Reinforcement Learning for Multi-Step Expert Advice

Patrick Philipp
Institute AIFB
Karlsruhe Institute of Technology
patrick.philipp@kit.edu

Achim Rettinger
Institute AIFB
Karlsruhe Institute of Technology
rettinger@kit.edu

ABSTRACT

Complex tasks for heterogeneous data sources, such as finding and linking named entities in text documents or detecting objects in images, often require multiple steps to be solved in a processing pipeline. In most of the cases, there exist numerous, exchangeable software components for a single step, each an "expert" for data with certain characteristics. Deciding which expert to apply to which observed data instance given potential prior and future steps becomes a challenge that is even hard for humans to decide. We therefore define so-called "meta-dependencies" to assess expert performances, and use them for decision-making via Online Model-Free- and Batch Reinforcement Learning (RL) approaches, building on techniques from Contextual Bandits (CBs) and Statistical Relational Learning (SRL). The resulting system automatically learns to pick the best pipeline of experts for a given set of data points. We evaluate our approach for Entity Linking on text corpora with heterogeneous characteristics (such as news articles or tweets). Our empirical results improve the estimation of expert accuracies as well as the out-of-the-box performance of the original experts without manual tuning.

Keywords

Decision-Making with Multi-Step Expert Advice, Expert Processes, Reinforcement Learning, Entity Linking

1. INTRODUCTION

Complex tasks for heterogeneous data sources often require multiple steps to be solved in a pipeline. Entity linking, for example, is a two-step problem where text is tokenized to (i) find named entities and (ii) linked them to a knowledge base. In most of the cases, there exist numerous, exchangeable software components for a single step, each an "expert" for data with certain characteristics. We call such settings *Decision Making with Multi-Step Expert Advice.*

Combining the outputs of exchangeable experts, e.g. by majority vote, can improve the overall performance, but might fail if experts correlate or display high degrees of variance in their performances. Moreover, two experts solving two subsequent steps might be syntactically compatible but produce bad results when plugged together. Learning weights for experts is, thus, crucial to query promising experts and keep "good" candidates throughout multiple steps.

Well-established decision-theoretic frameworks partially deal with these problems, including Decision-Making with Expert Advice (EA) [6] where exchangeable experts are sequentially weighted for single-step problems, Budgeted EA [1] where additional budgets are set on expert queries, Contextual Bandits (CBs) [3] where all experts might be queried but only the subset with the chosen action get feedback and, finally, Contextual Markov Decision Processes (CMDPs) [8] which consider a multi-step CB problem setting, where one outcome is rewarded per step.

Still, none of the frameworks cover our setting at once, leading us to define so-called *Expert Processes* (EPs). EPs enable learning fine-grained expert weights by considering the impact of current decisions on future experts and not being overly restrictive in rewarding outcomes.

A key problem for such processes is knowledge transfer, as state spaces might be highly heterogeneous. For Entity Linking, the input space might comprise tweets, news articles or scraped web pages which cause high variance in the performances of experts. Having learned models from tweets, thus, does not directly transfer to news articles. To this end, we introduce so-called meta-dependencies which enable to learn context-specific non-parametric supervised models based on domain-dependent kernels and single- as well as pairwise behaviours of experts.

We first define EPs, before introducing meta-dependencies and our learning approaches. We describe our evaluation, summarize our work and give a short outlook.

2. EXPERT PROCESSES

An expert process EP [10] is a 7-tuple $(S, E, A, R, T, \mathrm{H}, \mathrm{M})$ based on Markov Decision Processes (MDPs) [11] with S the set of all states, E the set of all experts, A the set of all possible actions, $T : S^h \times A^h \rightarrow S^{h+1}$ the deterministic transition function, $R : S^h \times A^h \rightarrow [0,1]^{\mathrm{D}^h}$ the reward function, H the number of steps and M the budget per step. The state set $S = S^1 \cup S^2 \cup \ldots \cup S^{\mathrm{H}+1}$ consists of $\mathrm{H}+1$ disjunct, heterogeneous state spaces. The reward $R(s^h, a^h)$ is a vector of local feedback received after choosing action a^h in state s^h. We are given a set E of N^h experts, where in step h of the EP a subset of $E = E^1 \cup E^2 \cup \ldots \cup E^{\mathrm{H}}$ is available for being queried. An expert e^h is a function $e^h : S^h \rightarrow AR$, where $AR \subseteq S^h \times S^{h+1}$ are action relations for mapping decision candidates. We denote $\Delta_E : S^h \rightarrow S^{h+1}$ the action suggestion of expert e^h for state s^h. The budget M^h confines the

Appears in: *Proceedings of the 1st Workshop on Transfer in Reinforcement Learning (TiRL) at the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), A. Costa, D. Precup, M. Veloso, M. Taylor (chairs), May 8–9, 2017, São Paulo, Brazil.*

number of (state, expert)-pairs to query in step h.

The Q-function estimate is defined as $Q(s^h, a^h) = \sum_{{}^d s \in s^h} \sum_{e \in E^h} w_e({}^d s) * \mathbb{1}_{\{\Delta_e({}^d s) = {}^d a\}}$ with expert weights:

$$w_{e^h}({}^d s^h) = R({}^d s^h, \Delta_{e^h}({}^d s^h)) \\ + \ \delta \sum_{s^{h+1} \in \hat{S}^{h+1}} T(s^{h+1}|s^h, a^h) \max_{a^{h+1} \in A^{h+1}} Q(s^{h+1}, a^{h+1}) \tag{1}$$

where δ defines the impact of future rewards and $\Delta_{e^h}({}^d s^h)) \in a^h$. Finally, the reward of the decision process can be defined as $R^{cum} = \mathbb{E}[\sum_{h=1}^{\mathrm{H}} \sum_{d=1}^{\mathrm{D}^h} R^h({}^d s^h, {}^d a^h)]$.

To learn expert weights for EPs, we now introduce the concept of meta-dependencies.

3. META-DEPENDENCIES

A meta-dependency expresses a relation between a decision candidate ${}^d s^h$ and either (i) two experts e_i^h, e_j^h of the same step, (ii) two experts e_i^h, e_j^{h+1} of subsequent steps or (iii) a single expert e^h. It approximates how single- or pairs of experts behave for a decision candidate based on behavioral measures $\mu : S^h \times A^h \times A^h \rightarrow [0, 1]$ to assess ${}^d s_i^h$ on similar decision candidates ${}^c s_j^h$ in its neighbourhood, using a domain dependent kernel $\kappa \in K$ with $\kappa : S^h \times S^h \rightarrow [0, 1]$.

While (iii) is the analogy to function approximation techniques for the meta-analysis of single experts, (i) and (ii) are classes of pairwise measures which enable an agent to decide which experts cooperate well. The central idea is that adequate domain-dependent kernels (e.g. based on word embeddings) can construct neighbourhoods for a decision candidate which comprise related decision candidates with known rewards weighted by their distance. A meta-dependency might then be able to better reuse known rewards from different data distributions.

As an example for a pairwise intra-step expert meta-dependency (type (i)), the (ϵ, θ)-dependency of a pair $(e_i^h, e_j^h, {}^d s^h)$ with κ, μ is expressed as:

$$\mathrm{MD}_1^{\kappa,\mu}(e_i^h, e_j^h, {}^d s^h) = \\ \frac{\sum_{{}^l s^h \in N_\kappa({}^d s^h)} \kappa({}^d s^h, {}^l s^h) \mu({}^l s^h, \Delta_{e_i^h}({}^l s^h), \Delta_{e_j^h}({}^l s^h))}{\sum_{{}^l s^h \in N_\kappa({}^d s^h)} \kappa({}^d s^h, {}^l s^h)} \tag{2}$$

with $N_\kappa({}^d s_i^h) = \{{}^l s_j^h | \kappa({}^d s_i^h, {}^l s_j^h) \geq \gamma\}$ the decreasingly sorted neighbourhood of ${}^d s^h$ with minimal similarity γ, $\epsilon = \frac{\sum_{{}^l s^h \in N_\kappa({}^d s^h)} \kappa({}^d s^h, {}^l s^h)}{|N_\kappa({}^d s^h)|}, |N_\kappa({}^d s^h)| \geq \theta$ and $\mu({}^d s^h, {}^d a_i^h, {}^d a_j^h) = \frac{R({}^d s^h, {}^d a_i^h) + R({}^d s^h, {}^d a_j^h)}{2}$ the joint precision of two experts.

Based on various domain-dependent kernels and behavioural measures, resulting meta-dependencies have to be weighted and combined to derive an estimate for expert weights. We approach this by two Reinforcement Learning (RL) approaches in online and batch settings.

4. RL FOR EXPERT PROCESSES

The main challenge to solve EPs – in addition to taking into account heterogeneous state spaces – is dealing with step-specific budgets and the resulting partial feedback, making RL key for EPs. We thus explore different relationships among meta-dependencies to express expert weights and, then, learn so-called *meta-weights*.

In our Online RL approach, we exploit existing approaches for partial feedback in one-step decision-making (i.e. CBs and Budgeted EA) and take into account future rewards with inter-step meta-dependencies. The baseline is the Multiplicative Weights / Hedge (MWH) algorithm [2] for the budget-free EA setting (where experts are weighted exponentially to their performance) which we adapt to EPs.

The Batch RL approach assumes the option to collect Z_{batch} samples before updating expert weights. To learn expert weights, we use Probabilistic Soft Logic (PSL) [7, 5] – a modeling language for Hinge-loss Markov Random Fields (HL-MRFs) [4] where a HL-MRF is a conditional probabilistic model over continuous random variables. A model in PSL comprises weighted, first-order logic rules with features defining a Markov network. We argue and show that – among others – modelling meta-dependencies as PSL rules and learning their *meta-weights* is well-suited for collectively learning expert weights.

5. EMPIRICAL EVALUATION

We evaluate our methods based on an EP for named entity recognition (NER) and disambiguation (NED) for news articles and tweets. The EP has H = 2 with S^1 a state space over words, S^2 a state space over named entities and S^3 a state space over disambiguated named entities, i.e. resources available in a structured knowledge base (e.g. DBpedia [9] or YAGO [12]). Our evaluation measures (i.e precision, recall and f1-measure) are based on benchmark GERBIL [13].

The results show that our approaches can determine the accuracy of NER and NED Web services as experts. In addition to accuracies, our approaches improve the performance of the original web services in terms of f1-measure and precision, whereas the highest recall of a web service could not be reached. Finally, learnt weights for tweets were successfully applied to articles, making knowledge transfer possible [10].

6. CONCLUSION

We formalized EPs for Multi-Step Expert Advice by reusing central concepts of EA, Budgeted EA, MDPs and CMDPs. For learning distribution-independent expert weights, we introduced meta-dependencies based on characteristics of data points. We argued that these meta-dependencies can be best exploited by Model-Free Online RL Learning based on MWH and Model-Free Batch RL with PSL as function approximator, both enabling to learn meta-weights which can be used for unseen data. Our approaches were evaluated for an Entity Linking EP, where they were able to provide superior results for estimating accuracies of experts and optimizing the overall outcome.

7. FUTURE WORK

We solve EPs assuming a central agent, but a decentralized Multi-Agent System might enable higher degrees of scalability due to parallelization and higher degrees of generalizability of learned weights due to self-interest of each agent.

Also, while we only evaluate the capability of transferring knowledge to different data distributions, knowledge transfer between different EPs might be a next step. Here, learned meta-weights for meta-dependencies potentially need to be adapted to novel sub-steps. This also entails transferring known inter-step meta-dependencies to new expert pairs.

REFERENCES

[1] K. Amin, S. Kale, G. Tesauro, and D. S. Turaga. Budgeted prediction with expert advice. In *AAAI*, pages 2490–2496, 2015.

[2] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The non-stochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002.

[4] S. H. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss markov random fields: Convex inference for structured prediction. In *UAI*, 2013.

[5] M. Bröcheler, L. Mihalkova, and L. Getoor. Probabilistic similarity logic. In *UAI*, pages 73–82, 2010.

[6] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *J. ACM*, 44(3):427–485, 1997.

[7] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming: Foundations and Applications*, pages 1–4, 2012.

[8] A. Krishnamurthy, A. Agarwal, and J. Langford. PAC reinforcement learning with rich observations. In *NIPS*, pages 1840–1848, 2016.

[9] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[10] P. Philipp and A. Rettinger. Reinforcement learning for multi-step expert advice. In *AAMAS*, 2017.

[11] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[12] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.

[13] R. Usbeck, M. Röder, A. N. Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke, A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis, and L. Wesemann. GERBIL: general entity annotator benchmarking framework. In *WWW*, pages 1133–1143, 2015.