CC Prediction in the Enron Corpus

Student: Vitor R. Carvalho
11-742: Information Retrieval Lab Report
Language Technologies Institute
Carnegie Mellon University
Fall 2006

Abstract

Email is the most popular communication tool on the web. To improve the way we handle email messages, machine learning techniques have been proposed in different areas, from adaptive spam filtering to automated message foldering (i.e., predicting the correct folder to store a message). One of the ideas recently proposed is to automatically predict the recipients of an already composed message; a problem also referred to as CC Prediction problem. The goal of this problem is to predict the most likely recipients for a message, given its current text and given the email addresses already specified. If successfully automated, this idea can be a valuable addition to email clients, particularly in the large corporations. It can prevent a user from forgetting to add an important collaborator or manager as recipient. Also, it can be used to identify people in an organization that are working in a similar topic or project, or to find people with appropriate expertise or skills. In this work we investigate this problem using the textual contents and social network information features from a large real world collection of email messages, the Enron Email corpus [14]. Using a classification-based reranking scheme to combine the two different types of features, experiments indicate that we can correctly predict CC-recipients in more than 56% of the test cases.

1 Introduction and Related Work

Machine learning techniques have been successfully applied to email communication recently. Some of the most well known applications are adaptive spam filtering [8], email foldering [13], email filtering [18], automatic integration with address book [9] and automatic integration with to-do lists [4]. In this work we focus on another application to improve email clients: automatically predicting email recipients, a.k.a. the CC Prediction problem. The CC Prediction problem is the task of predicting the recipients (TO, CC or BCC) of an email message already composed. The prediction is typically based on the textual contents of the message, as well as on the presence of other email addresses as recipients [17].

The CC Prediction problem is closely related to the Expert Finding task in Email. The

Expert finding in email is defined as the task of finding expertise using only the email messages exchanged inside an organization, as described by Dom et al. [10], Balog et al. [3, 2], Campbell et al. [6] and Sihn an Heeren [20]. This area of research has received considerable attention from the TREC community, and an expert-finding task has been run under the TREC Enterprise track since 2005.

Another important area of work related to this problem is collaborative filtering or recommending systems [5][15]. The CC Prediction problem can be seen as a special type of recommendation system where email recipients are recommended based on the current message being composed and the previous messages exchanged among users.

The CC Prediction problem was initially described by Pal & McCallum [17], where Naive Bayes and Factor Graphs models were used to predict recipients in a personal collection of emails. In their paper, performance was evaluated in terms of an accuracy-like metric in the top 5 candidates (prediction was considered correct if contained within the top 5 recipients predicted by the model). In this work we focus on the task of predicting CC and BCC addresses given two types of information: the textual message and the addresses already included in the TO field. We intend to extend Pal & McCallum's ideas in several ways: we implement different baseline algorithms for the problem, we test the algorithms on a considerably larger real data collection (the Enron Email Corpus [14]), we combine textual and social network features using a reranking scheme and evaluate the performance using a more comprehensive set of metrics.

There are a number of different algorithms to address the CC prediction problem. Some of the baselines we plan to add are: (1) a random baseline, (2) a simple TF-IDF cosine similarity between user's communications, (3) a learning algorithm (KNN) score between the current message and previous messages and (4) a classification-ranking method using (2) or (3) as base score in addition to non-textual features such as relative frequency of sent messages, received messages, co-occurrence of recipients, etc.

The main motivation to attempt this problem in the Enron Dataset is that, since email is a noisy and the email usage varies considerably among users, it is important to study this problem on a larger collection of data. In other words, we expect that the CC Prediction task will vary considerably among different Enron users. Not only do different users store distinct amounts of messages, but also different ways to use email (for instance, secretaries typically exchange email messages with more people than technical positions).

In order to obtain labeled data, a straightforward procedure is to remove one or more of the "true" recipients from the original message, and then consider this address as the label to be predicted. The output of the algorithms will output a ranked list with the most likely recipients for each message in a test collection. To evaluate performance, we intend to use three different methods: precision at rank 1 (accuracy), average precision and precision-recall curves. These methods in combination should reveal more details of the task than the evaluation criterion used in [17].

This report is organized as follows. In Section 2 we introduce the dataset and the preprocessing steps utilized in this report. In Section 3 we explain the two main ideas of this report: the textual-based approaches to the CC Prediction are presented in Section 3.1 and in Section 3.2 we use a classification-based reranking scheme to accommodate social network features to the problem. We finish by presenting our conclusions in Section 4.

2 Enron Corpus and Preprocessing

Although email is ubiquitous, large, public and realistic email corpora are not easy to find. The limited availability is largely due to privacy issues. For instance, in most US academic institutions, an email collection can only be distributed to researchers if all senders of the collection also provided explicit written consent.

In all experiments of this paper we used the Enron Email Corpus, a large collection of real email messages from managers and employees of the Enron Corporation. This collection was originally made public by the Federal Energy Regulatory Commission during the investigation of the Enron accounting fraud. We used the Enron collection to create a number of simulated user email accounts and address books, as described below, on which we conducted our experiments.

As expected, real email data have several inconsistencies. To help mitigate some of these problems, we used the Enron dataset version compiled by Jitesh and Adibi [19], in which a large number of repeated messages were removed. This version contains 252,759 messages from 151 employees distributed in approximately 3000 folders.

Another particularly important type of inconsistency in the corpus is the fact that a single user may have multiple email addresses. We addressed part of these inconsistencies by mapping between 32 "raw" email address and the normalized email address for some email users. This mapping (author-normalized-author.txt) was produced by Andres Corrada-Emmanuel, and is currently available from the Enron Email webpage [7].

For each Enron user, we considered two distinct sets of messages: messages sent by the user (*sent collection*) and messages received by the user (*received collection*). The received collection contains all messages in which the user's email address was included in the *TO*, *C.C.* or *B.C.C.* fields. The sent collection was sorted chronologically and then split into two parts, *sent_train* and *sent_test*. *Sent_train* contains 90% messages sent by the user, corresponding to the oldest ones. The most recent messages, 10% of the total sent collection, were placed in *sent_test*. The final message counts for 20 target Enron users is illustrated in Table 1.

Enron user	**received**	**sent_train**	**sent_test**	**sent_test***
hyatt	1797	566	63	13
germany	466	729	82	27
white	922	441	50	12
whitt	836	414	46	13
geaccone	889	396	44	14
kaminski	1042	1097	122	17
kitchen	5681	876	98	49

Table 1: Number of Email Messages in the Different Collections. **sent_test*** contains only messages with addresses in the TO and in the CC fields.

This 90%/10% split was used to simulate a typical scenario in a user's desktop — where the user already has several sent and received messages, and the goal is to predict the recipients of the next sent message. In order to make the received collection consistent with this, we removed from it all messages that were more recent than the most recent message in

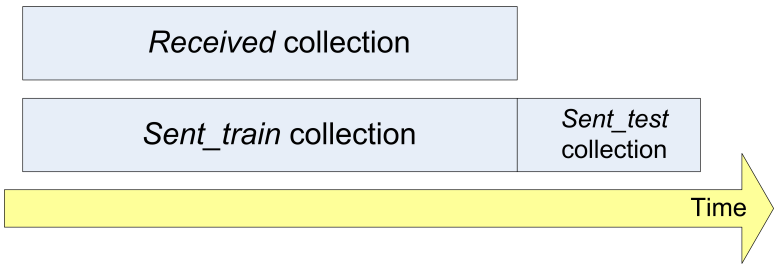

Figure 1: Time frames of different email collections.

sent_train. The general time frames of the different email collections is pictured in Figure 1.

We tried to this same train-test splitting with 20 different Enron users, but unfortunately only the six users shown in Table 1 had a reasonable number of valid messages (messages with non-empty TO and CC fields). We also simulated each user's address book: for each Enron user u, we build an address book set $AB(u)$, which is a list with all email addresses that were addressed by the user — these addresses were extracted from all recipients fields (TO,CC,BCC) in the *sent_train* collection.

In all our experiments we represented the content of the messages as a "bag of words", where the counts of all tokens in a message were extracted and taken as feature weights. In this process, a small set of stop words[1] was removed from the email body. In addition, self-addressed messages with no other recipients were also disregarded.

3 Methods for CC Prediction

3.1 Baselines: Using Textual Content

In this Section we develop different techniques for the CC prediction problem based on the textual contents of the messages. The main idea here is to model the "recipient-message" pairs, and then rank the most likely pairs according to some measure of confidence. Perfect performance would be translated in true cases of CC in the top of the rank.

The first method was based on cosine similarity between two vector-based representations of email messages. Given a message from user u to a set of recipients A (in this case, the union of the TO, CC and BCC sets), we derived the message's TF-IDF (Term Frequency-Inverse Document Frequency) vector representation from its textual contents and then normalized the vector to length 1.0. The second representation was built from a concatenation of all previous messages sent from user u to a particular recipient a_i in A. In other words, we concatenated the normalized TF-IDF representations of all previous messages sent from u to a_i and considered it to be one single large document. We then computed the cosine similarities between the current message vector and the $|A|$ concatenated vectors. For each test message, the recipients then are ranked according to this cosine similarity. A post-processing step removes all TO addresses from the rank. We refer to this method as *Centroid*.

[1] about, all, am, an, and, are, as, at, be, been, but, by, can, cannot, did, do, does, doing, done, for, from, had, has, have, having, if, in, is, it, its, of, on, that, the, they, these, this, those, to, too, want, wants, was, what, which, will, with, would

The second method was based on the K-Nearest Neighbors algorithm described by Yang & Liu [21]. Given a message from user u addressed to a set of recipients $A = \{a_1..a_n\}$, we found its 30 most similar messages in the training set. The notion of similarity here is also defined as the cosine distance between the text of two normalized TF-IDF vectors. With the top 30 most similar messages selected from the training set, we then computed the weight of each recipient a_i according to the sum of similarity scores of the messages in which a_i was one of the recipients. All recipients are ranked according to this final weight and, as a postprocessing step, the addresses in the TO set are removed from the ranking. We refer to this method as *Knn-30*.

The overall results in this section are shown in Tables 2 and 3. These Table shows the experimental results for each of the seven Enron users. In Table 2, results are expressed in terms of Precision at rank 1 (or Prec@1), i.e., the number of times that the predicted top recipient is an actual CC or BCC address. In Table 3, we illustrate the Average Precision score [1]. Additionally, in both Tables we included a random baseline — it shows the performance when ranking of addresses is chosen randomly from the address book, averaged over 20 trials.

From Table 2 and 3 we observe that, in average, the Centroid method presents a clearly better performance than the random baseline. Also, the Knn-30 method outperforms both the Random baseline and the Centroid methods in average. In the next section, we will improve the Knn-30 method by adding social network features to the problem.

It is worth noticing that, besides the three methods implemented in Tables 2 and 3, we also implemented linear classification models to this problem (Maxent and Voted-Perceptron) in a one-versus-all scheme. Unfortunately, these two methods were considerably more expensive than the Centroid and Knn-30 methods because the number of classes to be learned in the one-vs-all scheme was very large (number of email addresses in the address book). Indeed, the problem of efficient learning when the numbers of classes or concepts is very large is an interesting research problem in its own. A recent approach and survey to it has been presented by Madani & Greiner [16].

Enron user	**Random**	**Cosine**	**Knn-30**
germany-c	0.022	0.333	0.556
kaminsky-v	0.035	0.412	0.647
geaccone-t	0.014	0.000	0.357
kitchen-l	0.009	0.653	0.245
whitt-m	0.015	0.077	0.462
white-s	0.016	0.417	0.417
hyatt-k	0.011	0.308	0.615
Mean	0.017	0.314	0.471

Table 2: Email CC Prediction Results: **Prec@1**.

3.2 Classification-Based Method: Using Social Network Information

So far we have considered only the textual contents of emails in the task of leak prediction. Yet, it is reasonable to consider social network features for this problem, such as the number of received messages, number of sent messages, number of times two recipients were

Enron user	**Random**	**Cosine**	**Knn-30**
germany-c	0.043	0.415	0.531
kaminsky-v	0.049	0.537	0.683
geaccone-t	0.027	0.135	0.320
kitchen-l	0.013	0.616	0.338
whitt-m	0.020	0.403	0.603
white-s	0.033	0.451	0.485
hyatt-k	0.018	0.458	0.714
Mean	0.029	0.431	0.525

Table 3: Email CC Prediction Results: **Average Precision.**

copied in the same message, etc. In this Section we describe how these network features can be exploited to improve the performance on this problem.

In order to combine textual and social network features, we used a classification-based scheme. The idea is to perform the CC prediction in two steps. In the first step we calculate the textual similarity scores using a cross-validation procedure in the training set. In the second step, we extract the network features and then we learn a function that combines those with the previously calculated textual scores.

The textual scores are calculated in the following way. We split the training set (sent_train collection) in 10 parts. Using a 10-fold cross-validation procedure, we compute the score for the knn-30 on 90% of the training data and use it to make predictions in the remaining 10%. In the end of this process, each training set examples will have, associated with it, a list of email addresses (from the top 30 messages selected by Knn-30) and their predicted scores. Now we have, for each message in the training set, a score associated with each recipient in the address book. These scores will be used as features in the second step of the classification procedure.

In addition to the textual scores, we used two different sets of social network features. The first set is based on the relative frequency of a recipient's email address in the training set. For each recipient we extracted the normalized sent frequency (i.e., the number of messages sent to this recipient divided by the total number of messages sent by this particular Enron user) and the normalized received frequency (i.e., the number of messages received from this recipient divided by the total number of messages received by this particular Enron user). We refer to these features as *Frequency* features.

The second set of social network information is based on co-occurrence with the TO-recipients of the same message. The intuition behind this feature is that we expect related CC-recipients to co-occur more frequently with the recipients already addressed in the TO field of the email message. Given a message with three TO-recipients to_1, to_2 and to_3, and a given CC candidate address cc_i, let the frequency of co-occurrence between recipients cc_i and to_j be $F(to_j, cc_i)$ (i.e., the number of messages in the training set that had to_j as well as cc_i as recipients). Then, for a given message, the *Relative Recipient Frequency* (or RRF) of cc_i with the TO-recipients will be:

$$RRF_i = \frac{\sum_j F(to_j, cc_i)}{F(cc_i)}$$

where $F(cc_i)$ is the number of times the address cc_i received a message in the training set.

The second type of social network based feature is the *Relative Co-Occurrence Frequency* (or RCF). For a given message with J TO-recipients, RCF is defined as the percentage of the TO-users that ever co-occurred in the training set with the current CC user. Obviously the RCF feature are used only when the number of addresses in the TO field (i.e., J) is larger than one. We refer to the RCF and RRF features as the *Cooccurrence* features.

In order to combine the textual features (from the 10-fold crossvalidation procedure) with the social network based features (*Frequency* and *Cooccurrence* features), we used a classification-based reranking scheme. More specifically, we use the confidence of a classifier as ranking score — a scheme also know as Direct Re-Ranking by Classification [12].

We used the Voted Perceptron [11] as learning algorithm, as an example of a learning method which is robust and effective, but efficient enough to be plausibly embedded in an email client. It was trained using five passes through the same training data, and training examples for each Enron user's were generated from the entire training collection for the particular Enron user. The learning proceeded in the following way. For each training message with J recipients (TO+CC+BCC), we created N binary examples (N is the number of emails in the address book): $N - J$ negative example with the features associated with non-recipients addresses and J positive examples associated with the true recipients. The CC prediction thus becomes a binary classification problem, where the final ranking score will be determined by the classifier's confidence. It is important to point that, as a post-processing step during the testing phase, addresses in the TO list are removed from the final ranking.

Experimental results using the textual and network features are illustrated in Tables 4 (Prec@1) and 5 (Average Precision). For comparison, the second column presents Knn-30 results, from the best text-only method from Tables 2 and 3. The third column shows the Prec@1 values of our method using the cross-validation score in addition to the Frequency features only. The forth column displays results associated with experiments were the cross-validation scores are used in combination with the Cooccurrence features only. As we can see, results indicate some performance improvements by using the Frequency features reranked. The fifth column shows results when all features are used. The last columns indicates the relative gain of the *"+All"* column compated with the Knn30 results.

Tables 4 and 5 clearly show that the reranking scheme with the social network features improve performance for the CC prediction task. When compared to the textual-only baselines, the addition of both types of network features on average seem to improve predictions independently. While the Frequency features impose a considerable gain, the Cooccurrence features do not cause a significant improvement in performance. When combined, the system reached the best results overall: on average, more than 56% of accuracy (Prec@1) and 58% of average precision on the test messages — a clear indication of the success of the approach.

Even though the reranking scheme deteriorated the performance for some users, on average results in Tables 4 and 5 are a clear indication that the proposed method is very effective to predict CC addresses.

Figure 2 shows the seven Precision-Recall curves associated with the experiments in Tables 4 and 5. Qualitatively speaking, it shows the same behavior as Tables 4 and 5, where using the social network based features improved performance for the task when compared to the other textual based approaches.

Enron user	Knn-30	Reranking Scheme			Δ(%) (From +All to Knn-30)
		+Frequency Features	+Cooccurrence Features	+All Features	
germany-c	0.556	0.593	0.593	0.556	00.00
kaminsky-v	0.647	0.588	0.647	0.588	-09.10
geaccone-t	0.357	0.429	0.357	0.357	00.00
kitchen-l	0.245	0.571	0.286	0.714	191.40
whitt-m	0.462	0.538	0.462	0.538	16.50
white-s	0.417	0.417	0.417	0.417	00.00
hyatt-k	0.615	0.769	0.615	0.769	25.00
Mean	**0.471**	**0.558**	**0.482**	**0.563**	**32.0**

Table 4: CC Prediction Results: **Prec@1**.

Enron user	Knn-30	Reranking Scheme			Δ(%) (From +All to Knn-30)
		+Frequency Features	+Cooccurrence Features	+All Features	
germany-c	0.531	0.520	0.519	0.472	-11.11
kaminsky-v	0.683	0.643	0.678	0.642	-06.00
geaccone-t	0.320	0.358	0.367	0.373	16.56
kitchen-l	0.338	0.571	0.360	0.624	84.61
whitt-m	0.603	0.652	0.623	0.669	10.94
white-s	0.485	0.498	0.475	0.485	00.00
hyatt-k	0.714	0.812	0.729	0.812	13.72
Mean	**0.525**	**0.579**	**0.536**	**0.582**	**15.53**

Table 5: CC Prediction Results: **Average Precision**.

4 Concluding Remarks

This report addressed the email CC prediction problem: how to predict recipients of email messages. This task can be a valuable addition to email clients, particularly in the large corporations — it can prevent a user from forgetting to add an important collaborator or manager as recipient. Also, it can be used to identify people in an organization that are working (or have worked) in a similar topic or project, or that have an expertise in a specific subject.

We proposed several methods to automate this task. By combining textual-based features with social-network features, the best scheme was able to reach more than 56% of accuracy and 58% of average precision on test messages. These results are encouraging also because the proposed solution can be easily incorporated in any email client — with no changes in the email server side.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Addison-Wesley, 1999. [Online]. Available: http://sunsite.dcc.uchile.cl/irbook

[2] K. Balog, L. Azzopardi, and M. de Rijke, "Formal models for expert finding in enterprise corpora," in *SIGIR 2006: Proceedings of the 29th Annual International ACM*

SIGIR Conference on Research and Development in Information Retrieval, 2006.

[3] K. Balog and M. de Rijke, "Finding experts and their eetails in e-mail corpora," in *Proceedings of the 15th international conference on World Wide Web, WWW 2006*, 2006, pp. 1035–1036.

[4] P. N. Bennett and J. Carbonell, "Detecting action-items in e-mail," in *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005, pp. 585–586.

[5] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *UAI*, 1998.

[6] C. S. Campbell, P. P. Maglio, A. Cozzi, and B. Dom, "Expertise identification using email communications," in *CIKM*, 2003.

[7] W. W. Cohen, *Enron Email Dataset Webpage*, http://www.cs.cmu.edu/ enron/.

[8] G. Cormack and T. Lynam, "On-line supervised spam filter evaluation," *ACM Transactions on Information Systems*, 2006.

[9] A. Culotta, R. Bekkerman, and A. McCallum, "Extracting social networks and contact information from email and the web," in *CEAS*, 2004.

[10] B. Dom, I. Eiron, A. Cozzi, and Y. Zhang, "Graph-based ranking algorithms for e-mail expertise analysis," in *Data Mining and Knowledge Discovery Workshop(DMKD2003) in ACM SIGMOD*, 2003.

[11] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," *Machine Learning*, vol. 37, no. 3, pp. 277–296, 1999.

[12] H. Ji, C. Rudin, and R. Grishman, "Re-ranking algorithms for name tagging," in *HLT/NAACL 06 Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, 2006.

[13] B. Klimt and Y. Yang, "The enron corpus: A new dataset for email classification research," in *ECML*, 2004.

[14] ——, "Introducing the enron corpus," in *First Conference on Email and Anti-Spam*, 2004.

[15] G. Lebanon, "A c/matlab toolkit for collaborative filtering," 2003.

[16] O. Madani and W. Greiner, "Learning when concepts abound," Yahoo! Research, Tech. Rep. CMU-LTI-06-002, 2006.

[17] C. Pal and A. McCallum, "Cc prediction with graphical models," in *CEAS*, 2006.

[18] J. Rennie, "ifile: An application of machine learning to e-mail filtering," in *Proc. KDD00 Workshop on Text Mining*, 2000.

[19] J. Shetty and J. Adibi, "Enron email dataset," USC Information Sciences Institute, Tech. Rep., 2004, available from http://www.isi.edu/ adibi/Enron/Enron.htm.

[20] W. Sihn and F. Heeren, "Expert finding within specified subject areas through analysis of e-mail communication," in *Proceedings of the Euromedia 2001*, 2001.

[21] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *22nd Annual International SIGIR*, August 1999, pp. 42–49.

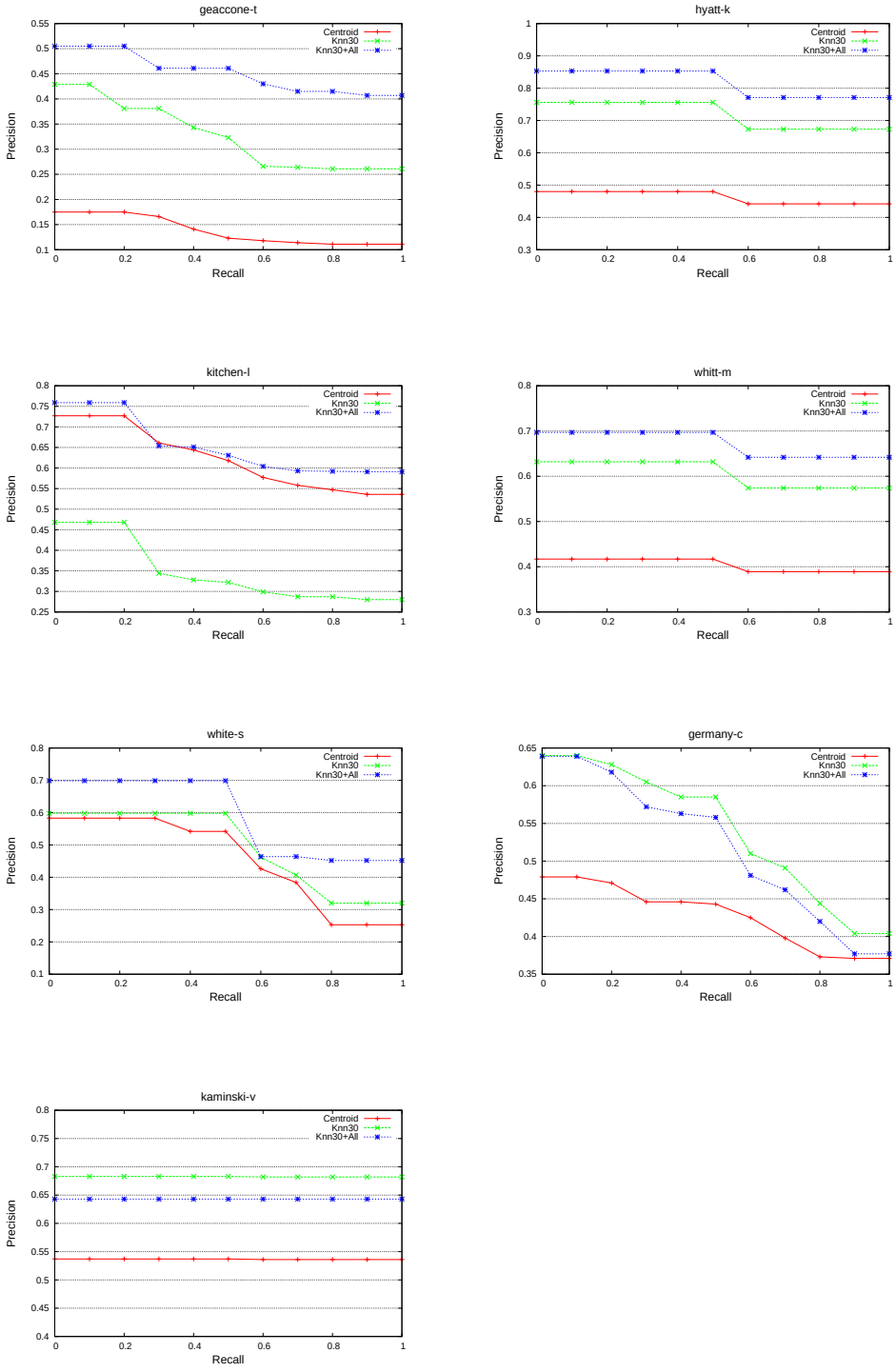

Figure 2: Precision-Recall Curves