Unified Neural Topic Model via Contrastive Learning and Term Weighting

Sungwon Han[1] **Mingi Shin**[1] **Sungkyu Park**[2*] **Changwook Jung**[1] **Meeyoung Cha**[3,1*]
[1]School of Computing, KAIST, South Korea
[2]Department of AI Convergence, Kangwon National University, South Korea
[3]Data Science Group, Institute for Basic Science (IBS), South Korea
{lion4151, mingi.shin, wookidoki}@kaist.ac.kr, shaun@kangwon.ac.kr, mcha@ibs.re.kr

Abstract

Two types of topic modeling predominate: generative methods that employ probabilistic latent models and clustering methods that identify semantically coherent groups. This paper newly presents UTopic (Unified neural Topic model via contrastive learning and term weighting) that combines the advantages of these two types. UTopic uses contrastive learning and term weighting to learn knowledge from a pretrained language model and discover influential terms from semantically coherent clusters. Experiments show that the generated topics have a high-quality topic-word distribution in terms of topic coherence, outperforming existing baselines across multiple topic coherence measures. We demonstrate how our model can be used as an add-on to existing topic models and improve their performance.

1 Introduction

One of the most common tasks in natural language processing (NLP) is to find cohesive topics in a text corpus. Topic modeling is widely used in various applications, including trend extraction from real-time streams such as social media (Lau et al., 2012; Park et al., 2021) and identification of notable events upon risk (Shin et al., 2020). Two representative lines of work exist: *generative methods* and *clustering-based methods*.

Generative methods follow the assumption of probabilistic latent semantic analysis (hereafter p-LSA) such that every word token in a document is sampled from a mixture of latent topics (Hoffman, 1999). Such methods estimate both the latent topic distribution per document and word distribution per topic from Bag-of-Words (BoW) representation (Blei et al., 2003; Fisher et al., 2020; Miao et al., 2017; Yan et al., 2013). Latent Dirichlet Allocation (LDA), for example, uses a Bayesian model to estimate the latent topic distribution (Blei et al., 2003). Variational Autoencoder (VAE) retrieves latent topics from the topic distribution while also recovering the original input (Srivastava and Sutton, 2017). Recent techniques have employed pretrained language models on top of VAE-based models to improve topic quality (Bianchi et al., 2021a,b). Generative probabilistic models work on the assumption that each topic is a mix of words from a larger set. However, they share a common weakness: the BoW representation only contains word-level co-occurrences and fails to capture each word token's importance in the document's context information. Consequently, the quality of the estimated topic is reliant on the choice of the word set.

Clustering methods, on the other hand, regard topics as semantic clusters discovered over the document or word embedding space (Angelov, 2020; Grootendorst, 2022; Sia et al., 2020). They utilize the knowledge of the pretrained language model (e.g., BERT (Devlin et al., 2019)) to generate high-level summaries of given documents (or words). Then, document (or word) clusters are identified according to the embedding distance via clustering methods, such as DBSCAN or k-means. They do not require word selection and effectively leverage context information for discovering topics. Nevertheless, their core is clustering; hence, they cannot assign a mixture of topics to each document.

This research takes a step further by combining the benefits of the previous methods by including novel considerations: contrastive learning and term weighting. We start with a generative method and add a term weighting scheme that mimics the clustering method. We use a contrastive learning framework to help the language model instill pretrained knowledge and enable the model to focus on influential words. Our model, **UTopic** (Unified neural Topic model via contrastive learning and term weighting), has three stages: (1) identifying semantic clusters, (2) calculating term weights, and (3) estimating the latent topic distribution. In

*Corresponding Authors

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1802–1817

Stage 1, input documents with similar meanings are grouped into a single cluster. This procedure uses a pretrained language model, which extracts meaningful embeddings from the input document. Stage 2 computes the term weights for each word to represent the importance of each term, where high weights are given to frequent terms in one semantically coherent cluster (but not in other clusters). The model selects words with top-k term weights as the final word set for BoW representations. In Stage 3, the model estimates the latent topic distribution by reconstructing BoW representations from inputs. Term weights can represent each term's importance, allowing the model to learn a coherent topic. Contrastive learning manages the entire process of instilling pre-trained knowledge from the language model and generating a more distinct topic distribution with a predefined prior.

Experiments show that UTopic outperforms conventional methods in a wide range of scenarios, including human-annotated datasets (Table 1). When compared to SOTA models like CTM (Bianchi et al., 2021a), ClusterTM (Sia et al., 2020), and BERTopic (Grootendorst, 2022), our model consistently achieves the overall highest topic coherence score, while others excel in only one or two metrics (Table 2). In the 20-Newsgroups dataset, for example, the NPMI score increased by 3.65%. Our word set selection scheme (Stages 1&2 in Fig. 1) can be used as a standalone module to improve other models (Table 8). Major contributions are as follows:

- We present a unified method for combining the benefits of two topic modeling approaches (i.e., generative and clustering) into a single framework via term weighting.
- We modify the contrastive objective to discover semantically coherent clusters, which gives distinct and interpretable word sets for each cluster.
- Our model consistently outperforms existing baselines across multiple topic coherence measures and produces topics that align well with human labels.

Codes and implementation details of the model are available at a GitHub repository.[1]

[1] https://github.com/mingi-sid/utopic

2 Related Works

Topic modeling algorithms can be divided into two major streams: generative and clustering-based.

2.1 Generative approach

Most generative approaches follow the p-LSA assumption such that a set of tokens in each document are independently sampled from a mixture of topics (Hoffman, 1999). LDA (Blei et al., 2003) is a representative example that establishes the prior distribution following the p-LSA assumption. Many variants have been proposed, including Prod-LDA, Neural-LDA, and Hierarchical-LDA (Srivastava and Sutton, 2017; Blei et al., 2010). Some algorithms like NVDM, VTMRL, Wasserstein-LDA, and TAN-NTM use VAE for estimating the latent topic distribution (Miao et al., 2016; Gui et al., 2019; Nan et al., 2019; Panwar et al., 2021), while other algorithms, including DocNADE, use auto-regressive architecture (Larochelle and Lauly, 2012).

Adding a pretrained language model (or word embeddings) to traditional topic models has yielded promising results (Grootendorst, 2022; Liu et al., 2015; Qiang et al., 2017; Sia et al., 2020). Contextualized topic model (CTM), for example, obtains the embedding vector for each sentence using a model like BERT (Devlin et al., 2019) and bundles the embedding and the BoW (bag-of-words) (Bianchi et al., 2021a). The bundled vector based on VAE is then used to reconstruct BoW and uncover latent topics. However, these approaches have common limitations; word-level co-occurrences cannot fully convey words that represent themes. Naturally, the topic quality depends on the vocabulary set used for calculating word co-occurrences (Gui et al., 2019).

2.2 Clustering based approach

Keeping up with the recent achievement on deep representation learning in NLP domain (Devlin et al., 2019; Mikolov et al., 2013a), several works showed that discovering clusters over the sentence/word representation space can be one viable way to represent topics (Angelov, 2020; Grootendorst, 2022; Sia et al., 2020). Unlike generative approaches, they first extract high-level abstract features from the input document. Then, centroid-based clustering approaches are applied to identify dense clusters, which are finally regarded as topics. For example, Top2Vec (Angelov, 2020) utilizes Doc2Vec embedding (Le and Mikolov, 2014) to

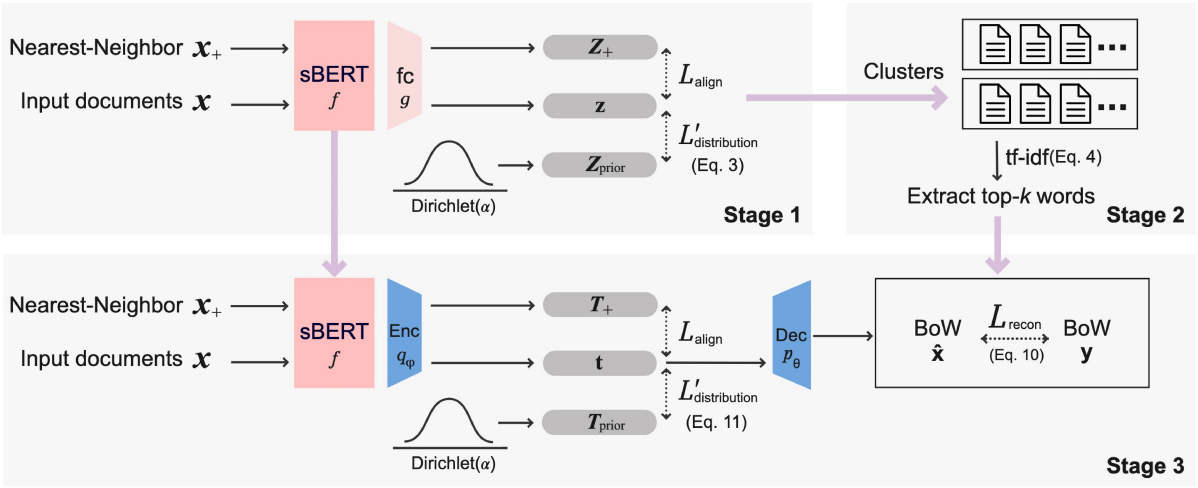


Figure 1: The illustration of the proposed neural topic model, UTopic . It divides documents into clusters based on their context (Stage 1). Then, term weights are computed according to every term's importance. As a topic word set, influential words with the highest term weight are retrieved (Stage 2). UTopic is fine-tuned by estimating the latent topic distribution with the help of term weights and contrastive learning (Stage 3).

jointly embed word and sentence representations, while BERTopic (Grootendorst, 2022) introduces BERT's sentence embedding to identify topic clusters. Another work demonstrates that word-level clusters can represent latent themes from the corpus (Sia et al., 2020). Nevertheless, these methods are clustering-based, so they cannot easily assign multiple topics to each document.

3 Methods

Problem definition: Let $\mathbf{x}$ be an input document and $\mathcal{X}$ be a collection of documents in the corpus (i.e., $\mathbf{x} \in \mathcal{X}$). A primary goal of our topic model is to estimate the topic distribution $\mathbf{p}$ from $\mathcal{X}$. Given a finite number of topics K, we extract top-k relevant words (hereafter referred to as *topic words*) $\mathcal{W}_i$ for each topic i to interpret the underlying theme and evaluate the topic quality. The topic word comes from the word set $\mathcal{W}_{dict}$, which has a size N (i.e., $\mathcal{W}_i \subset \mathcal{W}_{dict}$ for all i).

We propose to compute the *weight importance* of each term on semantically coherent clusters and then estimate the latent topic distribution by reconstructing the BoW representations based on term weights. Our idea is depicted in Figure 1, where a clustering model groups documents with similar contexts in the corpus (Section 3.1). Term weights are computed by treating each document cluster as a single unified document. If certain terms are significant in one group but not in others, we consider them *influential*. Top-k influential subject terms are selected as the word set $\mathcal{W}_{dict}$ for constructing BoW representation (Section 3.2). The model estimates topic distribution using the discovered word set and its term weight to recover the BoW representation of documents (Section 3.3). *Contrastive learning* (hereafter CL) improves representation quality at every phase, and the Dirichlet prior controls the entropy of the topic distribution. These steps are outlined in the following sections.

3.1 Stage 1: Document grouping via CL

Let f be a language model (e.g., BERT) that has been pre-trained, and g be an unsupervised classifier attached on top of f's sentence embedding (e.g., BERT's [CLS] token embedding). First, the models f and g are trained to classify incoming documents into K topic categories. CL facilitates this task with the InfoNCE loss (Oord et al., 2018), which learns an embedding space to maximize agreement between comparable examples while minimizing agreement between different instances. Assume that $\mathbf{z}$ is an input sample's embedding in the form of cluster assignment probability (i.e., $\mathbf{z} = \text{softmax}(g \circ f(\mathbf{x}))$), and that $\mathcal{Z}_+$ and $\mathcal{Z}_-$ are a collection of positive and negative samples' embeddings, respectively. The InfoNCE loss for each sample is defined as follows:

$$L_{\text{CL}}(\mathbf{z}) = -\log \frac{\sum_{\mathbf{z}' \in \mathcal{Z}_+} e^{\text{sim}(z,z')/\tau}}{\sum_{\mathbf{z}' \in (\mathcal{Z}_+ \cup \mathcal{Z}_-)} e^{\text{sim}(z,z')/\tau}} \quad (1)$$

$$= -\log \sum_{\mathbf{z}' \in \mathcal{Z}_+} e^{\text{sim}(z,z')/\tau} + \log \sum_{\mathbf{z}' \in (\mathcal{Z}_+ \cup \mathcal{Z}_-)} e^{\text{sim}(z,z')/\tau}$$

$$= -L_{\text{align}}(\mathbf{z}) + L_{\text{distribution}}(\mathbf{z}), \quad (2)$$

where sim(·) represents the similarity metric between two embeddings, and τ is a temperature value to control the entropy (Hinton et al., 2015). The InfoNCE loss can be broken down into two parts: *alignment loss* and *distribution loss*. The first term, alignment loss (L_{align}), directs the model to place positive pair embeddings closer together. The second term, distribution loss ($L_{\text{distribution}}$), pushes the embeddings of each sample as far apart as possible. This policy requires the model to match each instance's embedding into the predefined prior distribution with high entropy (Chen and Li, 2020; Wang and Isola, 2020).

The contrastive loss is used as follows in document clustering. For a given anchor $\mathbf{x}$, we establish a set of positive samples $\mathcal{Z}_+$ and negative samples $\mathcal{Z}_-$. Given that our model is unsupervised, there is no ground-truth label to determine which samples should be placed closer together in the embedding space. Instead, we mine the pretrained language model's top-1 nearest neighbor in the embedding space and consider it a positive sample (i.e., $\mathbf{x}_+$ in Fig. 1). The remaining instances in the same batch are considered negative samples.

Next, we guide the instance's embedding distribution to follow the prior with low entropy by modifying $L_{\text{distribution}}$ and adopting the optimal transport theory (Peyré et al., 2019; Rabin et al., 2011), a mathematical framework for transporting two sets of points while minimizing the transportation cost. The Sliced Wasserstein Distance (SWD) (Kolouri et al., 2019), a distance measure based on optimum transport that projects embeddings to random orthogonal subspaces and sums the 1-dimension Wasserstein Distance for each subspace, is one of its techniques. We replace $L_{\text{distribution}}$ with the SWD between embedding distribution and predefined low-entropy prior $\mathcal{Z}_{prior}$ (Eq. 3).

The model does not directly minimize entropy because documents can contain numerous topics; instead, it assigns the prior $\mathcal{Z}_{prior}$ to Dirichlet distribution with $\alpha < 1$ to assure low entropy of the embedding space. The alignment loss with adjusting factor λ is used to optimize the modified distribution loss at the same time.

$$L_{\text{stage1}}(\mathbf{z}) = -L_{\text{align}}(\mathbf{z}) + \lambda \cdot L'_{\text{distribution}}(\mathbf{z}),$$
$$\text{where } L'_{\text{distribution}}(\mathbf{z}) = \text{SWD}(\mathbf{z}, \mathcal{Z}_{prior}) \quad (3)$$

We use the cluster assignment probability $\mathbf{z}$ to discover K clusters after training (i.e., $\arg\max_j \mathbf{z}_j$ where j indexes the vector dimension).

3.2 Stage 2: Computing term weights

Given the newly identified cluster set $\mathcal{C}$ (i.e., $c \in \mathcal{C}$), the next stage calculates the term importance weight for each word from $\mathcal{C}$ and constructs the word set $\mathcal{W}_{dict}$. If words frequently appear in one cluster but not in others, they are deemed influential. This concept is consistent with TF-IDF (Ramos et al., 2003), which measures the importance of each word in a document by multiplying two terms: term frequency (TF) and inverse document frequency (IDF)[2]. Similar to (Grootendorst, 2022), we treat each document cluster as a single unified document and calculate the TF-IDF of each word for term weights. The TF-IDF measure for each cluster is as follows:

$$\text{tf-idf}(w, c, \mathcal{C}) = \text{tf}(w, c) \cdot \text{idf}(w, \mathcal{C}) \quad (4)$$

$$\text{tf}(w, c) = \frac{\text{freq}(w, c)}{\sum_{w' \in c} \text{freq}(w', c)} \quad (5)$$

$$\text{idf}(w, \mathcal{C}) = \log \frac{|\mathcal{C}|}{|\{c \in \mathcal{C} : w \in c\}|}, \quad (6)$$

where $\text{freq}(w, c)$ denotes the number of times a given word w is found in cluster c.

To build the word set $\mathcal{W}_{dict}$ for building BoWs in the next stage, we select the top-k influential topic words for each cluster with the highest term weight (Eq. 7). The value of k is adaptively chosen based on the dictionary size $N = |\mathcal{W}_{dict}|$.

$$\mathcal{W}_{dict} = \bigcup_{c \in \mathcal{C}} \{w | \text{ tf-idf}(w, c, \mathcal{C}) \text{ is top-}k \text{ in } c\} \quad (7)$$

3.3 Stage 3: Estimating topics via BoW reconstruction with term weights

The model is then trained to estimate the latent topic distribution in the final stage. We construct an encoder-decoder network on top of the pretrained language model f for training, following the literature (Bianchi et al., 2021a; Srivastava and Sutton, 2017). The encoder and decoder networks are referred to as q_ϕ and p_θ, respectively. The input document $\mathbf{x}$ is decomposed into the BoW representation $\mathbf{y}$ given the word set $\mathcal{W}_{dict}$. To create the context-aware representation, $\mathbf{x}$ is fed into the fixed backbone language model f. The encoder network q_ϕ and softmax function (Eq.8) are used to generate the latent topic distribution $\mathbf{t}$ over $\mathbf{x}$. The decoder

[2]A word's relative importance within a document is represented by TF, which divides the word count in a document by the total word count in the corpus. IDF represents the target word's uncommonness across the corpus as the logarithm of the total document count divided by the document count containing the target word.

network, followed by the softmax function, reconstructs the input document's BoW representation as $\hat{\mathbf{x}}$ (Eq. 9) with this topic distribution $\mathbf{t}$.

$$\mathbf{t} = \text{softmax}((q_\phi \circ f)(\mathbf{x})) \quad (8)$$

$$\hat{\mathbf{x}} = \text{softmax}(p_\theta(\mathbf{t})) \quad (9)$$

Our topic model with the backbone network f is trained by reconstructing the original document's BoW representation $\mathbf{y}$ from the latent topic distribution $\mathbf{t}$ (depicted as a box in Figure 1). To focus more on influential words when learning topics, term weights $w_\mathbf{y}$ from the previous stage are multiplied by the loss objective (Eq. 10). This leads the model to filter out unnecessary terms and discover more coherent and human-interpretable topics.

$$L_{\text{recon}}(\mathbf{x}, \mathbf{y}) = -\mathbb{E}_{q_\phi(\mathbf{t}|f(\mathbf{x}))}[w_\mathbf{y} \cdot \log(p_\theta(\mathbf{y}|\mathbf{t}))]. \quad (10)$$

The next step is to improve the representation quality by matching the encoder's posterior distribution to the predefined prior. This approach is modulated by contrastive loss, which assumes that documents with semantically identical content have similar topic distributions. As a positive sample, we use the top-1 nearest neighbor $\mathbf{x}_+$ from the embedding space of the pretrained language model. All other instances in a batch are considered negative samples. The model then uses the alignment loss L_{align} (Eq. 11) to induce the topic distribution of positive pairs ($\mathcal{T}_+$) to be similar. Distribution loss is used to match the posterior topic distribution to the prior distribution $\mathcal{T}_{prior}$ and maintain a low entropy. We set the prior $\mathcal{T}_{prior}$ to Dirichlet distribution with α<1 to discover distinctive topics from the corpus.

$$L_{\text{CL}}(\mathbf{t}) = -L_{\text{align}}(\mathbf{t}) + \lambda \cdot L'_{\text{distribution}}(\mathbf{t}),$$
$$\text{where } L'_{\text{distribution}}(\mathbf{t}) = \text{SWD}(\mathbf{t}, \mathcal{T}_{prior}) \quad (11)$$

The complete loss function is described in (Eq. 12). There are no adjusting weight parameters for objectives to reduce the tuning cost needed for hyper-parameters.

$$L_{\text{stage3}}(\mathbf{x}, \mathbf{y}) = L_{\text{recon}}(\mathbf{x}, \mathbf{y}) + L_{\text{CL}}(\mathbf{t}),$$
$$\text{where } \mathbf{t} \text{ is calculated by Eq. 8.} \quad (12)$$

4 Experiments

4.1 Performance evaluation

Datasets: For evaluation, we employ a variety of datasets with varying topics. 20-Newsgroups[3], Reuters-21578[4], Wikipedia, and BBC[5]. 20-Newsgroups is a collection of 11,314 posts shared in the newsgroups with a balanced set of topic labels. Reuters-21578 is a news post collection with 10,778 training data. We also use the Wikipedia dataset, containing randomly collected 20,000 paragraphs from 2021-09-22. Finally, BBC is a dataset of 2,225 documents excerpted from the BBC news website. All datasets are in English.

Evaluation: NPMI, C_p, and word2vec similarity are used as evaluation criteria. The first two statistics measure the degree of topic coherence between each topic's top-M term (M is set to 10 in our experiments) using a reference corpus. The last one, word2vec similarity, measures the semantic similarity between the top-M terms. The details of each metric are as follows.

- *NPMI*, the Normalized Point-wise Mutual Information, is a metric that scores high if the joint probability of pairs in the top-M words is greater than their marginal probability (Aletras and Stevenson, 2013). The co-occurrence probabilities can be measured from external corpora, such as Wikipedia (*NPMI-Wiki*) or the source data itself (*NPMI-In*).
- C_p is another coherence metric computed by an aggregation of Fitelson's confirmation measure based on the sliding window over the reference corpus. It is known to be one of the most well-correlated coherence scores with human ratings. (Röder et al., 2015).
- *word2vec similarity* evaluates topic coherence using an external embedding model to ensure fair comparisons with the baselines (Ding et al., 2018). The average cosine similarity value is reported across all pairs of top-M words' word2vec embeddings.

We used *Palmetto*, a publicly available tool for measuring topic quality[6] (Röder et al., 2015), and its version of Wikipedia dump to calculate *NPMI-Wiki* and C_p. For word2vec similarity, we used the pre-trained model from Mikolov et al. (2013b).

Implementation details: As the pre-trained language model, we used Sentence-BERT, *all-MiniLM-L6-v2* (Reimers and Gurevych, 2019). The

[3] http://qwone.com/~jason/20Newsgroups/.

[4] http://daviddlewis.com/resources/testcollections/reuters21578/.

[5] http://mlg.ucd.ie/datasets/bbc.html.

[6] https://github.com/dice-group/Palmetto.

	20-Newsgroups				Reuters-21578			
	NPMI-Wiki	NPMI-In	C_p	Word2vec	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0052	0.0652	0.0652	0.1719	-0.0723	0.0708	-0.1496	0.1549
NeuralLDA	-0.0180	-0.0229	-0.0540	0.1847	-0.0373	0.0651	-0.0347	0.1523
ETM	0.0192	**0.0988**	0.1066	0.2217	-0.0506	0.0791	-0.0855	0.1741
CTM	-0.0135	0.0464	-0.0296	0.1755	-0.0465	**0.1314**	-0.0658	0.1656
Top2Vec	0.0248	-0.0279	-1.4809	**0.2646**	-0.0532	0.0466	-0.2426	0.2126
ClusterTM	0.0134	-0.2810	0.0006	**0.2659**	**-0.0101**	-0.1745	-0.0703	**0.3082**
BERTopic	0.0351	-0.0733	0.1555	0.2257	-0.0656	0.0156	-0.2252	0.1759
UTopic	**0.0716**	**0.1016**	**0.3796**	0.2175	**-0.0117**	**0.1179**	**0.0852**	0.2162

	Wikipedia				BBC			
	NPMI-Wiki	NPMI-In	C_p	Word2vec	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0071	0.0296	0.0304	0.1458	-0.0721	-0.0186	-0.0711	0.1478
NeuralLDA	0.0133	0.0700	0.0862	0.1729	0.0017	0.0084	0.0562	0.1601
ETM	-0.0294	0.0319	0.0543	0.1484	-0.0270	0.0313	0.0643	0.1949
CTM	0.0610	**0.1547**	**0.2261**	0.1962	0.0339	**0.0991**	0.2622	0.1746
Top2Vec	-0.0215	-0.0876	-0.2111	0.1768	(0.0463)	(0.0124)	(0.1625)	(0.2495)
ClusterTM	0.0567	-0.3212	0.1835	**0.3398**	0.0243	0.0873	0.0739	**0.2959**
BERTopic	**0.0682**	-0.0295	0.2010	0.2196	(0.0225)	(0.0853)	(0.1652)	(0.1842)
UTopic	**0.0702**	**0.1129**	**0.2565**	0.2171	**0.0848**	**0.1057**	**0.4587**	0.2199

Table 1: Performance comparison over four datasets. Results are averaged over five trials from three different number of topics (K=10,20,50). The best or comparable performances are highlighted. In the table, parentheses indicate that the model failed for some topic counts; hence, only the success cases' results are averaged.

Method	NPMI-Wiki	NPMI-In	C_p	Word2vec	Total
LDA	7.0	5.0	6.3	7.8	6.5
NeuralLDA	5.5	5.0	5.3	6.8	5.6
ETM	6.0	3.5	5.0	5.0	4.9
CTM	4.3	2.0	3.3	5.8	3.8
Top2Vec	4.5	6.3	7.0	3.0	5.2
ClusterTM	3.5	6.8	4.5	**1.0**	3.9
BERTopic	4.0	6.0	3.8	3.5	4.3
UTopic	**1.3**	**1.5**	**1.0**	3.3	**1.8**

Table 2: Performance comparison based on the averaged rank for each evaluation metric across four datasets.

	20-Newsgroups		BBC	
	Match Acc.	NMI	Match Acc.	NMI
LDA	0.258	0.270	0.432	0.170
NeuralLDA	0.180	0.157	0.473	0.347
ETM	0.340	0.349	0.856	0.580
CTM	0.276	0.274	0.850	0.653
Top2Vec	0.264	0.260	0.636	0.520
ClusterTM	0.183	0.162	0.381	0.158
BERTopic	0.317	0.376	0.568	0.360
UTopic	**0.509**	**0.454**	**0.898**	**0.692**

Table 3: Performance comparison on topic assignment quality. The proposed model shows superb results.

Model	LDA	NeuralLDA	ETM	CTM	**Ours**
Match Acc.	0.112	0.098	0.204	0.298	**0.351**

Table 4: Performance on assigning multiple topics to the newly synthesized 20-NewsGroup dataset.

topic count K was set to 10, 20, and 50, while the corresponding Dirichlet prior α was set to 0.1, 0.05, and 0.02 (i.e., 1/# of topics), respectively[7]. λ was set to 1, and the vocabulary size to 2,000 following the literature (Bianchi et al., 2021a).

In stage 1, a single layer perceptron, g, is added to the Sentence-BERT model and trained for 100 epochs using the RangerLars (RAdam + LARS + Lookahead) optimizer with an initial learning rate of 0.001 and a decay factor of 0.99. Input text that exceeds the maximum sequence length is truncated. In stage 3, an encoder, q_ϕ, and a decoder, p_θ, are trained for 50 epochs using the Adam optimizer with a learning rate of 2E-2. The encoder is a three-layer MLP with dropout and batch normalization, and the decoder is a one-layer linear model without bias. The backbone network, f, is fixed during this stage.

Baselines: We used seven baselines for comparison. The first four are generative methods: (*1*) LDA (Blei et al., 2003), (*2*) NeuralLDA (Srivastava and Sutton, 2017), which is an LDA implementation with Autoencoding Variational

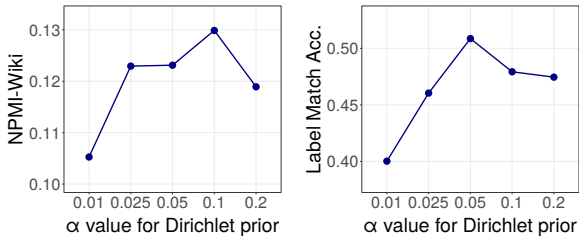
[7]We set the topic range to 10∼50 because clustering-based baselines (e.g., HDBSCAN in BERTopic or Top2Vec) failed when the topic size exceeded 70.

Setup	NPMI-Wiki	NPMI-In	C_p	Word2vec	Label acc.
UTOPIC	**.0653**	**.1211**	**.3629**	.2143	**.5087**
V1 (only stage 3)	.0409	.1052	.3144	.2222	.4539
V2 (no stage 3)	.0639	.1082	.3286	.2292	.3607
V3 (no L_{align} in L_{stage3})	.0485	.0920	.3020	.2275	.4974
V4 (no SWD loss in L_{stage3})	.0546	.1023	.3192	.2027	.4515
V5 (no term weight $w_{\mathbf{y}}$ in L_{stage3})	.0638	.1065	.3609	.2129	.4733
V6 (only L_{recon} in L_{stage3})	.0169	.0561	.2190	**.2327**	.4926
V7 (only L_{recon} without term weight $w_{\mathbf{y}}$ in L_{stage3})	.0082	.0465	.2149	.2290	.4834

Table 5: The ablation study results upon the overall architecture on 20-Newsgroup (K=20) confirm the substantial contribution of every model component we designed (V stands for *version*). The best results are highlighted.

	NPMI-Wiki	NPMI-In	C_p
Ours (AE)	**0.0653**	**0.1231**	**0.3709**
V8 (VAE)	0.0611	0.0943	0.3562

Table 6: Ablation study results on the model architecture (AE vs. VAE): 20-Newsgroups dataset is used.



(a) Effect on NPMI-Wiki (b) Effect on Label Acc.

Figure 2: The effect of the hyper-parameter α for Dirichlet prior on two evaluation metrics on 20-Newsgroups. Consistent performance is achieved by keeping α within a suitable range ($0.025 \sim 0.2$).

Inference for Topic Models (AVITM), (*3*) ETM (Embedded Topic Model) (Dieng et al., 2020), where the likelihood of a word is produced by the dot product between the word and topic embeddings, and (*4*) CTM (Contextualized Topic Model), where a vector representation of the document from SentenceBERT is additionally encoded on top of ProdLDA (Bianchi et al., 2021a). The remaining three are clustering methods: (*5*) Top2Vec (Angelov, 2020), where Doc2Vec is used for jointly embedding sentences and words to discover topic clusters, (*6*) ClusterTM (Sia et al., 2020), a word-level clustering-based approach, and (*7*) BERTopic (Grootendorst, 2022), which applies BERT's sentence embeddings to identify topics.

We applied common preprocessing procedures, such as removing English stop-words based on *NLTK* (Bird et al., 2009), non-alphabetic words, and words with less than three characters, and lemmatizing all tokens using the WordNet lemmatizer. All baselines used the same standard parameter values, such as topic count and vocabulary size, to ensure a fair comparison.

Results: Tables 1 and 2 show the performance comparisons and their summaries. UTopic provides the best or comparable topic coherence scores against other baselines over all datasets. Our technique consistently achieves satisfactory results in all other measures, while generative methods perform poorly on Word2vec similarity and clustering methods often fail on NPMI measures. Our model also shows the best scores based on the average rank of each evaluation metric.

Topic assignment quality analysis: It is important to check whether the discovered topics from each model are well aligned with the actual ground-truth labels. Two additional evaluation metrics were observed: *Label Matching Accuracy (Match Acc.)* and *Normalized Mutual Information (NMI)*. Given the estimated topic from each document, the Hungarian method (Kuhn, 1955) was applied to obtain the best bijection permutation mapping between the estimated topics and labels. Then, the label matching accuracy is computed with top-1 classification accuracy. The NMI measure is the mutual information between the mapping and labels with normalization to $0\sim1$ range.

Table 3 reports the comparison results among baselines over the 20-Newsgroups and BBC-News dataset because only these datasets contain human-annotated labels. Our model outperformed all baselines in accuracy by a large margin, even when compared with recent approaches, implying that it can generate more human-interpretable topics.

Evaluation for assigning multiple topics: We next conducted an experiment to assess UTopic 's

ability to assign multiple topics. We created a synthetic dataset with multiple topic labels by merging two random documents from 20-Newsgroups. Then, each topic model trained only using the original 20-Newsgroups dataset was evaluated using the newly created synthetic test set. The evaluation compares the accuracy of counting whether the top-2 estimated topics match the ground truth. Here, random selection will have an accuracy of 0.1 (=2/20). Table 4 shows that our model again outperformed baselines on this new test. Clustering-based approaches were excluded from the comparison because they are limited in assigning multiple topics.

4.2 Component analysis

Ablation studies iteratively remove one module or component to assess its unique contribution to the overall model. We now present results from the ablation study and hyper-parameters' effect over the 20-Newsgroup dataset.

Ablation study: The proposed model has three stages: document clustering, computing term weights, and estimating latent topic distribution. Our first ablation is a model with only stage 3 by directly estimating the latent topic distribution on top of the pretrained language model (*V1* in Table 5). The model without stage 3 utilizes the discovered clusters from the first stage as a topic embedding (*V2*). In this ablation, topic words extracted from the second stage are used for evaluating topic coherence. The next set of ablations remove each module in stage 3 to assess its effect (*V3–V7*). Table 5 reports the results for each ablation with five evaluation metrics: four topic coherence measures and label matching accuracy with human-annotated ground truths. The model with all components achieves the best topic coherence with label accuracy, indicating that each component plays an important role.

In contrast to previous works based on VAE (Bianchi et al., 2021a; Srivastava and Sutton, 2017), we used Autoencoder to enforce that the final topic distribution follows the predefined prior. To validate our design choice, we consider another ablation that uses the VAE architecture (*V8*). Experiments on 20 newsgroups show that using VAE can have a negative impact on overall performance (Table 6).

Hyper-parameter analysis: We study the effect of adjusting hyper-parameter α and λ in terms of various evaluation metrics. The α controls the density of Dirichlet prior to the match with the model's latent topic distribution. The smaller the α<1, the less the topic distribution overlaps. The results are summarized in Figure 2. Our model delivers favorable results for both measures when α is within a suitable range of 0.025∼0.2. However, setting α to a value that is too low can result in poor score. This is because decreasing entropy to an excessive degree can cause the model to learn incoherent topics and have a heterogeneous word distribution. We set α to $1/K$ (i.e., 0.05 for K=20), which is a common value in LDA (Rehurek and Sojka, 2011).

λ	NPMI-Wiki	NPMI-In	C_p	Word2vec
0.1	0.055	0.105	0.364	0.210
0.2	0.058	0.102	0.354	0.206
0.5	0.061	0.102	0.367	0.208
1	**0.072**	0.102	0.380	**0.218**
2	**0.072**	0.103	0.376	0.215
5	0.066	**0.112**	**0.382**	0.211
10	0.060	0.109	0.362	0.202

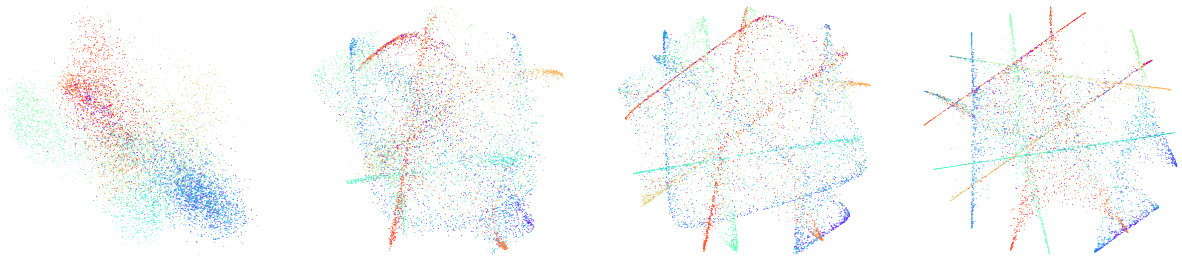
Table 7: Hyper-parameter analysis on the loss adjusting factor λ. Results are based on 20-Newsgroups.

Table 7 reports additional results on the effect of λ, the ratio between the alignment loss and the distribution loss (Eq. 11). Our model performs stable within a reasonable lambda range for most metrics. One exception is NPMI-Wiki, where the model appears influenced by extreme lambda values (λ=0.1 or 10).

4.3 Qualitative analysis

Cluster visualization: To understand the method's inner workings, we visualize the cluster assignment snapshots on the 20-Newsgroups dataset after stage 1 over four different training epochs in Figure 3. Ground-truth labels for 20 themes are displayed in different colors. More visually separable class-coherent clusters emerge as training advances. Evaluating the clustering performance via the Hungarian approach (Kuhn, 1955), epoch 30 in stage 1 already reaches a prediction accuracy of 48% without using any labels.

Application: The suggested topic word selection approach is a stand-alone module that can be used in various topic models. For example, we created the vocabulary set W_{dict} by extracting 2,000 topic words from our stage-2 model. Then we used W_{dict} to train three existing topic models: LDA, NeuralLDA, and CTM, and compared them with a traditional technique of employing the most



(a) Epoch=0, p=0.30 (b) Epoch=2, p=0.41 (c) Epoch=5, p=0.45 (d) Epoch=30, p=0.48

Figure 3: The 20-Newsgroups dataset was used to visualize intermediate clustering results at Stage 1 of UTopic . Colors represent the ground-truth class names (i.e., one of the 20 news topics), whereas dots reflect document cluster assignment. The p-value indicates that the model has already achieved a clustering accuracy of 48% after 30 epochs.

Setup	NPMI-Wiki	NPMI-Internal
LDA	-0.0056	0.0661
+ Our method	**-0.0016**	**0.0841**
ETM	0.0234	**0.0927**
+ Our method	**0.0404**	0.0752
CTM	-0.0086	0.1149
+ Our method	**-0.0021**	**0.1208**

Table 8: Performance improvement with our topic word extraction strategy on existing topic models on the 20-Newsgroups dataset (K=20).

frequent 2,000 words after ignoring stop-words. For all models, we limited the number of topics to 20. The findings are provided in Table 8 for the 20-Newsgroups dataset, which shows that using our context-aware topic word selection technique improves NPMI-Wiki by about 0.7∼1.7 percent point for all evaluated models.

5 Conclusion

This work proposed a new way to leverage the benefits of combining generative and clustering methods of topic modeling into a single framework. Diverse topic coherence measures have been used to assess the quality of topics. Compared to baselines that excel in only one or two coherence measures, UTopic showed consistent improvement across multiple coherence measures and discovered topics that align well with human annotations. Our method has a wide range of applicability because it can be added as a module to other existing approaches.

Limitations

This work comes with several limitations. First, because the nearest neighbor in the embedding space of the pretrained language model is considered a positive sample, the pretrained knowledge can impact overall performance. We plan to develop advanced text data augmentation and positive sample selection approaches for topic modeling to overcome this limitation. Second, since our clustering step needs a large batch (more than 128), the proposed method may be unattainable in some real-world scenarios due to memory constraints. To reduce this memory cost, we introduce a dynamic queue to save many samples' embeddings for every iteration (see Appendix for further details). However, training is only required once per dataset and is, therefore, acceptable.

Ethical Consideration

We acknowledge that presenting a data summary using topic modeling may not adequately represent the voice of minorities and that pre-trained knowledge from an online corpus may exacerbate such bias. When applying topic modeling in high-risk real-world scenarios, human annotators can help understand the predominant view of each topic to ensure that minor viewpoints are included. This is part of the larger challenge of AI ethics, and we plan to consider this challenge in the future.

Acknowledgement

This research was supported by the National Research Foundation of Korea (RS-2022-00165347), the Institute for Basic Science (IBS-R029-C2), and the 2022 Research Grant from Kangwon National University in South Korea.

References

Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *proc. of the International Conference on Computational Semantics (IWCS)*, pages 13–22.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2021a. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. In *proc. of the Association for Computational Linguistics (ACL)*, pages 759–766.

Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. 2021b. Cross-lingual contextualized topic models with zero-shot learning. In *proc. of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1676–1683.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

David M Blei, Thomas L Griffiths, and Michael I Jordan. 2010. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)*, 57(2):1–30.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Ting Chen and Lala Li. 2020. Intriguing properties of contrastive losses. *arXiv preprint arXiv:2011.02803*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *proc. of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Ran Ding, Ramesh Nallapati, and Bing Xiang. 2018. Coherence-aware neural topic modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 830–836.

Dan Fisher, Mark Kozdoba, and Shie Mannor. 2020. Topic modeling via full dependence mixtures. In *proc. of the International Conference on Machine Learning (ICML)*, pages 3188–3198.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Lin Gui, Jia Leng, Gabriele Pergola, Yu Zhou, Ruifeng Xu, and Yulan He. 2019. Neural topic model with reinforcement learning. In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3478–3483.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *stat*, 1050:9.

Thomas Hoffman. 1999. Probabilistic latent semantic analysis. In *proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. 2019. Generalized sliced wasserstein distances. *Advances in Neural Information Processing Systems*, 32:261–272.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. *Advances in Neural Information Processing Systems*, 25:2708–2716.

Jey Han Lau, Nigel Collier, and Timothy Baldwin. 2012. On-line trend analysis with topic models:# twitter trends detection topic model online. In *proc. of the International Conference on Computational Linguistics (COLING)*, pages 1519–1534.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *proc. of the AAAI Conference on Artificial Intelligence (AAAI)*.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *proc. of the International Conference on Machine Learning (ICML)*, pages 2410–2419.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *proc. of the International Conference on Machine Learning*, pages 1727–1736.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems (NeuIPS)*, 26.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic modeling with wasserstein autoencoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6345–6381.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Madhur Panwar, Shashank Shailabh, Milan Aggarwal, and Balaji Krishnamurthy. 2021. TAN-NTM: Topic attention networks for neural topic modeling. In *proc. of the Association for Computational Linguistics (ACL))*, pages 3865–3880.

Sungkyu Park, Sungwon Han, Jeongwook Kim, Mir Majid Molaie, Hoang Dieu Vu, Karandeep Singh, Jiyoung Han, Wonjae Lee, and Meeyoung Cha. 2021. Covid-19 discourse on twitter in four asian countries: Case study of risk communication. *Journal of Medical Internet Research*, 23(3):e23272.

Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607.

Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. 2017. Topic modeling over short texts by incorporating word embeddings. In *proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 363–374. Springer.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. 2011. Wasserstein barycenter and its application to texture mixing. In *proc. of the International Conference on Scale Space and Variational Methods in Computer Vision (SSVM)*, pages 435–446.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *proc. of the International Conference on Machine Learning (ICML)*, volume 242, pages 29–48.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3982–3992.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *proc. of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 399–408.

Mingi Shin, Sungwon Han, Sungkyu Park, and Meeyoung Cha. 2020. A risk communication event detection model via contrastive learning. In *proc. of the NLP4IF Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 39–43.

Suzanna Sia, Ayush Dalmia, and Sabrina J Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1728–1736.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations (ICLR)*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *proc. of the International Conference on Machine Learning (ICML)*, pages 9929–9939.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *proc. of the Web Conference (WWW)*.

A Appendix

A.1 Release

Codes and implementation details of the model are available at https://github.com/mingi-sid/utopic.

A.2 Dataset details

For evaluation, we employ a variety of datasets with varying topics: (1) 20-Newsgroups[8] contains 11,314 posts from the newsgroups from 20 number of balanced topic labels, (2) Reuters-21578[9] includes 10,778 news posts from Reuters newswire in 1987, (3) Wikipedia, comprising 20,000 abstracts that have more than 200 characters, was randomly sampled from the Wikipedia dump from 2021-09-22, and (4) NeurIPS[10] has 7,241 titles of articles from the NeurIPS conference. Since the topic modeling algorithm is an unsupervised approach, we use the whole dataset for evaluation without splitting.

A.3 Implementation details

For the pretrained language model f, we use Sentence-BERT, *all-MiniLM-L6-v2* specifically (Reimers and Gurevych, 2019). In stage 1, a fully connected layer g with a single layer perceptron is appended on top of the [CLS] representation from the Sentence-BERT. The total number of parameters is about 22.8M, including the parameters from the backbone language model. The original text is used as input, and any input tokens that exceed the maximum input sequence length are truncated. We train both f and g for 100 epochs. RangerLars (RAdam + LARS + Lookahead) optimizer with exponential learning rate decay is utilized. The initial learning rate and decay factor are set to 0.001 and 0.99, respectively.

In stage 3, we use three-layer MLP, including dropout and batch normalization as an encoder q_ϕ, and use one linear layer without bias as a decoder p_θ. For all phases, λ is set to 1, and the vocabulary size is set to 2,000. The backbone network f is fixed, and the encoder-decoder network is trained for 20 epochs. The Adam optimizer is used, with a learning rate of 2e-2. The number of topics K is set to 20, the batch size is set to 128, and the Dirichlet prior hyper-parameter α is set to 0.05 (i.e., 1/# of topics) for all phases.

Computing the Sliced Wasserstein Distance (SWD) between the embedding distribution and the prior requires a large batch for concise estimation. However, expanding the batch size is unattainable in many real-world scenarios due to memory constraints. We use a simple approach involving a dynamic queue, which does not require computing many samples' embeddings in a single iteration because embeddings from earlier iterations are saved. Embeddings computed from the current batch are concatenated with saved embeddings from the queue during training. Then, the stacked embeddings are finally used to calculate the distribution loss (Eq. 3, 11).

We should point out that the model's computing cost is not excessive. It took less than an hour for all datasets to perform all training phases with four A100 GPU processors.

A.4 Full evaluation results

In Table 9-20, we present the full performance evaluation results over four datasets with three different numbers of topics: 10, 20, and 50. For each experiment, results are averaged over five trials; the mean and standard deviation are reported. Also, to determine if a model successfully distinguish latent topics, we report results of topic diversity analysis for 20-Newsgroup, which shows that our method continues to outperform the baselines (Table 21). Top2Vec and ClusterTM are excluded in comparison, since they directly apply clustering algorithm (e.g.., DBSCAN) over the word set and inevitably make a complete split (i.e., topic diversity = 1), which is unfair to compare with.

A.5 Qualitative analyses

Table 22 reports the top 10 words for each topic extracted from our model. As discussed, we can find that the proposed model produces sufficiently representative words to interpret topics. For example, we can easily infer that topic #0 refers to the "computer" theme and topic #1 represents the "sports" theme. Our topic word selection strategy and contrastive learning framework successfully estimate the latent topics with improved interpretability.

[8] http://qwone.com/~jason/20Newsgroups/.

[9] http://daviddlewis.com/resources/testcollections/reuters21578/.

[10] https://www.kaggle.com/datasets/benhamner/nips-papers.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	0.0057±0.0023	0.0801±0.0057	0.0727±0.0103	0.1845±0.0028
NeuralLDA	-0.0158±0.0051	-0.0610±0.0158	-0.0429±0.0195	0.1741±0.0063
ETM	0.0052±0.0091	0.1219±0.0060	0.0527±0.0316	0.2027±0.0040
CTM	-0.0161±0.0080	0.1244±0.0135	-0.1415±0.0421	0.1818±0.0059
Top2Vec	0.0253±0.0017	0.0586±0.0020	-4.6113±0.0034	0.2677±0.0009
ClusterTM	0.0135±0.0025	-0.2870±0.0082	0.0160±0.0257	0.238±0.002
BERTopic	0.0609±0.0049	-0.0903±0.0180	0.2318±0.0063	0.2331±0.0070
UTopic	0.1069±0.0029	0.1130±0.0038	0.4850±0.0052	0.2416±0.0024

Table 9: Performance comparison over 20-Newsgroup with the number of topic $K = 10$. **[Note]** Mean and standard error over five trials are reported for Table 9-20.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0056±0.0051	0.0661±0.0074	0.0719±0.0118	0.1751±0.0042
NeuralLDA	-0.0227±0.0056	-0.0083±0.0090	-0.0634±0.0182	0.1933±0.0054
ETM	0.0234±0.0027	0.0927±0.0096	0.1207±0.0157	0.2247±0.0033
CTM	-0.0086±0.0048	0.1149±0.0138	0.0156±0.0173	0.1793±0.0055
Top2Vec	0.0302±0.0022	-0.0811±0.0046	0.1328±0.0089	0.2740±0.0033
ClusterTM	0.0154±0.0007	-0.2863±0.0062	0.0082±0.0103	0.2614±0.0021
BERTopic	0.0322±0.0035	-0.0563±0.0057	0.1515±0.0134	0.2210±0.0054
UTopic	0.0653±0.0030	0.1231±0.0059	0.3709±0.0098	0.2143±0.0029

Table 10: Performance comparison over 20-Newsgroup with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0157±0.0007	0.0495±0.0042	0.0511±0.0062	0.1561±0.0022
NeuralLDA	-0.0154±0.0020	0.0008±0.0104	-0.0557±0.0111	0.1867±0.0030
ETM	0.0290±0.0030	0.0817±0.0025	0.1465±0.0078	0.2377±0.0019
CTM	-0.0159±0.0024	-0.1000±0.0038	0.0371±0.0072	0.1654±0.0012
Top2Vec	0.0188±0.0033	-0.0610±0.0042	0.0357±0.0126	0.2520±0.0054
ClusterTM	0.0115±0.0016	-0.2698±0.0033	-0.0223±0.0097	0.2982±0.0037
BERTopic	0.0122±0.0036	-0.0734±0.0092	0.0830±0.0177	0.2230±0.0043
UTopic	0.0425±0.0064	0.0685±0.0038	0.2829±0.0194	0.1966±0.0051

Table 11: Performance comparison over 20-Newsgroup with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0771±0.0041	0.0854±0.0039	-0.1658±0.0158	0.1629±0.0034
NeuralLDA	-0.0446±0.0063	0.0536±0.0081	-0.0458±0.0193	0.1504±0.0031
ETM	-0.0569±0.0050	0.1031±0.0049	-0.1054±0.0137	0.1742±0.0021
CTM	-0.0467±0.0026	0.1262±0.0097	-0.0512±0.0093	0.1806±0.0025
Top2Vec	-0.0589±0.0069	0.0274±0.0093	-0.2803±0.0263	0.2206±0.0026
ClusterTM	-0.0164±0.0019	-0.2389±0.0096	-0.1115±0.0213	0.2942±0.0055
BERTopic	-0.0807±0.0022	0.0115±0.0060	-0.2706±0.0083	0.1870±0.0042
UTopic	-0.0138±0.0039	0.1225±0.0064	0.0884±0.0111	0.2350±0.0051

Table 12: Performance comparison over Reuters-21578 with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0663±0.0040	0.0793±0.0037	-0.1327±0.0122	0.1622±0.0024
NeuralLDA	-0.0374±0.0035	0.0584±0.0017	-0.0264±0.0058	0.1521±0.0022
ETM	-0.0552±0.0019	0.0733±0.0021	-0.1063±0.0080	0.1740±0.0019
CTM	-0.0438±0.0048	0.1259±0.0055	-0.0580±0.0082	0.1669±0.0029
Top2Vec	-0.0448±0.0020	0.0595±0.0094	-0.2350±0.0073	0.2207±0.0033
ClusterTM	-0.0084±0.0025	-0.1105±0.0062	-0.0814±0.0119	0.2998±0.0069
BERTopic	-0.0724±0.0016	0.0126±0.0052	-0.2647±0.0136	0.1609±0.0052
UTopic	-0.0094±0.0052	0.1231±0.0047	0.0832±0.0192	0.2198±0.0050

Table 13: Performance comparison over Reuters-21578 with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0734±0.0005	0.0477±0.0025	-0.1502±0.0050	0.1397±0.0019
NeuralLDA	-0.0300±0.0019	0.0833±0.0042	-0.0319±0.0122	0.1542±0.0022
ETM	-0.0399±0.0025	0.0609±0.0013	-0.0449±0.0078	0.1740±0.0024
CTM	-0.0491±0.0011	0.1420±0.0041	-0.0881±0.0050	0.1494±0.0011
Top2Vec	-0.0559±0.0024	0.0529±0.0054	-0.2125±0.0072	0.1966±0.0012
ClusterTM	-0.0056±0.0019	-0.1742±0.0034	-0.0181±0.0087	0.3307±0.0017
BERTopic	-0.0436±0.0030	0.0228±0.0055	-0.1403±0.0140	0.1797±0.0023
UTopic	-0.0117±0.0035	0.1081±0.0076	0.0841±0.0154	0.1938±0.0018

Table 14: Performance comparison over Reuters-21578 with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0095±0.0041	0.0519±0.0067	0.0345±0.0140	0.1591±0.0032
NeuralLDA	0.0018±0.0060	0.0301±0.0074	0.0744±0.0145	0.1677±0.0032
ETM	-0.0152±0.0042	0.0545±0.0027	0.0843±0.0077	0.1642±0.0032
CTM	0.0608±0.0064	0.1556±0.0077	0.2320±0.0168	0.2058±0.0052
Top2Vec	-0.0121±0.0064	-0.0774±0.0308	-0.2059±0.0439	0.1858±0.0013
ClusterTM	0.0501±0.0020	-0.3748±0.0047	0.1637±0.0087	0.3132±0.0148
BERTopic	0.0578±0.0093	0.0106±0.0097	0.1580±0.0341	0.2136±0.0044
UTopic	0.0663±0.0021	0.0895±0.0054	0.1987±0.0089	0.2127±0.0023

Table 15: Performance comparison over Wikipedia with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0072±0.0047	0.0618±0.0074	0.0377±0.0116	0.1462±0.0017
NeuralLDA	0.0121±0.0035	0.0792±0.0060	0.0784±0.0129	0.1628±0.0027
ETM	-0.0262±0.0016	0.0331±0.0030	0.0850±0.0074	0.1566±0.0007
CTM	0.0594±0.0037	0.1507±0.0026	0.2200±0.0084	0.2000±0.0019
Top2Vec	-0.0288±0.0038	-0.0730±0.0125	-0.2245±0.0265	0.1745±0.0046
ClusterTM	0.0533±0.0007	-0.3175±0.0068	0.1764±0.0084	0.3541±0.0043
BERTopic	0.0569±0.0044	-0.0374±0.0135	0.1786±0.0123	0.2164±0.0043
UTopic	0.0772±0.0037	0.1538±0.0044	0.2954±0.0092	0.2162±0.0018

Table 16: Performance comparison over Wikipedia with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0047±0.0046	-0.0250±0.0055	0.0189±0.0143	0.1322±0.0028
NeuralLDA	0.0258±0.0021	0.1008±0.0032	0.1058±0.0090	0.1882±0.0028
ETM	-0.0468±0.0023	0.0081±0.0022	-0.0064±0.0040	0.1245±0.0008
CTM	0.0628±0.0023	0.1579±0.0025	0.2263±0.0081	0.1827±0.0023
Top2Vec	-0.0238±0.0026	-0.1124±0.0069	-0.2030±0.0089	0.1700±0.0039
ClusterTM	0.0667±0.0029	-0.2714±0.0093	0.2104±0.0115	0.3520±0.0032
BERTopic	0.0898±0.0043	-0.0617±0.0042	0.2664±0.0146	0.2287±0.0021
UTopic	0.0670±0.0018	0.0955±0.0031	0.2754±0.0085	0.2224±0.0027

Table 17: Performance comparison over Wikipedia with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0746±0.0041	-0.0199±0.0009	-0.0684±0.0101	0.1573±0.0012
NeuralLDA	0.0001±0.0026	-0.0105±0.0124	0.0647±0.0076	0.1604±0.0061
ETM	-0.0212±0.0060	0.0441±0.0010	0.0829±0.0123	0.1865±0.0052
CTM	0.0436±0.0083	0.0714±0.0202	0.2543±0.0308	0.1822±0.0003
Top2Vec	0.0463±0.0022	0.0124±0.0101	0.1625±0.0056	0.2495±0.0002
ClusterTM	0.0255±0.0041	0.0656±0.0043	0.0588±0.0044	0.2732±0.0076
BERTopic	-0.0007±0.0145	0.0943±0.0006	0.0747±0.0074	0.1741±0.0068
UTopic	0.0938±0.0048	0.1256±0.0138	0.5388±0.0167	0.2265±0.0006

Table 18: Performance comparison over BBC with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0718±0.0033	-0.0205±0.0025	-0.0709±0.0094	0.1486±0.0010
NeuralLDA	0.0084±0.0010	0.0110±0.0089	0.0569±0.0036	0.1622±0.0026
ETM	-0.0333±0.0032	0.0251±0.0002	0.0416±0.0086	0.1908±0.0046
CTM	0.0289±0.0062	0.1109±0.0014	0.3254±0.0245	0.1715±0.0012
Top2Vec	N/A	N/A	N/A	N/A
ClusterTM	0.0339±0.0020	0.0990±0.0127	0.0908±0.0019	0.2858±0.0083
BERTopic	0.0456±0.0031	0.0762±0.0121	0.2556±0.0140	0.1943±0.0006
UTopic	0.0708±0.0061	0.1018±0.0113	0.3925±0.0124	0.2121±0.0006

Table 19: Performance comparison over BBC with the number of topic $K = 20$. **[Note]** N/A represents the model failed to detect topics for Table 19 and 20.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0700±0.0010	-0.0153±0.0032	-0.0739±0.0036	0.1375±0.0006
NeuralLDA	-0.0035±0.0022	0.0248±0.0014	0.0471±0.0087	0.1577±0.0018
ETM	-0.0264±0.0018	0.0246±0.0010	0.0683±0.0109	0.2073±0.0098
CTM	0.0291±0.0022	0.1149±0.0059	0.2069±0.0109	0.1701±0.0004
Top2Vec	N/A	N/A	N/A	N/A
ClusterTM	0.0136±0.0032	0.0973±0.0007	0.0721±0.0037	0.3286±0.0043
BERTopic	N/A	N/A	N/A	N/A
UTopic	0.0897±0.0024	0.0898±0.0070	0.4448±0.0022	0.2212±0.0027

Table 20: Performance comparison over BBC with the number of topic $K = 50$.

Model	LDA	NeuralLDA	ProdLDA	ETM	CTM	BERTopic	UTopic
Diversity	0.7	0.81	0.85	0.52	0.83	0.91	**0.92**

Table 21: Performance comparison over 20-Newsgroup in terms of topic diversity ($K = 20$).

Topic #	Top-10 words from each topic
0	drive, disk, scsi, floppy, hard, ide, controller, cd, card, rom
1	game, baseball, team, pitching, brave, year, player, run, hitter, hit
2	window, widget, xterm, server, lib, x11r5, motif, program, client, libxmu
3	clinton, president, government, bush, tax, administration, libertarian, fbi, id, party
4	encryption, key, chip, clipper, phone, algorithm, government, escrow, encrypted, patent
5	god, jesus, satan, christian, heaven, christ, hell, sin, faith, bible
6	voltage, game, input, motorola, chip, amp, circuit, amplifier, board, audio
7	space, orbit, nasa, satellite, spacecraft, lunar, moon, earth, comet, jupiter
8	christian, islam, religion, jew, church, morality, christianity, quran, muslim, objective
9	food, patient, disease, doctor, treatment, pain, blood, medical, infection, cancer
10	mac, cpu, simms, modem, computer, 040, scsi, system, clock, motherboard
11	armenian, israel, israeli, turkish, arab, jew, armenia, turk, turkey, greek
12	homosexual, hey, sex, gay, men, steve, homosexuality, serdar, life, sexual
13	file, window, zip, directory, bmp, printer, format, microsoft, convert, program
14	test, max, article, printer, eric, matthew, pl, 145, david, email
15	car, engine, ford, oil, battery, dealer, gt, taurus, auto, vehicle
16	card, monitor, video, driver, printer, window, color, vga, ati, bus
17	bike, motorcycle, honda, helmet, bmw, rider, ride, riding, shaft, rear
18	gun, weapon, crime, firearm, handgun, police, criminal, homicide, defense, assault
19	game, team, hockey, nhl, playoff, player, season, goal, ranger, detroit

Table 22: Top-10 words selected for each topic from our model. The model is trained over the 20-Newsgroups dataset and the number of topics is set to 20. The discovered word set is human-interpretable and substantially different from one another.