Topic Modeling on Document Networks with Adjacent-Encoder

Ce Zhang, Hady W. Lauw
School of Information Systems
Singapore Management University
{cezhang.2018, hadywlauw}@smu.edu.sg

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 often 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 germane, because of their importance and wide applicability.

Problem. Let $G = (D, \mathcal{E})$ be a given document network. $D = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_N\}$ is a set of documents, where each document $\mathbf{d} \in \mathbb{R}^{|V|}$ 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 $\varepsilon_{ij} = 1$ indicates document $\mathbf{d}_i$ links to $\mathbf{d}_j$, and $\varepsilon_{ij} = 0$ otherwise. Here we model an undirected network, i.e., $\varepsilon_{ij} = \varepsilon_{ji}$ 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 $\mathbf{d}$, its neighbors are those directly linked to $\mathbf{d}$. For simplicity, we use $\mathcal{N}(\mathbf{d}) = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_k\}$ to denote $\mathbf{d}$'s neighbors. The definition of neighborhood here is reflexive, i.e., we also regard $\mathbf{d}$ as its own neighbor, $\mathbf{d} \in \mathcal{N}(\mathbf{d})$ 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-

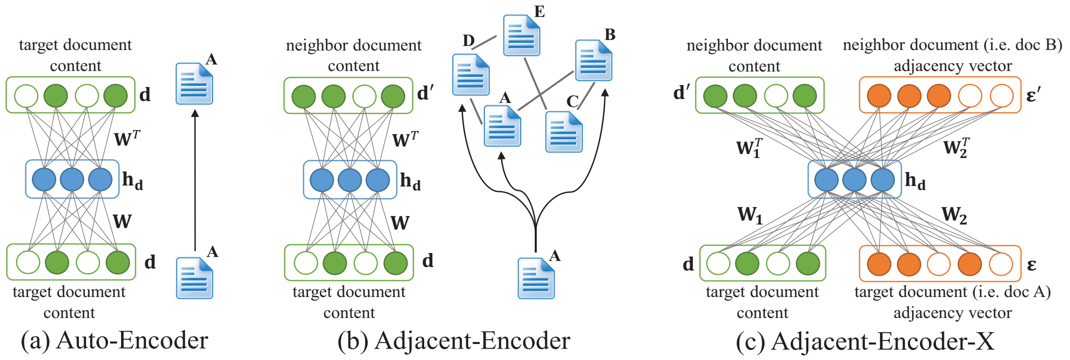


Figure 1: Comparison among Auto-Encoder, Adjacent-Encoder, and Adjacent-Encoder-X.

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 contrast to LDA's mixture assumption. These variants reconstruct only the input document. *AdjEnc* extends this to networked documents via neighbor competition for topic propagation and neighbor reconstruction for semantic capture.

In the context of topic models, there have been extensions of LDA to cover document networks. Relational Topic Model (RTM) (Chang and Blei 2009) regards links as binary variables conditioned on the topic distributions. PLANE (Le and Lauw 2014) extracts topics and 2D visualization coordinates simultaneously. NRTM (Bai et al. 2018) extends VAE to document networks, outperforming another model RDL (Wang, Shi, and Yeung 2017) that extends DAE. These models capture only the first-order neighborhood. In addition to developing a neural approach for document network embodied by *AdjEnc*, we also consider higher-order neighborhood by modeling adjacency matrix explicitly in *AdjEnc-X*.

There are yet other models that learn representations for vertices of a graph, but they are not topic models per se, as they are not devised to model topic-word associations 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 embedddings 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).

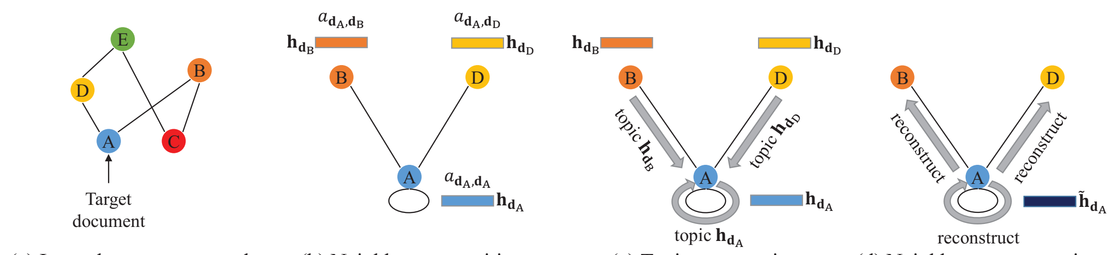
Model Architecture and Analysis

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 $\mathbf{h_d}$ for the input document $\mathbf{d}$ at the encoder: $\mathbf{h_d} = f(\mathbf{Wd}+\mathbf{b})$. The decoder reconstructs the original content of the input document by $\hat{\mathbf{d}} = f'(\mathbf{W}'\mathbf{h_d}+\mathbf{c})$. Here $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{c} \in \mathbb{R}^{|V|}$ are biases, $\mathbf{W} \in \mathbb{R}^{m\times|V|}$ and $\mathbf{W}' \in \mathbb{R}^{|V|\times m}$ are encoder and decoder parameters. Typically we use weight tying ($\mathbf{W}' = \mathbf{W}^T$) as regularization. V is the vocabulary, and m is the number of hidden neurons. By minimizing the reconstruction error, we obtain $\mathbf{h_d}$ as topic representations.

Adjacent-Encoder

Fig. 1 contrasts our proposed models *Adjacent-Encoder* (Fig. 1b) and *Adjacent-Encoder-X* (Fig. 1c) with tradi-


E
D
B
A
C
Target document
$a_{\mathbf{d}_A,\mathbf{d}_B}$
$\mathbf{h}_{\mathbf{d}_B}$
$a_{\mathbf{d}_A,\mathbf{d}_D}$
$\mathbf{h}_{\mathbf{d}_D}$
$a_{\mathbf{d}_A,\mathbf{d}_A}$
$\mathbf{h}_{\mathbf{d}_A}$
topic $\mathbf{h}_{\mathbf{d}_B}$
topic $\mathbf{h}_{\mathbf{d}_D}$
topic $\mathbf{h}_{\mathbf{d}_A}$
reconstruct
reconstruct
reconstruct
$\tilde{\mathbf{h}}_{\mathbf{d}_A}$

(a) Input document network (b) Neighbor competition (c) Topic propagation (d) Neighbor reconstruction

Figure 2: Illustration of neighbor competition, topic propagation, and neighbor reconstruction.

tional Auto-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.

Neighbor 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 $\mathbf{d} \in \mathbb{R}^{|V|}$, 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 $\mathbf{d}$. For a target document $\mathbf{d}$, we learn its hidden vector $\mathbf{h_d}$ at the feedforward phase by $\mathbf{h_d} = tanh(\mathbf{Wd} + \mathbf{b})$. The attention coefficients between $\mathbf{d}$'s neighbors and itself are $a_{\mathbf{d},\mathbf{d}'}$ as shown in Fig. 2b.

$$e_{\mathbf{d},\mathbf{d}'} = \mathbf{h}_\mathbf{d}^T\mathbf{h}_{\mathbf{d}'}, \qquad a_{\mathbf{d},\mathbf{d}'} = \frac{\exp(e_{\mathbf{d},\mathbf{d}'})}{\sum_{\mathbf{d}'\in\mathcal{N}(\mathbf{d})}\exp(e_{\mathbf{d},\mathbf{d}'})} \tag{1}$$

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

Links among documents indicate a shared relationship. Thus we propagate the topics of neighbors to the target document $\mathbf{d}$, 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.

$$\tilde{\mathbf{h}}_\mathbf{d} = \sum_{\mathbf{d}'\in\mathcal{N}(\mathbf{d})} a_{\mathbf{d},\mathbf{d}'}\mathbf{h}_{\mathbf{d}'}. \tag{2}$$

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 $\tilde{\mathbf{h}}_\mathbf{d}$ 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.

$$\hat{\mathbf{d}} = sigmoid(\mathbf{W}^T\tilde{\mathbf{h}}_\mathbf{d} + \mathbf{c}) \tag{3}$$

where $\hat{\mathbf{d}}$ is the reconstruction document. We use binary cross-entropy as the loss function.

$$l(\mathbf{d}', \hat{\mathbf{d}}) = -\sum_{i\in V}[d'_i \log(\hat{d}_i) + (1 - d'_i)\log(1 - \hat{d}_i)]. \tag{4}$$

Again, $\mathbf{d}' \in \mathcal{N}(\mathbf{d})$ is one of the adjacent neighbors. We have each target document $\mathbf{d}$ 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 $\mathbf{h_d} = tanh(\mathbf{Wd}+\mathbf{b})$. Here $\mathbf{h_d}$ 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 $\mathcal{E}$ represents the network structure. The i^{th} row (or column) ε_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 ε as another input in addition to the textual content $\mathbf{d}$. The hidden vector of the feedforward phase can be computed by $\mathbf{h_d} = tanh(\mathbf{W}_1\mathbf{d} +$

Model	#parameters
Adjacent-Encoder	$m\|V\| + m + \|V\|$
Adjacent-Encoder-X	$m\|V\| + mN + m + \|V\| + N$
AE, DAE, CAE, KSAE, KATE	$m\|V\| + m + \|V\|$
VAE	$3m\|V\| + 2m + \|V\|$

Table 1: Number of parameters.

Name	#classes	#documents	#edges	vocabulary
DS	9	570	1336	3085
HA	6	223	515	2073
ML	7	1980	5748	4431
PL	9	1553	4851	4105

Table 2: Dataset statistics.

$\mathbf{W}_2\boldsymbol{\varepsilon}+\mathbf{b})$. Here $\mathbf{W}_1 \in \mathbb{R}^{m\times|V|}$ and $\mathbf{W}_2 \in \mathbb{R}^{m\times N}$ are parameters for textual content and adjacency vector respectively. $\mathbf{b} \in \mathbb{R}^m$ is bias. N is the total number of documents.

The remaining process of neighbor competition for *Adjacent-Encoder-X* is similar to *Adjacent-Encoder*. Thereafter, we obtain the aggregate hidden vector $\tilde{\mathbf{h}}_{\mathbf{d}}$.

Neighbor Reconstruction. We still have each target document reconstruct its adjacent neighbors, but now in terms of both textual content and adjacency vector.

$$\hat{\mathbf{d}} = sigmoid(\mathbf{W}_1^T\tilde{\mathbf{h}}_{\mathbf{d}} + \mathbf{c}_1), \quad \hat{\boldsymbol{\varepsilon}} = sigmoid(\mathbf{W}_2^T\tilde{\mathbf{h}}_{\mathbf{d}} + \mathbf{c}_2) \tag{5}$$

where weight tying is used, and $\mathbf{c}_1 \in \mathbb{R}^{|V|}$ and $\mathbf{c}_2 \in \mathbb{R}^N$ are biases. The loss function for textual content is given by (4), and the loss function for adjacency vector is given below.

$$l(\boldsymbol{\varepsilon}', \hat{\boldsymbol{\varepsilon}}) = -\sum_{i\in N}[\varepsilon_i' \log(\hat{\varepsilon}_i) + (1-\varepsilon_i')\log(1-\hat{\varepsilon}_i)]. \tag{6}$$

$\boldsymbol{\varepsilon}'$ represents the adjacency vector of $\mathbf{d}$'s neighbors. Each target document reconstructs its neighbors in these two aspects, generating the total loss function $l = l(\mathbf{d}', \hat{\mathbf{d}}) + l(\boldsymbol{\varepsilon}', \hat{\boldsymbol{\varepsilon}})$.

Inference. Upon convergence we encode a testing document by $\mathbf{h}_{\mathbf{d}} = tanh(\mathbf{W}_1\mathbf{d}+\mathbf{W}_2\boldsymbol{\varepsilon}+\mathbf{b})$. $\mathbf{h}_{\mathbf{d}}$ is the topic representation encompassing text content and network structure.

Complexity Analysis

Model Complexity. Table 1 lists the parameter counts for our models and the Auto-Encoder family. For *Adjacent-Encoder*, we set $\mathbf{W}' = \mathbf{W}^T$ of dimensionality $m|V|$. The only other parameters are biases of size m and $|V|$. Note that compared to other Auto-Encoder models (AE, DAE, CAE, KSAE, KATE), *Adjacent-Encoder* does not add extra parameters as it models the network structure implicitly. For *Adjacent-Encoder-X*, because the adjacency matrix is another input in addition to the document content, the number of parameters is now $m|V| + mN + N + m + |V|$.

Computational Complexity. We use F to denote the number of input features ($|V|$ and $|V| + N$ respectively for *Adjacent-Encoder* and *Adjacent-Encoder-X*). The feedforward complexity for each target document is $\mathcal{O}(mF)$. For neighbor competition and topic propagation, let $\deg_{max}$ denote the maximum number of neighbors in the network. The complexity of each target document is $\mathcal{O}(m\deg_{max})$. For neighbor reconstruction, we reconstruct all adjacent neighbors, thus we have $\mathcal{O}(mF\deg_{max})$. Putting all three components together, for each target document, we obtain $\mathcal{O}(mF + m\deg_{max} + mF\deg_{max})$ 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 $sigmoid$, while those for VAE and KATE are $tanh$ (hidden) and $sigmoid$ (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 hyperparameter for RTM and PLANE. For KSAE and KATE, we set

[1]http://people.cs.umass.edu/mccallum/data/cora-classify.tar.gz
[2]https://github.com/cezhang01/Adjacent-Encoder

	Transductive Learning											
Model	Document Classification				Document Clustering				Link Prediction			
	DS	HA	ML	PL	DS	HA	ML	PL	DS	HA	ML	PL
Adjacent-Encoder	0.739	0.842	**0.864**	0.772	**0.470**	0.540	0.564	0.388	**0.396**	**0.331**	0.226	0.237
Adjacent-Encoder-X	**0.744**	**0.846**	0.857	**0.780**	0.445	**0.548**	**0.571**	**0.392**	0.374	0.326	**0.251**	**0.271**
AE	0.558	0.688	0.739	0.616	0.250	0.315	0.368	0.230	0.144	0.195	0.107	0.102
DAE	0.656	0.799	0.790	0.694	0.372	0.409	0.441	0.278	0.204	0.296	0.121	0.147
CAE	0.558	0.685	0.741	0.620	0.261	0.309	0.371	0.228	0.145	0.188	0.108	0.103
VAE	0.652	0.789	0.796	0.679	0.356	0.394	0.447	0.286	0.193	0.283	0.122	0.135
KSAE	0.537	0.672	0.710	0.581	0.245	0.295	0.345	0.222	0.136	0.182	0.092	0.088
KATE	0.628	0.808	0.762	0.651	0.325	0.378	0.342	0.267	0.174	0.267	0.095	0.114
ProdLDA	0.637	0.780	0.764	0.631	0.374	0.460	0.423	0.289	0.162	0.324	0.080	0.095
RTM	0.543	0.637	0.663	0.574	0.082	0.094	0.126	0.127	0.117	0.194	0.072	0.075
PLANE	0.690	0.799	0.750	0.648	0.417	0.406	0.439	0.288	0.284	0.226	0.107	0.160
NRTM	0.591	0.816	0.549	0.503	0.313	0.404	0.137	0.190	0.149	0.221	0.036	0.049
VGAE	0.671	0.827	0.807	0.718	0.335	0.362	0.495	0.308	0.285	0.265	0.132	0.171

Table 3: Transductive results on document classification (left), clustering (middle), and link prediction (right) at $m = 64$.

	Inductive Learning											
Model	Document Classification				Document Clustering				Link Prediction			
	DS	HA	ML	PL	DS	HA	ML	PL	DS	HA	ML	PL
Adjacent-Encoder	0.588	0.830	0.761	0.654	**0.417**	**0.551**	0.477	0.328	**0.421**	**0.462**	0.285	0.218
Adjacent-Encoder-X	**0.640**	**0.845**	**0.836**	**0.724**	0.416	0.489	**0.522**	**0.363**	0.400	0.427	**0.363**	**0.322**
AE	0.405	0.580	0.632	0.509	0.213	0.337	0.340	0.248	0.185	0.233	0.181	0.129
DAE	0.516	0.749	0.732	0.595	0.375	0.436	0.415	0.299	0.347	0.286	0.259	0.198
CAE	0.400	0.573	0.644	0.519	0.212	0.279	0.362	0.253	0.192	0.232	0.185	0.132
VAE	0.491	0.785	0.738	0.594	0.373	0.361	0.404	0.300	0.391	0.346	0.243	0.192
KSAE	0.390	0.569	0.614	0.491	0.269	0.319	0.334	0.232	0.188	0.238	0.148	0.111
KATE	0.484	0.800	0.712	0.573	0.321	0.440	0.354	0.290	0.277	0.336	0.205	0.178
ProdLDA	0.202	0.401	0.184	0.158	0.302	0.292	0.399	0.306	0.220	0.297	0.192	0.140
RTM	0.327	0.498	0.652	0.564	0.000	0.046	0.091	0.048	0.260	0.276	0.210	0.149
PLANE	0.282	0.544	0.275	0.218	0.162	0.192	0.000	0.000	0.306	0.345	0.176	0.134
NRTM	0.456	0.811	0.482	0.408	0.339	0.398	0.167	0.207	0.076	0.097	0.020	0.049
VGAE	0.509	0.748	0.736	0.607	0.280	0.185	0.442	0.291	0.315	0.309	0.237	0.274

Table 4: Inductive results on document classification (left), clustering (middle), and link prediction (right) at $m = 64$.

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.

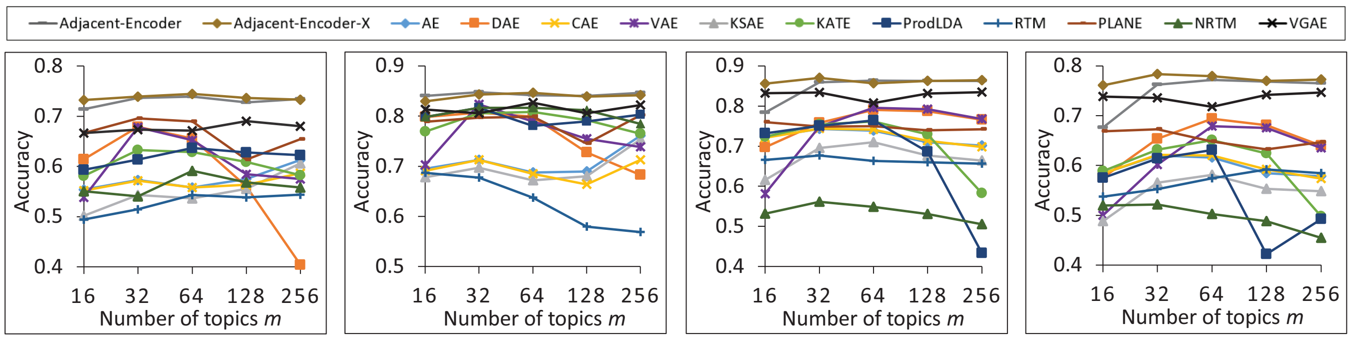
Transductive vs. Inductive Learning. There are two scenarios in which we can apply the models. In the transductive setting, the objective is to derive topic representations of the documents already in the corpus. In this case, all documents in the corpus are present during training. Conversely, in the 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

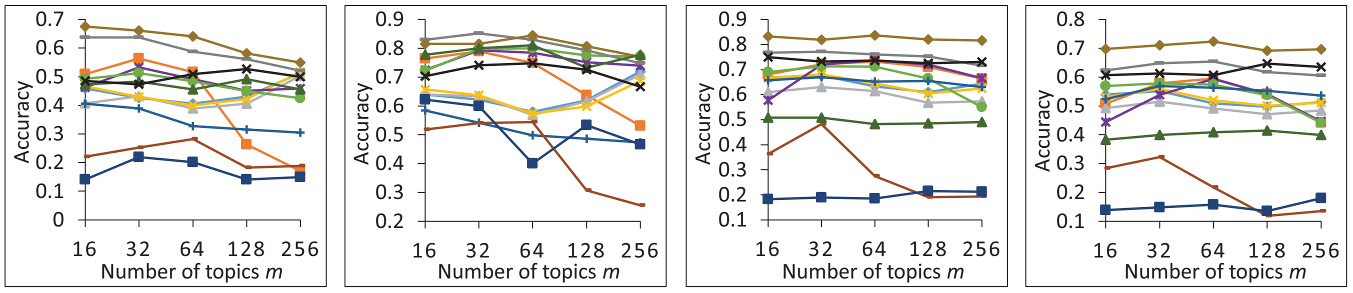
For validating the derived document representations, we rely on three evaluative tasks. The first two are document classification and clustering, evaluated via class labels (these are never part of any learning). The last is link prediction.

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-

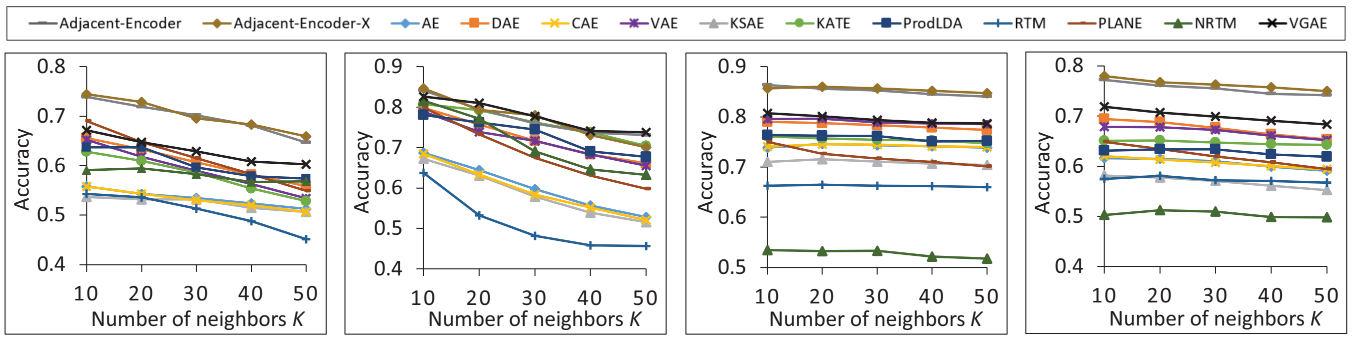


(a) Transductive on DS (b) Transductive on HA (c) Transductive on ML (d) Transductive on PL

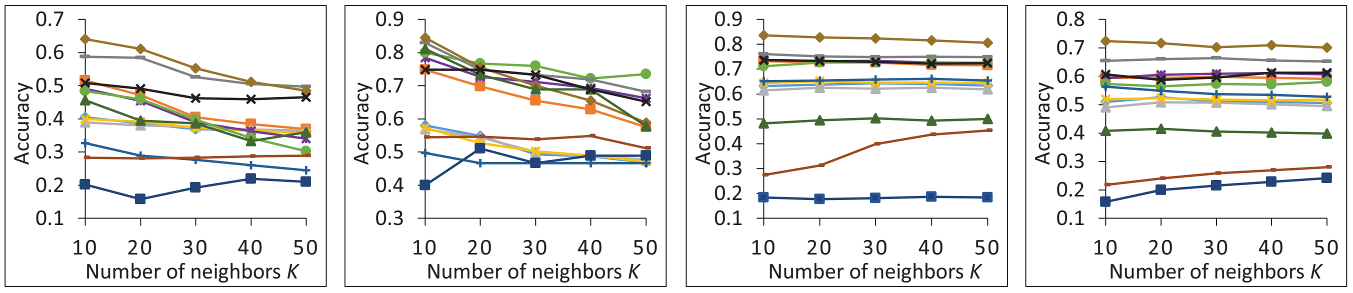


(e) Inductive on DS (f) Inductive on HA (g) Inductive on ML (h) Inductive on PL

Figure 3: Transductive and inductive classification accuracy at $K = 10$ when varying the number of topics m.



(a) Transductive on DS (b) Transductive on HA (c) Transductive on ML (d) Transductive on PL



(e) Inductive on DS (f) Inductive on HA (g) Inductive on ML (h) Inductive on PL

Figure 4: Transductive and inductive classification accuracy at $m = 64$ when varying the number neighbors K.

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 $P(\varepsilon_{ij} = 1|\mathbf{h}_{\mathbf{d}_i}, \mathbf{h}_{\mathbf{d}_j}) \propto \exp(-||\mathbf{h}_{\mathbf{d}_i} - \mathbf{h}_{\mathbf{d}_j}||^2)$. 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[3], we conduct experiments on topic analysis.

[3]VGAE is not a topic model, and is not included in this analysis.

Model	PMI			
	DS	HA	ML	PL
Adjacent-Encoder	**2.360**	**2.054**	2.180	**2.499**
Adjacent-Encoder-X	1.872	1.887	**2.337**	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	1.132
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

Table 5: Topic Coherence PMI when $m = 64$.

Topic Coherence. Our topic-word association is given by $\mathbf{W}$ for *Adjacent-Encoder*, and $\mathbf{W}_1$ for *Adjacent-Encoder-X*. We use PMI (Bouma 2009), defined as $PMI(w_i, w_j) = \log \frac{p(w_i,w_j)}{p(w_i)p(w_j)}$, to evaluate the coherence of predicted words. Using *Google Web 1T 5-gram Version 1* (Evert 2010), $p(w_i)$ is evaluated from 1-gram 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.

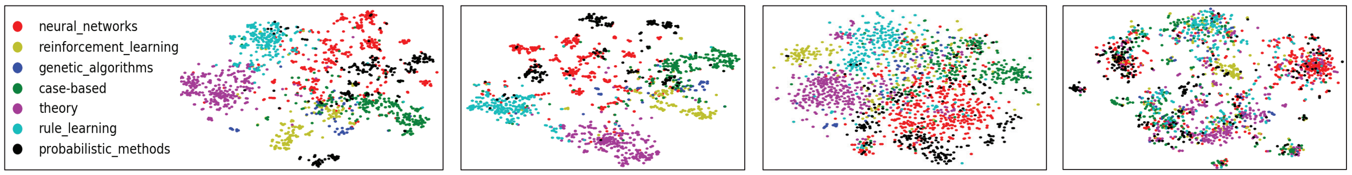
Table 5 shows that network-based models tend to perform well, benefitting from document relatedness in addition to text content. *Adjacent-Encoder* has higher topic coherence than *Adjacent-Encoder-X* except for ML, presumably in modeling the adjacency vector explicitly the latter may reduce the reconstruction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



(a) Adjacent-Encoder (b) Adjacent-Encoder-X (c) KATE (d) NRTM

Figure 5: t-SNE visualization on ML dataset. (best seen in color)

Topic	Adjacent-Encoder
1	maze, markov, mdp, observable, minute, intractable, severe, markovian, pomdp, analog
2	move, 0-1, image, nearest, promoter, neighbor, grid, k-nearest, analogy, sketch
3	reward, influence, recurrent, credit, exploratory, max, reinforcement, net, reactive, policy
4	inference, hmm, graphical, practitioner, translation, defense, methodological, causal, probable, assist
5	pair, net, coordination, backpropagation, stronger, broad, network, classic, pendulum, multiclass
Topic	**Adjacent-Encoder-X**
1	mdp, policy, clarify, identical, pomdp, observable, tradeoff, consequence, noisy, larger
2	cart, selective, exploratory, estimator, phoneme, categorization, stability, multiclass, terminate, axis-parallel
3	parent, graph, substructure, overlap, graphical, load, emulate, integration, fashion, generalisation
4	ica, toronto, detector, blind, maximization, derivation, facial, pca, nonparametric, expansion
5	sigmoidal, shift, logistic, treatment, loop, testing, net, quantify, razor, adversarial

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

Query Word	Adjacent-Encoder
solution	call, application, problem, solve, provide
neighbor	nearest, k-nearest, paper, describe, artificial
modeling	general, framework, provide, model, knowledge
supervised	learning, task, computational, include, general
speech	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 7: Top 5 words of 5 randomly selected query words.

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

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

Extensions and Variants

We investigate the complementarity and potential extensibility of our models by combining proposed architecture with other concepts previously used to enhance AE. For denoising variant, we add Gaussian noise of 0.25 std.dev. to input documents. For our contractive variant, we add Frobenius norm of Jacobian matrix to loss function of our models. For K-sparse variant, as in KSAE and KATE, we keep the values of $\frac{k}{2}$ top positive and $\frac{k}{2}$ top negative hidden neurons and zero others after neighbor competition and before reconstruction.

We test these variants for document classification on ML dataset. We set $m = 64$ and $K = 10$ (for KNN). Table 8 shows some enhancements tend to produce positive outcomes. Further adding denoising to original models shows the value of denoising. The regularization on loss function by the contractive enhancement provides better results. Indeed *K-Sparse Adjacent-Encoder(-X)* learn competitive representations with original models in terms of classification.

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 on 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 autoencoders. *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 *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P.-A. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* 11:3371–3408.

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.