Supervised graph summarization for structuring academic search results

Daniil Mirylenka
University of Trento
via Sommarive 5 - 38123 Trento, Italy
dmirylenka@disi.unitn.it

Andrea Passerini
University of Trento
via Sommarive 5 - 38123 Trento, Italy
passerini@disi.unitn.it

Abstract

In this paper we address the problem of visualizing the query results of the academic search services. We suggest representing the search results as concise topic hierarchies, and propose a method of building such hierarchies through summarization of the intermediate large topic graphs. We describe a supervised learning technique for summarizing the topic graphs in the most informative way using sequential structured prediction, and discuss our ongoing work on the interactive acquisition of the training examples.

1 Introduction

Perhaps every researcher nowadays has used an academic search engine. Google Scholar, Microsoft Academic Search, arXiv are but a few examples of the popular services for finding scientific publications. The typical presentation of the query results in the form of a relevance-sorted list is convenient when one searches for a specific paper, but provides very little help for understanding the retrieved results as a whole. In our recent work (Mirylenka & Passerini (2013b)) we proposed using hierarchical topic maps to serve as a concise visual representation of the results and an additional browsing control. A topic map is a small taxonomy built so as to summarize the topics present in the search results in an informative way. In order to build a topic map we first map the abstracts of the retrieved papers to the articles and categories of Wikipedia and extract a large graph of relevant topics linked with hierarchical relations. After a number of preprocessing steps, the topic graph is summarized into a concise topic map using a supervised learning approach (Mirylenka & Passerini (2013a)). In this paper we describe the proposed supervised method of summarizing the topic graphs. We also discuss some efficiency improvements and report on our current work in the direction of collecting the training data in an active and online fashion.

2 Summarizing the topic graph

We assume that we are given as an input a directed acyclic graph $G(V, E)$ where nodes V are topics, and links E correspond to their parent-child (subtopic) relations. In addition, we have a set of documents D representing the search results, and a topic-document relation R describing which topics are assigned to which documents. We refer the reader to (Mirylenka & Passerini (2013b), Mirylenka & Passerini (2013a)) for the description of how this graph is built. The goal is, for a given number of topics T, to select a subgraph $G_T \subset G$ that represents the documents and the topic graph in the most informative way. The notion of "informativeness" is vague and likely subjective, which is why it cannot be encoded explicitly. Instead, we define the properties favorable for good topic summaries and learn their correct proportions from examples. We elaborate on the properties in Section 2.3, while in the following section we describe the problems of learning and prediction.

2.1 The Learning Problem

Posed as a structured prediction problem, the goal is to learn a scoring function

$$F_w(G, R, G_T) = \langle w, \Psi(G, R, G_T) \rangle \tag{1}$$

that is maximized by good topic summaries, and then construct summaries for new graphs G by maximizing this function over the set of all possible subgraphs:

$$\hat{G}_T = \underset{|G_T|=T}{\operatorname{argmax}} F_w(G, R, G_T). \tag{2}$$

Computing the `argmax` is generally prohibitively expensive, as it requires evaluating the scoring function over $\binom{|V|}{T}$ subgraphs. We alleviate this problem by imposing an additional constraint that is natural for our settings. Specifically, we require that for a given input graph G the optimal topic summaries of different sizes should be nested:

$$\hat{G}_1 \subset \hat{G}_2 \subset ... \subset \hat{G}_T.$$

In other words, bigger summaries are obtainable from smaller ones by only adding new topics:

$$\hat{G}_t(\hat{V}_t, \hat{E}_t), \hat{G}_{t+1}(\hat{V}_{t+1}, \hat{E}_{t+1}) \Rightarrow \hat{V}_{t+1} = \hat{V}_t \cup \{v_{t+1}\}.$$

This requirement is justified by the principle of least surprise: when switching from less to more detailed summaries, the user will likely not expect the topics to disappear. Considering this requirement, the problem can be reformulated as predicting the sequence of topics $\hat{v}_1, \hat{v}_2, ..., \hat{v}_T$ whose prefixes constitute the node sets of the intermediate summary graphs. Assuming that we have "ground truth" examples of the form $((G, R), (v_1, ..., v_T))$, we can view this as an imitation learning problem, in which we want to copy the expert's behaviour in selecting the topics $(v_1, ..., v_T)$.

DAGGER (Dataset Aggregation) framework (Ross et al. (2011)) allows us to cast this problem into the problem of training a local policy that predicts the best next action (topic v_{t+1}) given the current state (initial input (G, R) plus the current summary $\hat{G}_t$). In essence, DAGGER ensures that such a policy behaves well when applied to its own-generated rather then optimal states. The way this is accomplished is by iteratively retraining the policy on a updated training set. At each iteration the training set is augmented with examples $((G, R, \hat{G}_t), v^{opt}_{t+1})$, in which inputs $(G, R, \hat{G}_t)$ are produced by the current policy, and outputs v^{opt}_{t+1} are optimal actions provided by the expert.

Applying DAGGER requires two ingredients:

1. a policy $\pi : (G, R, \hat{G}_t) \mapsto \hat{v}_{t+1}$ that can be trained on examples $((G, R, \hat{G}_t), v_{t+1})$, and
2. an 'expert' $\pi^* : (G, R, v_1, v_2, ..., v_T, \hat{G}_t) \mapsto v^{opt}_{t+1}$ that can produce optimal actions v^{opt}_{t+1} given the true optimal sequence $v_1, v_2, ..., v_T$ and a current (non-optimal) state $(G, R, \hat{G}_t)$.

Providing the policy. In order to build the policy π, we need a classifier that can learn how to map an intermediate topic graph $(G, R, \hat{G}_t)$ to the best next topic $\hat{v}_{t+1}$. We view this as structured prediction problem similar to the formulations (1, 2). Specifically, during training we would like to learn a linear function

$$F_w(G, R, \hat{G}_t, v_{t+1}) = \langle w, \Psi(G, R, \hat{G}_t, v_{t+1}) \rangle \tag{3}$$

that is maximized by optimal decisions v_{t+1}. The prediction is then performed by maximizing the learned function for a given input $(G, R, \hat{G}_t)$ over the possible set of topics:

$$\hat{v}_{t+1} = \underset{v_{t+1} \notin \hat{V}_t}{\operatorname{argmax}} F_w(G, R, \hat{G}_t, v_{t+1}). \tag{4}$$

The important distinction from the formulations (1, 2) is that the `argmax` is computed over the set of topics rather then subgraphs, which is feasible. We solve this prediction problem by using the SVM^{rank} instantiation of the SVM^{struct} software (Joachims (2006)). In order to compute the partial ranking $r_{G,R,\hat{G}_t}$ of different solutions $\hat{v}_{t+1}$ for a given input $(G, R, \hat{G}_t)$, we define a loss function between the sequences

$$\ell_{G,R}((v_1, v_2, ..., v_{t+1}), (v'_1, v'_2, ..., v'_{t+1})), \tag{5}$$

and compute it with respect to the optimal decision:

$$r_{G,R,\hat{G}_t}(\hat{v}_{t+1}) = -\ell_{G,R}((\hat{v}_1, \hat{v}_2, ..., \hat{v}_{t+1}), (\hat{v}_1, \hat{v}_2, ..., v_{t+1})).$$

In our experiments we defined $\ell_{G,R}$ as a 0/1 loss, which corresponds to specifying no preferences between non-optimal decisions, which in turn results in an easier problem for SVM^{rank}.

Providing the expert actions. At each iteration of DAGGER we need to compute the optimal actions v_{t+1}^{opt} for all states $(G, R, \hat{G}_t)$ generated by our current policy. This is accomplished by minimizing the loss with respect to the true optimal sequence:

$$v_{t+1}^{opt} = \underset{\hat{v}_{t+1}}{\operatorname{argmin}}\, \ell'_{G,R}((\hat{v}_1, \hat{v}_2, ..., \hat{v}_{t+1}), (v_1, v_2, ..., v_{t+1})).$$

In these settings 0/1 loss would be inappropriate, as it would score all non-optimal sequences equally, rendering the minimization problem meaningless in most cases. An obvious candidate for $\ell'_{G,R}$ is the Jaccard distance between the sequences viewed as sets of topics. However, it turns out that Jaccard distance does not take into account the similarity between the topics: it tends to add topics present in the optimal sequence, even when the non-optimal partial sequence already contains similar topics. In other words, it encourages redundancy in the built topic summaries.

We designed a matching-based loss function $\ell'_{G,R}$ that does not suffer from this problem:

$$\ell'_{G,R}(..., ...) = 1 - match(..., ...). \tag{6}$$

The matching score greedily assigns best-scoring candidate topics to the topics from the optimal sequence, starting from the first optimal topic v_1. The score of the assignment (v, v') is computed as the Jaccard distance between the sets of documents transitively associated with the topics v and v', plus a constant α if the topics are the same. The final matching score is the average of the assignment scores divided by $1 + \alpha$.

2.2 Connecting topics and documents

The learning procedure described above allows us to sequentially select the topics $v_1, v_2, ..., v_T$ to be included into the topic summary G_T. In order to completely define the summary graph, we need to decide how to connect the topics with links. On one hand, we want to maintain the hierarchical relations between the topics in the graph, but on the other hand, we do not want to clutter the topic summary with superfluous links. The way we solve this problem is by introducing the *minimum* possible number of links that still maintain the hierarchical structure of the original topic graph: for every pair $v_i, v_j \in V_T$ such that v_i is an ancestor of v_j in the original graph G, v_i must be the ancestor of v_j in the topic summary G_T. Technically this amounts to computing the transitive closure $G^+(V, E^+)$ of the original graph, selecting the subgraph of G^+ containing the nodes $v_1, v_2, ..., v_T$ and computing the transitive reduction of the result.

It is important to mention how documents are assigned to the nodes of the summary graph. After the summary is built, we compute a new transitive topic-document relation R_T^+. A document d is assigned to a topic v whenever it was assigned to any of the subtopics of v in the original graph:

$$R_T^+ = \{(v, d) | v \in V_T, \exists v' \in V, (v, v') \in E^+, (v', d) \in R\}.$$

2.3 Features

Computing the joint feature representation $\Psi(G, R, \hat{G}_t, v_{t+1})$ for the problem (3, 4) is an important step of the overall procedure. It is the features that allows the policy π to learn what makes for a good topic summary. The features $\Psi(G, R, \hat{G}_t, v_{t+1})$ describe various properties of the graph $\hat{G}_{t+1}$ that results from adding the topic v_{t+1} to the summary $\hat{G}_t$. First of all we encode the properties that we think could be important for discriminating informative summaries. We have thus implemented various features describing the relevance of the topics to the search results, the diversity of the topics, as well as the shape of the resulting summary as a graph. Here we list some of these features in no particular order: *a*) *document coverage* (the number of documents assigned to any of the topics in the summary), *b*) *transitive document coverage* (the number of documents assigned to any subtopic of any topic in the summary), *c*) average and minimum *topic frequency* (a number of documents

assigned to a topic), *d*) average and minimum *transitive frequency* (a number of documents assigned to any subtopic of a topic), *e*) average and maximum topic *overlap* (Jaccard distance between the two topics viewed as sets of documents), *f*) average and maximum parent-child *overlap*, *g*) average pairwise *distance* between the topics, where *distance* is the length of the shortest path through a common ancestor in the original graph, *h*) *partition coefficient* (measuring the crispness of topics viewed as fuzzy clusters of documents).

The features describing the shape of the summary graph include: *a*) the number of connected components, *b*) the number of links, *c*) the height of the graph, *d*) the average number of children for topics having children, *e*) the average number of parents for topics having parents.

An important practical concern is the efficient implementation of the features. For a single prediction task $G_t \mapsto v_{t+1}$, we need to compute the features $\Psi(G, R, G_t, v_{t+1})$ for every possible topic v_{t+1}. Being the graph properties, most of the features require the time polynomial in the number of nodes and links in the topic graph. As a result, the frequent evaluation of the features accounts for the most of the computational cost of the overall problem. We implemented most of the required graph properties in an incremental way, such that the properties of G_{t+1} could be efficiently computed given the properties of G_t. For instance, it is easy to see how *the minimum topic frequency* of G_{t+1} could be computed from the minimum topic frequency of G_t in just a constant time. For the features describing the shape of the topic graph the incremental computation is not always straightforward. Given our procedure for establishing the links in the topic summary (see Section 2.2), we have not yet come up with the incremental computation of the features relying on the links (such as the average number of links, the average number of parents/children, etc.). We plan to investigate the efficient computation of these and other graph-based features.

3 Collecting the "Ground Truth" Topic Summaries

The main bottleneck of the proposed method lies in the necessity to collect "ground truth labels"—the sequences of topics constituting the subjectively optimal summaries of the topic graph. Selecting even a short sequence of informative topics from a large graph is a nontrivial task that requires considerable expertise in the area. A single expert is thus able to provide labels only for a limited—both in number and diversity—set of queries. This motivates us to look for the ways to simplify this procedure and possibly involve users in the process of acquiring the labels.

For this purpose we developed an interface that assists the collection of the "ground truth" topic sequences (Figure 1). The interface allows the user to interactively select the topics, one at a time, from a set of suggested candidates. Both the candidate topics and the topics selected so far are shown together in a topic graph. When a topic is selected, the set of candidates is recomputed to take into account the new information. If the user considers a certain candidate topic useless, he/she can chose to "hide" it, at which point it is replaced with a new candidate and is never suggested again in this session. Both selecting and hiding a topic takes a single click. The user can discard the collected topic sequence at any time, or save it and proceed with a new query.

3.1 Suggesting the candidate topics

In order to compute the suggestions we maintain the policy trained on the previously collected sequences. Given an intermediate summary, the policy predicts the best next topic to be included into the summary. The prediction is done by maximizing a scoring function (see Section 2.1, Eq. 4). This allows us to compute the candidate list as the top k highest scoring topics with respect to the current summary (consisting of the topics selected so far), and according to the current policy.

Ideally, we would like to update the current policy at each step, with every new user input. The current implementation of the learning procedure, however, makes this impractical. The iterative nature of DAGGER does not allow incremental updates to the policy: with every new data point, the iterations have to be run from scratch on the whole dataset. This means that in practice we have to retrain the current policy periodically in the background, in the batch fashion. As a future research direction we plan to work on a truly online version of the described learning method that could update the policy efficiently with every single training example. We envision that, in addition to the original goal of building the static topic maps, this kind of online method could open the way for personalized browsing interfaces that adapt to the users' preferences and information seeking tasks.

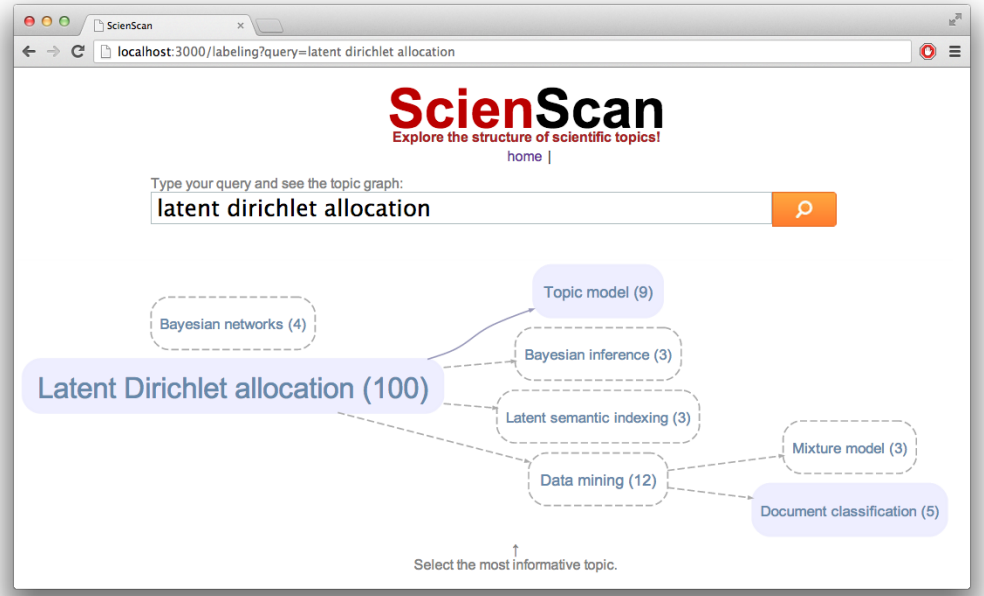

Figure 1: The interface for collecting the "ground truth" topic sequences.

References

Joachims, Thorsten. Training linear SVMs in linear time. In *SIGKDD*, pp. 217–226. ACM, 2006.

Mirylenka, Daniil and Passerini, Andrea. Learning to grow structured visual summaries for document collections. In *ICML Workshop on Structured Learning: Inferring Graphs from Structured and Unstructured Inputs*, 2013a.

Mirylenka, Daniil and Passerini, Andrea. Navigating the topical structure of academic search results via the Wikipedia category network. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM 2013)*. ACM, 2013b.

Ross, Stéphane, Gordon, Geoffrey J., and Bagnell, Drew. A reduction of imitation learning and structured prediction to no-regret online learning. *Journal of Machine Learning Research - Proceedings Track*, 15: 627–635, 2011.