Document Informed Neural Autoregressive Topic Models with Distributional Prior

Pankaj Gupta,1,2 Yatin Chaudhary,1 Florian Buettner,1 Hinrich Schütze2
1Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
2CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, yatin.chaudhary, buettner.florian}@siemens.com
pankaj.gupta@campus.lmu.de — inquiries@cislmu.org

Abstract

We address two challenges in topic models: (1) Context information around words helps in determining their actual meaning, e.g., “networks” used in the contexts artificial neural networks vs. biological neuron networks. Generative topic models infer topic-word distributions, taking no or only little context into account. Here, we extend a neural autoregressive topic model to exploit the full context information around words in a document in a language modeling fashion. The proposed model is named as iDocNADE. (2) Due to the small number of word occurrences (i.e., lack of context) in short text and data sparsity in a corpus of few documents, the application of topic models is challenging on such texts. Therefore, we propose a simple and efficient way of incorporating external knowledge into neural autoregressive topic models: we use embeddings as a distributional prior. The proposed variants are named as DocNADEe and iDocNADEe.

We present novel neural autoregressive topic model variants that consistently outperform state-of-the-art generative topic models in terms of generalization, interpretability (topic coherence) and applicability (retrieval and classification) over 7 long-text and 8 short-text datasets from diverse domains.

Introduction

Probabilistic topic models, such as LDA (Blei, Ng, and Jordan 2003), Replicated Softmax (RSM) (Salakhutdinov and Hinton 2009) and Document Autoregressive Neural Distribution Estimator (DocNADE) (Larochelle and Lauly 2012) are often used to extract topics from text collections and learn document representations to perform NLP tasks such as information retrieval (IR), document classification or summarization.

To motivate our first task of incorporating full contextual information, assume that we conduct topic analysis on a collection of research papers from NIPS conference, where one of the popular terms is “networks”. However, without context information (nearby and/or distant words), its actual meaning is ambiguous since it can refer to such different concepts as artificial neural networks in computer science or biological neural networks in neuroscience or Computer data networks in telecommunications. Given the context, one can determine the actual meaning of “networks”, for instance, “Extracting rules from artificial neural networks with distributed representations”, or “Spikes from the presynaptic neurons and postsynaptic neurons in small networks” or “Studies of neurons or networks under noise in artificial neural networks” or “Packet Routing in Dynamically Changing Networks”.

Generative topic models such as LDA or DocNADE infer topic-word distributions that can be used to estimate a document likelihood. While basic models such as LDA do not account for context information when inferring these distributions, more recent approaches such as DocNADE achieve amplified word and document likelihoods by accounting for words preceding a word of interest in a document. More specifically, DocNADE (Larochelle and Lauly 2012; Zheng, Zhang, and Larochelle 2016) (Figure 1, Left) is a probabilistic graphical model that learns topics over sequences of words, corresponding to a language model (Manning and Schütze 1999; Bengio et al. 2003) that can be interpreted as a neural network with several parallel hidden layers. To predict the word $v_i$, each hidden layer $h_i$ takes as input the sequence of preceding words $v_{<i}$. However, it does not take into account the following words $v_{i+1}$ in the sequence. Inspired by bidirectional language models (Mousa and Schuller 2017) and recurrent neural networks (Elman 1990; Gupta, Schütze, and Andrassy 2016; Vu et al. 2016b; 2016a), trained to predict a word (or label) depending on its full left and right contexts, we extend DocNADE and incorporate full contextual information (all words around $v_i$) at each hidden layer $h_i$ when predicting the word $v_i$ in a language modeling fashion with neural topic modeling.

While this is a powerful approach for incorporating contextual information in particular for long texts and corpora with many documents, learning contextual information remains challenging in topic models with short texts and few documents, due to (1) limited word co-occurrences or little context and (2) significant word non-overlap in such short texts. However, distributional word representations (i.e. word embeddings) have shown to capture both the semantic and syntactic relatedness in words and demonstrated impressive performance in natural language processing (NLP) tasks. For example, assume that we conduct topic analysis over the two short text fragments: “Goldman shares drop sharply downgrade” and “Falling market homes...
Figure 1: DocNADE (left), iDocNADE (middle) and DocNADEe (right) models. Blue colored lines signify the connections that share parameters. The observations (double circle) for each word $v_i$ are multinomial. Hidden vectors in green and red colors identify the forward and backward network layers, respectively. Symbols $\vec{v}_i$ and $\overrightarrow{v}_i$ represent the autoregressive conditionals $p(v_i | v_{<i})$ and $p(v_i | v_{>i})$, respectively. Connections between each $v_i$ and hidden units are shared, and each conditional $\overrightarrow{v}_i$ (or $\overleftarrow{v}_i$) is decomposed into a tree of binary logistic regressions, i.e. hierarchical softmax.

Neural Autoregressive Topic Models

RSM (Salakhutdinov and Hinton 2009), a probabilistic undirected topic model, is a generalization of the energy-based Restricted Boltzmann Machines RBM (Hinton 2002) that can be used to model word counts. NADE (Larochelle and Murray 2011) decomposes the joint distribution of observations into autoregressive conditional distributions, modeled using non-linear functions. Unlike for RBM/RSM, this leads to tractable gradients of the data negative log-likelihood but can only be used to model binary observations.

DocNADE (Figure 1, Left) is a generative neural autoregressive topic model to account for word counts, inspired by RSM and NADE. For a document $v = [v_1, ..., v_D]$ of size $D$, it models the joint distribution $p(v)$ of all words $v_i$, where $v_i \in \{1, ..., K\}$ is the index of the $i$th word in the dictionary of vocabulary size $K$. This is achieved by decomposing it as a product of conditional distributions i.e. $p(v) = \prod_{i=1}^{D} p(v_i | v_{<i})$ and computing each autoregressive conditional $p(v_i | v_{<i})$ via a feed-forward neural network for $i \in \{1, ..., D\}$,

$$p(v_i = w | v_{<i}) = \frac{\exp(g(c + \sum_{k<k} W_{:,v_k} \overrightarrow{v}_k) + b_w + U_w \cdot \overleftarrow{v}_i)}{\sum_{w'} \exp(g(c + \sum_{k<k} W_{:,v_k} \overrightarrow{v}_k) + b_{w'} + U_{w'} \cdot \overleftarrow{v}_i)}$$

(1)

where $v_{<i} \in \{v_1, ..., v_{i-1}\}$, $g(\cdot)$ is a non-linear activation function, $W \in \mathbb{R}^{H \times K}$ and $U \in \mathbb{R}^{K \times H}$ are weight matrices, $c \in \mathbb{R}^H$ and $b \in \mathbb{R}^K$ are bias parameter vectors. $H$ is the number of hidden units (topics). $W_{:,v_{<i}}$ is a matrix made of the $i - 1$ first columns of $W$. The probability of the word $v_i$ is thus computed using a position-dependent hidden layer $\overrightarrow{v}_i$ that learns a representation based on all previous words $v_{<i}$; however it does not incorporate the following words $v_{>i}$. Taken together, the log-likelihood of any document $v$ of arbitrary length can be computed as:

$$\mathcal{L}_{DocNADE}(v) = \sum_{i=1}^{D} \log p(v_i | v_{<i})$$

(2)
iDocNADE (Figure 1, Right), our proposed model, accounts for the full context information (both previous \(v_{<i}\) and following \(v_{>i}\) words) around each word \(v_i\) for a document \(v\). Therefore, the log-likelihood \(L^{iDocNADE}\) for a document \(v\) in iDocNADE is computed using forward and backward language models as:

\[
\log p(v) = \frac{1}{2} \sum_{i=1}^{D} \left( \log p(v_i | v_{<i}) + \log p(v_i | v_{>i}) \right) \tag{3}
\]

i.e., the mean of the forward \(\mathcal{F}\) and backward \(\mathcal{B}\) log-likelihoods. This is achieved in a bi-directional language modeling and feed-forward fashion by computing position dependent forward \(\mathcal{F}(h_i)\) and backward \(\mathcal{B}(h_i)\) hidden layers for each word \(i\), as:

\[
\mathcal{F}(h_i) = g(\mathcal{C} + \sum_{k<i} W_k \cdot v_k) \tag{4}
\]

\[
\mathcal{B}(h_i) = g(\mathcal{C} + \sum_{k>i} W_k \cdot v_k) \tag{5}
\]

where \(\mathcal{C} \in \mathbb{R}^H\) and \(\mathcal{C} \in \mathbb{R}^H\) are bias parameters in forward and backward passes, respectively. \(H\) is the number of hidden units (topics).

Two autoregressive conditionals are computed for each \(i\)th word using the forward and backward hidden vectors,

\[
p(v_i = u | v_{<i}) = \frac{\exp(\mathcal{F}(U_w \cdot h_i(v_{<i}))}{\sum_{u'} \exp(\mathcal{F}(U_w \cdot h_i(v_{<i}))} \tag{6}
\]

\[
p(v_i = u | v_{>i}) = \frac{\exp(\mathcal{B}(U_w \cdot h_i(v_{>i}))}{\sum_{u'} \exp(\mathcal{B}(U_w \cdot h_i(v_{>i}))} \tag{7}
\]

for \(i \in [1, ..., D]\) where \(\mathcal{B} \in \mathbb{R}^K\) and \(\mathcal{B} \in \mathbb{R}^K\) are biases in forward and backward passes, respectively. Note that the parameters \(W\) and \(U\) are shared between the two networks.

**Deep DocNADEs with/without Embedding priors:**

We introduce additional semantic information for each word into DocNADE-like models via its pre-trained embedding vector, thereby enabling better textual representations and semantically more coherent topic distributions, in particular for short texts. In its simplest form, we extend DocNADE with word embedding aggregation at each autoregressive step \(k\) to generate a complementary textual representation, i.e., \(\sum_{k<i} E_{vk}\). This mechanism utilizes prior knowledge encoded in a pre-trained embedding matrix \(E_i \in \mathbb{R}^{H \times K}\) when learning task-specific matrices \(W\) and latent representations in DocNADE-like models. The position dependent forward \(\mathcal{F}(h_i)\) and (only in iDocNADEe) backward \(\mathcal{B}(h_i)\) hidden layers for each word \(i\) now depend on \(E\) as:

\[
\mathcal{F}(h_i(v_{<i})) = g(\mathcal{C} + \sum_{k<i} W_k \cdot v_k + \lambda \sum_{k<i} E_{vk}) \tag{8}
\]

\[
\mathcal{B}(h_i(v_{>i})) = g(\mathcal{C} + \sum_{k>i} W_k \cdot v_k + \lambda \sum_{k>i} E_{vk}) \tag{9}
\]

where, \(\lambda\) is a mixture coefficient, determined using validation set. As in equations 6 and 7, the forward and backward autoregressive conditionals are computed via hidden vectors \(\mathcal{F}(h_i(v_{<i}))\) and \(\mathcal{B}(h_i(v_{>i}))\), respectively.

\begin{algorithm}
\caption{Computation of \(\log p(v)\) in \(iDocNADE\) or \(iDocNADEe\) using tree-softmax or full-softmax}
\textbf{Input}: A training document vector \(v\), Embedding matrix \(E\)
\textbf{Parameters}: \{\(\mathcal{B}, \mathcal{B}, \mathcal{C}, \mathcal{C}, W, U\)\}
\textbf{Output}: \(\log p(v)\)
\begin{verbatim}
1: \(\alpha_i \leftarrow \alpha^\kappa\)
2: if iDocNADE then
3: \(\alpha_i \leftarrow \alpha^\kappa + \sum_{i=1}^{D} W_{k,v}\)
4: if iDocNADEe then
5: \(\alpha_i \leftarrow \alpha^\kappa + \sum_{i=1}^{D} W_{k,v} + \lambda \sum_{i=1}^{D} E_{k,v}\)
6: \(q(v) = 1\)
7: for \(i\) from 1 to \(D\) do
8: \(\h_i \leftarrow g(\alpha^\kappa); \ h_i \leftarrow g(\alpha^\kappa)\)
9: if tree-softmax then
10: \(p(v_i | v_{<i}) = 1; \ p(v_i | v_{>i}) = 1\)
11: for \(m\) from 1 to \(|\pi(v_i)|\) do
12: \(p(v_i | v_{<i}) \leftarrow p(v_i | v_{<i}) p(v_i | v_{>i})\)
13: \(p(v_i | v_{>i}) \leftarrow p(v_i | v_{<i}) p(v_i | v_{>i})\)
14: if full-softmax then
15: compute \(p(v_i | v_{<i})\) using equation 6
16: compute \(p(v_i | v_{>i})\) using equation 7
17: \(q(v) \leftarrow q(v) p(v_i | v_{<i}) p(v_i | v_{>i})\)
18: if iDocNADE then
19: \(\alpha' \leftarrow \alpha + W_{k,v}; \ \alpha'' \leftarrow \alpha - W_{k,v}\)
20: if iDocNADEe then
21: \(\alpha' \leftarrow \alpha + W_{k,v} + \lambda E_{k,v}\)
22: \(\alpha'' \leftarrow \alpha - W_{k,v} - \lambda E_{k,v}\)
23: \(\log p(v) \leftarrow \frac{1}{2} \log q(v)\)
\end{verbatim}
\end{algorithm}
for each word $i$ via embedding aggregation of its context $v_{<i}$ (and $v_{>i}$). Similarly, we compute $\vec{h}_i^{(1)}$.

Learning: Similar to DocNADE, the conditionals $p(v_i = w|v_{<i})$ and $p(v_i = w|v_{>i})$ in DocNADEe, iDocNADE or iDocNADEe are computed by a neural network for each word $v_i$, allowing efficient learning of informed representations $\vec{h}_i^1$ and $\vec{h}_i^2$ (or $\vec{h}_i^1(v_{<i})$ and $\vec{h}_i^2(v_{>i})$), as it consists simply of a linear transformation followed by a non-linearity. Observe that the weight $W$ (or prior embedding matrix $E$) is the same across all conditionals and ties contextual observables (blue colored lines in Figure 1) by computing each $\vec{h}_i^1$ or $\vec{h}_i^2$ (or $\vec{h}_i^1(v_{<i})$ and $\vec{h}_i^2(v_{>i})$).

Binary word tree (tree-softmax) to compute conditionals: To compute the likelihood of a document, the autoregressive conditionals $p(v_i = w|v_{<i})$ and $p(v_i = w|v_{>i})$ have to be computed for each word $i \in [1, 2, ..., D]$, requiring time linear in vocabulary size $K$. To reduce computational cost and achieve a complexity logarithmic in $K$ we follow Larochelle and Laulu (2012) and decompose the computation of the conditionals using a probabilistic tree. All words in the documents are randomly assigned to a different leaf in a binary tree and the probability of a word is computed as the probability of reaching its associated leaf from the root. Each left/right transition probability is modeled using a binary logistic regressor with the hidden layer $\vec{h}_1^1$ or $\vec{h}_1^2$ ($\vec{h}_2^1$ or $\vec{h}_2^2$) as its input. In the binary tree, the probability of a given word is computed by multiplying each of the left/right transition probabilities along the tree path.

Algorithm 1 shows the computation of $\log p(v)$ using iDocNADE (or iDocNADEe) structure, where the autoregressive conditionals (lines 14 and 15) for each word $v_i$ are obtained from the forward and backward networks and modeled into a binary word tree, where $\pi(v_i)$ denotes the sequence of binary left/right choices at the internal nodes along the tree path and $l(v_i)$ the sequence of tree nodes on that tree path. For instance, $l(v_{1/2})$ will always be the root of the binary tree and $\pi(v_{1/2})$ will be 0 if the word leaf $v_i$ is in the left subtree or 1 otherwise. Therefore, each of the forward and backward conditionals are computed as:

$$p(v_i = w|v_{<i}) = \prod_{m=1}^{\pi(v_i)} \frac{1}{K} p(\pi(v_i)_m|v_{<i})$$

$$p(v_i = w|v_{>i}) = \prod_{m=1}^{\pi(v_i)} \frac{1}{K} p(\pi(v_i)_m|v_{>i})$$

$$p(\pi(v_i)_m|v_{<i}) = g(\vec{b}_l(v_i)_m + \vec{U}(v_i)_m; \vec{h}_i^(v_{<i}))$$

$$p(\pi(v_i)_m|v_{>i}) = g(\vec{b}_r(v_i)_m + \vec{U}(v_i)_m; \vec{h}_i^(v_{>i}))$$

where $\vec{U} \in \mathbb{R}^{T \times H}$ is the matrix of logistic regressions weights, $T$ is the number of internal nodes in binary tree, and $\vec{b}_l$ and $\vec{b}_r$ are bias vectors.

Each of the forward and backward conditionals $p(v_i = w|v_{<i})$ or $p(v_i = w|v_{>i})$ requires the computation of its own hidden layers $\vec{h}_1^1(v_{<i})$ and $\vec{h}_1^2(v_{>i})$ (or $\vec{h}_2^1(v_{<i})$ and $\vec{h}_2^2(v_{>i})$), respectively. With $H$ being the size of each hidden layer and $D$ the number of words in $v$, computing a single layer requires $O(HD)$, and since there are $D$ hidden layers to compute, a naive approach for computing all hidden layers would be in $O(D^3H)$. However, since the weights in the matrix $W$ are tied, the linear activations $\vec{a}$ and $\vec{a}$ (algorithm 1) can be re-used in every hidden layer and computational complexity reduces to $O(HD)$.

With the trained iDocNADEe (or DocNADE variants), the representation ($\vec{h}^T \in \mathbb{R}^D$) for a new document $v^*$ of size $D^*$ is extracted by summing the hidden representations from the forward and backward networks to account for the context information around each word in the words’ sequence, as

$$\vec{h}^T(v^*) = \sum_{k \leq D^*} \vec{W}_k:v^*_k + \lambda \sum_{k \geq 1} \vec{E}_k:v^*_k$$

(10)

$$\vec{h}^T(v^*) = g(\vec{c} + \sum_{k \leq D^*} \vec{W}_k:v^*_k + \lambda \sum_{k \geq 1} \vec{E}_k:v^*_k)$$

Therefore, $\vec{h}^T(v^*) = \vec{h}^T(v^*)$.

The DocNADE variants without embeddings compute the representation $\vec{h}$ excluding the embedding term $\vec{E}$. Parameters $\{\vec{b}, \vec{b}_l, \vec{b}_r, \vec{h}, \vec{W}, \vec{U}\}$ are learned by minimizing the average negative log-likelihood of the training documents using stochastic gradient descent (algorithm 2).

Evaluation

We perform evaluations on 15 (8 short-text and 7 long-text) datasets of varying size with single/multi-class labeled documents from public as well as industrial corpora. See the supplementary material for the data description, hyper-parameters and grid-search results for generalization and IR tasks. Table 1 shows the data statistics, where 20NS: 20NewsGroups and R21578: Reuters21578. Since, Gupta
<table>
<thead>
<tr>
<th>Data</th>
<th>Train</th>
<th>Val</th>
<th>Test</th>
<th>K</th>
<th>L</th>
<th>C</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>20NSshort</td>
<td>1.3k</td>
<td>0.1k</td>
<td>0.5k</td>
<td>2k</td>
<td>13.5</td>
<td>20</td>
<td>News</td>
</tr>
<tr>
<td>TREC6</td>
<td>5.5k</td>
<td>0.5k</td>
<td>0.5k</td>
<td>2k</td>
<td>9.8</td>
<td>6</td>
<td>Q&amp;A</td>
</tr>
<tr>
<td>R21578title</td>
<td>7.3k</td>
<td>0.5k</td>
<td>3.0k</td>
<td>2k</td>
<td>7.3</td>
<td>90</td>
<td>News</td>
</tr>
<tr>
<td>Subjectivity</td>
<td>8.0k</td>
<td>0.5k</td>
<td>2.0k</td>
<td>2k</td>
<td>23.1</td>
<td>2</td>
<td>Senti</td>
</tr>
<tr>
<td>Polarity</td>
<td>8.5k</td>
<td>0.5k</td>
<td>2.1k</td>
<td>2k</td>
<td>21.0</td>
<td>2</td>
<td>Senti</td>
</tr>
<tr>
<td>TMNtitle</td>
<td>22.8k</td>
<td>2.0k</td>
<td>7.8k</td>
<td>2k</td>
<td>4.9</td>
<td>7</td>
<td>News</td>
</tr>
<tr>
<td>TMN</td>
<td>22.8k</td>
<td>2.0k</td>
<td>7.8k</td>
<td>2k</td>
<td>19</td>
<td>7</td>
<td>News</td>
</tr>
<tr>
<td>AGnewstitle</td>
<td>118k</td>
<td>2.0k</td>
<td>7.6k</td>
<td>2k</td>
<td>6.8</td>
<td>4</td>
<td>News</td>
</tr>
<tr>
<td>Avg (short)</td>
<td>509</td>
<td>5.5</td>
<td>486</td>
<td>59</td>
<td>435</td>
<td>54</td>
<td>417</td>
</tr>
<tr>
<td>Avg (long)</td>
<td>417</td>
<td>6.1</td>
<td>383</td>
<td>65</td>
<td>416</td>
<td>56</td>
<td>396</td>
</tr>
<tr>
<td>Avg (all)</td>
<td>469</td>
<td>5.7</td>
<td>442</td>
<td>62</td>
<td>426</td>
<td>54</td>
<td>407</td>
</tr>
</tbody>
</table>

Table 1: Data statistics of short and long texts as well as small and large corpora from various domains. State-of-the-art comparison in terms of PPL and IR (i.e., IR-precision) for short and long text datasets. The symbols are- L: average text length in number of words, K: dictionary size, C: number of classes, Senti: Sentiment, Avg: average, ‘k’:thousand and ‘f’: multi-label data. PPL and IR (IR-precision) are computed over 200 (T200) topics at retrieval fraction = 0.02. For short-text. L < 25. The underline and bold numbers indicate the best scores in PPL and retrieval task, respectively in FS setting. See Larochelle and Lauly (2012) for LDA (Blei, Ng, and Jordan 2003) performance in terms of PPL, where DocNADE outperforms LDA.

![Diagram](a) PPL: 20NS  (b) NLL of 14 Words

Figure 2: (a) PPL (T200) by iDocNADE and DocNADE for each of the 50 held-out documents of 20NS. The filled circle points to the document for which PPL differs by maximum. (b) NLL of each of the words in the document marked by the filled circle in (a), due to iDocNADE and DocNADE.

Table 2 illustrates the generalization performance of deep variants, where the proposed extensions outperform the baseline DocNADE on all datasets except baseline DocNADE with full-softmax (or tree-softmax). In total, we show a gain of 5.2% (404 vs 426) in PPL score on an average over the 15 datasets.

Table 1 shows the average held-out perplexity (PPL) per word as, $PPL = \exp \left( -\frac{1}{N} \sum_{t=1}^{N} \log p(v^t) \right)$ where $N$ and $|v^t|$ are the total number of documents and words in a document $v^t$. To compute PPL, the log-likelihood of the document $v^t$, i.e., $\log p(v^t)$, is obtained by $\mathcal{L}^{DocNADE}$ (eqn. 2) in the DocNADE (forward only) variants, while we average PPL scores from the forward and backward networks of the iDocNADE variants.

Table 1 shows that the proposed models achieve lower perplexity for both the short-text (413 vs 435) and long-text (393 vs 416) datasets than baseline DocNADE with full-softmax (or tree-softmax). In total, we show a gain of 5.2% (404 vs 426) in PPL score on an average over the 15 datasets.

**Inspection:** We quantify the use of context information in learned informed document representations. For 20NS dataset, we randomly select 50 held-out documents from its test set and compare (Figure 2a) the PPL for each of the held-out documents under the learned 200-dimensional DocNADE and iDocNADE. Observe that iDocNADE achieves lower PPL for the majority of the documents. The filled circle(s) points to the document for which DocNADE and DeepDNE and proposed variants (iDocNADE, DocNADEe, iDocNADEe, iDeepDNE, DeepDNEe and iDeepDNEe) using 50 (in supplementary) and 200 (T200) topics, set by the hidden layer size $H$.

**Quantitative:** Table 1 shows the average held-out perplexity (PPL) per word as, $PPL = \exp \left( -\frac{1}{N} \sum_{t=1}^{N} \log p(v^t) \right)$ where $N$ and $|v^t|$ are the total number of documents and words in a document $v^t$. To compute PPL, the log-likelihood of the document $v^t$, i.e., $\log p(v^t)$, is obtained by $\mathcal{L}^{DocNADE}$ (eqn. 2) in the DocNADE (forward only) variants, while we average PPL scores from the forward and backward networks of the iDocNADE variants.

Table 1 shows that the proposed models achieve lower perplexity for both the short-text (413 vs 435) and long-text (393 vs 416) datasets than baseline DocNADE with full-softmax (or tree-softmax). In total, we show a gain of 5.2% (404 vs 426) in PPL score on an average over the 15 datasets.

**Inspection:** We quantify the use of context information in learned informed document representations. For 20NS dataset, we randomly select 50 held-out documents from its test set and compare (Figure 2a) the PPL for each of the held-out documents under the learned 200-dimensional DocNADE and iDocNADE. Observe that iDocNADE achieves lower PPL for the majority of the documents. The filled circle(s) points to the document for which DocNADE and DeepDNE and proposed variants (iDocNADE, DocNADEe, iDocNADEe, iDeepDNE, DeepDNEe and iDeepDNEe) using 50 (in supplementary) and 200 (T200) topics, set by the hidden layer size $H$.

**Quantitative:** Table 1 shows the average held-out perplexity (PPL) per word as, $PPL = \exp \left( -\frac{1}{N} \sum_{t=1}^{N} \log p(v^t) \right)$ where $N$ and $|v^t|$ are the total number of documents and words in a document $v^t$. To compute PPL, the log-likelihood of the document $v^t$, i.e., $\log p(v^t)$, is obtained by $\mathcal{L}^{DocNADE}$ (eqn. 2) in the DocNADE (forward only) variants, while we average PPL scores from the forward and backward networks of the iDocNADE variants.

Table 1 shows that the proposed models achieve lower perplexity for both the short-text (413 vs 435) and long-text (393 vs 416) datasets than baseline DocNADE with full-softmax (or tree-softmax). In total, we show a gain of 5.2% (404 vs 426) in PPL score on an average over the 15 datasets.
**PPL** differs by a maximum between iDocNADE and DocNADE. We select the corresponding document and compute the negative log-likelihood (NLL) for every word. Figure 2b shows that the NLL for the majority of the words is lower (better) in iDocNADE than DocNADE. See the supplementary material for the raw text of the selected documents.

**Interpretability (Topic Coherence)** Beyond PPL, we compute topic coherence (Chang et al. 2009; Newman, Karimi, and Cavedon 2009; Das, Zaheer, and Dyer 2015; Gupta et al. 2018b) to assess the meaningfulness of the underlying topics captured. We choose the coherence measure proposed by Röder, Both, and Hinneburg (2015) that identifies context features for each topic word using a sliding window over the reference corpus. The higher scores imply more coherent topics.

**Quantitative:** We use gensim module (coherence type = c_v) to estimate coherence for each of the 200 topics (top 10 and 20 words). Table 3 shows average coherence over 200 topics using short-text and long-text datasets, where the high scores for long-text in iDocNADE (.636 vs .602) suggest that the contextual information helps in generating more coherent topics than DocNADE. On top, the introduction of embeddings, i.e., iDocNADEe for short-text boosts (.847 vs .839) topic coherence. **Qualitative:** Table 5 illustrates example topics each with a coherence score.

**Applicability (Document Retrieval)** To evaluate the quality of the learned representations, we perform a document retrieval task using the 15 datasets and their label information. We use the experimental setup similar to Lauly et al. (2017), where all test documents are treated as queries to retrieve a fraction of the closest documents in the original training set using cosine similarity measure between their representations (eqn. 12 in iDocNADE and $\overrightarrow{\text{D}}$ in DocNADE). To compute retrieval precision for each fraction (e.g., 0.0001, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, etc.), we average the number of retrieved training documents with the same label as the query. For multi-label datasets, we average the precision scores over multiple labels for each query. Since Salakhutdinov and Hinton (2009) and Lauly et al. (2017) showed that RSM and DocNADE strictly outperform LDA on this task, we only compare DocNADE and its proposed extensions.

Table 1 shows the IR-precision scores at retrieval fraction 0.02. Observe that the introduction of both pre-trained embedding priors and contextual information leads to im-

<table>
<thead>
<tr>
<th>data</th>
<th>DeepDNE PPL</th>
<th>iDeepDNE PPL</th>
<th>DeepDNE IR</th>
<th>iDeepDNE IR</th>
</tr>
</thead>
<tbody>
<tr>
<td>20NSshort</td>
<td>917 .21</td>
<td>841 .22</td>
<td>827 .25</td>
<td>830 .26</td>
</tr>
<tr>
<td>TREC6</td>
<td>114 .50</td>
<td>69 .52</td>
<td>69 .55</td>
<td>68 .55</td>
</tr>
<tr>
<td>R21578title</td>
<td>253 .50</td>
<td>231 .52</td>
<td>236 .63</td>
<td>230 .61</td>
</tr>
<tr>
<td>Subjectivity</td>
<td>428 .77</td>
<td>393 .77</td>
<td>392 .81</td>
<td>395 .82</td>
</tr>
<tr>
<td>Polarity</td>
<td>408 .51</td>
<td>385 .51</td>
<td>383 .55</td>
<td>387 .53</td>
</tr>
<tr>
<td>TMN</td>
<td>681 .60</td>
<td>624 .62</td>
<td>627 .63</td>
<td>623 .66</td>
</tr>
<tr>
<td>Avg (short)</td>
<td>467 .51</td>
<td>424 .53</td>
<td>422 .57</td>
<td>421 .57</td>
</tr>
<tr>
<td>Reuters8</td>
<td>216 .85</td>
<td>192 .89</td>
<td>191 .88</td>
<td>191 .90</td>
</tr>
<tr>
<td>20NS</td>
<td>551 .25</td>
<td>504 .28</td>
<td>504 .29</td>
<td>506 .29</td>
</tr>
<tr>
<td>R21578</td>
<td>318 .71</td>
<td>299 .73</td>
<td>297 .72</td>
<td>298 .73</td>
</tr>
<tr>
<td>AGNews</td>
<td>572 .75</td>
<td>441 .77</td>
<td>441 .75</td>
<td>440 .80</td>
</tr>
<tr>
<td>RCV1V2</td>
<td>489 .86</td>
<td>464 .88</td>
<td>466 .89</td>
<td>466 .89</td>
</tr>
<tr>
<td>Avg (long)</td>
<td>429 .68</td>
<td>380 .71</td>
<td>379 .71</td>
<td>379 .72</td>
</tr>
<tr>
<td>Avg (all)</td>
<td>450 .39</td>
<td>404 .61</td>
<td>403 .63</td>
<td>402 .64</td>
</tr>
</tbody>
</table>

Table 2: Deep Variants (+ Full-softmax) with T200: PPL and IR (i.e., IR-precision) for short and long text datasets.
To perform document categorization, we employ a logistic regression classifier with regularization. We also construct distributional priors via pre-trained word embeddings. We show that leveraging contextual information and interpreting words and their applicability in text retrieval and classification.

Table 3: Topic coherence with the top 10 (W10) and 20 (W20) words from topic models (T200). Since, (Gupta et al. 2018a) have shown that DocNADE outperforms both glove-DMM and glove-LDA, therefore DocNADE as the baseline.

Table 4: Text classification for short and long texts with T200 or word embedding dimension (Topic models with FS) proved performance on the IR task for short-text and long-text datasets. We report a gain of 11.1% (.60 vs .54) in precision on an average over the 15 datasets, compared to DocNADE. On top, the deep variant i.e. iDeepDNE (Table 2) demonstrates a gain of 8.5% (.64 vs .59) in precision over the 11 datasets, compared to DeepDNE. Figures (3a, 3b, 3c) and (3d, 3e and 3f) illustrate the average precision for the retrieval task on short-text and long-text datasets, respectively.

Applicability (Text Categorization) Beyond the document retrieval, we perform text categorization to measure the quality of word vectors learned in the topic models. We consider the same experimental setup as in the document retrieval task and extract the document representation (latent vector) of 200 dimension for each document (or text), learned during the training of DocNADE variants. To perform document categorization, we employ a logistic regression classifier with $L_2$ regularization. We also compute document representations from pre-trained glove (Pennington, Socher, and Manning 2014) embedding matrix by summing the word vectors and compute classification performance. On top, we also extract document representation from doc2vec (Le and Mikolov 2014).

Table 4 shows that glove leads DocNADE in classification performance, suggesting a need for distributional priors. For short-text dataset, iDocNADE (and DocNADE) outperforms glove (.700 vs .685) and DocNADE (.700 vs .661) in F1. Overall, we report a gain of 5.2% (.664 vs .631) in F1 due to iDocNADEe over DocNADE for classification on an average over 13 datasets.

Inspection of Learned Representations: To analyze the meaningfulness captured, we perform a qualitative inspection of the learned representations by the topic models. Table 5 shows topics for 20NS dataset that could be interpreted as religion, which are (sub)categories in the data, confirming that meaningful topics are captured. Observe that DocNADEe extracts a more coherent topic.

For word level inspection, we extract word representations using the columns $W_{vi}$ as the vector (200 dimension) representation of each word $v_i$, learned by iDocNADE using 20NS dataset. Table 6 shows the five nearest neighbors of some selected words in this space and their corresponding similarity scores. We also compare similarity in word vectors from iDocNADE and glove embeddings, confirming that meaningful word representations are learned.

Conclusion

We show that leveraging contextual information and introducing distributional priors via pre-trained word embeddings in our proposed topic models result in learning better word/document representation for short and long documents, and improve generalization, interpretability of topics and their applicability in text retrieval and classification.
References


Zheng, Y.; Zhang, Y.-J.; and Larochelle, H. 2016. A deep and autoregressive approach for topic modeling of multimodal data. In IEEE transactions on pattern analysis and machine intelligence, 1056–1069. IEEE.