Marom, Yuval and Ingrid Zukerman. 2009. An empirical study of corpus-based response automation methods for an e-mail-based help-desk domain. *Computational Linguistics*, uncorrected proof.

An Empirical Study of Corpus-Based Response Automation Methods for an E-mail-Based Help-Desk Domain

Yuval Marom*
Monash University

Ingrid Zukerman**
Monash University

This article presents an investigation of corpus-based methods for the automation of help-desk e-mail responses. Specifically, we investigate this problem along two operational dimensions: (1) information-gathering technique, and (2) granularity of the information. We consider two information-gathering techniques (retrieval and prediction) applied to information represented at two levels of granularity (sentence-level and document-level). Document-level methods correspond to the reuse of an existing response e-mail to address new requests. Sentence-level methods correspond to applying extractive multi-document summarization techniques to collate units of information from more than one e-mail. Evaluation of the performance of the different methods shows that in combination they are able to successfully automate the generation of responses for a substantial portion of e-mail requests in our corpus. We also investigate a meta-selection process that learns to choose one method to address a new inquiry e-mail, thus providing a unified response automation solution.

1. Introduction

E-mail inquiries sent to help desks often "revolve around a small set of common questions and issues."[1] 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 low level of technical content, addressing, for example, inquiries sent to the wrong group, or requests containing insufficient detail about the customer's problem. Organizations and clients would benefit if an automated process was employed to deal with the easier problems, and the efforts of human operators were focused on difficult, atypical problems.

However, even the automation of responses to the "easy" problems is a difficult task. Although such inquiries revolve around a relatively small set of issues, specific

* Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia. Currently employed at Pacific Brands Services Group, Building 10, 658 Church St, Richmond, Victoria 3121, Australia. E-mail: yuvalmarom@gmail.com.
** Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia. E-mail: Ingrid.Zukerman@infotech.monash.edu.au.
1 http://customercare.telephonyonline.com/ar/telecom_next_generation_customer.

Submission received: 7 November 2007; revised submission received: 20 March 2009; accepted for publication: 3 June 2009.

circumstances can make each inquiry unique, and hence care must be taken to compose a response that does not confuse, irritate, or mislead the customer. It is therefore no surprise that early attempts at response automation were knowledge-driven (Barr and Tessler 1995; Watson 1997; Delic and Lahaix 1998). These systems were carefully designed to produce relevant and correct responses, but required significant human input and maintenance (Delic and Lahaix 1998).

In recent times, such knowledge-intensive approaches to content delivery have been largely superseded by data-intensive, statistical approaches. An outcome of the recent proliferation of statistical approaches, in particular in recommender systems and search engines, is that people have become accustomed to responses that are not precisely tailored to their queries. This indicates that help-desk customers may have also become more tolerant of inaccurate or incomplete automatically generated replies, provided these replies are still relevant to their problem, and so long as the customers can follow up with a request for human-generated responses if necessary. Despite this, to date, there has been little work on corpus-based approaches to help-desk response automation (Carmel, Shtalhaim, and Soffer 2000; Lapalme and Kosseim 2003; Bickel and Scheffer 2004; Malik, Subramaniam, and Kaushik 2007). A major factor limiting this work is the dearth of corpora—help-desk e-mails tend to be proprietary and are subject to privacy issues. Further, this application lacks the kind of benchmark data sets that are used in question-answering and text summarization.[2]

In this article, we report on our experiments with corpus-based techniques for the automation of help-desk responses. Our study is based on a large corpus of request–response e-mail dialogues between customers and operators at Hewlett-Packard. Observations from this corpus have led us to consider several methods that implement different types of corpus-based strategies. Specifically, we have investigated two types of methods (retrieval and prediction) applied at two levels of granularity (document and sentence). In this article, we present these methods and compare their performance. A key issue in the generation of help-desk responses is the ability to determine when an automatically generated response for a particular query can be sent to a user, and when the query should be passed to an operator. In Section 3, we employ method-specific, empirically determined, applicability thresholds to make this decision; in Section 6, we propose a meta-level process for selecting a response-generation method, which obviates the need for these thresholds.

The rest of the article is organized as follows. In the next section, we discuss properties of the help-desk domain. In Section 3, we describe our response-generation methods. An automatic evaluation of these methods is presented in Section 4, and a small user-based evaluation in Section 5. In Section 6, we describe the meta-level process which learns to select between the different methods, and evaluate its performance in Section 7. Related work is discussed in Section 8, and concluding remarks are presented in Section 9.

2. The Help-Desk Domain

The help-desk domain offers interesting challenges to response automation in that on one hand, responses are generalized to fit standard solutions, and on the other hand, responses are tailored to the initiating request in order to meet specific customer needs.

2 http://trec.nist.gov/pubs/trec15/t15_proceedings.html and http://www-nlpir.nist.gov/projects/duc/data.html.

There is an internal battery inside this ipaq which I believe has to be sent in, in order to have it replaced. The model is i3650, serial# 4G12DW36K9RK. The operating sys. is the version later than 3.1. Battery will not hold a charge. I need to know how much it would cost to send it in for this type of service.
The approximate cost for replacing the battery would be $120.00. Please call technical support at `888-phone-number`, options 3, 1 (enter 10 digit phone number). If you are a first time caller, your phone number won't be registered in the database. Enter your phone number twice and then wait for the routing center to put you through to a technician to set up a repair with HP Services Technical Support.

(a) Mixed specific/generic response

Do I need Compaq driver software for my armada 1500 docking station? This in order to be able to re-install win 98?
I would recommend to install the latest system rompaq, on the laptop and the docking station. Just select the model of computer.

(b) Specific response

Is there a way to disable the NAT firewall on the CP-2W so I don't get a private ip address through the wireless network?
Unfortunately, you have reached the incorrect eResponse queue for your unit. Your device is supported at the following link, or at `888-phone-number`.

(c) Generic response

Figure 1
Sample responses from our corpus: (a) mixed generic–specific, (b) specific, and (c) generic.

For example, the first sentence of the response in Figure 1(a) is tailored to the user's request, whereas the rest of the response is generic, and may be used when replying to other queries.[3] In addition to responses that contain such a mixture of specific and generic information, there are inquiries that warrant very specific or completely generic responses, as seen in Figures 1(b) and 1(c), respectively.

A distinctive feature of the help-desk domain is that help-desk e-mail responses contain a high level of repetition and redundancy. This may be attributed to commonalities in customer issues combined with the provision of in-house manuals to help-desk operators. These manuals connect particular topics with standard response templates, prescribe a particular presentation style, and even suggest specific responses to certain queries. For example, Figure 2 shows two rather different response e-mails which share a sentence (italicized). Thus, having access to these manuals would enable us to easily identify prescribed sentences. More importantly, it would enable us to determine the context in which these sentences are used, which in turn would allow us to postulate additional response sentences. An interesting avenue of investigation would involve adapting our approach to help-desk situations where such manuals are accessible.

3 The examples shown in this article are reproduced verbatim from the corpus (except for URLs and phone numbers which have been disguised by us), and some have user or operator errors.

If you are able to see the Internet then it sounds like it is working, you may want to get in touch with your IT department to see if you need to make any changes to your settings to get it to work. *Try performing a soft reset by pressing the stylus pen in the small hole on the bottom left hand side of the Ipaq and then release.*

I would recommend doing a soft reset by pressing the stylus pen in the small hole on the left hand side of the Ipaq and then release. Then charge the unit overnight to make sure it has been long enough and then see what happens. If the battery is not charging then the unit will need to be sent in for repair.

Figure 2
Sample responses that share a sentence.

Despite the high degree of repetition in help-desk responses, the specific issues raised by different customers imply that the responses sent to these customers contain varying degrees of overlap (rather than being identical). Hence, providing a response for a new request may involve reusing an existing response in its entirety, putting together parts of responses that match individual components of the request, or composing a completely new response. This suggests that different response-generation strategies may be suitable, depending on the content of the initiating request and how well it matches previous requests or responses. In our work, we focus on the first two of these situations, where either complete existing responses or parts of responses are reused to address a new request.

The example in Figure 1(b) illustrates a situation where specific words in the request (*docking station* and *install*) are also mentioned in the response. This situation suggests a response-automation approach that follows the document retrieval paradigm (Salton and McGill 1983), where a new request is matched with existing response documents (e-mails). However, specific words in the request do not always match a response well, and sometimes do not match a response at all, as demonstrated by the examples in Figures 1(a) and 1(c), respectively.

Sometimes requests match each other quite well, suggesting an approach where a new request is matched with an old one, and the corresponding response is reused. However, analysis of our corpus shows that this does not occur very often, because unlike response e-mails, request e-mails exhibit a high language variability: There are many customers who write these e-mails, and they differ in their background, level of expertise, and pattern of language usage. Further, there are many requests that raise multiple issues, hence matching a new request e-mail in its entirety is often not possible.

In situations where requests do not match existing responses or other requests, it may be possible instead to find *correlations* between requests and responses. For example, the generic portion of the response in Figure 1(a), and the entire response in Figure 1(c), may be repeated for many different kinds of requests. If repeated sufficiently, entire responses or parts of responses will be strongly correlated with particular combinations of request words, such as *send*, *battery* and *replace* in Figure 1(a), and *firewall*, *ip*, and *network* in Figure 1(c).

3. Response Generation Methods

The properties of the help-desk domain outlined in the previous section have motivated us to study the response-automation task along two dimensions. The first dimension pertains to the strategy applied to determine the information in a response, and the

second dimension pertains to the granularity of the information. We implemented two alternative strategies. The first is a retrieval strategy that attempts to match a new request with previous requests and responses, and the second is a prediction strategy that looks for correlations between requests and responses in order to predict a response for a new request. For both types of strategies, we considered two levels of granularity for a unit of information: document and sentence.

We implemented four methods according to the two alternatives for each dimension: **Document Retrieval**, **Document Prediction**, **Sentence Retrieval**, and **Sentence Prediction**. We also implemented a fifth method for addressing situations such as the example shown in Figure 1(a), which warrant a mixed response that has both a generic component and a component tailored to the user's request. This is a hybrid prediction–retrieval method implemented at the sentence level: **Sentence Prediction–Retrieval Hybrid**. These five methods are summarized in Table 2 (Section 3.3). The implementation of these methods relies on the judicious selection of thresholds for different aspects of the processes in question. In principle, machine learning techniques could be used to determine optimal threshold values. However, owing to practical considerations, we selected these values by trial and error. Table 3 shows the range of values we tried for these thresholds, and the values we selected (Section 3.3).

3.1 Document-Level Methods

A document-level method attempts to reuse an existing response document (e-mail) in its entirety. Unlike sentence-based methods that attempt to put together portions of different responses (Section 3.2), a document-level approach avoids issues pertaining to the coherence and completeness of a response, as a response composed by a help-desk operator is likely to be both coherent and complete. Therefore, if a particular request can be addressed with a single existing response document, then a document reuse approach would be preferred. An important capability of a response-generation system is to be able to determine when such an approach is appropriate, and when there is insufficient evidence to reuse a complete response document.

As stated herein, we studied two document-based methods: **Document Retrieval** and **Document Prediction**.

3.1.1 Document Retrieval (Doc-Ret). This method follows a traditional Information Retrieval paradigm (Salton and McGill 1983), where a query is represented by the content terms it contains, and the system retrieves from the corpus a set of documents that best match this query. In our case, the query is a new request e-mail to be addressed by the system, and we have considered three views of the documents in the corpus: (1) previous response e-mails, (2) previous request e-mails, or (3) previous request–response pairs. The first alternative corresponds to the more traditional view of retrieval as applied in question-answering tasks, where the terms in the question are matched to those in the answer documents. We consider the second alternative in order to address situations such as the example in Figure 1(c), where a request might not match a particular response, but it may match another request, yielding the response to that request. The third alternative addresses situations where a new request matches part of another request and part of its response.

We use cosine similarity (between a request e-mail and each document in the corpus) to determine a retrieval score, and pick the document with the highest score. The similarity is calculated using a bag-of-lemmas representation with TF.IDF (term-frequency-inverse-document frequency) weightings (Salton and McGill 1983), but in the

request-to-response option we use TF = 1, as it yields the best results. We posit that this happens because a response to a request does not necessarily contain multiple instances of request terms. Hence, what is important when matching a request to a response is the number of (significant) terms in common, rather than their frequency. In contrast, when matching a request to a request, or a request to a request–response pair, term frequency would be more indicative of the goodness of the match, as the document also has a request component.

We consider retrieval to be successful only if the similarity score is higher than an applicability threshold, which is currently set empirically (Table 3). If retrieval is successful, then the response associated with the retrieved document is reused to reply to the user's request.

We carried out a preliminary experiment in order to compare the three variants of the Doc-Ret method. The evaluation is performed by considering each request e-mail in turn, removing it and its response from the corpus, carrying out the retrieval process, and then comparing the retrieved response with the actual response (if there are several similar responses in the corpus, an appropriate response can still be retrieved). The results of this experiment are shown in Table 1. The first column shows which document retrieval variant is being evaluated. The second column shows the proportion of requests for which one or more documents were retrieved (using our applicability threshold). We see that matching on requests yields more retrieved documents than matching on responses, and that matching on request–response pairs yields even more retrieved documents. For the cases where retrieval took place, we used F-score (van Rijsbergen 1979; Salton and McGill 1983) to determine the similarity between the response from the top-ranked document and the real response (the formulas for F-score and for its contributing factors, recall and precision, appear in Section 4.2). The third column in Table 1 shows the proportion of requests for which this similarity is non-zero. Again, the third variant (matching on request–response pairs) retrieves the highest proportion of responses that bear some similarity to the real responses. The fourth column shows the average similarity between the top retrieved response and the real response for the cases where retrieval took place. Here too the third variant yields the best similarity score (0.52).

From this preliminary experiment it appears that the third document retrieval variant is superior. Hence, we use this variant as the Doc-Ret method in subsequent experiments.

3.1.2 Document Prediction (Doc-Pred). The Doc-Ret method may fail in situations where the presence or absence of some terms in the requests triggers a generic template response. For instance, in the example in Figure 1(c), given the terms *firewall* and *CP-2W*,

Table 1
Comparison between the three document retrieval variants.

Match type	Percent retrievals	Percent retrieved docs with sim > 0	Average similarity for retrieved docs
request to response	11%	11%	0.40
request to request	37%	26%	0.50
request to request–response	43%	32%	0.52

we would like to retrieve the generated response. However, the first Doc-Ret alternative would fail, as the response has no common terms with the request. The other two Doc-Ret alternatives would fail if different requests in the corpus mention different issues about these two terms, and thus no single request or request–response document in the corpus would yield a good match with a new request that mentions these terms.

In order to handle such cases, we offer a predictive approach, which is guided by *correlations* between terms, rather than matches. In principle we could look for direct correlations between request terms and response terms. However, since we have observed strong regularities in the responses at the document level, we decided to reduce the dimensionality of the problem by abstracting the response documents and then looking for correlations at this higher level. This approach did not seem profitable for request e-mails, which unlike responses, have a high language variability. Hence, we keep their representation at a low level of abstraction (bag-of-lemmas).

The idea behind the Doc-Pred method is similar to Bickel and Scheffer's (2004): Response documents are grouped into clusters, one of these clusters is predicted for a new request on the basis of the request's features, and the response that is most representative of the predicted cluster (closest to the centroid) is selected. In our case, the clustering is performed by the program Snob, which implements mixture modeling combined with model selection based on the Minimum Message Length (MML) criterion (Wallace and Boulton 1968; Wallace 2005). We chose this program because the number of clusters does not have to be specified in advance, and it returns a probabilistic interpretation for its clusters (this interpretation is used by the Sent-Pred method, Section 3.2.2). The input to Snob is a set of binary vectors—one vector per response document. The values of a vector correspond to the presence or absence of each (lemmatized) corpus word in the document in question (after removing stop-words and words with very low frequency).[4] The predictive model is a Decision Graph (Oliver 1993), which, like Snob, is based on the MML principle. The Decision Graph is trained on unigram and bigram lemmas in the request as input features,[5] and the identifier of the response cluster that contains the actual response for the request as the target feature. The model predicts which response cluster is most suitable for a given request, and returns the probability that this prediction is correct. This probability is our indicator of whether the Doc-Pred method can address a new request. As for the Doc-Ret method, an applicability threshold for this parameter is currently determined empirically (Table 3).

3.2 Sentence-Level Methods

The document-level methods presented in the previous section are designed to address situations where requests are sufficiently specific to strongly match a previous request or response e-mail (Doc-Ret), or requests contain terms that are predictive of complete template response e-mails (Doc-Pred).

4 We used a binary representation, rather than a representation based on TF.IDF scores, because important domain-related words, such as 'monitor' and 'network', are actually quite frequent. Thus, their low TF.IDF score may have an adverse influence on clustering performance. Nonetheless, in the future, it may be worth investigating a TF.IDF-based representation.

5 Significant bigrams are obtained using the N-gram statistics package **NSP** (Banerjee and Pedersen 2003), which offers statistical tests to decide whether to accept or reject the null hypothesis regarding a bigram (that it is not a collocation).

As discussed in Section 2, there are situations that cannot be addressed by a document-level approach, because requests only predict or match *portions* of responses. An alternative approach is to look for promising sentences from one or more previous responses, and collate them into a new response. This task can be cast as extractive multi-document summarization. Unlike a document reuse approach, sentence-level approaches need to consider issues of discourse coherence in order to ensure that the extracted combination of sentences is coherent or at least understandable. In our work, we gather sets of sentences, and assume (but do not employ) existing approaches for their organization (Goldstein et al. 2000; Barzilay, Elhadad, and McKeown 2001; Barzilay and McKeown 2005).

The appeal of a sentence-level approach is that it supports the generation of a "combination response" in situations where there is insufficient evidence for a single document containing a full response, but there is enough evidence for parts of responses. Although such a combined response is generally less satisfactory than a full response, the information included in it may address a user's problem or point the user in the right direction. As argued in the Introduction, when it comes to obtaining fast information on-line, this option may be preferable to having to wait for a human-generated response. In contrast, the document-level approach is an all-or-nothing approach: If there is insufficient evidence for a complete response, then no automated response is generated.

3.2.1 Sentence Retrieval (Sent-Ret). As we saw in Section 2 (Figure 1(b)), there are situations where the terms in response sentences match well the terms in request sentences. To address these situations, we consider the Sent-Ret method, which employs retrieval techniques as for the Doc-Ret method, but at the sentence level.

There are two main differences between Sent-Ret and Doc-Ret: (1) in Sent-Ret we look for response sentences that match individual request sentences, rather than entire documents; and (2) in Sent-Ret we perform recall-based retrieval, rather than retrieval based on cosine-similarity, where the request sentence is the reference document for the recalled terms. The second difference is a result of the first, as a good candidate for sentence retrieval contains terms that appear in a request sentence, but is also likely to contain additional terms that expand on the matching terms (Figure 1(b)). Thus, recall is calculated for each response sentence with respect to each request sentence as follows (since TF=1 yielded the best result for the request–response option in Doc-Ret, we also use it for Sent-Ret).

$$\textbf{recall} = \frac{\text{sum of TF.IDF of lemmas in request sentence \& response sentence}}{\text{sum of TF.IDF of lemmas in request sentence}} \quad (1)$$

We retain the response sentences whose recall exceeds an empirically determined applicability threshold (Table 3), and produce a response if at least one sentence was retained.

3.2.2 Sentence Prediction (Sent-Pred). As for the Doc-Pred method, the Sent-Pred method starts by abstracting the responses. It clusters response sentences using the same

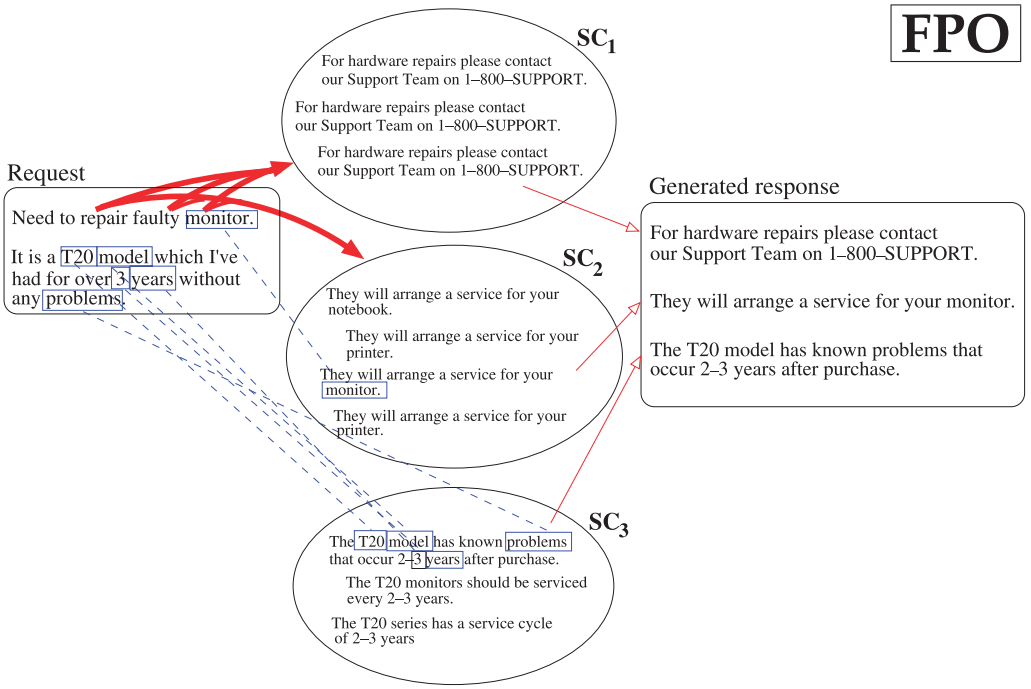


Figure 3
A fictitious example that demonstrates the Sent-Pred and Sent-Hybrid methods.

clustering program (Snob) and bag-of-lemmas representation as Doc-Pred.[6] Unlike the Doc-Pred method, where only a single response cluster is predicted (resulting in a single response document being selected), the Sent-Pred method may predict several promising **Sentence Clusters (SCs)**. A response is then composed by extracting sentences from the predicted SCs.

To illustrate these ideas, consider the fictitious example in Figure 3, which shows three small SCs (in practice SCs can have tens and even hundreds of sentences). The thick arrows correspond to high-confidence predictions, while the thin arrows correspond to sentence selections. The other components of the diagram demonstrate the workings of the Sent-Hybrid method (Section 3.2.3). In this example, three of the request terms, *repair*, *faulty* and *monitor*, result in a confident prediction of two SCs: SC_1 and SC_2. The sentences in SC_1 are identical, so we can arbitrarily select a sentence for inclusion in the generated response. The sentences in SC_2 are similar but not identical, hence we are less confident in arbitrarily selecting a sentence from SC_2, and may select more than one sentence (see the subsequent section, **Removing redundant sentences**).

We use a Support Vector Machine (SVM) with a Radial Basis Function kernel to predict SCs from users' requests.[7] A separate SVM is trained for each SC, with unigram and bigram lemmas in a request as input features, and a binary target feature specifying whether the SC contains a sentence from the response to this request. During the

6 For Sent-Pred we also experimented with grammatical and sentence-based syntactic features, such as number of syntactic phrases, grammatical mood, and grammatical person (Marom and Zukerman 2006), but the simple binary bag-of-lemmas representation yielded similar results.
7 We employed the LIBSVM package (Chang and Lin 2001).

prediction stage, the SVMs predict zero or more SCs for each request, as shown in Figure 3. We then apply the following steps.

1. Calculate the scores of the sentences in the predicted SCs.
2. Remove redundant sentences from **cohesive** SC; these are SCs which contain similar sentences.
3. Calculate the confidence of the generated response.

Calculating the score of a sentence. The score of each sentence s_j is calculated using the following formula.

$$\textbf{Score}(s_j) = \sum_{i=1}^{m} \Pr(SC_i) \times \Pr(s_j|SC_i) \tag{2}$$

where m is the number of SCs, $\Pr(s_j|SC_i)$ is the probability that s_j appears in SC_i (obtained from Snob), and $\Pr(SC_i)$ is approximated as follows.[8]

$$\Pr(SC_i) = \begin{cases} \text{Precision}(SC_i) & \text{if } SC_i \text{ is very } \textbf{cohesive} \text{ and predicted with } \textbf{high probability} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where

- $\text{Precision}(SC_i)$ is defined as follows.

$$\text{Precision}(SC_i) = \frac{\text{\# of times } SC_i \text{ was correctly predicted}}{\text{\# of times } SC_i \text{ was predicted}} \tag{4}$$

  This measure, which reflects the reliability of an SVM that was trained to predict SC_i, is obtained by performing 10-fold cross-validation on the performance of this SVM for the training data.

- An SC is **cohesive** if the sentences in it are similar to each other. This means that it is possible to obtain a sentence that represents the cluster adequately (this is not the case for an uncohesive SC). The cohesion of an SC is calculated as follows.

$$\text{Cohesion}(SC) = \frac{1}{N} \sum_{k=1}^{N} [\ \Pr(w_k \in SC) \leq \alpha\ \vee\ \Pr(w_k \in SC) \geq 1 - \alpha\ ] \tag{5}$$

  where N is the number of content lemmas under consideration; $\Pr(w_k \in SC)$ is the probability that (lemmatized) word w_k appears in

8 We tried several alternatives for representing $\Pr(SC_i)$. The best results were obtained with Equation (3).

the SC (this probability is obtained from the centroid[9]); and α is an empirically determined threshold that establishes an upper and lower bound for this probability (Table 3). For values of α close to zero, Equation (5) behaves like entropy, in the sense that it favors extreme probabilities. It implements the idea that a cohesive group of sentences should agree strongly on both the words that appear in these sentences and the words that are omitted. For example, the italicized sentences in Figure 2 belong to an SC with cohesion 0.93, whereas the opening response sentence in Figure 1(b) belongs to an SC that contains diverse sentences (about the Rompaq power management) and has cohesion 0.7. We employ an empirically determined SC-cohesion threshold (Table 3) to determine whether an SC is sufficiently cohesive for redundant sentences to be safely removed from it.

- A **high-probability** prediction is one where for the request in question, the SVM has predicted the SC with a probability that exceeds an empirically determined threshold (Table 3).

To illustrate the distinction between prediction probability and SVM reliability (precision), let us return to the request in Figure 3. SC_1 is predicted with high probability for this request, because SC_1 includes sentences from responses to many other requests that contain the words *repair* and *faulty*. The word "monitor" also contributes to this high prediction probability, but not as strongly as *repair* and *faulty*. This is because SC_1 includes sentences from responses whose requests mention different faulty products, for example, *monitor*, *printer*, or *notebook*. However, if there are more cases of faulty monitors in the corpus than other faulty products, then requests about repairing monitors will have a higher prediction probability than requests about repairing other products. In contrast to prediction probability, SVM reliability reflects its overall performance (on the training data), and is independent of particular requests. Thus, the SVM for SC_1 has a higher reliability than that for SC_3, because it is easier for an SVM to learn when SC_1 is appropriate (predominantly from the presence of the words *faulty* and *repair*).

In order to ensure the relevance of the generated replies, we have placed tight restrictions on prediction probability and cluster cohesion (Table 3), which cause the Sent-Pred method to often return partial responses.

Removing redundant sentences. After calculating the raw score of each sentence, we use a modified version of the Adaptive Greedy Algorithm by Filatova and Hatzivassiloglou (2004) to penalize redundant sentences in cohesive clusters. This is done by decrementing the score of a sentence that belongs to an SC for which there is a higher or equal scoring sentence (if there are several highest-scoring sentences, we retain one sentence as a reference sentence—i.e., its score is not decremented). Specifically, given a sentence s_k in cluster SC_l which contains a sentence with a higher or equal score, the contribution of SC_l to $\mathbf{Score}(s_k)$ $(= \Pr(SC_l) \times \Pr(s_k|SC_l))$ is subtracted from $\mathbf{Score}(s_k)$. After applying these penalties, we retain only the sentences whose adjusted score is greater than zero (for a highly cohesive cluster, typically only one sentence remains).

9 For each feature in the input (i.e., lemmatized words), the centroid of the cluster contains a frequency-based estimate of the probability that an item with this feature value appears in this cluster.

Calculating the confidence of an automated response. The calculation of sentence scores described previously determines which sentences should be included in an automatically generated response. In order to decide whether this response should be used, we need an overall measure of the confidence in it. Our confidence measure aggregates into a single number the values of the attributes used to assign a score to the individual sentences in a response, as follows.

$$\textbf{Confidence} = \frac{\text{\# of usable SCs}}{\text{\# of possible SCs}} \tag{6}$$

where

- usable SCs are those that satisfy the cohesion and prediction probability thresholds mentioned for sentence scoring, and also satisfy a further threshold for SC precision (Table 3).
- possible SCs are those that satisfy a minimum prediction probability threshold (Table 3).

This measure combines our confidence in the SCs selected to generate response sentences with the completeness of the resultant response. Confidence is represented by the thresholds employed to select suitable SCs; and completeness is represented by the ratio of the number of SCs that were deemed suitable, and the number of SCs that could possibly be used to generate a response. These are SCs whose prediction probability is greater than 0 (i.e., there is some evidence in the corpus for their use in the generation of a response sentence for the current request). We also use a minimal applicability threshold of 0 for the confidence measure (Table 3). This threshold reflects our notion that a partial response, even a response with one sentence, may still be useful.

3.2.3 Sentence Prediction–Retrieval Hybrid (Sent-Hybrid). As seen in cluster SC_2 in Figure 3, it is possible for an SC to be strongly predicted without it being sufficiently cohesive for a confident selection of a representative sentence. However, sometimes the ambiguity can be resolved through cues in the request. In this example, one of the sentences in SC_2 matches the request terms better than the other sentences, as it contains the word *monitor*. In order to capture such situations, we combine prediction confidence with retrieval score to guide sentence selection (as for Sent-Pred, we use a recall measure with TF = 1; the values for the thresholds mentioned herein appear in Table 3).

- For highly cohesive SCs predicted with high confidence, we select representative sentences as described in Section 3.2.2.
- For SCs with medium cohesion that were predicted with high confidence, we attempt to match the candidate response sentences with the request sentences. We can use a liberal (low) recall threshold here, because the high prediction confidence guarantees that the sentences in the cluster are suitable for the request, so there is no need for a conservative (high) recall threshold. The role of retrieval in this situation is to select the sentence whose content terms best match the request, regardless of how good is the match. For instance, in the example in Figure 3, the sentence in SC_2 that best matches the request only matches on one word (*monitor*), but this is sufficient to distinguish the winning sentence from the other sentences in SC_2.

- For uncohesive clusters or clusters predicted with low confidence, we can rely only on retrieval. Now we must use a more conservative recall threshold to ensure that only sentences that are a good match for the request sentences are included in the response. SC_3 in Figure 3 is an example of an SC for which there is insufficient evidence to form strong correlations between it and request terms. However, one of its sentences is a very good match for the second sentence in the request. In fact, all the content lemmas in that request sentence are matched, resulting in a perfect recall score of 1.0 (the non-matching words are stop words), which means that this response sentence is likely to be informative.

Once we have a set of candidate response sentences that satisfy the appropriate recall thresholds, we remove redundant sentences as follows. Redundant sentences are removed from cohesive clusters as described in Section 3.2.2; for SCs with medium cohesion, we retain the sentence with the highest recall;[10] and for uncohesive SCs, we retain all the sentences. The rationale for this policy is that the sentences in an SC with medium cohesion are sufficiently similar to each other, so the selection of more than one sentence may introduce redundancy. In contrast, sentences that belong to uncohesive SCs are deemed sufficiently dissimilar to each other, so we can select all the sentences that satisfy the recall criterion.

As for Sent-Pred, the Sent-Hybrid method produces a response if it can return at least one response sentence. This happens when (a) the confidence in the highly cohesive SCs exceeds an applicability threshold; or (b) the confidence in one of the SCs with medium or low cohesion exceeds an applicability threshold, and the number of sentences retrieved for this SC exceeds an applicability threshold. As for Sent-Pred, confidence is calculated using Equation (6). Both applicability thresholds (confidence and number of retrieved sentences) are set to 0 (Table 3).

3.3 Summary

The focus of our work is on the general applicability of the different response automation methods, rather than on comparing the performance of particular implementation techniques. Hence, throughout the course of this project, the different methods had minor implementational variations, which do not affect the overall insights of this research. Specifically, we used Decision Graphs (Oliver 1993) for Doc-Pred, and SVMs (Vapnik 1998) for Sent-Pred.[11] Additionally, we used unigrams for clustering documents and sentences, and unigrams and bigrams for predicting document clusters and sentence clusters (Sections 3.1.2 and 3.2.2). Because this variation was uniformly implemented for both approaches, it does not affect their relative performance. These methodological variations are summarized in Table 2.

As indicated at the beginning of this section, the implementation of these methods requires the selection of different thresholds, which are subjective and application dependent. Table 3 summarizes the thresholds required for the different methods, the

10 We also experimented with the retention of sentences with high F-scores and with different weights for recall and precision. Our results were insensitive to these variations.

11 This change of technique had a practical motivation: As seen in Section 3.2.2, we generated many predictive models for sentence clusters. This process could be automated with SVMs, but would have to be done manually if Decision Graphs had been used.

Table 2
Summary of the response generation methods.

Method	Implementation	Features
Doc-Ret	Cosine similarity	bag of lemmas (TF.IDF)
Doc-Pred	Clustering: Snob Classification: Decision Graphs	bag of lemmas (binary) lemma unigrams and bigrams (binary)
Sent-Ret	Recall	bag of lemmas (TF.IDF)
Sent-Pred	Clustering: Snob Classification: SVMs	bag of lemmas (binary) lemma unigrams and bigrams (binary)
Sent-Hybrid	Combine sentence prediction with sentence retrieval	

range of values we considered, and the values we selected. The applicability thresholds are boldfaced, and those learned by the meta learning process (Section 6) are indicated in the rightmost column (Sent-Ret is not considered by this process owing to its poor performance, Section 4).

Table 3
Thresholds for the different response generation methods.

Method	Threshold	Range tried	Selected value	Learned
Doc-Ret	**cosine similarity score**	0.1–0.7	0.2	YES
Doc-Pred	**prediction probability**	0.6–0.9	0.7	YES
Sent-Ret	**recall score**	0.4–0.9	0.6	N/A
Sent-Pred	**confidence**		0	YES
	SC inclusion (Equation (3))			
	SC prediction probability	0.1–0.9	0.5	
	SC cohesion	0.7–1.0	0.9	
	probability of a lemma in SC (α, Equation (5))	0.01–0.05	0.01	
	confidence (Equation (6))			
	SC precision	0.7–0.9	0.7	
	SC minimum prediction probability	0.1–0.2	0.1	
Sent-Hybrid	**confidence**		0	YES
	number of sentences		0	YES
	SC inclusion			
	SC prediction probability	0.1–0.9	0.5	
	SC high cohesion	0.7–1.0	0.95	
	SC medium cohesion	0.6–0.8	0.75	
	sentence recall score			
	liberal	0.2–0.4	0.4	
	conservative	0.4–0.9	0.6	
	confidence (Equation (6))			
	SC precision	0.7–0.9	0.7	
	SC minimum prediction probability	0.1–0.2	0.1	

4. Automatic Evaluation of Individual Methods

In this section, we offer a comparative evaluation of the response automation methods presented in Section 3, where we measure the ability of the different methods to address the requests in the corpus. We first describe the data used in our experiments, followed by the experimental set-up and results.

4.1 The Corpus

Our initial corpus consisted of 30,000 e-mail dialogues between customers and help-desk operators at Hewlett-Packard. The 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. To focus our work on simple dialogues, we extracted a sub-corpus that satisfies two conditions:

1. The dialogues contain exactly two turns: a request e-mail followed by a response e-mail. These dialogues represent situations where a request can be resolved with a single response.
2. The response e-mails are reasonably concise (15 lines at most). This restriction is based on the observation that longer responses are quite complex—they often address multiple issues or have a strong temporal structure (i.e., a sequence of steps).

The resultant sub-corpus consisted of 6,659 dialogues, which deal with a wide range of topics. We were hoping to account for significant differences between groups of dialogues on the basis of their topic. In addition, there was a practical motivation to break up this large sub-corpus into smaller chunks for ease of handling. We therefore applied Snob to automatically cluster the sub-corpus into separate topic-based data sets. The clustering, which was done using as input the lemmas in the subject line of the users' e-mails, produced 51 data sets, some of which were quite small (11 data sets had less than 25 dialogues). Because Snob returns the significant terms in each cluster, we merged the smaller data sets manually according to these terms—a process that yielded 15 data sets in total, each data set containing between 135 and 1236 e-mail dialogues. Owing to the time limitations of the project, the procedures described in this article were applied to 8 of the 15 data sets, which contain a total of 4,904 dialogues (73.6% of the sub-corpus, and 16.3% of the original corpus). These data sets, which were chosen on the basis of their coverage of the corpus, are described in Table 4,[12] together with a qualitative overview of our results, which are discussed in Section 4.3.

It is worth noting that there may be factors other than topic that distinguish between dialogues, and cause differences in the performance of response generation methods for different types of dialogues. However, these factors are not readily apparent upon initial analysis. Topic-based clustering, which is readily apparent, is a reasonable starting point for distinguishing between different data sets. The analysis presented in Section 4.3 considers other features that characterize the data sets and the behavior of the response generation methods.

12 The topics of the omitted data sets were: servers, laptops specializing in EVO notebooks, desktops, and miscellaneous.

Table 4
Details of the data sets included in our experiments, and overview of results per data set.

Data set #	Topic	Sub-category	# of dialogues	Best method Retrieval	(coverage/ performance) Prediction
1	hand helds	general	268	Doc-Ret (lo/lo)	NONE
2	hand helds	DG models	1,160	Doc-Ret (med/lo)	SENTENCE (med/hi)
3	product replacement		1,236	Doc-Ret (lo/hi)	ALL (hi/hi)
4	laptops	Armada models	561	Doc-Ret (med/lo)	NONE
5	laptops	general	305	Doc-Ret (med/lo)	SENTENCE (lo/hi)
6	laptops	merged (7 clusters)	632	Doc-Ret (med/lo)	NONE
7	desktops	merged (7 clusters)	389	Doc-Ret (med/lo)	Sent-Pred (lo/hi)
8	misc.	merged (10 clusters)	353	Doc-Ret (med/lo)	Sent-Hybrid (med/lo)
	TOTAL		4,904		

4.2 Experimental Set-Up

Our experimental set-up is designed to evaluate the ability of the different response-generation methods to address unseen request e-mails. In particular, we want to determine the applicability of our methods to different situations, namely, whether different requests are addressed only by some methods, or whether there is a significant overlap between the methods.

Our evaluation is performed by measuring the quality of the generated responses. Quality is a subjective measure, which is best judged by the users of the system (i.e., the help-desk customers or operators). In Section 5, we discuss the difficulties associated with such user studies, and describe a human-based evaluation we conducted for a small subset of the responses generated by our system (Marom and Zukerman 2007b). However, our more comprehensive evaluation is an automatic one that treats the responses generated by the help-desk operators as model responses, and performs text-based comparisons between the model responses and the automatically generated ones.

We employ 10-fold cross-validation, where we split each data set in the corpus into 10 test sets, each comprising 10% of the e-mail dialogues; the remaining 90% of the dialogues constitute the training set. For each of the cross-validation folds, the responses generated for the requests in the test split are compared against the actual responses generated by help-desk operators for these requests. Although this method of assessment is less informative than human-based evaluations, it enables us to evaluate the performance of our system with substantial amounts of data, and produce representative results for a large corpus such as ours.

We use two measures from Information Retrieval to determine the quality of an automatically generated response: **precision** and **F-score** (van Rijsbergen 1979; Salton and McGill 1983). Precision measures how much of the information in an automatically generated response is correct (appears in the model response), and F-score measures the overall similarity between the automatically generated response and the model response. F-score is the harmonic mean of precision and **recall**, which measures how much of the information in the model response appears in the generated response. We consider precision separately because it does not penalize missing

information, enabling us to better assess our sentence-based methods. Precision, recall, and F-score are calculated as follows using a word-by-word comparison (stop-words are excluded).[13]

$$\text{Precision} = \frac{\text{\# words in both model response and automated response}}{\text{\# of words in automated response}} \tag{7}$$

$$\text{Recall} = \frac{\text{\# words in both model response and automated response}}{\text{\# of words in model response}} \tag{8}$$

$$\text{F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$

These measures are applied to responses generated after the thresholds in Table 3 have been used to determine the applicability or coverage of each response-generation method. Recall that the sentence-based methods can generate partial responses, many of which contain only one obvious and non-informative sentence, such as *Thank you for contacting HP* and *Thank You, Mike, HP eServices*. We have manually excluded such responses from the calculation of coverage, in order to prevent these responses from artificially improving this metric for the sentence-based methods. This was done by visually inspecting the sentence clusters created by Snob for these methods, and removing clusters composed of non-informative sentences such as the above (between four and six clusters were removed from each data set in this manner). Once a response is deemed to cover a request, then the full response (including these sentences) is used to calculate its quality. This has a small impact on the results of our evaluation, as a typical response includes two such sentences (opening and closing), and the average length of a response is 8.11 sentences.

4.3 Results

Figure 4 shows the coverage, average precision, and average F-score of each response-generation method per data set, where the averages are computed only for requests that are covered by the method in question. For example, the average precision of the Doc-Ret method for data set no. 2, 0.39, is calculated over the 59% of the data set that was covered by this method. Table 4 shows data set descriptions and sizes, together with an overview of the best retrieval-based and prediction-based method for each data set (SENTENCE refers to both Sent-Pred and Sent-Hybrid). The best method was selected manually on the basis of the results in Figure 4. To this effect, we considered the methods whose coverage exceeds some minimum (e.g., 10%), and chose the method(s) which could adequately answer the largest number of queries in the data set (based on coverage combined with F-score and precision). Table 5 presents the coverage and **unique/best** coverage of each method (the percentage of queries covered *only* by this method or for which this method produces a better reply than other methods), and the average and standard deviation of the precision and F-score obtained by each method (calculated over the requests that are covered).

13 We also employed sequence-based measures using the ROUGE tool set (Lin and Hovy 2003), with similar results to those obtained with the word-by-word measures.

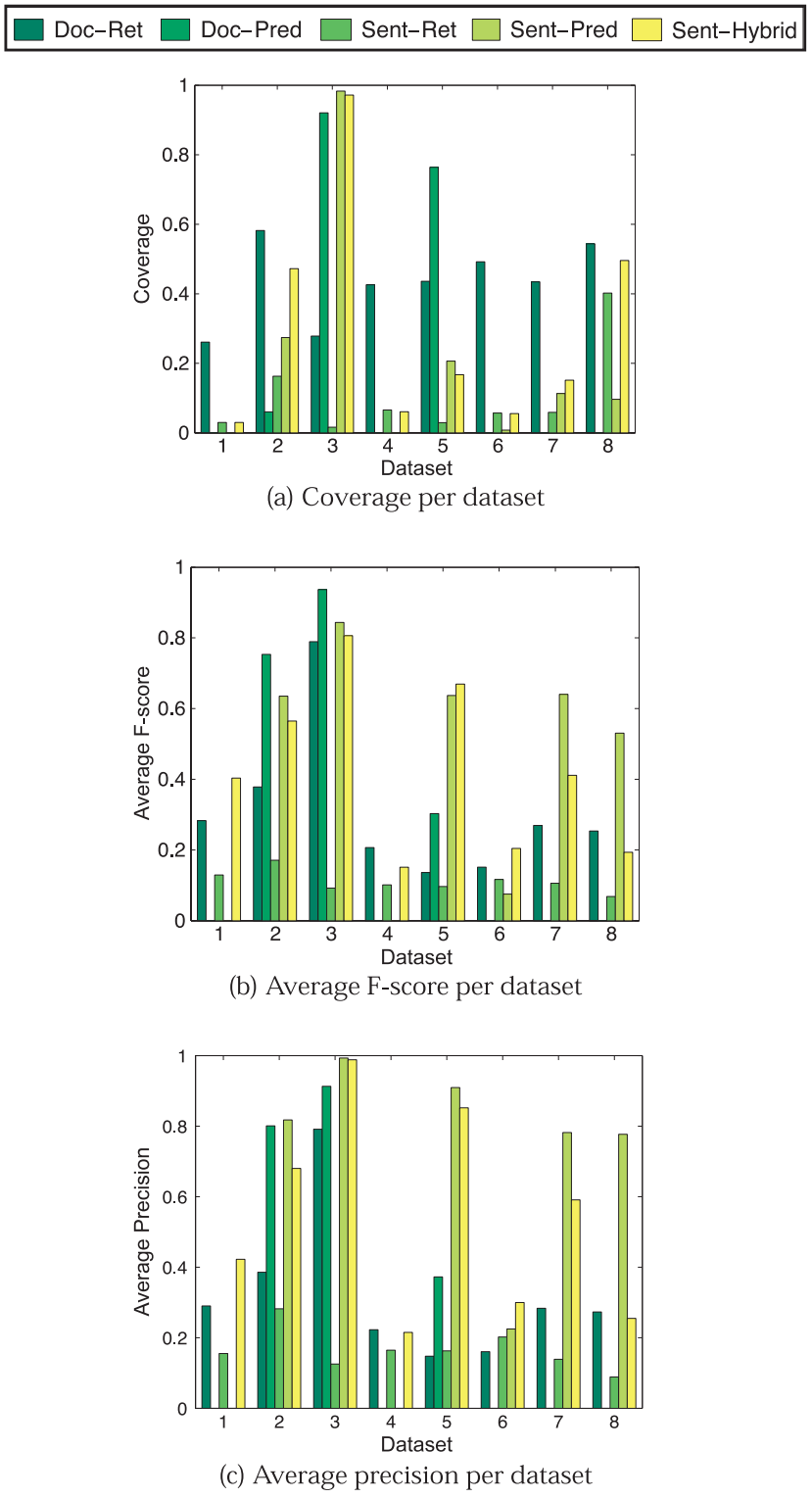


Figure 4
Performance of the different methods for each data set: (a) coverage, (b) F-score, and (c) precision.

Table 5
Coverage, uniqueness, precision, and F-score for the response generation methods.

Method	Coverage	Unique or best	Avg. (St dev.) Precision	F-score
Doc-Ret	43%	22%	0.37 (0.34)	0.35 (0.33)
Doc-Pred	29%	3%	0.82 (0.21)	0.82 (0.24)
Sent-Ret	9%	0%	0.19 (0.19)	0.12 (0.11)
Sent-Pred	34%	5%	0.94 (0.13)	0.78 (0.18)
Sent-Hybrid	43%	10%	0.81 (0.29)	0.66 (0.25)
Combined	72%		0.80 (0.25)	0.50 (0.33)

As seen in Figure 4 and Tables 4 and 5, there is great variability in coverage and performance of the different methods for the different data sets (this result was confirmed with an ANOVA statistical test). Our results support the following specific observations.

- All prediction methods perform well for data set no. 3 (product replacement). The vast majority of the requests in this data set, which comprises 18% of the corpus, ask for a return shipping label to be mailed to the customer, so that he or she can return a faulty product. Although these requests often contain detailed product descriptions, the responses rarely refer to the actual products, and often contain the generic response shown in Figure 5. Thus, the prediction methods are well suited for this data set, as the mention of a shipping label is a strong predictor of a generic response, either in its entirety (as done by Doc-Pred) or broken up into individual sentences (as done by Sent-Pred or Sent-Hybrid). The generation of complete responses by Doc-Pred explains its slightly higher F-score but lower precision compared to Sent-Pred, as a complete response may include sentences that are not appropriate for the situation at hand. Also note that Doc-Ret has a much lower coverage than the prediction methods for data set no. 3, because each request has precise information about the actual product, so a new request can neither match an old request (about a different product) nor can it match the generic response.
- No method performs well for data sets no. 1 (hand-helds general), 4 (laptops general), and 6 (laptops merged). Although Doc-Ret has some applicability to these data sets, it has low performance for all of them. This indicates that these data sets do not contain sufficient recurring cases for Doc-Ret to perform well, nor do they contain sufficient request–response pairs that support the generation of predictive patterns.
- Sent-Ret performs poorly on all data sets. We postulate that this happens because the important terms in a request and a response are spread across

> 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.

Figure 5
A sample response from the product replacement data set (data set no. 3).

several sentences in the respective documents. Hence, the match between individual response and request sentences paints only a partial (and inaccurate) picture, and the aggregation of matching response sentences does not add up to an appropriate response.

- Overall, there is a tension between coverage and performance, whereby higher coverage yields lower performance, and lower coverage results in higher performance. This tension can be observed for the sentence-based prediction methods with respect to data sets no. 2 (hand-held DG models), 5 (laptops general), 7 (desktops merged), and 8 (miscellaneous merged); for Doc-Pred with respect to data sets no. 2 and 5 (Doc-Pred is not applicable to data sets no. 7 and 8); and for Doc-Ret for most data sets (nos. 1 and 4–8). This suggests that our applicability thresholds may need further adjustments in order to strike a better balance between coverage and performance (Table 3).

Let us now examine our results separately for each of the four response-generation methods that perform well.

Doc-Ret. As seen in Table 5, Doc-Ret uniquely addresses 22% of the requests. However, the performance of this method is quite variable (high standard deviation), which may be due to an overly liberal setting of the applicability threshold (which results in both poor and good responses being generated, hence the high variability). Nevertheless, there are some cases where this method uniquely addresses requests quite well. This happens in situations such as that in Figure 1(b), where the initiating request is sufficiently similar to other requests with the same response. In contrast, Doc-Ret would not work well for requests such as that in Figure 1(a), which is quite detailed and specific, and hence unlikely to match any other request–response document.

Doc-Pred. Only about a tenth of the requests covered by Doc-Pred are uniquely addressed by this method, but the generated responses are of a fairly high quality, with an average precision and F-score of 0.82. As indicated previously, the higher F-score and lower precision of Doc-Pred (compared to Sent-Pred) may be explained by the fact that the complete responses produced by the Doc-Pred method sometimes contain specific sentences that are inappropriate for the current situation. The rather large standard deviation for F-score and precision suggests that Doc-Pred exhibits a somewhat inconsistent behavior. This may be explained by Figure 4, which shows that Doc-Pred performs very well on data set no. 3 (product replacement), but not so well on the others (Doc-Pred also has good F-score and precision scores for data set no. 2, but poor coverage).

Sent-Pred. In contrast to Doc-Pred, Sent-Pred can find regularities at the sub-document level, and generate partial responses, which typically omit inappropriate sentences that may be included by Doc-Pred. As a result, the responses generated by Sent-Pred have a consistently high precision (average 0.94, standard deviation 0.13), but this can be at the expense of recall, which explains the lower F-score (compared to Doc-Pred). Overall, the Sent-Pred method outperforms the other methods in 5% of the cases, where it either uniquely addresses requests, or produces responses with a higher F-score than those generated by other methods. As an example of situations where Sent-Pred outperforms all other methods, consider Figure 1(a), where Sent-Pred outputs the response shown

FPO

My screen is coming up reversed (mirrored). There must be something loose electronically because if I put the stylus in it's hole and move it back and forth, I can get the screen to display properly momentarily. Please advise where to send for repairs.

To get the iPAQ *serviced*, you can call `1-800-phone-number`, options 3, 1 (enter a 10 digit phone number), 2. Enter your phone number twice and then wait for the routing center to put you through to a technician with Technical Support. They can get the unit picked up and brought to our service center.

To get the iPAQ *repaired (battery, stylus lock and screen)*, please call `1-800-phone-number`, options 3, 1 (enter a 10 digit phone number), 2. Enter your phone number twice and then wait for the routing center to put you through to a technician with Technical Support. They can get the unit picked up and brought to our service center.

Figure 6
Example showing an appropriate response generated by the Sent-Hybrid method (bottom) that differs from the model response.

in this example, but without the first sentence. In other words, the generic portion of the response is confidently produced, and the specific portion is left out due to insufficient evidence. This example shows the benefit of a partial response: Although the response does not actually answer the user's specific question (which would be difficult to automate due to the complex nature of the request), it can potentially save the user valuable time by referring him or her to the appropriate repair service.

Sent-Hybrid. The Sent-Hybrid method extends the Sent-Pred method by performing sentence retrieval. Sent-Hybrid's higher coverage is achieved by the retrieval component, which disambiguates between groups of candidate sentences, thus enabling more sentences to be included in a generated response. However, this is at the expense of precision. Although retrieval selects sentences that match closely a given request, these sentences can differ from the "selections" made by a human operator in the model response. Precision (and hence F-score) penalizes such sentences, even when they are more appropriate than those in the model response. For instance, consider the example at the top of Figure 6. The response is quite generic, and is used almost identically for several other requests. The Sent-Hybrid method produces a very similar response, shown in the text at the bottom of Figure 6. Only the first sentence differs from the first sentence in the model response (the different parts have been italicized). The sentence selected by the Sent-Hybrid method, which matches more request words than the first sentence in the model response, was chosen from a sentence cluster with medium cohesion (Section 3.2.3), which contains sentences that describe different reasons for setting up a repair (the matching word is *screen*). The rather high standard deviation in the precision (and hence F-score) for Sent-Hybrid may be due to these kinds of situations. Nonetheless, this method outperforms the other methods in about 10% of the cases, where it either uniquely addresses requests, or produces responses with a higher F-score than those generated by other methods.

All the methods combined. The bottom row of Table 5 shows that all the methods together have a coverage of 72%, which means that at least one of the methods can produce a non-empty and non-trivial response for 72% of the requests. The combined F-score and precision averages are calculated on the basis of the best-performing method for each request. At first glance, using the best method may appear too lenient,

as in practice, we cannot always automatically select this method in advance. However, these averages also suffer from the fact that in many cases only the Doc-Ret method is applicable, but its performance is poor. As mentioned previously, this tension between coverage and performance may be attributed to our empirically determined applicability thresholds (Section 3).

4.4 Summary

We have investigated the suitability of different response generation methods for the help-desk task. These methods, which vary significantly in their approach to response automation, cover a wide range of situations that arise in a help-desk corpus.

Ideally, we would like to explain our results in terms of features of the data sets, so that users of our system can select a single best response-generation method on the basis of these features. However, with the exception of data set no. 3, no clear set of features presents itself to support such a selection. Further, as seen in Table 4, superficial features of the data sets, such as topic and size, are not sufficient to characterize the applicability and performance of the different methods. These results indicate that (1) there are deeper features of data sets that must be considered in order to select a single suitable technique; or (2) the selection of a technique does not depend on features of the data set itself, but on the spread of situations in the data set. That is, the applicability and performance of a response-generation method depends on the specifics of the situation, and different data sets contain a different spread of situations. This second conjecture is supported by the results in Figure 4 and Table 5, which indicate that different methods have pockets of unique applicability in each data set. Of course, this still begs the question of why different data sets have different spreads of situations. Unfortunately, we do not have an answer to this question, and circumvent it by using the meta-learning technique described in Section 6, which has the added benefit of obviating the problem of selecting applicability thresholds.

However, if one wishes to select a response-generation method without meta-learning, the following considerations can be applied.

- Predictive methods are suitable for situations where there are relatively few responses for many, varied requests. That is, the responses generalize common answers to a variety of problems. Specifically, Doc-Pred is suitable when this observation applies to complete answers, as is the case for data set no. 3, while sentence prediction methods are appropriate when parts of replies generalize common answers to a variety of problems. Because the sentence-based prediction methods outperform Doc-Pred for all data sets except # 3 (where the three prediction methods perform similarly), we recommend the sentence-based methods. Among these, we prefer Sent-Pred due to its high and consistent precision and F-score, pending a further investigation of Sent-Hybrid.
- Doc-Ret applies to situations where there are specific problems which warrant a specific complete answer.

5. User-Based Evaluation of Individual Methods

The size of our corpus necessitates an automatic evaluation in order to produce meaningful results, especially because we are comparing several methods under a number of

experimental settings. Although our automatic evaluation has yielded useful insights, it has two main limitations.

- As we saw in Section 4.3 (Figure 6), appropriate responses are sometimes penalized when they do not match precisely the model response. However, it is often the case that there is not one single appropriate response to a query, and even a help-desk operator may respond to the same question in different ways on different occasions.
- The relationship between the results obtained by the automatic evaluation of the responses generated by our system and people's assessments of these responses is unclear, in particular for partial responses.

These limitations reinforce the notion that automated responses should be assessed on their own merit, rather than with respect to some model response.

In Marom and Zukerman (2007a) we identified several systems that resemble ours in that they provide answers to queries. These systems addressed the evaluation issue as follows.

- Only qualitative observations of the responses were reported (no formal evaluation was performed) (Lapalme and Kosseim 2003; Roy and Subramaniam 2006).
- Only an automatic evaluation was performed, which relied on having model responses (Berger and Mittal 2000; Berger et al. 2000).
- A user study was performed, but it was either very small compared to the corpus (Carmel, Shtalhaim, and Soffer 2000; Jijkoun and de Rijke 2005), or the corpus itself was significantly smaller than ours (Feng et al. 2006; Leuski et al. 2006). The representativeness of the sample size was not discussed in any of these studies.

There are significant practical difficulties associated with conducting the user studies needed to produce meaningful results for our system. Firstly, the size of our corpus and the number of parameters and settings that we need to test mean that in order for a user study to be representative, a fairly large sample involving several hundreds of request–response pairs would have to be used. Further, user-based evaluations of the output produced by our system require the subjects to read relatively long request–response e-mails, which quickly becomes tedious.

In order to address these limitations in a practical way, we conducted a small user study where we asked four judges (graduate students from the Faculty of Information Technology at Monash University) to assess the responses generated by our system (Marom and Zukerman 2007a). Our judges were instructed to position themselves as help-desk customers who know that they are receiving an automated response, and that such a response is likely to arrive quicker than a response composed by an operator. Our user study assessed the response-generation methods from the following perspectives, which yield information that is beyond the F-score and precision measures obtained in the automatic evaluation.

- **Informativeness:** Is there anything useful in the response that would make it a good automatic response, given that otherwise the customer has

to wait for a human-generated response? We used a scale from 0 to 3, where 0 corresponds to "not at all informative" and 3 corresponds to "very informative."

- **Missing information:** Is any crucial information item missing? Y/N.
- **Misleading information:** Is there any misleading information? Y/N. We asked the judges to consider only information that might misguide the customer, and ignore information that is so irrelevant that it would be ignored by a customer who knows that the response is automated (for example, receiving an answer for a printer, when the request was for a laptop).
- **Compare to model response:** How does the generated response compare with the model response? Worse/Same/Better. This is a summary question that rates a "customer's" overall impression of a response.

5.1 Experimental Set-Up

We had two specific goals for this evaluation. First, we wanted to compare document-level versus sentence-level methods. Second, we wanted to evaluate cases where only the sentence-level methods can produce a response, and establish whether such responses, which are often partial, provide a good alternative to a non-response. We therefore presented two evaluation sets to each judge.

1. The first set contained responses generated by Doc-Pred and Sent-Hybrid. These two methods obtained similar precision values in the automatic evaluation (Table 5), so we wanted to compare how they would fare with our judges.
2. The second set contained responses generated by Sent-Pred and Sent-Hybrid for requests for which Doc-Pred could not produce a response. The added benefit of this evaluation set is that it enables us to examine the individual contribution of the sentence retrieval component.

Each evaluation set contained 80 cases, randomly selected from the corpus in proportion to the size of each data set, where a case contained a request e-mail, the model response, and the responses generated by the two methods being compared. Each judge was given 20 of these cases, and was asked to assess the generated responses on the four criteria listed previously.[14]

We maximized the coverage of this study by allocating different cases to each judge, thus avoiding a situation where a particularly good or bad set of cases is evaluated by all judges. In addition, we tried to ensure that the sets of cases shown to the judges were of similar quality, so that the judges' assessments would be comparable. Because the judges do not evaluate the same cases, we could not employ standard inter-tagger agreement measures (Carletta 1996). However, it is still necessary to have

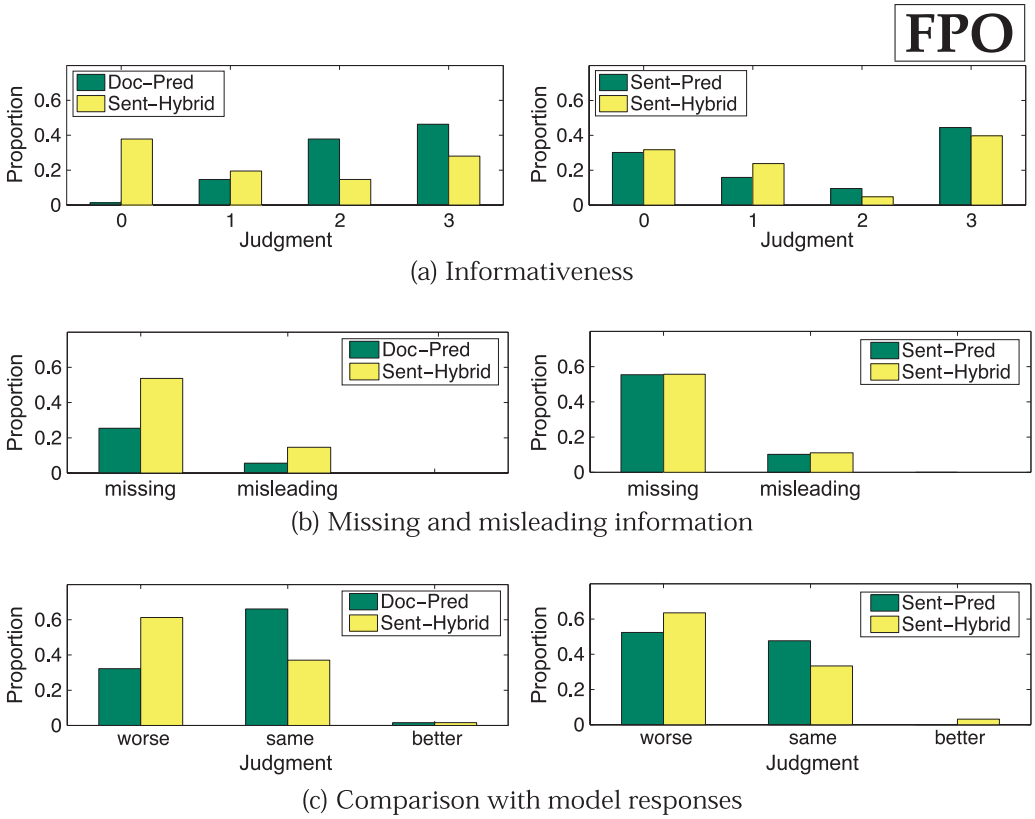
14 We asked the judges to leave a question unanswered if they felt they did not have the technical knowledge to make a judgement, but this did not occur.

some measure of agreement, and control for bias from specific judges or specific cases. This was done by performing pairwise significance testing, treating the data from two judges as independent samples (we used the Wilcoxon rank sum test for equal medians). We conducted this significance test separately for each method and each of the four criteria, and eliminated the data from a particular judge if he or she had a significant disagreement with other judges. This happened with one of the judges, who was significantly more lenient than the others on the Sent-Pred method for the first, second and fourth criteria, and with another judge, who was significantly more stringent on the Sent-Hybrid method for the third criterion.

5.2 Results

Figure 7 shows the results for the four criteria. The left-hand side of the figure pertains to the first evaluation set, and the right-hand side to the second set.

The left-hand side of Figure 7(a) shows that when both Doc-Pred and Sent-Hybrid are applicable, the former is generally preferred, rarely receiving a zero informativeness judgment. Because the two methods are evaluated for the same set of cases, we can perform a paired significance test for differences between them. Using a Wilcoxon signed rank test for a zero median difference, we obtain a p-value $\ll 0.01$, indicating that the differences in judgments between the two methods are statistically significant. The



(a) Informativeness

(b) Missing and misleading information

(c) Comparison with model responses

Figure 7
Results of the human study for the evaluation of generated responses.

right-hand side of Figure 7(a), which compares the two sentence-based methods, shows that there do not appear to be significant differences between our subjects' assessment of these methods. This result is confirmed by the paired significance test, which produces a p-value of 0.13.

As shown in Figure 7(b), the results for the missing-information and misleading-information criteria also favor the Doc-Pred method. The responses produced by Doc-Pred were judged to have significantly less missing information than those generated by Sent-Hybrid (the paired significance test produces a p-value $\ll 0.01$). The responses produced by Doc-Pred were also judged to have less misleading information than those generated by Sent-Hybrid, but the paired differences between the two methods are not statistically significant (the p-value is 0.125). The second evaluation set was judged to have missing information in approximately 55% of the cases for both sentence-level methods (the p-value is 0.11, indicating an insignificant difference between these methods). This high proportion of missing information is in line with the relatively low F-scores obtained in the automatic evaluation (Table 5), as missing information results in low recall and hence a lower F-score. The results for the misleading-information criterion also indicate no significant difference between the sentence-level methods (the p-value is 1). The low proportion of misleading information is in line with the high precision values obtained in the automatic evaluation (Table 5)—whereas responses with a high precision may be incomplete, they generally contain correct information.

The left-hand side of Figure 7(c) shows that Doc-Pred receives more "same" than "worse" judgments, although the opposite is true for Sent-Hybrid, and that both Doc-Pred and Sent-Hybrid receive a small proportion of "better" judgments. The paired significance test produces a p-value $\ll 0.01$, confirming that these differences are significant. The right-hand side of Figure 7(c) shows smaller differences between Sent-Pred and Sent-Hybrid, and indeed the p-value for the paired differences is 0.27. Notice that Sent-Pred does not receive any "better" judgements, whereas Sent-Hybrid does.

5.3 Summary

The results of this study show that responses provided by document-level methods were preferred to responses provided by sentence-level methods, but when document-level methods cannot be used, the sentence-level methods provide a good alternative. Additionally, although our trial subjects showed a slight preference for the output produced by the Sent-Hybrid method compared to Sent-Pred, this preference was not statistically significant.

Although these results confirm those obtained by the automatic evaluation (Section 4), the result regarding the Sent-Hybrid method is somewhat disappointing. This is because we hoped that our trial subjects would prefer Sent-Hybrid to Sent-Pred, as the former is designed to better tailor a response to a request. However, we cannot determine from this result whether indeed there is no difference between the sentence-based methods, or whether such a difference simply could not be observed from our test sample of at most 80 cases, which constitutes 1.8% of the corpus used in our automatic evaluation (as indicated previously, it would be quite difficult to conduct user studies with a much larger data set).

This outcome reinforces the previously mentioned problems associated with conducting meaningful user studies for a large corpus such as ours. These problems are exacerbated by our proportional data-selection policy, which is necessary to make the

test set representative of the corpus, but increases the difficulty of drawing specific conclusions, for example, determining whether a particular method is favored for specific data sets.

6. Meta-Learning

In Section 4, we employed empirically determined applicability thresholds to circumscribe the coverage of the different response-generation methods. However, as shown by our results, these thresholds were sometimes sub-optimal. In this section, we describe a meta-level process which can automatically select a response-generation method to address a new request without using such thresholds.

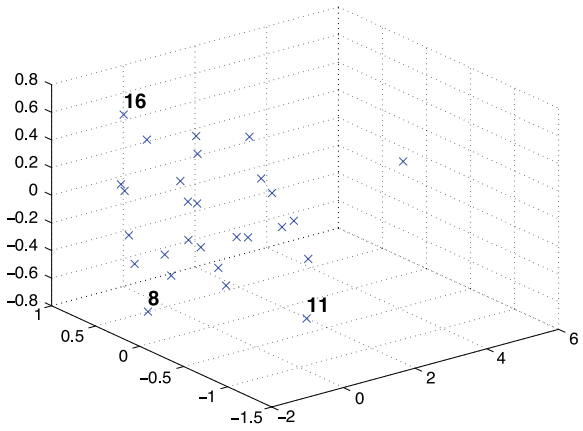
A common way to combine different models consists of selecting the model that is most confident regarding its decision (Burke 2002). However, in our case, the individual confidence (applicability) measures employed by our response-generation methods are not comparable (e.g., the retrieval score in Doc-Ret is different in nature from the prediction probability in Doc-Pred). Hence, prior to selecting the most confident method, we need to find a way to compare the different measures of confidence. Because the performances of the different methods are comparable, we do this by establishing a link between confidence and performance. In other words, our meta-level process learns to predict the performance of the different methods from their confidence levels on the basis of previous experience. These predictions enable our system to recommend a particular method for handling a new (unseen) request (Marom, Zukerman, and Japkowicz 2007).

Following Lekakos and Giaglis (2007), one approach for achieving this objective consists of applying supervised learning, where a winning method is selected for each case in the training set, all the training cases are labeled accordingly, and then the system is trained to predict a winner for unseen cases. However, in our situation, there is not always one single winner (two methods can perform similarly well for a given request), and there are different ways to pick winners (for example, based on F-score or precision). Therefore, such an approach would require the utilization of subjective heuristics for creating labels, which would significantly influence what is being learned. Instead, we adopt an unsupervised approach that finds patterns in the data—confidence values coupled with performance scores (Section 6.1), and then attempts to fit unseen data to these patterns (Section 6.2). Heuristics are still needed in order to decide which response-generation method to apply to an unseen case, but they are applied only after the learning is complete (Section 6.3). In other words, the subjective process of setting performance criteria (which should be conducted by the organization running the help-desk) does not influence the machine learning process.

6.1 Training

We train the system by clustering the "experiences" of the response-generation methods in addressing requests, where each experience is characterized by the value of the confidence measure employed by a method and its subsequent performance, reflected by precision and recall (Equations (7) and (8), respectively). We then use the program Snob (Wallace and Boulton 1968; Wallace 2005) to cluster these experiences. Figure 8(a) is a projection of the centroids of the clusters produced by Snob into the three most significant dimensions discovered by Principal Component Analysis (PCA)—these dimensions account for 95% of the variation in the data. The bottom part of Figure 8(b)

FPO



(a) Dimensions produced by PCA

Unseen case

Method	Confidence	Performance
Doc–Ret	0.13	???
Doc–Pred	0.75	???
Sent–Pred	0.77	???
Sent–Hybrid	0.50	???

	Method	conf	conf	conf
Confidence	Doc–Ret	0.37	0.19	0.16
	Doc–Pred	0.25	0.55	0.71
	Sent–Pred	0.66	0	0.83
	Sent–Hybrid	0.56	0	0.66
		(prec,rec)	(prec,rec)	(prec,rec)
Performance	Doc–Ret	**(0.91, 0.76)**	(0,0)	(0, 0)
	Doc–Pred	(0.56, 0.51)	(0,0)	**(0.90, 0.89)**
	Sent–Pred	(0.89, 0.44)	(0,0)	**(0.97, 0.78)**
	Sent–Hybrid	(0.90, 0.38)	(0,0)	(0.97,0.66)
		Cluster 8	**Cluster 11**	**Cluster 16**

(b) Centroids of three clusters

Figure 8
Clusters of response-generation methods obtained from the training set: (a) dimensions produced by PCA and (b) sample clusters.

shows the (unprojected) centroid values of three of the clusters (the top part of the figure will be discussed subsequently).[15] These clusters were chosen because they illustrate clearly three situations of interest.

- **Single winner –** Cluster 8 shows a case where a single strategy is clearly preferred. In this case the winner is Doc-Ret. Its precision and recall values in this cluster are 0.91 and 0.76, respectively.
- **No winner –** Cluster 11 shows a case where none of the methods do well. They all yield precision and recall values of 0.
- **Multiple winners –** In Cluster 16, both Doc-Pred and Sent-Pred are competitive, yielding precision and recall values of (0.90, 0.89) and (0.97, 0.78), respectively. A decision between the two methods depends on whether we favor precision or recall, as discussed subsequently.

6.2 Prediction

We test the system with an unseen set of requests, which we feed to each response-generation method. Each method then outputs a value for its confidence measure. We do not know in advance how each method will perform—this information is missing, and we predict it on the basis of the clusters obtained from the training set. Our prediction of how well the different methods will perform on an unseen case is based on (1) how well the unseen case fits each of the clusters and (2) the average performance values in each cluster as indicated by its centroid.

15 The Sent-Ret method was excluded from our experiments due to its poor performance in the comparative study described in Section 4.

The top part of Figure 8(b) shows an example of an unseen case whose confidence values are most similar to those in the centroid of Cluster 16. In this case, the selection of a method depends on whether we favor recall or precision, as Doc-Pred has a higher recall than Sent-Pred, but Sent-Pred has a higher precision. Now, Cluster 15 (not labeled in Figure 8(a)) contains similar confidence values to those of Cluster 16, but its (precision, recall) values for Doc-Pred and Sent-Pred are (0.76, 0.66) and (0.84, 0.67) respectively. If Cluster 15 had the strongest match with the unseen case, then Sent-Pred would have been chosen, regardless of any preferences for precision or recall. However, it is not clear that the best policy consists of simply choosing the method suggested by the best-matching cluster. This is particularly the case when an unseen example has a reasonably good match with more than one cluster (e.g., Clusters 15 and 16).

The prediction step is implemented using Snob, which is able to accept data with missing values. For each unseen data point x (with missing performance values), Snob calculates $\Pr(c_i|x)$, the posterior probability of each cluster c_i given this data point. These probabilities indicate how well an unseen case matches each of the clusters. For example, for the unseen case in Figure 8(b), Snob may assign posterior probabilities of 0.5 and 0.3 to Clusters 16 and 15, respectively (and lower probabilities to weaker-matching clusters, such as Cluster 8).[16]

We utilize these probabilities in two alternative ways for estimating the performance of each response-generation method: **Max**, which considers only the best-matching cluster (i.e., that with the highest posterior probability); and **Weighted**, which considers all clusters, weighted by their posterior probabilities. These techniques are used to calculate the estimated precision ($\widehat{p_k}$) and estimated recall ($\widehat{r_k}$) of each response-generation method $\textbf{method}_k \in \{\text{Doc-Ret, Doc-Pred, Sent-Pred, Sent-Hybrid}\}$ as follows.

- **Max:**

$$\widehat{p_k} = p_k^{(i^*)} \quad \text{and} \quad \widehat{r_k} = r_k^{(i^*)}, \quad \text{for } i^* = \text{argmax}_{i=1,\ldots,N} \Pr(c_i|x) \tag{10}$$

  where $p_k^{(i)}$ and $r_k^{(i)}$ are the precision and recall components, respectively, for $\textbf{method}_k$ in the centroid of cluster i, and N is the number of clusters.

- **Weighted:**

$$\widehat{p_k} = \sum_{i=1}^{N} \Pr(c_i|x) \times p_k^{(i)} \quad \text{and} \quad \widehat{r_k} = \sum_{i=1}^{N} \Pr(c_i|x) \times r_k^{(i)} \tag{11}$$

6.3 Method Selection

In order to select a method for a given request, we need to combine our estimates of precision and recall into an overall estimate of performance, and then choose the method with the best estimated performance. The standard approach for combining precision and recall is to compute their harmonic mean, F-score, as we have done in our

16 In principle, we could have used a classification method to predict clusters from the values of the confidence measures for unseen cases. We posit that this would not have a significant effect on the results, in particular for MML-based classification techniques, such as Decision Graphs (Oliver 1993).

comparative evaluation in Section 4. However, in order to accommodate different levels of preference towards precision or recall, as discussed herein, we use the following weighted F-score calculation (van Rijsbergen 1979).

$$\text{F-score} = \left\{ \frac{w}{\text{Precision}} + \frac{1-w}{\text{Recall}} \right\}^{-1} \tag{12}$$

where w is a weight between 0 and 1 given to precision. When $w = 0.5$ we have the standard usage of F-score (Equation (9)), and for $w > 0.5$, we have a preference for high precision. For example, for $w = 0.5$, the precision and recall values of Cluster 16 (Figure 8(b)) translate to F-scores of 0.895 and 0.865 for Doc-Pred and Sent-Pred, respectively, leading to a choice of Doc-Pred. In contrast, for $w = 0.75$, the respective F-scores are 0.897 and 0.914, leading to a choice of Sent-Pred.

7. Evaluation of Meta-Learning

We evaluate the meta-learning system by looking at the quality of the response produced by the method selected by this system, where, as done in Section 4, quality is measured using F-score and precision. However, here we employ 5-fold cross-validation (instead of 10-fold) to ensure that we get a good spread of selected methods in each testing split. This is particularly important when only a few methods dominate for a data set.

7.1 Experimental Set-Up

In our evaluation, we compare the alternative approaches for estimating performance (Equations (10) and (11)), and consider the effect of favoring precision when selecting a method via the weighted F-score calculation (Equation (12)). To perform these comparisons we employ the following configurations.

- **Max50:** Use the argmax alternative for estimating performance (Equation (10)), and $w = 0.5$ in Equation 12.
- **Max75:** As Max50, but with $w = 0.75$.
- **Weighted50:** Use the weighted alternative for estimating performance (Equation (11)), and $w = 0.5$ in Equation (12).
- **Weighted75:** As Weighted50, but with $w = 0.75$.

We also devised the following baselines to help ground our results.

- **Random:** Select between the methods randomly.
- **Gold50:** Select between the methods based on their actual performance (as opposed to their estimated performance), using $w = 0.5$ in Equation (12).
- **Gold75:** As Gold50, but with $w = 0.75$.

As we saw from Cluster 11 in Figure 8(b), the estimated performance can be low for all the response-generation methods. Therefore, we also test these configurations in

Table 6
Precision and F-score for the meta-learning methods averaged over the corpus.

	All cases		Cases with precision ≥ 0.8		
	Precision Avg. (St dev)	**F-score Avg. (Std ev)**	**Precision Avg. (St dev)**	**F-score Avg. (St dev)**	**Coverage Avg.**
			(actual precision ≥ 0.8)		
Random	0.558 (0.37)	0.376 (0.33)	0.955 (0.06)	0.696 (0.25)	37.6%
Gold50	0.725 (0.26)	0.548 (0.30)	0.934 (0.06)	0.732 (0.26)	53.0%
Gold75	0.781 (0.25)	0.537 (0.29)	0.952 (0.06)	0.689 (0.26)	60.6%
			(estimated precision ≥ 0.8)		
Max50	0.704 (0.28)	0.507 (0.31)	0.844 (0.22)	0.648 (0.30)	56.7%
Max75	0.768 (0.27)	0.498 (0.28)	0.911 (0.17)	0.629 (0.27)	56.7%
Weighted50	0.727 (0.27)	0.512 (0.30)	0.874 (0.18)	0.649 (0.29)	57.1%
Weighted75	0.776 (0.26)	0.499 (0.28)	0.919 (0.16)	0.626 (0.27)	57.1%

a practical setting where the system has the choice of not selecting any method if the estimated performance of all the methods is poor. We envisage that a practical system would behave in this manner, in the sense that a request for which none of the existing methods can produce an appropriate response would be passed to an operator. As mentioned in Section 4.2, we consider precision to be an important practical criterion because it does not penalize partial but correct responses. Therefore, we "implement" our practical system by selecting only responses whose estimated precision is above 0.8. For these tests we also report on coverage, that is, the percentage of cases where this condition is met. Note that the baselines do not have an estimated precision because they do not use meta-learning. However, for completeness, we implement the practical system for them as well, with a threshold of 0.8 on actual precision.

7.2 Results

Table 6 shows the results of our tests averaged over all the cases in the corpus (with standard deviations in brackets). The left-hand side corresponds to the setting where the system always selects a response-generation method, and the right-hand side corresponds to the setting where a method is selected only if its precision equals or exceeds 0.8 (this is an estimated precision for the Max and Weighted configurations, and an actual precision for the Gold and Random baselines).

Let us first consider the left-hand side of Table 6. As expected, the Random baseline has the worst performance. The Gold baselines outperform their corresponding meta-learning counterparts (except for the precision of Weighted50), but the differences in precision are not statistically significant between the Gold and the Weighted configurations (using a t-test with a 1% significance level). Comparing the corresponding Weighted and Max configurations, the former is superior, but this is statistically significant only for the difference in precision values between Weighted50 and Max50. Comparing a standard F-score calculation with a precision-favoring calculation ($w = 0.5$ versus $w = 0.75$ in Equation (12)), as expected, precision is significantly higher for the latter in all testing configurations ($p < 0.01$). This increase in precision is at the expense of a reduced F-score, but the increase is larger than the reduction.

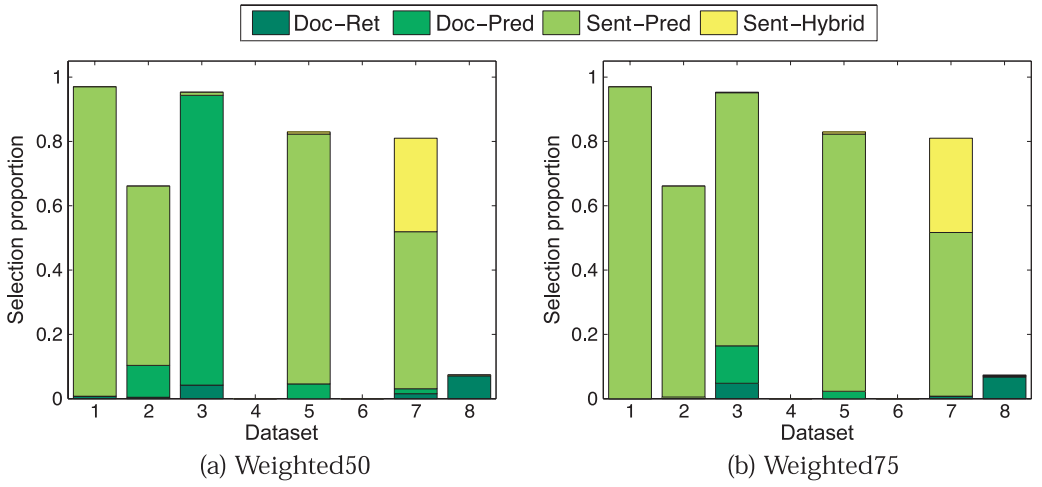


Figure 9
Proportion of methods selected by meta-learning with the Weighted50 and Weighted75 configurations for each data set.

Now, in the right-hand side of Table 6, we see that the Random configuration has the best precision and the second-best F-score,[17] but its coverage is quite low (only 37.6%). In contrast, the meta-learning configurations cover a proportion of the requests that is comparable to the coverage of the Gold baselines (approximately 57%), and all the results are substantially improved; as expected, all the precision values are high, and also more consistent than before (they have a lower standard deviation). These results are quite impressive for the meta-learning configurations, as their selection between methods is based on *estimated* precision, as opposed to the baselines, whose selections are based on *actual* precision, which is not available in practice. Comparing the corresponding Weighted and Max configurations, there are no significant differences in F-score, but Weighted outperforms Max on precision (the difference between Weighted75 and Max75 produces a p-value of 0.035). Finally, comparing $w = 0.5$ with $w = 0.75$ for both Weighted and Max, as for the All-cases results, the increase in precision is larger than the reduction in F-score ($p < 0.01$).

Now that we have evaluated the ability of the meta-level process to select between response-generation methods, let us inspect what happens for each data set. We saw in Section 4 that the various methods differ in their applicability to the different data sets (Figure 4). Hence, we would expect the meta-level process to select between the methods differently for each data set. Figure 9 shows the method-selection proportions for each data set for the Weighted50 and Weighted75 configurations, using the practical setting where the system can choose not to select any method. What is immediately notable is that no method is selected for two of the data sets (nos. 4 and 6). This follows from the poor performance of all the methods for these data sets (Figures 4(b) and 4(c)). At first glance, it appears that the selection of the Sent-Pred method for data set 1 contradicts the results in Figure 4, which shows a low coverage of Sent-Pred for this data set. However, this selection is justified by the fact that the meta-learning procedure selects

17 This seemingly good performance represents the average precision and resultant F-score of all the responses with actual precision ≥ 0.8, and is not the result of the application of a selection strategy.

methods based on their historical performance (precision and recall), without filtering on applicability threshold (which is the basis for coverage). Figure 9 also highlights the impact of favoring precision on the selection of the Sent-Pred method instead of Doc-Pred. This effect is most dramatic for data set 3, but it can also be observed for data sets 2 and 5.

Overall, Sent-Pred dominates for most data sets (except data set # 3 under the Weighted50 configuration, and data set # 8). Also notable is the fact that Sent-Hybrid is selected only for data set 7. We postulate that this happens because data set 7 merges several sub-topics, and has different versions of similar responses, where each version has a generic component combined with a request-specific component (this feature is not shared by the other merged data sets 6 and 8). Sent-Pred is not confident enough to select one of these specific components, whereas Sent-Hybrid is. This ability to select a specific response is demonstrated in Figure 10, which shows a request from data set 7 and its automated response. This response belongs to a medium-cohesion cluster which contains responses that share the generic segment up to *regarding*, but the rest of the response refers specifically to terms in the request.

7.3 Summary

The meta-learning results may be summarized as follows.

- The meta-learning system significantly outperforms the random selection baseline, and is competitive with the gold baseline.
- The Weighted option for estimating performance (Equation (11)) is preferable to the Max option (Equation (10)), as it is better able to handle the uncertainty that arises when a particular method has a wide range of precision and recall values.
- A precision-favoring approach is recommended, as the resultant increase in precision (reflecting a higher proportion of correct information) is larger than the reduction in F-score.
- Overall, Doc-Pred and Sent-Pred were the preferred methods, with Sent-Pred clearly dominating for the Weighted75 configuration. Sent-Hybrid was selected often for data set 7, and Doc-Ret was the only method selected, albeit seldom, for data set 8.
- Because the system can estimate performance prior to producing a response, it is able to opt for a non-response rather than risk producing a bad one. The decision of what is a bad response should be made by the organization using the system. With the stringent criterion we have chosen (precision$\geq$ 0.8), the system yields a good performance for approximately 57% of the requests.

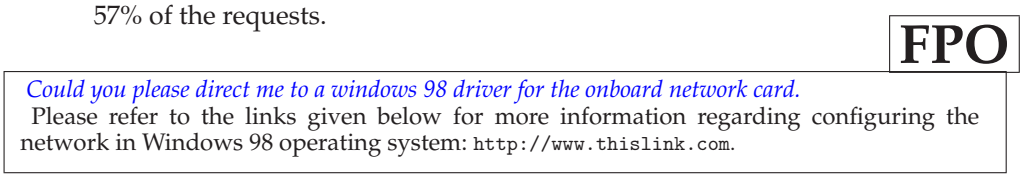


Figure 10
Example showing an appropriate response generated by the Sent-Hybrid method.

8. Related Research

The automation of help-desk responses has been previously tackled using mainly knowledge-intensive paradigms, such as expert systems (Barr and Tessler 1995) and case-based reasoning (Watson 1997). Such technologies require significant human input, and are difficult to create and maintain (Delic and Lahaix 1998). In contrast, the techniques examined in this article are corpus-based and data-driven. The process of composing a planned response for a new request is informed by probabilistic and lexical properties of the requests and responses in the corpus.

There are very few reported attempts at corpus-based automation of help-desk responses (Carmel, SHtalhaim, and Soffer 2000; Lapalme and Kosseim 2003; Bickel and Scheffer 2004; Malik, Subramaniam, and Kaushik 2007).

eResponder, the system developed by Carmel, Shtalhaim, and Soffer (2000), retrieves a list of request–response pairs and presents a ranked list of responses to the user. If the user is unsatisfied with this list, an operator is asked to generate a new response. The operator is assisted in this task by the retrieval results: The system highlights the request-relevant sentences in the ranked responses. However, there is no attempt to automatically generate a single response.

Bickel and Scheffer (2004) compared the performance of document retrieval and document prediction for generating help-desk responses. Their retrieval technique, which is similar to our request-to-request Doc-Ret method, matches user questions to the questions in a database of question–answer pairs. Their prediction method, which is similar to Doc-Pred, is based on clustering the responses in the corpus into semantically equivalent answers, and then training a classifier to match a query with one of these classes. The generated response is the answer that is closest to the centroid of the cluster. Bickel and Scheffer's results are consistent with ours, in the sense that the performance of the Doc-Ret method is significantly worse than that of Doc-Pred. However, it is worth noting that their corpus is significantly smaller than ours (805 question–answer pairs), their questions seem to be much simpler and shorter than those in our corpus, and the replies shorter and more homogeneous.

Malik, Subramaniam, and Kaushik (2007) developed a system that builds question–answer pairs from help-center e-mails, and then maps new questions to existing questions in order to retrieve an answer. This part of their approach resembles our Doc-Ret method, but instead of retrieving entire response documents, they retrieve individual sentences. In addition, rather than including actual response sentences in a reply, their system matches response sentences to pre-existing templates and returns the templates.

Lapalme and Kosseim (2003) investigated three approaches to the automatic generation of response e-mails: text classification, case-based reasoning, and question answering. Text classification was used to group request e-mails into broad categories, some of which, such as requests for financial reports, can be automatically addressed. The question-answering approach and the retrieval component of the case-based reasoning approach were data driven, using word-level matches. However, the personalization component of the case-based reasoning approach was rule-based (e.g., rules were applied to substitute names of individuals and companies in texts).

With respect to these systems, the contribution of our work lies in the consideration of different kinds of corpus-based approaches (namely, retrieval and prediction) applied at different levels of granularity (namely, document and sentence).

Two applications that, like help-desk, deal with question–answer pairs are: summarization of e-mail threads (Dalli, Xia, and Wilks 2004; Shrestha and McKeown 2004), and answer extraction in FAQs (Frequently Asked Questions) (Berger and Mittal 2000;

Berger et al. 2000; Jijkoun and de Rijke 2005; Soricut and Brill 2006). An important difference between these applications and help-desk is that help-desk request e-mails are not simple queries. In fact, some e-mails do not contain any queries at all, and even if they do, it is not always straightforward to distinguish the queries from the text that provides background information. Therefore, the generation of a help-desk response needs to consider a request e-mail in its entirety, and ensure that there is sufficient evidence to match the request with a response or parts of responses.

In e-mail-thread summarization, Dalli, Xia, and Wilks (2004) applied a procedural approach where they recognized named entities and performed anaphora resolution prior to applying ranking metrics to select sentences for inclusion in a thread summary. However, their approach does not specifically address the question–answer aspect of an e-mail thread, potentially omitting important information. This problem was addressed by Shrestha and McKeown (2004), who performed supervised learning in order to match questions with answers in e-mail threads, as a first step in the summarization of such threads. A significant difference between our approach and theirs is in our use of unsupervised learning, which is necessitated by the size of our data set. Also, Shrestha and McKeown used high-level features for machine learning, as well as word-based features. As indicated in Section 3.2.2, our Sent-Pred experiments with high-level features (specifically syntactic features) did not improve our results. Finally, Shrestha and McKeown used paragraphs as a unit of information—an approach we tried late in our project with encouraging results. This suggests that there are situations where one can generalize a response that is longer than a sentence but shorter than a whole document. Unfortunately, we could not pursue this avenue of research owing to time limitations.

In FAQs, Berger and Mittal (2000) employed a sentence retrieval approach based on a language model where the entire response to an FAQ is considered a sentence, and the questions and answers are embedded in an FAQ document. They complemented this approach with machine learning techniques that automatically learn the weights of different retrieval models. Berger et al. (2000) compared two retrieval approaches (TF.IDF and query expansion) and two predictive approaches (statistical translation and latent variable models). Jijkoun and de Rijke (2005) compared different variants of retrieval techniques. Soricut and Brill (2006) compared a predictive approach (statistical translation), a retrieval approach based on a language-model, and a hybrid approach which combines statistical chunking and traditional retrieval. Two significant differences between help-desk and FAQs are the following.

- The responses in the help-desk corpus are personalized, which means that on one hand, we must abstract from them sufficiently to obtain meaningful regularities, and on the other hand, we must be careful not to abstract away specific information that addresses particular issues.
- Help-desk responses have much more repetition than FAQs, because the corpus is made up of individual dialogues, rather than typical request–response pairs. This motivates the use of multi-document summarization techniques, rather than question-answering approaches, to extract individual answers.

These issues also differentiate the help-desk application from other types of question-answering applications, specifically those found in the field of restricted domain question answering (Mollá and Vicedo 2007).

In addition to the different response-generation methods, we have proposed a meta-level strategy to combine them. This kind of meta-learning is referred to as **stacking** by the Data Mining community (Witten and Frank 2000). Lekakos and Giaglis (2007) implemented a supervised version of this approach for a recommender system, as opposed to our unsupervised version. They also proposed two major categories of meta-learning approaches for recommender systems, **merging** and **ensemble**, each subdivided into the more specific subclasses suggested by Burke (2002) as follows. The merging category corresponds to techniques where the individual methods affect each other in different ways (this category encompasses Burke's **feature combination**, **cascade**, **feature augmentation**, and **meta-level** sub-categories). The ensemble category corresponds to techniques where the predictions of the individual methods are combined to produce a final prediction (this category encompasses Burke's **weighted**, **switching**, and **mixed** sub-categories).

Our system falls into the ensemble category, because it combines the results of the various methods into a single outcome. More specifically, it belongs to Burke's switching sub-category, where a single method is selected on a case-by-case basis. A similar approach is taken in Rotaru and Litman's (2005) reading comprehension system, but their system does not perform any learning. Instead it uses a voting mechanism to select the answer given by the majority of methods. The question answering system developed by Chu-Carroll et al. (2003) belongs to the merging category of approaches, where the output of an individual method can be used as input to a different method (this corresponds to Burke's cascade sub-category). Because the results of all the methods are comparable, no learning is required: At each stage of the "cascade of methods," the method that performs best is selected. In contrast to these two systems, our system employs methods that are not comparable, because they use different metrics. Therefore, we need to learn from experience when to use each method.

9. Conclusion

Despite its theoretical importance and commercial impact, the generation of e-mail-based help-desk responses has received scant attention to date. In this article, we have investigated complementary corpus-based, information-gathering methods for automatically addressing help-desk requests. Our results show that a large corpus of request–response e-mail pairs supports the automation of a significant portion of the help-desk task with data-driven techniques that reuse responses or parts thereof. These techniques are particularly suitable for repetitive, non-technical issues, allowing help-desk operators to focus on more challenging, technical requests.

Our results also show that different methods are applicable to different situations that arise in the help-desk domain, and that the performance of different methods varies for different data sets. This suggests that there must be an underlying relationship between methods and features of data sets that needs to be accounted for (Section 4.3). Additionally, our results yield insights regarding the following issues.

- **Retrieval versus prediction –** Some situations warrant a retrieval approach, whereas others require a predictive approach. The former applies when the content of a request matches previous cases in the corpus. The latter applies when requests and responses do not match on content, but correlations exist between a few predictive words in a request and a response, which is often generic in these cases. A hybrid prediction–retrieval approach was also investigated. Our hybrid method

extracts generalizations at the sentence level, and employs a retrieval component that tailors the selection of sentences to the specific issues raised in a request e-mail. Although this appears to be a sensible approach, the results of our evaluation are not conclusive, and further investigation is required.

- **Levels of granularity –** Although simple document-level reuse methods are sufficient in many cases, more complex sentence-level reuse methods, which involve extractive multi-document summarization, provide a viable alternative when a complete response document cannot be reused. This happens when there is insufficient evidence for the reuse of such a document, but there is enough evidence for a partial response.

We have also performed a small user study, which highlighted issues regarding the evaluation of a large corpus such as ours. Despite its modest size, our user study was useful, as it provided a subjective evaluation of the methods considered, and linked the results of our automatic evaluation with these subjective assessments.

Our work also provides a unified solution to help-desk response automation via the meta-learning component. Although our comparative investigation demonstrates the applicability of the different methods, the meta-learning component provides a way to automatically select a method. We have offered an unsupervised approach that learns which is the most promising method based on previous experience. It does so by learning the relationship between the value of the confidence (applicability) measure of each method and its subsequent performance. This eliminates the need to set subjective thresholds on these confidence values, and instead transfers subjective decision-making to a more intuitive part of the system, namely, the actual performance of the methods, measured by the quality of the generated responses. In this way, help-desk managers can decide how strict the system should be, for example, on the precision of responses.

We are encouraged by the fact that we have achieved a reasonable level of performance using only a simple, low-level bag-of-words representation. One avenue for future research is to investigate more sophisticated representations, such as incorporating word-based similarity metrics into the bag-of-words representation, employing query expansion during retrieval, and taking into account syntactic features during retrieval and prediction (recall that we incorporated grammatical and sentence-based syntactic features into the Sent-Pred method without significantly affecting performance, Section 3).

Another avenue for future research is the investigation of intermediate levels of granularity, such as paragraphs. Ideally there should be a mechanism that determines dynamically the most suitable level of granularity for capturing the regularities in a collection of e-mails. An information-theoretic approach, such as the MML criterion (Wallace and Boulton 1968; Wallace 2005), may be a promising way to address this problem.

Acknowledgments

This research was supported in part by grant LP0347470 from the Australian Research Council and by an endowment from Hewlett-Packard. The authors also thank Hewlett-Packard for the extensive anonymized help-desk data, Nathalie Japkowicz for her advice on the meta-learning portions of this work, and the anonymous reviewers for their insightful comments.

References

Banerjee, S. and T. Pedersen. 2003. The design, implementation, and use of

the Ngram Statistics Package. In *CICLing 2003 – Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 370–381, Mexico City. Mexico.

Barr, A. and S. Tessler. 1995. Expert systems: A technology before its time. *AI Expert*, available at `www.stanford.edu/group/scip/avsgt/expertsystems/aiexpert.html`.

Barzilay, R., N. Elhadad, and K. R. McKeown. 2001. Sentence ordering in multidocument summarization. In *HLT01 – Proceedings of the First Human Language Technology Conference*, pages 1–7, San Diego, CA.

Barzilay, R. and K. R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Berger, A., R. Caruana, D. Cohn, D. Freitag, and V. Mittal. 2000. Bridging the lexical chasm: Statistical approaches to answer-finding. In *SIGIR'00 – Proceedings of the 23rd Annual International ACM International Conference on Research and Development in Information Retrieval*, pages 192–199, Athens.

Berger, A. and V. Mittal. 2000. Query-relevant summarization using FAQs. In *ACL2000 – Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 294–301, Hong Kong.

Bickel, S. and T. Scheffer. 2004. Learning from message pairs for automatic email answering. In *ECML04 – Proceedings of the European Conference on Machine Learning*, pages 87–98, Pisa.

Burke, R. 2002. Hybrid recommender systems. *User Modeling and User-Adapted Interaction*, 12(4):331–370.

Carletta, J. 1996. Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, 22(2):249–254.

Carmel, D., M. Shtalhaim, and A. Soffer. 2000. eResponder: Electronic question responder. In *CoopIS'02 – Proceedings of the 7th International Conference on Cooperative Information Systems*, pages 150–161, Eilat.

Chang, C. C. and C. J. Lin, 2001. *LIBSVM: A Library for Support Vector Machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chu-Carroll, J., K. Czuba, J. M. Prager, and A. Ittycheriah. 2003. In question answering, two heads are better than one. In *HLT-NAACL 2003 – Proceedings of the 2003 Language Technology Conference*, pages 24–31, Edmonton.

Dalli, A., Y. Xia, and Y. Wilks. 2004. Adaptive information management: FASiL email summarization system. In *COLING'04 – Proceedings of the 20th International Conference on Computational Linguistics*, pages 23–27, Geneva.

Delic, K. A. and D. Lahaix. 1998. Knowledge harvesting, articulation, and delivery. *The Hewlett-Packard Journal*, May:74–81.

Feng, D., E. Shaw, J. Kim, and E. Hovy. 2006. An intelligent discussion-bot for answering student queries in threaded discussions. In *IUI'06 – Proceedings of the 11th International Conference on Intelligent User Interfaces*, pages 171–177, Sydney.

Filatova, E. and V. Hatzivassiloglou. 2004. Event-based extractive summarization. In *Proceedings of the ACL'04 Workshop on Summarization*, pages 104–111, Barcelona.

Goldstein, J., V. Mittal, J. Carbonell, and M. Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of the ANLP/NAACL 2000 Workshop on Automatic Summarization*, pages 40–48, Seattle, WA.

Jijkoun, V. and M. de Rijke. 2005. Retrieving answers from frequently asked questions pages on the Web. In *CIKM'05 – Proceedings of the ACM 14th Conference on Information and Knowledge Management*, pages 76–83, Bremen.

Lapalme, G. and L. Kosseim. 2003. Mercure: Towards an automatic e-mail follow-up system. *IEEE Computational Intelligence Bulletin*, 2(1):14–18.

Lekakos, G. and G. M. Giaglis. 2007. A hybrid approach for improving predictive accuracy of collaborative filtering algorithms. *User Modeling and User-Adapted Interaction*, 17(1):5–40.

Leuski, A., R. Patel, D. Traum, and B. Kennedy. 2006. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 18–27, Sydney.

Lin, C. Y. and E. H. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *HLT-NAACL 2003 – Proceedings of the 2003 Language Technology Conference*, pages 71–78, Edmonton.

Malik, R., L. V. Subramaniam, and S. Kaushik. 2007. Automatically selecting answer templates to respond to

customer emails. In *IJCAI'07 – Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1659–1664, Hyderabad.

Marom, Y. and I. Zukerman. 2006. Automating help-desk responses: A comparative study of information-gathering approaches. In *Proceedings of the COLING-ACL Workshop on Task-Focused Summarization and Question Answering*, pages 40–47, Sydney.

Marom, Y. and I. Zukerman. 2007a. Evaluation of a large-scale email response system. In *Proceedings of the IJCAI'07 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 28–33, Hyderabad.

Marom, Y. and I. Zukerman. 2007b. A predictive approach to help-desk response generation. In *IJCAI'07 – Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1665–1670, Hyderabad.

Marom, Y., I. Zukerman, and N. Japkowicz. 2007. A meta-learning approach for selecting between response automation strategies in a help-desk domain. In *AAAI-07 – Proceedings of the 22nd Conference on Artificial Intelligence*, pages 907–912, Vancouver.

Mollá, D. and J. L. Vicedo. 2007. Question answering in restricted domains: An overview. *Computational Linguistics*, 33(1):41–61.

Oliver, J. J. 1993. Decision graphs—an extension of decision trees. In *Proceedings of the 4th International Workshop on Artificial Intelligence and Statistics*, pages 343–350, Fort Lauderdale, FL.

Rotaru, M. and D. J. Litman. 2005. Improving question answering for reading comprehension tests by combining multiple systems. In *Proceedings of the AAAI 2005 Workshop on Question Answering in Restricted Domains*, pages 46–50, Pittsburgh, PA.

Roy, S. and L. V. Subramaniam. 2006. Automatic generation of domain models for call-centers from noisy transcriptions. In *COLING-ACL'06 – Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 737–744, Sydney.

Salton, G. and M. J. McGill. 1983. *An Introduction to Modern Information Retrieval*. McGraw Hill, New York.

Shrestha, L. and K. R. McKeown. 2004. Detection of question-answer pairs in email conversations. In *COLING'04 – Proceedings of the 20th International Conference on Computational Linguistics*, pages 889–895, Geneva.

Soricut, R. and E. Brill. 2006. Automatic question answering using the Web: Beyond the factoid. *Information Retrieval*, 9(2):191–206.

van Rijsbergen, C. J. 1979. *Information Retrieval*. Buttersworth, London.

Vapnik, V. N. 1998. *Statistical Learning Theory*. Wiley-Interscience, New York.

Wallace, C. S. 2005. *Statistical and Inductive Inference by Minimum Message Length*. Springer, Berlin.

Wallace, C. S. and D. M. Boulton. 1968. An information measure for classification. *The Computer Journal*, 11(2):185–194.

Watson, I. 1997. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufmann Publishers, San Mateo, CA.

Witten, I. H. and E. Frank. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, San Francisco, CA.