Active Information Retrieval

Tommi Jaakkola
MIT AI Lab
Cambridge, MA
tommi@ai.mit.edu

Hava Siegelmann
MIT LIDS
Cambridge, MA
hava@mit.edu

Abstract

In classical large information retrieval systems, the system responds to a user initiated query with a list of results ranked by relevance. The users may further refine their query as needed. This process may result in a lengthy correspondence without conclusion. We propose an alternative active learning approach, where the system responds to the initial user's query by successively probing the user for distinctions at multiple levels of abstraction. The system's initiated queries are optimized for speedy recovery and the user is permitted to respond with multiple selections or may reject the query. The information is in each case unambiguously incorporated by the system and the subsequent queries are adjusted to minimize the need for further exchange. The system's initiated queries are subject to resource constraints pertaining to the amount of information that can be presented to the user per iteration. We also provide preliminary theoretical results concerning the rate of information acquisition.

1 Introduction

An IR system consists of a collection of documents and an engine that retrieves documents described by users queries. In large systems, such as the Web, queries are typically too vague, and hence, an iterative process in which the users refine their queries gradually has to take place. Since much dissatisfaction of IR users stems from long, tedious repetitive search sessions, our research is targeted at shortening the search session. We propose a new search paradigm of *active information retrieval* in which the user initiates only one query, and the subsequent iterative process is led by the engine/system. The active process exploits optimum *experiment design* to permit minimal effort on the part of the user.

Our approach is related but not identical to the interactive search processes called *relevance feedback*. The primary differences pertain to the way in which the feedback is incorporated and queried from the user. In relevance feedback, the system has to deduce a set of "features" (words, phrases, etc.) that characterize the set of selected relevant documents, and use these features in formulating a new query (e.g., [5, 6]). In contrast, we cast the problem as a problem of estimation and the goal is to recover the unknown *document weights* or relevance assessments.

Our system also relates to the Scatter/Gather algorithm of browsing information systems [2], where the system initially scatters the document collection into a fixed number k of clusters whose summaries are presented to the user. The user select clusters from a new sub-collection, to be scattered again into k clusters, and so forth, until enumerating single documents. In our approach, documents are not discarded butg rather their weighting is updated appropriately. Like many other clustering methods, the scatter/gather is based on hierarchical orderings. Overlapping clusters were recently proposed to better match real-life grouping and allow natural summarizing and viewing [4].

This short paper focuses on the underlying methodology of the active learning approach. The more general perspective will be discussed in the longer version of the paper.

2 The Framework

Let $\mathcal{X}$ be the set of documents in the database. We define a weighting of these documents as a distribution θ_x, $\sum_{x\in\mathcal{X}}\theta_x = 1$. We assume that in the context of any specific retrieval task, the user preferences are (probabilistically) consistent with one such weighting θ^*_x. The goal of a retrieval algorithm is therefore to recover this underlying weighting through interactions with the user. The resulting (approximation to) θ^*_x can be used to correctly rank the documents or, for example, to display all the documents with sufficiently large weight (cf. coverage) or the top 10 documents. Naturally, θ^*_x changes from one retrieval task to another and has to be inferred separately in each task. We might estimate a user specific prior (model) over the document weights to reflect consistent biases that different users have across retrieval tasks.

We express our prior belief about the document weights as a Dirichlet distribution

$$P(\theta) = \frac{1}{Z}\cdot\prod_{x\in\mathcal{X}}\theta_x^{\alpha_x-1}, \text{ where } Z = \frac{\prod_{x\in\mathcal{X}}\Gamma(\alpha_x)}{\Gamma(\sum_{x=1}^{n}\alpha_x)} \tag{1}$$

and $\sum_{x\in\mathcal{X}}\theta_x = 1$.

Let $\mathcal{C} = \{C_1, \ldots, C_m\}$ be the set of available clusters of documents. This set typically includes also the individual documents in the database and may have been obtained through a flat, hierarchical, or overlapping clustering method. For simplicity, we will initially assume that the clusters are either nested or disjoint, i.e., can be organized hierarchically. The clustering need not be static, however, and could be easily defined dynamically at each iteration.

Given the set of available clusters, we may choose a *query set*, the set of clusters that are presented to the user for selection. The user is expected to choose the best matching cluster in this set. In case of multiple selections, we will interpret the marked clusters as a redefined cluster of the query set. While this interpretation will result in suboptimal choices for the query set assuming the user consistently selects multiple clusters, the interpretation nevertheless obviates the need for modeling user's selection biases in this regard. An empty selection, on the other hand, suggests that the clusters outside the query set are deemed more likely. In the longer version of the paper, we will also consider feed-back other than the simple selection. This includes, for example, non-stochastic relevance annotation of the clusters in the query set.

The retrieval algorithm flows as follows: (1) It finds a small subset $\mathcal{S}$ of clusters to present, along with their summaries, to the user; (2) It waits until the user selects none, one or more of the presented clusters which best match their interest; (3) It uses the evidence from the user's selections to update the distribution $P(\theta)$ over the document weightings; (4) It outputs the top documents so far, ranked by their weights, and the iteration continues until terminated by the user or the system (based on any remaining uncertainty about the weights or the implied ranking).

The following sections address two primary issues: how to select the clusters to present to the user and how to incorporate the information from user selections. Since the second topic is a prerequisite for the former, we commence with the inference problem.

3 Inference

Suppose a distribution $P(\theta)$ over the document weights and a fixed query set $\mathcal{S} = \{C_{s_1}, \ldots, C_{s_k}\}$. We show next how to evaluate the posterior distribution $P(\theta|y)$ given the user response y. The key is to transform the *flat* Dirichlet distribution $P(\theta)$ into a hierarchical form so as to explicate the portion of the distribution potentially affected by the user response. The hierarchy, illustrated in Figure 1, contains three levels: the first level corresponds to selecting $\mathcal{S}$ or $\mathcal{X} \setminus \mathcal{S}$; the second one pertains to choices within the query set $\mathcal{S}$ (of most interest to us) as well as those under $\mathcal{X} \setminus \mathcal{S}$; the third level explicates the selections within the clusters C_{s_l} in $\mathcal{S}$.

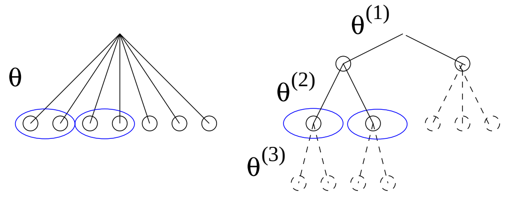

Figure 1: A three level hierarchical transform of a flat Dirichlet.

We use $\theta_i^{(1)}, i = 1, 2$ to denote the top level parameters, $\theta_{j|1}^{(2)}$, $j = 1, \ldots, k$ for the cluster choices within the query set whereas $\theta_{x|2}^{(2)}$, $x \notin \mathcal{S}$ gives the document choices outside $\mathcal{S}$. Finally, $\theta_{x|j}^{(3)}$ for $x \in C_{s_j}$ indicate the parameters associated with the cluster $C_{s_j} \in \mathcal{S}$. The original flat Dirichlet $P(\theta)$ can be written as

$$P(\theta^{(1)}) P(\theta_{\cdot|1}^{(2)}) P(\theta_{\cdot|2}^{(2)}) \left[\prod_{l=1}^{k} P(\theta_{\cdot|l}^{(3)}) \right] \tag{2}$$

with the appropriate normalization constraints. If clusters in $\mathcal{S}$ overlap, the expansion is carried out in terms of the disjoint subsets. The parameters governing the

Dirichlet component distributions are readily obtained by gathering the appropriate parameters α_x of the original Dirichlet (cf. [3]). For example,

$$\alpha_0^{(1)} = \sum_{x \in \mathcal{S}} \alpha_x, \tag{3}$$

$$\alpha_{j|1}^{(2)} = \sum_{x \in C_{s_j}} \alpha_x, \text{ for } j = 1, \ldots, k, \tag{4}$$

$$\alpha_{x|2}^{(2)} = \alpha_x \text{ for } x \notin \mathcal{S}, \tag{5}$$

$$\alpha_{x|j}^{(3)} = \alpha_x, \text{ whenever } x \in C_{s_j}, j = 1, \ldots, k \tag{6}$$

If user selects cluster C_{s_y}, we will update $P(\theta_{\cdot|1}^{(2)})$ which reduces to adjusting the counts $\alpha_{y|1}^{(2)} \leftarrow \alpha_{y|1}^{(2)} + 1$. The resulting new parameters give rise to the posterior distribution $P(\theta_{\cdot|1}^{(2)}|y)$ and by including the other components, to the overall posterior $P(\theta|y)$. If the user selects "none of these items," only the first level parameters $\theta_i^{(1)}$ will be updated.

4 Query set optimization

Our optimization criterion for chosing the query set $\mathcal{S}$ is the information that we stand to gain from querying the user with it. Let y indicate the user choice, the mutual information between y and the parameters θ is given by (derivation provided in a longer version of the paper)

$$I(y;\theta) = I(y;\theta_{\cdot|1}^{(2)}) = \sum_{y=1}^{k} P(y) \int P(\theta_{\cdot|1}^{(2)}|y) \log \frac{P(\theta_{\cdot|1}^{(2)}|y)}{P(\theta_{\cdot|1}^{(2)})} d\theta \tag{7}$$

$$= \sum_{y=1}^{k} P(y)\Psi(\alpha_{y|1}^{(2)} + 1) - \Psi\left(\sum_{x=1}^{k} \alpha_{x|1}^{(2)} + 1\right) + H(y) \tag{8}$$

where $P(y) = \alpha_{y|1}^{(2)} / (\sum_{x=1}^{k} \alpha_{x|1}^{(2)})$ defines our current expectation about user selection from $\mathcal{S}$; $H(y) = -\sum_{y=1}^{k} P(y) \log P(y)$ is the entropy of the selections y, and $\Psi(\cdot)$ is the Di-gamma function, defined as $\Psi(z) = d/dz \log \Gamma(z)$. Extending the criterion to "no selection" is trivial.

To simplify, we expand the counts $\alpha_{\cdot|1}^{(2)}$ in terms of the original (flat) counts α_x, and define for all clusters (whether or not they appear in the query set) the weights

$$a_i = \sum_{x \in C_i} \alpha_x, \tag{9}$$

$$b_i = a_i \Psi(a_i + 1) - a_i \log a_i \tag{10}$$

The mutual information criterion now depends only on $a_{\mathcal{S}} = \sum_{i=1}^{k} a_{s_i} = \sum_{x \in \mathcal{S}} \alpha_x$, the overall weight of the query set and $b_{\mathcal{S}} = \sum_{i=1}^{k} b_{s_i}$ which provides an overall measure of how informative the individual clusters in $\mathcal{S}$ are. With these changes, we obtain:

$$I(y;\theta_{\cdot|1}^{(2)}) = \frac{b_{\mathcal{S}}}{a_{\mathcal{S}}} + \log(a_{\mathcal{S}}) - \Psi(a_{\mathcal{S}} + 1) \tag{11}$$

We can optimize the choice of $\mathcal{S}$ with a simple greedy method that successively finds the next best cluster index i to include in the information set. This algorithm scales as $O(km)$, where m is the number of clusters in our database and k is the the maximal query set size in terms of the number of clusters.

Note that this simple criterion excludes nested or overlapping clusters from $\mathcal{S}$. In a more general context, the bookkeeping problem associated with the overlapping clusters is analogous to that of the Kikuchi expansion in statistical physics (cf. the use of this approximation in a belief propagation context [7]).

5 Projection back to a flat Dirichlet

The hierarchical posterior is not a flat Dirichlet anymore. To maintain simplicity, we project it back into a flat Dirichlet in the following KL-divergence sense

$$P'_{\theta|y} = \arg\min_{Q_\theta} KL(P_{\theta|y}||Q_\theta) \tag{12}$$

where $P(\theta|y)$ is the hierarchical posterior expressed in terms of the original flat variables $\theta_x, x \in \mathcal{X}$ (but no longer a flat Dirichlet). The transformation from hierarchical to flat variables is given by

$$\theta_x = \begin{cases} \theta_1^{(1)}\theta_{j|1}^{(2)}\theta_{x|j}^{(3)}, & x \in C_{s_j}, j = 1, \ldots, k \\ \theta_2^{(1)}\theta_{x|2}^{(2)}, & x \in \mathcal{S} \setminus \mathcal{X} \end{cases} \tag{13}$$

As a result, when $x \in C_{s_j}$ for some $j = 1, \ldots, k$ we get (derivation omitted)

$$E_{\theta|y} \log \theta_x = \Psi(\alpha_x) - \Psi(\sum_{z \in \mathcal{X}} \alpha_z) + \frac{\delta_{y,j}}{\sum_{z \in C_{s_j}} \alpha_z} - \frac{1}{\sum_{z \in \mathcal{S}} \alpha_z} \tag{14}$$

where y denotes the user selection. For $x \in \mathcal{X} \setminus \mathcal{S}$

$$E_{\theta|y} \log \theta_x = \Psi(\alpha_x) - \Psi(\sum_{z \in \mathcal{X}} \alpha_z) \tag{15}$$

If we define $r_x = E_{\theta|y} \log \theta_x$ for all $x \in \mathcal{X}$, then the counts β_x corresponding to the flat approximation Q_θ can be found by minimizing

$$D(P_{\theta|y}||Q_\theta) = \sum_{x \in \mathcal{X}} [\log \Gamma(\beta_x) - \beta_x r_x] - \log \Gamma(\sum_{x \in \mathcal{X}} \beta_x) + \text{const.} \tag{16}$$

where we have omitted any terms not depending on β_x. This is a strictly convex optimization problem over the convex set $\beta_x \geq 0, x \in \mathcal{X}$ and therefore admits a unique solution. Furthermore, we can efficiently apply second order methods such as Newton-Raphson in this context due to the specific structure of the Hessian: $\mathcal{H} = D - c\mathbf{1}\mathbf{1}^T$, where D is a diagonal matrix containing the derivatives of the di-gamma function[1] $\Psi'(\beta_x) = d/d\beta_x \, \Psi(\beta_x)$ and $c = \Psi'(\sum_{x \in \mathcal{X}} \beta_x)$. Each Newton-Raphson iteration requires only $\mathcal{O}(m)$ space/time.

[1] These derivatives can be evaluated efficiently on the basis of the highly accurate approximation to the di-gamma function (REFS).

6 Decreasing entropy

Since the query set was chosen to maximize the mutual information between the user selection and the parameters θ, we get the maximal reduction in the expected entropy of the parameters:

$$I(y;\theta) = H(P_\theta) - E_y\, H(P_{\theta|y}) \tag{17}$$

As discussed in the previous section, we cannot maintain the true posterior but have to settle for a projection. It is therefore no longer obvious that the expected entropy of the projected posterior possesses any analogous guarantees; indeed, projections of this type typically increase the entropy. We can easily show, however, that the expected entropy is non-increasing:

$$E_y \left\{ D(P_{\theta|y} \| P_\theta) - \min_{Q_\theta} D(P_{\theta|y} \| Q_\theta) \right\} = H(P_\theta) - E_y \left\{ H(P'_{\theta|y}) \right\} \geq 0 \tag{18}$$

since $P'_{\theta|y}$ is the minimizing argument. It is possible to make a stronger statement indicating that the expected entropy of the projected distribution decreases monotonically after each iteration.

Theorem 1 *For any* $\epsilon > 0$

$$E_y \left\{ H(Q_{\theta|y}) \right\} \leq H(P_\theta) - \left(\frac{k-1}{\alpha_{\mathcal{S}}} \right) \epsilon + \mathcal{O}(\epsilon^2) \tag{19}$$

where k is the size of the query set and $\alpha_{\mathcal{S}} = \sum_{z \in \mathcal{S}} \alpha_z$.

While this result is not tight, it does demonstrate that the projection back into a flat Dirichlet still permits a semi-linear descrease in the entropy. The denominator of the first order term, i.e., $\alpha_{\mathcal{S}}$, can increase only by 1 at each iteration.

It is important to note that the entropy of a Dirichlet distribution is not bounded from below (it is bounded from above). The manner in which the Dirichlet updates are carried out (how α_x change) still keeps the entropy a meaningful quantity.

7 Discussion

The active learning approach proposed here provides the basic methodology for optimally querying the user at multiple levels of abstraction. There are a number of extensions to the approach presented in this short paper. For example, we can encourage the user to provide a weighted selection among the presented clusters. In this case the selection is no longer stochastic but pertains more closely to the distribution (parameters θ) that we wish to recover. The information criterion (reducing to an ordinary KL-divergence) can be evaluated in this case and will be discussed in more detail in the longer version of the paper.

Our formulation also does not yet incorporate any information about the user confidence. If the level of confidence in the user selection is known, the rate of the update and learning can fit the user the best. Any confidence measure attached to the user selection can be incorporated into the update equations through equivalent sample size. This assumes consistence on the part of the user and, in future work, we will instead introduce a latent variable for user's confidence, and infer its value through consecutive user selections.

Our queries do not request a classification of all the presented clusters in terms of their relevance as is typically done in "classical" relevance feedback systems. The relevance is a binary decision and such cluster specific feedback can be incorporated into $P(\theta)$ through constraints (step functions) operating on the document weights θ_x. This formulation requires approximations to maintain the posterior through multiple iterations and for optimizing the query set. The details of this approach and the associated large deviation approximations will be discussed in the longer version of the paper.

We are also currently in the process of analyzing the fundamental trade-offs between the size of the query set (resource constraints) and the expected completion time of the retrieval process.

References

[1] A. C. Atkinson and A. N. Donev, Optimum experimental designs, Clarendon Press, 1992.

[2] D. R. Cutting, D. R. Karger, J. O. Pederson, J. W. Tukey, Scatter/Gather: A cluster Based Approach to Browse Document Collections, In Proceedings of the Fifteenth Annual International ACM SIGIR Conference, Denmark, June 1996.

[3] D. Heckerman, D. Geiger, and D. M. Chickering, Learning Bayesian Networks: The Combination of Knowledge and Statistical Data, Machine Learning, Vol 20, 1995.

[4] H. Lipson and H.T. Siegelmann, Geometric Neurons for Clustering, Neural Computation 12(10), August 2000

[5] J. J. Jr. Rocchio, Relevance Feedback in Information Retrieval, In The Smart System - experiments in automatic document processing, 313-323, Englewood Cliffs, NJ: Prentice Hall Inc.

[6] G. Salton and C. Buckley, Improving Retrieval Performance by Relevance Feedback, Journal of the American Society for Information Science, 41(4): 288-297, 1990.

[7] J.S. Yedidia, W.T. Freeman, Y. Weiss, Generalized Belief Propagation, Neural Information Processing Systems 13, 2001.