Informative and Controllable Opinion Summarization

Abstract

Opinion summarization is the task of automatically generating summaries for a set of opinions about a specific target (e.g., a movie or a product). Since the number of input documents can be prohibitively large, neural network-based methods sacrifice end-to-end elegance and follow a two-stage approach where an extractive model first pre-selects a subset of salient opinions and an abstractive model creates the summary while conditioning on the extracted subset. However, the extractive stage leads to information loss and inflexible generation capability. In this paper we propose a summarization framework that eliminates the need to pre-select salient content. We view opinion summarization as an instance of multi-source transduction, and make use of all input documents by condensing them into multiple dense vectors which serve as input to an abstractive model. Beyond producing more informative summaries, we demonstrate that our approach allows to take user preferences into account based on a simple zero-shot customization technique. Experimental results show that our model improves the state of the art on the Rotten Tomatoes dataset by a wide margin and generates customized summaries effectively.

Introduction

The proliferation of opinions expressed in online reviews, blogs, internet forums, and social media has created a pressing need for automated systems which enable customers, companies, or service providers to make informed decisions without having to absorb large amounts of opinionated text. Opinion summarization is the task of automatically generating summaries for a set of opinions about a specific target BIBREF0. Figure FIGREF1 shows various reviews about the movie “Coach Carter” and example summaries generated by humans and automatic systems. The vast majority of previous work BIBREF1 views opinion summarization as the final stage of a three-step process involving: (1) aspect extraction (i.e., finding features pertaining to the target of interest, such as battery life or sound quality); (2) sentiment prediction (i.e., determining the sentiment of the extracted aspects); and (3) summary generation (i.e., presenting the identified opinions to the user). Textual summaries are created following mostly extractive methods which select representative segments (usually sentences) from the source text BIBREF2, BIBREF3, BIBREF4, BIBREF5. Despite being less popular, abstractive approaches seem more appropriate for the task at hand as they attempt to generate summaries which are maximally informative and minimally redundant without simply rearranging passages from the original opinions BIBREF6, BIBREF7, BIBREF8, BIBREF9. General-purpose summarization approaches have recently shown promising results with end-to-end models which are data-driven and take advantage of the success of sequence-to-sequence neural network architectures. Most approaches BIBREF10, BIBREF11, BIBREF12, BIBREF13 encode documents and then decode the learned representations into an abstractive summary, often by attending to the source input BIBREF14 and copying words from it BIBREF15. Under this modeling paradigm, it is no longer necessary to identify aspects and their sentiment for the opinion summarization task, as these are learned indirectly from training data (i.e., sets of opinions and their corresponding summaries). These models are usually tested on domains where the input is either one document or a small set of documents. However, the number of opinions tends to be very large (150 for the example in Figure FIGREF1). It is therefore practically unfeasible to train a model in an end-to-end fashion, given the memory limitations of modern hardware. As a result, current approaches BIBREF16, BIBREF17, BIBREF18, BIBREF19 sacrifice end-to-end elegance in favor of a two-stage framework which we call Extract-Abstract: an extractive model first selects a subset of opinions and an abstractive model then generates the summary while conditioning on the extracted subset (see Figure FIGREF5). The extractive pass unfortunately has two drawbacks. Firstly, on account of having access to a subset of opinions, the summaries can be less informative and inaccurate, as shown in Figure FIGREF1. And secondly, user preferences cannot be easily taken into account (e.g., the reader may wish to obtain a summary focusing on the acting or plot of a movie as opposed to a general-purpose summary) since more specialized information might have been removed.

In this paper, we propose Condense-Abstract, an alternative two-stage framework which uses all input documents when generating the summary (see Figure FIGREF5). We view the opinion summarization problem as an instance of multi-source transduction BIBREF20; we first represent the input documents as multiple encodings, aiming to condense their meaning and distill information relating to sentiment and various aspects of the target being reviewed. These condensed representations are then aggregated using a multi-source fusion module based on which an opinion summary is generated using an abstractive model. We also introduce a zero-shot customization technique allowing users to control important aspects of the generated summary at test time. Our approach enables controllable generation while leveraging the full spectrum of opinions available for a specific target. We perform experiments on a dataset consisting of movie reviews and opinion summaries elicited from the Rotten Tomatoes website (BIBREF16; see Figure FIGREF1). Our framework outperforms state-of-the-art models by a large margin using automatic metrics and in a judgment elicitation study. We also verify that our zero-shot customization technique can effectively generate need-specific summaries.

Related Work

Most opinion summarization models follow extractive methods (see BIBREF21 and BIBREF22 for overviews), with the exception of a few systems which are able to generate novel words and phrases not featured in the source text. BIBREF6 propose a graph-based framework for generating ultra concise opinion summaries, while BIBREF8 represent reviews by discourse trees which they aggregate to a global graph from which they generate a summary. Other work BIBREF7, BIBREF23 takes the distribution of opinions and their aspects into account so as to generate more readable summaries. BIBREF9 present a hybrid system which uses extractive techniques to select salient quotes from the input reviews and embeds them into an abstractive summary to provide evidence for positive or negative opinions. More recent work has seen the effective application of sequence-to-sequence models BIBREF24, BIBREF14 to various abstractive summarization tasks including headline generation

BIBREF10, single- BIBREF15, BIBREF25, and multi-document summarization BIBREF16, BIBREF17, BIBREF18. Closest to our approach is the work of BIBREF16 who generate opinion summaries following a two-stage process which first selects documents bearing pertinent information, and then generates the summary by conditioning on these documents. Specifically, they use a ridge regression model with hand-engineered features such as TF-IDF scores and word counts, to estimate the importance of a document relative to its cluster (see also BIBREF17 for a survey of additional document selection methods). The extracted documents are then concatenated into a long sequence and fed to an encoder-decoder model. Our proposed framework eliminates the need to pre-select salient documents which we argue leads to information loss and less flexible generation capability. Instead, a separate model first condenses the source documents into multiple dense vectors which serve as input to a decoder to generate an abstractive summary. Beyond producing more informative summaries, we demonstrate that our approach allows to customize them. Recent conditional generation models have focused on controlling various aspects of the output such as politeness BIBREF26, length BIBREF27, BIBREF28, content BIBREF28, or style BIBREF29. In contrast to these approaches, our customization technique requires neither training examples of documents and corresponding (customized) summaries nor specialized pre-processing to encode which tokens in the input might give rise to customization.

Condense-Abstract Framework

We propose an alternative to the Extract first, Abstract later (EA) approach which eliminates the need for an extractive model and enables the use of all input documents when generating the summary. Figure FIGREF5 illustrates our Condense-Abstract (CA) framework. In lieu of an integrated encoder-decoder, we generate summaries using two separate models. The Condense model returns document encodings for N input documents, while the Abstract model uses these encodings to create an abstractive summary. This two-step approach has at least three advantages for multi-document summarization. Firstly, optimization is easier since parameters for the encoder and decoder weights are learned separately.

Secondly, CA-based models are more space-efficient, since N documents in the cluster are not treated as one very large instance but as N separate instances when training the Condense model. Finally, it is possible to generate customized summaries targeting specific aspects of the input since the Abstract model operates over the encodings of all available documents.

Condense-Abstract Framework ::: The Condense Model

Let $\mathcal {D}$ denote a cluster of N documents about a specific target (e.g., a movie or product). For each document $X=\lbrace w_1,w_2,...,w_M\rbrace \in \mathcal {D}$, the Condense model learns an encoding d, and word-level encodings $h_1, h_2, ..., h_M$. We use a BiLSTM autoencoder as the Condense model. Specifically, we employ a Bidirectional Long Short Term Memory (BiLSTM) encoder BIBREF31:

where $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$ are forward and backward hidden states of the BiLSTM at timestep i, and $;$ denotes concatenation. Training is performed with a reconstruction objective. Specifically, we use a separate LSTM as the decoder where the first hidden state z_0 is set to d (see Equation (5)). Words $w^{\prime }_t$ are generated using a softmax classifier:

The auto-encoder is trained with a maximum likelihood loss:

An advantage of using a separate encoder is increased training data, since we treat a single target with N input documents as N different instances. Once training has taken place, we use the Condense model to obtain N pairs of document encodings $\lbrace d_i\rbrace $ and word-level encodings $\lbrace h_{i,1}, h_{i,2}, ..., h_{i,M}\rbrace $, $1 \le i \le N$ as representations for the documents in $\mathcal {D}$.

Condense-Abstract Framework ::: The Abstract Model

The Abstract model first fuses the multiple encodings obtained from the Condense stage and then generates a summary using a decoder.

Condense-Abstract Framework ::: The Abstract Model ::: Multi-source Fusion

The N pairs of document encodings $\lbrace d_i\rbrace $ and word-level encodings $\lbrace h_{i,1}, h_{i,2}, ..., h_{i,M}\rbrace $, $1 \le i \le N$ are aggregated into a single pair of document encoding $d^{\prime }$ and word-level encodings $h^{\prime }_1, h^{\prime }_2, ..., h^{\prime }_V$, where V is the number of total unique tokens in the input. We fuse document encodings, using an attentive pooling method which gives more weight to important documents. Specifically, we learn a set of weight vectors $a_i \in \mathbb {R}^{D_d}$, where D_d is the dimension of d_i, to weight-sum the document encodings:

where the mean encoding $\bar{d}$ is used as the query vector, and $W_p \in \mathbb {R}^{D_d \times D_d \times D_d}$ is a learned tensor. We also fuse word-level encodings, since the same words may appear in multiple documents. To do this, we simply average all encodings of the same word, if multiple tokens of the word exist:

where V_{w_j} is the number of tokens for word w_j in the input.

Condense-Abstract Framework ::: The Abstract Model ::: Decoder

The decoder generates summaries conditioned on the reduced document encoding $d^{\prime }$ and reduced word-level encodings $h^{\prime }_1,h^{\prime }_2,...,h^{\prime }_V$. We use a simple LSTM decoder enhanced with attention BIBREF14 and copy mechanisms BIBREF32. We set the first hidden state s_0 to $d^{\prime }$, and run an LSTM to calculate the current hidden state using the previous hidden state s_{t-1} and word $y^{\prime }_{t-1}$ at time step t:

At each time step t, we use an attention mechanism over word-level encodings to output the attention weight vector a_t and context vector c_t:

Finally, we employ a copy mechanism over the input words to output the final word probability $p(y^{\prime }_t)$ as a weighted sum over the generation probability $p_g(y^{\prime }_t)$ and the copy probability $p_c(y^{\prime }_t)$:

where W, v, and b are learned parameters, and t is the current timestep.

Condense-Abstract Framework ::: The Abstract Model ::: Salience-biased Extracts

The model presented so far treats all documents as equally important and has no specific mechanism to encourage saliency and eliminate redundancy. In order to encourage the decoder to focus on salient content, we can straightforwardly incorporate information from an extractive step. In experiments, we select k documents using SummaRunner BIBREF33, a state-of-the-art neural extractive model where each document is classified as to whether it should be part of the summary or not. We concatenate k preselected documents into a long sequence and encode it using a separate BiLSTM encoder. The encoded sequence serves as input to an LSTM decoder which generates a salience-biased hidden state r_t. We then update hidden state s_t in Equation (DISPLAY_FORM19) as $s_t = [s_t; r_t]$. Notice that we still take all input documents into account, while acknowledging that some might be more descriptive than others.

Condense-Abstract Framework ::: The Abstract Model ::: Training

We use two objective functions to train the Abstract model. Firstly, we use a maximum likelihood loss to optimize the generation probability distribution $p(y^{\prime }_t)$ based on gold summaries $Y=\lbrace y_1,y_2,...,y_L\rbrace $ provided at training time:

Secondly, we propose a way to introduce supervision and guide the attention pooling weights W_p in Equation () when fusing the document encodings. Our motivation is that the resulting fused encoding $d^{\prime }$ should be roughly equivalent to the encoding of summary y, which can be calculated as $z=\text{\textsc {Condense}}(y)$. Specifically, we use a hinge loss that maximizes the inner product between $d^{\prime }$ and z and simultaneously minimizes the inner product between $d^{\prime }$ and n_i, where n_i is the encoding of one of five randomly sampled negative summaries:

The final objective is then the sum of both loss functions:

Condense-Abstract Framework ::: Zero-shot Customization

Another advantage of our approach is that at test time, we can either generate a general-purpose summary or a need-specific summary. To generate the former, we run the trained model as is and use beam search to find the sequence of words with the highest cumulative probability. To generate the latter, we employ a simple technique that revises the query vector $\bar{d}$ in Equation (DISPLAY_FORM16). More concretely, in the movie review domain, we assume that users might wish to obtain a summary that focuses on the positive or negative aspects of a movie, the quality of the acting, or the plot. In a different domain, users might care about the price of a product, its comfort, and so on. We undertake such customization without requiring access to need-specific summaries at training time. Instead, at test time, we assume access to background reviews to represent the user need. For example, if we wish to generate a positive summary, our method requires a set of reviews with positive sentiment which approximately provide some background on how sentiment is communicated in a review. We use these background reviews conveying a user need x (e.g., acting, plot, positive or negative sentiment) during fusion to attend more to input reviews related to x. Let C_x denote the set of background reviews. We obtain a new query vector $\hat{d} = \sum _{c=1}^{|C_x|} d_c / |C_x|$, where d_c is the document encoding of the c'th review in C_x, calculated using the Condense model. This change allows the model to focus on input reviews with semantics similar to the user need as conveyed by the background reviews C_x. The new query vector $\hat{d}$ is used instead of $\bar{d}$ to obtain document encoding $d^{\prime }$ (see Equation (DISPLAY_FORM16)).

Experimental Setup ::: Dataset

We performed experiments on the Rotten Tomatoes dataset provided in BIBREF16. It contains 3,731 movies; for each movie we are given a large set of reviews (99.8 on average) written by professional critics and users and a gold-standard consensus, i.e. a summary written by an editor (see an example in Figure FIGREF1). On average, reviews are 19.7 tokens long, while the summary length is 19.6 tokens. The dataset is divided into 2,458 movies for training, 536 movies for development, and 737 movies for testing. Following previous work BIBREF16, we used a generic label for movie titles during training which we replace with the original movie names at test time.

Experimental Setup ::: Training Configuration

For all experiments, our model used word embeddings with 128 dimensions, pretrained on the training data using GloVe BIBREF34. We set the dimensions of all hidden vectors to 256, the batch size to 8, and the beam search size to 5. We applied dropout BIBREF35 at a rate of 0.5. The model was trained using the Adam optimizer BIBREF36 and l_2 constraint BIBREF37 of 2. We performed early stopping based on model performance on the development set. Our model is implemented in PyTorch.

Experimental Setup ::: Comparison Systems

We present two variants of our approach: (a) AE+Att+Copy uses the Condense and Abstract models described above, but without salience-biased extracts, while (b) AE+Att+Copy+Salient does incorporate them. We further compared our approach against two types of methods: one-pass methods and methods that use the EA framework. Fully extractive methods include (c) LexRank BIBREF38, a PageRank-like summarization algorithm which generates a summary by selecting the n most salient units, until the length of the target summary is reached; (d) SubModular BIBREF39, a supervised learning approach to train submodular scoring functions for extractive multi-document summarization; (e) Opinosis BIBREF6 a graph-based abstractive summarizer that generates concise summaries of highly redundant opinions; and (f) SummaRunner BIBREF33. EA-based methods include (g) Regress+S2S BIBREF16, an instantiation of the EA framework where a ridge regression model with hand-engineered features implements the Extract model, while an attention-based sequence-to-sequence neural network is the Abstract model; (h) SummaRunner+S2S, our implementation of an EA-based system which uses SummaRunner instead of Regress as the Extract model; and (i) SummaRunner+S2S+Copy, the same model as (h) but enhanced with a copy mechanism BIBREF32. For all EA-based systems, we set $k=5$, which is tuned on the development set. Larger k leads to worse performance, possibly because the Abstract model becomes harder to optimize.

Results ::: Automatic Evaluation

We considered two evaluation metrics which are also reported in BIBREF16: METEOR BIBREF40, a recall-oriented metric that rewards matching stems, synonyms, and paraphrases, and ROUGE-SU4 BIBREF41 which is calculated as the recall of unigrams and skip-bigrams up to four words. We also report F1 for ROUGE-1, ROUGE-2, and ROUGE-L, which are widely used in summarization BIBREF41. They respectively measure word-overlap, bigram-overlap, and the longest common subsequence between the reference and system summaries. Our results are presented in Table TABREF28. The first block shows one-pass systems, both supervised (SubModular, SummaRunner) and unsupervised (LexRank, Opinosis). We can see that SummaRunner is the best performing system in this block; despite being extractive, it benefits from training data and the ability of neural models to learn task-specific representations. The second block in Table TABREF28 shows several two-pass abstractive systems based on the EA framework. Our implementation of an EA-based system, SummaRunner+S2S+Copy, improves over the purely extractive SummaRunner and the previously reported best EA-based system, Regress+S2S. The third block presents two models using the proposed CA framework. Both systems outperform all other models across all metrics; AE+Att+Copy+Salient is the best model overall which exploits information about all documents and most salient ones.

Results ::: Human Evaluation

In addition to automatic evaluation, we also assessed system output by eliciting human judgments. Participants compared summaries produced from the best extractive baseline (SummaRunner), and the best EA- and CA-based systems (SummaRunner+S2S+Copy and AE+Att+Copy+Salient, respectively). As an upper bound, we also included Gold standard summaries. The study was conducted on the Amazon Mechanical Turk platform using Best-Worst Scaling (BWS; BIBREF42), a less labor-intensive alternative to paired comparisons that has been shown to produce more reliable results than rating scales BIBREF43. Specifically, participants were shown the movie title and basic background information (i.e., synopsis, release year, genre, director, and cast). They were also presented with three system summaries and asked to select the best and worst among them according to Informativeness (i.e., does the summary convey opinions about specific aspects of the movie in a concise manner?), Correctness (i.e., is the information in the summary factually accurate and does it correspond to the information given about the movie?), and Grammaticality (i.e., is the summary fluent and grammatical?). Examples of system summaries are shown in Figure FIGREF1 and Figure FIGREF37. We randomly selected 50 movies from the test set and compared all possible combinations of summary triples for each movie. We collected three judgments for each comparison. The order of summaries and movies was randomized per participant. The score of a system was computed as the percentage of times it was chosen as best minus the percentage of times it was selected as worst. The scores range from -1 (worst) to 1 (best) and are shown in Table TABREF36. Perhaps unsurprisingly, the human-generated gold summaries were considered best, whereas our model (AE+Att+Copy+Salient) was ranked second, indicating that humans find its output more informative, correct, and grammatical compared to other systems. SummaRunner was ranked third followed by SummaRunner+S2S+Copy. We inspected the summaries produced by the latter system and found they were factually incorrect bearing little correspondence to the movie (examples shown in Figure FIGREF37), possibly due to the huge information loss at the extraction stage. All pairwise system differences are statistically significant using a one-way ANOVA with posthoc Tukey HSD tests ($p < 0.01$).

Results ::: Customizing Summaries

We further assessed the ability of CA-based systems to generate customized summaries at test time. As discussed earlier, customization at test time is not trivially possible for EA-based systems and as a result we cannot compare against them. Instead, we evaluate two CA-based systems, namely AE+Att+Copy and AE+Att+Copy+Salient. Similar to EA-based systems, the latter biases summary generation towards the k most salient extracted opinions using an additional extractive module, which may not contain information relevant to the user's need (we set $k=5$ in our experiments). We thus expect this model to be less effective for customization than AE+Att+Copy which makes no assumptions regarding which summaries to consider. In this experiment, we assume users may wish to control the output summaries in four ways focusing on acting- and plot-related aspects of a movie review, as well as its sentiment, which may be positive or negative. Let Cust(x) be the zero-shot customization technique discussed in the previous section, where x is an information need (i.e., acting, plot, positive, or negative). We sampled a small set of background reviews C_x ($|C_x|$=1,000) from a corpus of 1 million reviews covering 7,500 movies from the Rotten Tomatoes website, made available in BIBREF29. The reviews contain sentiment labels provided by their authors and heuristically classified aspect labels. We then ran Cust(x) using both AE+Att+Copy and AE+Att+Copy+Salient models. We show in Figure FIGREF37 customized summaries generated by the two models. To determine which system is better at customization, we again conducted a judgment elicitation study on AMT. Participants read a summary which was created by a general-purpose system or its customized variant. They were then asked to decide if the summary is generic or focuses on a specific aspect (plot or acting) and expresses positive, negative, or neutral sentiment. We selected 50 movies (from the test set) which had mixed reviews and collected judgements from three different participants per summary. The summaries were presented in random order per participant.

Table TABREF40 shows what participants thought of summaries produced by non-customized systems (see column No) and systems which had customization switched on (see column Yes). Overall, we observe that AE+Att+Copy is able to customize summaries to a great extent. In all cases, crowdworkers perceive a significant increase in the proportion of aspect x when using Cust(x).
AE+Att+Copy+Salient is unable to generate need-specific summaries, showing no discernible difference between generic and customized summaries. This shows that the use of an extractive module, which is used as one of the main components of EA-based approaches, limits the flexibility of the abstractive model to customize summaries based on a user need.

Conclusions

We proposed the Condense-Abstract (CA) framework for opinion summarization. Both automatic and human-based evaluation show that CA-based approaches produce more informative and factually correct summaries compared to purely extractive models and models including an extractive summary pre-selection stage. We also show that a simple zero-shot customization technique is able to generate aspect- and sentiment-based summaries at test time. In the future, we plan to apply CA-based approaches to other multi-document summarization tasks and domains. It would also be interesting to investigate an unsupervised or semi-supervised approach where reviews are available but no (or only a few) gold-standard summaries are given.