Unsupervised Aspect-Based Multi-Document Abstractive Summarization

Maximin Coavoux,[2*] **Hady Elsahar,**[1] **Matthias Gallé**[1]
[1]Naver Labs Europe
[2]Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
{maximin.coavoux,hady.elsahar,matthias.galle}@naverlabs.com

Abstract

User-generated reviews of products or services provide valuable information to customers. However, it is often impossible to read each of the potentially thousands of reviews: it would therefore save valuable time to provide short summaries of their contents. We address opinion summarization, a multi-document summarization task, with an unsupervised abstractive summarization neural system. Our system is based on (i) a language model that is meant to encode reviews to a vector space, and to generate fluent sentences from the same vector space (ii) a clustering step that groups together reviews about the same aspects and allows the system to generate summary sentences focused on these aspects. Our experiments on the Oposum dataset empirically show the importance of the clustering step.

1 Introduction

Nobody reads all available user-generated comments about products they might buy. Summarizing reviews in a short paragraph would save valuable time, as well as provide better insights into the main opinions of previous buyers. In addition to traditional difficulties of summarization, the specific setting of opinion summarization faces the entanglement of multiple facets in reviews: polarity (including contradictory opinions), aspects, tone (descriptive, evaluative).

Obtaining large parallel corpora for opinion summarization is costly and makes unsupervised methods attractive. Very recently, a neural method for unsupervised multi-document abstractive summarization was proposed by Chu and Liu (2019, Meansum), based on an auto-encoder which is given the average encoding of all documents at inference time. Major limitations identified by the authors of this work are factual inaccuracies and the inability to deal with contradictory statements. We argue that this can be attributed to feeding the decoder the summation of sentence representations in the embedding space, which is not equivalent to the average meaning representation of all the input sentences.

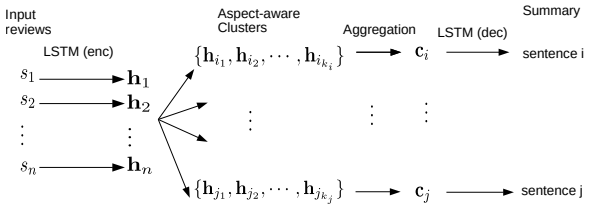

Figure 1: Aspect-aware unsupervised summarization system. The decoder LSTM shares weights with the encoder LSTM.

In this paper, we present a work in progress that investigates better ways of aggregating sentence representations in a way that preserves semantics. While available gold summaries might be expensive to acquire, we leverage more attainable training signals such as a small amount of sentiment and aspect annotations. We adopt a strategy based on a language model – used both for encoding reviews and for generating summaries – and aspect-aware sentence clustering. This clustering ensures coverage of all relevant aspects and allows the system to generate independently a sentence for each aspect mentioned in reviews. Our system proceeds by projecting reviews to a vector space, clustering them according to their main aspect, and generating one sentence for each cluster that has been discovered. Our experiments, performed on the Oposum dataset (Angelidis and Lapata, 2018), demonstrate the importance of the clustering step and assess the effect of leveraging aspect information to improve clustering.

[*]Work done at Naver Labs Europe.

Proceedings of the 2nd Workshop on New Frontiers in Summarization, pages 42–47
Hong Kong, China, November 4, 2019.

2 Related Work

Since obtaining large parallel corpora for opinion summarization is costly, a line of work has focused on unsupervised methods. Proposals in unsupervised opinion summarization include both extractive and abstractive methods. Unsupervised extractive summarization methods consists in selecting the most salient sentences from a text. Saliency can be quantified with the centroid method (Radev et al., 2004; Gholipour Ghalandari, 2017; Rossiello et al., 2017), which consists in computing vector representations for sentences and selecting which sentences are the closest to their centroid, and thus the most representative of the set. Other proposals make use of the PageRank algorithm (Mihalcea and Tarau, 2004; Erkan and Radev, 2004) to compute sentence saliency. The weakly supervised method of Angelidis and Lapata (2018) uses a pretrained polarity classifier and an aspect-based autoencoder to compute the saliency of reviews segments.

On the other hand, non-neural abstractive methods are based on graphs (Ganesan et al., 2010; Nayeem et al., 2018). They consist in constructing a graph whose nodes are words and extract paths that correspond to valid sentences.[1]

This work is inspired by the Meansum model (Chu and Liu, 2019), which leverages an encoder-decoder trained using self-supervision with a sentence reconstruction objective to reconstruct each of the input sentences. It performs summarization by averaging the encoded vectors of each input sentence and feeding them to the pre-trained decoder to generate a summary out of them. The main difference with that work is our use of an LSTM instead of a auto-encoder, and in particular experimenting with different ways of aggregating the documents.

3 Proposed Model

Each product p is associated with a set of reviews represented by a set of sentences $S^{(p)} = \{s_1^{(p)}, \dots s_n^{(p)}\}$. The task consists in predicting a set of sentences (under a word budget) that contains the important information in $S^{(p)}$. Ideally, a good summary should cover all aspects (e.g. price, quality, ease of use) mentioned in reviews, and express judgements that are consistent with those in the reviews.

Overview of the approach Our approach consists in the following general pipeline:

1. Encoding: compute vector representations for sentences;
2. Clustering step: cluster sentence representations into meaningful groups (i.e. cluster together sentences that are about the same aspect);
3. Aggregation: compute a single vector representation for each cluster, from the representations of sentences in the cluster;
4. Generation step: generate a sentence for each cluster.

Each of these modules has a wide range of possible instantiations. In the next four paragraphs, we describe the architecture we implemented for each step.

3.1 RNN Language Model

The main module of our model is a standard LSTM trained with a language model objective. We construct a representation for a sentence s by running the LSTM on the sentence and retrieving the last LSTM state $\mathbf{h}$.

3.2 Sentence clustering

We use a function f_{aspect} that associates a sentence vector representation $\mathbf{h}$ to an aspect identifier $a \in \{1, \dots, n\}$, where n is the total number of aspects. Many possibilities exist to instantiate f_{aspect}, ranging from unsupervised topic modelling (Latent Dirichlet Allocation, Blei et al., 2003), or unsupervised aspect extraction (Aspect-based Autoencoder, He et al., 2017), to weakly supervised approaches (Multi-seed Aspect Extractor, Angelidis and Lapata, 2018). In this paper, we use a supervised aspect classifier to instantiate f_{aspect}, trained jointly with the language model. This choice requires annotated data.

We score possible aspects with a single linear layer followed by a softmax activation:[2]

$$p(A = \cdot | s_i) = \mathsf{Softmax}(\mathbf{W}^{(A)} \cdot \mathbf{h}_i),$$
$$f_{aspect} = \operatorname*{argmax}_a p(A = a | s_i),$$

[1] These methods are semi-extractive: they produce sentences that are not in input reviews, but only use words that occur in them.

[2] During training, we use a sigmoid activation instead, since a segment may be annotated with several aspects, thus treating each aspect as a single binary variable.

where $\mathbf{W}^{(A)}$ is a parameter matrix, and $\mathbf{h}_i$ is the LSTM sentence encoding.

For comparison purposes, we also experiment with k-means clustering, an unsupervised method whose only hyperparameter is a predefined number of clusters.

3.3 Constructing cluster representations

For each cluster $C_a = \{(s_1, \mathbf{h}_1), (s_2, \mathbf{h}_2), \ldots, (s_{k_a}, \mathbf{h}_{k_a})\}$, containing pairs made of a sentence and its vector representation, we need to compute a single representation $\mathbf{h}^{(a)}$ that retain most important information from the original sentences. To do so, we first select the most salient sentences from the cluster, and then compute the centroid of selected sentences.

Following Angelidis and Lapata (2018), we use the output of a polarity classifier to define saliency. In particular, we define the saliency score sal for a single sentence s as the prediction confidence of the classifier:

$$p(Pol = \cdot|s_i) = \mathsf{Softmax}(\mathbf{W}^{(Pol)} \cdot \mathbf{h}_i),$$
$$sal(s_i) = \max_{pol} p(Pol = pol|s_i),$$

where $\mathbf{W}^{(Pol)}$ is a parameter matrix. Finally, we prune the cluster C to the k most salient sentences $C' \subset C$, and compute their centroid:

$$\mathbf{c}_a = \frac{1}{|C'|} \sum_{(s_i, \mathbf{h}_i) \in C'} \mathbf{h}_i.$$

This method can be seen as a form of hard attention, where a few items are attended to, whereas the majority does not participate in the final representation.

3.4 Generating summary sentences

The last step of the summary construction process consists in generating a sentence per cluster. We do so by initializing the language model LSTM with the cluster representation $\mathbf{c}_a$, and performing decoding in the same fashion as a translation model (without attention).

Our decoding method is based on top-k sampling decoding (Fan et al., 2018), i.e. at each time step, we extract the k most probable next tokens, renormalize their probabilities and sample from the resulting distribution. We perform top-k sampling decoding K times. We then rerank the K generated sentences according to the cosine similarity of their representation, as computed by the LSTM, to the cluster representation $\mathbf{c}_a$. This process makes sure that non-relevant sampled sentences are rejected and is meant to improve the semantic similarity between the centroid of the cluster and the generated sentence.

3.5 Multi-Task Training Objective

We train the model using a multitask learning (MTL, Caruana, 1997) objective. We jointly optimize a language modelling objective, as well as the two supervised classification task (aspect and polarity):

$$\mathcal{L}_{\text{lm}} = \sum_{i=1}^{n} -\log P(w_i|w_0^{i-1}; \theta_{\text{LSTM}}),$$
$$\mathcal{L}_{\text{polarity}} = -\log P(y_p|w_0^n; \theta_{\text{LSTM}}, \theta_{\text{polarity}}),$$
$$\mathcal{L}_{\text{aspect}} = -\log P(y_a|w_0^n; \theta_{\text{LSTM}}, \theta_{\text{aspect}}),$$
$$\mathcal{L}_{\text{MTL}} = \mathcal{L}_{\text{lm}} + \mathcal{L}_{\text{polarity}} + \mathcal{L}_{\text{aspect}},$$

where w_0^n is a sentence, y_p is its polarity label, y_a is its aspect labels. In some experiments, we only use the language modelling objective (we optimize $\mathcal{L}_{\text{lm}}$ instead of $\mathcal{L}_{\text{MTL}}$). It is important to note here that while our method uses an MTL objective, it does not require aspect and polarity annotations for the input summaries but rather a small number of annotated examples for training. For the rest of the dataset (not annotated with aspect nor polarity) our model shifts training to solely a language modeling objective.

4 Experiments

Dataset We perform experiments on the Oposum dataset (Angelidis and Lapata, 2018). This dataset contains $3,461,603$ Amazon reviews for 6 types of products, extracted from the *Amazon Product Dataset* (McAuley et al., 2015). We use the raw reviews from Oposum to train the language models (separately for each product type). For each product type, small subsets of reviews are annotated with aspects (1400 sentences), polarities (330 sentences) which we use to train our polarity and aspect classifiers. We use the 10 gold summaries (per product type) additionally provided in the dataset for final evaluation. To train the sentiment and aspect classifiers, we use respectively the development and test sets from Oposum as train

Model	Bags_and_cases	Bluetooth	Boots	Keyboards	TV	Vacuums
TextRank	0.35	0.28	0.31	0.30	0.30	0.30
Mean	0.18 ±0.03	0.15 ±0.02	0.16 ±0.02	0.17 ±0.02	0.16 ±0.03	0.15 ±0.02
Kmeans	0.38 ±0.02	0.37 ±0.01	0.37 ±0.01	0.37 ±0.01	0.35 ±0.01	0.38 ±0.02
Kmeans + MTL	0.38 ±0.01	0.36 ±0.01	**0.38** ±0.02	0.35 ±0.01	0.35 ±0.02	0.36 ±0.02
Aspect + MTL	**0.4** ±0.02	**0.38** ±0.01	**0.38** ±0.01	**0.38** ±0.01	**0.37** ±0.01	**0.39** ±0.01

Table 1: ROUGE-L evaluation per product type.

Model	ROUGE-1	ROUGE-2	ROUGE-L
TextRank	0.27 ±0.02	0.03 ±0.0	0.31 ±0.02
Mean	0.12 ±0.02	0.01 ±0.01	0.16 ±0.03
Kmeans	0.32 ±0.02	0.05 ±0.01	0.37 ±0.02
Kmeans + MTL	0.31 ±0.02	0.05 ±0.01	0.36 ±0.02
Aspect + MTL	**0.33** ±0.02	0.05 ±0.01	**0.38** ±0.02
(Angelidis and Lapata, 2018)	**0.44**	**0.21**	**0.43**

Table 2: ROUGE-{1, 2, L} metrics on the full dataset.

and development sets.[3] We split the polarity annotated sets into a train (90%) and development set (10%).

Protocol and hyperparameters To optimize the objective function in Section 3.5, at each training step, we sample a batch of sentences and perform an update on the language modelling loss ($\mathcal{L}_{\text{lm}}$), then sample a batch of sentences (from the annotated subset) and perform an update on one of the supervised classification losses ($\mathcal{L}_{\text{polarity}}$ on even steps, aspect $\mathcal{L}_{\text{aspect}}$ on odd steps).

For the language model, we use a 2-layer monodirectional LSTM with state size 1000 and randomly initialized word embeddings of size 200. Minibatches have size 10 for the language modelling objective and size 8 for aspect and polarity classification. For the k-means clustering method we set the number of clusters to 8. For the aspect-based clustering we do a grid search over different pruning sizes (16, 100). Finally, at inference time using top-k with re-ranking we set $k = 20$ and $K = 10$ (see Section 3.4). For each product type we run the training process with 2 different seeds and the inference process with 3 different seeds. The results 5 reported are the *mean* and the *std* of the 6 train/inference combinations.

External comparisons As a baseline, we use a publicly available implementation[4] (Barrios et al., 2016) of the TextRank algorithm (Mihalcea and Tarau, 2004). We also compare to the results of the extractive system of Angelidis and Lapata (2018).

Model variations We experiment with four variations of the model:

- No clustering, $\mathcal{L}_{\text{lm}}$ training objective: the summary is generated from the centroid representation of all reviews (as in the Meansum model);
- K-means, $\mathcal{L}_{\text{lm}}$ training objective;
- K-means, $\mathcal{L}_{\text{MTL}}$ objective, this setting assess whether k-means clustering provides better information when the LSTM is trained to incorporate aspect information in its representations (via MTL training);
- Aspect prediction clustering, $\mathcal{L}_{\text{MTL}}$.

5 Results and discussion

We present results in Tables 1 and 2. We report ROUGE-1, ROUGE-2 and ROUGE-L F scores (Lin, 2004) as computed by the `py-rouge` package implementation of ROUGE.[5]

First we observe that clustering reviews and generating a review sentence per cluster (Kmeans) provides a huge benefit over generating a full summary from the centroid of all reviews (Mean), as also done by the MeanSum model. Using K-means clustering with a model trained with multitask learning (Kmeans+MTL) has no effect over the quality of the summaries. Furthermore, we observe that clustering reviews based on the aspect classifier provides a small improvement (+0 to +4 ROUGE-L over K-means clustering). This model outperforms the Textrank baseline on all metrics.

We report in Table 2 the results published by Angelidis and Lapata (2018) on the Oposum dataset. Our system falls short of matching their results. However, the Oposum gold summaries are

[3] The provided split does not include a training set, since the authors only used the annotations for evaluation.

[4] https://github.com/summanlp/textrank

[5] https://github.com/Diego999/py-rouge

extractive, and thus are biased towards extractive methods.

Overall, we also observe that our ROUGE-2 metric is quite low in absolute value, with scores ranging around 0.05. However, those results are consistent with other published results in unsupervised abstractive summarization (on other datasets), e.g. Chu and Liu (2019). This might be related to the fact that the language model is good, so it uses on-topic words (Rouge-1) and does so in the correct order (Rouge-L); but the broader sense of what is being said might not necessarily match with reference summaries.

6 Conclusion

We have presented an unsupervised opinion summarization method, based on language modelling and aspect-based clustering. Preliminary experiments showed the benefits of clustering review sentences into meaningful groups, instead of aggregating them into a single vector as done by the MeanSum model, thus addressing an important limitation of that model. Furthermore, our experiments showed that incorporating aspect information, as predicted by a supervised classifier is beneficial to opinion summarization, and leverages only a small amount of annotated data that is easier to acquire than parallel summarization data.

References

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, Brussels, Belgium. Association for Computational Linguistics.

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606.

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

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Eric Chu and Peter Liu. 2019. MeanSum: a neural model for unsupervised multi-document abstractive summarization. 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(1):457–479.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. 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)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Demian Gholipour Ghalandari. 2017. Revisiting the centroid-based method: A strong baseline for multi-document summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 85–90, Copenhagen, Denmark. Association for Computational Linguistics.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 388–397, Vancouver, Canada. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

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

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. 2018. Abstractive unsupervised multi-document summarization using paraphrastic sentence fusion. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1191–1204, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Dragomir R. Radev, Hongyan Jing, Magorzata Sty, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40(6):919 – 938.

Gaetano Rossiello, Pierpaolo Basile, and Giovanni Semeraro. 2017. Centroid-based text summarization through compositionality of word embeddings. In *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, pages 12–21, Valencia, Spain. Association for Computational Linguistics.