| supervised | recognition, introduce, call, include, paper                                     |

| Query Word | Adjacent-Encoder-X                                                               |
|------------|----------------------------------------------------------------------------------|
| dna        | protein, produce, attach, promoter, examination                                  |
| production | unpredictable, manufacture, inventory, discrete-event, key                       |
| binary     | bit, general, term, include, paper                                               |
| encode     | intelligent, describe, version, representation, form                             |
| easily     | associate, problem, solve, consider, provide                                     |

Table 8: Classification accuracy of model variants on ML dataset when $m = 64$ and $K = 10$.

| Model                                      | Transductive | Inductive |
|--------------------------------------------|--------------|-----------|
| Adjacent-Encoder                           | 0.864        | 0.761     |
| Denoising Adjacent-Encoder                 | 0.855        | 0.780     |
| Contractive Adjacent-Encoder               | 0.856        | 0.780     |
| K-Sparse Adjacent-Encoder                  | 0.847        | 0.765     |
| Adjacent-Encoder-X                         | 0.857        | 0.836     |
| Denoising Adjacent-Encoder-X               | 0.865        | 0.839     |
| Contractive Adjacent-Encoder-X             | 0.872        | 0.844     |
| K-Sparse Adjacent-Encoder-X                | 0.866        | 0.823     |

Conclusion

We propose Adjacent-Encoder and Adjacent-Encoder-X, neural topic models that learn unified representations for networked documents. Adjacent-Encoder incorporates the network structure implicitly, with similar number of parameters as Auto-Encoder family, yet outperforms the latter. Adjacent-Encoder-X that models the network structure explicitly performs even better. Empirical analysis on public datasets support these findings, showcasing the effectiveness of factoring network structure for neural topic modeling. The model extensions, such as denoising, contractive, and sparsity, further improve the performance.

Acknowledgments

This research is supported by the National Research Foundation, Prime Minister’s Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).
References

Agibetov, A., and Samwald, M. 2018. Global and local evaluation of link prediction tasks with neural embeddings. CoRR abs/1807.10511.

Bai, H.; Chen, Z.; Lyu, M. R.; King, I.; and Xu, Z. 2018. Neural relational topic models for scientific article analysis. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM ’18, 27–36. New York, NY, USA: ACM.

Bishop, C. M. 1995. Neural Networks for Pattern Recognition. New York, NY, USA: Oxford University Press, Inc.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3:993–1022.

Bouma, G. 2009. Normalized (pointwise) mutual information in collocation extraction. In From Form to Meaning: Processing Texts Automatically, Proceedings of the Biennial GSCL Conference 2009, volume Normalized, 31–40.

Cao, S.; Lu, W.; and Xu, Q. 2015. Grarep: Learning graph representations with global structural information. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM ’15, 891–900. New York, NY, USA: ACM.

Chang, J., and Blei, D. M. 2009. Relational topic models for document networks. In Dyk, D. A. V., and Welling, M., eds., AISTATS, volume 5 of JMLR Proceedings, 81–88. JMLR.org.

Chen, Y., and Zaki, M. J. 2017. Kate: K-competitive autoencoder for text. In Proceedings of the ACM SIGKDD International Conference on Data Mining and Knowledge Discovery.

Evert, S. 2010. Google web 1t 5-grams made easy. In Proceedings of the NAACL HLT 2010 Sixth Web As Corpus Workshop, WAC-6 ’10, 32–40. Stroudsburg, PA, USA: Association for Computational Linguistics.

García-Durán, A., and Niepert, M. 2017. Learning graph representations with embedding propagation. CoRR abs/1710.03059.

Grover, A., and Leskovec, J. 2016. Node2vec: Scalable feature learning for networks. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, 855–864. New York, NY, USA: ACM.

Kingma, D. P., and Welling, M. 2014. Auto-encoding variational bayes. In ICLR.

Kipf, T. N., and Welling, M. 2016. Variational graph auto-encoders. NIPS Workshop on Bayesian Deep Learning.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In ICLR. OpenReview.net.

Le, T. M., and Lauw, H. W. 2014. Probabilistic latent document network embedding. In IEEE International Conference on Data Mining.

Makhzani, A., and Frey, B. J. 2013. k-sparse autoencoders. CoRR abs/1312.5663.

McCallum, A. K.; Nigam, K.; Rennie, J.; and Seymore, K. 2000. Automating the construction of internet portals with machine learning. Inf. Retr. 3(2):127–163.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14, 701–710. New York, NY, USA: ACM.

Rifai, S.; Vincent, P.; Muller, X.; Glorot, X.; and Bengio, Y. 2011. Contractive auto-encoders: Explicit invariance during feature extraction. In Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML’11, 833–840. USA: Omnipress.

Rumelhart, D. E.; Hinton, G. E.; and McClelland, J. L. 1986. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. Cambridge, MA, USA: MIT Press. chapter A General Framework for Parallel Distributed Processing, 45–76.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The graph neural network model. Trans. Neur. Netw. 20(1):61–80.

Srivastava, A., and Sutton, C. 2017. Autoencoding variational inference for topic models. In ICLR.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, WWW ’15, 1067–1077. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee.

van der Maaten, L., and Hinton, G. 2008. Visualizing data using t-sne.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In NIPS Workshop on Bayesian Deep Learning.

Wang, H.; Shi, X.; and Yeung, D. 2017. Relational deep learning: A deep latent variable model for link prediction. In AAAI, 2688–2694.

Zhu, S.; Yu, K.; Chi, Y.; and Gong, Y. 2007. Combining content and link for classification using matrix factorization. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR ’07, 487–494. New York, NY, USA: ACM.

6745