GRAPH STRUCTURE LEARNING FOR TASK ORDERING

Yiming Yang, Abhimanyu Lad, Henry Shu, Bryan Kisiel
Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{yiming,alad,henryshu,bkisiel}@cs.cmu.edu

Chad Cumby, Rayid Ghani, Katharina Probst
Accenture Technology Labs, Chicago, IL, USA
{chad.m.cumby,rayid.ghani,katharina.a.probst}@accenture.com

Keywords: Task-order optimization, graph structure learning, link analysis.

Abstract: In many practical applications, multiple interrelated tasks must be accomplished sequentially through user interaction with retrieval, classification and recommendation systems. The ordering of the tasks may have a significant impact on the overall utility (or performance) of the systems; hence optimal ordering of tasks is desirable. However, manual specification of optimal ordering is often difficult when task dependencies are complex, and exhaustive search for the optimal order is computationally intractable when the number of tasks is large. We propose a novel approach to this problem by using a directed graph to represent partial-order preferences among task pairs, and using link analysis (HITS and PageRank) over the graph as a heuristic to order tasks based on how important they are in reinforcing and propagating the ordering preference. These strategies allow us to find near-optimal solutions with efficient computation, scalable to large applications. We conducted a comparative evaluation of the proposed approach on a form-filling application involving a large collection of business proposals from the Accenture Consulting & Technology Company, using SVM classifiers to recommend keywords, collaborators, customers, technical categories and other related fillers for multiple fields in each proposal. With the proposed approach we obtained near-optimal task orders that improved the utility of the recommendation system by 27% in macro-averaged F1, and 13% in micro-averaged F1, compared to the results obtained using arbitrarily chosen orders, and that were competitive against the best order suggested by domain experts.

1 INTRODUCTION

Search, classification and recommendation techniques have been intensively studied and successfully applied as sub-domains of Information Retrieval (IR). An open research challenge is to make the best use of these techniques in a global context, to accomplish a higher level goal through a multiple-step process where task-specific search engines, classifiers and recommendation systems are needed in different steps. User relevance judgments in early steps are automatically propagated to later steps to enrich the context for later tasks. While research on “IR in context” has recently gained prominence [5], how to define and leverage context remains an important and challenging problem.

Consider the process of creating a project proposal in a large consulting organization. The principal investigators would rely on past information to make several key decisions about the new project, e.g., retrieving past proposals that are similar to the current one, finding companies that have been targeted (successfully or unsuccessfully) with this kind of project, choosing experts who should be included, engagements and achievements from the past that are relevant to the current proposal, and so on.

That is, in order to attain a complex goal, the users need to go through a multi-step process, and interact with different systems for retrieval, classification, expert finding, customer selection, etc., which provide useful suggestions to the user at each decision-making step, and receive feedback from the user. The performance of the system at each step is “context-sensitive”, i.e., it depends on the previously accomplished tasks because the user feedback at any of the earlier steps can be used as additional input to the system in each later step. The over-all performance, or the "utility", of the system in the multi-step process depends on the order of the tasks being carried out. Hence, finding the optimal order of tasks is important. However, manual specification of the near-optimal ordering may be non-trivial when the high-level goal is sufficiently complex or the dependencies among a large number of tasks are subtle.

We present a formulation of the task ordering problem as utility optimization for systems in a multi-step process and propose a relaxation of the objective function in terms of pair-wise order task order preferences. We show that the problem of finding the optimal order from partial orders is equivalent to a known NP-Hard problem. We propose an approximation in the form of a novel application of link analysis (HITS and PageRank) that is effective as well as computationally efficient.

2 A CONCRETE EXAMPLE

Consulting organizations generate documents as part of every project they undertake. Consider the task of assigning meta-data to these documents so that they can be searched and browsed more effectively in future.

Each document has multiple fields (Table 1), some containing free text, and others filled with categorical or nominal values. We consider filling each meta-data field by the authors or curators as a task. The tasks in each document are carried out step by step by the authors, using multiple prediction systems like enterprise-internal search engines, classifiers and recommendation systems to find relevant contents or items for each field. From the prediction systems' point of view, the input of the first task is a "bag of features" obtained from the text in the document. The input of the second task is the merged bag of features, including both the input and output of the previous task. Similarly, the input of any later task contains the initial input and all the outputs of the tasks prior the current one. Since the task orders are not fixed, any subset of the tasks can be possibly carried out before some task. Note that the output of each task is examined and corrected by the user, which corresponds to the user feedback that is propagated to subsequent prediction tasks.

Given a large collection of archived documents, we want to learn an ordering of the tasks so that the overall utility of the system is maximized through efficient propagation of user feedback. However, the order in which the fields in these documents were filled by the authors was not recorded. Therefore, from such a data collection, we cannot directly ascertain which task orders are more preferred, in terms of a given utility function. We must develop a new way for learning the optimal task order, as described below.

3. LEARNING TO ORDER

We want to formulate the problem in a way that good ordering of tasks can be automatically learned from the data. For this, we split the dataset into three subsets, and call them the *task-training set*, the *order-training set* and the *test set*, respectively. We use the *task-training set* to learn task-specific models for each task. We use the *order-training set* to empirically find the best ordering of the prediction tasks. We use the *test set* to evaluate the utility of the system in performing multiple tasks in the optimal order, as determined by each of the ordering methods, for cross-method comparison. We restrict our attention to classification tasks in the rest of the paper. However, the proposed task ordering method, which is the main focus of this paper, applies to any type of prediction tasks in general.

In the next section, we consider a naïve approach, which gives the exact solution to the problem but is computationally intractable. In Section 3.2, we propose a relaxation of the objective function in terms of partial order preferences among prediction tasks. In Section 3.3, we demonstrate a novel application of link analysis algorithms to find an approximately optimal task order based on a graph-theoretic representation of pair-wise preferences.

3.1 A Naïve Approach

Let *n* be the number of training instances (e.g., IT proposals), and *m* be the number of tasks per instance. Assume that all the task-specific models, i.e. classifiers, are already pre-trained, and the utility metric is pre-specified as well.

Given a specific order, i.e., a sequence of m tasks, its expected utility on all possible instances can be empirically estimated by running the task-specific models in the specified order on each of the training instances, and using human-assigned labels to evaluate the utility of the system-made predictions. Comparing the average utility scores of all possible orders, we can find the best order. However, this approach is computationally intractable because the number of possible permutations of tasks is *m!,* leading to a time complexity of O(*m!n*).

3.2 Partial-order Preferences

A natural choice is to relax the objective as follows: find the sequence that best satisfies pair-wise partial-order preferences among the tasks. In other words, we want *the sequence that maximizes the sum of the partial-order preference scores for all the pairs in the sequence*. By "partial order preference" we mean that task *i* should be carried out before task *j* but the two tasks do not need to be adjacent in the sequence. We can empirically estimate the expected benefit, or the "preference score", for each task pair by calculating the utility scores for a large number of randomized orders of M tasks conditioned on "*i* before *j*" and "*i* after *j*", respectively. We define the preference score of pair (i, j) as the difference between the mean utility under the first condition and the mean utility under the second condition. Since the number of task pairs is m^2, the partial-order utility estimation is computationally tractable for a moderate *m*.

However, no efficient algorithm is known for deriving the optimal order based on the partial-order preference scores. In fact, this problem is exactly equivalent to the Linear Ordering Problem (LOP) [10], which is known to be NP-hard. Nevertheless, this problem formulation opens an opportunity for us to solve the problem heuristically with link structure analysis over the task dependence network.

3.3 A Novel Application of Link Structure Analysis

The pair-wise partial-order preferences over tasks can be represented using a fully connected directed graph that we call the *partial-order-preference network*, or POPNET for short. The nodes represent tasks. A directed edge from *node i* to *node j* represents the preference (in terms of difference in utility scores) for ordering *task j* before *task i*. If it is more preferable to order *task j* before *task i*, then the directed edge from *node i* to *node j* will have a positive weight, and the edge from *node j* to *node i* will have a corresponding negative weight of equal magnitude. We do this for every pair of nodes to derive a fully connected graph, which can be interpreted as a dependence network – a positive edge from *node i* to *node j* means that *task i* depends on *task j*, in the sense that we expect higher utility by ordering *task j* before *task i*. Intuitively, this means that our goal is to place those tasks earlier in the global order, that many other tasks depend on.

This intuition enables us to leverage standard link analysis algorithms for heuristically ordering the tasks. Recall that in conventional link structure methods like HITS, a good "authority" is defined as a node with in-links from many good hubs, and a good "hub" is defined as a node with out-links pointing to many good authorities [7]. Applying this concept to our problem, a good authority corresponds to a task that meets the partial-order preferences simultaneously for multiple task pairs if it is carried out earlier. Intuitively, the authority scores computed from POPNET are good heuristics for task ordering. Moreover, the authority scores also imply the transitivity of partial-order preferences, i.e., if task A is more preferable than task B for earlier placement, and if task B is more preferable than task C, then task A is more preferable than task C.

Similarly, we can also apply the PageRank algorithm [1] to the POPNET matrix, so that "popular" nodes (i.e., nodes with high PageRank) will correspond to those nodes that many other nodes in the graph link to, and hence correspond to tasks that many other tasks depend on. On the other hand, nodes that are not dependencies of other nodes will receive a low PageRank and end up towards the later part of the total order. This makes intuitive sense since these nodes correspond to tasks that are less crucial towards successfully solving other tasks.

Note that due to the special semantics of the ordering problem, the adjacency matrix (say, P) of the POPNET graph is skew symmetric i.e. $P^T = -P$. This special structure is different from the structure of Web graph, where link analysis algorithms are conventionally applied [1][7]. It is easy to show that for a skew symmetric adjacency matrix, the hub score and authority score of each node are identical (since $PP^T = P^T P = -P \times P$).

To summarize, our solution for task ordering consists of two steps: first, use a training set to estimate the pair-wise partial-order preferences (Section 3), and second, apply link analysis to POPNET for task ordering. In the second step, we can use either HITS or PageRank as two alternative algorithms. We name these variants of our approach as POP.HITS and POP.PAGERANK, respectively.

The time complexity of the first step is $O(m^2 k)$ where *k* is the number of randomized sequences of size m for each task pair in the calculation of partial-order preferences. The time complexity of the second step is $O(m^2)$ with a conventional Subspace Iteration algorithm to compute the first eigenvectors in the HITS or PageRank method.

4. DATASETS

We collected a set of 33,000 business proposals ("documents") from Accenture, a large consulting organization. Each of the documents is associated with a list of 18 metadata fields, as shown in Table 1.

The number of tasks for our dataset is 18, which is relatively small compared to the number of tasks that might arise in some other potential applications,

Table 1 Meta data fields of documents

Keywords,
ItemType,
IndustryKeywords,
TechnologyKeywords,
VendorProductKeywords,
BusinessFunctionKeywords,
TopicTags,
Creator,
Submitter,
Modifier,
Alliances,
Offerings,
Client,
PertinentToOrgUnit,
PertinentToServiceLine,
PertinentToDomainSpecialty,
PertinentToWorkgroup,
PertinentToCountry

e.g. diagnostic tests in the medical domain, or potentially hundreds of tests for troubleshooting complex aircraft failures. Solving the NP-hard LOP problem has an exponential time complexity, which may not be tractable for the large number of tasks; heuristic approaches using HITS or PageRank provide alternative solutions for scalability. Nevertheless, the relatively small number of tasks in the Accenture dataset enables us to compute the LOP optimal solution, and to compare its performance with that of using HITS, PageRank and other baseline methods in a controlled setting. Evaluations using datasets with larger numbers of tasks requires substantial development of those benchmark datasets, which is beyond the scope of this paper.

5. EXPERIMENTS

We conducted a comparative evaluation of the two approaches proposed in this paper – namely POP.HITS, and POP.PAGERANK. We further compare these approaches against three baselines – (i) RANDOM: The average performance of using arbitrary orders, (ii) EXPERT: A task order proposed by a team of domain experts, and (iii) LOP: An exact solution to the LOP problem. We use an off-the-shelf state of the art exact solver as described in [3], that uses a branch and bound search with several heuristics to speed up the search for the best order. Nevertheless, while matrices up to 20x20 take less than 5 seconds to solve, a 35x35 matrix takes more than 10 minutes, due to a time complexity that depends on the factorial of the size of the matrix. The output of each of these methods is an order in which the task of filling each field should be performed, aiming to maximize the utility of all systems in the process and hence minimizing human effort. Therefore, comparing these methods amounts to comparing the overall accuracy of the predictions made by the classifiers used in the corresponding filling order.

5.1 Experimental Setting

We used 33,000 structured documents which is a subset of the Accenture data (cf. Section 4), and divided them into three non-overlapping parts as described in Section 3 – *task-training set* (12,000 documents), *order-training set* (3,000 documents), and *test set* (18,000 documents).

Using the *task-training set*, we trained an SVM classifier for each of the meta-data fields – i.e., one field at a time was chosen as the target for prediction, and the rest of the fields were used as the source or predictor variables. We pick SVM classifiers for each task, which have been reported as competitive performers on a variety of domains [2]. Next, the fields of the documents in the *order-training set* were randomly permuted and the classifiers trained in the previous step were used to make predictions on the fields in these random orders. Note that we can use multiple permutations of fields for each document to derive a better estimate of the pair-wise preferences, limited only by computational resources. In this paper, we limit ourselves to one permutation per document. The performance statistics of the classifiers were then used to populate the POPNET matrix. Finally, the optimal order derived in the previous step was used to determine the order in which predictions would be made for the fields of all documents in the *test set*. The overall performance of the classifiers was then measured on the Test set.

5.2 Performance Measures

We evaluated the performance of the proposed and the baseline methods using the F1 metric, which is the harmonic mean of recall (fraction of relevant items retrieved) and precision (fraction of retrieved items that are relevant) [11]. Higher F1 values signify better performance, and are attained only when both recall and precision are high, thus providing a reliable summary of classification performance.

To measure effectiveness across a wide range of categories that may appear with different frequencies in the dataset, we report micro-averaged F1 as well as macro-averaged F1. Micro-averaging gives equal weight to each individual prediction, and hence, is dominated by categories with a large number of documents. On the other hand, macro-averaging gives equal weight to each category, and hence, may be useful in highlighting the performance on rare categories [8].

5.3 Results

Table 2 shows the performance obtained using different task ordering approaches. We run each method (except EXPERT) ten times to estimate the mean and the standard error of the performance in terms of micro-averaged and macro-averaged F1. For each pair of methods, we conducted a *sign test* (with n = total number of binary decisions on which the two methods in the pair differed, for each field of each document in the test set). The differences among the performance of EXPERT, LOP, POP.HITS, and POP.PAGERANK are statistically insignificant, whereas the improvement of any of these methods over the RANDOM baseline is highly statistically significant (*p-value* << 1%).

POP.HITS beats RANDOM by a substantial margin – 27% in terms of macro F1 and 13% in terms of micro F1. Hence, our approach provides an effective way of choosing an optimal order in the absence of any domain knowledge as well as understanding of the prediction systems, in which case, the user may be forced to choose an arbitrary (random) task order. Given that the space of possible permutations grows at a factorial rate with respect to the number of tasks, it is unlikely that such an arbitrary order would perform well on average.

POP.HITS provides competitive performance against the EXPERT order, with the difference being statistically insignificant. Note that the EXPERT order requires domain knowledge and sufficient understanding of the sequence of tasks and their dependencies. Such expertise might not be readily available. Our proposed approach automatically infers such task dependencies and suggests a task order that performs close enough to the order that would be proposed by a panel of experts.

The differences in the performance of the link analysis based approaches (POP.HITS and POP.PAGERANK) and the exact LOP solution are also statistically insignificant. While LOP is known to be an NP-hard problem, making it not scalable to applications with a large number of tasks, the proposed approaches provide a computationally efficient alternative through the use of HITS and PageRank, which have been successfully applied to Web-scale problems.

An interesting observation is that – on the one hand, the domain experts have sufficient knowledge of the tasks, but they may not understand the behavior and dynamics of various prediction systems – i.e., how the systems combine various features to make a sequence of predictions. This might lead to suboptimal task orders that do not take into account the complex interactions of system predictions. On the other hand, our proposed approach lacks any domain knowledge whatsoever, but it can observe the empirical behavior of systems (i.e., any choice of retrieval or classification systems) on the given set of tasks, to suggest a near-optimal order. How to combine the knowledge of domain experts and the statistical prowess of the proposed method is an interesting topic for future exploration.

Table 2 Comparison of task ordering approaches (mean ± s.e.)

Method	Micro averaged F1	Macro averaged F1
RANDOM	0.5261 ± 0.0002	0.3294 ± 0.0015
EXPERT	0.6005 ± 0.0000	0.4216 ± 0.0000
LOP	0.5991 ± 0.0009	0.4175 ± 0.0007
POP. HITS	0.5943 ± 0.0000	0.4198 ± 0.0000
POP. PAGERANK	0.5927 ± 0.0050	0.4140 ± 0.0015

6. DISCUSSIONS

The importance of research on *IR in context* has been realized in recent years [5]. Users rarely perform IR tasks in isolation; instead, studies show that an overall information-seeking task often involves a multitasking search session comprising of a sequence of related searches, affected by various contextual factors like time, place, and user feedback [9]. However, thorough investigations in leveraging context are relatively sparse so far. Yang et al. investigated how to improve information distillation tasks (retrieval, adaptive filtering and novelty detection) by leveraging the history of user interactions as the context [12]. In this paper, we approach the problem of *IR in context* from a different angle, i.e., how to improve the total utility of multiple systems in a multi-step process with user interactions, specifically, by offering a novel solution to the problem of optimally ordering a set of interrelated prediction tasks.

The partial-order preferences part of our formulation for the task ordering problem relates our work to Cohen et al. [4], who used a similar pair-wise preference formulation for the problem of ranking Web pages when user feedback is available in the form of pair-wise preferences. However, their main focus was on finding a good linear combination of preference functions. The problem of finding the optimal total order was addressed using a simple greedy algorithm, which provides an approximate solution within a factor of two of the optimal order. It is noteworthy that our solution for the problem goes beyond other approximation algorithms for LOP. That is, the link-analysis based approach for heuristically optimizing the task order enables us to leverage the transitivity of partial-order preferences, which is not a property of other algorithmic approximations for LOP. By modelling such transitive relations explicitly, we hope to make our method more robust against noisy or insufficient data by estimating all pair-wise preferences more reliably through the use of the transitivity. It is possible to combine the strength of our current method with those of other approximation algorithms for LOP, which is an interesting topic for future research.

In our proposed approach, the pair-wise task order preferences are empirically estimated by directly observing the performance of the classifiers with respect to different task orders. This has the benefit of allowing direct optimization of arbitrary performance metrics, for instance, domain-specific utility metrics that assign different costs to each prediction task. It would be interesting to experiment with different choices of the metric used to populate the partial order matrix and the metric used to evaluate the system, and assessing the effect of matching vs. mismatching the two choices.

7. CONCLUSIONS

This paper examines the task ordering problem for prediction systems in a multi-step process. We propose a formulation of the problem in terms of pair-wise preferences of task orders that are learned in a supervised fashion and represented using directed graphs. Such a formulation naturally lends itself to the application of link analysis approaches like HITS and PageRank, which provide reasonable heuristics for optimizing the overall utility of a sequence of prediction tasks, and more importantly, enable efficient computation of optimal sequence for applications with a large number of tasks. Experiments on a real collection of structured documents provide promising empirical evidence for the effectiveness of the proposed methods: the performance in terms of macro and micro F1 of the classifiers improved by 27% and 13%, respectively, over the performance of random ordering, and was statistically indistinguishable from the performance obtained when using an expert-suggested ordering of the tasks.

REFERENCES

[1] Brin, S. and Page, L. The anatomy of a large-scale hypertextual Web search engine. Proceedings of the 7th World-Wide Web Conference, (1998)

[2] Caruana, R. and Niculescu-Mizil, A. An empirical comparison of supervised learning algorithms. Proceedings of the 23rd international conference on Machine learning (2006)

[3] Charon, I. and Hudry, O. A branch-and-bound algorithm to solve the linear ordering problem for weighted tournaments. Discrete Applied Mathematics (2006)

[4] Cohen, W.W. and Schapire, R.E. and Singer, Y. Learning to order things. Journal of Artificial Intelligence Research (1999)

[5] Ingwersen, P. and Jarvelin, K. Information retrieval in context: IRiX. ACM SIGIR Forum (2005)

[6] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. Proceedings of ECML-98, 10th European Conference on Machine Learning (1998)

[7] Kleinberg, J. Authoritative sources in a hyperlinked environment. ACM-SIAM Symposium on Discrete Algorithms, (1998)

[8] Lewis, L. Evaluating Text Categorization, Proceedings of Speech and Natural Language Processing Workshop (1991)

[9] Ozmutlu, S. and Ozmutlu, H.C. and Spink, A. Multitasking Web searching and implications for design. Proceedings of the American Society for Information Science and Technology (2003)

[10] Reinelt G. The Linear Ordering Problem: Algorithms and Applications. Research and Exposition in Mathematics, (1985)

[11] Rijsbergen, C.J. Information Retrieval. Butterworths (1979)

[12] Yang, Y. and Lad, A. and Lao, N. and Harpale, A. and Kisiel, B. and Rogati, M. Utility-based information distillation over temporally sequenced documents. Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (2007)