Temporal Summarization of Knowledge Evolutionary Trend

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Miscellaneous; H.3.3 [**Information Search and Retrieval**]: Text Mining

General Terms

Algorithms, Experimentation

Keywords

Cross collaboration, Social network, Predictive model

1. DOMAIN KNOWLEDGE

The standard topic modeling techiques decomposing the observed data into latent topics according to a purely data-driven objective function. This means that topic models inherit some of the disadvantages of unsupervised learning. For example, there may be multiple candidate partitions of the dataset which capture different aspects of the underlying structure.

Purely unsuperivsed topic modeling discover topics which represent strong statistical patterns but do not always correspond to user expectations of semantically meaningful topics.

Supervised LDA[Blei and McAuliffe, 2008] can be applied to labeled documents, augmenting each document d with a label variable y_d, which either categorical or continuous. Each y_d value is modeled by a Generalized Linear Model in the vector of mean topic counts $\bar{Z} = \frac{1}{N_d}\sum_{N_d}^{n=1} z_n$ for the document. This approach can therefore make label prediction by calculating the posterior topic assignments for a test document to obtain a $\bar{z}$ value. This model tends to produce topics that are able to "explain" the label value y for the training set. In this way the label information indirectly influences the topic decomposion discovered by the model.

In dynamic topic models, the corpus is partitioned into disjoint time slices. Using logistic normal distributions, both the document-topic mixtures and topic-word multinomials evolve via multivariate Gaussian dynamics (i.e., at time step s natural parameter v_s is Gaussian distributed with mean v_{s-1}). Topics over time modeling timestamps as being genreated by the model itself.

In standard LDA, topic-word multinomial distributions $\Phi_z = p(w|z)$ are drawn from a Dirichlet prior with hyperparameter β. To some extent, domain knowledge can be encoded into this Dirichlet prior by setting the values in the β hyperparameter vector. The standard Dirichlet prior can be replaced with a more expressive Dirichlet Forest Prior.

The Dirichlet Tree distribution reparameterizes and generalized the standard Dirichlet distribution, while maintaining conjugacy to the multinomial. In the Dirichlet Tree, the leaf nodes correspond to the mulitnomial probabilities. The root node is assigned probability mass 1, which then "flows" to its children in proportion to a sample from Dirichlet distriution parametrized by the out-edge weights. Each internal node then distributes the probability mass it recives to its children in the same way.

The Dirichlet forest prior encodes both "Must-Link" and "Cannot-Link". It yeilds a mixture model of Dirichlet subtree within each connected component, each corresponds to a maximal clique. For each topic, one subtree is selected according to probability

$$P(r) = |M_{rq}|, q = 1...Q^{(r)}.$$

Enssentially the selected subtree indexed by q tends to redistribute nearly all probability mass to the words within M_{rq}. Since there is no mass left for other cliques, it is impossible for a word outside clique M_{rq} to have a large probability. Therefore, no Cannot-Link will be violated.

LogicLDA allows the user to express domain knowledge in First-Order Logic.

- **Constants** are symbols that represent an actual object in the problem domain.
- **Variables** are symbols that can take on values from the set of constants.
- **Predicates** are symbols that express relations, and evaluate to *true* or *false* for different arguments.
- **Functions** are symbols that express mappings.
- **Terms** are any expressions that refer to objects in the domain.
- **Atoms** are predicates applied to terms.
- **Formulas** are constructed from atoms using logical connectives ($\wedge, \vee, \neg, \Rightarrow$).

WSDM'13, February 4–8, 2013, Rome, Italy.

- **Clauses** are formulas consisting of a disjunction of literals.

- **Ground** terms, atoms, or formulas contain no variables.

Markov Logic Networks are a class of graphical models operate over this type of logical domain. A "possible world" consists of a set of binary assignments for all possible ground predicates. A MLN then assigns probabilities to all possible worlds.

- $Z(i,t)$ is true if the hidden topic $z_i = t$, and false otherwise.

- $W(i,v)$ is true if word $w_i = v$, and false otherwise.

- $D(i,j)$ is true if $d_i = j$, and false otherwise.

The model probability of LogicLDA can be interpreted as the product of the individual MLN and LDA contributions. Another perspective is that LogicLDA consists of an MLN augmented with continuous varibles (Θ, Φ) and associated potential functions.

The goal of LogicLDA is to learn the most likely ϕ and θ in the model. As in standard LDA, the latent topic assignment z cannot be marginalized out in practice due to their combinatorial nature. We instead aim to find the maximum a posteriori estimate of z, Θ, Φ jointly. This can be formulated as maximizing the logarithm of the unnormalized probability.

2. SUMMARIZATION FRAMEWORK

The major motivation of our work is to find a concise and intuitive summarization of a given research topic. More concretely, we want to select a set of representative terms that best describe the hot spots and major achievements along the development of the research topic, meanwhile, capture the temporal evolutionary pattern within the research area. We formulated the problem as a graph partitioning task. At a high level, the proposed summarization framework consists of three stages.

- **Document retrieval.** First, given a research topic q needs to be summarized, we find a documents collection $D = \{d_t\}^T$ that can represent the development of the research topic, where d_t denote the documents at time t.

- **Dynamic concept graph construction.** Second, we extract knowledge concept terms mentioned in each document and construct a graph G, where each term w_i at time slide t is associated with a node n_i^t. We explicitly model the semantic relationship and evolving patterns with the graph by associating feature scores on the nodes and edges.

- **Trend partitioning.** Third, we utilize a message passing algorithm on the constructed dynamic concept graph to select a set of terms based on their *authority*, *burstiness* and network structures, and further partition the graph into evolutionary trends.

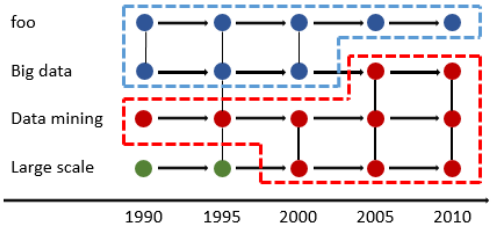
2.1 Document Retrieval

To summarize a given research topic t (i.e., user's query), we need to first map the topic to a set of documents. A straightforward idea is to use some traditional information retrieval technique to find the relevant documents. However, such an approach is not appropriate for the dynamic summarization task due to the following reasons. First, we want to summarize the development of the topic over time, if we use such an approach, we won't be able to trace how a knowledge concept evolve from and emerge into some other related concepts. Second, it has obvious bias that most retrieved documents will contain the queried words. Third, if a query contains a rare used term or a new term, it will face the problem of data sparseness.

To overcome the disadvantages above, we use a community-base document retrieval method. We first find a core research community (a group of experts) related to the topic, and then aggregate all the members' research work as the document collection to summarize. Such an approach is very intuitive since these researchers are leading the development the research area. To avoid some authoritative experts' work dominated in the documents and introduce potential bias, we normalize each member's contribution averagely.

2.2 Dynamic Concept Graph Construction

After the initial retrieval of related document collection, we extract knowledge concepts mentioned in each documents. In this work, we simply use wikipedia titles as a vocabulary to extract terms. With the extracted terms of each document over time, we construct a dynamic concept graph $G = \{V, E_r, E_e\}$ to model the semantic relationship and evolutionary pattern between terms uniformly. For a term w_i, we create a node n_i^t for each time slide t. There are two types of edges in the graph, E_r are representative edges denotes how well-suited for a term to act as a representative of another term at a given time. E_e are evolving edges connecting the same terms within two adjacent time slides indicating to what extent the meaning of the term has shifted.



- **Representativeness:** To capture the semantic relationship between terms, we connect terms co-occurred in the documents at the same time slide with representative edges. Formally, the weight of the directed edge (n_i^t, n_j^t) is defined based on mutual information, indicates how appropriate it is to use term w_i to represent w_j at time t. We calculate mutual information between term w_i and term w_j at the given time slide

t as:

$$I(n_i^t, n_j^t) = p(w_i \in d_t, w_j \in d_t) log \frac{p(w_i \in d_t, w_j \in d_t)}{p(w_i \in d_t)p(w_j \in d_t)} + p(w_i \in d_t, w_j \notin d_t) log \frac{p(w_i \in d_t, w_j \notin d_t)}{p(w_i \in d_t)p(w_j \notin d_t)} + p(w_i \notin d_t, w_j \in d_t) log \frac{p(w_i \notin d_t, w_j \in d_t)}{p(w_i \notin d_t)p(w_j \in d_t)} + p(w_i \notin d_t, w_j \notin d_t) log \frac{p(w_i \notin d_t, w_j \notin d_t)}{p(w_i \notin d_t)p(w_j \notin d_t)} \quad (1)$$

where d_t denotes all the retrieved documents at time t. The representativeness of term w_i to term w_j at time t is then defined as normalized mutual information

$$s(n_i^t, n_j^t) = \frac{I(n_i^t, n_j^t)}{I(n_i^t, I_i^t)}$$

- **Meaning shift:** Nodes corresponding to the term w_i within two adjacent time slides $-1t$ and t are connected by an evolving edge (n_i^{t-1}, n_i^t), indicates that the term is evolved from itself at the last time slide. The weight of the evolving edge indicates whether the meaning of the term is staying the same or shifted from time $t-1$ to t. Formally the weight is defined based on Jaccard coeffient.

$$s(n_i^{t-1}, n_i^t) = \frac{|NB(n_i^{t-1}) \cap NB(n_i^t)|}{|NB(n_i^{t-1}) \cup NB(n_i^t)|}$$

- **Burstiness:** We model burstiness by assuming the arrival of terms as an unknown binomial distribution, and use χ^2 tests to check for significant association between words and time periods. We calculate the contingency table as below and $\chi^2 = \frac{(ad-bc)^2 n}{(a+b)(c+d)(a+c)(b+d)}$.

-	W	$\overline{W}$
t	a	b
$< t$	c	d

We define burstiness of term w_i at time t as $b_i^t = \chi^2$ value.

- **Authority:** We simply defined the authority of a term at a given time by $u_i^t = df_t(w_i)/|d_t|$, where $df_t(w_i)$ indicates the document frequency of w_i at time t, and $|d_t|$ is the number of documents at time t.

2.3 Trend Partitioning

In order to give a concise and intuitive summarization, we need to partition the dynamic concept graph into evolutionary trends, and select a set of representative terms best describe the development of the research topic. Here, we use a massage passing algorithm to perform the selection and partition. The method is analogous to affinity propagation(AP) algorithm proposed in Frey et al. and Tang et al, the difference is that our method is modified to fit the dynamic setting. The AP algorithm performs clustering by identifying exemplars. Essentially it solves the following optimization problem

$$c^* = argmin(-\sum S(i, c_i))$$

Input: dynamic concept graph $G = \{V, E_r, E_e\}$;
Output: a set of representative nodes V_r and the par;
Initialize all $r_{ij}^t \leftarrow \mathbf{0}$;
repeat
  foreach *edge* (n_i^t) **do**
    //Initialization;
    $L \leftarrow$ initialization list;
    Factor graph $FG \leftarrow BuildFactorGraph(L)$;
    // Learn the parameter θ for factor graph model;
    repeat
      foreach $v_i \in order$ **do**
        Update the messages of v_i by Eqs. **??** and **??**;
      end
    until *(all messages μ do not change)*;
    foreach $\theta_i \in \theta$ **do**
      Calculate gradient ∇_i according to Eq. **??**;
      Update $\theta^{new} = \theta^{old} + \eta \cdot \nabla_i$;
    end
  end
until *converge*;

Algorithm 1: Message passing algorithm on dynamic concept graph.

where $C = (c_i)$ is the mapping between nodes and exemplars, $S(i, c_i)$ indicates the similarity between i and its exemplar. $S(i, i)$ is the penalty for i to being exemplar of itself.

In the algorithm, we introduce three sets of variables $\{r_{ij}^t\}$, $\{a_{ij}^t\}$ and $\{e_i^{t-1,t}\}$. Where r_{ij}^t indicates the how well-suited term w_j is to serve as an exemplar of term w_i (i.e. w_j covers the meaning of w_i) at time slide t. a_{ij} denotes the availabilities of term w_j to serve as an exemplar of w_i at time t. $\{e_i^{t-1,t}\}$ indicates the equivalence between the term w_i at two adjacent time slides which captures to what extent the meaning of term w_i has shifted from $t-1$ to t. All the r_{ij}^t, a_{ij}^t and $e_i^{t-1,t}$ are set to 0 initially, their values are updated iteratively with following rules until convergence.

$$r_{ij}^t = (1-\lambda)\rho_{ij}^t + \lambda r_{ij}^t \quad (2)$$

$$a_{ij}^t = (1-\lambda)\alpha_{ij}^t + \lambda a_{ij}^t \quad (3)$$

where λ is a damping factor, ρ_{ij}^t and α_{ij}^t are messages passing from n_i^t to n_j^t. The massage passing rules are defined as follows:

$$\rho_{ij}^t = s(n_i^t, n_j^t) - \max_{k \in NB(n_j^t)} \{s(n_i^t, n_k^t) + a_{ik}^t\} \quad (4)$$

$$\alpha_{ii}^t = \sum_{k \in NB(n_i^t)} \max\{0, r_{ij}^t\} \quad (5)$$

$$\alpha_{ij}^t = min\{0, r_{jj}^t + \sum_{k \in NB(n_j^t)} \max\{0, r_{ik}^t\}\} \quad (6)$$

where $NB(n_j^t)$ denotes the neighboring nodes of node n_j^t, $s(n_i^t, n_j^t)$ is w_i's representativeness of w_j at time t. $s(n_i^t, n_i^t)$ is set to $s(n_i^t, n_i^t) = \beta b_i^t + (1-\beta)u_i^t$ indicates that the nodes with high authority and burstiness are preferred to be choose as exemplars, where β is a parameter controlling the trade-off between timeliness and importance.

3. REFERENCES