Intent-aware temporal query modeling for keyword suggestion

Conference Paper · November 2012

DOI: 10.1145/2389686.2389703

4 authors, including:

Fredrik Johansson
Chalmers University of Technology
22 PUBLICATIONS **519** CITATIONS
SEE PROFILE

Vinay Jethava
Chalmers University of Technology
16 PUBLICATIONS **210** CITATIONS
SEE PROFILE

Svetoslav Marinov
Seal Software
14 PUBLICATIONS **898** CITATIONS
SEE PROFILE

Intent-aware temporal query modeling for keyword suggestion

Fredrik Johansson, Tobias Färdig
Chalmers University
Gothenburg, Sweden
{frejohk,tobiasf}@student.chalmers.se

Vinay Jethava
Chalmers University
Gothenburg, Sweden
jethava@chalmers.se

Svetoslav Marinov
Findwise AB
Gothenburg, Sweden
svetoslav.marinov@findwise.com

ABSTRACT

This paper presents a data-driven approach for capturing the temporal variations in user search behaviour by modeling the dynamic query relationships using query-log data. The dependence between different queries (in terms of the query words and latent user intent) is represented using hypergraphs which allows us to explore complex relationships compared to graph-based approaches. This time-varying dependence is modeled using the framework of probabilistic graphical models. The inferred interactions are used for query keyword suggestion - a key task in web information retrieval. Preliminary experiments using query logs collected from internal search engine of a large health care organization yield promising results. In particular, our model is able to capture temporal variations between queries relationships that reflect known trends in disease occurrence. Further, hypergraph-based modeling captures the query relationships significantly better compared to graph-based approaches.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation, Theory

Keywords

user intent, keyword suggestion, graphical model, dynamic query interaction, query hypergraph

1. INTRODUCTION

Understanding the intent behind search queries is crucial towards improving user search experience [41, 40]. The traditional classification of user intent [12] as *navigational*, *transactional* or *informational* has been extended [35, 5, 22] to provide a multi-faceted description of user intent.

It is well recognized that both user intent and the search queries used to express this intent change over time [41]. Consequently, a number of methods have been devised to understand this temporal variation [17, 25, 3, 29, 37, 4, 34]. What has hitherto not been considered is *how search queries relate to each other, and how these relationships change with time*.

A key challenge when modeling query relationships is that multiple queries might be used to express the same underlying intent. For example, consider the queries *job openings*, *job bank* and *employment*. These queries reflect the same underlying user intent; and consequently show similar trends i.e. in periods of economic slowdown, more users are likely to search for these than otherwise. It is necessary to treat these queries in terms of a single coherent interaction in order correctly model the underlying intent. Graph-based methods [2, 38, 23] focussing on pair-wise relationships fail to address multi-way interaction. In this paper, we model such multi-way interactions using hypergraphs [8] wherein each node represents a distinct search query, and each hyperedge corresponds to one interaction among several queries related in terms of their user intent.

Capturing user intent behind search queries is crucial to correct inference of query relationships. However, this poses additional challenge since manual intent classification of search queries is needed for each of the multiple facets of interest [35, 5, 22]. Recent work has partly addressed this by modeling the inter-dependence of multiple facets describing user intent [24]. In this paper, we use an implicit representation of user intent in terms of the content or topic distribution of the click-through document obtained using Latent Dirichlet Allocation (LDA) [10] - a probabilistic topic model [9] widely used in text retrieval. This representation allows finding queries with similar user needs without explicit classification, thus making it possible to scale up the algorithm.

This paper presents a systematic approach for understanding the time-varying search query relationships extracted from query log data. A query relationship or *query interaction* expresses commonality in user intent among multiple search queries at a given time, which is represented by a hyperedge consisting of the search queries. The time-varying query interactions therefore reflect the changing user needs over some time period, and are represented by the time-varying interaction strength of the hyperedge. This dependence is modeled using the framework of graphical mod-

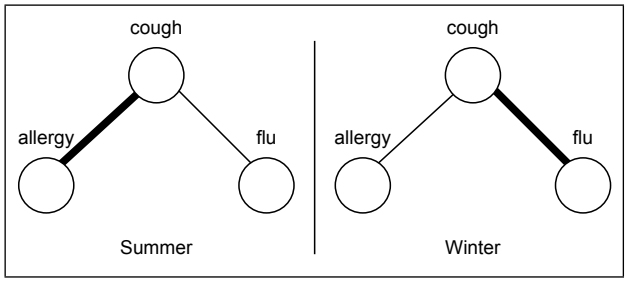

Figure 1: Example of query interaction dynamics. Nodes represent queries and edges interactions between them. The thickness of an edge indicates the strength of the interaction.

els [28, 42]. Specifically, a generative model is presented wherein the time-varying interaction strength of an interaction between two queries is modulated by the (latent) user intents of the search queries. The model allows tractable inference of query interactions from query log data using modified Expectation-Maximization (EM) algorithm [16, 32].

Exploration of search query relationships and their temporal variation promises two-fold gain in terms of better understanding of changing user needs, as well as the possibility to spot new trends in user search. For example, consider the queries *cough*, *flu* and *allergy* shown in Figure 1 (representative example extracted from algorithm results described further in Experimental section). The edges between them represent query interactions, while the weight of the edges represents strength of the interaction. During summer, the model infers that *cough* is more related to *allergy* than to *flu*. During winter, the opposite is inferred i.e. *cough* and *flu* are more related. This information can be used e.g. to rank results or suggesting alternate keywords for search query depending on season.

In this paper, we use the inferred query interactions for query keyword suggestion - a key task in web information retrieval. More specifically, alternate keywords are suggested based on queries which strongly interact with test query at current time. Preliminary experiments on synthetic data as well query extracted by the consulting company, Findwise AB[1], from the internal search engine of large health care organization show that a) our hypergraph-based approach does significantly better than graph-based approaches for query keyword suggestion, and b) temporal query modeling improves keyword suggestion compared to previous approaches.

2. RELATED WORK

We discuss research in two fields that is relevant to our work on modeling time-varying query relationships and their use in improving user's search experience using probabilistic graphical models, namely, a) Modeling time-varying intent and its application in improving web search focussing on query suggestion, and b) probabilistic graphical models used to capture (mostly pair-wise) interactions between several entities.

[1] `http://www.findwise.com`

Various methods for modelling query relationships have been explored, for example using measures of query similarity and clickthrough URLs [6, 7] and topic models [21]. Baeza-Yates et al. [6] approaches the problem of query recommendation by relating queries through a semantic similarity measure based on word occurrence in click-through URLs and query popularity. The method also ranks the recommended queries according to a relevance criterion. Guo et al. [21] performs intent-aware query recommendation using a query similarity measure based on a regularized topic model. The similarity measure gives a sense of which queries are related, and enables recommending new queries for search users.

Modeling user intent has been studied in context of query suggestion [41, 14] and related problems including query recommendation [34, 21] and query expansion [36]. This problem is related to the problem of keyword extraction [26] wherein new documents are assigned a set of keywords when they are entered into a search engine. Another attempt [7] relates queries by analyzing a bipartite query-document graph created using query logs. The graph is then used to infer semantic relationships between queries. The quality and type of such relationships are then analyzed enabling detection of *equivalent* and *similar* queries.

Recent work has explored query dynamics towards improving the search experience. Alfonseca et. al. [3] use temporal variations in query volume to improve query suggestions. Kulkarni et. al. [29] study the temporal dynamics in queries, their underlying intent and the associated documents over a 10 week period for a small set of queries identifying key patterns of variation in query popularity, document content and query intent. Radinsky et al. [34] model the temporal dynamics of user search behaviour, based on querylogs, and consider detection of periodicity and surprises etc. The model is used in predicting user activity and can be useful in query suggestion and result ranking. However, these approaches does not consider how queries relate to each other or how such relationships evolve.

Probabilistic graphical models [28, 42] have been widely used in number of fields including study of social networks [2], biological networks [23, 38], text streams [1]. However, to the best of the author's knowledge, this is the first attempt to model query relationships using a Bayesian approach. This allows one to incorporate domain knowledge in terms of appropriate priors; and possibly extend based on additional information e.g. knowledge of session information [14], user profiles [39], etc.

A related model for inferring pair-wise interactions in a graph using node observations is Mixed-Membership Stochastic Blockmodels (MMSB) [2]; which has been extended to track temporal dynamics of such interactions [19]. However, extension to multi-way interactions is highly non-trivial and computationally infeasible since it involves inference of tensors rather than matrices.

3. METHODOLOGY

This paper presents a method for exploring the time-varying relationships among search queries depending on the underlying user intent. This is used suggesting temporally relevant keywords for new documents in a search-engine, de-

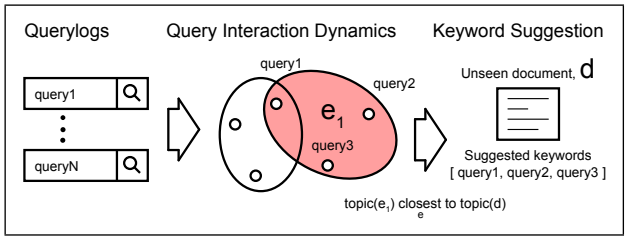

Figure 2: Simplified view of the keyword suggestion process. Interaction strengths inferred by the model using query logs are used for suggesting relevant keywords for a new document.

pending on the query relationships at that time. A high-level view of the keyword suggestion process is shown in Figure 2. The main components in the above task are: a) inference of query interactions from query logs, and b) keyword suggestion using inferred query interactions; which are discussed in Section 3.1 and Section 4.2 respectively.

3.1 Inferring query interaction dynamics

Graphs [11] have been widely used for modeling interactions between multiple entities in several domains [28]. However, graphs capture pair-wise interactions (or correlations) only, which is not suited for query interactions since multiple queries might express the same intent. For example, misspellings and different wordings cause similar phrases to be different queries, different nodes in a graph, but they should be strongly related.

Often, multi-way interactions are reduced to pair-wise interactions by adding an edge between every two entities which are part of the original multi-way interaction. This results in a clique corresponding to each multi-way interaction in the reduced graph representation. Consequently, the overall number of edges in the resultant graph increases. Further, graphs representing real interactions often have weights associated with each of the edges which are indicative of the strength or degree of interaction between two nodes relative to other interactions in the graph. In the case of a multi-way interaction, the resulting graph representation has weights for each of the edges which are part of the clique representing the multi-way interaction, which might differ significantly.

In this paper, we adopt the increasingly recognized view [27, 20] that in order to successfully capture multi-way interactions, one must treat them accordingly. This represents one of the major departures of this work from preceding studies. In order to capture this effectively, we use hypergraphs [8], a generalization of graphs which allow us to express interactions between several queries.

The remainder of this section is organized as follows: Section 3.1.1 presents basic hypergraph notation which is used in the remainder of this paper, Section 3.1.2 describes the query log data used, Section 3.1.3 describes the construction of the base hypergraph which identifies queries expressing similar intent, Section 3.1.4 describes the Markovian assumption which is used to infer changes in query relationships based on query log data.

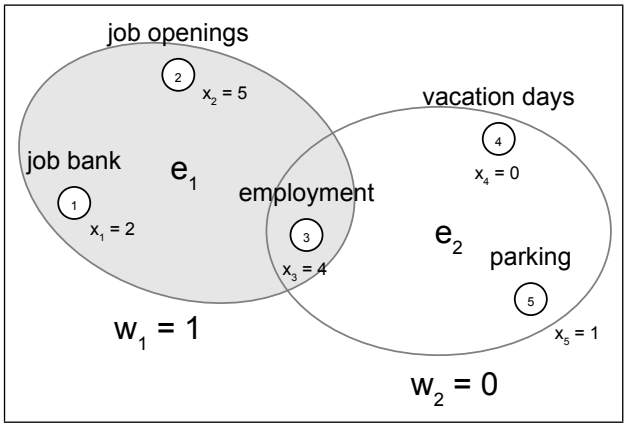

Figure 3: Hypergraph representation of query interaction. Each node in the hypergraph represent a unique query. A hyperedge (circle) represents the interaction of the queries inside it. Each hyperedge is associated with a weight representing the strength of the interaction.

3.1.1 Hypergraph representation

We denote simple (pairwise) graphs $G = (\mathcal{V}, \mathcal{E}^P)$ and hypergraphs $H = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ denotes the set of nodes, $\mathcal{V} = \{v_1, \ldots, v_N\}$ where N is the total number of nodes. The edge sets are denoted $\mathcal{E}^P = \{e_1^P, \ldots, e_{MP}^P\}$ and $\mathcal{E} = \{e_1, \ldots, e_M\}$ respectively. An pairwise edge is a pair of nodes $e^P = (v_i, v_j)$ and a hyperedge is a tuple (or a subset of $\mathcal{V}$), $e = (v_i, v_j, v_k, \ldots)$ of arbitrary size $k_e \geq 2$. The size k_e of an hyperedge is also called *cardinality*. We use the notation $v \in e$ to indicate a node n part of the hyperedge e. We will sometime use the notation $e = (i, j, ..)$ to mean $e = (v_i, v_j, \ldots)$ for convenience. In general, we consider *edge* to mean hyperedge, unless otherwise specified. This means that $k_e \geq 2, \forall e \in \mathcal{E}$. Throughout the paper, we will use the terms *node* and *query* interchangeably as well as *edge* instead of *interaction* or *relationship*

We model the query interaction network as a hypergraph, where each node represents one unique query, and each hyperedge represents a subset of queries expressing similar user intent. Figure 3 shows a hypergraph representation of query interactions. Every edge e is associated with a weight w_e^t, representing the strength of the interaction that depends on the time point t. Weights take on values in a small set W. If $W = \{0, 1\}$ we interpret the case of $w_e^t = 0$ as the queries involved in the edge e having no interaction at time t.

3.1.2 Data description

The input to our model consists of two components, query usage data x and a base hypergraph H, such as that of Figure 3. Query usage measurements are expected in the form of time sequences $x = ((x_j^t)_{t=1}^T)_{j=1}^N$ with j the index of the query and t the time point of measurement. We let $t \in \{1, \ldots, T\}$ be an index representing the interval $[\tau(t), \tau(t+1)]$ where $\tau(t)$ is the starting time of the time period with index t. $\tau(0)$ is the earliest measurement point and $\tau(T)$ the latest. Usage measurements x_j^t is taken to be the number of times a query j has been made at time t.

Our model also requires an initial specification of which interactions to consider. The hypergraph $H = \{\mathcal{V}, \mathcal{E}\}$ is a specification of which edges to infer interactions for. Each node in the graph represents a unique query. Edges represent query interactions and are denoted $e = (v_{e,1}, \ldots, v_{e,k_e})$ where k_e is the cardinality of edge e. Note that the nodes $\{v_{e,1}, \ldots, v_{e,k_e}\}$ are a subset of the node $\mathcal{V}$. This graph can be thought of as a static representation of relationships between the queries involved, i.e. which interactions that are expected to take place at all during the entire time period of the data. This graph is constructed based on the underlying intent of the queries

In the query application, data comes in the form of a query log $\mathcal{Q} = \{Q_1, \ldots, Q_M\}$ made up of tuples $Q_i = (q_i, \tau_i, d_i)$. We call such a tuple a *search query instance* or simply *instance*. Each such instance represents the event of a user searching for something and then clicking on a result. q_i are the query words for instance i, τ_i is the time stamp and d_i is a document identifier associated with the document the user clicked on after making the query. To transform the query logs into query usage data, we simply count the occurrences of each query in each interval t

$$x_j^t = \sum_{i=1}^{M} \delta_j(q_i, \tau_i) \quad (1)$$

$$\delta_j^t(q_i, \tau_i) = \begin{cases} 1 & q_i = \tilde{q}_j \text{ and } \tau_i \in [\tau(t), \tau(t+1)] \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

with $\tilde{Q} = \{\tilde{q}_j\}_{j=1}^{\tilde{M}}$ the set of unique queries.

Each set of query words q_i or document identifier d_i may occur arbitrarily many times in the log. This simply means that the same query has been made, or the same document has been clicked on at different times. These re-occurrences is what we count to form query usage data. For instance, if the query *flu* has been made 13 times in February and our time intervals are the months of the year, we get $x_{flu}^2 = 13$.

3.1.3 Intent-based query interaction hypergraph

We construct the base hypergraph H by identifying subsets of queries which express the same underlying intent. However, explicit description of user intent [5] requires manual classification or sufficient training data - which makes it impractical for large-scale scenarios. Instead, we use an implicit representation of intent in terms of the click-through documents using LDA [10].

More precisely, we associate with each click-through document d, a topic distribution $\vec{A}_d$ extracted using LDA [10] with appropriate priors. From the query logs, each query q_i is associated with a set of documents, $\mathcal{D}_i = \{d_{i,1}, \ldots, d_{i,m}\}$, one document for each instance of the query. From a topic model, each document is associated with a topic distribution $\vec{A}_d = [A_{d,1}, \ldots, A_{d,H}]^\top$, such that $\sum_{h=1}^{H} A_{d,h} = 1$ for all d. An element $A_{d,i}$ can be thought of as the influence of topic i in document d.

We calculate query-topic distributions $\vec{\kappa}_i = [\kappa_{i,1}, \ldots, \kappa_{i,H}]^\top$ by averaging over document-topic distributions for each query.

$$\kappa_{i,h} = \frac{1}{|\mathcal{D}_i|} \sum_{d \in \mathcal{D}_i} A_{d,h}. \quad (3)$$

We can construct a static interaction graph by comparing query topic distributions, κ. First we construct a pairwise graph, which is then converted to a hypergraph.

We denote the pairwise graph of query interactions $\mathcal{G} = (\mathcal{V}, \mathcal{E}^P)$. We add a node v_i to $\mathcal{V}$ for each unique query q_i. Then we calculate the Kullback-Leibler [30] divergence

$$D_{KL}(P||Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)} \quad (4)$$

for each pair of queries (v_i, v_j) to get a measure of how related two queries are by topic. We add an edge $e^P = (v_i, v_j)$ to $\mathcal{E}^P$ if $D_{KL}(\vec{\kappa}_i||\vec{\kappa}_j) > \varepsilon_{KL}$ where ε_{KL} is a parameter of the tool. The pairwise graph is then converted to a hypergraph $H = \{\mathcal{V}, \mathcal{E}\}$ using Bron-Kerbosch algorithm [13].

3.1.4 Modelling query interaction dynamics

The base hypergraph H specifies sets of queries (as hyperedges) which have the same user intent, expressed in terms of topic distribution $\vec{\kappa}_i$ for each query q_i. However, user needs vary over time. For example, during periods of economic downturn, more users might be looking for a job; and this in turn will be reflected in search queries expressing this need. In terms of topic distribution $\vec{\kappa}_i$ obtained by LDA, some of the topics will be more active at each time. These active topics govern which query interactions are active at that time. We note that the active topics at each time are unknown - however, their influence is observed in terms of the (latent) active interactions and consequently in the observed query statistics at that time.

Dependence of query statistics on interactions. The dependence of observed query statistics $\vec{x}^t = \{x_v^t\}_{v \in \mathcal{V}}$ on query interactions $\vec{w}^t = \{w_e^t\}_{e \in \mathcal{E}}$ present at that time is modeled using the Boltzmann distribution,

$$P(\vec{X}^t = \vec{x}^t \mid \vec{W}^t = \vec{w}^t) = \frac{1}{Z(w^t)} \exp\left(\sum_{e \in \mathcal{E}} w_e^t \phi(e, t)\right) \quad (5)$$

with $Z(w^t)$ the normalization factor. Here, $\phi(e)$ is a potential function. In this model we use,

$$\phi(e, t) = \prod_{v \in e} x_{i(v)}^t \quad (6)$$

with $i(v)$ the index of node v.

One can interprete each weight w_e^t as indicative of the dependence (higher-order correlation) between the query statistics x_v^t for all queries $v \in e$ present in the hyperedge. A positive value of w_e^t indicates the search queries in edge e are more likely to be present at that time; while a negative value of w_e^t indicates those search queries are less likely to be present.

Dynamics of interactions. We assume that the set of active topics changes smoothly over time. Under this assumption, we motivate modeling interactions as a Markov model. More precisely, the state $\vec{w}^t$ of the interaction network at time t is made up of the interaction weights of edges $\{w_e^t\}_{e \in E}$. Transitions are governed by transition probabilities Q_e where an element $Q_e(i, j)$ is the probability of mov-

ing from weight state W_i to weight state W_j in edge e, with $W_i, W_j \in W$.

We make the assumption that edge dynamics are conditionally independent conditioned on the active topics at that time i.e. the interactions in one group of queries is independent of the interactions in other groups if we know which topic is influencing each set of queries. This conditional independence assumption enables us to treat every edge independently of other edges while doing inference - thus allowing the algorithm to be scalable to large hypergraphs.

Given the Markovian assumption, the transition probability of the interaction strengths w_e^t of an edge e can be written as

$$P(W_e^t = w_e^t \mid W^1, \ldots, W^{t-1}) = P(W_e^t = w_e^t \mid W^{t-1}). \quad (7)$$

We denote the probability of an edge $e \in \mathcal{E}$ moving from state w_l to w_k, both in $\mathcal{W}$,

$$Q_e(k,l) = P(W_e^t = w_l \mid W_e^{t-1} = w_k) \quad (8)$$

with

$$\sum_{l=1}^{K} Q_e(k,l) = 1 \ \text{ for all } k = 1, \ldots, K. \quad (9)$$

where Q_e denotes the transition probability for the edge e. As noted previously, the transition probabilities Q_e for different edges are not independent. Rather, the transition probability depends on the active topic at each time. This is captured using a mixture model approach which we describe below.

Influence of user intent on interaction strength. We assume that there exists a set of topics $\mathcal{H} = \{h_1, \ldots, h_{N_H}\}$. The number of topics N_H is a parameter of the model. Also, we assume that for each $h \in \mathcal{H}$ there exist an unknown transition probability matrix Q_h which we call *topic transition probability*.

We model the dependence of edge transitions on user intent as follows: if $h' \in \mathcal{H}$ is the active topic for edge e at time t, then the transition probability for edge e at time t is given by

$$Q_e^t = Q_{h'}$$

Mathematically, this is equivalent to considering a mixture model over the set of topic transition probabilities for each edge e. In the mixture model, edge transition probabilitis Q_e depend on topic transition probabilities Q_h and mixture proportions $\alpha_{e,h}$. The parameter $\alpha_{e,h}$ govern the influence of topic h in the evolution of edge e and the relationship between Q_e and Q_h can be written as follows,

$$Q_e(k,l) = \sum_{h \in \mathcal{H}} \alpha_{e,h} Q_h(k,l) \quad (10)$$

with

$$\sum_{h \in \mathcal{H}} \alpha_{e,h} = 1 \ \text{ for all } e \in \mathcal{E}. \quad (11)$$

The (unknown) parameter α can be thought of as a matrix with element $\alpha_{e,h}$ at index (e,h). We will sometimes use the notation $\vec{\alpha}_e = \{\alpha_{e,h}\}_{h \in \mathcal{H}}$ which can be thought of as a row vector of α with dimension $|\mathcal{E}|$.

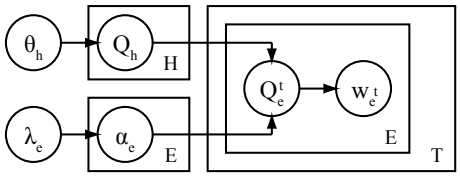

Figure 4: Plate notation representation of the generative model. Letters in bottom right corners of boxes indicate repetitions of the variables inside.

This completes the model specification with the unknown parameters being the mixing proportions $\alpha_{e,h}$ and topic transition probabilities Q_h. The generative model is shown in Figure 4. We now describe the choice of appropriate priors for the unknown parameters.

Parameter priors. In our model, as described in Section 3.1.4, each edge e is assumed to be associated with a topic mixture $\vec{\alpha}_e$ with elements $\alpha_{e,h}$ for topics $h \in \mathcal{H}$. The parameter $\alpha_{e,h}$ is a measure of how active the topic h is in the edge e. We impose a Dirichlet prior over $\vec{\alpha}_e$ with parameters $\vec{\lambda}_e$, i.e.

$$P(\vec{\alpha}_e) \sim \text{Dir}(\alpha_{e,1}, \ldots, \alpha_{e,H}; \lambda_{e,1}, \ldots, \lambda_{e,|\mathcal{H}|}) \ . \quad (12)$$

We interpret the topic mixture hyperparameter, $\Lambda = \{\vec{\lambda}_e\}_{e \in \mathcal{E}}$ as a set of topic distributions, one for each edge e. When constructing a prior in the query application, we exploit the fact that we have calculated a static topic distribution for queries, as described in a previous section. As a prior $\vec{\lambda}_e$ for an edge $e = (v_1, v_2, \ldots, v_{k_e})$, we simply use the mean topic distribution over the queries in e

$$\lambda_{e,h} = \frac{1}{k_e} \sum_{v \in e} \kappa_{i(v),h}, \quad e \in \mathcal{E}, \quad h \in \mathcal{H}. \quad (13)$$

where κ_i is the topic distribution for query i, k_e is the cardinality of edge e, and $i(v)$ is the query index of node v.

For transition probabilities we use a Laplacian prior,

$$\Theta(i,j) = \frac{e^{-\gamma|W_j|}}{\sum_{k=1}^{K} e^{-\gamma|W_k|}} \quad (14)$$

which controls the sparsity of the output interaction weight sequence by controlling the probability of reaching a certain weight state using the parameter γ. In other words, a high γ gives a high frequency of small weights (in terms of absolute value), such as 0, while a low γ gives more frequent larger weights.

Inference & parameter estimation. To estimate the parameters of our model we use a modified version of the expectation-maximization (EM) procedure [16, 33]. As part of the E-step, the procedure computes the forward and back-

ward iterates and then computes the MAP estimates of the model parameters $\alpha_{e,h}$, Q_h and therefore also Q_e. The conditional independence of each interaction conditioned over the topic distribution of its nodes allows parallel implementation of the E-step over each of the interactions in the graph. Thus, the algorithm can be parallelized using existing frameworks like MapReduce [15] or GraphLab [31].

4. EXPERIMENTS

This section presents preliminary results on synthetic and real-world datasets demonstrating the utility of our approach.

4.1 Evaluation on synthetic data

As a first experiment, we test the performance of our model, and the implementation of it, on synthetically generated data. The data is generated according to the distributions of the graphical model described in earlier sections. We compare our performance to graph-based approach [23].

We start off generating a random hypergraph, $H = (\mathcal{V}, \mathcal{E})$ using the procedure described by Ghosal et. al. [20]. A copy of the hypergraph is then converted to a pairwise (simple) graph $G = (\mathcal{V}, \mathcal{E}^P)$ by connecting all pair of nodes (n_j, n_k) of every hyperedge e with an edge $e^P = (j, k)$ in order to allow comparison with graph-based approaches [23]. Next, any duplicate edges are removed.

We generate two types of hypergraphs in this procedure, one with uniform edge cardinality $k = 3$ and one with random cardinality $k \leq 4$. In the general case, the cardinality of hyperedges follow the distribution $P(k = 2) = 0.70$, $P(k = 3) = 0.25$, $P(k = 4) = 0.05$, $P(k = i) = 0, i \notin \{2, 3, 4\}$. Then we generate random parameters, the transition matrices Q_h and the mixtures $\alpha_{e,h}$. These are now assumed to be known throughout the test and form the origin of the generative process.

We generate a weight sequence W_e^t and observations x_i^t for the hypergraph using the known parameters Q_h and $\alpha_{e,h}$. We construct one instance of our model using the hypergraph H and one using the pairwise graph G. We perform inference on both of the models using the observations x and the two graphs. This results in two weight sequences, $\tilde{W}_e^t$ and $\tilde{W}_e^{P,t}$, one inferred from each model, to compare to the original weight sequence, W^t.

Since the pairwise graph has more and different edges than the hypergraph used as the ground truth, we need to infer a weight sequence for the corresponding hypergraph using the pairwise weight sequence. We do this by using one of the simplest decision rules - majority vote. For each pairwise edge e_i making up hyperedge e_j, we perform majority vote. Furthermore, we compute Fleiss' Kappa[18] , κ_F as a measure of the agreement between edges in the vote.

As a measure of the error of the inferred weight sequence, we count the number of elements that differ from the truth in sequences W_e and $\tilde{W}_e$. For the experiment we use weights $W = \{-1, 0, 1\}$, $N = 20$ nodes, $T = 100$ time points, $H = 20$ topics, and $E = 50$ hyperedges. In table 1, we present the results of comparing performance for pair-wise graphs and hypergraphs on synthetically generated data. The average

Model	Error, $k = 3$	Error, $k \leq 4$
Graph-based approach [23]	66%	37%
Our model	26%	30%

Table 1: Error in inferred weights on synthetic data. k is the cardinality of hyperedges. The error is the percentage of weights in the sequences inferred by the models that differed from the true sequence.

Fleiss kappa was $\kappa_F = -0.01$ which indicates poor agreement in the majority vote [18].

From table 1, we can clearly see that for data that naturally lends itself to a hypergraph representation, our model outperforms the graph based method. For 3-hypergraphs the difference is larger in the general case because every edge has more than two nodes which means that the pairwise representation will always have to use majority vote.

4.2 Evaluation on keyword suggestion

In this section we use the results of our model trained on the query data training set to suggest keywords for new documents. We will now formalize what we take to be a suggestion, and what an ideal suggestion is. As stated in the introduction, we want to suggest keywords that are actually used as search queries. This would make the keywords relevant in terms of result ranking and query matching. The underlying idea for the suggestions we make is to make use of the query logs in the training set and suggest keywords that have been used as queries.

In this test, we use querylogs and click-through documents from the internal search-engine of a Swedish county council. The dataset consists of around 350 documents 1 000 000 query instances of 20 000 unique queries. We construct the training and testing sets by partitioning the data chronologically. Also, we make the training set contain 90% of the query instances. To do this, we sort the instances making up the total query log by time and take the first 90% of the to be the training sets and the remainder the test set.

We define a keyword suggestion for document d at time t as a set of queries $K_d^t = \{q_{K,1}, \ldots, q_{K,m}\}$. Please note that we only suggest keywords that have been used as queries in the training set, not any words occurring in documents. Here we differ from existing keyword extraction software [26].

Formally we define,

$$K_d^t = \{q : q \in e, e = \arg\min_{e \in \mathcal{E}} [D_{KL}(\alpha_e || A_d)] \text{ s.t. } |w_e^{t-1}| > 0\}, \tag{15}$$

where A_d is the topic distribution for document d and α_e is the topic distribution for edge e. D_{KL} is the KL divergence as defined in (4).

In words, the above definition means that the queries we suggest for keywords are those making up the interacting edge with topic distribution closest to that of the document in question. To be able to evaluate our suggestions, we define for a document d a set $Q_d^t = \{q_{d,1}, \ldots, q_{d,n}\}$ where Q_d^t is the set of queries used to access d in the time interval t. We consider K_d^t a successful suggestion of keywords for d at

Model	Avg. successful suggestions	Avg. recall
Our model	66%	0.62

Table 2: Results for keyword suggestion on query dataset. For 66% of documents, our algorithm is successful in suggesting *every* query that has been used to search for it.

time t if

$$Q_d^t \subseteq K_d^t \tag{16}$$

or in other words, if we at least suggest as keywords all queries that are used to reach the document.

This definition suggests that we can evaluate suggestions in terms of recall r defined in the usual way. In addition to the recall calculation, we also simply count the number of documents that are given successful suggestions according to (16) and calculate a percentage. This is done for every document in the test set. Both the recall and the percentage of successful suggestions averaged over 10 tests are presented in table 2.

We see that for 66% of documents, our algorithm is successful in suggesting *every* query used to reach the document as a keyword. One should also note that this indicates predictive capabilities since suggestions are made for unseen data. An example of a successful suggestion is the set of keywords [expense, travel expenses] for a document reached by queries [pension, expense, travel expenses, foundation]. An example of a failure is the keywords [jobs, job openings, job bank, region service] for a document reached by queries [jobs, job openings, vacancies], a suggestion that is still reasonable.

5. CONCLUSIONS

In this paper we have constructed a probabilistic graphical model for inference of time- search query interaction. We have found through tests on synthetic as well as real-world data that our model is capable of recovering hidden interaction dynamics in search query networks.

In synthetic tests, we have found that the hypergraph based model of this paper outperforms existing graph-based models. Future work should include theoretical analysis of time complexity to determine the practical usability of our model. A more rigorous set of tests of the quality of query interaction weight sequences is also needed to confirm the accuracy of the model.

We have also found that our model can be used in an application to document keyword suggestions, and that the suggested keywords are relevant in that they are actually used as search terms. However, more rigorous testing needs to be done in order to validate the performance of the method compared to state-of-the-art keyword extraction methods. Further, the impact of the temporal modeling needs to be demonstrated fully.

6. ACKNOWLEDGMENTS

The authors would like to thank Devdatt Dubhashi and Chiranjib Bhattacharyya for helpful suggestions during the course of this project. We would also like to thank Findwise AB, a consulting company in search solutions, and their employees for their insight and help and for providing us with data.

7. REFERENCES

[1] A. Ahmed and E. Xing. Timeline: A dynamic hierarchical dirichlet process model for recovering birth/death and evolution of topics in text stream. In *Proceedings of UAI*, 2010.

[2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *JMLR*, 9:1981–2014, 2008.

[3] E. Alfonseca, M. Ciaramita, and K. Hall. Gazpacho and summer rash: lexical relationships from temporal patterns of web search queries. In *Proceedings of EMNLP 2009*, pages 1046–1055, 2009.

[4] G. Amodeo, G. Amati, and G. Gambosi. On relevance, time and query expansion. In *Proceedings of CIKM 2011*, pages 1973–1976, 2011.

[5] R. Baeza-Yates, L. Calderón-Benavides, and C. González-Caro. The intention behind web queries. In *Proceedings of SPIRE*, volume 409 of *LNCS*, pages 98–109, 2006.

[6] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In *Proceedings of EDBT*, pages 588–596, 2004.

[7] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *Proceedings of KDD*, pages 76–85, 2007.

[8] C. Berge. *Graphs and hypergraphs*. Elsevier, 1976.

[9] D. M. Blei. Introduction to probabilistic topic models. *Commun. ACM*, 2011.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[11] B. Bollobás. *Modern graph theory*, volume 184. Springer Verlag, 1998.

[12] A. Z. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[13] C. Bron and J. Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM*, 16(9):575–577, 1973.

[14] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of KDD*, pages 875–883, 2008.

[15] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[16] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[17] F. Diaz and R. Jones. Using temporal profiles of queries for precision prediction. In *Proceedings of SIGIR 2004*, pages 18–24, 2004.

[18] J. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[19] W. Fu, L. Song, and E. P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of ICML 2009*, pages 329–336, 2009.

[20] G. Ghoshal, V. Zlatic, G. Caldarelli, and M. E. J.

Newman. Random hypergraphs and their applications. *CoRR*, abs/0903.0419, 2009.

[21] J. Guo, X. Cheng, G. Xu, and X. Zhu. Intent-aware query similarity. In *Proceedings of CIKM*, pages 259–268, 2011.

[22] J. Hu, G. Wang, F. Lochovsky, J. T. Sun, and Z. Chen. Understanding user's query intent with wikipedia. In *Proceedings of the WWW '09*, pages 471–480, New York, USA, 2009. ACM.

[23] V. Jethava, C. Bhattacharyya, D. Dubhashi, and G. N. Vemuri. Netgem: Network embedded temporal generative model for gene expression data. *BMC Bioinformatics*, 12(327), 2011.

[24] V. Jethava, L. Calderón-Benavides, R. Baeza-Yates, C. Bhattacharyya, and D. Dubhashi. Scalable multi-dimensional user intent identification using tree structured distributions. In *Proceedings of SIGIR*, pages 395–404, 2011.

[25] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 25(3), July 2007.

[26] J. Kaur and V. Gupta. Effective approaches for extraction of keywords. *Journal of Computer Science*, 7(6):144–148, 2010.

[27] S. Klamt, U. Haus, and F. Theis. Hypergraphs and cellular networks. *PLoS computational biology*, 5(5):e1000385, 2009.

[28] D. Koller and N. Friedman. *Probabilistic Graphical Models - Principles and Techniques.* MIT Press, 2009.

[29] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais. Understanding temporal query dynamics. In *Proceedings of WSDM 2011*, pages 167–176, 2011.

[30] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 1951.

[31] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. Hellerstein. Graphlab: A new parallel framework for machine learning. In *Conference on Uncertainty in Artificial Intelligence (UAI), Catalina Island, California*, 2010.

[32] R. Neal and G. E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[33] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

[34] K. Radinsky, K. Svore, S. Dumais, J. Teevan, A. Bocharov, and E. Horvitz. Modeling and predicting behavioral dynamics on the web. In *Proceedings of WWW 2012*, pages 599–608, 2012.

[35] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proceedings of WWW 2004*, pages 13–19, 2004.

[36] E. Sadikov, J. Madhavan, L. Wang, and A. Halevy. Clustering query refinements by user intent. In *Proceedings of WWW 2010*, pages 841–850, 2010.

[37] M. Shokouhi. Detecting seasonal queries by time-series analysis. In *Proceedings of SIGIR 2011*, pages 1171–1172, 2011.

[38] L. Song, M. Kolar, and E. P. Xing. KELLER: estimating time-varying interactions between genes. *Bioinformatics*, 25:i128–i136, 2009.

[39] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proceedings of WWW 2004*, pages 675–684, 2004.

[40] J. Teevan, S. Dumais, and E. Horvitz. Potential for personalization. *ACM Trans. on Computer*, Jan 2010.

[41] J. Teevan, S. Dumais, and D. Liebling. To personalize or not to personalize: modeling queries with variation in user intent. *Proceedings of SIGIR 2008*, pages 163–170, 2008.

[42] M. Wainwright and M. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in ML*, 1(1-2):1–305, 2008.