Unsupervised Class-Specific Abstractive Summarization of Customer Reviews

Thi Nhat Anh Nguyen, Mingwei Shen, Karen Hovsepian
PARS *, Amazon.com, Seattle
{tnguynr, mingweis, khhovsep}@amazon.com

Abstract

Large-scale unsupervised abstractive summarization is sorely needed to automatically scan millions of customer reviews in today's fast-paced e-commerce landscape. We address a key challenge in unsupervised abstractive summarization – reducing generic and uninformative content and producing useful information that relates to specific product aspects. To do so, we propose to model reviews in the context of some topical classes of interest. In particular, for any arbitrary set of topical classes of interest, the proposed model can learn to generate a set of class-specific summaries from multiple reviews of each product without ground-truth summaries, and the only required signal is class probabilities or class label for each review. The model combines a generative variational autoencoder, with an integrated class-correlation gating mechanism and a hierarchical structure capturing dependence among products, reviews and classes. Human evaluation shows that generated summaries are highly relevant, fluent, and representative. Evaluation using a reference dataset shows that our model outperforms state-of-the-art abstractive and extractive baselines.

1 Introduction

As volume and scope of online customer reviews continue to explode, so does the need for online sellers to digest and draw insights to improve products. Today, both sellers and customers manually sift through hundreds of reviews across competing products to decipher systemic or trending concerns from isolated or irrelevant issues. Opinion summarization technology run across millions of reviews has drawn much attention due to its potential for streamline defect discovery, trend analysis, product development, provided that summaries are informative and fluent.

This work concerns abstractive summarization of product reviews. Abstractive summaries which contain new phrases and words not found in original documents are often more fluent, more concise and more informative given the same length than extractive ones which only contain words, phrases and sentences from the original documents.

Current state-of-the-art methods for abstractive summarization are based on supervised deep-learning language models (Sutskever et al., 2014; Nallapati et al., 2016; Gu et al., 2016; See et al., 2017), and rely on large amount of human-written ground-truth summaries. Because text summarization systems are domain-sensitive (Isonuma et al., 2017) and ground-truth opinion summaries are expensive to obtain, unsupervised opinion summarization has recently garnered significant attention with noteworthy efforts (Ma et al., 2018; Wang and Ren, 2018; Bražinskas et al., 2020b). Unfortunately, summaries generated by unsupervised models are often generic and uninformative, and do not provide useful information about different aspects of the product.

To make review summaries more useful to users, we propose a class-specific unsupervised abstractive summarization model, which can generate class-specific summaries from multiple reviews of each product, according to any predefined set of topical classes of interest for the users. The model can be trained without using any ground-truth summaries. The only additional signal required for the model is the class probabilities or class label for each review generated by an independent black-box classifier, or provided by annotators. An example set of such topical classes is the set of major issue classes that products might be subjected to, including misleading product description, poor quality, sizing/fit/style issues, etc. See Table 1 for example class-specific summaries generated from reviews of a product according those topical classes.

To generate class-specific summaries for arbi-

* Product Assurance, Risk, and Security https://www.amazon.jobs/en/teams/product-assurance-risk-security

Proceedings of the 4th Workshop on e-Commerce and NLP (ECNLP 4), pages 88–100

trary classes, existing opinion summarization models require either i) a large training set of ground-truth summaries per class, or ii) a complicated and costly training set comprising multiple types of manual annotations such as token tags, aspect-opinion phrase pairs, and phrase labels (Suhara et al., 2020). In contrast, our model only needs review-class label pairs.

The proposed model Class-CopyCat combines a generative variational autoencoder (VAE) model with a hierarchical structure that captures dependence among products, reviews and classes, allowing the representation of class-specific information through class latent variables. We also propose an integrated two-layer filter mechanism consisting of a class-correlation gate and a set of class-specific importance coefficients which focus on class-related words, and thus reduce irrelevant or generic information and increase informativeness with respect to (w.r.t.) each class of interest.

Our contributions can be summarized as follows:

- We solve a new, practical problem that has not been addressed before: to train a model to generate class-specific summaries from multiple reviews of each product using only class probabilities or class label for each review.
- We propose a novel hierarchical latent variable generative model to capture dependence among group/class/review latent variables and reviews. This allows us to generate class-specific summaries from the variational distributions of respective class latent variables.
- We propose a two-layer filter mechanism to extract class-specific information and key words, and reduce irrelevant information in the summaries, as detailed in Sec. 3.2.3.
- Our human evaluation and experiments with a reference dataset show that the proposed model outperforms state-of-the-art baselines in a wide range of evaluation metrics.

2 Prior Work

2.1 Abstractive Summarization

Prior to deep-learning language models, abstractive summarization is considered a very hard problem, with limited success using graph mining (Ganesan et al., 2010; Filippova, 2010; Yang and Fang). More recent approaches view abstractive summarization as a text-to-text generation problem using sequence to sequence (Seq2Seq) neural models (Sutskever et al., 2014). These models usually employ an encoder-decoder structure. The encoder encodes documents into feature space, from which the decoder generates summaries. Such models tend to be "over creative" and may generate completely new outputs, which is not desirable. The prominent strategy to mitigate this problem is to use pointer networks as used in (Nallapati et al., 2016; Gu et al., 2016). Pointer networks (Vinyals et al., 2015) are an extension of attentive recurrent neural networks (RNN); they use attention as a pointer to select which tokens of the input sequence should be copied to the output. More recently, pointer-generator networks (See et al., 2017) add a switching mechanism to select between copying and generating new words. These supervised deep models require a large amount of text and human-written summary pairs for training (Hermann et al., 2015; Sandhaus, 2008; Narayan et al., 2018). Recent works on unsupervised abstractive summarization include SummaryLoop (Laban et al., 2020) for single document summarization, and MeanSum (Chu and Liu, 2019) and CopyCat (Bražinskas et al., 2020b) for multi-document summarization.

Summ. for chosen classes of interest	**Class 1. Misleading Product Description:** These tights are not pink. The color is very much nude. **Class 2. Poor Quality:** The tights ripped after one wash. I would not buy these. **Class 3. Sizing/fit/style issue:** The sizing was way off.
Reviews	... the "pink" color these come in is not the pink ... ‖ It lasted through one wear and one wash. After that the threads started streaking. Not worth the buy... ‖ ... The sizing for these tights was not clear, ... way too big. ‖ ... These tights ripped the first time my daughter wore them. Take a pass on these. ‖ ... these tights are not pink... ‖ ... They all have holes after first time wear... ‖ The color says pink, but these are not pink. They are nude... ‖ These tights are incredibly small compared to other brands and the pink color is more in line with nude...

Table 1: Summaries generated by our model for chosen classes; colors encode their alignment to input reviews. The reviews are truncated, and delimited with '‖'.

2.2 Context-aware document summarization

Our class-specific summarization problem is also related to context-aware summarization. In (Ma et al., 2018) and (Wang and Ren, 2018), the sentiment class of a product review is used as contextual information for summarization. In (Khatri et al., 2018), a contextual text summarization model based on Seq2Seq architecture is proposed for product description summarization. It includes

three different components as contextual information: metadata provided by sellers (e.g. product title, tags and category), search query, and document titles used to discover the document (e.g. via recommendation). In (Narayan et al., 2018) and (Wang et al., 2018), document topics are pre-learned by a Latent Dirichlet Allocation topic model, and used as contextual information for text summarization. These context-aware models are useful for generating more document centric summaries, overcoming the problem of generic summaries. However, these models only target single-document summarization, and they must be trained via supervised-learning using a large set of human-written ground-truth summaries.

OpinionDigest is an opinion summarization framework that does not rely on gold summaries for training (Suhara et al., 2020). However, to generate summaries specific to an arbitrary topic or class, OpinionDigest has to use an Opinion Extraction model (Miao et al., 2020) pretrained for that topic using a training set produced with significant human annotation effort for each review, including token tagging, aspect-opinion phrase pairing, and labelling selected phrases per topic. The high cost of obtaining such training sets motivated our proposed solution.

3 Proposed Abstractive Class-Specific Multi-Review Summarization

We start with a high level description of the proposed model (Sec. 3.1), before presenting in greater detail the encoder and decoder subnets (Sec. 3.2, 3.3). Later, we introduce the loyalty term that discourages summaries containing false information (Sec. 3.4), and describe how the trained model generates class-specific summaries (Sec. 3.5).

3.1 Overview of the proposed model

Given a predefined set of T topical classes, our model generates multiple summaries, one for each class that may be present in a group of reviews; each group corresponds to a product. The model makes use of an independent classifier $\beta(.)$, which probabilistically assign each review to these classes; $\beta(r_i)_j$ denotes the probability that review r_i belongs to a class j for $j = 1, ..., T$. We propose a hierarchical latent variable structure to capture relation among products, reviews and classes as shown in Figure 1.

The model defines three sets of latent variables to represent products, classes, and individual reviews. Each product (or group) is associated with a group variable g, which captures the group's overall semantics. Within each product, each class j is associated with a class variable t_j ($j = 1, ..., T$). Each class latent variable conditions on g, but focuses more on class-related words and information; and hence, it captures common themes and opinions about the product for that class. Finally, each review r_i ($i = 1, ..., N$) is associated with a review latent code $\mathbf{z}_i = [z_{ij}]_{j=1}^{T}$, which conditions on the class representation and captures content of individual reviews; z_{ij} denotes the review code for r_i given a class j. The class distribution $\beta(r_i)$ is used to soft-gate $\mathbf{z}_i$ in review reconstruction.

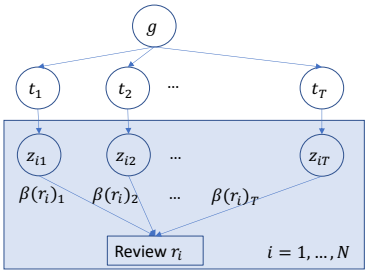

Figure 1: Graphical representation of the model

Our model's posterior inference is based on the VAE model (Kingma and Welling, 2013), also used in the CopyCat summarization model (Bražinskas et al., 2020b), with the latter serving as the inspiration for our model. As is standard with VAEs, our encoder, parameterized with ϕ, produces the variational posterior distributions of the latent variables $g \sim q_\phi(g|r_{1:N})$, $t_j \sim q_\phi(t_j|r_{1:N}, [\beta(r_i)]_{i=1}^{N}, g)$, and $z_{ij} \sim q_\phi(z_{ij}|r_i, t_j)$. As shown in Sec. 3.2, the variational posterior of t_j is designed to depend on class distributions of all reviews in the group $[\beta(r_i)]_{i=1}^{N}$, and that class-related information represented by t_j is computed using a two-layer filter mechanism. We also note that we choose to represent each review r_i using a collection of review variables $[z_{ij}]_{j=1}^{T}$, each corresponding to a class, instead of using a single review encoding as in CopyCat and typical VAE models, for better representation of class-related information. Encoder design is discussed in details in Sec. 3.2.

The decoder, parameterized by θ, reconstructs the review r_i from the posterior samples $\mathbf{z}_i$, $\beta(r_i)$ and all other reviews in the group r_{-i}. The reconstruction probability is hence defined as $p_\theta(r_i|\beta(r_i), \mathbf{z}_i, r_{-i})$. Here, we follow CopyCat's recommendation and let the decoder to directly access other reviews in the group to allows the reconstruction of fine-grain common group details, such

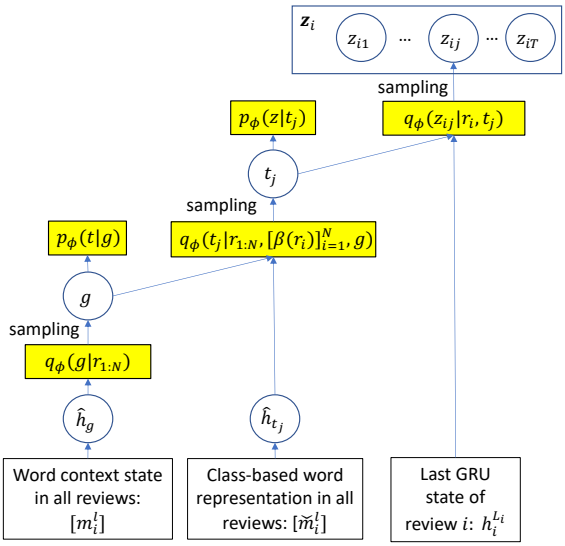

Figure 2: Generation of the latent code $\mathbf{z}_i$ for a review r_i by the encoder. Yellow boxes represent the neural networks that compute the prior and variational posterior distributions of latent codes.

as product names, product-specific attributes and characteristics. As we detail in Sec. 3.5, the trained model can generate a summary for a group of reviews for a given class j by decoding the mean of the class-dependent and review-agnostic z_{*j} prior.

The variational loss objective for our model is

$$\mathcal{L}_{VAE}(\theta, \phi, r_{1:N}) = \mathop{\mathbb{E}}_{g\sim q_\phi(g|r_{1:N})}\Bigg[\mathop{\mathbb{E}}_{\mathbf{t}\sim q_\phi(\mathbf{t}|r_{1:N},\,[\beta(r_i)]_{i=1}^N,\,g)}\Bigg(\sum_{i=1}^{N}\mathop{\mathbb{E}}_{\mathbf{z}_i\sim q_\phi(\mathbf{z}_i|r_i,\mathbf{t})}[-\log p_\theta(r_i|\mathbf{z}_i,\beta(r_i),r_{-i})] + \sum_{i=1}^{N} D_{KL}[q_\phi(\mathbf{z}_i|r_i,\mathbf{t})||p_\phi(\mathbf{z}_i|\mathbf{t})]\Bigg) + D_{KL}[q_\phi(\mathbf{t}|r_{1:N},[\beta(r_i)]_{i=1}^N,g)||p_\phi(\mathbf{t}|g)]\Bigg] + D_{KL}[q_\phi(g|r_{1:N})||p_\phi(g)], \quad (1)$$

where $p_\phi(.)$ denotes a prior distribution, and D_{KL} denotes Kullback Leibler divergence between the variational posterior and prior distributions of a latent variable. Later, in Sec. 3.4, we will improve this loss with an additional loyalty term.

3.2 The Encoder

Fig. 2 shows how the encoder produces latent codes g, $\mathbf{t}$, and $\mathbf{z}_i$. As with standard VAE, we use Gaussian distributions with diagonal covariances for the prior and variational distributions.

3.2.1 Text representation component

The encoder starts with a text representation component which includes a word embedding unit, a GRU encoder (Cho et al., 2014), and a class-correlation gate, as shown in Fig. 3.

Words in reviews are embedded into word embeddings, and then transformed by a GRU encoder to obtain hidden states. Let L_i denote the length of review r_i; w_i^l and h_i^l denote the word embedding and GRU hidden state for the l-th word in review r_i, for $l = 1, ..., L_i$. Word embeddings and GRU hidden states are concatenated into word context states: $m_i^l = [w_i^l \circ h_i^l]$. They are later used to compute group latent codes g (Sec. 3.2.2).

Word context states are also fed to a class-correlation gate to generate class-based word representation, which pays more attention to words related to the class of the review. First, the concatenation of each word context state and the class vector $\beta(r_i)$ of the review is fed to a feed-forward neural network (FFNN) with $\tanh$ non-linearity to give a class influence vector for each word:

$$c_i^l = \tanh(W[m_i^l \circ \beta(r_i)] + b),$$

where c_i^l has the same dimension as m_i. The class-based word representation is then computed as

$$\breve{m}_i^l = m_i^l \odot c_i^l,$$

where $\odot$ is the element-wise multiplication operation. The class-based word representations later contribute to the class latent codes (Sec. 3.2.3).

3.2.2 Distributions for group latent codes g

The group latent code in our model plays a similar role to that in Copycat model, and its distributions are computed in a similar way. Its prior $p(g)$ is set to the standard normal distribution. To compute the variational posterior $q_\phi(g|r_{1:N})$, we first compute the importance coefficient of each word in the review group, which is

$$\alpha_i^l = \frac{\exp(f_\phi^\alpha(m_i^l))}{\sum_{i'=1}^{N}\sum_{l'=1}^{L_{i'}}\exp(f_\phi^\alpha(m_{i'}^{l'}))}, \quad (2)$$

for the l-th word in review r_i. Here, f_ϕ^α is a 2-layer FFNN with $\tanh$ non-linearity, which takes as input the word context states and returns a scalar.

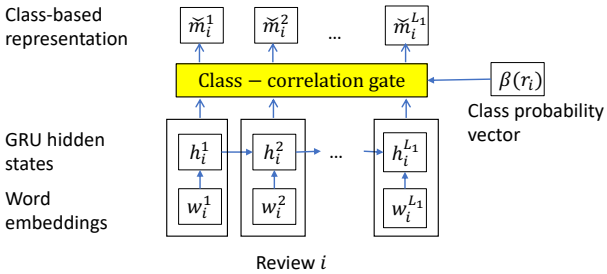

Figure 3: Generation of context states and class-based representations by text representation component.

The mean and log covariance of $q_\phi(g|r_{1:N})$ are then computed by separate affine projections of the intermediate group representation $\hat{h}_g$, which is the weighted sum of the word context states: $\hat{h}_g = \sum_{i=1}^{N}\sum_{l=1}^{L_i} \alpha_i^l m_i^l$.

We can then sample a latent code g from the above posterior distribution. The reparameterization trick (Kingma and Welling, 2013) is applied during sampling to allow backpropagation of the reconstruction error.

3.2.3 Distributions for class latent codes t

The prior for class latent codes is conditioned on the common group latent code g and shared across different classes, i.e., $p_\phi(t_j|g) = \mathcal{N}(t_j; \mu_\phi^t(g), \sigma_\phi^t(g)I)$ for $j = 1, ..., T$, where the mean and log covariance are computed as a linear transformation of g.

The variational posterior for latent code t_j of each class j depends on the common group code g, reviews $r_{1:N}$, and class probabilities $[\beta(r_i)]_{i=1}^N$; it is $q_\phi(t_j|r_{1:N}, [\beta(r_i)]_{i=1}^N, g)$. To compute this posterior of t_j, we also first compute the importance coefficient of each word in each review to the class representation t_j, but now we use class-based word representations instead of word context states as in g:

$$\check{\alpha}_{ji}^l = \frac{\exp(f_\phi^{\check{\alpha}_j}(\check{m}_i^l))}{\sum_{i'=1}^{N}\sum_{l'=1}^{L_{i'}} \exp(f_\phi^{\check{\alpha}_j}(\check{m}_{i'}^{l'}))}. \quad (3)$$

We then compute the intermediate class representation $\hat{h}_{t_j} = \sum_{i=1}^{N}\sum_{l=1}^{L_i} \check{\alpha}_{ji}^l \check{m}_i^l$. The computation of $\hat{h}_{t_j}$ can be viewed as a two-layer filter mechanism to extract class-specific information and key words, and reduce irrelevant and generic information to be represented in class latent codes. First, the class-correlation gate pays attention to key words related to the class of each review. Then, among those key words from different reviews, the importance coefficients $\check{\alpha}_{ji}^l$ pay attention to those related to the class j of interest.

Finally, we apply affine transformations on the concatenation of $\hat{h}_{t_j}$ and g to give the mean and log variance of the variational posterior for t_j. We can sample a latent code t_j from this posterior and generate an assembled code $\mathbf{t} = [t_j]_{j=1}^T$.

3.2.4 Distributions for review latent codes $\mathbf{z}_i$

The prior on the review code z_{ij} corresponding to review r_i and class j is conditioned on the class code t_j and is shared across different reviews, i.e. $p_\phi(z_{ij}|t_j) = \mathcal{N}(z_{ij}; \mu_\phi^z(t_j), \sigma_\phi^z(t_j)I)$ for $j = 1, ..., T$, where the mean and log covariance are computed as a linear transformation of t_j.

To compute the mean and log covariance of variational posterior $q_\phi(z_{ij}|r_i, t_j)$, we perform affine transformation on the concatenation of $h_i^{L_i}$ and t_j. $\mathbf{z}_i = [z_{ij}]_{j=1}^T$ is then sampled from these posteriors.

3.3 The Decoder

The decoder reconstructs the original reviews by computing the distribution $p_\theta(r_i|\mathbf{z}_i, \beta(r_i), r_{-i})$. First, the aggregated latent code $\hat{z}_i$ for each review r_i can be computed as: $\hat{z}_i = \sum_{j=1}^{N} \beta(r_i)_j z_{ij}$. After that, we follow the structure of CopyCat's decoder (Bražinskas et al., 2020b). The decoder takes $\hat{z}_i$ and r_{-i} as input and computes $p_\theta(r_i|\hat{z}_i, r_{-i})$. We use an auto-regressive GRU decoder with the attention mechanism and a pointer generator network.

3.4 Loyalty term

The VAE lower bound in Eq. (1) focuses on reconstructing a review r_i from its latent representation and other reviews r_{-i} of the same group. Because reviews may vary largely, and it is not always possible to reconstruct a review from other reviews, the decoder tends to be creative, and inclines toward generating a new word, instead of copying a word from other reviews. As a result, the generated summaries at test phase often contain many new words and possibly false information that is not present in original reviews. To remedy this problem, inspired by (Bražinskas et al., 2020a), we add a loyalty term $\mathcal{L}_0$ that encourages assigning the probability mass to words that appear in r_{-i}:

$$\mathcal{L}_0(\theta, \phi, r_{1:N}) = \mathop{\mathbb{E}}_{g\sim q_\phi(g|r_{1:N})} \left[\mathop{\mathbb{E}}_{\mathbf{t}\sim q_\phi(\mathbf{t}|r_{1:N},[\beta(r_i)]_{i=1}^N,g)} \left(\sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{z}_i\sim q_\phi(\mathbf{z}_i|r_i,\mathbf{t})} \left[\sum_{l=1}^{L_i} \sum_{w\in V(r_{-i})} -p_\theta(w|\mathbf{z}_i, \beta(r_i), r_{-i}, r_i^{1:l-1}) \right] \right) \right],$$

where $V(r_{-i})$ is the vocabulary of all words in r_{-i}. The final loss function is computed as

$$\mathcal{L} = \mathcal{L}_{VAE} + \alpha * \mathcal{L}_0, \quad (4)$$

where α is the trade-off hyperparameter. $\mathcal{L}$ is minimized w.r.t. both the inference network's parameter ϕ and the generative network's parameter θ.

3.5 Summary Generation

At test time, we can generate a summary per class for a new group of reviews $r_{1:N}$. This is equivalent to generating a new review that reflects common information from the reviews $r_{1:N}$. To do so, the latent variables are fixed to their respective means. The steps to generate a summary r_* for a class j from a group of reviews $r_{1:N}$ are as follows:

1. Fix g at the mean of its posterior $q_\phi(g|r_{1:N})$.
2. Fix t_j at the mean of its posterior $q_\phi(t_j|r_{1:N}, [\beta(r_i)]_{i=1}^N, g)$.
3. Fix z_{*j} at the mean of its prior $p_\phi(z_{*j}|t_j)$.
4. Assign $\hat{z}_* = z_{*j}$, and compute the decoder's probability for r_*: $p_\theta(r_*|\hat{z}_*, r_{1:N})$.

4 Experimental Results

4.1 Experimental setup

Our experiment is conducted on a subset of the public dataset of Amazon product reviews (He and McAuley, 2016). In this dataset, each review is written for a particular product, and accompanied by a rating value between 1 and 5. We apply our model to generate summaries for a group of average and low-rating reviews (1 to 3-star rating reviews) belonging to a product according to the classes of issues behind the poor ratings. Following (Bražinskas et al., 2020b), we use reviews from four product categories: Clothing Shoes and Jewelry, Electronics, Health and PersonalCare, and Home and Kitchen. We obtain reviews with 1 to 3-star ratings, and group them by products with each group having no less than 8 reviews. The dataset consists of 773,797 reviews for 54,706 products. From this, we sampled 1000 products for test, and split the remaining products into training/validation sets with a $9:1$ ratio. See Appendix A.3 for more details in data pre-processing, and Appendix A.4 for hyperparameter settings and implementation details of our model.

We created an independent classifier $\beta(.)$ that classifies reviews into 6 classes of possible issues: POOR QUALITY OR DEFECTIVE, SIZING/FIT/STYLE ISSUE, BAD/MISLEADING PRODUCT DESCRIPTION, COMPATIBILITY ISSUE, WRONG ITEM RECEIVED, and OTHERS. Definition for each class is given in Appendix A.2. The classifier produces a class probability for each issue class per review. Using the class probability output of this classifier $\beta(.)$ as input, we train our Class-CopyCat model to generate class-specific summaries for these 6 issue classes from product reviews. At test time, we compute an aggregated probability for each issue class per product by averaging the classifier probability outputs across all reviews for that product, and only generate summaries for classes whose aggregated class probabilities for the product is greater than a threshold of 0.1. This threshold is adjustable. This is meant to filter out those issues that are not supported by the reviews, assuming that there are much more pronounced issues that we want to summarize. Reading class-specific summaries for top issue classes enable users to quickly understand different predominant issues for a product. We note that our model can be applied for any set of topical classes and classifiers. The choice to use the 6 product issue classes is arbitrary, and we could easily choose some other classification (e.g. sentiment, rating, author).

For evaluation, we sampled 100 products and 8 reviews per product from the test set. We obtained gold summaries for these products from 2 external workers. We asked each worker to write a set of gold summaries per product, one for each issue class of the product, provided that the aggregated class probability for the product is greater than 0.1.

Table 2 shows an example of average and low-rating reviews for a product, the gold summaries, and summaries produced by different models. Additional examples are given in Appendix A.1.

4.2 Baseline Models

We prepare 7 baseline models for class-specific summarization.

Abstractive baselines

CopyCat with Class-embedding. We combine CopyCat model for multi-document review summarization (Bražinskas et al., 2020b) with the class-embedding mechanism proposed in (Narayan et al., 2018) for class-specific summarization. Particularly, the class probability vector $\beta(r_i)$ for a review r_i is appended to each word embedding of that review at both encoder and decoder during training. When generating a class-specific summary, the class probability vector at decoder becomes a one-hot encoding of that class.

Collection of CopyCat models. We train one CopyCat model for each issue class. Each class-specific Copycat model is trained only on reviews that are classified into that class (the top class). Class-specific summaries are generated by the corresponding class-specific model.

Collection of MeanSum models. MeanSum is another state-of-the-art multi-review summarization method (Chu and Liu, 2019). This baseline is similar to the second baseline above.

Extractive baselines

Highest probability. The review with highest class probability is used as class-specific summary.

Clustroid. The clustroid review among the set of reviews belonging to a class is used as the class-specific summary. It is the review with the highest

Top issue classes	POOR QUALITY OR DEFECTIVE	COMPATIBILITY ISSUE
Gold summary for each issue	The mount does not hold weight when closed. It doesn't work properly. It bends down.	The mount is not suitable for big TV screens. It does not fit a 42" TV. It should be for 32" or less.
Our class-specific summaries	I am very disappointed in this product. It is hard to get it to hold the weight of the TV, and it will not work.	I bought this for my 42 inch TV. It does not fit the TV. It is too small.
Collection of CopyCat models' summaries	I bought this to use with my Samsung TV. It did not work at all. I tried to adjust the TV but it didn't work. I would not recommend this product to anyone. Very disappointed.	The title of this says it would fit a 49 inch TV, but my TV will not fit. The mount didn't work either. Would not recommend this product for the specific model of TV.
Collection of MeanSum models' summaries	Bought this for my Samsung TV and it did not fit my LG TV. I would not recommend this product to anyone. Don't waste your time and money on this piece of junk. Do not buy!	The end of this mount doesn't work with my Samsung TV. The mount is too wide and there is no space for my TV, which defeats the purpose of being able to mount it in my TV!
Review 1	I would not use this mount for any big tvs . I had a hard time trying to make this work and junked it in the end.	
Review 2	JUNK!!! I have a 65 " Toshiba 91 # bigger than stated but weight is weight. Sags, will not hold weight when closed. Also very hard the hook the tv mount to the wall mount. I am afraid to even pull it out to full length. I am taking it down tomorrow.	
Review 3	don 't work well with heavy lcd tvs. installation was easy, find two stud screw it in. mount my samsung 52 " lcd and just points down, tilt all the way up and as soon as i let go off my hand it just tilt down just cant have the tv parallel to the wall. you get what you pay for.	
Review 4	This mount will not hold a 50 LCD TV. As soon as we placed the tv on the mount it sagged and bent a little. When we tried to angle it it tilted a lot. If you have a 40 or under TV you will be fine.	
Review 5	size of the bracket is too small trying to mount a 55 inch - please reconsider buying this product this product only safe with 15 - 32 inch only not recommended for 55inch tv.	
Review 6	I bought this item for a 42 " TV. It does not fit the TV ! Partially my own fault for not doing a bit more research but it is way to small to holder an older 42 " TV. I used it on a smaller 32 " TV which saved me some trouble but it is a bit of overkill for a smaller TV.	
Review 7	12 inches too narrow for my 37 inch LG TV. This mount should only be used on a 32 inch or less.	
Review 8	The product hung on the wall crooked. Cheap. Don 't buy. Had to return. Waste of time. I guess you get what you pay for.	

Table 2: Examples of reviews, gold summaries, and summaries generated by various models. Generic or redundant information not related to the class is marked in Blue; hallucinating or incorrect information is marked in Red.

ROUGE-L score w.r.t. other reviews in the set.

Lead. We construct the class-specific summary by concatenating leading sentences from reviews belonging to that class. "Lead" baseline has been shown to be a strong baseline for both single- and multi-document summarization (See et al., 2017; Bražinskas et al., 2020b; Chu and Liu, 2019).

Random. A random review belonging to a class is used as the class-specific summary.

See Appendix A.5 for training procedure of our model and those of the abstractive baselines.

4.3 Automatic Evaluation

We measure semantic overlap between generated and gold summaries using ROUGE scores (Lin, 2004). A higher ROUGE score associates with more semantic overlap between pairs of texts. The class-specific gold summaries are used as reference. Table 3 reports ROUGE scores based on F1 on the 100 test products.

Class-Copycat outperforms all the baselines. The summaries of extractive baselines ("Highest probability" , "Clustroid", "Lead" and "Random") are extracted from the subset of reviews that belong to the class of interest only, and therefore, contain a good amount of class-specific information. This results in relatively higher ROUGE scores for these baselines compared to abstractive ones, except for Class-CopyCat. The low ROUGE scores for "CopyCat with Class-embedding" indicate that class-embedding mechanism does not sufficiently extract class-specific information from reviews. Similarly, limiting the training set to only reviews belonging to the class of interest as in "Collection of CopyCat models" and "Collection of MeanSum models" does not allow the models to sufficiently focus on class-related information and omit generic and irrelevant information from reviews, as shown in Table 2 and in Appendix A.1. This results in low ROUGE scores for these two methods, and is confirmed with human evaluation (Sec. 4.4).

Model	R1	R2	RL
Class-CopyCat (ours)	**0.308**	**0.102**	**0.238**
CopyCat w Class-embed	0.186	0.037	0.133
Collection of CopyCat(s)	0.190	0.038	0.141
Collection of MeanSum(s)	0.182	0.034	0.138
Highest probability	0.254	0.092	0.207
Clustroid	0.242	0.085	0.204
Lead	0.258	0.098	0.210
Random	0.213	0.061	0.171

Table 3: ROUGE scores on the 100 test products.

4.4 Human evaluation

We performed human evaluation on 50 products from the test set. Two annotators are asked to rate each summary on a scale of 1 (very poor) to 5 (very good) based on 6 criteria as follows:

Model	Informativeness	Conciseness & non-redundancy	Content support	Opinion consensus	Fluency	Overall
Collection of CopyCat models	2.87	2.69	2.91	3.09	3.83	2.80
Collection of MeanSum models	2.80	2.55	2.63	2.70	3.80	2.54
Class-CopyCat	**3.60**	**4.30**	**3.98**	**3.72**	**4.57**	**3.84**

Table 4: Average scores of human evaluation for six criteria. Score ranges from 1 (very poor) to 5 (very good).

- Informativeness: how well summary covers dominant and repeated issues in the class.
- Content support: how well the content of summaries is supported by input reviews
- Conciseness and non-redundancy: The summary should be concise and not contain unrelated information or unnecessary repetition.
- Opinion consensus: the summary should reflect common opinions expressed in reviews.
- Fluency: summary sentences should be grammatically correct, and easy to understand.
- Overall: based on annotators' judgment.

Class-CopyCat outperforms both reference models by a large margin across all criteria (Table 4). The biggest gain of Class-CopyCat over the two baselines is in conciseness and non-redundancy (4.30 vs 2.55 and 2.69). Examples in Table 2 and Appendix A.1 show qualitatively that our summaries are more concise compared to baselines. The latter contains more generic and irrelevant information (highlighted in blue in the examples).

Class-CopyCat also outperforms the two baselines in terms of content support. Both baselines produce much information not present in original reviews; such information is highlighted in red in Table 2 and Appendix A.1. Our model performs well in this criteria due to the loyalty term introduced in Sec. 3.4. Moreover, since our model often generates more concise and non-redundant summaries from salient class-related information, it has less chance of introducing incorrect information.

As our model focuses on and includes more class-related information, it also does better in informativeness (3.60 vs 2.80 and 2.87). Because class-related key information is often salient and consistent, this also results in better opinion consensus.

4.5 Model Variant Ablation Studies

Here, we compare Class-CopyCat with its variants. The result is shown in Table 5. In "*without class-correlation gate*" variant, we omit the class-correlation gate in Fig. 3, and use word context states directly (in place of class-based representation in Fig. 2) to compute the posterior of the class latent code **t**. In "*class-embedding*" variant, instead of using class-correlation gate, we append the class probability vector $\beta(r_i)$ to each word context state of the review r_i to generate class-based word representations as in (Narayan et al., 2018). In "*without loyalty term*" variant, the loyalty term (Sec. 3.4) is not added to loss function. Finally, in "*without* g" variant, we remove the group latent code g, as we question whether g is still needed, in the presence of the class code t per class per group.

The result in Table 5 shows that each model component indeed contributes to final performance. *Without class-correlation gate*, performance drops most significantly. Using *class-embedding* instead improves ROUGE scores compared to not using any class-based representation, but its performance is still far from using class-correlation gate in the final Class-CopyCat. *Without loyalty term*, the model generates more 'hallucinating' words (eg. product names, models), resulting in lower ROUGE scores. Finally, "*without* g", ROUGE scores reduce slightly. This is because the class latent code t is designed to focus more on keywords for each class and without conditioning on g, it cannot represent differences among various products (eg. pants vs shirt).

Model	R1	R2	RL
Class-CopyCat	**0.312**	**0.107**	**0.216**
Without class-correlation gate	0.272	0.092	0.209
Class-embedding	0.287	0.098	0.212
Without loyalty term	0.282	0.096	**0.216**
Without g	0.304	0.103	0.208
Random	0.213	0.061	0.171

Table 5: Ablation results: ROUGE scores for different model variants using the gold summary dataset.

5 Conclusion

We have proposed a model for generating class-specific summaries from a collection of reviews. Our evaluation results show that our model outperforms many abstractive and extractive baselines, including state-of-the-art models, in term of ROUGE scores that measure the semantic overlap between generated and gold summaries. We also show through human evaluation that generated summaries of our model are highly relevant to the classes of interest, fluent, and representative of common opinion.

References

Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020a. Few-shot learning for opinion summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4119–4135.

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.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734.

Eric Chu and Peter Liu. 2019. Meansum: a neural model for unsupervised multi-document abstractive summarization. In *International Conference on Machine Learning*, pages 1223–1232.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd international conference on computational linguistics (Coling 2010)*, pages 322–330.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 240–250.

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*, pages 340–348.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

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*, pages 507–517.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Masaru Isonuma, Toru Fujino, Junichiro Mori, Yutaka Matsuo, and Ichiro Sakata. 2017. Extractive summarization using multi-task learning with document classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2101–2110.

Chandra Khatri, Gyanit Singh, and Nish Parikh. 2018. Abstractive and extractive text summarization using document context vector and recurrent neural networks. *arXiv preprint arXiv:1807.08000*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Philippe Laban, Andrew Hsi, John Canny, and Marti A Hearst. 2020. The summary loop: Learning to write abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, volume 1.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shuming Ma, Xu Sun, Junyang Lin, and Xuancheng Ren. 2018. A hierarchical end-to-end model for jointly improving text summarization and sentiment classification. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4251–4257.

Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. 2020. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*, pages 617–628.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

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*, pages 5789–5798.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.

Hongli Wang and Jiangtao Ren. 2018. A self-attentive hierarchical model for jointly improving text summarization and sentiment classification. In *Asian Conference on Machine Learning*, pages 630–645.

Li Wang, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu, and Qiang Du. 2018. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4453–4460.

Peilin Yang and Hui Fang. An opinion-aware approach to contextual suggestion.

A Appendices

A.1 Additional examples of reviews and summaries

Table 6 and Table 7 show additional examples of reviews for a product, the gold summaries, and the class-specific summaries produced by different models. Additionally, we also show the class-agnostic summaries produced by 3 opinion summarization models: CopyCat, MeanSum, and Lexrank (Erkan and Radev, 2004). Lexrank is an unsupervised extractive algorithm which selects sentences based on graph centrality. Sentences represent nodes in a graph whose edge weights denote tf-idf similarity.

Top issue classes	POOR QUALITY OR DEFECTIVE	MISLEADING PRODUCT DESCRIPTION
Gold summary for each issue	The stand and the screen angle are not adjustable. The monitor is too bright and the picture looks washed out.	The product did not have HDMI hook as it says on the description.
Our class-specific summaries	I was very disappointed in this monitor. It is not worth the price. The picture is very bright and the stand is not adjustable.	Bad description of the product. It didn't have a HDMI hookup.
Collection of CopyCat models' summaries	I bought this monitor for my husband for Christmas. It was a Christmas gift for Christmas and it was already broken. I would not recommend this monitor to anyone. Very disappointed in the quality of it.	It's not compatible with my monitor, even though it is advertised as a Samsung 46 inch memory. Very disappointed! Don't waste your time and money on this item. Don't buy this.
Collection of MeanSum models' summaries	The monitor is not bright enough to adjust the monitor. I had to send it back because it was a waste of my time and money. Don't waste your money on this one. Do not buy!	What a waste of money, but when I received it in the mail it looked much better. You can't see through the screen. It's not worth the trouble of returning it and returning.
CopyCat's summary	It's a nice monitor, but I had to return it for a refund. Also, the monitor doesn't work with the monitor. I would not recommend this product to anyone. Save your money	
MeanSum's summary	It's a nice looking monitor, but it is not what I expected. I have tried everything to get it to work, but the screen resolution is terrible. I would not recommend this product.	
Lexrank's summary (Erkan and Radev, 2004)	It is the worst monitor i have ever purchased, the stand is not adjustable and their is something wrong with the base. I was unable to adjust the viewing angle and therefore returned the monitor.	
Review 1	Like someone who reviewed this before me... this monitor is too bright. It looks like there is a bright white haze on everything and i do not like it. I just received this monitor and i will be returning it as soon as possible. Very disappointed.	
Review 2	Disappointment. The monitor does not allow you to adjust the screen angle without weakening the stability of the monitor. I sent it back. Amazon did refund full purchase price with no hassle.	
Review 3	Had to return this one. the plugin in the back didn 't work. Pain to get it and it not work, especially since it was a gift.	
Review 4	Picture is dusty looking! the stand is not adjustable. Not worth the low price. You don 't even get what you pay for!	
Review 5	The color is terrible, the contrast does not adjust, unless you look it from one very specific unrealistic angle the picture is washed out	
Review 6	Would be nice to have known it didn 't have hdmi hookup before is wasted money. Worst description of a product features	
Review 7	It is the worst monitor i have ever purchased, the stand is not adjustable and their is something wrong with the base.	
Review 8	I was unable to adjust the viewing angle and therefore returned the monitor. There was nothing in the instructions about this.	

Table 6: Example reviews for a product, the corresponding gold summaries written by human, and the summaries generated by different models. For class-specific summaries, generic or redundant information not related to the class is marked in Blue; hallucinating or incorrect information is marked in Red.

Top issue classes	SIZING/FIT/STYLE ISSUE	POOR QUALITY OR DEFECTIVE
Gold summary for each issue	The size is not true. The pants are too small and super skinny.	The material is shoddy, the stitch quality is terrible. They are not Levi's.
Our class-specific summaries	These pants are too small for me. They are too tight.	These pants are not what I expected. Their quality is very poor. The material is very cheap and the stitch quality is really poor.
Collection of CopyCat models' summaries	These are not true to size, I ordered a large and they are still too tight. I will not be buying these again. I'm a size 6 and these pants are not flattering at all.	These are not the same quality Levi's pants I have ever seen. They are made out of thin material and tear apart easily. I would not recommend these to anyone, especially for the price.
Collection of MeanSum models' summaries	It's a shame because the pants are too small for me. I normally wear a medium, but these pants were way too big for me and I could barely zip them up. Very disappointed.	The stitching on the pants are so stiff that they are uncomfortable. I would not recommend these pants to anyone. I had to return them because they were a waste of my money. Don't buy them.
CopyCat's summary	These are not the relaxed fit for me. I ordered a 34, and they were way too big for my legs. I would not recommend these pants to anyone else in the future.	
MeanSum's summary	These are not true to size. I ordered a large and they were too tight around the waist. I will have to return them. They are not worth the hassle of returning them. I am returning them.	
Lexrank's summary (Erkan and Radev, 2004)	These pants are not real Levi's. Super skinny jeans , more like chick pants guys shouldn't be wearing these, more like a joke cant believe i bought these pants	
Review 1	I bought "Levi 's Men's 513 Slim Straight Jean, Mr Blue, 28Wx32L". Size is good fit. But this item is too small. Same brand, same size is not same.	
Review 2	The jeans missing a loop what the ass ah hold loop in the back poor really poor need to inspect the items before	
Review 3	It is too big it is not the correct size I have another Levi s size 40 and its perfect.	
Review 4	Super skinny jeans, more like chick pants guys shouldn't be wearing these, more like a joke cant believe i bought these pants	
Review 5	I ordered up two sizes after reading that these run tight, but before putting them on I realized they aren't even real. The material is shoddy, the stitch quality is terrible, and the color doesn't resemble the photo online at all! Beware of these pants and hope that you don't get ripped off like I have.	
Review 6	you need legs like broomsticks for these guys to fit. Could barely slip them over my calves without splitting the seams, if you have muscular legs, these are not for you...	
Review 7	These pants aren't what I expected. The quality is really low and I'm pretty sure these pants won't last very long. It 's simply cheap material. The fit is good though.	
Review 8	These pants are not real Levi's. I was comparing the 511's I just bought in store to these ones and they are barely anything alike. The material is not the same. The only place it says Levi is on the button and the back patch. The stitching isn 't the same either.	

Table 7: Example reviews for a product, the corresponding gold summaries written by human, and the summaries generated by different models. For class-specific summaries, generic or redundant information not related to the class is marked in Blue; hallucinating or incorrect information is marked in Red.

A.2 Definition of issue classes

Issue class	Class definition
POOR QUALITY OR DEFECTIVE	Product is defective and does not perform its function, or contains a flaw that results in objectively poor performance and utilization of the product.
COMPATIBILITY ISSUE	Product is incompatible with another product that it is meant to be used with/for.
SIZING/FIT/STYLE ISSUE	Product is either too small/big to fit the customer's use case, without mentioning size incompatibility with another product.
BAD/MISLEADING PRODUCT DESCRIPTION	Product description contains misleading or insufficient information.
WRONG ITEM RECEIVED	Product shipped is different than the one the customer ordered.
OTHERS	Product issues that do not belong to any of the above classes.

Table 8: Definition of the issue classes.

A.3 Data Pre-Processing

We select only the reviews of which the star rating is between 1 to 3, and the length is between 20 to 70 words. These reviews are grouped by products, and only the products that have no less than 8 reviews satisfying the above conditions are selected. In addition, popular products with the number of reviews above the 90th percentile are removed, so that the dataset is not dominated by a small portion of products. During both training and test time, each group of reviews is formed from 8 reviews which are sampled without replacement from the set of reviews belonging to a same product.

A.4 Hyperparamters and Implementation Details

We use similar hyperparameter settings to those used in (Bražinskas et al., 2020b). The word embeddings are shared by both the encoder and decoder; their dimension is set to 200. The vocabulary size is 80000. Both the GRUs at encoder and decoder have the hidden state dimension of 600. The dimension of all the latent variables (g, t and z) is set to 600. Both the FFNNs that are used to compute the importance coefficients toward the posteriors of g and t in Eq. (2) and (3) have a 300-dimensional hidden layer. The decoder's attention network has a 200-dimensional hidden layer with a $\tanh$ non-linearity. The network for computing copy gate in the pointer-generator network also has a 100-dimensional hidden layer with the same non-linearity. The trade-off hyperparameter α in Eq. (4) is set to 2.

A.5 Initialization and Training

The CopyCat model (Sec. 4.6) and its variants (CopyCat with Class-embedding and the collection of class-specific CopyCat models, described in Sec. 4.2), which are used as baselines in our evaluation, are initialized with the CopyCat reference model provided by the authors of CopyCat (Bražinskas et al., 2020b). This reference model was previously trained on a larger dataset (183,103 products and 4,566,519 reviews) consisting of all the reviews with star rating from 1 to 5, unlike our training set which contains only the reviews with star rating from 1 to 3. We find that initializing the above baseline models (the CopyCat model and its variants) with this reference model gives a better performance for these baselines compared to training from scratch with our training set. For our Class-CopyCat model, the word embedding module and the two GRUs are also initialized with the corresponding components of the reference CopyCat model. Other 2D weights are initialized with Xavier uniform initialization (Glorot and Bengio, 2010), and 1D weights are initialized with the scaled normal noise with 0.1 standard deviation.

After initialization, we train each of the Class-CopyCat model, the CopyCat model, the CopyCat with Class-embedding and the collection of class-specific CopyCat models for 5 epoches on our training set of average and poor rating reviews (Sec. 4.1) using Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0001. For decoding a summary at test time, length-normalized beam search of size 5 is used. Cycling annealing (Fu et al., 2019) is applied for all the KL terms to mitigate the problem of "posterior collapse" (Bowman et al., 2016).

The MeanSum model and its variants are also pre-trained on the larger dataset of reviews with 1 to 5-star rating before being fine-tuned on our smaller training set of average and poor rating reviews.