ASSESS: Abstractive Summarization System for Evaluative Statement Summarization

Abstract:

The Internet provides many sources of unsolicited opinions, expressed through user reviews of consumer products, blogs, and forum discussions. Systems which could automatically summarize these opinions would be immensely useful to those wishing to use this information to make decisions. Most past work in automatic summarization has focused on *extractive summarization*, in which key sentences from the source text are identified and extracted to form the output. An alternative framework is *abstractive summarization.* Information from the source text is first extracted into the form of abstract data which is then processed, and from which the most important messages are inferred. This work built upon past work to create a completely automatic system which could produce abstractive summaries from a plain text corpus of product reviews without the need for any prior manual annotation. As an additional contribution, I also devised an improvement for a crucial step of the summarization process. To the best of our knowledge, this is the first complete system which effectively performs this task.

Author Information:

Nicholas FitzGerald

Undergraduate Student
BSc. Cognitive Systems (Computational Intelligence and Design stream)
University of British Columbia

nfitz@interchange.ubc.ca

Advisor:

Dr. Giuseppe Carenini
Department of Computer Science, CICSR #129,
University of British Columbia, 2366 Main Mall
Vancouver, B.C. Canada V6T 1Z4
Office Phone: (604) 822-5109
Fax: (604) 822-5485
email: carenini@cs.ubc.ca
www: http://www.cs.ubc.ca/~carenini/

ASSESS: Abstractive Summarization System for Evaluative Statement Summarization

Nicholas FitzGerald- *nfitz@interchange.ubc.ca*
UBC Laboratory of Computational Intelligence - Dr. Giuseppe Carenini & Dr. Raymond Ng

One benefit of the Internet is that large volumes of text are available on almost any topic – whether through web pages, blogs, or online databases of print resources. As this volume continues to increase, it has become impossible for any one person to read everything that is available. Specifically, many opinions are expressed through user reviews of consumer products, blogs, and forum discussions. Systems which could automatically summarize these opinions into a more manageable size and provide directed links back to the original source material would be immensely useful to those who wish to use this information to make decisions, eg. consumers looking to buy a product, or marketing managers interested in gauging public opinion.

In this work, I built upon the work of Carenini et al. [06a] to create a complete system which could produce abstractive summaries of reviews from a plain text corpus of reviews without the need for any prior annotation. To the best of our knowledge, this is the first complete system which effectively performs this task. As an additional contribution, I also devised an improvement for a crucial step of the summarization process (see step 3).

The only other input required for this system is a hierarchy of User Defined Features (UDFs). This is a hierarchical list of features of the product important to the user. For instance, for reviews of a Digital Camera, UDFs might include “Image Quality”, “Battery Life” and “Price”. “Image Quality” could be further separated into “Resolution”, “Focus” and “Contrast”, forming a hierarchy. These hierarchies are quick and easy to create.

The major steps involved in this system are as follows:

Step 1: Feature Extraction

Reviewers provide opinions on many features of a product within their reviews, and may use a variety of different terms to refer to the same feature. For example “picture”, “photo” and “clarity” could all refer to the feature “Image Quality”. The first step of the information extraction is to identify the feature terms each review mentions. This step produces a list of Crude Feature terms (CFs) which is then mapped to the User Defined Features (UDFs) in step 3. This allows the output summary to aggregate opinions expressed about one feature via different terms, and reduces redundancy in the summary.

For feature extraction the focus was on capturing as many features as possible (high recall rate), rather than on accuracy of output candidates (high precision). This is because non-features produced at this step tend to be pruned out later by the CF-UDF mapping (step 3). Feature extraction was achieved by implementing the algorithm described by Hu and Liu [04]. I experimented with other algorithms including [Scaffadi 06] however Hu, Liu produced the best recall while still producing a reasonable list of feature candidates.

Step 2: Semantic Orientation Scored

The ideal system would analyze each feature within a sentence separately, and determine the opinion expressed towards each feature. However, for sentences with multiple features and complex grammatical structure, this becomes a very difficult task (see [Kessler, Nicolov 09] sec. 6). An reasonable approximation is to calculate the overall sentiment of the sentence and apply that to all features within the sentence. The downside of this approach is that it will incorrectly handle sentences which express multiple, contrasting opinions eg. *“This camera produces great photos, but battery life is simply terrible”* wherein *image quality* is rated highly, but *battery life* is given a low rating. However, in the interest of creating a working pipeline, this approximation was necessary, and should be ameliorated in a sufficiently large corpus. [Voll, Taboda 07] provides a method for calculating a semantic orientation score for a sentence by averaging scores from lexicon of positive and negative terms, adjusted by syntactic functions such as negation or emphasis. This lexicon is built from a seed list which is expanded based on distributional co-occurrence using Google hits. These scores are then normalized onto the -3 to +3 scale used by [Carenini et al. 06a] and applied to all features found within the sentence. To avoid the problem described above, sentences which contain contrasting opinion terms are ignored. Empirically, this is approximately 15% of sentences in the corpora tested. See Table 1 for example outputs from step 1 and 2 – review sentences tagged with features and semantic orientation scores.

Sentence	Tags
olympus c5050 is a very solid camera with a metal body	Camera[+1]
i did n't care much for the tilting lcd screen , it seemed very limited and not very usefull	lcd screen[-1]

Table 1: Example output from steps 1 & 2 from Canon corpus. Sentences are tagged with the extracted features, which are assigned scores based on the polarity and strength of expressed opinions.

Step 3: CF to UDF Mapping

My main original contribution was an improvement of the CF-UDF mapping step. [Carenini et al. 06a] provide a set of simple algorithms for mapping which are moderately successful. The downside is that the most successful of these algorithms require a tuned parameter, Θ, which acts as a threshold for accepting mappings, and which must be empirically determined for each domain (eg. Cameras, DVD players, Restaurants etc). Ideally, a mapping algorithm should not require such tuning and should function well in a wide range of domains.

I started by surveying the NLP literature on synonym matching and implementing 17 different algorithms, and analyzed their effectiveness on the metrics and corpora of [Carenini et al. 06a]. These algorithms fell into two main categories: Lexicon-Based algorithms using [WordNet] – a lexical database of terms encompassing relationships between words such as homonymity and hyponymity (X “is-a” Y), and Distribution-Based scores using search-engine co-occurance data. Several of these algorithms did quite well, however most still required optimization of Θ. In order to improve upon these scores I created a simple voting algorithm: if the majority of the mapping-scores in the vote mapped a given CF to a UDF, the mapping would be applied, otherwise it would be discarded. For the scores requiring a tuned parameter, the parameter was set such as to maximize recall (Θ = 0). I then tested mapping for all possible combinations of 3, 5 and 7 scores. See Table 2 for a summary of scores of the top 3 individual algorithms, and 3- and 5-score voting groups.

The highest accuracy achieved by an individual metric is .787 for lch_score. This represents the maximum score achieved by the metric with Θ optimized. For certain combinations of voting groups, accuracy scores exceeded those of any of the individual algorithms – even when those algorithms' parameters were perfectly optimized. This means that better mappings can be achieved with no empirical optimization of parameters required. The precision which is lost by setting parameters for maximal recall is made up for by the regulatory effect of voting. Also interesting were the patterns evident in the most successful voting groups, which allow us to reliably pick a successful group of metrics to use. A good general rule is to pick one lexical metric, one term-level distributional metric, and syn_score (for a vote n=3).

Name	**Θ**	**Acc**	**RR**	**prec**	**recall**
lch_score	0.744	0.787	0.179	0.923	0.562
sim_score	0.583	0.782	0.197	0.902	**0.578**
wup_score	0.640	0.772	0.145	*0.939*	0.484
lch_tngd_syn	0	*0.790*	**0.242**	0.907	**0.609**
lin_tngd_syn	0	0.790	*0.212*	0.925	**0.578**
lin_tpmi_syn	0	0.787	0.197	0.923	*0.563*
pmi_lin_lesk_tngd_syn	0	**0.801**	**0.220**	0.929	**0.609**
pmi_lin_tavmi_lesk_syn	0	**0.797**	0.198	**0.973**	0.563
tpmi_pmi_lin_lesk_syn	0	**0.797**	0.190	**0.972**	0.547

Table 2: Scores of top 3 individual mapping algoriths, 3- and 5-score voting groups on APEX corpus. See [Carenini et al 06a] for definition of Accuracy and RR metrics. Note that the individual algorithms have optimal theta value, while voting groups require no parameter optimization.

Step 4: Summary Generation

Having extracted the necessary feature and semantic content, and produced a mapping of CF-UDFs, our abstractive summarizer produces a succinct, grammatical summary of the most relevant information. Furthermore, each statement in the summary provides a link to the source material (the reviews) which illustrates the generalization. This means that output test functions not only as a summary, but as a means to access the details of the source material. The information extracted from the review corpus is also displayed as a TreeMap visualization, described in [Carenini et al. 06b].

Results

Preliminary results for the completed pipeline are very encouraging. Summaries of the two corpora tested in [Carenini et al. 06a] seem to correspond quite well to expectations. In addition, the pipeline was tested on an entirely new domain – restaurant reviews. Despite none of the component parts of the system having been designed with this domain in mind, the pipeline seems to handle it quite well and produce reasonable outputs.

The following is an example summary for reviews of Chambar Belgian Restaurant downloaded from *http://dinehere.ca*. In the HTML output of the program, the bracketed numbers link to specific sentences in the corpora which illustrate the point being made:

Almost all users thought the Chambar Belgian Restaurant [1] was good because many diners found the service [2] to be good. This was because even though customers had mixed opinions about the reservations [3],[4], many users liked the servers [5]. Furthermore, many customers found the price [6] to be good. Also, many diners thought the ambiance [7] was very good possibly because several customers found the noise level [8] to be good. Finally, almost all customers liked the menu [9] because many customers found the drinks [10] to be good. Also, many users liked the set menu[11]. Finally, almost all users found the food [12] to be good.

Future Work

One line of future work will focus on dealing with the problem of inter-sentential opinion differentiation described in Step 2. Ideally, out system should be able to distinguish the feature targeted by an individual sentiment expression (see [Kessler, Nicolov 09]). Furthermore, improvements to the Natural Language Generation component should increase summary coherence and readability.

References

[Carenini et al. 06a] C. Carenini, R. Ng, and A. Pauls. Multi-Document Summarization of Evaluative Text. In Proc. of the Conf. of the European Chapter of the Association for Computational Linguistics, 2006.

[Carenini et al. 06B] Giuseppe Carenini , Raymond T. Ng, Adam Pauls, Interactive multimedia summaries of evaluative text, Proceedings of the 11th international conference on Intelligent user interfaces, January 29-February 01, 2006, Sydney, Australia

[Hu, Liu 04] Hu, M., and Liu, B. (2004c). Mining opinion features in customer reviews. In Proc. AAAI.

[Scaffidi 06] Scaffidi, C. Application of a Probability-Based Algorithm to Extraction of Product Features from Online Reviews. Tech. Report CMU-ISRI-06-111, School of Computer Science, Carnegie Mellon University, 2006.

[Kessler, Nicolov 09] Kessler, J. and Nicolov, N. “Targeting Sentiment Expressions through Supervised Ranking of Linguistic Configurations”. 3rd International AAAI Conference on Weblogs and Social Media (ICWSM 2009), 2009. San Jose, California.

[Voll, Taboda 07] Voll, K. and Taboada, M. (2007) “Not All Words are Created Equal: Extracting Semantic Orientation as a Function of Adjective Relevance”, 20th Australian Joint Conference on A.I. 2007, Gold Coast, Australia, Dec 2-6, pp. 337-346

[WordNet] WordNet an electronic lexical database. Cambridge, Mass: MIT, 1998. Print.