Expertise Trees Resolve Knowledge Limitations in Collective Decision-Making

Axel Abels [1 2] **Tom Lenaerts** [1 2 3] **Vito Trianni** [4] **Ann Nowé** [2]

Abstract

Experts advising decision-makers are likely to display expertise which varies as a function of the problem instance. In practice, this may lead to sub-optimal or discriminatory decisions against minority cases. In this work, we model such changes in depth and breadth of knowledge as a partitioning of the problem space into regions of differing expertise. We provide here new algorithms that explicitly consider and adapt to the relationship between problem instances and experts' knowledge. We first propose and highlight the drawbacks of a naive approach based on nearest neighbor queries. To address these drawbacks we then introduce a novel algorithm — expertise trees — that constructs decision trees enabling the learner to select appropriate models. We provide theoretical insights and empirically validate the improved performance of our novel approach on a range of problems for which existing methods proved to be inadequate.

1. Introduction

The following example from medical diagnostics illustrates the problem settings we examine in this work: an online platform repeatedly presents a group of medical experts with patients demonstrating sets of symptoms. Based on these symptoms, the patient's medical record, and the experts' knowledge and experience, the most appropriate treatment must be selected. This treatment is then applied to the patient, and its effectiveness is evaluated based on some measure of recovery, side effects, and cost, for example. A rational objective in this setting is to maximize the effectiveness of the treatments administered as a function of the advice of the group of medical experts. Ideally, we would hope that medical experts agree on the best treatment based on objective characteristics of each treatment, such as its success rate, side effects, and cost. However, expert opinions are likely to be influenced by both objective factors and subjective views. For example, based on previous experience with comparable patients, a clinician may have views about the efficacy of a treatment for a patient that differs from another clinician's views. Such divergences can result from factors which are clearly outside of an expert's control, but they can also be the result of cognitive biases. For example, medical experts might overstate the efficacy of a treatment as a result of confirmation bias (Trope et al., 1997) or the primacy effect (Bond et al., 2007). Experts thus provide individual advice based on their previous experience with similar problems. When we have access to a diverse set of experts who might disagree, we wish to determine on which advice to act. Particularly, we consider the case wherein multiple iterations (on different problem instances, e.g., different patients with different medical records) of the decision-making process occur, allowing us to adapt to expertise. The challenge of identifying an appropriate way to act on expert advice is heightened by the partial feedback: we only observe the outcome of the decisions we make, e.g., the chosen treatment.

The use of expert advice has been extensively studied to solve complex decision problems in the face of uncertainty, known as *bandits* (Auer et al., 2002a; Foster & Rakhlin, 2020). Prior approaches to leveraging expert advice in this setting are however not conditioned on the characteristics of each problem instance. In contrast, we consider here *localized expertise*, specifically the case wherein at each iteration the quality of experts may change as a function of the problem instance, i.e., an observed set of features. Such varying levels of expertise are normal for human experts who specialize in particular areas of a larger problem domain. Similarly, artificial experts may exhibit performance fluctuations as a consequence of insufficient training, or of incomplete or biased data sets. Specifically, (un)conscious biases may result in judgmental errors caused by an erroneous weighting of sensitive traits such as gender or ethnicity (Pohl, 2017). Such biases can occur both in humans and in artificial experts as a result of biased design or data bias (Gianfrancesco et al., 2018). Methods capable of de-

[1]Machine Learning Group, Université Libre de Bruxelles, Brussels, Belgium [2]AI Lab, Vrije Universiteit Brussel [3]Center for Human-Compatible AI, UC Berkeley, Berkeley, USA [4]Institute of Cognitive Sciences and Technologies, National Research Council, Rome, Italy. Correspondence to: Axel Abels <axel.abels@ulb.be>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

tecting and counteracting such biases are not only essential in terms of performance, but also in terms of fairness. Concretely, we explore the improvements in performance that can be achieved when accounting for changes in expertise as a function of a set of features, for example as a result of specializations or biases. Thus, while there may not be a globally optimal way of acting on expert advice, partitioning the problem space and developing partial models on these partitions may provide an effective piece-wise model. In this paper, we argue that algorithms designed for collective decision-making (CDM) must account for localized expertise to maximize performance. In particular, algorithms for bandits with expert advice should learn policies conditioned on the problem instance.

To address this need, we introduce two novel algorithms, based first on a nearest neighbors approach, and second on a model tree approach. To highlight the difference with model trees we name the latter approach *expertise trees*. The increasing complexity of these approaches is required to tackle the challenges of localized expertise. We then provide some bounds on the performance of algorithms in the localized setting and provide theoretical insights into the losses incurred by inappropriately partitioning the problem space. To conclude, we empirically evaluate these algorithms in terms of performance and execution time and show that the expertise tree approach achieves the best results, at an acceptable increase in algorithmic complexity.

2. Bandits with Localized Expert Advice

Bandits formalize problems wherein a learner repeatedly chooses one out of K arms over a number of rounds with the aim of optimizing the outcomes of its choices when only the outcome of the chosen arm is observed (Thompson, 1933; Auer et al., 2002a). In the contextual setting (Chu et al., 2011; Valko et al., 2013), outcomes are characterized by a function $f : [K] \times \mathbb{R}^d \to [0, 1]$, which maps an arm k and the context $\vec{x}_t$ (i.e., a set of features characterizing the current instance) of the decision at time t to an expected reward. At time t, the reward observed for choosing arm k_t as a function of a context $\vec{x}_t$ is $r_t = f(k_t, \vec{x}_t) + \epsilon$, with ϵ a noise term with mean 0. For example, the treatment (arm) with the best outcome (reward) in terms of some relevant measure (e.g., QALY, see Whitehead & Ali, 2010), varies as a function of a patient's medical records (the context). When the complexity of the problem is such that the cost of learning is unacceptable, a learner can exploit the advice of experts to make decisions, as formalized by the problem of *bandits with expert advice* (Auer et al., 2002a; Foster & Rakhlin, 2020). In particular, when approximating f from scratch is unfeasible, the learner can observe the advice of a set of N experts in the form of N estimates of each arm's expected reward. We denote by $\vec{\xi}^n : \mathbb{R}^d \to \mathbb{R}^K$ expert n's vector function of K estimates, one per arm.

To maximize the rewards from the arms it chooses, a learner must optimize its decisions as a function of these advice vectors. Algorithms for bandits with expert advice therefore learn to act on the advice matrix by maintaining a policy π_t parameterized by the matrix of advice $\boldsymbol{\xi}(\vec{x}_t) := \{\vec{\xi}^1(\vec{x}_t), ..., \vec{\xi}^N(\vec{x}_t)\}$. For the sake of readability, we drop the parameters of ξ when no ambiguity exists, e.g., we will write $\boldsymbol{\xi}_t$ as shorthand for the $N \times K$ matrix of advice induced by the context at time t. Similarly, $\vec{\xi}_{k,t}$ denotes the k-th column of this matrix, i.e., the advice for arm k. In every round t, the policy π_t induces a choice of arm k_{π_t} for which a reward r_{π_t} is observed. The aim of the learner is to update its policy as a function of the observed rewards to maximize the average of collected rewards, $\frac{1}{T}\sum_{t=1}^{T} r_t$. Given an optimal policy π^*, performance can also be stated as the expected gap to this policy, i.e., the regret: $R(T) = \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} r_{\pi^*} - r_{\pi_t}]$, where the expectation is over possible contexts and resulting outcomes.

2.1. Localized Expertise

We generalize the problem by explicitly considering cases wherein the quality of experts' advice changes as a function of the context they are presented with. This is naturally present for human experts who tend to have an area of specialization. When the instance falls outside of an expert's specialization, this should be accounted for. Similarly, artificial experts may be well-fitted to only a subset of the problem space. Biases, whether conscious or not, might also cause inaccuracies induced by some sensitive features, such as gender or ethnicity. Identifying the dependence of expert quality on a set of features has the potential to significantly increase the quality of the collective decisions. We hypothesize that we can obtain an effective model by partitioning the context space and learning partial models on these partitions.

We formalize this dependence by introducing the *expertise context*, which is defined as a subset of the full context observed by the experts $\vec{x}_t$ (see Figure 1). For example, a patient's gender might affect the effectiveness of treatments, but also (unconsciously) affect the quality of expertise. While we might reasonably want to include the full context $\vec{x}_t$ when deciding on which expertise to act, this subset formulation accounts for cases wherein some features cannot be meaningfully captured. For example, textual descriptions of the problem can readily be incorporated by human experts, but the computational cost might be too high for them to be taken into account by an algorithm for bandits with expert advice. We denote the subset of dimension $g \leq d$ by $\vec{z}_t \in \mathbb{R}^g$. In the localized expertise setting, the optimal way of acting on expert advice is thus conditioned on this subset of features. For example, if a

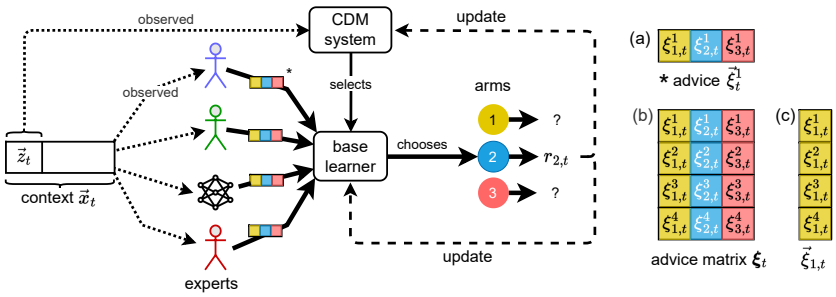

Figure 1: Illustration of a bandit with localized expertise with $N = 4$ experts and $K = 3$ arms. The CDM system observes the expertise context $\vec{z}_t$ and selects an appropriate (localized) base learner. Each expert observes the full context $\vec{x}_t$ at time t and then provides an advice vector to the base learner (e.g., (a) $\vec{\xi}_t^1$ is the first expert's advice). Note that one can have both human and artificial experts as depicted in the figure. The advice matrix (b) $\boldsymbol{\xi}_t$ is a concatenation of these advice vectors, and (c) $\vec{\xi}_{1,t}$ is the advice for arm 1. The base learner takes all expert advice into account to select one of the possible arms. The chosen arm ($k = 2$) provides a reward $r_{2,t}$, which in this work is sampled from a distribution with mean $f(2, \vec{x}_t)$. This reward is then used to update both how the base learner acts on advice, and how the CDM system chooses an appropriate learner.

patient's symptoms relate to the heart, the advice of medical experts with matching specialization should be prioritized. In practical terms, $\vec{z}_t$ encodes a patient's category, and the decision-maker should optimize the use of expert advice as a function of the category encoded in $\vec{z}_t$. In particular we are interested in settings wherein the relation between the expertise context $\vec{z}$ and experts is unknown a priori, and must thus be learned through feedback from decisions.

2.2. Reduction to a Contextual Bandit

Similar to approaches evaluated in the classification setting (e.g., Keswani et al. [2021]), we consider as a baseline a method which selects at each timestep a single expert whose advice we act on. This requires determining the expert whose advice leads to a maximized expected reward given the observable expertise context $\vec{z}_t$. In practice, we are reducing the problem to a contextual multi-armed bandit. Specifically, each arm in the constructed bandit corresponds to an expert in the original problem, and selecting that arm implies following the corresponding expert's advice. The reward we observe for the chosen arm allows us to then update the quality estimate of the expert whose advice we acted on. The quality of each expert (i.e., arm in the constructed bandit) in turn is a function of the expertise context. We can learn to approximate this function while ensuring sufficient exploration by applying an appropriate contextual bandit algorithm to the reduced problem. This approach can be seen as a generalization of a meta multi-armed bandit approach to the contextual case (see Auer et al., 2002a, which uses a standard multi-armed bandit algorithm to determine the best expert by modelling each expert as a an arm in a meta-bandit).

The chosen contextual bandit algorithm impacts the specializations that can be learned. For example, algorithms based on decision trees (e.g., *TreeHeuristic*, see Elmachtoub et al., 2017) are more suitable for discrete expertise regions. An additional benefit of tree-based algorithms is that they perform feature selection, which is essential if the expertise context contains many features of which only a few are predictive of expertise.

One significant drawback of this reduction is that decisions are taken on the basis of a single expert's advice. In addition to limiting knowledge acquisition in each round to a single expert — which typically induces poor performance when the number of experts is high — it is also impossible for the learner to surpass the performance of the best individual in the collective for any given expertise context. In contrast, when a decision is taken as a function of the advice of multiple experts, an opportunity for collective intelligence (i.e., performance better than the single best expert) emerges. With this goal in mind, we propose in the following section (i) a nearest neighbor approach inspired by local learning and highlight its drawbacks, and then (ii) address these drawbacks with our novel expertise tree approach.

3. Methods

Our goal is to learn a mapping from expert advice to arms in such a way that our decisions maximize the cumulative reward. Because expert quality is a function of the expertise context $\vec{z}_t$, our aim is to learn a mapping conditioned on $\vec{z}_t$, i.e., we hope to learn a policy $\pi : [0, 1]^N \times \mathbb{R}^{g \times K} \to \Delta^K$ which maps both the advice and the expertise context to a distribution over the arms. We propose to learn π by partitioning the expertise space into regions, and maintaining for each region an independent instance of a bandits-with-expert-advice learner, such as EXP4 (Auer et al., 2002a) or Meta-CMAB (Abels et al., 2023).

If prior knowledge is available on which features of $\vec{z}_t$ are likely to alter expert performance, this can be used to parti-

tion the space appropriately and then apply an independent algorithm for bandits with expert advice on each subset. If no such knowledge is available, or if the influence of specific features is over or underestimated, the chosen partitions will reflect these inaccuracies. When no knowledge about appropriate partitions is available a priori, a sensible solution is to act as a function of previous similar experiences.

We thus consider an algorithm which adapts the principles of local learning (Bottou & Vapnik, 1992) to the bandits with expert advice setting. When given an expertise context $\vec{z}_t$ at time t, we can sample a percentage p of previous experiences for which the expertise context is closest. These $\lceil \frac{p}{100} \cdot t \rceil$ experiences are then used to initialize a bandits-with-expert-advice learner. We refer to this approach as *Nearest* $p\%$. While this method requires no knowledge of the expected number of partitions, a proper choice of p should be made to avoid over or under-fitting. Alternatively, a distance threshold can be used instead of a threshold on the number of neighbors. In both cases, too large thresholds will include previous experience from different regions, while too small values can induce an under-fit model. Perhaps as significant as the choice of threshold is that the performance of this approach relies on the accuracy of the distance measure as a proxy for model similarity. In particular, if the expertise context contains features which have no bearing on expert quality, their inclusion in the distance computation will impact the quality of the learned model. To cope with such irrelevant features, and to automatically learn appropriate neighborhoods, we propose in the following section a novel approach which simultaneously learns to identify relevant features and partitions through a decision tree.

3.1. Expertise Trees to Capture Localized Knowledge

Model Trees (Wang & Witten, 1996) are decision trees wherein a regression model (typically linear) is fit on the leaves. This kind of decision tree effectively partitions the space into regions for which an individual model is beneficial. We develop here a model tree algorithm adapted to the setting of bandits with localized expert advice. This setting differs from the usual model tree setting in two aspects. First, the bandit setting induces a need for exploration and a different optimization target (reward as opposed to prediction error). Secondly, for traditional model trees, the features on which the space is split are the same features that are used in the leaf models. In contrast with model trees, we split the expertise space (i.e., on $\vec{z}_t$) and learn models on the expert advice (i.e., $\boldsymbol{\xi}_t$). Given this difference, we label these *expertise trees*.

3.1.1. Building Expertise Trees

Let $\mathcal{H}$ be a history of experience tuples consisting of the advice matrix, the chosen action, the collected reward, the probability for that action, and the expertise context, i.e., a $\langle \boldsymbol{\xi}, k, r, p, \vec{z} \rangle$ tuple. Given $\mathcal{H}$, we can partition this set of experiences along the ith dimension of the expertise contexts, resulting in two subsets we will denote as $\mathcal{H}_{<\tau^{(i)}} := \{\langle \boldsymbol{\xi}, k, r, p, \vec{z} \rangle \in \mathcal{H} | z^{(i)} < \tau^{(i)}\}$, and analogously $\mathcal{H}_{\geq \tau^{(i)}} := \mathcal{H} \setminus \mathcal{H}_{<\tau^{(i)}}$, wherein $z^{(i)}$ denotes the ith value of the expertise context $\vec{z}$ and $\tau^{(i)}$ is the threshold at which we partition. These two experience subsets can then be used to maintain two algorithms for bandits with expert advice, which we will denote respectively by $\mathrm{ALG}_{<\tau^{(i)}}$ and $\mathrm{ALG}_{\geq \tau^{(i)}}$. Let $Q(\mathrm{ALG}, \mathcal{H})$ be an estimate of the performance of some algorithm ALG on the set of experiences $\mathcal{H}$, to be defined more precisely in the following section. We only benefit from a split if the two subset learners perform better than the single learner ALG trained on $\mathcal{H}$, which is the case if

$$Q(\mathrm{ALG}_{<\tau^{(i)}}, \mathcal{H}_{<\tau^{(i)}})|\mathcal{H}_{<\tau^{(i)}}| + Q(\mathrm{ALG}_{\geq \tau^{(i)}}, \mathcal{H}_{\geq \tau^{(i)}})|\mathcal{H}_{\geq \tau^{(i)}}| > Q(\mathrm{ALG}, \mathcal{H})|\mathcal{H}| \quad (1)$$

Note that the two quality estimates are weighted by the sizes of their respective subsets.

3.1.2. Estimating Expertise Quality

Given a set of experiences $\mathcal{H}$ it is useful to ask the question how another policy would have performed in terms of average reward. Following Auer et al. [2002b], we can estimate this expected average reward of a learner ALG and its policy π^{ALG} on that set of experiences as

$$Q(\mathrm{ALG}, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{\langle \boldsymbol{\xi}, k, r, p, \vec{z} \rangle \in \mathcal{H}} \pi_k^{\mathrm{ALG}}(\boldsymbol{\xi}) r / p \quad (2)$$

Wherein $\pi_k^{\mathrm{ALG}}(\boldsymbol{\xi})$ denotes the policy's probability for action k given the advice vector $\boldsymbol{\xi}$. Because these experiences are acquired by a policy (and thus biased), the probability in the denominator ensures the expected reward is unbiased.

We consider therefore the following approach: build an expertise tree wherein the features on which we split are the features of $\vec{z}$, then run an algorithm for bandits with expert advice on the experiences in the leaves. More specifically, while a beneficial split exists (i.e., a split for which (1) holds), we repeatedly split the expertise space into subsets and maintain learners specific to those partitions. Among splits for which (1) holds, we select the split which maximizes the left-hand side of the inequality. Pseudocode is provided in Algorithm 1. This approach effectively combines algorithms for bandits with expert advice with the flexibility of decision trees. The result is a two-step model, on one hand, the decision tree on $\vec{z}$, and on the other hand the model in each leaf. This effectively splits the space of $\vec{z}$ into regions, each with its own policy.

As each potential split requires simulating the run of two algorithms for bandits with expert advice, considering all

Algorithm 1 (Incremental) ExpertiseTree

Require: N experts, contextual bandit with reward function $f : [K] \times \mathbb{R} \rightarrow [0, 1]$
  $root \leftarrow$ Initialize empty expertise tree
  for $t = 1, 2, ..., T$ **do**
    Experts observe the context $\vec{x}_t$
    Get the observable subset of $\vec{x}_t$, i.e., the expertise context $\vec{z}_t$
    Get expert advice $\boldsymbol{\xi}_t = \{\vec{\xi}^1(\vec{x}_t), ..., \vec{\xi}^N(\vec{x}_t)\}$
    $node \leftarrow root$
    while $node$ has a beneficial split **do**
      $node \leftarrow$ child of $node$ containing $\vec{z}_t$
    end while
    Let π_t be the policy of $node$'s learner
    Pull arm $k_t \sim \pi_t(\boldsymbol{\xi}_t)$ and collect resulting reward r_t
    $p \leftarrow \pi_{k_t,t}(\boldsymbol{\xi}_t)$ {get probability of chosen arm}
    if not incremental **then**
      $node \leftarrow root$ {update tree starting from its root}
    end if
    update $node$ with $\langle \boldsymbol{\xi}_t, k_t, r_t, p \rangle$
    while node is not empty **do**
      update candidate splits rooted in $node$ with $\langle \boldsymbol{\xi}_t, k_t, r_t, p \rangle$
      if the best split candidate is such that Equation (1) holds **then**
        split $node$
      end if
      set $node$ to child containing $\vec{z}_t$
    end while
  end for

possible splits for continuous variables becomes intractable. Instead, following Potts & Sammut [2005], we fix κ split candidates for each feature. While this slightly reduces flexibility, it also greatly reduces computation time.

3.1.3. Incremental Expertise Trees

Because retraining an expertise tree for each experience is expensive, we propose an algorithm which constructs the tree incrementally, i.e., at each time step, models are updated with regards to the collected experience, and splits are made if they improve the quality of the expertise tree. In contrast with the non-incremental variants, splits, once made, are not revisited. As a result, if the stochasticity of the problem induces a bad split, it can impact performance going forward. Note that the variance of the quality estimate (2) is such that heuristics which attempt to limit early splits with the use of for example Hoeffding bounds (Pfahringer et al., 2007) are of limited use. However, the additional exploration (as discussed in Section 4) induced by a split acts as an inhibitor or unnecessary splits.

The trade-off for a possible performance decrease is a significantly more efficient algorithm. Fully learning an expertise tree is more time-intensive than incrementally updating an expertise tree. More specifically, assuming an ideal tree depth of h (which is, in turn, logarithmic in the number of partitions), the dimension of the expertise context g, and the number of candidate splits per feature κ, building the corresponding expertise tree requires estimating $O(2^h g\kappa)$ models. Updating the expertise tree requires $O(hg\kappa)$ updates ($g\kappa$ updates for each node on the path to the leaf in which the expertise context resides). In contrast, updating an incremental tree requires only $O(g\kappa)$ model updates on average, as only leaf models are updated.

4. Theoretical Results

In the following, we provide some bounds on the problem of localized expertise, and establish the cost of inappropriately partitioning the context space.

Let $\mathcal{Z}$ be regions of expertise, i.e., a partition of $\mathbb{R}^g$ wherein expertise is homogeneous within each subset, in other words there is a single optimal policy for each subset.

We first establish a lower bound on the regret of algorithms for bandits with localized expertise.

Theorem 1. *The worst-case regret of a meta algorithm whose learners incur regret $R(T)$ is $|\mathcal{Z}|R(T/|\mathcal{Z}|)$.*

Assume $\mathcal{Z}$ is known, such that we can run $|\mathcal{Z}|$ instances of a bandit with expert advice algorithm and assign each experience with expertise context $\vec{z}$ to the corresponding learner. Denote by $p(Z)$ the probability that a sampled context is contained by the region $Z \in \mathcal{Z}$. Let $R(T)$ be the regret incurred by an instance of the bandit with expert advice algorithm. The total regret incurred by all instances is then $\sum_{Z \in \mathcal{Z}} R(p(Z)T)$. In the usual case wherein $R(T) \propto \sqrt{T}$ this measure is minimized when $\max_{Z \in \mathcal{Z}} p(Z) = 1$; which is equivalent to having no localized expertise, thus reducing the regret to $R(T)$. And conversely, it is maximized when the entropy of p is maximized; $p(Z) = 1/|\mathcal{Z}| \ \forall Z \in \mathcal{Z}$, leading to a regret of $|\mathcal{Z}|R(T/|\mathcal{Z}|)$. □

Next, we establish a bound on the localized regret of non-localized algorithms.

Theorem 2. *The worst-case expected regret of any non-localized bandit with expert advice algorithm is $\Omega(\frac{N-1}{N}T)$*

For any expert n, let $p'(n)$ be the proportion of expertise contexts for which expert n is optimal. It then follows that any non-specialized algorithm, i.e., any algorithm that selects the same expert regardless of context, has a probability of at most $p'(n^*) = \max_{n \in [N]} p'(n)$ of selecting the best expert in each region. In particular, the algorithm will incur 0 regret for contexts for which the expert is optimal, and constant regret for all other contexts. The resulting lower bound on the regret is thus $\Omega((1 - p'(n^*))T)$. Any non-

specialized algorithm must therefore incur linear regret in the localized setting. This lower bound is maximized when the entropy of p' is maximized, in which case the regret incurred is $\Omega(\frac{N-1}{N}T)$. □

Finally, we evaluate the conditions which lead to useful splits.

The cost of (not) splitting Let $\mathcal{H}$ be a set of experiences to be split, and $\tau^{(i)}$ the threshold on the i-th feature. Let $\mathcal{Q} = |\mathcal{H}|Q(ALG, \mathcal{H})$ be shorthand for the expected cumulative reward of a learner on the parent with history $\mathcal{H}$ as defined in Equation 2. Similarly, we let the performance on the left and right child be respectively $\mathcal{Q}_l = |\mathcal{H}_{<\tau^{(i)}}|Q(ALG, \mathcal{H}_{<\tau^{(i)}})$ and $\mathcal{Q}_r = |\mathcal{H}_{\geq\tau^{(i)}}|Q(ALG, \mathcal{H}_{\geq\tau^{(i)}})$.

A split is necessary if $\mathcal{Q} < \mathcal{Q}_l + \mathcal{Q}_r$. In other words, if the performance is improved by maintaining two policies.

Let p be the split ratio, i.e., the probability that a context is contained by the left child if it is contained by the parent. Conversely, the probability that an experience is contained by the right child is $1 - p$. We then have that the expected regret of the split is $R(pT) + R((1-p)T)$. Where pT and $(1-p)T$ are respectively the expected number of experiences assigned to the left and right child.

The performance on the unsplit node can be expressed as a function of the gap between the expected cumulative reward on the unsplit node and the weighted sum of expected cumulative rewards on the split.

In particular, let $\mathcal{R} = \max_\pi \mathbb{E}[f(k_{\pi(\boldsymbol{\xi}(\vec{x}))}, \vec{x})]$, where the expectation is over the context sampled from the relevant region of the context space, be the expected reward of the best policy on experiences contained by the parent node. And let $\mathcal{R}_l$ and $\mathcal{R}_r$ analogously be respectively the expected maximum reward in the left and right child.

We then let $\Delta = p\mathcal{R}_l + (1-p)\mathcal{R}_r - \mathcal{R}$ be the gap in expected average reward between the unsplit or split learners.

Theorem 3. *A split with reward gap Δ and split ratio p is beneficial iff $\Delta > (\sqrt{p} + \sqrt{1-p} - 1)R(T)/T$*

By the definition of regret, we have with expectation that the cumulative reward of a learner on the parent node's experiences is $\mathbb{E}[\mathcal{Q}] = \mathcal{R}T - R(T)$, i.e., the optimal performance minus the regret. Similarly, the expected performance on the children's experiences is $\mathbb{E}[\mathcal{Q}_l] = \mathcal{R}_l Tp - R(Tp)$, and $\mathbb{E}[\mathcal{Q}_r] = \mathcal{R}_\imath T(1-p) - R(T(1-p))$

We can then formulate the expected improvement of a split:

$$\begin{aligned}
&\mathbb{E}[\mathcal{Q}_l + \mathcal{Q}_r - \mathcal{Q}] \\
&= (\mathcal{R}_\ell p + \mathcal{R}_\imath(1-p) - \mathcal{R})T - R(Tp) - R(T(1-p)) \\
&\quad + R(T) \\
&= \Delta T - (R(Tp) + R(T(1-p)) - R(T)) \\
&= \Delta T - (\sqrt{p} + \sqrt{1-p} - 1)R(T)
\end{aligned}$$

Where the last equality assumes that $R(T) \propto \sqrt{T}$ (as is typical for expert advice algorithms, see for example EXP4-IX (Neu, 2015) or Meta-CMAB (Abels et al., 2023)). Theorem 3 follows by evaluating when

$$\mathbb{E}[\mathcal{Q}_l + \mathcal{Q}_r - \mathcal{Q}] = \Delta T - (\sqrt{p} + \sqrt{1-p} - 1)R(T) > 0$$

and rearranging. □

In particular this means that a split is only beneficial if the gap in performance is sufficiently large. This is a consequence of the additional exploration induced by the split. Thus, if experts are sufficiently close, i.e. if $\Delta < (\sqrt{p} + \sqrt{1-p} - 1)R(T)/T$ a split is detrimental. As $R(T)/T$ typically is monotonically increasing in both the number of arms K and the number of experts N, but is monotonically decreasing in the number of timesteps T, this confirms the intuition that splits are most beneficial when T is large or when K and N (and the exploration cost they induce) are small.

A special case of this is when $\Delta = 0$, i.e., when the optimal policy in the unsplit node is identical to the policy in the child nodes. In such a case all learners converge towards the same policy, but the child nodes perform some additional exploration, resulting in degraded performance. In particular, this additional exploration penalty is $R(Tp) + R(T(1-p)) - R(T)$. As reasonable algorithms for bandits with expert advice display a sub-linear dependence on T, the resulting sum is positive. And thus, the split incurs a regret proportional to $\sqrt{pT} + \sqrt{(1-p)T} = (\sqrt{p} + \sqrt{1-p})\sqrt{T}$. Which is $(\sqrt{p} + \sqrt{1-p})$ times larger than the regret of the parent node. In particular, this sum is largest when $p = 0.5$, in which case the unnecessary split magnifies the regret by $\sqrt{2}$.

5. Experimental Setting

We evaluate our algorithms in terms of average reward. Throughout we compare the performance of the following algorithms: *Meta-CMAB*, a state-of-the-art approach for bandits with (non-localized) expert advice; *TreeHeuristic* (Elmachtoub et al., 2017) applied to our reduction presented in Section 2.2; *Oracle*, which relies on prior knowledge of the optimal partition and maintains one independent instance of Meta-CMAB per region; *Nearest* 1% and 10%, which take decisions based on models trained with the nearest 1% or 10% of past experiences; Finally, the *(Incremental) Expertise Tree* algorithm, as presented in Section 3.1. Our results are averaged over 100 simulations, and for a varying number of arms and experts. The code to reproduce these results is provided at `https://github.com/axelabels/ExpertiseTrees`.

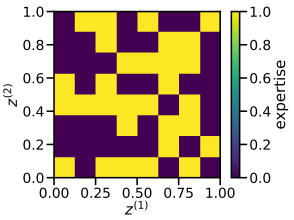

Figure 2: Example of an expertise heatmap with 64 regions. Heatmaps are generated by randomly assigning an expertise value of 0 or 1 to each cell. Each time step one point in this space is sampled and the expertise of each expert is determined by their individual heatmap. For high expertise values, experts provide honest advice, for low values they provide adversarial advice.

5.1. Bandit Problems

Following Riquelme et al. [2018], we build bandits out of classification problems. As a result, for a given context, a single arm (corresponding to the true label) delivers a reward of 1, and the other arms provide 0 reward. We evaluate on a variety of datasets presenting a diversity of feature distributions, arm counts, and reward distributions over the arms, chosen from the openml data repository (Vanschoren et al., 2014). The dataset selection and processing is provided in the supplementary information.

5.2. Simulating Localized Expertise

We consider changes in expertise characterized by expertise heatmaps which map expertise contexts to expert quality. Each round, experts advice on a context vector $\vec{x}_t$ from which a subset of g features (chosen randomly at the start of the experiment) form the expertise context $\vec{z}_t$. Among these g features, 2 features are correlated with expertise, as illustrated in Figure 2. An appropriate learner thus has to learn both which of the g features is relevant, and simultaneously learn an appropriate partition. Each expert is assigned an individual heatmap, and for a given value of $\vec{z}_t$, the expert's quality is determined by the corresponding expertise value. Values close to 0 result in adversarial advice, while high expertise results in honest advice. Heatmaps are obtained by assigning for each expert an expertise value of 0 or 1 to each of the regions (either 1, 4, 16 or 64 regions). Increased number of regions increases the difficulty of the problem. For example, to optimize performance in the hardest configuration we consider, 64 regions must be correctly identified based on a limited number of experiences. By increasing the size of g, we make the problem more complex by increasing the number of irrelevant features.

6. Results and Discussion

We show results in terms of average reward instead of regret as this allows us to display both the loss incurred by increasing the number of regions, as well as the loss in performance of individual algorithms. When presenting results in terms of regret, the former information would be obscured as this loss is incurred by both the oracle and the individual algorithms. Results are averaged over $T = 1000$ steps for all expert counts ($N \in \{4, 32\}$) and dataset combinations.

6.1. Expertise Tree Algorithms are More Robust to Increases in Problem Complexity

Learning in this setting is challenging in two ways, as both the appropriate regions must be identified, and for each region, an appropriate model needs to be learned. To isolate the loss of performance resulting from the need to identify these regions, we compare the performance of our algorithms to an Oracle algorithm. For this algorithm, we assume that perfect knowledge of the regions is known, hence any loss in performance of the Oracle variant results only from uncertainty about the appropriate model for each region. This allows us to isolate how much performance loss of non-oracular variants results from uncertainty about the structure of the expertise space.

Figure 3 illustrates the change in performance of our algorithms as the number of regions increases. These plots show that when specializations are present (i.e., more than 1 region), the non-specialized Meta-CMAB algorithm fails to learn. These plots also show that, as the number of regions increases, the performance of specialized algorithms declines more rapidly than the performance of Oracle. This is a consequence of the uncertainty about the partitioning of the expertise context space. This is because appropriate partitions for a high number of regions require several optimal splits. Figure 3 also shows that, for a given number of regions, the size of the expertise context makes it harder to identify the ideal partition, again illustrated by the growing gap with the performance of the Oracle. This is because random correlations can occur when the number of features is larger, inducing splits without actual discriminating power. Although this causes all specialized algorithms to degrade, it is less pronounced for both expertise tree variants. The incremental variant of the ExpertiseTree suffers more from the irrelevant features as splits resulting from random correlations cannot be reversed.

Nearest Neighbor models trained on either 1 or 10% of experiences assume that the expertise space is divided into approximately 100 or 10 regions. As the number of regions drifts towards the former, the performance of the 1% variant improves relative to the 10%. In addition, as the size of the expertise context increases, the quality of the distance metric as a proxy for similarity in terms of expert quality decreases. This results in a decline in the performance of the Nearest Neighbor algorithms, even when the number of regions is small.

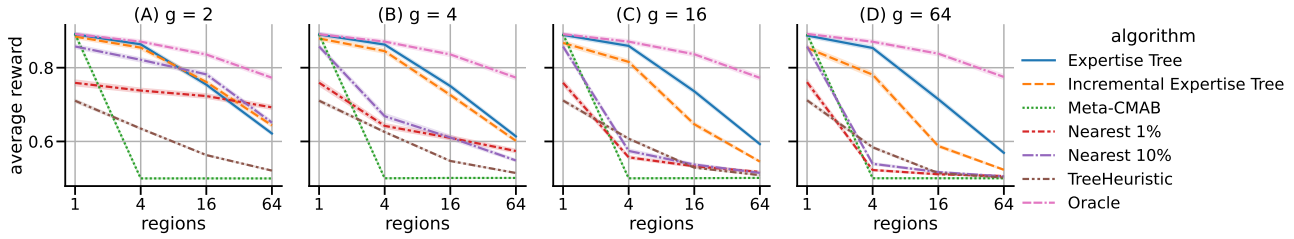

Figure 3: Average reward as a function of the number of regions (x-axis) and expertise context size (g, columns) for different algorithms. Shaded areas show the 95% confidence interval.

Finally, while the TreeHeuristic algorithm can perform the same feature selection and partitioning as the ExpertiseTree algorithm, it is penalized by its single expert approach. This not only impacts the best decision that can be made by TreeHeuristic at any timestep, but it also impacts its updates, as each experience is used to update a single expert's quality estimate. This effectively means that TreeHeuristic has N fewer experiences per expert than our methods.

Note that, for all algorithms, including the Oracle, a loss of performance relative to a non-localized setting is inevitable. More regions require more individual models, but, given the constant number of experiences, each individual model is trained less, leading to poorer performance.

6.2. Localized Algorithms on Global Expertise

If expertise is not localized, localized models are likely to perform worse, as they take fewer relevant experiences into account. The leftmost point of each sub-figure in Figure 3 illustrates the performance of various algorithms when experts are not localized (i.e., when there is a single region). Notably, the performance of TreeHeuristic is significantly inferior to other algorithms. This performance loss results from TreeHeuristic's policy which selects one expert per learned partition. As a result, its performance is bounded by the performance of the best expert. This is a limitation not shared by the other algorithms. It is however noteworthy that the Nearest Neighbor algorithms show some loss in performance compared to the non-specialized Meta-CMAB. Because these algorithms learn a model on a subset of all experiences, the learned model generalizes less well, leading to a loss of performance. This is particularly the case when only 1% of experiences are taken into account. In contrast, the ExpertiseTree approach will maintain a single partition, as no beneficial splits exist. The performance of the ExpertiseTree is almost identical to the simple Meta-CMAB, even when the size of the expertise context increases significantly. The Incremental ExpertiseTree shows a more significant performance loss as the expertise context increases in size. These larger sizes are more likely to result in the observation of false correlations, inducing unnecessary splits which the Incremental ExpertiseTree is unable to revert.

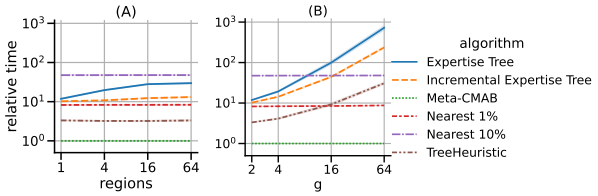

Figure 4: Relative execution time for a varying number of (A) regions or (B) expertise context size. Relative time is obtained by dividing the execution time of algorithms by the execution time of Meta-CMAB. Both axes are logarithmic. The execution times of the ExpertiseTree methods grow logarithmically with the number of regions, and quasilinearly with the expertise context size.

6.3. Execution Time

Figure 4 illustrates that the execution time of most algorithms is constant as a function of the number of regions or the expertise context size. In particular, the Nearest Neighbor algorithms learn a single model each round, on a set of experiences which is independent of the number of regions or the expertise context size. The neighborhood size does have a significant impact however. A larger neighborhood results in models trained on more experiences, hence a higher execution time. Note that a larger expertise context size increases the cost of distance calculations, resulting in slightly increased execution times.

The trade-off for ExpertiseTree's improvements in terms of average reward is a significant increase in computation time. These plots confirm the theoretical increases in execution time, with a logarithmic dependence on the number of regions and a linear dependence on the size of the expertise context. The resulting execution time can be prohibitive for time-sensitive problems but remains reasonable for, for example, problems with human expertise. Note that in particular, the cost of the ExpertiseTree results from the number of potential splits, thus from the size of the expertise context. This can be improved by dismissing irrelevant features (either preemptively or based on the observed data).

7. Conclusion

In this work, we proposed a generalization of bandits with expert advice which explicitly models localized expertise. Methods which account for localized expertise are essential in real-world settings in which human and artificial expertise depends on a set of features. Observing that prior algorithms in this setting were not conditioned on this set of features, we proposed two novel algorithms of increasing performance and complexity. The first algorithm relies on nearest-neighbor queries to identify relevant experiences but fails when the size of the neighborhood is not well chosen or when the dimensionality of the feature space is high, which is a known hurdle for nearest-neighbor approaches (Domingos, 2012). To address these drawbacks, our expertise tree approach learns a tree which splits the expertise space as a function of the relevant features and then learns to act on all expert advice in each learned partition. This allows it to perform strongly even when the number of regions or the size of the expertise context is large. To enhance its applicability, we believe future work should address ExpertiseTree's high computational cost. In particular, online feature selection could be beneficial to reduce the computational cost induced by high dimensional contexts. While this work's focus is to establish the viability of expertise trees, a reduction in computational cost would render common extensions to decision trees more practical in the context of expertise trees. Future work could for example evaluate the use of ensembles (i.e., Expertise Forests which would be analogous to Random Forests (Breiman, 2001)), or employ other more advanced techniques which have been developed for decision trees, such as augmented decision boundaries (Murthy et al., 1994).

Acknowledgments

A.A. is supported by a FRIA grant (nr. 5200122F) by the National Fund for Scientific Research (F.N.R.S.) of Belgium. T.L. is supported by the F.N.R.S. project with grant numbers 31257234 and 40007793, the F.W.O. project with grant nr. G.0391.13N, the Service Public de Wallonie Recherche under grant n°2010235–ARIAC by DigitalWallonia4.ai. T.L and A.N. benefit from the support of the Flemish Government through the AI Research Program. T.L., V.T. and A.N. acknowledge the support by TAILOR, a project funded by EU Horizon 2020 research and innovation program under GA No 952215. V.T. acknowledges the support by HACID, a project funded by EU Horizon Europe research and innovation program under GA No 101070588. The resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

References

Abels, A., Lenaerts, T., Trianni, V., and Nowé, A. Dealing with expert bias in collective decision-making. *Artificial Intelligence*, pp. 103921, 2023.

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

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b. doi: 10.1137/S0097539701398375.

Bond, S. D., Carlson, K. A., Meloy, M. G., Russo, J. E., and Tanner, R. J. Information distortion in the evaluation of a single option. *Organizational Behavior and Human Decision Processes*, 102(2):240–254, 2007.

Bottou, L. and Vapnik, V. Local learning algorithms. *Neural computation*, 4(6):888–900, 1992.

Breiman, L. Random forests. *Machine learning*, 45:5–32, 2001.

Chu, W., Li, L., Reyzin, L., and Schapire, R. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214, 2011.

Domingos, P. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

Elmachtoub, A. N., McNellis, R., Oh, S., and Petrik, M. A practical method for solving contextual bandit problems using decision trees. *CoRR*, abs/1706.04687, 2017.

Foster, D. and Rakhlin, A. Beyond ucb: Optimal and efficient contextual bandits with regression oracles. In *International Conference on Machine Learning*, pp. 3199–3210. PMLR, 2020.

Gianfrancesco, M. A., Tamang, S., Yazdany, J., and Schmajuk, G. Potential biases in machine learning algorithms using electronic health record data. *JAMA internal medicine*, 178(11):1544–1547, 2018.

Keswani, V., Lease, M., and Kenthapadi, K. Towards unbiased and accurate deferral to multiple experts. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 154–165, 2021.

Murthy, S. K., Kasif, S., and Salzberg, S. A system for induction of oblique decision trees. *Journal of artificial intelligence research*, 2:1–32, 1994.

Neu, G. Explore no more: Improved high-probability regret bounds for non-stochastic bandits. *Advances in Neural Information Processing Systems*, 28, 2015.

Pfahringer, B., Holmes, G., and Kirkby, R. New options for hoeffding trees. In *Australasian Joint Conference on Artificial Intelligence*, pp. 90–99. Springer, 2007.

Pohl, R. *Cognitive illusions : a handbook on fallacies and biases in thinking, judgement and memory*. 2017.

Potts, D. and Sammut, C. Incremental learning of linear model trees. *Machine Learning*, 61(1-3):5–48, 2005.

Riquelme, C., Tucker, G., and Snoek, J. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations*, 2018.

Thompson, W. R. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. doi: 10.2307/2332286.

Trope, Y., Gervey, B., and Liberman, N. Wishful thinking from a pragmatic hypothesis-testing perspective. *The mythomanias: The nature of deception and self-deception*, pp. 105–31, 1997.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pp. 654–663, 2013.

Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

Wang, Y. and Witten, I. H. Induction of model trees for predicting continuous classes. 1996.

Whitehead, S. J. and Ali, S. Health outcomes in economic evaluation: the qaly and utilities. *British medical bulletin*, 96(1):5–21, 2010.

Supplementary Information: Expertise Trees Resolve Knowledge Limitations in Collective Decision-Making

Axel Abels[1,2], **Tom Lenaerts**[1,2,3], **Vito Trianni**[4], **Ann Nowé**[2]

A. Dataset Selection

We sample datasets from the openmpl (Vanschoren et al., 2014) repository which match the following criteria.

1. Sufficient number of instances: in order to allow us to repeat our experiments on each dataset, we restrict our experiments to datasets which contain at least $10,000$ instances.
2. Classification: we restrict our experiments to classification bandits.
3. Missing Values: we only consider complete datasets and thus drop datasets which contain missing values.

These filters result in the selection of the following datasets: mushroom, adult, letter, nursery, pendigits, BNG(page-blocks,nominal,295245), BNG(glass,nominal,137781), BNG(tic-tac-toe), BNG(vote), electricity, covertype, kropt, BNG(breast-w), BNG(page-blocks), BNG(glass), mammography, eye_movements, mozilla4, KDDCup99, MagicTelescope, Click_prediction_small, artificial-characters, bank-marketing, eeg-eye-state, kr-vs-k, ldpa, skin-segmentation, spoken-arabic-digit, walking-activity, volcanoes-b1, creditcard, Amazon_employee_access, CreditCardSubset, PhishingWebsites, Diabetes130US, numerai28.6, fars, shuttle, Run_or_walk_information, tamilnadu-electricity, jungle_chess_2pcs_raw_endgame_complete, MiniBooNE, jannis, helena, microaggregation2

For each of these datasets we dummy and normalize all features.

B. Rank Results

Figure 5 illustrates the average rank of each algorithm as a function of the number of regions and the expertise context dimension.

C. Continuous Expertise

Results in the main paper consider experts whose expertise is in $\{0, 1\}$ depending on the region. Results here differ by considering expertise in $[0, 1]$ instead, as illustrated by Figure 6.

Figures 7 and 8 show that results are qualitatively the same, i.e., conclusions with regards to the efficacy of our approach remain valid. In particular, less extreme expertise makes regions of expertise harder to identify for all methods, resulting in scaled down performance. In addition, it results in decreased performance even when the regions are known, as the decreased performance of the oracle shows. Our proposed approach however remains advantageous as its benefits still apply.

[1]Machine Learning Group, Université Libre de Bruxelles, Brussels, Belgium
[2]AI Lab, Vrije Universiteit Brussel, Email: axel.abels@ulb.be
[3]Center for Human-Compatible AI, UC Berkeley, Berkeley, USA
[4]Institute of Cognitive Sciences and Technologies, National Research Council, Rome, Italy

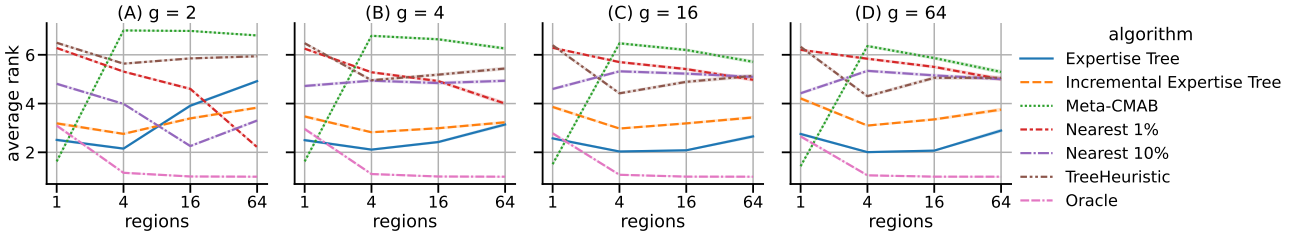

Figure 5: Average rank (lower is better) as a function of the number of regions (x-axis) and expertise context size (g, columns) for different algorithms. Shaded areas show the 95% confidence interval.

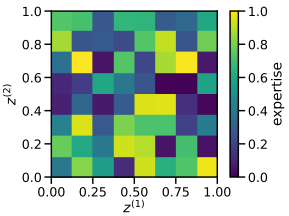

Figure 6: Example of an expertise heatmap with 64 regions. Heatmaps are generated by randomly assigning an expertise value in $[0, 1]$ to each cell. Each time step one point in this space is sampled and the expertise of each expert is determined by their individual heatmap. For high expertise values, experts provide honest advice, for low values they provide adversarial advice.

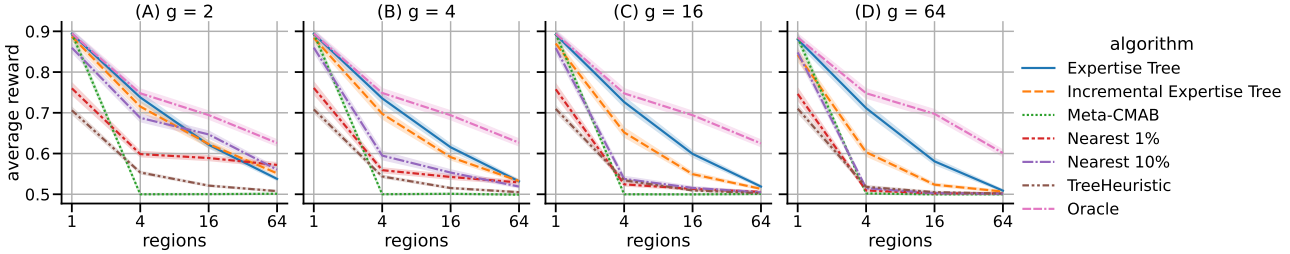

Figure 7: Average reward as a function of the number of regions (x-axis) and expertise context size (g, columns) for different algorithms when expertise is **continuous**. Shaded areas show the 95% confidence interval.

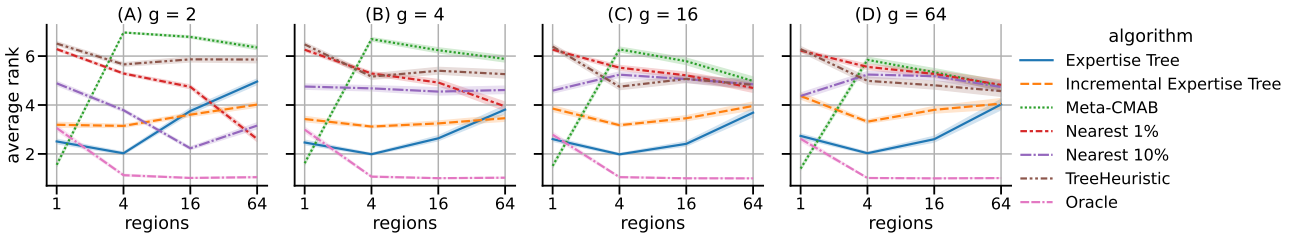

Figure 8: Average rank (lower is better) as a function of the number of regions (x-axis) and expertise context size (g, columns) for different algorithms when expertise is **continuous**. Shaded areas show the 95% confidence interval.