Towards a Personal Briefing Assistant

Nikesh Garera and Alexander I. Rudnicky

School of Computer Science, Carnegie Mellon Univeristy
5000 Forbes Avenue, Pittsburgh, PA 15213
{garera, air}@cs.cmu.edu

Abstract

We describe a system intended to help report writers produce summaries of important activities based on weekly interviews by members of a project. A key element of such a system is to *learn* different user and audience preferences in order to produce tailored summaries. The system learns desired qualities of summaries based on observation of user selection behavior, and builds a regression-based model using item features as parameters. The system's assistance consists of presenting the writer with a successively better order list of items from which to choose. The results with the assistance of our system show a significant improvement in average precision (and other metrics) by the end of the learning period as compared to the baseline of no learning. We also report our ongoing work of automatic feature extraction to make this approach domain independent.

Introduction

The preparation of summary reports from raw information is a common task in research projects. These reports highlight the important activities and are very useful for providing project overviews. A tool that highlights useful items for a summary would allow report writers to be more productive, by reducing the time needed to assess individual items

An important characteristic of the report writing task is producing *tailored* reports. For example, in an academic environment, a report writer might need to write a quarterly report for a research funding body which might be quite different from a report to the internal peers or a student report. Thus, the challenge involved here is to be able to learn different briefing models for catering the needs of different audiences. Not only these reports vary depending on the audience but they also vary depending on who writes them. Thus, different report writers may have different styles and the assistant should also be able to learn the preferences of these different users.

The idea of such an assistant leads to many interesting questions. What is the nature and form of assistance it can provide? How does it observe the user and learn from it? In what ways can the user provide direct instruction? Does the assistant need to be domain specific and have complete knowledge about the domain or can it be made domain independent?

In this paper, we address some these questions by describing the Briefing Assistant agent. The raw information from which summaries are created is a set of weekly project interviews. Our goal is to create a system that learns user preferences and assists the user for creating weekly summaries of project activity.

This work is done in the context of a broader project targeted towards automated assistants (RADAR). The overall goal of RADAR is to develop a "cognitive personal assistant" that will assist busy managers by saving time in routine tasks and will also assist them in making better decisions.

The remainder of the paper is organized as follows: Section 2 reviews related work in text summarization. Section 3 describes our approach in detail, including a description of interview data, the system architecture, the features used in learning, a thorough evaluation study and our ongoing work towards automatic feature extraction.

Related Work

Automatic text summarization has been attempted since 1950's [Luhn, 1958] and most of the work is targeted towards producing generic newswire summaries using simple statistical and similarity measures. Seminal work in corpus-based feature identification for extractive summaries was done by Edmundson in as early as 1960's and forms the basis for contemporary extractive summaries. [Edmundson, 1969]. Corpus based approaches have been proposed where a training corpus is used for improving summarization. [Kupiec et al., 1995], [Teufel and Moens, 1997] use a statistical classification approach for sentence extraction. [Lin and Hovy, 1997] learn a series of sentence positions. [Mani and Bloedorn, 1998] use a trainable approach aimed at both generic and user-specific summarization. These techniques involve learning a series of rules using location, cohesion and thematic features. The current approach differs in using an iterative human-in-the-loop process to train the selection model.

A few personalized interactive learning systems have also been proposed for summarization. [Amini, 2000] describes a query-relevant text summary system based on interactive learning. Learning is in the form of query expansion and sentence scoring by classification. A summarization system based on user's annotations was proposed by [Zhang, 2003]. Annotations and their contexts are extracted to represent features of sentences, which are given different weights for representation of the document.

Our work differs from previous work on news wire summaries in its focus on "raw" material rather than on finished text. One consequence of this is that features appropriate to finished text, such as position, cue phrases, and co-occurrences may not be relevant, particularly as our raw material is typically gathered from a variety of sources that may differ in style and in structure.

Approach

Our target application scenario involves a report writer producing summaries on a week-to-week basis and our goal is to make this person more efficient over time. We propose to do this by having the agent present the writer with successively better ordered lists of items (in the sense that digest-worthy items appear at the top of the ordered list), with the presumption that over time the writer will come to trust the agent's ordering and will examine fewer items towards the bottom of the list.

Target Domain (Interview Data)

Members of a medium-size academic research project were approached and asked to produce a weekly summary of their project-related activities, in the form of bullets. Participants could either fill in a structured interview (see Figure 1) on their own or sit with an administrative assistant who would interview them according to the themes in the form. With the exception of one participant, all chose to be personally interviewed. The interviewer was encouraged to ask follow-up questions if she felt that the information was incomplete or otherwise difficult to understand. Data collection ran over a 16 week period and yielded a total of 332 items from 7 participants. The total number of items per week varied considerably, consequently for purposes of this study we repartitioned the data into 12 periods each containing between 25 and 35 items (chronological order was maintained).

Figure 1 shows interview structure, containing seven different sections. The interviewer asks several questions about every section and writes few bullet items in the respective sections. For each of the 12 weeks, a human expert (one of the authors) prepared a weekly summary by annotating each item as either a summary item, or a non-summary item. The only constraint was that exactly five items had to be included in the summary.

- Concrete Achievements
e.g. Presented Project Plan to visiting project managers on 20th May 2003
- Problems and Solutions
e.g. May be neglecting the idea of "audience". All communication is centered on user request and neglects that the content producer has a point of view.
- People and Places
e.g. Met with X and Y, founders of Z, which is in the Knowledge Management business. It appears that the system and the tools they have built may be of use to us.
- News and Views
e.g. Read about the Shaken/KM knowledge base system
- Artifacts and Processes
e.g. Learned about PLONE – System for collaborative construction of websites
- Project Status
e.g. We are two weeks behind schedule in implementation of the prototype.
- General Activity Report
e.g. Considered attaching a "spy" module that tracks ALL of user's keystrokes and mouse clicks; purpose is to see if we can mine these data.

Fig. 1: The different sections in the interview structure. Examples of interview sentences are show in the respective sections.

Architecture

Figure 2 shows the architectural framework the briefing assistant. The function of each module can be described in the following user scenario:

User task: To produce weekly summaries of interviews conducted in every week.

1. **Input:** The set of weekly interviews is provided as input to the assistant.
2. **Re-ranking:** The summary module re-ranks the interview sentences of the current week according to the best learned user model (moving the important sentences to the top of the list).
3. **User feedback:** The user goes through the sentence list and selects important sentences for the current week's summary. (Fig. 3 shows a sample user session)

4. **Learning module:** The learning module takes the user selection as the training data[1] and fits a model over the features available to the system. This model then becomes the current best learned user model.
5. Steps 1-4 are repeated for the next week.

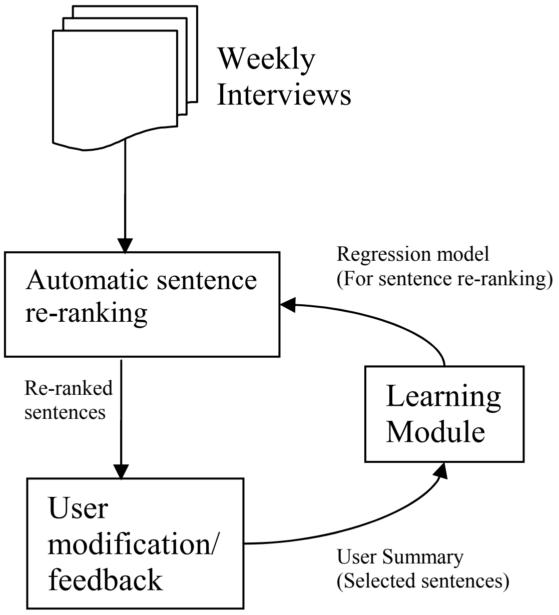
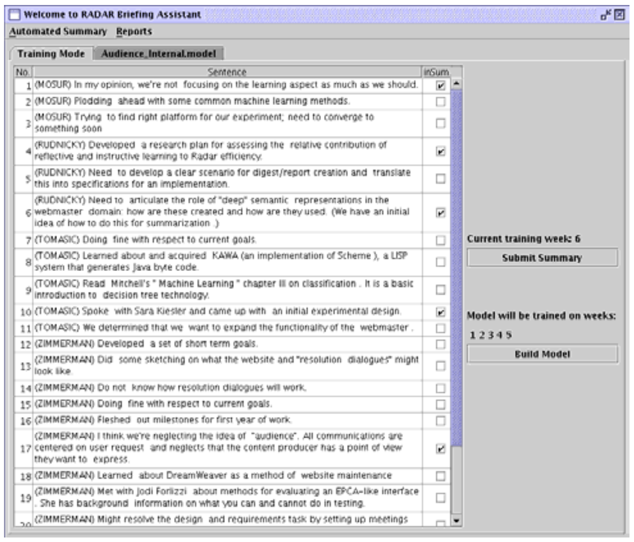

Weekly Interviews
Automatic sentence re-ranking
Regression model (For sentence re-ranking)
Re-ranked sentences
Learning Module
User modification/ feedback
User Summary (Selected sentences)

Fig. 2: Architecture of the Briefing Assistant. Demonstrates how user feedback is used to improve the summaries of future weeks.



No.	Sentence	InSum
1	(MOSUR) In my opinion, we're not focusing on the learning aspect as much as we should.	✓
2	(MOSUR) Plodding ahead with some common machine learning methods.	☐
3	(MOSUR) Trying to find right platform for our experiment; need to converge to something soon	☐
4	(RUDNICKY) Developed a research plan for assessing the relative contribution of reflective and instructive learning to Radar efficiency.	✓
5	(RUDNICKY) Need to develop a clear scenario for digest/report creation and translate this into specifications for an implementation.	☐
6	(RUDNICKY) Need to articulate the role of "deep" semantic representations in the webmaster domain: how are these created and how are they used. (We have an initial idea of how to do this for summarization .)	✓
7	(TOMASIC) Doing fine with respect to current goals.	☐
8	(TOMASIC) Learned about and acquired KAWA (an implementation of Scheme), a LISP system that generates Java byte code.	☐
9	(TOMASIC) Read Mitchell's " Machine Learning " chapter III on classification . It is a basic introduction to decision tree technology.	☐
10	(TOMASIC) Spoke with Sara Kiesler and came up with an initial experimental design.	✓
11	(TOMASIC) We determined that we want to expand the functionality of the webmaster .	☐
12	(ZIMMERMAN) Developed a set of short term goals.	☐
13	(ZIMMERMAN) Did some sketching on what the website and "resolution dialogues" might look like.	☐
14	(ZIMMERMAN) Do not know how resolution dialogues will work.	☐
15	(ZIMMERMAN) Doing fine with respect to current goals.	☐
16	(ZIMMERMAN) Fleshed out milestones for first year of work.	☐
17	(ZIMMERMAN) I think we're neglecting the idea of "audience". All communications are centered on user request and neglects that the content producer has a point of view they want to express.	✓
18	(ZIMMERMAN) Learned about DreamWeaver as a method of website maintenance	☐
19	(ZIMMERMAN) Met with Jodi Forlizzi about methods for evaluating an EPCA-like interface . She has background information on what you can and cannot do in testing.	☐
20	(ZIMMERMAN) Might resolve the design and requirements task by setting up meetings	☐

Fig. 3: A sample user session.

[1] The training data is collected *cumulatively*, that is, for ranking week n, the user selection from week 1 to week n-1 is used for training.

Identifying Features

Features play an important role in learning the user/audience model for summarization. We did a domain analysis and identified several *research-acts,* that is, acts that define the activities involved in a research project.

These research-acts were manually annotated[2] in the interview corpus and were used as features in the learning model. The final feature set consisted of 22 features that can be classified into three major groups. (Fig. 3)

1. **Action features**: These features indicate activities which take place in the project. For example, a project member learned about a new toolkit [INFO-GAIN], an issue which needs to be solved [ISSUE], a partial product like a prototype [PART-PRODUCT], acquiring a workstation computer [OBJECT-GAIN], etc.
2. **Named Entities**: These are mostly the proper nouns related to the project. For example, people involved in the project [PERSON], software names [SOFTWARE], organization names [ORGANIZATION], etc.

3. **Meta features (Author)**: It is possible that the interview content of some interviewees might be more suitable for a specific audience. For example, it is possible that interview content of a Principal Investigator may be more useful for preparing reports for an external audience such as a funding agency and the interview content of a Research Scientist might be used for preparing reports for the internal group. Hence, we also included the name of the interviewee as one of the features. This is the [AUTHOR] feature.

Although, these features were selected by doing a thorough domain analysis, we believe that feature sets may need to evolve over time (say as the nature of the project changes).

[2] We also report results with automatic annotation of features using approaches from information extraction. Another approach reported is using bag of words as features where no annotation is required.

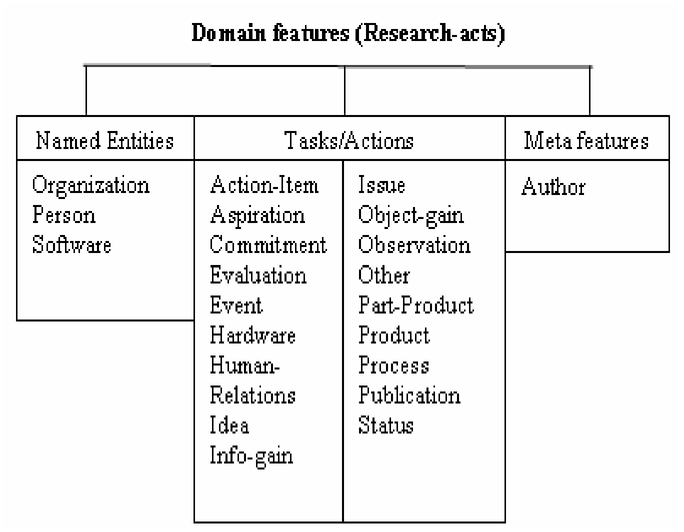


Fig. 4: Research-acts used as features in the learning model.

Supervised Learning

Feature representation: Each sentence in the interview was represented in the training data as a vector of integer valued features.

$$[\overline{F}] = < f_1, f_2, ..., f_{21}, f_{22} >$$

The feature value was the number of times the particular feature occurred in the sentence. For example, PERSON is an integer valued feature indicating how many persons were mentioned in the sentence. All the features are integer-valued except the [AUTHOR][3] feature which is a nominal feature.
The predicted class was binary indicating whether the sentence was selected in the human summary.

Training: In order to rank order items according to their importance, the model should provide a score for each item. Given the binary valued class attribute in the training data we used a logistic regression model to predict item rank. We use the summary class probability given by the logistic regression model as our item score. The Weka toolkit [Witten and Frank, 2000] was used for incorporating the logistic regression model in our system.

[3] [AUTHOR] is a nominal feature which was split into binary valued feature vector as [author$_1$ author$_2$ … author$_7$] for using it in the logistic regression model.

Evaluation Study

A total of seven subjects participated in the evaluation study. These were "expert" subjects who had a good knowledge of the research project. Each subject went through the task scenario described in the architecture section and the system tried to assist the subject by providing a good ranking every week for each of the 12 successive weeks.

Metrics: The system will lead to reduction in human effort if it can bring important items at the top and avoid the need for the user to go through the whole list in order to select a particular number of important items. Thus, we need to evaluate a ranked list given a binary prediction (important or not important) for each item in the list. We use the following metrics for evaluation:

1. Average Precision [Yates and Neto, 1999]: This metric measures the precision at every recall point and takes an average. For example, if the first summary item was at rank 6, the precision at this recall point is 1/6, if the second summary item is seen at rank 10, the precision at this recall point is 2/10 and so on .

2. Top 30%: This measures the percentage of summary covered in the top 30% of the list. For example, if the list contains 30 items and if 3 out of 5 summary items are in top 9 (30% of 30) then this metric value is 3/5 = 0.6

Both the above metric values increase as the ranking becomes better, thus the higher these values, the better the ranking.

Baseline: A good baseline is to measure how well the system does if there was no learning in the system. We can remove the learning and use the resulting system as a baseline in the following ways:

1. Random ordering: A simple idea is to provide the items in a random order and calculate the respective metrics. For random ordering the baseline values for average precision and top 30% were 0.27 and 0.26 respectively.

2. Standard summarization methods: We can also use the summarization methods developed for news wire summaries as they are based on a fixed heuristic. The baseline values using centroid based summarization [Radev et al., 2000] for average precision and top 30% were 0.3 and 0.36 respectively.
There are also other similar heuristic techniques for generating newswire summaries. For example, for newswire articles, one of the heuristics is to

give the first sentence and sometimes the concluding sentences, a relatively higher weight. Clearly, such heuristics will not work in our problem due to the lack of document structures. We also believe that the nature of the summaries in our domain is very user-specific and hence we need learning based techniques instead of techniques based on fixed heuristics.

We chose the baseline of random ordering as centroid based summarization does not provide much leverage. The random order simulation also provides a more robust estimate by averaging values over many simulations.

Evaluation Results:

The average precision and the top 30% values were noted for the rankings produced in each of the weeks. To attenuate the variance between individual weeks, we averaged the values for two consecutive weeks to get a bi-week value. Fig. 5 shows these bi-week values for each of the six bi-weeks. Each point in the plot is the mean bi-week values for all the seven subjects The increasing trend in the lines show the effect of learning as this trend implies that the system was able to produce successively better rankings given successively more user feedback.

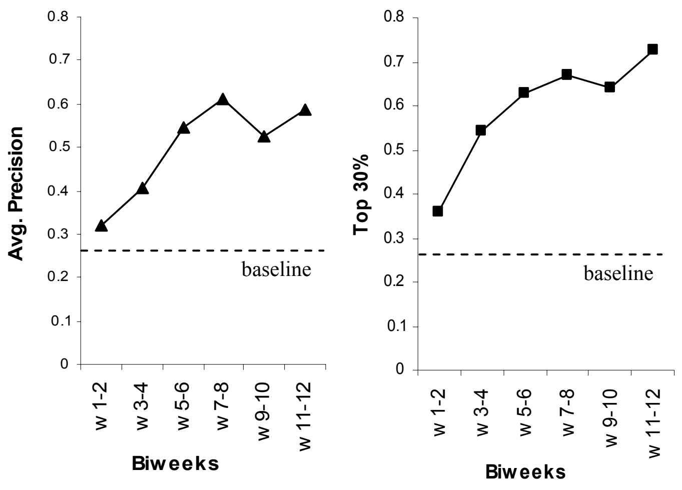


Fig. 5: Average precision and top 30% values for each of the six bi-weeks. Each point is the mean value across all subjects.

The nature of the items in bi-week 9-10 appears to have shifted and so the trained model needed to adjust. In general we should expect such shifts in dynamic activities such as research projects, where the nature of the items or the direction of the project will shift over time. The key metric is the speed with which the Briefing Assistant can react to such changes.

Measuring Elapsed Time:

Subjects were also timed during the experiment by asking them to use "start summary" and "submit summary" buttons while creating the summary for a particular week.[4] Thus, we could record the elapsed time, that is, the time taken to select all the summary items for a particular week.

Subjects showed a significant 22% decrease in task time over the 6 bi-weeks but we believe that it is premature to attribute the speed improvement due to assistance from the Briefing Assistant, as the subjects are likely to get better at the task with time since they are repeating the same task for every week. Showing speed improvements due to learning will likely require a cross-subject design, with one group getting the learning system and the other group not. Differences in time performance would point to gains due to the assistance provide by the system.

Assessing agreement among subjects:

The results demonstrate that the system learns subject's preferences and hence it does better as the learned model becomes better with more and more user feedback. But are the subjects really selecting different summary items ? We try to answer this question in the following ways:

Inter-rater agreement: This is used in a setting where there are n-raters that rate items into different categories and we want to measure the agreement in their rating. Kappa statistic is widely used in measuring inter-rater agreements. [Siegel and Castellan, 1988]. It is defined as follows:

$$K = \frac{P(A) - P(E)}{1 - P(E)}$$

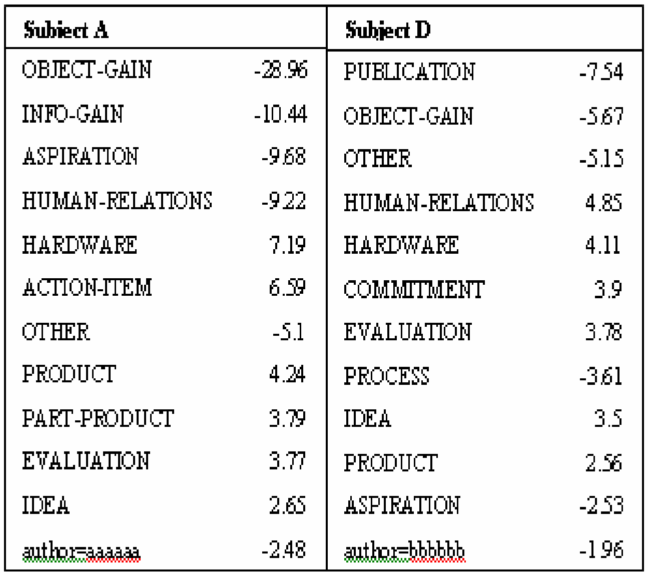
P(A) = Proportion of times n raters agree.
P(E) = Proportion of times we would expect the n raters to agree by chance

If there is complete agreement among raters, then K = 1 otherwise if there is no agreement (other than the agreement that is expected to occur by chance) then K = 0. The inter-rater agreement for the seven subjects that participated in our experiment was K = 0.26, which

[4] The timing data for one subject was not available.

indicates a very low agreement between the ratings of different subjects. Hence, different subject actually do produce different summaries.

Differences in feature weights: The differences in feature weights between the regression models of different subjects can also identify their differences with respect to the feature space[5]. The positive weights imply positively correlated features (with importance) and vice versa. Table 1 shows that some of the feature weights differ (in sign and value) for different subjects implying that the learned models are different.



Subject A		Subject D	
OBJECT-GAIN	-28.96	PUBLICATION	-7.54
INFO-GAIN	-10.44	OBJECT-GAIN	-5.67
ASPIRATION	-9.68	OTHER	-5.15
HUMAN-RELATIONS	-9.22	HUMAN-RELATIONS	4.85
HARDWARE	7.19	HARDWARE	4.11
ACTION-ITEM	6.59	COMMITMENT	3.9
OTHER	-5.1	EVALUATION	3.78
PRODUCT	4.24	PROCESS	-3.61
PART-PRODUCT	3.79	IDEA	3.5
EVALUATION	3.77	PRODUCT	2.56
IDEA	2.65	ASPIRATION	-2.53
author=aaaaaa	-2.48	author=bbbbbb	-1.96

Table 1: Top 12 features for 2 different subjects. Features are ranked according to absolute weight.

Automating feature extraction

The assistance framework described above is fairly general but the features used for learning were manually annotated in the corpus. In order to apply the assistance framework for other domains in the future, we need to have an automatic method of identifying features. We consider two approaches to overcome this problem:

1. Automatic annotation: Use information extraction approaches to automatically annotate the 22 domain features given manually annotated data as training.

2. Bag of words: Use features that do not require annotation such as bag of words. In this approach we use each word in the vocabulary of our corpus as a feature.

Automatic annotation: The corpus was first smoothed by removing stop words and stemming. As the corpus size was small, we used a jack-knifing approach to get training data for annotating rest of the features. A simple rote classifier was trained that just uses words present in the annotated spans to automatically annotate the test week. The three named-entity features were assumed to be known from domain as they could be simply annotated using a table of known domain terms.

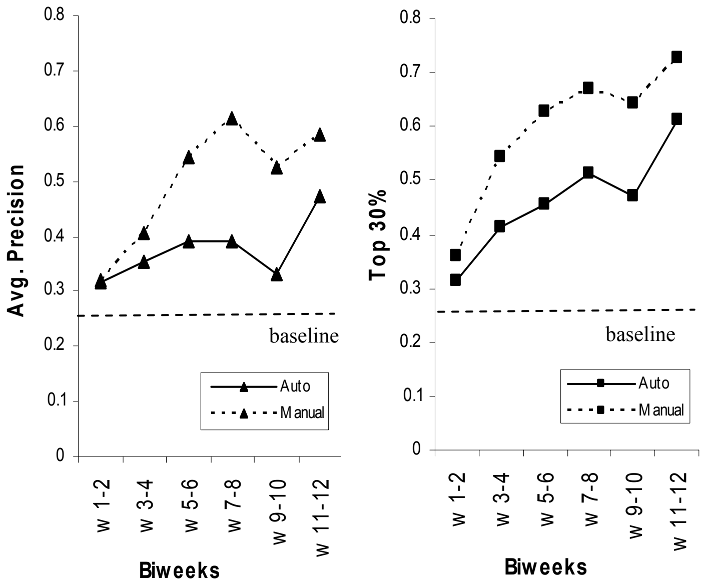


Fig. 6: Evaluation results for the original experiment using automatic annotation of 22 domain features.

We ran the previous experiment again to determine the effect due to the added noise in feature annotation. (Fig. 6) We can still see the increasing trend in the lines due to learning better with more and more with user feedback but the overall improvement in the metrics at the end of the learning period is less compared to manually annotated features. This is expected as some of the features might be incorrectly annotated. We also plan to

[5] Assuming that the features are uncorrelated.

incorporate more sophisticated information extraction approaches that make use of the context and structure near the annotated text span. [Cohen, 2004]

Bag of words: In this approach, all words present in the vocabulary of interview corpus are used as features. The words occurring only once (singletons) in the whole corpus were removed from the feature set. The feature values were the tf.idf weights as defined in Information Retrieval literature. In our case, the term frequency (tf) is the number of times a feature (word) occurs in the particular item and the inverse document frequency (idf) is inversely proportional to the number of times the feature (word) occurs in rest of the items in that week. The feature vectors were normalized using L1 norm. We used a voted perceptron classifier for learning user models due to the high dimensionality of feature space.

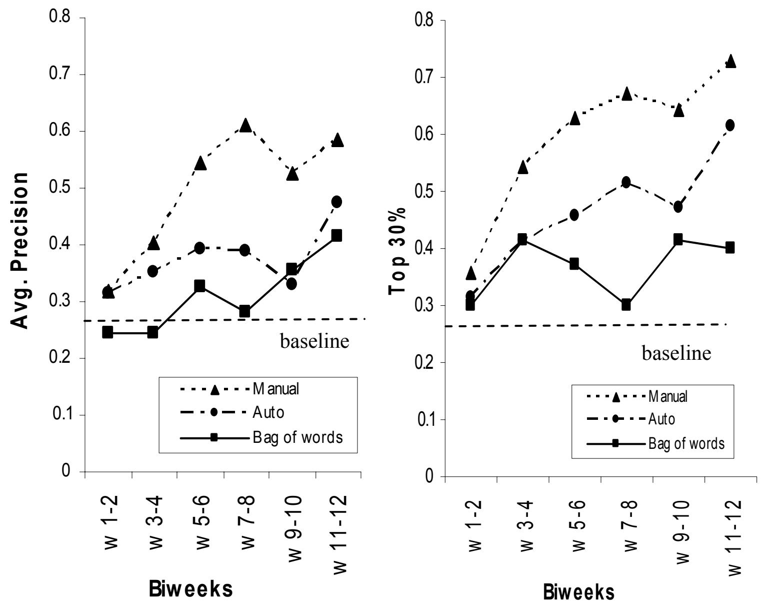


Fig. 7: Evaluation results for the original experiment using bag of words as features.

These results (Fig 7.) also show an increasing trend but the improvement is much lesser and implying that these features are inadequate. This may be also due to the small number of training examples which are not sufficient for learning in such a high dimensional feature space. Nevertheless, there is still improvement as compared to the baseline of no learning. The following table compares the metric values at the end of the learning period (6th biweek) for the briefing experiment using the different approaches for recognizing features.

	Metric values (6th biweek)	Improvement from baseline (factor)
Baseline	Avg. Prec: 0.27 Top 30%: 0.26	
Manual annotation	Avg. Prec: 0.586 Top 30%: 0.729	Avg. Prec: 2.17 Top 30%: 2.8
Auto-annotated	Avg. Prec: 0.473 Top 30%: 0.614	Avg. Prec: 1.75 Top 30%: 2.36
Bag of words	Avg. Prec: 0.413 Top 30%: 0.4	Avg. Prec: 1.53 Top 30%: 1.54

Table 2: Comparing different approaches of identifying features.

Future Challenges

Handling dynamic project situations: In our current training model, we accumulate the training data with every week. However, it is possible that to produce a weekly summary for the 12th week, the feedback given in 10th or 11th week might be more important than feedback given in 1st week. Decaying the importance of feedback or keeping a window of fixed size might be useful to take into account the dynamics of the project.

Learning by direct instruction: With a human in the loop, we could also provide a channel for direct instructions to the system. We are currently looking at providing this in the form of highlighting keywords while making the summary. Another interesting idea is allowing the user to provide rules that could directly be added to the learned model (assuming that classifier learns the model in form the form of rules. For example, a decision tree can be converted into a set of rules)

Active information acquisition: A human assistant not only learns different preferences for different situations but also knows how to actively acquire new relevant information. One of the solutions for the Briefing Assistant is to monitor different sources of data such as websites, meeting minutes and also have a dialog agent for conducting periodic interviews with project members for accumulating the relevant information. The relevance can be determined using the knowledge learned from user feedback.

Information Synthesis: Item selection or filtering is a first level of reporting. More useful reports involve synthesizing concise narratives from raw information. This may be possible to do through domain-specific form-based templates that are used to organize raw information into narratives graphs. Mappings between information and narrative elements would be learned through observation and instruction.
Once organized into narratives, actual text can be generated. More sophisticated generation strategies would allow the Briefing Assistant to express points of view (e.g., PI and PM versions of a report) and perhaps to generate text with particular persuasive goals.

Summary

We have described our work on a Personal Briefing Assistant whose goal is to assist report writers in creating briefings or summaries for different audiences in the academic domain. A learning based framework is proposed to address the needs for creating tailored reports and our evaluation study indicates a good potential for increase in user productivity. We also report encouraging initial results from our ongoing work on automatic feature extraction to be able to generalize this approach for different domains. There are still many interesting research problems to be solved in this application of cognitive assistants and we plan to proceed further with the future work outlined in the previous section.

References

[Amini, 2000] Amini M.-R. Interactive Learning for Text Summarization. Proceedings of the PKDD/MLTIA Workshop on Machine Learning and Textual Information Access. (PKDD/MLTIA 2000)

[Cohen, 2004] Cohen, William W. *Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data*, http://minorthird.sourceforge.net, 2004.

[Donaway *et al.*, 2000] Donaway, R. L., Drummey, K. W., and Mather, L. A. 2000. A Comparison of Rankings Produced by Summarization Evaluation Measures. Proceedings of the Workshop on Automatic Summarization, pp. 69-78.

[Edmundson, 1969] Edmundson, H.P. 1969. New methods in automatic abstracting. Journal of The Association for Computing Machinery, 16(2), pp. 264-285. Reprinted in Mani, I., and Maybury, M., eds., Advances in Automatic Text Summarization, MIT Press, pp. 21-42.

[Kupiec *et al.*, 1995] Kupiec J., Pedersen.,, and Chen F. A Trainable Document Summarizer. In Proceedings of ACM-SIGIR'95, Seattle, WA.

[Lin and Hovy, 1997] Lin, C.Y., and Hovy, E.H. Identifying Topics by Position", Proceedings of the Applied Natural Language Processing Conference, 1997.

[Mani and Bloedorn] Mani, I. and E. Bloedorn. 1998. Machine Learning of Generic and User-Focused Summarization. Proceedings of AAAI-98, pp. 821826.

[Luhn, 1958] H.P. Luhn. "The Automatic Creation of Literature Abstracts". IBM Journal of Research and Development, Vol. 2, No. 2, pp. 159-165, April 1958.

[Mani, 2001] Mani, I. Automatic Summarization. John Benjamins Publishing Company, 2001.

[Radev et al., 2000] Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *ANLP/NAACL Workshop on Summarization*, Seattle, WA, April 2000.

[Siegel and Castellan, 1988] S. Siegal and N. J. Castellan. *Non-Parametric statistics for the behavioural sciences*. McGraw Hill, 1988.

[Teufel and Moens, 1998] Teufel, S. and Moens, M, Sentence extraction and Rhetorical Classication for Flexible Abstracts", in Working Notes of the AAAI Spring Symposium on Intelligent Text Summarization, Spring 1998, Technical Report, AAAI, 1998.

[Witten and Frank, 2000] Witten, Ian and Frank, Eibe Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco, 2000.

[Yates and Neto, 1999] Yates, R.B., Neto,B.R.: *Modern Information Retrieval*. ACM Press (1999)

[Zhang *et al.*, 2003] Zhang H., Zheng C., Wei-ying M. and Qingsheng C. A Study for Document Summarization Based on Personal Annotation. HLT-NAACL 2003 Workshop: Text Summarization (DUC 03), pp 41—48