Learning Topic Models by Neighborhood Aggregation

Ryohei Hisano

Social ICT Research Center
Graduate School of Information Science and Technology
The University of Tokyo
em072010@yahoo.co.jp

Abstract

Topic models are frequently used in machine learning owing to their high interpretability and modular structure. However, extending a topic model to include a supervisory signal, to incorporate pre-trained word embedding vectors and to include a nonlinear output function is not an easy task because one has to resort to a highly intricate approximate inference procedure. The present paper shows that topic modeling with pre-trained word embedding vectors can be viewed as implementing a neighborhood aggregation algorithm where messages are passed through a network defined over words. From the network view of topic models, nodes correspond to words in a document and edges correspond to either a relationship describing co-occurring words in a document or a relationship describing the same word in the corpus. The network view allows us to extend the model to include supervisory signals, incorporate pre-trained word embedding vectors and include a nonlinear output function in a simple manner. In experiments, we show that our approach outperforms the state-of-the-art supervised Latent Dirichlet Allocation implementation in terms of both held-out document classification tasks and topic coherence.

1 Introduction

Topic models are widely used in both academia and industry owing to their high interpretability and modular structure [2, 11]. The highly interpretable nature of topic models allows one to gain important insights from a large collection of documents while the modular structure allows researchers to bias topic models to reflect additional information, such as supervisory signals [22], covariate information [33], time-series information [1, 31], document–document relational information [6], pre-trained word embedding vectors [27, 9, 20] and knowledge graph embeddings [39].

However, inference in a highly structured graphical model is not an easy task. This hinders practitioners from extending the model to incorporate various information besides text of their own choice. Furthermore, adding a nonlinear output function makes the model even more difficult to train.

The present paper shows that topic modeling with pre-trained word embedding vectors can be viewed as implementing a neighborhood aggregation algorithm [8, 15, 4] where the messages are passed through a network defined over words. From the network view of topic models, Latent Dirichlet Allocation (LDA) [2] can be thought of as creating a network where nodes correspond to words in a document and edges corresponds to either a relationship describing co-occurring words in a document or a relationship describing the same word in the corpus. The network view makes it clear how a topic label configuration of a word in a document is affected by neighboring words defined over the network and adding supervisory signals simply amounts to adding new edges to the network. Furthermore, by replacing the message passing operation with a differentiable neural network, as is done for neighborhood aggregation algorithms [15, 14, 8], we can learn the influence of the pre-trained word embedding vectors to the topic label configurations as well as the effect from the same label relationship, from the supervisory signal.

Our contribution is summarized as follows.

- We show that topic modeling with pre-trained word embedding vectors can be viewed as implementing a neighborhood aggregation algorithm where the messages are passed through a network defined over words.
By exploiting the network view of topic models, we propose a supervised topic model that has adaptive message passing where the parameters governing the message passing and the parameters governing the influence of the pre-trained word embedding vectors to the topic label configuration are learned from the supervisory signal.

Our model includes a nonlinear output function connecting document to their corresponding supervisory signal.

Our model incorporates pretrained word embedding vectors in a natural fashion.

Our approach outperforms state-of-the-art supervised LDA implementation [17, 42] on a wide variety of datasets in terms of both predictive performance and topic coherence.

We also provide a qualitative comparison on the learned topics highlighting the difference between topics learned by the state-of-the-art supervised LDA and our proposed models.

The remainder of the paper is organized as follows. Section 2 briefly summarizes the basic notations used throughout the paper. Section 3 reviews the literature directly relevant to our approach. We first show how LDA can be viewed as implementing a neighborhood aggregation algorithm in a network defined over words in the corpus. We then show how to add supervisory signals and pre-trained word embedding vectors to the model with a nonlinear output function. Section 4 describes our training procedure. Section 5 compares the performance of our approach with that of the state-of-the-art supervised LDA implementation [17, 42]. We compare both the prediction performance and topic coherence. We also provide a qualitative comparison on the learned topics. Section 6 presents conclusions drawn from the results of the study.

2 Notations

We briefly summarize the basic notations used throughout the paper. Let $1 \leq d \leq D$, $1 \leq w \leq W$, $1 \leq k \leq K$ and $1 \leq s \leq S$ respectively denote the document index, word index, topic number index and label index. We denote by $x_{d,w}$, the number of word counts for a particular document–word pair $(d, w)$. The task of topic modeling is to assign the average topic label configuration $z = z_{d,w}^k$ from the observed document word count matrix $X = \{x_{d,w}\}$ where the average topic label configuration is defined as $z_{d,w}^k := \sum_{i=1}^{N_d} \frac{x_{d,w}^i}{\sum_{k=1}^{K} z_{d,w}^k}$ for all document–word pairs $(d, w)$ in the corpus. The average topic label configuration sums to 1 over topic index $1 \leq k \leq K$ (i.e., $\sum_{k=1}^{K} z_{d,w}^k = 1$) and one of the tasks in this paper is to calculate the messages (which we denote $\mu_{d,w}^k(t)$ that can approximate all $z_{d,w}^k$ in the corpus. We denote by $\mu_{d,w}^k(t)$ the estimated message at iteration $t$. Omission of the subscript $k$ (i.e., $\mu_{d,w}(t)$) simply implies the vectorized form of $\mu_{d,w}(t)$s (i.e., $[\mu_{d,w}^1(t) \cdots \mu_{d,w}^K(t)]$).

We use $\theta_d$ to denote the topic proportion distribution of document $d$ and $\phi_w$ to denote the topic distribution. We also use $v_i$ to denote node attribute information attached to node $i$.

3 Background

3.1 Factor Graph Approach to LDA

From the perspective of topic modeling, our approach is related to the work of [40, 41], who reframed LDA as a factor graph and used the loopy belief propagation algorithm [25] for inference and parameter estimation.
The classical Bayesian network plate diagram representation of LDA is presented in Figure 1. The joint distribution of the model can be summarized as

\[
p(x, z | \alpha, \beta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \frac{\Gamma(\sum_{w=1}^{W} x_{d,w}^{k} + \alpha)}{\Gamma(\sum_{k=1}^{K} (\sum_{w=1}^{W} x_{d,w}^{k} + \alpha))} \times \prod_{w=1}^{W} \prod_{k=1}^{K} \frac{\Gamma(\sum_{d=1}^{D} x_{d,w}^{k} z_{d,w}^{k} + \beta)}{\Gamma(\sum_{d=1}^{D} (\sum_{w=1}^{W} x_{d,w}^{k} z_{d,w}^{k} + \beta))},\]

where \( \alpha \) and \( \beta \) denote hyperparameters of the Dirichlet prior distribution. Meanwhile, by designing the factor function as

\[
f_{\theta_d}(x, z, \alpha) = \prod_{k=1}^{K} \frac{\Gamma(\sum_{w=1}^{W} x_{d,w}^{k} + \alpha)}{\Gamma(\sum_{k=1}^{K} (\sum_{w=1}^{W} x_{d,w}^{k} + \alpha))}
\]

and

\[
f_{\phi_w}(x, z, \beta) = \prod_{k=1}^{K} \frac{\Gamma(\sum_{d=1}^{D} x_{d,w}^{k} z_{d,w}^{k} + \beta)}{\Gamma(\sum_{d=1}^{D} (\sum_{w=1}^{W} x_{d,w}^{k} z_{d,w}^{k} + \beta))},
\]

a factor graph representation of LDA (i.e., Figure 2) can be summarized as

\[
p(x, z | \alpha, \beta) = \prod_{d=1}^{D} f_{\theta_d}(x, z, \alpha) \prod_{w=1}^{W} f_{\phi_w}(x, z, \beta).
\]

Equation (4) shows that the factor graph representation encodes exactly the same information as Eq. (1). From the factor graph representation, it is possible to reinterpret LDA using a Markov random field framework and thus infer messages for words in a document using loopy belief propagation. The essence of their paper can be summarized by a message updating equation of the form

\[
\mu_{d,w}^{k}(t + 1) \propto \frac{\sum_{w'=1,w'\neq w}^{W} x_{d,w'}^{k} \mu_{d,w'}^{k}(t) + \alpha}{\sum_{k=1}^{K} (\sum_{w'=1,w'\neq w}^{W} x_{d,w'}^{k} \mu_{d,w'}^{k}(t) + \alpha)} \times \frac{\sum_{d'=1,d'\neq d}^{D} x_{d',w}^{k} x_{d',w}^{k} \mu_{d',w}^{k}(t) + \beta}{\sum_{w=1}^{W} (\sum_{d'=1,d'\neq d}^{D} x_{d',w}^{k} \mu_{d',w}^{k}(t) + \beta)}.
\]
After updating $\mu_{d,w}^k$ until convergence, the topic proportion distribution $\theta_d$ and topic distribution $\phi_w$ can be approximated by

$$
\theta_d^k = \frac{\sum_{w=1}^{W} x_{d,w} \mu_{d,w}^k + \alpha}{\sum_{k=1}^{K} (\sum_{w=1}^{W} x_{d,w} \mu_{d,w}^k + \alpha)}
$$

and

$$
\phi_w^k = \frac{\sum_{d=1}^{D} x_{d,w} \mu_{d,w}^k + \beta}{\sum_{w=1}^{W} (\sum_{d=1}^{D} x_{d,w} \mu_{d,w}^k + \beta)}
$$

respectively. We refer to [40] for further information concerning the belief propagation approach.

Our goal in this paper is to connect the factor graph approach of LDA with the neighborhood aggregation algorithm to make the main contributions summarized in the introduction.

### 3.2 Neighborhood Aggregation Algorithm

The goal of node embedding is to represent nodes as low-dimensional vectors summarizing the positions or structural roles of the nodes in a network [15][34][28]. The neighborhood aggregation algorithm is a recently proposed algorithm used in node embedding [8][15][4][14], which tries to overcome limitations of the more traditional direct encoding [32][12] and neighborhood autoencoder [5][38] approaches. The heart of a neighborhood aggregation algorithm lies in designing encoders that summarize information gathered by a node’s local neighborhood. In a neighborhood aggregation algorithm, it is easy to incorporate a graph structure into the encoder, leverage node attribute information and add nonlinearity to the model. The parameters defining the encoder can be learned by minimizing the supervised loss [8][15]. In [8], it was shown that these algorithms can be seen as replacing message passing operations with differential neural networks. We refer to [8] for a more detailed explanation.

The essence of neighborhood aggregation algorithms is characterized by three steps: aggregation, combination and normalization. In the aggregation step, we gather information from a node’s local neighborhood. A concatenation, sum-based procedure or elementwise mean is usually employed. In the combination step, we combine a node’s attribute information with the gathered information from the aggregation step. After passing the combined information through a nonlinear transformation, we normalize the message so that the new updated messages can be further used by neighboring nodes.

The overall process is succinctly summarized as

$$
\mu_i(t+1) = \text{Norm}(\sigma(\text{Comb}(v_i, \text{Agg}(\mu_j(t)); \forall j \in Nei(i)); W_A, W_C)),
$$

where $\mu_i(t+1)$ denotes the message of node $i$ at iteration $t+1$; $\text{Nei}(i)$ denotes neighboring nodes of node $i$; $\text{Agg}, \text{Comb}$ and $\text{Norm}$ respectively denote aggregation, combination and normalization functions; $W_A$ and $W_C$ denote parameters used in the combination step (e.g., $\text{Comb}(v_i, x_i) = W_C v_i + W_A x_i$) and $\sigma$ is an elementwise nonlinear transformation. With enough update iterations, the model converges and the desired messages are learned.

Suppose that we are given a training dataset $D = \{X, y_{1:D}\}$, where $X$ is the observed network and $y_{1:D}$ is the supervisory signal attached to each node. For a classification problem $\{y_d \in 1, \cdots, S\}$, we use a simple neural network with softmax output to transform the learned messages to probabilities and to minimize the cross-entropy loss. This is summarized as

$$
\begin{align*}
\text{score}_d &= S_C \sigma(S_B \sigma(S_A \mu_d + T_A) + T_B) + T_C \\
\exp(\text{score}_d^s) &= \sum_{s=1}^{S} \exp(\text{score}_d^s) \\
p_d^s &= \frac{\exp(\text{score}_d^s)}{\sum_{s=1}^{S} \exp(\text{score}_d^s)} \\
\min_{\{W_A, W_C, S_A, S_B\}} &= -\sum_{d=1}^{D} \sum_{s=1}^{S} y_d^s \log(p_d^s)
\end{align*}
$$

where $S_A$, $S_B$, $S_C$ are either $H_1 \times K$, $H_2 \times H_1$ and $S \times H_2$ weight matrix transforming the messages and $T_A$, $T_B$ and $T_C$ are bias vector of either size $H_1$, $H_2$ and $S$, $y_d^s = 1$ if the label of node $d$ is $s$ and zero otherwise and $\sigma$ is an elementwise nonlinear transformation. We use the classic sigmoid function in this paper.
For a regression problem \( y_n \in R \), parameters can be learned by minimizing the sum-of-squared loss:

\[
\min_{\{W_A, W_C, S_A, S_B\}} \sum_{d=1}^{D}(y_d - \text{score}_d)^2.
\] (10)

---

**4 Model**

**4.1 LDA and Neighborhood Aggregation**

The message-passing equation (Eq. (5)) of LDA can be seen as taking an elementwise product of two neighborhood aggregation operations and normalizing the result for probabilistic interpretation. We clarify this point with an illustrative example. Figure 3 shows a hypothetical corpus with three documents “platypus, clinic, ill”, “platypus, Australia, east” and “platypus, eat, shrimp”. Each document–word pair in the corpus has an associated message like the upper-right vector depicted next to the word “ill” in document 1. The red bold lines indicate a relationship describing co-occurring words in a document while the blue dotted lines indicate a relationship describing the same word in the corpus.

Suppose that we want to update the message for the word “platypus” found in document 2. According to the message passing equation (Eq. (5)), this message is updated by the elementwise product of messages gathered from edges describing co-occurring words in document 2 (as denoted by the red bold arrows) and messages gathered from edges describing the same word in the corpus (as denoted by the blue dotted arrows). In fact, assuming that the aggregation function in a document is

\[
Agg_d^k(\mu_j(t); \forall j \in Nei_d(d, w)) = \sum_{w'=1, w' \neq w}^W x_{d,w'} \mu_d^{k} + \alpha,
\] (11)

where \( Nei_d(d, w) \) denotes all other words in the same document, there are no additional node features to combine and, using \( Agg_d \) to denote \( [Agg_d^1 \cdots Agg_d^K] \), the neighborhood aggregation operation from co-occurring words in a document can be written as

\[
NA_d(t) = \text{Norm}_K(Agg_d(\mu_j(t); \forall j \in Nei_d(d, w))),
\] (12)

where \( \text{Norm}_K \) is defined to be a normalization function dividing by the sum over topics \( 1 \leq k \leq K \).

Following similar reasoning, it is assumed that the aggregation function for the edges describing the same word in the corpus is

\[
Agg_w^k(\mu_j(t); \forall j \in Nei_w(d, w)) = \sum_{d'=1, d' \neq d}^D x_{d',w} \mu_{d',w}^{k} + \beta,
\] (13)
where \( \text{Nei}_w(d, w) \) denotes all document–word pairs in the corpus that is the same word as \( w \) (i.e. nodes in the blue dashed square shown in Figure 3) and there are no additional node features to combine (such as pre-trained word embedding vectors described below). The neighborhood aggregation operation for this neighborhood system can be summarized as

\[
\text{NA}_w(t) = \text{Norm}_W(\text{Agg}_w(\mu_j(t)); \forall j \in \text{Nei}_w(d, w)),
\]

where \( \text{Norm}_W \) is defined as a normalization function dividing by the sum over words \( 1 \leq w \leq W \).

The message update equation for a document–word pair \((d, w)\) can thus be summarized by

\[
\mu_{d,w}(t + 1) = \text{Norm}_K(\text{NA}_d(t) \odot \text{NA}_w(t)),
\]

where \( \odot \) denotes elementwise multiplication. The messages are normalized in the final step so that they can be used as a proper probability distribution.

This explanation shows that the message passing equation (Eq. (5)) of LDA can be seen as an element-wise product of two neighborhood aggregation operations with an additional normalization step for probabilistic interpretation.

### 4.2 Supervised LDA and Neighborhood Aggregation

We extend the above formulation to incorporate supervisory signals. Supervisory signals such as label information can be thought of as defining an additional neighborhood system in the above formulation. We therefore add edges reflecting this additional neighborhood system. For example, in a classification problem where we have label information for each document, we add edges among document–word pairs that belong to other documents with the same label. Figure 4 illustrates this additional neighborhood system. For example, in a classification problem where we have label information for each document, we add edges among document–word pairs that belong to other documents with the same label. Figure 4 illustrates the same hypothetical corpus as shown in Figure 3. The green dashed arrows are the new edges added by the supervisory signal.

We define the aggregation function for this neighborhood system as

\[
\text{Agg}_s^k(\mu_j(t); \forall j \in \text{Nei}_s(d, w)) = \sum_{d'=1, d' \neq d}^D \sum_{w=1}^W x_{d', w}^k \mu_{d', w}^k \eta_{1(l(d') = l(d))},
\]

where the indicator function \( 1_{l(d') = l(d)} \) is used to select documents with the same label and \( l \) is a function that maps a document index \( d \) to its corresponding label \( s \). Neighborhood systems can easily be extended to the regression case where documents do not necessarily have exactly the same output value (e.g., real numbers). In this case, we replace the indicator function as \( 1_{|l(d') - l(d)| < \epsilon} \) for a given \( \epsilon > 0 \).

From this formulation, the neighborhood aggregation operation from the supervisory signal can be written as

\[
\text{NA}_s(t) = \text{Norm}_K(W_s \text{Agg}_s(\mu_j(t)); \forall \text{Nei}_s(d, w)),
\]

where \( \text{Nei}_s(d, w) \) denotes the neighborhood system of \( s \) for document–word pair \((d, w)\) and \( W_s \) denotes a diagonal matrix with positive entries only. We train \( W_s \) using a supervisory signal as is done in neighborhood aggregation algorithms [15].

In standard sum-product algorithm, we take the product of all messages from factors to variables. However as was noted in [10], a product operation cannot control the messages from different sources (i.e. supervised part and unsupervised part) and we therefore take the weighted sum of the two neighborhood aggregation operations to rebalance the effects. The whole message updating equation can be summarized as

\[
\mu_{d,w}(t + 1) = \text{Norm}_K(\eta \text{NA}_s(t) + (1 - \eta) \text{NA}_d(t)) \odot \text{NA}_w(t)),
\]

where \( \eta \) controls the strength of the supervisory signal in inferencing the topic label configuration. This type of rebalancing of the effect from the supervised part and unsupervised is also taken in the traditional supervised LDA approach where it is well known that the effect of the supervised part is reduced for documents with more words in the standard formulation [17],[10]. Eq. (18) is quite standard technique, contrary to what it might appear at first glance.
Table 1: Performance of held-out document classification

| Dataset          | Economy watcher survey | Amazon review | Subjectivity |
|------------------|-------------------------|---------------|--------------|
|                  | Performance measure     |               |              |
|                  | Cross entropy | Accuracy | Cross entropy | Accuracy | Cross–entropy | Accuracy |
| SLDA             | 1.577       | 0.325    | 1.943       | 0.446       | 0.661       | 0.650    |
| MedLDA           | -          | 0.417    | -           | 0.484       | -           | 0.824    |
| WE-MLP           | 1.369       | 0.460    | 1.305       | 0.508       | 0.689       | 0.674    |
| LFLDA            | 1.227       | 0.488    | 1.288       | 0.522       | 0.695       | 0.335    |
| LFDMMM           | 1.243       | 0.492    | 1.288       | 0.522       | 0.512       | 0.492    |
| NA-SLDA          | 1.223       | 0.473    | 1.271       | 0.525       | 0.389       | 0.832    |
| NA-WE-SLDA       | 1.207       | 0.502    | 1.243       | 0.535       | 0.353       | 0.848    |

4.3 Incorporating Pretrained Word Embedding Vectors

It is natural to assume that there is an association between pre-trained word embedding vectors \([23, 3]\) and topics, so long as topic models are used to summarize the semantic information in a document. Hence, learning the association between pre-trained word embedding vectors and topics is important especially at the time of testing when there might be many unseen words in the held-out documents. If these unobserved words are included in the pre-trained word embedding dictionary and we have trained a function mapping the word embedding vectors to the topics, we can leverage the pre-trained word embedding vectors to predict the topic label configuration more accurately. This issue was addressed in several recent papers \([9, 20, 13]\).

Within the neighborhood aggregation framework, the pre-trained word embedding vectors can be modeled as a node attribute vector \(v_i\) in the notation of Eq. (8). We first define the word embedding in the topic distribution transformation as

$$u_g(d, w) = \text{Norm}_K(\sigma(W_C v_g(d, w))),$$

where \(g\) is a function that maps the document–word index to the word index in the pre-trained word embedding vector dictionary, \(W_C\) is the \(K \times E\) weight matrix transforming word embedding vectors (which we assume to have dimension \(E\)) to the topic and \(\sigma\) is an element–wise nonlinear transformation.

Using the transformed vector \(u_g(d, w)\), we define the combine function as

$$\text{Comb}(u_g(d, w), NA_w(t), NA_d(t)) = u_g(d, w) \odot (\eta NA_w(t) + (1 - \eta) NA_d(t)) \odot NA_d(t).$$

The entire message updating equation is now summarized as

$$\mu_{d,w}(t + 1) = \text{Norm}_K(\text{Comb}(u_g(d, w), NA_w(t), NA_d(t), NA_s(t))).$$

This update equation is a particular instance of Eq. (8) assuming identity mapping for \(\sigma\) and applying elementwise multiplication for the combine function.

4.4 Training

Inferencing of the messages (i.e., \(\mu_{d,w}^k\)) and estimation of the parameters (i.e., \(W_S, W_C, S_A, S_B, S_C, T_A, T_B\) and \(T_C\)) can be performed by alternately inferring the topic label configurations given parameters of the model and updating the parameters minimizing the supervised loss given a topic label configuration.

In each iteration, we first infer the topic label configurations for randomly sampled document–word pairs, then update the parameters \(W_S\) and \(W_C\) by creating a mini–batch of document–word pairs and by employing a stochastic gradient descent \([18]\) and finally update the parameters related to the output function (e.g. \(S_A, S_B, S_C, T_A, T_B\) and \(T_C\)) using all the documents. For the regularization parameter governing \(W_S\) and \(W_C\), we set it to 0.001 and for the output function we set dropout probability of 0.5 for regularization. We repeat this iteration until convergence.
Table 2: Topic coherence measurement using the $C_V$ measure proposed in [35]

| Model          | Economy watcher survey | Amazon review | Subjectivity |
|----------------|------------------------|---------------|--------------|
| LDA            | 0.335                  | 0.301         | 0.342        |
| SLDA           | 0.336                  | 0.321         | 0.339        |
| MedLDA [12]    | 0.312                  | 0.319         | 0.337        |
| LFLDA [27]     | 0.321                  | 0.342         | 0.348        |
| LFDM [27]      | 0.321                  |               | 0.331        |
| NA-SLDA        | 0.333                  | 0.340         | 0.349        |
| NA-WE-SLDA     | 0.338                  | 0.345         | 0.342        |

5 Experiments

5.1 Datasets

We conduct experiments showing the validity of our approach. We use three datasets in our experiments. Two of our datasets are data of multiclass-label prediction tasks focusing on sentiment (economic assessments and product reviews) while the other dataset is data of a binary label prediction task focusing on subjectivity. We summarize the datasets below. The exact formatted version of each dataset and the replication code are available on the author’s website.

- The economic watcher survey is a dataset provided by the Cabinet Office of Japan [29]. The purpose of the survey is to gather economic assessments from working people in Japan to comprehend region-by-region economic trends in near real time. The survey is conducted every month and each record in the dataset consists of a multi-class label spanning from 1 (i.e., the economy will get worse) to 5 (i.e., the economy will get better) and text describing the reasons behind a judgement. We use a subset of this dataset describing the assessment of future economic conditions. We use the 3000 most frequently used words in the corpus and further restrict our attention to records that have more than a total of 20 nouns, verbs and adjectives in the pre-trained word embedding dictionary to perform a fair comparison among the models [2]. We randomly sample 1000 records for training and 1000 records each for development and testing. For the pre-trained word embedding vectors describing Japanese words, we use word2vec vectors provided by [37].

- Amazon review data are a dataset of gathered ratings and review information [21, 16]. We use a subset of the five-core video game dataset [4] and use the 3000 most frequently used words in the corpus. We sample 1000 records for training and 1000 each for development and testing focusing on reviews that have more than 20 words excluding stop words in the pre-trained word embedding dictionary [5]. For the pre-trained word embedding vectors describing English words, we use the word2vec embedding vectors provided by Google [23].

- Subjectivity data are a dataset provided by [30], who gathered subjective (label 0) and objective (label 1) snippets from Rotten Tomatoes and the Internet Movie Database. We focus on snippets that have more than nine words and sample 1000 snippets each for training, development and testing [6]. Other settings are the same as those of the Amazon review data.

5.2 Classification Performance

The main goal in this section is to compare the classification performance of our proposed models with that of the state-of-the-art supervised LDA implementations (which we denote as SLDA and MedLDA), nonlinear prediction using pre-trained word embedding vectors (which we denote as WE-MLP) and topic models incorporating pre-trained word embedding vectors.
We compare the performance of these models with that of our proposed model without pre-trained word embedding vectors (i.e., Eq. \((18)\), which we denote as NA-SLDA) and our full model incorporating word embedding vectors (i.e., Eq. \((21)\), which we denote as NA-WE-SLDA).

For the state-of-the-art SLDA implementation, we use the code provided by [17] owing to its speed and accuracy. We also compare our models against the maximum margin supervised topic models (i.e., MedLDA\(^{[42]}\)) which is reported to have better performance than traditional SLDA. For topic models incorporating pre-trained word embedding vectors we use the latent factor LDA (i.e., LFLDA) and latent factor Dirichlet multinomial mixture (i.e., LFDMM) models of [27] due to its high performance in both coherence and classification tasks. We use the default settings defined in the codes provided by the authors. We also fix the number of topics to 20 for all experiments performed in this section.

For our proposed model, we simply set \(\eta\) in our model to 0.4. We set the number of hidden units in Eq. \((9)\) to be \(H_1 = 50\) and \(H_2 = 50\) and performed training until convergence.

For WE-MLP, we use the average of all the pre-trained word embedding vectors found in a document as the feature describing a document and we connect this feature to our supervisory signal using a simple multilayer perceptron with softmax output. For LFLDA and LFDMM we use the document topic distribution as input and also connect this feature to our supervisory signal using a simple multilayer perceptron with softmax output. Parameters (e.g., the number of hidden units) of these models was found by utilizing the development dataset.

Table 1 summarizes the results. Cross-entropy simply denotes the cross-entropy loss and accuracy denotes the proportion of correct predictions in the held-out documents. For MedLDA we only report accuracy because the method does not use softmax output function. We see that NA-WE-SLDA is the best performing model in all cases and sometimes beats SLDA and MedLDA substantially, especially for multi-label classification tasks. Because NA-SLDA also beats SLDA, we safely conclude that there is an added benefit from our nonlinear formulation relative to SLDA, for which the output function is linear.

5.3 Topic Coherence

Our goal in this section is to compare topic coherence among the supervised topic models that we evaluated in the previous section and the plain LDA model [2]. Adding complex structures to topic models might have a side effect of downgrading interpretability [7]. We want to examine whether our models sacrifice interpretability to achieve better predictive performance.

Traditionally, topic coherence is evaluated on the basis of the perplexity measure, which is basically a measure of how likely a model predicts words in the held-out documents. However, as was shown in [7], higher perplexity does not necessarily correlate to better human interpretability and various topic coherence measures have been proposed to address this issue [26, 24, 36, 35].

The present paper uses the \(C_V\) measure proposed in [35], which is a coherence measure combining existing basic coherence measures, such as pointwise mutual information [26], normalized pointwise mutual information [36] and asymmetrical confirmation measure [24]. It was shown that the \(C_V\) measure is the best coherence measure among 237,912 coherence measures compared in their paper [35]. We employ the source code provided by the authors to calculate the coherence measure in our experiment focusing on the top 15 words in the topic distribution instead of Eq. \((7)\). We refer to the topic representation calculated using Eq. \((7)\) as the “one-hot” form.

Table 2 summarizes the results. We first see that supervised topic models do not downgrade interpretability compared with plain LDA. This is because the corpus analyzed in our experiments is mainly text that has a strong association with the supervisory signal. For instance, topics learned by plain LDA for the Amazon review dataset are “good, like, one, get, best”, which are words strongly associated with the sentiment of a review. The additional information provided by the supervisory signal encourages the topic distribution to be more coherent. Our second observation is that except for the subjectivity dataset, our model with word embedding vectors slightly outperforms the model without word embedding vectors. This is expected because word-embedding vectors are known to reflect semantic information and thus contribute to coherence. Finally, overall models with pre-trained word embedding vectors (i.e., LFLDA, LFDMM and NA-WE-SLDA) outperform models without them (i.e., LDA, SLDA), but our model without word embedding (i.e. NA-SLDA) gives competitive performance.

7The difference between LFLDA and LFDMM is that LFDMM assumes that all the words in a document share the same topic.
8The code is available at https://github.com/dice-group/Palmetto
9For the economy watcher survey, we translated words from Japanese to English.
Table 3: Topics learned by SLDA.

| Topic number | Top 5 words                               |
|--------------|-------------------------------------------|
| Topic 1      | sonic, like, battle, never, also          |
| Topic 2      | good, get, like, one, fun                 |
| Topic 3      | like, good, get, graphics, time           |
| Topic 4      | fun, get, play, new, one                  |
| Topic 5      | football, play, baseball, first, madden    |
| Topic 6      | controller, like, gt, cars, racing         |
| Topic 7      | even, get, cant, like, one                |
| Topic 8      | arcade, great, one, version, best          |
| Topic 9      | rpg, like, card, memory, think            |
| Topic 10     | evil, like, resident, hill, silent         |
| Topic 11     | pokemon, gameboy, also, tony, pro          |
| Topic 12     | characters, really, graphics, find, one    |
| Topic 13     | one, like, time, really, first             |
| Topic 14     | mario, super, nintendo, one, land          |
| Topic 15     | one, great, play, like, best              |
| Topic 16     | ps, get, dreamcast, system, like, xbox     |
| Topic 17     | like, get, buy, gta, many                 |
| Topic 18     | gba, also, one, get, stars                |
| Topic 19     | man, mega, one, levels, level              |
| Topic 20     | final, fantasy, characters, ff, story      |

5.4 Qualitative Comparison

To gain further insights into our learned topics, in Table 3, Table 4 and Table 5 we report the top five words of the 20 topics learned by SLDA, NA-SLDA and NA-WE-SLDA respectively for the Amazon review dataset. Since the five-core video game dataset is about video games we excluded the word “game” in the table to make the difference easier to spot.

We see that for SLDA, there are many proper nouns describing a specific video game (e.g., Pokemon, Resident Evil, Diablo, Final Fantasy, and Mario), while this feature is relatively lost in the topics learned by NA-WE-SLDA which focuses more on words judging the quality of a video game in a more nuanced manner (e.g. “pc, boring, got, bad” in Topic 4 and “good, really, pretty, graphics” in Topic 8). The difference between SLDA and NA-SLDA is less apparent but in SLDA topics are a mixture of proper nouns describing a specific video game, in NA-SLDA we see topics working more like a phrase (e.g. “fun, time, play” in Topic 2 and “ever, best, played” in Topic 17) contributing to the increased coherence measure reported in Table 2. The experiments thus show the validity of our approach.

6 Conclusion

We showed that topic modeling can be viewed as implementing a neighborhood aggregation algorithm where the messages are passed through a network defined over words. By exploiting the network view of topic models, we proposed new ways to model and infer supervised topic models equipped with a nonlinear output function. We also showed that we could incorporate pre-trained word embedding vectors in a natural fashion. Our extensive experiments performed over a range of datasets showed the validity of our approach. We also performed a topic coherence analysis showing that our proposed model does not downgrade topic coherence compared with both LDA and SLDA.

7 Acknowledgements

This work was supported by ACT-I, JST (Japanese Science and Technology Agency). We thank Glenn Pennycook, MSc, from Edanz Group (www.edanzediting.com/ac) for editing a draft of this manuscript.

10 Topics learned by NA-SLDA and NA-WE-SLDA provide similar insights.
Table 4: Topics learned by NA-SLDA.

| Topic number | Top 5 words                      |
|--------------|----------------------------------|
| Topic 1      | fun, get, buy, car, race         |
| Topic 2      | fun, time, play, get, playing    |
| Topic 3      | new, star, one, first, series    |
| Topic 4      | get, like, movie, level, gun     |
| Topic 5      | half, mansion, life, get         |
| Topic 6      | mission, ai, enemy, quake, weapons |
| Topic 7      | levels, man, mega, get, one      |
| Topic 8      | get, bad, like, gta, missions    |
| Topic 9      | diablo, fps, first, playing, months |
| Topic 10     | ps, system, one, buy, like       |
| Topic 11     | one, play, zelda, really, like   |
| Topic 12     | characters, world, different, battle, get |
| Topic 13     | like, graphics, one, great, good |
| Topic 14     | fighting, moves, characters, fighter, arcade |
| Topic 15     | puzzles, max, action, story, payne |
| Topic 16     | final, fantasy, characters, ff, rpg |
| Topic 17     | ever, best, played, great, graphics |
| Topic 18     | mario, super, nintendo, one, version |
| Topic 19     | get, cant, dont, buy, ps         |
| Topic 20     | play, fun, mode, player, also    |

Table 5: Topics learned by NA-WE-SLDA-OneHot.

| Topic number | Top 5 words                      |
|--------------|----------------------------------|
| Topic 1      | new, came, work, made, im        |
| Topic 2      | level, weapons, get, enemies, also |
| Topic 3      | play, time, one, ever, first     |
| Topic 4      | pc, boring, got, bad, graphics   |
| Topic 5      | like, system, controller, one, use |
| Topic 6      | man, star, mega, first, level    |
| Topic 7      | play, mode, also, player, new    |
| Topic 8      | good, really, pretty, graphics, like |
| Topic 9      | missions, get, city, like, fun   |
| Topic 10     | characters, story, final, fantasy, rpg |
| Topic 11     | get, buy, like, dont, fun        |
| Topic 12     | best, graphics, one, great, sonic |
| Topic 13     | max, moves, fighting, even, version |
| Topic 14     | race, racing, cant, time, tracks |
| Topic 15     | ps, play, one, want, get         |
| Topic 16     | enemy, around, mission, like, kill |
| Topic 17     | mario, super, nintendo, one, levels |
| Topic 18     | alien, seen, youll, face, sound  |
| Topic 19     | football, like, fun, madden, time |
| Topic 20     | like, find, way, little, much, well |
References

[1] David M. Blei and John D. Lafferty. Dynamic topic models. In Proceedings of the 23rd International Conference on Machine Learning, ICML ’06, pages 113–120, New York, NY, USA, 2006. ACM.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. J. Mach. Learn. Res., 3:993–1022, 2003.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606, 2016.

[4] Hongyun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques and applications. IEEE Transactions on Knowledge and Data Engineering, forthcoming.

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Deep neural networks for learning graph representations. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI’16, pages 1145–1152. AAAI Press, 2016.

[6] Jonathan Chang and David M. Blei. Relational topic models for document networks. In David A. Van Dyk and Max Welling, editors, AISTATS, JMLR Proceedings, pages 81–88. JMLR.org.

[7] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-graber, and David M. Blei. Reading tea leaves: How humans interpret topic models. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems 22, pages 288–296. Curran Associates, Inc., 2009.

[8] Hanjun Dai, Bo Dai, and Le Song. Discriminative embeddings of latent variable models for structured data. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML’16, pages 2702–2711, 2016.

[9] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In ACL (1), pages 795–804. The Association for Computer Linguistics.

[10] Mandar Dixit, Nikhil Rasiwasia, and Nuno Vasconcelos. Class-specific simplex-latent dirichlet allocation for image classification. In The IEEE International Conference on Computer Vision (ICCV), December 2013.

[11] T. L. Griffiths and M. Steyvers. Finding scientific topics. Proceedings of the National Academy of Sciences, 101(Suppl. 1):5228–5235, 2004.

[12] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, pages 855–864, 2016.

[13] Wayne Xin Zhao Jing Gao Aidong Zhang Guangxu Xun, Yaliang Li. A correlated topic model using word embeddings. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pages 4207–4213, 2017.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 1025–1035. Curran Associates, Inc., 2017.

[15] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, forthcoming.

[16] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In Proceedings of the 25th International Conference on World Wide Web, WWW ’16, pages 507–517, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[17] Angelos Katharopoulos, Despoina Paschalidou, Christos Diou, and Anastasios Delopoulos. Fast supervised lda for discovering micro-events in large-scale video datasets. In Proceedings of the 2016 ACM on Multimedia Conference, MM ’16, pages 332–336, New York, NY, USA, 2016. ACM.
[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations*, abs/1412.6980, 2015.

[19] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. In *In Proc. of EMNLP*, pages 230–237, 2004.

[20] Ximing Li, Jinjin Chi, Changchun Li, Jihong Ouyang, and Bo Fu. Integrating topic modeling with word embeddings by mixtures of vmfs. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 151–160. The COLING 2016 Organizing Committee, 2016.

[21] Julian McAuley, Christopher Targett, Qifeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’15, pages 43–52, New York, NY, USA, 2015. ACM.

[22] Jon D. McAuliffe and David M. Blei. Supervised topic models. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 121–128. Curran Associates, Inc., 2008.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[24] David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP ’11, 2011.

[25] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI’99, pages 467–475, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[26] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT ’10, pages 100–108, 2010.

[27] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3, 2015.

[28] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2014–2023, New York, New York, USA, 20–22 Jun 2016. PMLR.

[29] The Cabinet Office of Japan. utline of the economy watchers survey. pages 332–336, 2017.

[30] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, 2004.

[31] Sungrae Park, Wonsung Lee, and Il-Chul Moon. Supervised dynamic topic models for associative topic extraction with a numerical time series. In *Proceedings of the 2015 Workshop on Topic Models: Post-Processing and Applications*, TM ’15, pages 49–54, 2015.

[32] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’14, pages 701–710, 2014.

[33] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP ’09, pages 248–256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
[34] Leonardo F.R. Ribeiro, Pedro H.P. Saverese, and Daniel R. Figueiredo. Struc2vec: Learning node representations from structural identity. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’17, pages 385–394, New York, NY, USA, 2017. ACM.

[35] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM ’15, pages 399–408, 2015.

[36] Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring topic coherence over many models and many topics. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL ’12, pages 952–961, 2012.

[37] M. Suzuki, K. Matsuda, S. Sekine, N. Okazaki, and K. Inui. Wikipedia kiji ni taisuru kakucho koyuu hyougen label no tajyuu fuyo [multiple granting of extended named expression labels for wikipedia articles]. In The Association for Natural Language Processing. NLP2016, Mar 2016.

[38] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, pages 1225–1234, New York, NY, USA, 2016. ACM.

[39] Liang Yao, Yin Zhang, Baogang Wei, Zhe Jin, Rui Zhang, Yangyang Zhang, and Qinfei Chen. Incorporating knowledge graph embeddings into topic modeling. In AAAI, 2017.

[40] J. Zeng, W. K. Cheung, and J. Liu. Learning topic models by belief propagation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(5):1121–1134, May 2013.

[41] Jia Zeng. A topic modeling toolbox using belief propagation. J. Mach. Learn. Res., 13(1):2233–2236, July 2012.

[42] Jun Zhu, Amr Ahmed, and Eric P. Xing. Medlda: Maximum margin supervised topic models for regression and classification. In Proceedings of the 26th Annual International Conference on Machine Learning, ICML ’09, pages 1257–1264, New York, NY, USA, 2009. ACM.