Disentangling Text Representation With Counter-Template For Unsupervised Opinion Summarization

Yanyue Zhang and Deyu Zhou*
School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China
{yanyuez98,d.zhou}@seu.edu.cn

Abstract

Approaches for unsupervised opinion summarization are generally based on the reconstruction model and generate a summary by decoding the aggregated representation of inputs. Recent work has shown that aggregating via simple average leads to vector degeneration, generating the generic summary. To tackle the challenge, some approaches select the inputs before aggregating. However, we argue that the selection is too coarse as not all information in each input is equally essential for the summary. For example, the content information such as "great coffee maker, easy to set up" is more valuable than the pattern such as "this is a great product". Therefore, we propose a novel framework for unsupervised opinion summarization based on text representation disentanglement with counter-template. In specific, a disentangling module is added to the encoder-decoder architecture which decouples the input text representation into two parts: content and pattern. To capture the pattern information, a counter-template is utilized as supervision, which is automatically generated based on contrastive learning. Experimental results on two benchmark datasets show that the proposed approach outperforms the state-of-the-art baselines on both quality and stability.

1 Introduction

With the unprecedented development of online interactive platforms, opinion summarization has received significant interest in natural language processing communities. Unlike other summarization tasks for news, Wikipedia, and medical treatment records, opinion summarization pays more attention to user opinions in product reviews, blog journals, and social media texts. Opinion summarization has great potential in many application scenarios.

Due to the lack of large-scale annotated data, opinion summarization is generally formalized as an unsupervised learning framework. A series of reconstruction models have been developed (Bražinskas et al., 2020a; Amplayo and Lapata, 2020; Elsahar et al., 2021; Amplayo et al., 2021a). The training goal is to reconstruct input text through an encoder-decoder architecture such as autoencoders (AE), variational autoencoders (VAE). As shown in the upper part of Figure 1, when generating summaries, the encoder is employed to aggregate the text representations from a set of texts via averaging (the 'Mean' module in Figure 1) to obtain a summary representation. The representation is used to generate the summary by the decoder.

Recently, it has been demonstrated in Iso et al. (2021) that simply averaging tends to generate generic summaries, as shown in the upper part of Figure 1. To tackle the challenge, a 'Select' module is incorporated in Coop (Iso et al., 2021) to strictly select input text. However, we argue that it has such a limitation: coarse-grained selection. There might be some information hidden in the abandoned reviews which is important for summarization. Moreover, the kept reviews might contain some redundant information, leading to generic or incorrect sentences, as shown in the middle of Figure 1.

In this paper, instead of treating text vectors as a whole, we assume that the text representation consists of *content* and *pattern* information. The content information is summary-relevant and related to specific product characteristics, such as aspects, emotions, and opinions illustrated as the orange part of texts in Figure 1. The pattern information marked in blue describes some common patterns that occur frequently in the corpus and can be used to describe most products. Both are valuable for reconstruction training. However, when text representations are aggregated to obtain a summary representation, the repeated pattern information is more conspicuous, which squeezes valuable content information and results in the generation of

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2023, pages 6344–6357
July 9-14, 2023

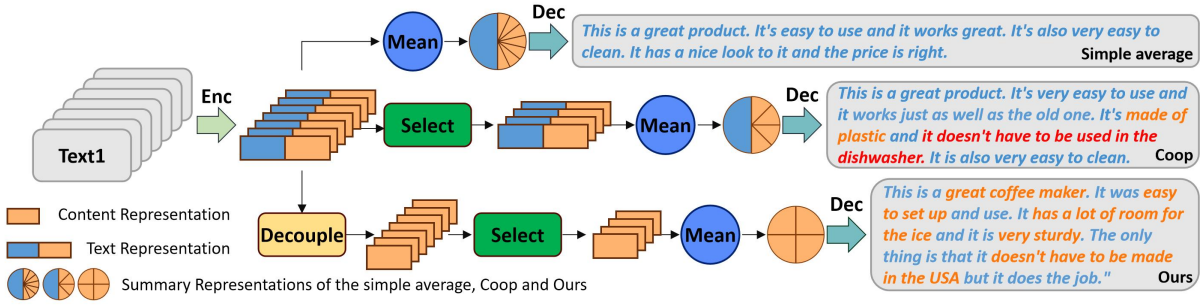

Figure 1: The processes of generating summaries in the approach using simple average (top), Coop (middle), and our proposed framework TRACE (bottom). The generated summaries are for the same product, a coffee maker.

overly generic summaries.

To separate the content information from the pattern information, the text representation needs to be disentangled. Recently, Qin et al. (2020) projects feature into orthogonal space to disentangle representation. However, it is not straightforward to disentangle in the same way as no supervision labels are available for the pattern information. To create supervision signals of the pattern information, we elaborately design a text template to capture the common semantic patterns shared in the corpus. It is called *counter-template* since the extracted pattern representation works as a negative guidance against the content information.

Therefore, we propose a novel framework based on Text Representation disentAnglement with Counter-templatE for unsupervised opinion summarization (TRACE). A disentangling module is added to the encoder-decoder architecture which orthogonally decouples the text representation into two parts: content and pattern. The content representation is used to reconstruct the input text and the pattern representation is supervised by the counter-template. Moreover, to construct the counter-template automatically, a novel approach based on contrastive learning and average strategy is proposed. A counter-template generator is trained to not only preserve the general generation constraints in VAE but also make all text representations in the same batch similar to each other using contrastive learning. The counter-template is obtained by decoding the average of several text representations.

The main contributions of this paper are summarized as follows:

- A novel framework based on **T**ext **R**epresentation disent**A**nglement with **C**ounter-templat**E** for unsupervised opinion summarization, TRACE, is proposed. A disentangling module is added in the encoder-decoder architecture which decouples the text representation into two parts: content and pattern.

- A novel approach based on contrastive learning is proposed to construct the counter-template automatically, which provides supervision for disentanglement.

- Experimental results on two benchmark datasets show that the proposed framework outperforms the state-of-the-art baselines on generation quality and stability.

2 Related Work

Opinion summarization generally focuses on short and numerous product reviews. According to whether summaries are extracted from original texts, they can be divided into extractive approaches and abstractive approaches.

Extractive approaches (Hu and Liu, 2004, 2006; Angelidis and Lapata, 2018; Zhao and Chaturvedi, 2020) capture the opinion of product reviews through emotional polarity or aspect information. Some methods attempt to obtain more flexible sentences through graphs (Ganesan et al., 2010) or decision theory (Di Fabbrizio et al., 2014; Carenini et al., 2013). Recently, Angelidis et al. (2021) leverage vector quantization (Van Den Oord et al., 2017) to capture a semantic sense. On this basis, SemAE (Chowdhury et al., 2022) uses dictionary learning to capture fine-grained and diverse semantics.

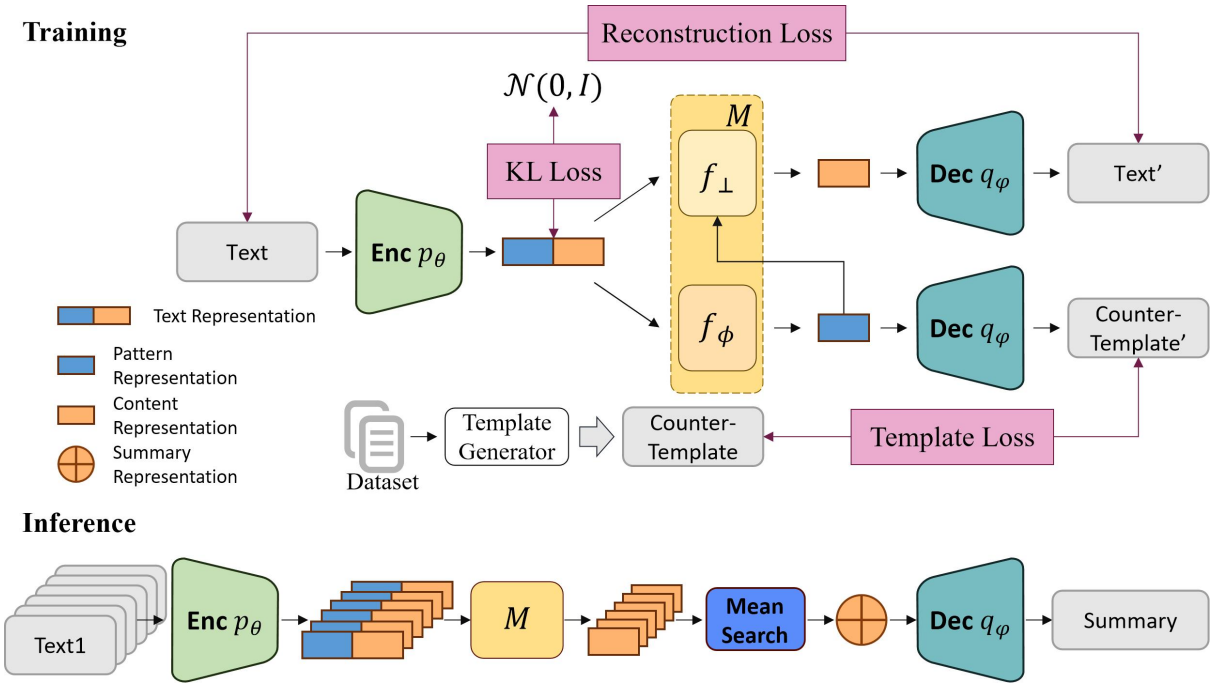

Figure 2: The overall architecture of TRACE. The disentanglement module **M** consists of the feature extractor f_ϕ and the orthogonal disentangled component $f_\perp$.

As text generation technology continues to evolve, a series of end-to-end abstractive methods have been developed for opinion summarization (Bražinskas et al., 2020a; Amplayo and Lapata, 2020; Iso et al., 2021). They use the encoder-decoder architecture and employ the average representation of inputs for summarization. Another type of method (Elsahar et al., 2021; Amplayo et al., 2021a; Suhara et al., 2020) focuses on modeling aspects and emotional information. Suhara et al. (2020) use a two-stage approach that first identifies opinion phrases and then uses the phrases to generate smooth sentences. Other works use aspect seed words (Elsahar et al., 2021; Amplayo et al., 2021a) or implicit aspect code (Amplayo et al., 2021b), making the summary generation more controllable.

Recently, Iso et al. (2021) show that averaging text representations simply tends to generate overly generic summaries. Then they use word overlapping to search for a better subset of input texts when inferring a summary. After that, Wassos (Song et al., 2022) optimizes the process of summary inference by approximating the Wasserstein barycenter to construct a better summary distribution. Although both Wassos and the proposed TRACE conduct text representation decoupling, TRACE differs from Wassos in the following two aspects: (1) different purposes. TRACE aims to tackle the coarse-grained selection problem in opinion summarization while Wassos employs text disentanglement for the better generation. (2) different ways. TRACE designs a novel way to separate the content and pattern information via the special design counter-template while Wassos employs a general way to disentangle syntactic and semantic spaces via the linearized parse tree sequence and the bag-of-words distribution (Bao et al., 2019).

3 Methodology

In this section, we describe TRACE, a novel disentangled framework with counter-template guidance for unsupervised opinion summarization. We first present the overview of the summarization model. Then, we introduce the counter-template generation approach based on a contrastive learning model and average inference strategy. Finally, we describe the detailed components of TRACE and explain how to train the model.

3.1 Overview of the Summarization Model

Figure 2 shows the overall architecture of TRACE. It contains four components, an encoder p_θ, a feature extractor f_ϕ, an orthogonal disentangled component $f_\perp$, and a decoder q_φ. f_ϕ and $f_\perp$ form the disentanglement module **M**.

Given a set of texts $T = \{t_1, \cdots, t_N\}$, where N is the number of texts. In the beginning, a counter-template $temp$ containing the pattern information

is automatically generated. In the training stage, each input text t_i is passed to the encoder $p_\theta(z_i \mid t_i)$ to get a text representation z_i. Then z_i is disentangled to content representation c_i and common pattern representation $\overline{p}_i$ by the disentanglement module **M**. $\overline{p}_i$ is used to predict the counter-template text $temp$ by decoder $q_\varphi(temp \mid \overline{p}_i)$. c_i is used to reconstruct the input text t_i through the same decoder $q_\varphi(t_i \mid c_i)$.

After training, each set of input texts T is passed to the encoder $p_\theta(z_i \mid t_i)$ and the disentanglement structure **M** to obtain a content representation set $C = \{c_1, \cdots, c_N\}$. Then the summary representation $\overline{c}$ is computed following Iso et al. (2021). The summary s is inferred from $\overline{c}$ by the decoder $q_\varphi(s \mid \overline{c})$.

3.2 Counter-Template Construction

To automatically generate the counter-template, a generator based on contrastive learning is constructed. Then the counter-template is inferred via average strategy. The counter-template generator includes an encoder E_s and a decoder D_s. To avoid the performance gap between counter-template generating and summary generating, we use bidirectional long short-term memory (Bi-LSTM) and a mean pooling layer as the encoder, LSTM as the decoder in the counter-template generator, same with the summary generator.

Generator Training To train the generator, the reconstruction loss and Kullback–Leibler (KL) regularization of VAE are employed to ensure the fluency of generation. Moreover, the contrastive loss is employed to make each latent vector z_i^s more similar. Contrastive learning (Chen et al., 2020) aims to shorten the distance between positive samples and extend the distance between negative samples. Here we design two ways of selecting positive samples: text-level and summary-level contrastive learning. The distance of the negative samples is set to a constant because of no negative samples.

For text-level contrastive learning, all input texts within the batch are treated as positive samples of each other. Given a set of texts $T = \{t_1, \cdots, t_N\}$, where N is the number of texts, the latent vector z_i^s can be computed by encoding text t_i via E_s. Following Chen et al. (2020), the normalized temperature-scaled cross-entropy loss is adopted as the contrastive objective:

$$\mathcal{L}_{text} = -[\sum_{i=1}^{n} \log \frac{\sum_{j=1}^{n} \mathbb{1}_{[j \neq i]} \exp\left(sim\left(z_i^s, z_j^s\right)/\tau\right)}{\exp\left(C/\tau\right)}]/n, \quad (1)$$

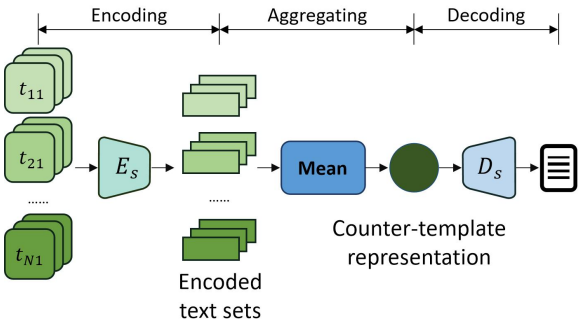

Figure 3: The procedure of counter-template generation.

where sim(·) indicates the cosine similarity function, n is the batch size in training, τ controls the temperature, $\mathbb{1}$ is the indicator and C is set to the maximum value of cosine similarity which is 1.0.

For summary-level contrastive learning, several sets of texts belonging to different objects (here, products or business) are input into the model. For the i-th object, there is a set of text $T_i = \{t_{i1}, \cdots, t_{iN_i}\}$, where N_i is the number of texts in T_i. After encoded by E_s, the summary representation $\overline{z}_i^s$ of i-th object is averaged from the latent vector set $Z_i^s = \{z_{i1}^s, \cdots, z_{iN_i}^s\}$. For each summary representation, its corresponding latent vector z_{ij}^s and other summary representations form the positive samples. The loss function is similar to that of text-level contrastive learning:

$$\begin{aligned}\mathcal{L}_{sum} = & \\ -[\sum_{i=1}^{n} \log \frac{\sum_{j=1}^{n} \mathbb{1}_{[j \neq i]} \exp\left(sim\left(\overline{z}_i^s, \overline{z}_j^s\right)/\tau\right)}{\exp\left(C/\tau\right)}]/n & \\ -[\sum_{i=1}^{n} \log \frac{\sum_{k=1}^{N_i} \exp\left(sim\left(\overline{z}_i^s, z_{ik}^s\right)/\tau\right)}{\exp\left(C/\tau\right)}]/n, &\end{aligned} \quad (2)$$

The overall loss function is defined as:

$$\begin{aligned}\mathcal{L}(\gamma, \psi) &= \mathcal{L}_{rec} + \mathcal{L}_{KL} + \mathcal{L}_{con} \\ \mathcal{L}_{rec}(\gamma, \psi) &= \sum_{i=1}^{N} \mathop{\mathbb{E}}_{p_\gamma\left(z_i^s \mid t_i\right)} [\log q_\psi(t_i \mid z_i^s)] \\ \mathcal{L}_{KL}(\gamma) &= \mathbb{D}_{KL}(p_\gamma(z_i^s \mid t_i) || p(z_i^s))\end{aligned} \quad (3)$$

where γ and ψ are the parameters of the encoder E_s and decoder D_s. $\mathcal{L}_{con}$ is the contrastive loss. For the text-level contrastive learning, $\mathcal{L}_{con} = \mathcal{L}_{text}$. For the summary-level contrastive learning, $\mathcal{L}_{con} = \mathcal{L}_{sum}$. We choose the standard Gaussian distribution as the prior distribution $p(z_i^s)$.

Counter-Template Inference When generating the counter-template, the average strategy is employed. As shown in Figure 3, there are three steps: encoding, aggregating, and decoding. Firstly, several sets of texts belonging to different objects are used as input to the encoder E_s. Then, an average

of the text representations is obtained as counter-template representation z_{temp}. Thus, z_{temp} contains a wide variety of information from different products or businesses. And the common pattern information dominates in z_{temp} compared with the content information. Finally, z_{temp} is used to generate the counter-template via the trained decoder D_s.

3.3 Model Components

The Encoder p_θ Iso et al. (2021) show that large pre-training language models such as BERT (Kenton and Toutanova, 2019) and GPT-2 (Radford et al.) do not necessarily lead to performance improvement in unsupervised opinion summarization. Therefore, we employ the BIMEANVAE model (Iso et al., 2021) which uses BiLSTM as encoder $p_\theta(z_i \mid t_i)$ and applies a mean pooling layer to the BiLSTM layer to obtain the primitive text representation z_i.

The Feature Extractor f_ϕ The common pattern representation $\overline{p}_i$ is extracted from z_i by the feature extractor $f_\phi(\overline{p}_i \mid z_i)$. Because the pattern information occurs in z_i naturally, a linear projection network is used as the feature extractor f_ϕ to compute the common pattern representation $\overline{p}_i$:

$$\overline{p}_i = W_P^T z_i + b_P$$

where W_P and b_P are the parameters. Due to the supervision of the invariant counter-template, $\overline{p}_i$ contains sample-independent pattern information.

The Orthogonal Disentangled Component $f_\perp$ Similar to Qin et al. (2020), the content representation c_i is disentangled from z_i by projecting z_i onto the orthogonal direction of the common pattern representation $\overline{p}_i$ in the orthogonal disentangled component $f_\perp$.

We first project z_i onto $\overline{p}_i$ to get p_i:

$$p_i = Proj(z_i, \overline{p}_i) \tag{4}$$

where $Proj$ is a projection function.

$$Proj(x, y) = \frac{x \cdot y}{\mid y \mid} \frac{y}{\mid y \mid} \tag{5}$$

where x, y are vectors.

Then the content representation c_i is obtained in the orthogonal direction of p_i:

$$c_i = Proj(z_i, (z_i - p_i)) \tag{6}$$

The Decoder q_φ Following Iso et al. (2021), LSTM is employed as the decoder q_φ with two functions. Firstly, the distribution $q_\varphi(t_i \mid c_i)$ is computed by the reconstruction of the input t_i from the content representation c_i. And $q_\varphi(temp \mid \overline{p}_i)$ is obtained by the prediction of the counter-template $temp$ via $\overline{p}_i$.

3.4 Model Training

The reconstruction loss, the template loss, and the KL loss are employed for model training.

The content representation c_i is used as the input of the decoder to reconstruct the input text t_i. And the reconstruction loss is defined as:

$$\mathcal{L}_{recon}(\theta, \phi, \varphi) = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{p_\theta(z_i|t_i)} [\log q_\varphi(t_i \mid c_i) f_\perp(c_i \mid p, z_i) f_\phi(p \mid z_i)] \tag{7}$$

The reconstruction loss improves the quality of the decoded text and forces the text representation z_i and content representation c_i to store content information.

The common pattern representation $\overline{p}_i$ is used to predict the counter-template $temp$. The loss is imposed by minimizing:

$$\mathcal{L}_{temp}(\theta, \phi, \varphi) = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{p_\theta(z_i|t_i)} [\log q_\varphi(temp \mid p) f_\phi(p \mid z_i)], \tag{8}$$

The template loss ensures the common pattern representation $\overline{p}_i$ contains common pattern information of the dataset.

Finally, following the typical VAE and variational inference, we add the regularizers L_{KL} which controls the amount of information in z_i by penalizing KL divergence of the estimated posteriors $p_\theta(z_i \mid t_i)$ from the corresponding priors. We choose the standard Gaussian distribution as the prior distribution $p(z_i)$. The final loss function is defined as:

$$\begin{aligned} \mathcal{L}(\theta, \phi, \varphi) &= \mathcal{L}_{temp} + \mathcal{L}_{recon} + \mathcal{L}_{KL} \\ \mathcal{L}_{KL} &= \mathbb{D}_{KL}(p_\theta(z_i \mid t_i) || p(z_i)) \end{aligned} \tag{9}$$

where θ, ϕ,and φ are the parameters of the model.

4 Experiments

To investigate the effectiveness of the proposed method, the best and the average results were performed on two datasets. In addition to the best performance, the mean and standard deviation were

Dataset	Category	Number	PCT
Amazon	Electronics	922957	59.80%
	Health	181017	11.73%
	Home	296053	19.18%
	Clothing	143474	9.30%
Yelp	Catering	3493385	74.98%
	Other	1165583	25.02%

Table 1: The number of reviews in different categories in Yelp and Amazon after preprocessing.

reported to evaluate the stability of the model. Besides, two analysis experiments were conducted for the counter-template.

4.1 Experimental Datasets

The experiments were conducted on two publicly available datasets, Amazon product reviews (He and McAuley, 2016) and Yelp business reviews (Chu and Liu, 2019). More details can be found in Appendix.

After preprocessing, we encountered a difference between Amazon and Yelp. Amazon has category labels for each product. However, there is only some meta information about categories on Yelp. Through data analysis in Table 1, we discovered the proportion of reviews from different businesses on Yelp is highly imbalanced with about 75% of reviews related to food and beverage. An overwhelming number of catering reviews affects our counter-template generation, leading to templates with obvious catering information. Therefore, we only keep reviews of food-related categories in training, validation, and test sets of Yelp to form Yelp-Res.

4.2 Baselines

Following prior work (Iso et al., 2021; Song et al., 2022), we compare TRACE with **TextVAE** (Ganesan et al., 2010), **Opinosis** (Ganesan et al., 2010), **MeanSum** (Chu and Liu, 2019), **Copycat** (Bražinskas et al., 2020b), **Coop** (Iso et al., 2021) and **Wassos** (Song et al., 2022). The detailed introduction is in Appendix.

4.3 Implementation Setting

We used Adam optimizer (Kingma and Ba, 2015) with a linear scheduler, whose initial learning rate is set to 10^{-3}. To mitigate the KL vanishing issue, we also applied KL annealing (Kingma et al., 2016; Li et al., 2019; Iso et al., 2021). For beam search in the generation, the beam size is set to 4. To generate summary-like texts, we employed the first-person pronoun blocking (Iso et al., 2021), which prohibits generating first-person pronouns (e.g. I, my, me) during summary generation. The ROUGE-1/2/L scores based on F1 (Lin and Hovy, 2002) are reported for automatic evaluation. For each model, we ran 4 times and reported the best, the mean, and standard deviation in Table 2. All experiments were conducted on NVIDIA GeForce RTX 3090. The training time of our model is about 6.5 hours, with 8 epochs on Amazon and 6 on Yelp-res.

Besides, we also reported the results of the human design counter-template (TRACE-h). In particular, we train a VAE summarization model and use simple averaging to generate several summaries. Then we clip and combine the generic sentences without content information in the generated summaries. In the process of combination, we also try to place the selected sentences in the same position as they originally had in the summary and control the length of the counter-template to about half of the average length of the golden summaries.

4.4 Results

As shown in the upper of Table 2, The *Max* part reports the best performance of models. Our framework (TRACE) significantly obtains the new state-of-the-art performance on both benchmark datasets with counter-templates whether the automated (TRACE-a) or the human design (TRACE-h). On Yelp-Res, the model with human-designed counter-templates (TRACE-a) is better than the auto-generated ones. But on Amazon, the situation is reversed. Probably because we tested 12 different counter-templates on Amazon than 4 on Yelp, due to the analysis experiments. The more auto-generated counter-templates are tested, the more likely to find a better one, which can even exceed the human design in performance.

As shown in the lower of Table 2, the *Mean* part reports the mean and standard deviation of the model performance, which represent the stability of the model. In this perspective, TRACE significantly outperforms all competing models on both Amazon and Yelp-Res.

In general, the performance of TRACE-h is better than TRACE-a both in the *Max* and the *Mean*. We consider that this is because the human design process eliminates all the unidentified content information in the counter-templates and ensures the purity of the pattern information.

	Amazon			Yelp-Res		
	R1	**R2**	**RL**	**R1**	**R2**	**RL**
			Max			
TextVAE‡	22.87	2.75	14.46	25.42	3.11	15.04
Opinosis‡	28.42	4.57	15.50	24.88	2.78	14.09
MeanSum	28.32	3.28	16.19	28.27	3.54	16.00
Copycat	31.17	6.49	19.71	28.31	5.38	17.72
Coop	36.93	7.05	21.34	34.21	6.49	19.19
Wassos(T)	30.80	6.38	19.45	27.43	6.03	18.38
Wassos(0)	33.24	7.25	21.31	25.50	4.84	17.78
TRACE-a	**37.90**	**7.53**	**22.48**	34.39	7.07	19.90
TRACE-h	37.34	7.51	21.49	**34.80**	**7.81**	**20.10**
			Mean			
MeanSum	27.63 ± 0.72	3.05 ± 0.22	15.87 ± 0.51	28.00 ± 0.34	3.53 ± 0.19	15.80 ± 0.14
Copycat	29.46 ± 1.50	5.18 ± 0.44	19.06 ± 0.67	27.21 ± 0.92	4.98 ± 0.52	17.74 ± 0.28
Coop	35.00 ± 1.11	5.85 ± 0.77	19.56 ± 0.99	33.24 ± 0.58	6.57 ± 0.23	19.18 ± 0.14
Wassos(T)	27.90 ± 2.42	5.65 ± 0.74	18.48 ± 1.10	27.95 ± 1.40	5.82 ± 0.26	18.28 ± 0.06
Wassos(0)	30.78 ± 3.84	6.75 ± 1.07	19.09 ± 1.45	24.65 ± 1.46	4.35 ± 0.25	16.30 ± 0.73
TRACE-a	36.33 ± 0.29	7.01 ± 0.32	21.30 ± 0.42	$\mathbf{34.11} \pm 0.26$	6.74 ± 0.16	19.41 ± 0.29
TRACE-h	$\mathbf{36.52} \pm 0.41$	$\mathbf{7.16} \pm 0.23$	$\mathbf{21.32} \pm 0.01$	34.08 ± 0.45	$\mathbf{6.86} \pm 0.47$	$\mathbf{19.56} \pm 0.34$

Table 2: Experimental results on Amazon and Yelp-Res. The bold and underlined scores denote the best and second-best scores respectively. The a and h of TRACE represent the automatic generation and human design. ‡means the results are from (Bražinskas et al., 2020b).

	R1	**R2**	**RL**
w/o con	$\mathbf{36.51} \pm 0.62$	6.95 ± 0.31	21.28 ± 0.48
w/ sum-b	36.33 ± 0.29	$\mathbf{7.01} \pm 0.32$	$\mathbf{21.30} \pm 0.42$
w/ sum-l	35.50 ± 0.60	6.57 ± 0.31	20.98 ± 0.44
w/ text-b	36.29 ± 0.34	6.82 ± 0.21	21.14 ± 0.33
w/ text-l	35.84 ± 0.74	6.52 ± 0.50	21.09 ± 0.51
human	$\mathbf{36.52} \pm 0.41$	$\mathbf{7.16} \pm 0.23$	$\mathbf{21.32} \pm 0.10$

Table 3: Results of the different counter-template generators. The w/o con is without contrastive learning. The sum and text are summary-level and text-level contrastive learning. b and l is the best checkpoint in Rouge scores or the last.

4.5 Counter-Template Analysis

For the proposed counter-template generation method, we separately conducted two experiments on different contrastive learning methods in Table 3 and different generation inputs in Table 4. Some examples of counter-templates are shown in Table 5.

Because using golden validation and test sets for counter-template generation might leak some information, they are only used for analysis. Besides, we randomly sampled some products from the training set and grouped their reviews as set 1. Repeating this procedure yielded set 2. Then, they were mixed to obtain the third set, labeled as 1+2. For each model, we test three counter-templates generated from set 1, set 2, and set 1+2. Of the three results, the result with the highest mean value will be broadcasted with the standard deviation. The best results are in Table 7 inside the appendix.

As shown in Table 3, the performance of the model is slightly affected by the ablation of contrastive learning (w/o con), which means our average generation strategy plays a major role. In general, summary-level contrastive learning is more effective than text-level. The first possibility is that the training batch size of the text-level counter-template generator has to be a quarter of the summary-level model because of the time complexity. Besides, We suspect that summary-level contrastive learning will not only make the text representations more similar to each other but also make them more similar to generic summary representations.

In addition, We test the performance of summarization on the golden validation and test sets during the training of the counter template generator. Choosing the checkpoint that performs best on the

	R1	R2	RL	Number
Coop	35.00 ± 1.11	5.85 ± 0.77	19.56 ± 0.99	-
1	36.03 ± 0.26	6.35 ± 0.17	20.93 ± 0.27	294
2	35.42 ± 0.25	$\underline{6.97} \pm 0.28$	20.52 ± 0.11	221
1+2	$\mathbf{36.33} \pm 0.29$	$\mathbf{7.01} \pm 0.32$	$\mathbf{21.30} \pm 0.42$	515
1-large	36.12 ± 0.42	6.72 ± 0.42	20.97 ± 0.64	490
2-large	36.00 ± 0.19	6.49 ± 0.16	20.81 ± 0.30	475
1-l+2-l	35.68 ± 0.72	6.61 ± 0.20	20.66 ± 0.44	965
Val	35.65 ± 0.49	$\underline{6.75} \pm 0.11$	20.78 ± 0.17	224
Test	$\underline{36.28} \pm 0.58$	6.70 ± 0.34	$\underline{20.99} \pm 0.36$	256

Table 4: Results of counter-template generation via different input sets on Amazon. **Number** is the number of text to generate a counter-template. 'Val' and 'Test' represent the golden validation and test sets. The others are the extracted text sets from the train data randomly.

Test	This is a great product. It's very easy to use and clean. The only thing that would be better is the size of this one, but it's not a big deal for the money.
1	This is a great product. It's very easy to use and clean. The only problem is that it doesn't take up much room in the kitchen, and has a nice feel to it.
1+2	This is a great product. It's very easy to use and it works well. The only drawback is that the blue light is a little weak, but it does not have to be in the way.
1-large	This is a great product. It's very easy to use and clean. The only drawback is that it has to be a little bit more expensive than the brand but it works.
1-l+2-l	This is a great product. It's very easy to use and it works well. The only drawback is that the blue light is a little weak, but you can't get it to work with.

Table 5: Generated counter-template examples on Amazon via the input set in Table 4.

summarization test (b) achieves better results than the last (l). It may be because the counter-template generation model with better Rouge performance in the summary generation is more likely to learn pattern information which the corresponding summarization model will learn.

Since using summary-level contrastive learning and selecting the model with the best rouge (sum-b) perform best, we build on this to conduct the analysis experiment using different inputs for the counter-template generation. The set labeled as 1-large is obtained by randomly selecting products from the training set based on 1, and 2-large the same.

As shown in Table 4, although all the results outperform the SOTA baseline, different counter-templates lead to obviously different results. As the examples shown in Table 5, the counter-templates are very similar to each other. There is no obvious relationship between the number of input reviews and the generated counter-templates or the performance of the model. In general, the advantage of automatic generation is continuously generating different counter-templates by extracting different input sets to achieve higher performance. But at present, only experiments can determine the impact of the template on the model, which consumes time and computing resources.

5 Conclusion

To avoid generating generic summaries, we propose TRACE, a novel framework for unsupervised opinion summarization based on text representation disentanglement with counter-template. The additional disentanglement module inside the encoder-decoder architecture decouples the pattern and content information in the text representation under the guidance of the special counter-template. Experimental results on Amazon and Yelp-Res show the proposed approach outperforms the state-of-the-art baselines on both quality and stability.

6 Limitations

As shown in Table 1, an overwhelming number of catering reviews on Yelp makes the counter-templates with obvious catering information. For example, "This is a great place for a quick bite to eat. The food is delicious and the staff is very friendly. They have a good selection of beer and wine. The place is always busy, but it's worth the wait." In this case, the pattern information in the text is not consistent with other businesses irrelevant to catering. As shown in Table 6, although our method has a slight improvement over the previous methods in the mean and standard deviation, it is only comparable to the SOTA at the best performance. Since the counter-template is exactly the same text for the whole data set, the performance of the model is affected perhaps when the pattern information from different texts in the data set has large differences. When we extract the restaurant-related parts of the dataset as Yelp-Res that have more similar pattern information, our model performs better.

7 Acknowledgments

We would like to thank anonymous reviewers for their valuable comments and helpful suggestions. This research work is supported by the Big Data Computing Center of Southeast University. We would also like to thank Professor Yulan He for her valuable suggestions for our paper. This work was funded by the National Natural Science Foundation of China (62176053).

References

Reinald Kim Amplayo, Stefanos Angelidis, and Mirella Lapata. 2021a. Aspect-controllable opinion summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Reinald Kim Amplayo, Stefanos Angelidis, and Mirella Lapata. 2021b. Unsupervised opinion summarization with content planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Reinald Kim Amplayo and Mirella Lapata. 2020. Unsupervised opinion summarization with noising and denoising. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Stefanos Angelidis and Mirella Lapata. 2018. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3675–3686.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xinyu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020a. Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020b. Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics.

Giuseppe Carenini, Jackie Chi Kit Cheung, and Adam Pauls. 2013. Multi-document summarization of evaluative text. *Computational Intelligence*, 29(4):545–576.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*. PMLR.

Somnath Basu Roy Chowdhury, Chao Zhao, and Snigdha Chaturvedi. 2022. Unsupervised extractive opinion summarization using sparse coding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1209–1225.

Eric Chu and Peter Liu. 2019. Meansum: a neural model for unsupervised multi-document abstractive summarization. In *International Conference on Machine Learning*.

Giuseppe Di Fabbrizio, Amanda Stent, and Robert Gaizauskas. 2014. A hybrid approach to multi-document summarization of opinions in reviews. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 54–63, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Hady Elsahar, Maximin Coavoux, Jos Rozen, and Matthias Gallé. 2021. Self-supervised and controlled multi-document opinion summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010).

Ruining He and Julian McAuley. 2016. 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*.

Minqing Hu and Bing Liu. 2004. Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760.

Minqing Hu and Bing Liu. 2006. Opinion extraction and summarization on the web. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, page 1621–1624. AAAI Press.

Hayate Iso, Xiaolan Wang, Yoshihiko Suhara, Stefanos Angelidis, and Wang-Chiew Tan. 2021. Convex aggregation for opinion summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3885–3903.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *EMNLP/IJCNLP (1)*.

Chin-Yew Lin and Eduard Hovy. 2002. Manual and automatic evaluation of summaries. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*, pages 45–51.

Qi Qin, Wenpeng Hu, and Bing Liu. 2020. Feature projection for improved text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8161–8171.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

Jiayu Song, Iman Munire Bilal, Adam Tsakalidis, Rob Procter, and Maria Liakata. 2022. Unsupervised opinion summarisation in the wasserstein space.

Yoshihiko Suhara, Xiaolan Wang, Stefanos Angelidis, and Wang-Chiew Tan. 2020. Opiniondigest: A simple framework for opinion summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Chao Zhao and Snigdha Chaturvedi. 2020. Weakly-supervised opinion summarization by leveraging external information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9644–9651.

A Dataset Preparation

Following the similar pre-processing way (Bražinskas et al., 2020b; Chu and Liu, 2019; Iso et al., 2021), only products with a minimum of 10 reviews were used within the maximum of 128 tokens each respectively. Besides reviews used for training, these two datasets also contain gold-standard summaries for 200 and 60 sampled objects, respectively. In Amazon, each product is with 3 human-created summaries, released by (Bražinskas et al., 2020b). And only 1 human-created summary for each business in Yelp, released by (Chu and Liu, 2019). For both datasets, the summaries are manually created from 8 input reviews. We used the same dev/test split, 100/100 for Yelp and 28/32 for Amazon, released by their authors for our experiments.

B Baselines

The following approaches are chosen as baselines: **TextVAE** (Ganesan et al., 2010): A vanilla text VAE model that has a unidirectional LSTM layer and uses the last hidden state to calculate the posterior distribution. **Opinosis** (Ganesan et al., 2010): A graph-based summarization framework that generates concise abstractive summaries with highly redundant opinions. **MeanSum** (Chu and Liu, 2019): An unsupervised multi-document summarization model that minimizes the auto-encoder reconstruction loss and the similarity loss. **Copycat** (Bražinskas et al., 2020b): An unsupervised multi-document summarization model that captures the dependency relationship between the product and reviews by defining a hierarchical VAE. **Coop** (Iso et al., 2021): An unsupervised multi-document summarization framework that searches input combinations for the summary aggregation using the input-output word overlapping. **Wassos** (Song et al., 2022): An unsupervised multi-document summarization framework that uses the Wasserstein barycenter of the semantic and syntactic distributions to obtain the summary.

	Yelp			Yelp-Res		
	R1	**R2**	**RL**	**R1**	**R2**	**RL**
			Max			
Coop	34.36	**7.12**	**19.96**	34.21	6.49	19.19
TRACE-a	34.24	7.00	19.68	34.39	7.07	19.90
TRACE-h	**34.66**	7.05	19.70	**34.80**	**7.81**	**20.10**
			Mean			
Coop	33.92 ± 0.35	6.52 ± 0.35	19.13 ± 0.42	33.23 ± 0.58	6.57 ± 0.23	19.18 ± 0.13
TRACE-a	33.90 ± 0.57	6.60 ± 0.22	19.29 ± 0.27	$\mathbf{34.11} \pm 0.26$	6.67 ± 0.16	19.41 ± 0.29
TRACE-h	$\mathbf{34.13} \pm 0.58$	$\mathbf{6.64} \pm 0.25$	$\mathbf{19.44} \pm 0.32$	34.08 ± 0.45	$\mathbf{6.86} \pm 0.47$	$\mathbf{19.56} \pm 0.34$

Table 6: Results on Yelp and Yelp-Res. The bold and underlined scores denote the best and second-best scores respectively.

	R1	**R2**	**RL**
w/o con	**37.90**	7.53	**22.48**
w/ sum-b	36.92	7.13	21.84
w/ sum-l	37.41	7.00	21.75
w/ text-b	36.60	7.21	21.79
w/ text-l	37.32	**7.65**	21.97
human	37.34	7.51	21.49

Table 7: The maximum of Results using the different counter-template generators. The w/o con means without contrastive learning. The sum and rev are summary-level and text-level contrastive learning. And b and l is the best checkpoint in Rouge or the last.

	R1	**R2**	**RL**	**Number**
Val	36.18	6.80	21.02	224
Test	**37.61**	7.10	21.63	256
1	36.32	6.66	21.47	294
2	36.73	6.74	21.21	221
1+2	36.58	**7.29**	**21.69**	515
1-large	36.54	7.10	21.64	490
2-large	36.04	6.52	20.83	475
1-l+2-l	36.80	6.70	21.69	965

Table 8: Best results of counter-templates generated via different inputs on Amazon. **number** is the number of input text to generate a counter-template. The bold and underlined scores denote the best and second-best scores respectively.

Val	This is a great product. It's very easy to use and it holds up well. The only problem is that the cord is a little weak, but it doesn't seem to be as good for the price.
Test	This is a great product. It's very easy to use and clean. The only thing that would be better is the size of this one, but it's not a big deal for the money.
1	This is a great product. It's very easy to use and clean. The only problem is that it doesn't take up much room in the kitchen, and has a nice feel to it.
2	This is a great product. It's very easy to use and it works well. The only drawback is that the blue light is a little weak, but you can't get it to work for the price.
1+2	This is a great product. It's very easy to use and it works well. The only drawback is that the blue light is a little weak, but it does not have to be in the way.
1-large	This is a great product. It's very easy to use and clean. The only drawback is that it has to be a little bit more expensive than the brand but it works.
2-large	This is a great product for the price. It's very comfortable and looks good. The only problem is that it doesn't hold up to the side of the screen, but it is a nice size.
1-l+2-l	This is a great product. It's very easy to use and it works well. The only drawback is that the blue light is a little weak, but you can't get it to work with.

Table 9: Generated counter-template via the inputs in Table 4.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
6

☒ A2. Did you discuss any potential risks of your work?
6

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

3

☑ B1. Did you cite the creators of artifacts you used?
3.1 3.4

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
describe in 4.2

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.2

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
No response.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4.4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4.4

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.