Analysis and Synthesis of Help-desk Responses

Yuval Marom and Ingrid Zukerman

School of Computer Science and Software Engineering
Monash University
Clayton, VICTORIA 3800, AUSTRALIA
{yuvalm,ingrid}@csse.monash.edu.au

Abstract. We present a corpus-based approach for the automatic analysis and synthesis of email responses to help-desk requests. This approach can be used to automatically deal with repetitive requests of low technical content, thus enabling help-desk operators to focus their effort on more difficult requests. We propose a method for extracting high-precision sentences for inclusion in a response, and a measure for predicting the completeness of a planned response. The idea is that complete, high-precision responses may be sent directly to users, while incomplete responses should be passed to operators. Our results show that a small but significant proportion (14%) of our automatically generated responses have a high degree of precision and completeness, and that our measure can reliably predict the completeness of a response.

1 Introduction

Email inquiries sent to help desks are often repetitive, and generally "revolve around a small set of common questions and issues" (`http://customercare.telephonyonline.com/ar/telecom_next_generation_customer/C`). This means that help-desk operators spend most of their time dealing with problems that have been previously addressed. Further, a significant proportion of help-desk responses contain a very low level of technical content, corresponding, for example, to inquires addressed to the wrong group, or insufficient detail provided by the customer about his/her problem. Organizations and clients would therefore benefit if the efforts of human operators were focused on difficult, atypical problems, and an automated process was employed to deal with the easier problems.

In this paper, we present an initial report of our corpus-based approach to achieving this objective. This approach consists of automatically generating responses to users' "easy" requests on the basis of similar responses seen in a corpus of email dialogues (easy requests have a low level of specific technical detail). Our approach is essentially that used for extractive multi-document summarization, in that similar documents (email responses) are first identified, followed by the automatic extraction of important sentences. However, there is an important difference between our task and traditional multi-document summarization. Normally, the inclusion of an irrelevant information item in a summary does not invalidate the summary. In contrast, in our application, a response email that contains even one incongruous sentence may alienate a user. As a result, the responses generated by our system must have very high relevance (often at the expense of completeness).

Request:
Return label was not under the shipping tag and I have been waiting nearly two weeks for a label after reporting it not attached to the box.
Complete response:
I apologize for the delay in responding to your issue. Your request for a return airbill has been received and has been sent for processing. Your replacement airbill will be sent to you via email within 24 hours.

Request:
Hi There, I acquired a HP T3000 1.6G/3.2G Colorado Tape Drive from a friend and would like to know how I go about setting it up for use with WinXP. XP does not seem to detect the drive at all. HELP?
Incomplete response:
Thank you for contacting Hewlett-Packard's Customer Care Technical Center. We are only able to assist customers with in warranty products through our email services. At the present time, we have the following number*s* to contact technical support for your out of warranty product. *Please call PHONENUM. This facility will be available from Monday to Friday between 9.00 AM to 5.00 PM. For additional information, please visit the link given below: WEBSITE.*

Fig. 1. Request with a complete response (top) and request with an incomplete response (bottom)

To generate such responses, we have developed a procedure that selects high-precision sentences from a cluster of similar responses, and a measure that predicts the completeness of the resultant responses from the features of their source cluster. The idea is that high-precision responses with a high predicted degree of completeness may be sent directly to users, while incomplete responses should be passed to an operator. For example, the top part of Figure 1 shows a request and a complete response automatically generated by our system; the bottom part shows a request and an incomplete response (the additional information in the operator's response and the extra plural in "numbers" in our system's response have been italicized).

Our corpus consists of 30000 email dialogues between users and help-desk operators at Hewlett-Packard. These dialogues deal with a variety of user requests, which include requests for technical assistance, inquiries about products, and queries about how to return faulty products or parts. As a first step, we have automatically clustered the corpus according to the subject line of the first email. This process yielded 38 topic-based datasets that contain between 25 and 8000 email dialogues. Owing to time limitations, the procedures described in this paper were applied to approximately 40% of the data.

2 System Description

Our system analyzes email responses in a particular topic, and then synthesizes responses in two modes: generic or user-driven, as shown in Figure 2. Generic synthesis involves generating model responses, that is, responses that are representative of all the responses seen in the corpus. User-driven synthesis involves generating the most appropriate response for a given user request.

In the **analysis** phase sentences are first extracted from the help-desk responses, and represented by means of binary vectors of size N (number of fea-

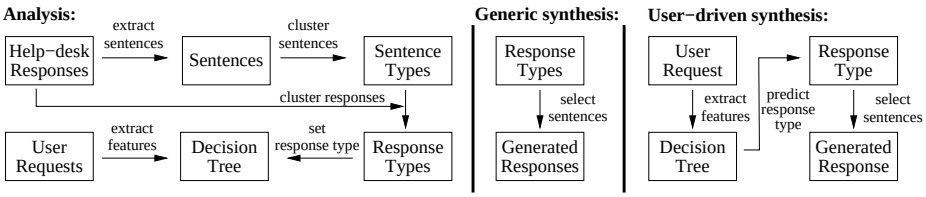

Fig. 2. Analysis and synthesis of responses

ture words in the dataset), where element j is 1 if word w_j is present in the sentence, and 0 otherwise. A clustering program is used to group the sentences into *Sentence Types (STs)*. The responses are then represented in terms of the sentence types that they contain. This is achieved by inspecting all the sentences in a response, and assigning a value of 1 for a sentence type if there is a sentence that belongs to this sentence type, and 0 otherwise. This binary representation is used to cluster the responses into *Response Types (RTs)*.

The *Semantic Compactness* (*SemCom*) of each response type is then calculated. This is a measure that predicts whether it is possible to generate a complete, high-precision response from this response type. It calculates, for each response type RT_j, the proportion of highly cohesive and frequent sentence types among the sentence types that have some presence in that response type:

$$SemCom(RT_j) = \frac{\sum_{i=1}^{m} [\ Cohesion(ST_i) > 0.9 \quad \wedge \quad Prop_j(ST_i) > 0.75\ ]}{\sum_{i=1}^{m} Prop_j(ST_i) > 0.1}$$

where $Prop_j(ST_i)$ corresponds to the proportion of responses in RT_j that use sentence type i, m is the number of sentence types discovered in the analysis phase, and $Cohesion(ST_i)$ is a cohesion score calculated for each sentence type:

$$Cohesion(ST_i) = \frac{1}{N} \sum_{k=1}^{N} [\ \Pr(w_k \in ST_i) \leq 0.01 \ \vee \ \Pr(w_k \in ST_i) \geq 0.99\ ]$$

where $\Pr(w_k \in ST_i)$ is the probability that word w_k is used in sentence type i, and N is the number of words in the dataset. The rationale for this cohesion measure is that a cohesive group of sentences should agree strongly on the words they use and the words they omit. Hence, it is easier to find a sentence that adequately represents a cohesive sentence type than a non-cohesive one.[1]

Overall, *SemCom* provides a level of confidence that the generated response will be representative of the responses actually used by the help-desk operators. If its value is high, the response type is deemed semantically compact, which means that it is a good candidate for automatic response generation. As the value decreases, so does the confidence of automatically generating a complete response from the response type in question. Before generating a response from a response type, the system compares its semantic compactness with an empirically determined threshold, in order to determine whether an operator should

[1] The thresholds used in the equations were determined empirically, and chosen specifically to implement a cautious approach that avoids including potentially incongruous sentences in automatically generated responses. However, we have performed a sensitivity analysis which shows that the quality of our responses is largely maintained even if we relax some of these thresholds. For more details on this analysis, see [1].

participate in the composition of the reply. In Section 3, we evaluate the semantic compactness measure, and suggest a value for its threshold.

After clustering the responses into response types, a decision tree is trained to predict the response type from features of a user's request. The features currently extracted from the requests are the words that they contain. For each user request in the dataset, the response type is set to the one that the actual response in the corpus belongs to (recall that a response type is a cluster of responses). Thus, user requests are paired with response types and these pairs act as supervised examples for the decision tree.

The **synthesis** phase involves generating a responses from response types. For this purpose we use a modified version of the *adaptive greedy algorithm* proposed in [2] for sentence selection. When selecting sentences for inclusion in a response, the system favours sentences that are representative of sentence types that (a) have a high probability of appearing in the response type in question, and (b) are highly cohesive. The generic synthesis mode involves generating a response for each response type, while the user-driven mode involves generating a single response from the response type predicted by the decision tree for a particular user request.

Examples

The example at the top of Figure 1 is generated from a dataset about product replacements. This dataset contains 1205 email dialogues, and the response emails contain 3598 individual sentences. These sentences are encoded into binary vectors of size 76 (the number of feature words) and clustered into 25 sentence types.[2] Then, the response emails are encoded as vectors of size 25 (the number of sentence types) and clustered, yielding 10 response types. Response type 10, which was used to generate the output in this example, has a perfect semantic compactness (1.0). It represents about 860 responses (71% of the dataset), which all use three highly cohesive sentence types. This means that the sentences shown in the figure are identical to almost all the other sentences in the respective sentence types.

In contrast, the example shown at the bottom of Figure 1 (from a different dataset) is generated from a response type with semantic compactness 0.25. This means that the three highly cohesive and probable sentence types that it uses to generate a response only account for about a quarter (on average) of the sentence types used by the responses represented by this response type. The completing text in the figure shows an example of the kind of sentences that the response type is uncertain about — this kind of information is too specific (phone numbers and operation times).

The user-driven component of the system is currently in development, but we provide here two examples of its preliminary operation. In one dataset the decision tree contains a split on the word "xp", which differentiates two response types. The two response types are very similar, both requesting more information from the user and providing contact numbers for out-of-warranty products. The

[2] The clustering program we are using automatically decides on the number of clusters to generate.

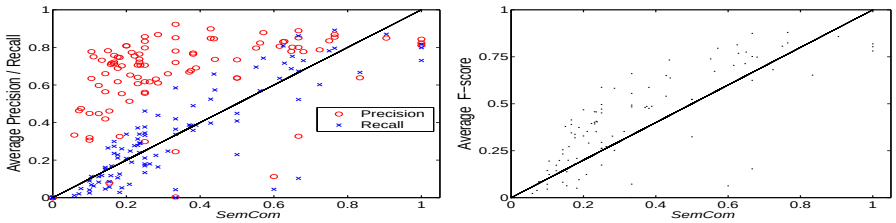

Fig. 3. Relationship between *SemCom* and precision/recall (left), and F-score (right)

main difference between them is additional information for XP users, who are referred to another service for additional support. In a different dataset, the responses are so varied that for most of them the system can only generate the sentence "Thank you, HP eServices". However, the decision tree predicts that if the words "cp-2e" or "cp-2w" are present, referring to specific router models, then a response type with very high semantic compactness can be generated, informing the user that he or she has contacted the wrong group and providing the correct address.

3 Evaluation

In this section, we demonstrate the predictive power of our semantic compactness measure, and the ability of our procedure to generate high-precision generic responses with a high level of completeness. Our *SemCom* measure is designed to predict the completeness of an automatically-generated response composed of high-precision sentences. In order to determine the utility of this measure, we examine how well it correlates with the quality of the generated responses.

We assess the quality of a generated response r_g by comparing it with the actual responses in the response type from which r_g was sourced. These comparisons were performed both manually by a panel of human judges, and automatically using three well-known Information Retrieval measures: precision, recall and F-score. Precision gives the proportion of words in r_g that match those in an actual response; recall gives the proportion of words in the actual response that are included in r_g; and F-score is the geometric average of precision and recall. Precision, recall and F-score are then averaged over the responses in r_g's response type to give an overall evaluation of r_g.

Figure 3 shows the relationship between semantic compactness and precision, recall and F-score for the 135 response types created for the different datasets we have used. From the Figure we see that precision is generally high, and is uncorrelated with *SemCom*. This is not surprising, as the sentence-selection process is designed to select high-precision sentences. Hence, so long as at least one sentence is selected, the text in the generated response r_g will agree with the text that appears in the responses represented in r_g's response type. In contrast, recall is highly correlated with *SemCom*. A decrease in *SemCom* indicates that fewer sentences are included in the generated response, which therefore covers less of the information in the original responses. As expected from these results, the overall F-score is also highly correlated with semantic compactness (the linear and log correlations between our measure and the F-score are 0.89 and 0.9 respectively). Figure 3 suggests a threshold of 0.7 to indicate high semantic

compactness, and a further threshold of 0.4 to indicate medium semantic compactness. The idea is that these thresholds will assist in the selection a response generation strategy for a response type.

As indicated above, we also conducted the following small study in order to assess whether people agree with the predictions made by *SemCom*. We constructed four evaluation sets by selecting four response types with high semantic compactness (≥ 0.7), automatically generating a response from each response type, and selecting 15 actual responses from each response type for comparison.[3] Each evaluation set was given to two judges, who were asked to rate the precision and completeness of the generated response compared to each of the 15 responses in the set. Our judges gave all the automatically generated responses high precision ratings, and completeness ratings which were consistent with our semantic compactness measure.

The overall performance of our system was measured in terms of the proportion of high-precision, complete responses that can be generated from our corpus without human intervention. These are the responses that are represented by response types with high semantic compactness. As mentioned above, Figure 3 suggests a threshold of 0.7 for high semantic compactness. That is, responses that are generated from response types that exceed this threshold could be directly sent to users. This would result in the automatic remittance of approximately 14% of the responses. The application of the medium semantic compactness threshold of 0.4 would result in a further 6% of the generated responses being passed to an operator. The remaining 80% of the responses would have to be mostly written by an operator. However, this may be a pessimistic estimate, as some response types with a low *SemCom* yield reasonable responses, such as that at the bottom of Figure 1. It is also worth noting that the above percentages vary across the different datasets, which indicates that it may be fruitful to focus the automatic response-generation effort on particular topics.

4 Related Research

The idea of clustering text and then generating a summary from the clusters has been implemented in previous multi-document summarization systems [3, 4, 2]. A key issue highlighted in such work is the choice of features used in the clustering. Radev et al. used low-level word-based features [3], while Hatzivassiloglou and colleagues used higher-level, grammatical features obtained through part-of-speech tagging [4, 2].

Our work differs from previous work on clustering and summarization in two respects. Firstly, the high-level features (sentence types) we use to cluster documents are learned from the corpus in an unsupervised manner, using as input only low-level, word-based features. Secondly, our reliance on sentence types enables us to identify response patterns beyond those identified by topic words, and hence allows us to generate different types of summaries within a single topic.

Some examples of user-driven summarization are [5, 6]. The former involves spreading activation from the terms in a query to the terms in news articles to be

[3] Several of our automatically-generated responses match perfectly the operators' responses. Since these are obvious matches, they were not included in our study.

summarized. The latter involves selecting an answer with the highest posterior probability on the basis of its probability in the corpus and its match with a user's query. The corpus here corresponds to an FAQ, in which there are unique question-answer pairs. The difference between these examples and our user-driven approach is that our system not only matches a user's request with a response, but it can also provide a guarantee of how representative the response is to previous, similar requests in the corpus.

The work that most resembles our approach to automating response generation is [7]. This system retrieves and ranks responses for a query, and then presents a sorted list to a human operator. In this list the most relevant sentences are highlighted, thus assisting the composition of a response. In contrast to our approach, there is no attempt here to fully automate response generation.

5 Conclusion

We have offered a corpus-based approach for the automatic analysis and synthesis of responses to help-desk requests — a task where users exhibit a very low tolerance to irrelevant information. We have also proposed a novel measure that reliably predicts the completeness of a high-precision response, and that can be used to select a response-generation strategy.

Our approach, which uses an unsupervised learning perspective in combination with a simplistic word-based representation, has enabled our system to generate a small but significant proportion (14%) of the email responses in our corpus, without the need for human intervention. We believe that with a more powerful approach, further advances are possible for automating the remaining responses. We are tackling such improvements in our on-going work, which includes performing linguistic analysis to extract higher-level discourse features, and applying machine learning techniques to extract pragmatic features.

References

1. Marom, Y., Zukerman, I.: Corpus-based generation of easy help-desk responses. Technical Report 2005/166, School of Computer Science and Software Engineering, Monash University, Clayton, Australia (2005)
2. Filatova, E., Hatzivassiloglou, V.: Event-based extractive summarization. In: Proceedings of ACL'04 Workshop on Summarization, Barcelona, Spain (2004) 104–111
3. Radev, D., Jing, H., Budzikowska, M.: Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In: ANLP/NAACL2000 Workshop on Summarization, Seattle, Washington (2000)
4. Hatzivassiloglou, V., Klavans, J., Holcombe, M., Barzilay, R., Kan, M., McKeown, K.: SimFinder: A flexible clustering tool for summarization. In: Proc. NAACL Workshop on Automatic Summarization, Pittsburgh, Pennsylvania (2001)
5. Mani, I., Bloedorn, E.: Multi-document summarization by graph search and matching. In: AAAI97 – Proceedings of the Fourteenth National Conference on Artificial Intelligence, Providence, Rhode Island (1997) 622–628
6. Berger, A., Mittal, V.: Query-relevant summarization using FAQs. In: Proc. 38th Annual Meeting of the Association for Computational Linguistics (ACL2000), Hong Kong (2000) 294–301
7. Carmel, D., Shtalhaim, M., Soffer, A.: eResponder: Electronic question responder. In: CooplS '02: Proceedings of the 7th International Conference on Cooperative Information Systems, Eilat, Israel (2000) 150–161