Using Temporal Profiles of Queries for Precision Prediction

Fernando Diaz*
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
fdiaz@cs.umass.edu

Rosie Jones
Yahoo! Research Labs
74 N Pasadena Ave, 3rd Floor
Pasadena, CA 91103
jonesr@yahoo-inc.com

ABSTRACT

A key missing component in information retrieval systems is self-diagnostic tests to establish whether the system can provide reasonable results for a given query on a document collection. If we can measure properties of a retrieved set of documents which allow us to predict average precision, we can automate the decision of whether to elicit relevance feedback, or modify the retrieval system in other ways. We use meta-data attached to documents in the form of time stamps to measure the distribution of documents retrieved in response to a query, over the time domain, to create a temporal profile for a query. We define some useful features over this temporal profile. We find that using these temporal features, together with the content of the documents retrieved, we can improve the prediction of average precision for a query.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation*

General Terms

Algorithms, Experimentation, Theory

Keywords

time, clarity, precision prediction, language models

1. INTRODUCTION

Many document collections, such as email and news, have a timestamp attached to each document. Although many systems have been evaluated using such corpora, few of these systems have explicitely considered this temporal information.

*This research was carried out while this author was at Yahoo! Research Labs

SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK.

Swan and Jensen [9] studied temporal information in the context of retrieving Topic Detection and Tracking topics from a stream of news. The authors perform feature discovery with the matching documents, and characterize the queries into types, based on the number of features found by TimeMines. They describe those queries for which TimeMines found many features as having many subtopics. They suggest that queries for which few or no features were discovered can be described as topics which are fairly static.

Li and Croft proposed the inclusion of temporal evidence into the document prior of a language modeling retrieval system [8]. The authors first manually triaged TREC queries into temporal classes based on the distribution of known relevant documents. Queries in particular classes were then given different document priors. This work demonstrates how to incorporate temporal information if we know which temporal class a query belongs to.

We would like to investigate the role of time when the system is ignorant of the true class or temporal distribution of a query. To this end, we consider the task of *precision prediction*. Cronen-Townsend et al. introduced content clarity [3] as a content-based method for predicting system performance given a query. We expand on this work by adding time to the content features.

The remainder of this paper is organized as follows. We begin in Section 2 by describing the estimation of temporal profiles for arbitrary queries. In Section 3, we define a set of features over these temporal profiles used in performance prediction. In Section 4, we describe the TREC collections and regression models used in our experiments. We conclude in Section 5 by showing that we improve precision prediction over the consideration of content-based features alone.

2. TEMPORAL PROFILES

One way to analyze a query is to look at the type of documents it retrieves. This can be accomplished by inspecting the top N documents of an initial retrieval and calculating the statistical properties of terms occurring in this set of documents [3, 7]. In a language modeling context, we rank the documents in the collection according to their likelihood of having generated the query:

$$P(Q|D) \quad = \quad \prod_{w \in V} P(w|D)^{q_w} \qquad (1)$$

Here, q_w is the number of times the word w occurs in the query. Document language models, $P(w|D)$, are estimated using the words in the document [2]. Using this ranking, we

can build a query language model, $P(w|Q)$, out of the top N documents,

$$P(w|Q) = \sum_{D \in R} P(w|D) \frac{P(Q|D)}{\sum_{D' \in R} P(Q|D')} \quad (2)$$

where R is the set of top N documents and there is a uniform prior over the documents.

We are interested in describing the *temporal* nature of a query. Thus we wish to examine a *temporal profile* of the query, by analogy with the content-based profile described above. Our temporal query model is initially defined as

$$\tilde{P}(t|Q) = \sum_{D \in R} \tilde{P}(t|D) \frac{P(Q|D)}{\sum_{D' \in R} P(Q|D')} \quad (3)$$

where the granularity is on the day scale and

$$\tilde{P}(t|D) = \begin{cases} 1 & \text{if } t \text{ is equal to the document date, } t_D \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

It is often helpful to smooth maximum likelihood models such as $\tilde{P}(t|Q)$. We used a two-stage process to smooth our models. First, we smoothed $P(t|Q)$ with a background model. Background smoothing plays two roles. Firstly, background smoothing handles potential irregularities in the collection distribution over time. For example, certain dates may have a large number of articles compared to others. Secondly, background smoothing replaces zero probability events with a very small probability, allowing us to assign a very small likelihood of a topic being discussed on days where we have no explicit evidence. We use the distribution of the collection over time as a background model. This collection temporal model is defined by

$$\tilde{P}(t|C) = \frac{1}{|D|} \sum_{D} \tilde{P}(t|D), \quad (5)$$

Our estimate can then be linearly interpolated with this reference model such that

$$P'(t|Q) = \lambda \tilde{P}(t|Q) + (1 - \lambda) \tilde{P}(t|C) \quad (6)$$

Since our model is discrete at the level of a single day, and news stories on a single topic may occur over a period of several days, we smooth our estimate of the model for a single day with the model for adjacent days. These kinds of smoothing techniques have been explored in the field of time series analysis. We use simple moving average smoothing. The smoothed estimate for a particular day is defined according to the previous p days,

$$P(t|Q) = \frac{1}{\phi} \sum_{i=0}^{\phi - 1} P'(t - i|Q) \quad (7)$$

In our experiments, the period, ϕ, is always 14, smoothing the probability for a day with the 14 preceding days, but not subsequent days. Improvements could be made by smoothing with days both before and after the reference day. $P(t|Q)$ is our final estimate of the distribution.

Figure 1 shows the temporal profile for the query, "leveraged buyouts".

3. FEATURES OF TEMPORAL PROFILES

In the previous section, we described the estimation of temporal profiles. Figure 2 depicts the temporal profile for the query "poaching" over the AP88-89 corpus. Expectedly, this profile represents a query with relatively uniform representation in the corpus. This stability is clearer when compared to the temporal profile for the query "hostage taking" shown in Figure 3. In this section, we will define a set features for discriminating between temporal profiles. These features will then be used in Section 5 to predict the precision of queries.

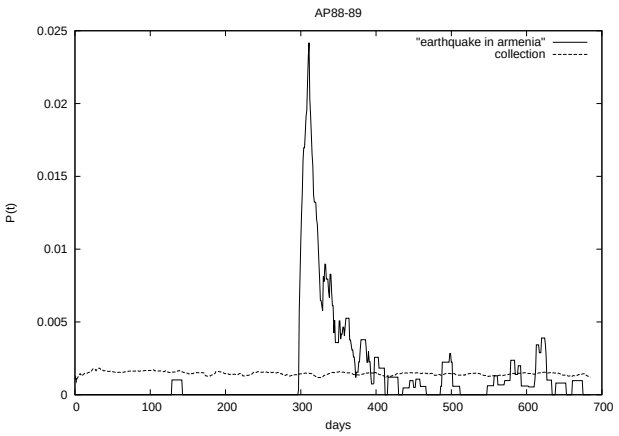


Figure 1: Temporal profile, $P(t|Q)$, of the query "earthquake in armenia" using the AP88-89 collection. The collection distribution, $P(t|C)$, is shown for reference.

3.1 Kullback-Leibler Divergence

In language model based information retrieval, query clarity [3] is meant to capture the effectiveness of the query at retrieving a precise topic. This content clarity measure assumes that the distribution of words in documents retrieved for a good query will be distinct from the background distribution. The clarity measure is defined as the Kullback-Leibler (KL) divergence between the query language model $P(w|Q)$ and the collection language model. Formally, the clarity score is defined as,

$$D_{\mathrm{KL}}(P(w|Q), P(w|C)) = \sum_{w \in V} P(w|Q) \log \frac{P(w|Q)}{P(w|C)} \quad (8)$$

A larger KL divergence indicates a clearer query. We will refer to this clarity measure as *content clarity*.

We propose an analog to content clarity for the temporal domain, by measuring the difference between the distribution over time of documents retrieved in response to a query, and the distribution over time of documents in the collection as a whole. This can be quantified by taking the KL divergence between the collection temporal model and the query temporal model. That is,

$$D_{\mathrm{KL}}(P(t|Q), P(t|C)) = \sum_{t=1}^{T} P(t|Q) \log \left(\frac{P(t|Q)}{P(t|C)} \right) \quad (9)$$

We will refer to this feature as temporal KL divergence, or temporalKL. The spiky nature of our example query, "hostage taking" (Figure 3) is clearly captured by this feature. At the same time, the relatively a-temporal query, "poaching", exhibits a much lower KL divergence.

Note that although temporalKL shows the deviation of documents retrieved for a query from the general distribution of documents over time, it may not allow us to distinguish between queries corresponding to events taking place at a single time, (such as "turkish earthquake 1999") and temporally ambiguous queries (such as "iraq war").

3.2 Autocorrelation

While the KL divergence gives us a test of similarity to the temporal background model ($P(t|C)$), it does not provide a measure of the randomness of the query time series. To test this, we use the first-order autocorrelation of the time series,

$$r_1 = \sum_{t=1}^{T-1} \frac{(P(t|Q) - \frac{1}{T})(P(t+1|Q) - \frac{1}{T})}{\sum_{t=1}^{T}(P(t|Q) - \frac{1}{T})^2} \quad (10)$$

The autocorrelation of a uniform distribution is $r_1 = 0$. A high autocorrelation value suggests a structure to the time series. This will be the case for queries which contain a strong inter-day dependency. For example, autocorrelation is high in cases where a high $P(t|Q)$ tends to predict a high $P(t+1|Q)$; likewise with low values. Such behavior indicates that there is predictability to the time series.

In Figure 3, the bursty episodes indicative of hostage events contribute to a higher autocorrelation. Similarly, the relative uniformity of the "poaching" query leads to a smaller autocorrelation.

3.3 Statistics of the Rank Order of $P(t|Q)$

Another way to capture the dynamics of the time series is to consider the rank order of the time series. In these cases, we reorder the days in decreasing $P(t|Q)$. The features then are the statistical properties of the decay of $P(t|Q)$. Specifically, we look at the kurtosis of the rank order. The kurtosis is defined by,

$$\text{kurtosis} = \frac{\mu_4}{\mu_2^2}. \quad (11)$$

where μ_i is the ith central moment. The kurtosis measures the "peakedness" of the curve.

As with temporalKL, the peaky nature of Figure 3 is represented in this feature. However, in this case, we inspect the *rank ordered distribution*.

3.4 Burst Model

An alternative measure for temporality follows from Kleinberg's burst model [5]. In this model we assume a state machine which emits some number of relevant documents in each state. In one state, the low or idle state, few documents are emitted. In the other state, the event state, relatively more documents are emitted. If our time series is assumed to have been generated by such a machine, the idle state corresponds to a uniform distribution of N documents over the time span being considered. The event state corresponds to a faster rate of document production.

We should note here that we are dealing with the original document sample (the top N retrieved documents) used to estimate the temporal profile, not any estimated model, $P(t|Q)$, described in Section 2. That is, we look at the actual number of documents occuring on a day as opposed to a probability. A description of the full algorithm is beyond the scope of this paper. We implement Kleinberg's $\mathcal{B}_s^2$ automaton.

Given this automaton, we can use dynamic programming to find the most likely state sequence which replicates the data. For our two-state model, we are interested in the transitioning behavior of the machine. We chose three features of this transition sequence. First, we can compute what Kleinberg refers to as the weight of a burst. The weight of a burst is essentially the savings of taking the burst path over the idle path in the decoding. Second, we can compute the average length of time the machine is in the idle state before transitioning into the event state. Finally, we can compute the number of transitions to the event emission state.

We expect the burst weight to show the "intensity" of the time profile when relevant documents are found. This may reflect queries corresponding to high-intensity situations which are distinct from the background model. The average length of time the machine is in the idle state gives a measure of the overall significance of the topic over the time span in the collection. The number of transitions may capture the number of episodes present in the collection.

The burst model for the profile for "poaching" (Figure 2) spends most of the time in the idle state and has few transitions into the event state. Meanwhile, the model for "hostage taking" (Figure 3) transitions 5 times and spends, on average, half as much time in the idle state. Combined with the intensity measure, these features point to a more temporally structured query.

4. EXPERIMENTAL SETUP

Our goal is to evaluate the contribution of temporal features to predicting average precision. In this section we define a training and test collection, as well as performance measures. We use three performance measures. We selected Spearman rank correlation and linear regression because their explanatory power in showing correlation between features and average precision. We selected neural networks as they allow us to model complex nonlinear relationships for best predictive power.

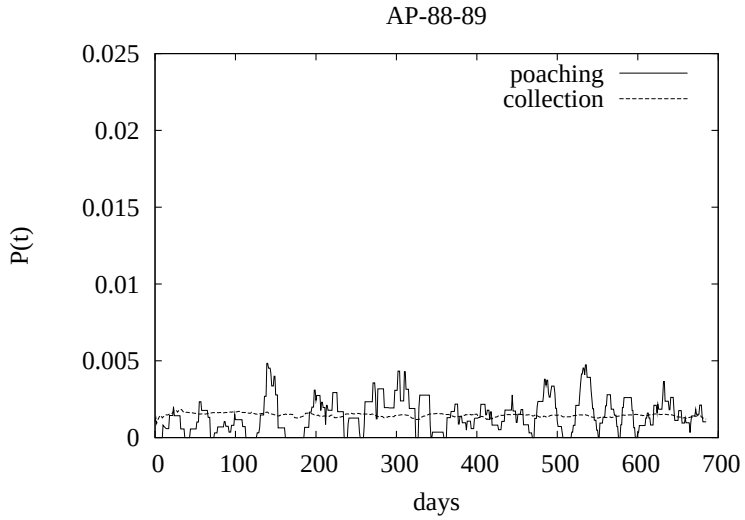
4.1 Dataset

All retrieval experiments use the TREC collection of news documents and queries. TREC collections often consist of several different sources covering different periods of time. In order to ensure that documents evenly cover the time period, we deal with subsets of the TREC collections. Specifically, we used the AP documents in the range 1988-1989 and Wall Street Journal documents from 1987-1992. Unless otherwise noted, the Lemur language modeling toolkit was used for text retrieval [1]. Query likelihood ranking was performed with a document model smoothing of 0.6. All documents and queries were stopped using the SMART stopword list and stemmed using the Krovetz stemmer [6].

Both the AP and WSJ collections have roughly 100 TREC queries associated with them. Because the queries were constructed with respect to the larger collections, we often found queries with few relevant documents in our AP and WSJ collections. Therefore, we only used queries with more than 15 relevant documents in our collections.

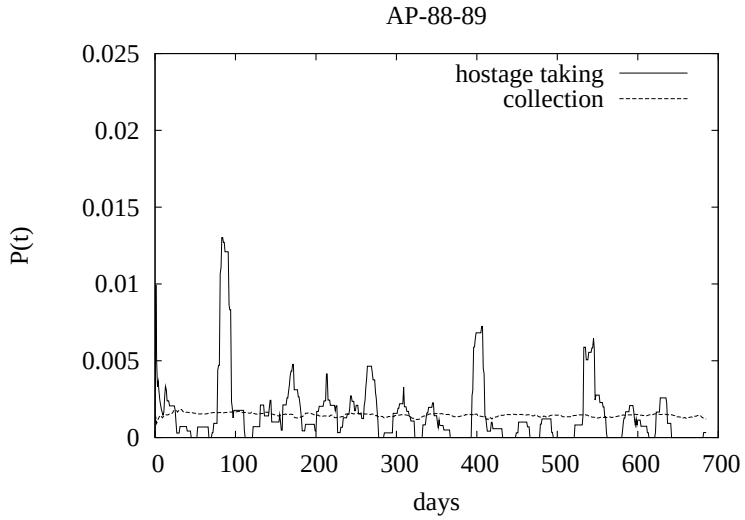
4.2 Spearman Rank Correlation

The Spearman rank correlation coefficient allows us to examine the relationship between predictor and predicted variables, without assuming any particular structure to that relationship. For example, with the Spearman rank correla-



Feature	Value	Value'
temporalKL	0.247	0.000
autocorrelation	0.921	0.611
kurtosis	2.476	0.000
burst weight savings	5.313	0.001
average amount of time in idle state	341.0	0.993
number of transitions into event state	2	0.000
content clarity	1.796	0.451
average precision	0.565	0.565

Figure 2: Temporal profile and raw and normalized feature values for the query "Poaching" over the AP88-89 collection. We normalized feature values by shifting and scaling them to lie between zero and one. This query had the minimum score for both temporal KL and number of transiitons into the event state, and so the scores for both of these features are zero after normalization.



Feature	Value	Value'
temporalKL	0.570	0.126
autocorrelation	0.938	0.719
kurtosis	3.921	0.124
burst weight savings	8.465	0.025
average amount of time in idle state	132.6	0.225
number of transitions into event state	5	0.429
content clarity	1.422	0.265
average precision	0.139	0.139

Figure 3: Temporal profile and raw and normalized feature values for the query "Hostage Taking" over the AP88-89 collection. We normalized feature values byshifting and scaling them to lie between zero and one.

tion coefficient, we can measure whether increases in content clarity lead to increases in average precision, without assuming that these are, for example, *linear* increases. Positive correlation using the Spearman rank correlation test tells us that there is a relationship between the variables. However, it does not tell us how to predict one variable from the other. The Spearman rank correlation is also not defined over multiple variables simultaneously. Thus we cannot use it to find whether we can improve our understanding of the average precision of a query by combining predictor variables.

4.3 Linear Regression

Linear regression allows us to combine multiple predictor variables, to predict *linear* changes in the average precision. That is, we are finding a linear relationship between our predictor variables and average precision. The coefficients of the variables show us their relative importance in predicting average precision, though two variables which are correlated may wind up with lower coefficients. For this reason we will also show the coefficients for each variable when used in isolation for predicting average precision. Since linear regression looks for linear relationships between variables, it is a stronger test with strong assumptions about the relationship between variables. We are less likely to find statistically significant correlations with linear regression than with Spearman rank correlation, particularly when the underlying relationship is nonlinear.

4.4 Neural Networks

Neural networks allow us to model nonlinear relationships between combinations of predictor variables. The hidden layers allow the representation of sub-combinations of features, which may aid with prediction. A neural network outputs a prediction of average precision for any input, and we can then compare the prediction with the actual average precision for a query. We measure the difference between actual and predicted average precision.

5. PREDICTING AVERAGE PRECISION

Cronen-Townsend et al. [3] showed that content clarity correlates with average precision, when using the Spearman rank-correlation test. This allows an information retrieval system to rank a set of queries by the likely quality of results. This may allow further processing or feedback from the user to improve results. However, it does not permit the system to predict the likely precision of any individual query, beyond a binary classification. In this section we analyze the Spearman rank correlation of temporal features with average precision. We also build models to predict the average precision of queries using a combination of temporal and content features, that is, using the temporal features we described in Section 3 along with content clarity as a feature describing the content. While any relationship between these features and average precision may be nonlinear, we first perform linear regression. This will allow us to compare the importance of features by examining their coefficients. We then use our measures of temporal clarity along with content clarity as input features to a neural network for predicting average precision.

For all experiments described in this section, we normalize the input features to lie between zero and one, by shifting and scaling the values.

	AP		WSJ	
Feature	R^2	**Prob R^2**	R^2	**Prob R^2**
autocorrelation	**0.36**	**0.01**	-0.05	0.72
burstAverageIdleTime	0.10	0.50	-0.02	0.87
burstWeightSavings	**0.28**	**0.05**	0.05	0.73
contentClarity	**0.54**	**5.0e-05**	**0.50**	**2.0e-04**
kurtosis	0.14	0.32	0.22	0.12
temporalKL	0.01	0.94	0.14	0.31

Table 1: Spearman rank correlation coefficient for correlation with average precision, for content clarity, and our proposed temporal features, for AP and WSJ. Statistically significant rank-correlations are shown in bold along with their significance levels.

Independent Variables	**Correlation Coefficient**			
	AP		WSJ	
	train	test	train	test
content clarity	0.33	0.21	0.41	0.36
temporal features	0.40	0.15	0.38	0.17
content + temporal features	**0.71**	**0.52**	**0.75**	**0.60**

Table 2: Correlation from Linear Regression: Average precision is the dependent variable. Independent variables are content clarity, and our measures of temporal clarity. Test correlation was found by cross-validation, by fitting the line using training data, then measuring correlation with held-out test data. For the row labeled "content + temporal features" we use all our temporal features, as well as content clarity as inputs to the linear regression.

5.1 Spearman Rank Correlation

Table 1 shows the Spearman rank correlation with average precision for each feature in isolation. We see that the correlation of content clarity with average precision is much higher than all other features, and that this correlation is statistically significant. This reproduces the results obtained by Cronen-Townsend et al. [3]. For our temporal features, the correlation is much lower, and for most features the measure of correlation is not statistically significant. This means that most of the temporal features are not predictive of average query precision, when used in isolation. However, note that a combination of these features may be predictive of average query precision. For the AP dataset, autocorrelation was positively correlated with the rank of average query precision at the 0.01 level, and burst weight savings was correlated at the 0.05 level. That means these two features may contribute to an improved predictive model of average precision, when combined with each other and content clarity, if they are not redundantly correlated with one another.

5.2 Linear Regression

Table 2 shows the correlation using linear regression lines between average precision and measures of query clarity. Test correlation was found by cross-validation, by fitting the line using training data, then measuring correlation with held-out test data. Note that for both AP and WSJ, the combination of content and temporal measures shows a much stronger correlation with average precision than content clarity or temporal features alone. This means that our mea-

Feature	Coeff. AP (s^2)	Coeff. WSJ (s^2)
autocorrelation	0.43 (0.22)	-0.24 (0.19)
burstAverageIdleTime	- (0.19)	0.70 (6.58)
burstNumTransitions	0.19 (0.19)	1.0 (8.23)
contentClarity	0.97 (0.34)	0.96 (0.13)
kurtosis	0.28 (0.31)	- (0.21)
temporalKL	-1.3 (0.45)	-0.83 (0.26)

Table 3: Coefficients of individual features in linear ridge regression for predicting average precision for AP and WSJ data. In brackets is shown the sample standard deviation over 10,000 iterations of bootstrapping pairs linear regression, without normalization.

sures of temporal clarity contribute to the understanding of the likely effectiveness of a query with respect to a corpus. Note also that the correlation scores remain high when tested using cross-validation.

The variables with strong predictive power are shown in Table 3 with their coefficients for predicting average precision. The coefficients shown are with ridge regression, which performs normalization and removes potentially irrelevant feature. This led to the best predictive results. To estimate the standard deviation of the coefficients, we performed pairs bootstrapping for 10,000 iterations, without normalization. Thus these standard deviations give an upper bound on the uncertainty of the estimates of coefficient strength. Note that while some of the features are unstable, we obtain high correlation scores, even on a held-out set of data not used for fitting the regression lines.

Interestingly, the magnitude of the coefficient for content clarity is not the largest. We find that temporalKL has a negative coefficient. This shows that queries with temporal profiles very different from the background model are likely to have low average precision. This suggests that queries which retrieve documents from an unusual subset of days in the collection are likely to perform poorly. These may be good candidates for a relevance feedback interface which highlights the days on which retrieved documents appeared, and allows the user to select the appropriate timeframe. We discuss a possible interface of this form in related work [4].

How do we explain temporal KL, which does not predict average precision when used in isolation, but which has a negative coefficient in conjunction with the other features? We can infer that it explains some parts of average precision which are not explained by the other features, but only when we know the values of the other features.

5.3 Neural Networks

Neural networks can be used to learn non-linear functions. We used the Weka implementation of neural networks [10]. Our target function is average precision. In all cases we used one hidden layer. In the case of content clarity only as an input feature, we used a single node in the hidden layer. In all other cases we used three nodes in the single hidden layer. All input features are connected to all nodes in the hidden layer. We used a learning rate of 0.3 and momentum of 0.2, 10% validation set and 500 training epochs, with early termination of training if the accuracy on the validation set grew worse over 20 epochs. We used 80% of the data for training, 10% for validation, and 10% for testing.

As a baseline, we guessed the mean average precision over all queries. The mean average precision was 0.28 over all queries on WSJ, and 0.30 over all queries on AP. When we use this value for average precision for every query in the test set, we can calculate how much each query deviates from this level of average precision, and calculate root-mean squared error (error in terms of difference between predicted and actual query average precision), and root relative squared error (error in terms of a percentage of the actual average precision of a query). With these baseline average precision values, predicted average precision was on average 0.20 from the true value (root mean squared error), with 100% root relative squared error, as shown in the first row of Table 4. By using the other features as input to a neural network, we hope to gain predictions of average precision for each query which are more reliable than guessing these baselines.

Input Features	AP		WSJ	
	RMSE	RRSE	RMSE	RRSE
baseline	0.20	100%	0.22	100%
NN: content clarity	0.53	101%	0.18	87%
NN: temporal features	0.31	98%	0.27	131%
NN: content + temporal features	**0.27**	**88%**	**0.13**	**63%**
LR: content + temporal features	**0.23**	**92%**	**0.18**	**82%**

Table 4: Error in predicting average precision using a neural net with 1 hidden layer. RMSE is root mean squared error of the predicted value on a held-out test set. RRSE is root relative squared error on the held-out test set. In the final row we show these measures on the model built with linear regression. We see that neural networks performed better at prediction than linear regression, so we gain predictive power from the non-linear transformations.

We see in Table 4 that content clarity in isolation reduced the root relative squared error for WSJ to 87%, and that temporal features in isolation do not reduce the root relative squared error. However, the final row of Table 4 shows that using the combination of temporal and content clarity with a neural net, we can reduce both the root mean squared error and the root relative squared error from the default. This means that we are able to predict average precision with greater accuracy than the default, and greater accuracy than using content clarity alone. In the final row we show these measures on the model built with linear regression. We see that neural networks performed better at prediction than linear regression, so we gain predictive power from the non-linear transformations.

6. CONCLUSIONS AND FUTURE WORK

We have presented the concept of temporal profiles of queries. In addition to defining this representation of a query, we also presented results on the use of these profiles in information retrieval settings. We showed that we are able to improve prediction of the average precision of a query by adding information from the temporal profile.

This is an important step towards self-diagnosing information retrieval systems, that perform extra computation or relevance feedback, or elicit other information from the user or a user profile, depending on expected quality of the results. We showed that the temporal profile of a query

can provide valuable information about the likely quality of query results. While some of our features proved unstable depending on the sample, we were able to improve prediction across two corpora. This opens up a broad range of research questions, about identifying features for prediction of average precision, both in the temporal domain and using other features of queries and document collections. Another important question is whether models of query precision generalize across query sets and document collections.

7. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor. We would like to thank Ed Fox, Dave Pennock and the anonymous reviewers for helpful comments and feedback.

8. REFERENCES

[1] J. Allan, J. Callan, K. Collins-Thompson, B. Croft, F. Feng, D. Fisher, J. Lafferty, L. Larkey, T. N. Truong, P. Ogilvie, L. Si, T. Strohman, H. Turtle, and C. Zhai. The lemur toolkit for language modeling and information retrieval. http://www-2.cs.cmu.edu/~lemur/, 2003.

[2] W. B. Croft and J. Lafferty. *Language Modeling for Information Retrieval*. Kluwer Academic Publishers, 2003.

[3] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 299–306, August 2002.

[4] F. Diaz and R. Jones. Temporal profiles of queries. Technical Report YRL-2004-022, Yahoo! Research Labs, 2004.

[5] J. Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 91–101, July 2002.

[6] R. Krovetz. Viewing morphology as an inference process. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1993)*, pages 191–203, 1993.

[7] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 120–127. ACM Press, 2001.

[8] X. Li and W. B. Croft. Time-based language models. In *Proceedings of the 2003 ACM CIKM International Conference on Information and Knowledge Management (CIKM 2003)*, pages 469–475. ACM, November 2003.

[9] R. Swan and D. Jensen. TimeMines: Constructing timelines with statistical models of word usage. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2000)*, pages 73–80, August 2000.

[10] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999. http://www.cs.waikato.ac.nz/ml/weka/.