Proactive-DIEL in Evolving Referral Networks

Ashiqur R. KhudaBukhsh[1], Jaime G. Carbonell[1], and Peter J. Jansen[1]

Carnegie Mellon University
{akhudabu, jgc, pjj}@cs.cmu.edu

Abstract Distributed learning in expert referral networks is a new Active Learning paradigm where experts—humans or automated agents—solve problems if they can or refer said problems to others with more appropriate expertise. Recent work augmented the basic learning-to-refer method with proactive skill posting, where experts may report their top skills to their colleagues, and proposed a modified algorithm, proactive-DIEL (Distributed Interval Estimation Learning), that takes advantage of such one-time posting instead of using an uninformed prior. This work extends the method in three main directions: 1) Proactive-DIEL is shown to work on a referral network of automated agents, namely SAT solvers, 2) Proactive-DIEL's reward mechanism is extended to another referral-learning algorithm, ϵ-Greedy, with some appropriate modifications. 3) The method is shown robust with respect to evolving networks where experts join or drop off, requiring the learning method to recover referral expertise. In all cases the proposed method exhibits superiority to the state of the art.

Keywords: active learning; evolving referral network; proactive skill posting

1 Introduction

Learning-to-refer in expert referral networks is a recently proposed active learning setting where an expert can refer problem instances to appropriate colleagues if she finds the task at hand difficult to solve [1]. Such a network draws inspiration from the real world examples of expert networks among physicians or within consultancy firms. The key problem is learning to direct the referrals based on the subject matter of the problem and on estimated expertise of colleagues. The state-of-the-art referral learning algorithm, DIEL (Distributed Interval Estimation Learning), has been further improved in an augmented learning setting where experts can advertise their top skills to colleagues upon joining the network [2]. The modified algorithm, dubbed proactive-DIEL, demonstrated superior performance even in the presence of noise in skill self-estimates and showed empirical evidence of being near-Bayesian-Nash Incentive Compatible, i.e., misreporting skills to receive more referrals provided little or no benefit.

Results presented in [2] were limited to synthetic data with well-behaved (e.g. Gaussian) distributions. In this work, we use a suite of 100 real SAT solvers as experts and SAT problem distributions as topics, and show that proactive-DIEL's superiority over DIEL also holds when the problem distributions and the behavior of the experts is not predictable in the aggregate, i.e., it is not following a known parameterizable distribution. Then we show that the reward-penalty mechanism and initialization proposed in proactive-DIEL translates well to another referral algorithm with some suitable modifications: proactive-ϵ-Greedy obtained superior performance than its corresponding non-proactive version. Finally, we relax a major assumption stated in both [1, 2] by allowing dynamic addition and drop-off of experts, and demonstrate proactive-DIEL's strong resilience to such changes in network nodes and topologies even when substantial (e.g. multiple 20% changes).

After discussing related work, we review the DIEL model, describe skill posting and outline the distributed referral learning algorithms. Next, we describe our experimental set-up, and present our results and observations. We end with some general conclusions and some future research ideas.

2 Related Work

In terms of the learning setting and referral learning algorithm, our primary basis for this work was [1] which first proposed a novel learning setting in a context of Active Learning where experts are connected through a network and can refer instances to one another. The proposed algorithm (DIEL) built upon research on interval estimation learning, a reinforcement learning technique which strives to strike a balance between exploration and exploitation, first proposed in [3, 4], and successfully applied to estimate the accuracy of multiple labelers in [5]. The referral framework drew inspiration from earlier work in referral chaining, first proposed in [6] and subsequently extended, for example, in[7–10]. In the current work, we have adopted the setting, along with some of the algorithms and assumptions for comparison.

This learning setting was further extended in [2], by allowing advertisement to colleagues (partially available priors), and the original DIEL algorithm modified, both for improved performance and to encourage incentive compatibility. There is a large body of literature where truthful mechanism design is the principal focus [11–14]. We have implemented and extended the same mechanism used in proactive-DIEL in our present work, without however proving incentive compatibility for our newly introduced algorithms, or taking any further steps to establish strategyproofness [15].

Finally, we used SATenstein [16], a highly parameterized Stochastic Local Search (SLS) SAT solver for experimental validation of proactive-DIEL on a real task of SAT solving. With a rich design space of 2.01×10^{14} candidate solvers drawing inspiration from most of the prominent SLS SAT solvers proposed in the literature, high-performance SATenstein solvers for specific SAT distributions can be obtained by using an automatic algorithm configurator. In our experiments, we used 100 such solvers obtained from the experiments in [17].

3 Referral Network

3.1 Preliminaries

Referral network: Represented by a graph (V, E) of size k in which each vertex v_i corresponds to an expert e_i $(1 \leq k)$ and each bidirectional edge $\langle v_i, v_j \rangle$ indicates a *referral link* which implies e_i and e_j can refer problem instances to each other.
Subnetwork: The *subnetwork* of an expert e_i is the set of experts linked to e_i by a referral link.
Scenario: Set of m instances $(q_1, \ldots, q_m)$ belonging to n topics $(t_1, \ldots, t_n)$ that are to be addressed by the k experts $(e_1, \ldots, e_k)$.
Expertise: Expertise of an expert/question pair $\langle e_i, q_j \rangle$ is the probability with with e_i can solve q_j.
Referral mechanism: For a query budget Q = 2 (fixed across all our experiments), consists of the following steps.

1. A user issues q_j (*initial query*) to a randomly chosen expert e_i (*initial expert*)
2. Initial expert e_i examines the instance and solves it if possible. This depends on the *expertise* of e_i wrt. q_j.
3. If not, a *referral query* is issued by e_i to a *referred expert*, e_j, within her subnetwork. *Learning-to-refer* involves improving the estimate of who is most likely to solve the problem.
4. If the referred expert succeeds, she communicates the solution to the initial expert, who in turn, communicates it to the user.

Advertising unit: a tuple $\langle e_i, e_j, t_k, \mu_{t_k} \rangle$, where e_i is the *target expert*, e_j is the *advertising expert*, t_k is the topic and μ_{t_k} is e_j's (advertised) topical expertise.
Advertising budget: the number of advertising units available to an expert, following [2], set to twice the size of that expert's subnetwork. Effectively means that each expert reports her top two skills to everyone in her subnetwork.
Advertising protocol: a one-time advertisement that happens right at the beginning of the simulation or when an expert joins the network. The advertising expert e_j reports to each target expert e_i in her subnetwork the two tuples $\langle e_i, e_j, t_{best}, \mu_{t_{best}} \rangle$ and $\langle e_i, e_j, t_{secondBest}, \mu_{t_{secondBest}} \rangle$, i.e., the top two topics in terms of the advertising expert's topic means.

Further details regarding the assumptions involving expertise, network parameters, proactive skill posting mechanism and simulation details can be found in [1, 2].

3.2 Referral Algorithms

From the point of view of a single expert, the decision to refer a problem is essentially an action selection problem where an action corresponds to picking one of a possible set of connected experts. Here, we give a short description of the action selection procedure of the referral algorithms we considered and cite relevant literature for further details.

DIEL: DIEL uses Interval Estimation Learning for action selection. The action a is chosen for which the upper-confidence interval $UI(a)$ is largest, where $UI(a) = m(a) + \frac{s(a)}{\sqrt{n}}$
$m(a)$ is the mean observed reward, $s(a)$ is the standard deviation of the observed rewards and n is the number of observations so far. The intuition behind DIEL is to combine exploitation (via high mean) and exploration (via high variance) as needed. Following the earlier work [1, 2], we initialize the mean reward, standard deviation and number of observations for all actions to 0.5, 0.7071 and 2 respectively for a smooth start (this is equivalent to an initialization with the two rewards of 0 and 1).

proactive-DIEL: proactive-DIEL differs from DIEL in two key ways [2]. First, the mean reward and standard deviation of the rewards are initialized differently. Standard deviation is initialized to 0 (to put more emphasis on the advertised priors). In presence of an advertisement unit, $\langle e_i, e_j, t_k, \mu_{t_k} \rangle$, the mean reward $reward_{mean}(e_i, t_k, e_k)$ is initialized to μ_{t_k}. In absence of such advertisement unit, (recall that the budget was assumed to suffice for advertising an expert's top two skills only) proactive-DIEL initializes the rewards as if the expert's skill was the same as on her second best topic, that is, with $\mu_{t_{secondBest}}$, effectively being an upper bound on the actual value. Second, in addition to binary rewards indicating success and failure, a failed task receives a probabilistic penalty to discourage willful misreporting.

ϵ-Greedy: Unlike DIEL, ϵ-Greedy's action selection choice is guided purely by the mean reward (it greedily picks the highest one) [18]. As a diversification step, with a small probability ϵ, a random action (in this case, selecting a connected expert at random) is chosen. There are several ways to choose an effective value for ϵ. In this work, we set ϵ to $\frac{\alpha * K}{N}$ (where K is the subnetwork size and N is the number of total observations). The value of the hyper-parameter α is set by a parameter sweep on a training set created with the same distributional parameters as the test set (for parameter description, see [1]).

4 Experimental Setup

The data set presented in [1] consisted of 1000 scenarios, each with 100 experts, 10 topics and a referral network. We use a random subset of 200 such scenarios in all our experiments. Our performance measure is the overall task accuracy of our multi-expert system.

The 100 `SATenstein` solvers we used are obtained by configuring `SATenstein2.0` version (described in [17]) on six well-known SAT distributions. Each of these solvers is configured on one of the six SAT distributions. We used the test sets of the SAT distributions as our pool of tasks. Detailed descriptions of the SAT distributions can be found in [16]. We carried out our experiments involving SAT solvers on a cluster of dual-core 2.4 GHz machines with 3 MB cache and 32 GB RAM running Linux 2.6.

5 Results

Before delving into the details, we first give a short description of our **key findings**:

- On an application where SAT solvers are experts, SAT distributions are topics and the task is to solve a SAT instance, our results show that here too proactive-DIEL beats DIEL in the early phase of learning.
- Proactive-ϵ-Greedy, the distributed referral learning version of ϵ-Greedy we proposed that uses proactive skill posting, beat the original version under the condition of truthful reporting of skills.
- While DIEL was well able to cope with small changes to the network, proactive-DIEL proved substantially more robust in the face of large and repeated changes, while consistently maintaining higher performance.

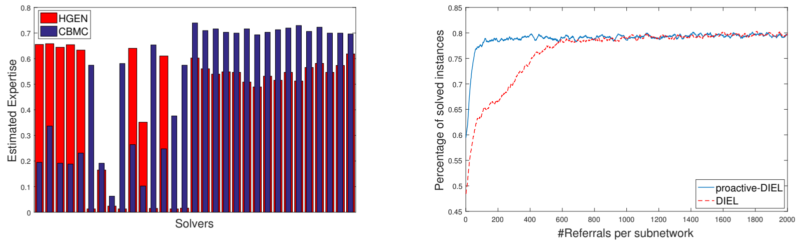
5.1 SATenstein SLS solvers as experts

All performance comparisons between DIEL and proactive-DIEL so far, have been on synthetic data [2]. We were curious to evaluate how that translates to a real task where, for instance, the noise on self-estimates does not follow a known parameterizable distribution. For this, we constructed a referral network of SAT solvers. Besides the importance of SAT solvers in industry and academia, what made this domain an attractive choice for us was our access to a large number of solvers with varying expertise on six well-known SAT distributions via SATenstein [16], and the fact that solutions can be verified trivially.

For our experiments, the budget C for solving each instance is set to 1 CPU second. A solver earns a reward of 1 if it finds a satisfying solution within C seconds, and 0 otherwise (on top of this, proactive-DIEL computes additional penalties depending on advertised skills). Expertise estimates for the SAT solvers are computed based on the number of correct solutions on a set of background data on the SAT distributions under consideration. Figure 1(a) gives an inkling how expertise levels between solvers may differ on two tasks. Figure 1(b) presents a comparison of the performance of proactive-DIEL with that of DIEL when applied to a referral network of SAT solvers (we considered 10 different referral networks randomly selected from our data set), with skill postings determined by these expertise self-estimates. We see that, while the eventual performance levels are very similar, proactive-DIEL ramps up significantly faster than its predecessor, in spite of the noise and uncertainty on the expertise self-estimates.

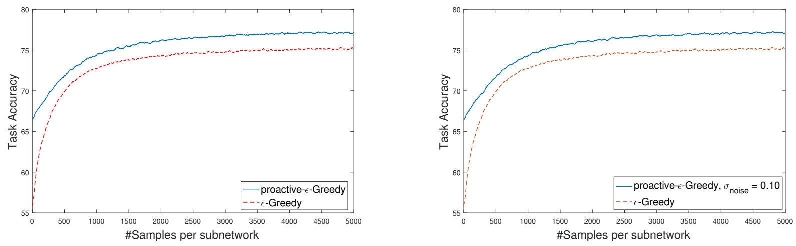
5.2 Proactive-ϵ-Greedy

We extended the ϵ-Greedy algorithm with the reward-penalty mechanism used in proactive-DIEL in a straightforward way, with the exception that we do not impose a penalty during a diversification step even when the referred expert fails. Figure 2 shows that, with a similar initialization, the new algorithm (proactive-ϵ-Greedy) performs substantially better than the old. Figure 2(b) shows that this performance improvement holds even in the presence of noise in the self-estimates.



(a) Skill estimates (b) Performance comparison

Figure 1. Expertise estimates of a subset of solvers on background data of two SAT distributions and performance comparison with `SATenstein` solvers as experts.



(a) Accurate skill estimation (b) Noisy skill estimation

Figure 2. Performance comparison of proactive-ϵ-`Greedy` and ϵ-`Greedy`.

Following [2], we considered $\hat{\mu} = \mu + \mathcal{N}(0, \sigma_{noise})$, where $\hat{\mu}$ is an expert's own estimate of her true topic-mean μ, and σ_{noise} is a small constant.

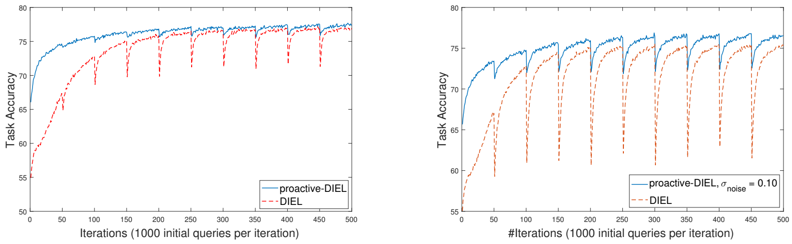
5.3 Evolving networks

In practice, networks are not static – new experts join in and old experts leave – and robustness to such network changes is crucial for a referral algorithm's performance on real-world data. In these results, we explore a steady rate of network changes occurring at regular intervals (every 50 iterations, an iteration being 1000 initial queries). It is clear from Figure 3 that proactive-`DIEL` is much more resilient to these network changes than `DIEL` (smaller dip, faster recovery). The acid test for proactive-`DIEL` is shown in Figure 3(b), where its recovery in the face of a repeated 20% network change, with noisy self-estimates, led to only minimal degradation [1] .

6 Concluding Remarks

In this work, we extended the referral-learning method proposed in [2] in three directions. First, we obtained experimental validation of proactive-`DIEL`'s su-

[1] Note that the choice to use 50-iteration bursts is purely for visualization reasons and our results do not change qualitatively when we consider similar changes distributed across the entire course of the simulation. We also ran experiments with a large one-time network change from which both `DIEL` and proactive-`DIEL` recovered well.



(a) 5% network change

(b) 20% network change and noisy self-estimates

Figure 3. proactive-DIEL on dynamic networks.

perior performance over DIEL in learning to refer among actual agents (SAT solvers) without distributional assumptions. Next, we extended the initialization and reward-penalty technique proposed in proactive-DIEL to other methods and show an improved performance resilient to noisy self-estimates. Finally, we showed proactive-DIEL's robustness to evolving networks under severe network changes, an excellent property of proactive-DIEL hereto unexplored.

Possible future extensions to our work include:

– **Strategyproofness**: While willful skill misreporting was shown to be of little or no benefit when all other experts report truthfully in proactive-DIEL [2], we are yet to empirically evaluate the same for ϵ-Greedy. Also, a stronger degree of incentive compatibility, strategyproofness, would require truthfulness for optimal performance in all cases.
– **Expertise drift**: Whereas we explored experts joining and dropping off the referral network, the expertise of individual experts did not change with time. But in practice expertise may improve with practice or degrade due to fatigue or other factors. Devising algorithms to deal with time-varying expertise would be a meaningful research challenge.
– **Continuous rewards**: With the experiments with SAT solvers, it is very easy to conceive continuous rewards. If we treat (cutoff - run time) as reward, we would get 0 reward on a failure and higher rewards will imply faster solutions. Exploring reward mechanisms to handle continuous rewards could further improve network performance as an effective referral will maximize not only solution likelihood but also solution quality.

Acknowledgements: This research is partially funded by the National Science Foundation grant EAGER-1649225.

Bibliography

[1] KhudaBukhsh, A.R., Jansen, P.J., Carbonell, J.G.: Distributed learning in expert referral networks. In: European Conference on Artificial Intelligence (ECAI), 2016. (2016) 1620–1621

[2] KhudaBukhsh, A.R., Carbonell, J.G., Jansen, P.J.: Proactive skill posting in referral networks. In: Australasian Joint Conference on Artificial Intelligence, Springer (2016) 585–596

[3] Kaelbling, L.P.: Learning in embedded systems. MIT press (1993)

[4] Kaelbling, L.P., Littman, M.L., Moore, A.P.: Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research **4** (1996) 237–285

[5] Donmez, P., Carbonell, J.G., Schneider, J.: Efficiently learning the accuracy of labeling sources for selective sampling. Proc. of KDD 2009 (2009) 259

[6] Kautz, H., Selman, B., Milewski, A.: Agent amplified communication. (1996) 3–9

[7] Yolum, P., Singh, M.P.: Dynamic communities in referral networks. Web Intelligence and Agent Systems **1**(2) (2003) 105–116

[8] Yu, B.: Emergence and evolution of agent-based referral networks. PhD thesis, North Carolina State University (2002)

[9] Yu, B., Venkatraman, M., Singh, M.P.: An adaptive social network for information access: Theoretical and experimental results. Applied Artificial Intelligence **17** (2003) 21–38

[10] Yu, B., Singh, M.P.: Searching social networks. Proc. of AAMAS '03 (2003)

[11] Babaioff, M., Sharma, Y., Slivkins, A.: Characterizing truthful multi-armed bandit mechanisms. In: Proceedings of the 10th ACM conference on Electronic commerce, ACM (2009) 79–88

[12] Biswas, A., Jain, S., Mandal, D., Narahari, Y.: A truthful budget feasible multi-armed bandit mechanism for crowdsourcing time critical tasks. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems (2015) 1101–1109

[13] Tran-Thanh, L., Stein, S., Rogers, A., Jennings, N.R.: Efficient crowdsourcing of unknown experts using multi-armed bandits. In: European Conference on Artificial Intelligence. (2012) 768–773

[14] Tran-Thanh, L., Chapman, A., Rogers, A., Jennings, N.R.: Knapsack based optimal policies for budget-limited multi-armed bandits. arXiv preprint arXiv:1204.1909 (2012)

[15] Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.V.: Algorithmic game theory. Volume 1. Cambridge University Press Cambridge (2007)

[16] KhudaBukhsh, A.R., Xu, L., Hoos, H.H., Leyton-Brown, K.: Satenstein: Automatically building local search sat solvers from components. In: IJCAI. Volume 9. (2009) 517–524

[17] KhudaBukhsh, A.R., Xu, L., Hoos, H.H., Leyton-Brown, K.: Satenstein: Automatically building local search sat solvers from components. Artificial Intelligence **232** (2016) 20–42

[18] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning **47**(2-3) (2002) 235–256