Ideal Abstractions for Decision-Focused Learning

Michael Poli* Microsoft
Stefano Massaroli* Mila
Stefano Ermon Stanford
Bryan Wilder Carnegie Mellon
Eric Horvitz Microsoft

Abstract

We present a methodology for formulating simplifying abstractions in machine learning systems by identifying and harnessing the utility structure of decisions. Machine learning tasks commonly involve high-dimensional output spaces (e.g., predictions for every pixel in an image or node in a graph), even though a coarser output would often suffice for downstream decision-making (e.g., regions of an image instead of pixels). Developers often hand-engineer abstractions of the output space, but numerous abstractions are possible and it is unclear how the choice of output space for a model impacts its usefulness in downstream decision-making. We propose a method that configures the output space automatically in order to minimize the loss of decision-relevant information. Taking a geometric perspective, we formulate a step of the algorithm as a projection of the probability simplex, termed *fold*, that minimizes the total loss of decision-related information *in the H-entropy sense*. Crucially, learning in the abstracted outcome space requires less data, leading to a net improvement in decision quality. We demonstrate the method in two domains: data acquisition for deep neural network training and a closed-loop wildfire management task.

1 INTRODUCTION

Modern machine learning systems process high-dimensional data such as gigapixel images (Litjens et al., 2022) or graphs with billions of nodes (Zheng et al., 2020). How can machine learning efforts and outputs at this scale be most appropriately matched to predictions made in support of real-world decision-making? Further, how does one go about handling domains where the dimensionality of the problem is so large that one cannot simply collect enough data for a predictive model to "explore" its ambient space?

It has been shown that if collecting a sufficient amount of data is possible, deep learning provides effective methods to compress the information content into a set of parameters (Bommasani et al., 2021) which can then be adapted to overcome data constraints in other similar tasks. We focus on domains where it is not possible to acquire enough data for systematic generalization of large models to occur, based in the intrinsic properties of the domain, e.g., sufficient data simply does not exist (Hersbach et al., 2020) or is too expensive to acquire. We introduce and develop a framework to tame this fundamental challenge with the traditional collect-data-and-compute–first approach by incorporating knowledge about downstream tasks. The key direction of distilling ideal abstractions for decision-focused machine learning is inspired by earlier work on utility, abstraction, and information selection in a decision-making setting (Horvitz and Klein, 1993; Poh et al., 1994; Horvitz and Barry, 1995; Bach et al., 2006a; Kapoor and Horvitz, 2009; Azuma et al., 2006).

In this work, we adopt a decision-theoretic perspective to machine learning. We derive a computationally efficient method to abstract away information that is not relevant to the decision task at hand, harnessing clues about problem structure, and in the process, reduce the dimensionality of upstream prediction problems. Concretely, we cast the search for the right abstractions into an optimization problem based on a geometric perspective. We introduce a class of algorithms we refer to as `ORIGAMI` that iteratively aggregate sets of outcomes through projections, termed *folds*, of the probability simplex. Such projections are driven by the information content of each outcome with respect to the downstream task, which can be naturally measured via the Bayes loss of an optimal decision maker (DeGroot, 1962; Zhao et al., 2021). Each fold hides information from downstream agents, gradually coarsening the support of context random variables, and allowing upstream predictive models to learn over sets with less data. The method notably decouples upstream prediction with downstream decision-making, allowing inspection of the learned abstractions used to drive policies.

* Equal contribution authors. Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

The structure of the paper and key contributions are as follows. §3 contains background on decision-theoretic information, and describes the operational primitives of the novel class of ORIGAMI algorithms. We also discuss the choice of projection operators and extensions of decision losses to sets. In §4, we detail three different objective functions to drive the projections, outlining computation-accuracy trade-offs. We further discuss a deep neural network surrogate for ORIGAMI that can be trained to approximate the algorithm over a class of decision losses. In §5 we validate ORIGAMI in data-limited deep active learning as in a closed-loop decision task involving wildfire management, where policies based on predictions over ORIGAMI abstractions are shown to perform with lower losses.

2 BACKGROUND

Notation Let $p(x, z)$ denote the underlying data generating process relating variables x and outcome variables z, and $p(z|x)$ the conditional distribution over a finite set $\mathbb{Z}$ of size $|\mathbb{Z}| = C$. An agent observes x, and given a model $p_\theta(z|x)$ of $p(z|x)$, returns an action a following the policy $\pi(Z)$. Domain-knowledge about the task is represented as a loss function $\ell : \mathbb{Z} \times \mathbb{A} \to \mathbb{R}$ measuring the cost of performing action a when the outcome is z.

As both outcome and action spaces are assumed to be of finite dimensions, the loss function can be conveniently represented by a matrix $\boldsymbol{L} \in \mathbb{R}^{|\mathbb{A}|\times C}$ defined as

$$\boldsymbol{L}_{ij} = \ell(z_i, a_j).$$

Further, $\boldsymbol{p} = \{p(z_i|x)\}_i$ is a vector in $\mathbb{R}^C$ taking values in the probability simplex Δ^C. In practice $p(x, z)$ is not known and we are given a dataset $\{x_k, z_k\}$ of samples, in addition to a decision loss ℓ, with the final objective of identifying the best policy.

Problem setting We are interested in domains where the space of outcomes $\mathbb{Z}$ for the random variable Z is high-dimensional, e.g., the set of all possible medical conditions, or the space of geographical locations. As an example, consider the setting where a clinician is tasked with choosing an optimal treatment for a patient given the distribution $\boldsymbol{p}_\theta$ over a set of patient states, given measurements x. Here, optimizing a model for $\boldsymbol{p}_\theta$ or the policy π can be challenging and require a large number of samples from $p(x, z)$. To overcome this limitation, we propose to reduce the dimensionality of Z in a way that preserves as much useful information as possible for downstream decision-making. Our main targets are scalable methods, including approaches that generate explainable abstractions to enable compatibility with human decision-makers.

The core insight behind our approach is that not all the information contained in $\boldsymbol{p}_\theta$ is necessary for decision-making. We take inspiration from human decision-making, where action under uncertainty appears to be taken swiftly with redundant information being abstracted away (Lindig-León et al., 2019; Ho, 2019).

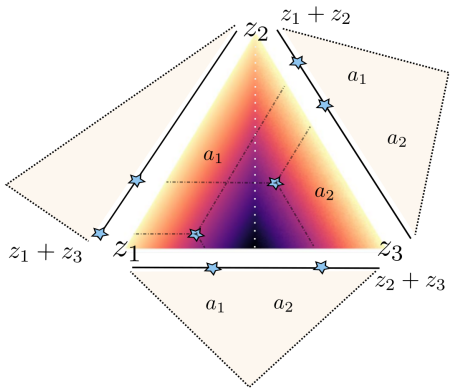

Figure 2.1: Folding the probability simplex can introduce a sub-optimality gap in downstream decision-making. Some distributions $\boldsymbol{p}$ remain on the same side of the decision boundary (in white), whereas others switch sides.

3 DECISION-THEORETIC INFORMATION

Our goal to find a complete partition for the support of $p(z|x)$ i.e., $\mathcal{P}(\mathbb{Z}) = \{\mathbb{Z}_k\}$, $\mathbb{Z}_i \bigcap \mathbb{Z}_j = \emptyset$ for any $i \neq j$ and $\bigcup \mathbb{Z}_k = \mathbb{Z}$. Out of all possible partitions of the set, we seek those that minimally affect decision-making, as measured by the loss ℓ. In other words, we aim to hide information that is not relevant to the decision task. A natural quantity to consider is the H-entropy (DeGroot, 1962; Zhao et al., 2021) of $p(z|x)$:

$$\begin{aligned} H_\ell(\boldsymbol{p}) &= \inf_{a\in\mathcal{A}} \mathbb{E}_{p(z|x)}[\ell(Z, a)] \\ &= \min_a(\boldsymbol{L}\boldsymbol{p}). \end{aligned} \tag{1}$$

where $\boldsymbol{L}\boldsymbol{p} \in \mathbb{R}^{|\mathbb{A}|}$. H-entropy is the Bayes optimal loss for an agent required to select an optimal action a in expectation over $p(z|x)$, and generalizes other notions of information.

For convenience of notation, we will henceforth denote vectors $p(z|x)$ with $\boldsymbol{p}$. Defining a partition $\mathcal{P}(\mathbb{Z})$ naturally induces a distribution $\boldsymbol{q}$ with support $\mathcal{P}(\mathbb{Z})$. Thus, we can quantify the increase of H-entropy caused by partitioning the support $\mathbb{Z}$:

$$\delta(\boldsymbol{q}, \boldsymbol{p}) = H_{\tilde{\ell}}(\boldsymbol{q}) - H_\ell(\boldsymbol{p})$$

which we refer to as the H-entropy suboptimality gap of $\mathcal{P}(\mathbb{Z})$. For the above to be well-defined, we require a decision loss over the sets in $\mathcal{P}(\mathbb{Z})$, denoted as $\tilde{\ell}$. We detail how to define set extensions of ℓ in Sec. 3.2.

3.1 How to Fold a Simplex

We cast the search for a partition $\mathcal{P}(\mathbb{Z})$ through a geometric lens, leveraging the structure of the simplex Δ^C. Our basic

operation will involve *folding* the simplex:

Definition 3.1 (Simplex fold)**.** *A fold is a map* $f_{i\to j} : \Delta^C \to \Delta^{C-1}$; $\boldsymbol{p} \mapsto \boldsymbol{q}$ *defined as*

$$f_{i\to j} = \begin{cases} \boldsymbol{q}_k = \boldsymbol{p}_k & \forall k \neq i, \forall k \neq j \\ \boldsymbol{q}_j = \boldsymbol{p}_i + \boldsymbol{p}_j & \textit{otherwise} \end{cases}$$

A fold projects elements of Δ^C onto Δ^{C-1}. There is an intuitive interpretation for the output of a folding operation: two outcomes z_i, z_j are grouped together into a set, and $\boldsymbol{q}_j$ is the probability that either z_i or z_j occur.

Example: Consider a three-dimensional simplex Δ^3, i.e., with $|\mathbb{Z}| = 3$, and a loss function

$$\boldsymbol{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The simplex and decision boundaries are visualized in Fig. 2.1. Along each side of the simplex, we show the decision loss over three projections to Δ^2 obtained by summing the probabilities of outcomes z_1, z_2, z_3 pairwise. Some points $\boldsymbol{p} \in \Delta^C$ do not cross the decision boundary after the projection, whereas others do, introducing a suboptimality gap. The decision boundary is linear in this example since $|\mathbb{A}| = 2$[a].

[a] The general case studied in this paper is $|\mathbb{A}| > 2$, where the decision boundaries are piecewise-linear.

Partition as a sequence of folds We uniquely identify a partition $\mathcal{P}(\mathbb{Z})$ via the sequence of folds

$$f_{i_N\to j_N} \circ \cdots \circ f_{i_1\to j_1}$$

where i_n, j_n are the folding indexes at algorithm iteration n. Consider the example in Figure 3.1, where the partition $\mathcal{P}(\mathbb{Z}) = \{1, \{2,3\}, \{4,5\}\}$ is identified through $f_{4\to 5} \circ f_{3\to 2}$.

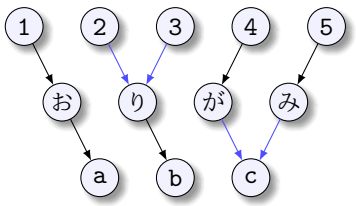

Figure 3.1: $\mathcal{P}(\mathbb{Z}) = \{1, \{2,3\}, \{4,5\}\}$ identified via two folds of Δ^5. Name changes to classes indicate the implicit change of probability space at every folding.

Iterative folding constructs a tree, starting from the C vertices of Δ^C as leaves. Every fold adds a level, merging two nodes. At termination, each top-level node defines a set in the final partition $\mathcal{P}(\mathbb{Z})$, with elements identified as the leaves reachable from it.

3.2 Computing Decision Losses on Sets

We seek an algorithmic procedure that iteratively folds the simplex until a reaching a stopping condition. We require an extension to ℓ that admits set-valued inputs. In the following, we consider the natural worst-case extension,

$$\tilde{\ell}(S, a) = \max_{z\in S} \ell(z, a), \quad S \subset \mathbb{Z}.$$

The matrix representation follows by replacing, for each row k, column i with the maximum of columns i and j: $\tilde{\boldsymbol{L}}_{kj} \leftarrow \max\{\boldsymbol{L}_{ki}, \boldsymbol{L}_{kj}\}$, in time $\Theta(|\mathbb{A}|)$.

We note that, while theoretically possible, other free-form (mass preserving) projections may not have a sensible physical interpretation. Folding the simplex, as prescribed by (3.1), has the effect of grouping two outcomes together into a set, such that is remains possible to reason about worst-case decision losses. Numerically, this choice is key to preserving fast updates $\Theta(|\mathbb{A}|)$ to the decision loss matrix $\boldsymbol{L}$ required to compute ℓ over sets.

Properties of partitions Assume to be given a perfect model of $\boldsymbol{p}$ and a perfect decision making policy. Hiding information by partitioning its support can never improve the policy. Intuitively, this is due to the fact that information about events is now conveyed at a coarser level via sets $\{\mathbb{Z}\}_k$ in the partition, rather than at the finer level of individual events.

Proposition 1 (Folding increases H-entropy)**.** *Let* $\boldsymbol{p} \in \Delta^C$ *and* $\boldsymbol{q} = f \circ f \cdots \circ f(\boldsymbol{p})$ *be any sequence of folds. Then,*

$$H_\ell(\boldsymbol{p}) \leq H_{\tilde{\ell}}(\boldsymbol{q}).$$

In words, partitioning the support of $\boldsymbol{p}$ raises the optimal lower bound decision loss. Specifically, the Bayes optimal loss lower bound increases due to the worst-case set extension.

Remark: Other set extensions for ℓ are available. We discuss a weighted sum extension where

$$\tilde{\ell}(S, a) = \sum_{k:z_k\in S} \frac{\ell(z_k, a)\boldsymbol{p}_k}{\sum_{k:z_k\in S} \boldsymbol{p}_k}.$$

In this case, prop. 1 holds with equality: H-entropy is preserved by folding. How can the optimal decision loss not be affected by projecting the simplex down to a smaller dimension, effectively hiding information? This paradox is explained by noting that, through a fold and corresponding choice of set extension, one affects not only the **information content** but also **the downstream task**. For example, a summing extension

$$\tilde{\ell}(S, a) = \sum_{z\in S} \ell(z, a).$$

penalizes an outcome based on other outcomes in the same set, regardless of whether they occur or not. This results in penalizing larger sets in the par-

tition $\mathcal{P}(\mathbb{Z})$, biasing the projections found by the algorithm.

Interestingly, we observe that the utility of decisions can improve if one optimizes a model $\boldsymbol{q}_\theta$ on the *lower resolution* support given by sets in $\mathcal{P}(\mathbb{Z})$ rather than on the original support, particularly in data-limited regimes. An interpretation of this phenomenon is that partitioning into sets acts as a form of regularization for $\boldsymbol{q}_\theta$ by hiding information not relevant to the downstream task.

4 ORIGAMI: ALGORITHMIC FOLDING

Each fold renders two outcomes indistinguishable from the perspective of the decision-maker. If $H_\ell(\boldsymbol{p}) = H_{\tilde{\ell}}(f_{i\to j}(\boldsymbol{p}))$, z_i and z_j are already equivalent for the decision task induced by ℓ, and can thus be treated as a unique outcome without suboptimality. We have thus established a high-level desideratum for a folding algorithm: minimize at each step the suboptimality gap $\delta(\boldsymbol{p}, \boldsymbol{q})$ induced by the projection. However, the gap discussed so far is local in the simplex, evaluating δ on two vectors $\boldsymbol{p} \in \Delta^C$ and $\boldsymbol{q} \in \Delta^{C-1}$.

In practice, we have access to a dataset with samples from $p(z, x)$, yielding conditionals $p(z|x) \in \Delta^C$. Here, applying a fold $f_{i\to j}$ introduces a suboptimality gap at each point.

Folding objective Following this reasoning, one can cast each ORIGAMI step as the following program:

$$i^*, j^* = \arg\min_{i,j,i\neq j} \mathcal{L}(i, j, \boldsymbol{L}) \tag{2}$$

where i^*, j^* are the indices of the optimal fold $f_{i^*\to j^*}$. The following discussion details three choices of objectives $\mathcal{L}$ that take into account different global information about the suboptimality induced by $f_{i\to j}$: *total*, *worst-case*, and *vertex-only*.

4.1 Integral Objective

The first objective relies on evaluating δ over the entire simplex:

$$\mathcal{L} = \frac{1}{\lambda} \int_{\Delta^C} \left[H_{\tilde{\ell}}(f_{i\to j}(\boldsymbol{p})) - H_\ell(\boldsymbol{p}) \right] \mathrm{d}\boldsymbol{p}. \tag{3}$$

where $H_{\tilde{\ell}}$ denotes the H-entropy endowed with the *folded* loss matrix, $H_{\tilde{\ell}}(f_{i\to j}(\boldsymbol{p})) = \min_a(\tilde{\boldsymbol{L}}\boldsymbol{q})$. This choice of objective corresponds to the L_1 norm of H-entropy increase and can be evaluated via Monte Carlo (MC) integration, thus requiring computationally costly sampling of N vectors $\boldsymbol{p}$ in Δ^C and evaluation of $\delta(f_{i\to j}(\boldsymbol{p}), \boldsymbol{p})$ for all choices of i, j. By standard Law of Large Numbers arguments, the variance $\mathbb{V}$ of a Monte Carlo estimate $\hat{\mu}_N$ of the total integral loss (3)

$$\hat{\mu}_N(i, j) = \frac{1}{N} \sum_{k=1}^{N} \left[H_{\tilde{\ell}}(f_{i\to j}(\boldsymbol{p}_k)) - H_\ell(\boldsymbol{p}_k) \right]$$

can be shown to converge linearly i.e., $\mathbb{V}[\hat{\mu}_N(i, j)] = \mathcal{O}(1/N)$ in the number of samples regardless of the dimension C.

Proposition 2 (Integral objective cost)**.** ORIGAMI *driven by the objective* (3)*, with an* ϵ *requirement* $\mathbb{V}[\hat{\mu}_N] \leq \epsilon$ *has an asymptotic time cost of* $\mathcal{O}(\frac{1}{\epsilon}|\mathbb{A}|C^2)$.

Proof. We report here a proof sketch. For each pair of vertices in the simplex Δ^C, $\frac{1}{2}C(C-1)$, we incur a cost C^2 to compute $\boldsymbol{L}\boldsymbol{p}$ and $|\mathbb{A}|$ to find its minimum entry. This process has to be repeated $1/\epsilon$ for the variance of the MC estimate to be smaller than ϵ. □

The minimization of the empirical estimate of (3) is then practically achieved by constructing the upper triangular portion of the matrix $\boldsymbol{M}_{ij} = \mu_N(i, j)$ and subsequently choosing the indices (i^*, j^*) of the smallest entry of $\boldsymbol{M}$. We report pseudocode below[1].

```
# Fold with integral objective.
# Input: Δ^c, L, N.
M = zeros(c, c) #
M = M + 10^4 # large initial distance
p = uniform_sample(N, c) # on the simplex
for (i, j) in combinations(range(c), 2):
    H_p = einsum("ac,bc->ba", L, p).min(dim=1)
    q, Lt = fold(p, L, i, j)
    H_q = einsum("ac,bc->ba", Lt, q).min(dim=1)
    M[i, j] = (H_q - H_p).mean(dim=0)
i_fold, j_fold = argmin2d(M)
```

We further note that importance sampling and other variance reduction techniques may offer slight improvement to the convergence rate of $\hat{\mu}_N(i, j)$, reducing the overall cost of an ORIGAMI fold. Instead, we leverage the structure of H_ℓ to develop alternative formulations to the integral objective.

4.2 Max-Increase Objective

Instead of the total loss of H-entropy (in a L_1 sense), we can choose folds that minimize the worst-case increase:

$$\begin{aligned} \mathcal{L} &= \sup_{\boldsymbol{p}\in\Delta^C} \left[H_{\tilde{\ell}}(f_{i\to j}(\boldsymbol{p})) - H_\ell(\boldsymbol{p}) \right] \quad \Leftrightarrow \\ &= \max_{\boldsymbol{p}\in\Delta^C} \left[\min_a(\tilde{\boldsymbol{L}}\boldsymbol{q}) - \min_a(\boldsymbol{L}\boldsymbol{p}) \right]. \end{aligned} \tag{4}$$

That is, the infinity norm of H-entropy increase induced by a fold $i \to j$. To find $i^*, j^* = \min_{i,j} \mathcal{L}(i, j, \boldsymbol{L})$ one has to solve, for each pair of indices, the inner optimization

[1] The inner for-loop is fully parallelizable.

problem

$$\max_{\boldsymbol{p}\in\Delta^C} \left[\underbrace{\min_a(\tilde{\boldsymbol{L}}\boldsymbol{q})}_{\text{concave}} - \underbrace{\min_a(\boldsymbol{L}\boldsymbol{p})}_{\text{concave}} \right] \tag{5}$$

which belongs to the class of *difference of convex or concave* (DC) problems (Hartman, 1959). Here, we employ the *concave-convex procedure* (Lipp and Boyd, 2016), a class of heuristic algorithms to find local solutions to DC problems.

Solving the inner-loop problem The simplest variant of a concave-convex procedure to compute $\mathcal{L}$ starts by sampling an initial candidate maximizer $\boldsymbol{p}^0 \in \Delta^C$. Then, the candidate maximizer $\boldsymbol{p}^k$ is updated as follows: the convex part of the problem is linearized around $\boldsymbol{p}^k$,

$$\hat{H}_\ell(\boldsymbol{p}, \boldsymbol{p}^k) = \min_a \boldsymbol{L}\boldsymbol{p}^k + g_k^\top(\boldsymbol{p} - \boldsymbol{p}^k)$$

where g_k is a subgradient of H_ℓ i.e., $g_k \in \partial H_\ell(\boldsymbol{p}^k)$. The candidate maximizer is then updated by solving the concave problem resulting from substituting $H_\ell(\boldsymbol{p})$ with its linearization, i.e.

$$\boldsymbol{p}^{k+1} = \arg\max_{\boldsymbol{p}\in\Delta^C} \left[\min_a \tilde{\boldsymbol{L}}\boldsymbol{q} - \hat{H}_\ell(\boldsymbol{p}, \boldsymbol{p}^k) \right]$$

The algorithm is iterated until convergence, e.g., when the improvement in the true objective is less than a specified threshold. This adaptation of the convex-concave procedure to compute the objective for ORIGAMI folding leverages on the assumption that, at each step, the concavified problems can be solved efficiently (see Lipp and Boyd (2016) for further details and variants of this method).

Similar to the integral loss case, the objective needs to be computed for each unordered tuple (i, j) in order to chose the optimal folding.

```
# Fold with max-increase objective.
# Input: Δ^c, L.
M = zeros(c, c) #
M = M + 10^4 # large initial distance
for (i, j) in combinations(range(c), 2):
    p = solve_inner_dc_problem(c)
    H_p = einsum("ac,bc->ba", L, p).min(dim=1)
    q, Lt = fold(p, L, i, j)
    H_q = einsum("ac,bc->ba", Lt, q).min(dim=1)
    M[i, j] = H_q - H_p
i_fold, j_fold = argmin2d(M)
```

Note that if $|\mathbb{A}| = 1$, $L \in \mathbb{R}^{1\times C}$ is a vector and the inner problem is the linear program $\max_{\boldsymbol{p}\in\Delta^C}[\tilde{L}q - Lp]$.

4.3 Vertex Objective

Not all points on the simplex carry the same information for ORIGAMI. Due to concavity, H-entropy is always minimized at a vertex of the simplex:

Proposition 3 (H–entropy is minimized on vertices)**.** *The minimizer* $\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}\in\Delta^C} H_\ell(\boldsymbol{p})$ *is a vertex of* Δ^C.

Therefore, we may wish to focus on the regions of the simplex corresponding to confident (peaked) predictions of the upstream model $p_\theta(z|x)$ i.e., close to the vertices. We propose an objective for ORIGAMI where folding indices are obtained after comparing the H-entropy at all vertices:

$$\mathcal{L} = |H_\ell(\boldsymbol{p}^{(i)}) - H_\ell(\boldsymbol{p}^{(j)})| = |\min_a(\boldsymbol{L}_i) - \min_a(\boldsymbol{L}_j)| \tag{6}$$

where $\boldsymbol{p}^{(i)}$ and $\boldsymbol{p}^{(j)}$ are in the vertex set of the simplex. The vertex loss can be computed efficiently in $\Theta(|\mathbb{A}|C^2)$. In particular, it does not require updating L for each pair i, j: the decision matrix is updated to $\tilde{L}$ only after optimal pair i^*, j^* is found, in contrast to integral and max-increase objectives.

```
# Fold with vertex objective.
# Input: Δ^c, L.
M = zeros(c, c) #
M = M + 10^4 # large initial distance
for (i, j) in combinations(range(c), 2):
    p, q = one_hot(i, c), one_hot(j, c)
    H_p = einsum("ac,bc->ba", L, p).min(dim=1)
    H_q = einsum("ac,bc->ba", L, q).min(dim=1)
    M[i, j] = H_q - H_p
i_fold, j_fold = argmin2d(M)
```

Setting a stopping condition ORIGAMI iterations may be stopped after a predetermined number of folds, or alternatively after the total suboptimality gap δ reaches a tolerance threshold. Interestingly, other ORIGAMI runs may also be recursively initialized within each set in the output partition of the first run, yielding a hierarchical tree-of-sets abstraction of $\mathbb{Z}$.

5 NUMERICAL EXPERIMENTS

We now showcase how ORIGAMI and set abstractions can be used in different learning contexts. The goal is to validate the scalability of ORIGAMI to settings with thousands of outcomes, and to investigate whether abstractions improve downstream policies. If not specified, we use ORIGAMI with the vertex objective.

5.1 Folding for Decision Problems

We evaluate support folding and ORIGAMI in decision-making pipelines as a way to improve downstream policies. We consider wildfire management (Jain et al., 2020), and seek, in the frame of the definition of the problem, to identify a policy to minimize the damage caused by a wildfire at a given location.

Experimental details We design and construct a new wildfire dataset named FIRE! that contains information on active fires from *Visible Infrared Imaging Radiometer Suite* (VIIRS), as well as climate (Hersbach et al., 2020), vegetation, and topographic information (Rollins, 2009).

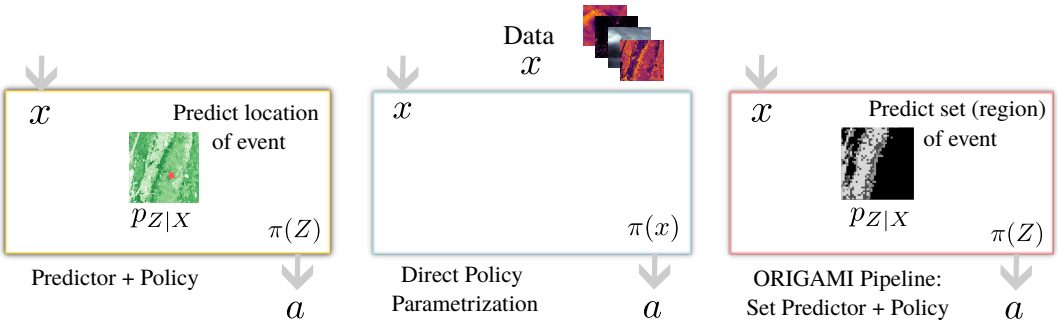

Figure 4.1: Applying ORIGAMI to wildfire management, from prediction to action. **[Left]** Location predict uses a location-level event predictor, then picks the best action that the predicted location **[Middle]** policy directly parameterizes the distribution over actions **[Right]** ORIGAMI uses a region predictor, then picks the best action in the predicted region.

FIRE! includes 1.3 million fire instances collected over the years 2020 and 2021. We focus on a region in California. The overall dataset contains 29 features, spanning climate and climate variables such as temperature and wind speeds, vegetation types and the radiative power of a given wildfire at each location. We aggregate temporal data in weekly periods, resulting with 102 weeks between 2020 and 2021. In this case, the variable Z indicates a geographical location, and we consider 1600 possible locations (a discretized 40 by 40 grid). Additional details on the FIRE! dataset are provided in the Appendix.

Predictive task Each predictive model takes as input a snapshot x (1 week, aggregated as described above) and is tasked with predicting whether the largest wildfire will occur in that particular location in the next week. Given a prediction, a policy picks among three wildfire management strategies: (1) sending a land team to actively suppress the fire, (2) sending aircraft, or (3) applying an indirect approach to slow down the spread (Group, 1996). For our example challenge problem, we craft a decision loss based on insights provided by (Group, 1996), where each strategy is weighted depending on various factors. For instance, sending a land team in regions with high altitudes and slopes might incur larger losses due to challenging terrain.[2]

We formulate three different decision-making pipelines: Direct policy parametrizes directly the distribution over actions, given x; Location predict introduces a location predictor $p_\theta(z|x)$ trained on historical data, and a downstream policy $\pi(x) = \arg\min_a \mathbb{E}_{p_\theta(z|x)}\ell(Z, a)$. ORIGAMI is equivalent to *Location predict* except the model $q_\theta(\mathbb{Z}|x)$ is trained on sets generated by folding geographical locations. The policy in this case involves computing the Bayes optimal action in each location $z \in \mathbb{Z}$ of the predicted partition, then keeping the one most frequently optimal. Fig.4.1 provides an overview of different approaches. Models p_θ and q_θ are parametrized as UNets (Zhou et al., 2019).

[2]We note that our choice of decision loss serves as a proxy for expert decision losses and is not meant to be optimal or take into account every available factor.

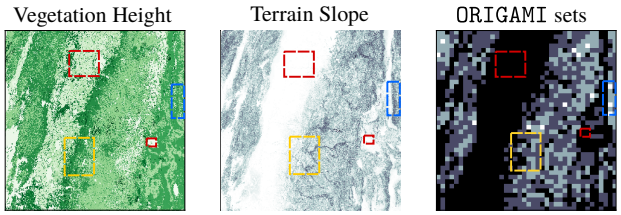

Figure 5.1: **[Left, Center]**: two features of the wildfire management dataset. **[Right]**: sets (regions) output of ORIGAMI given a decision loss based on different features, including vegetation height and terrain slope. As is visible, the sets share common characteristics that are indicative of the features on the left e.g., red highlights regions of no slope and low vegetation, whereas blue highlights areas of tall vegetation.

Pipeline	Predict acc. ↑	Decision loss ↓
Random action	N/A	0.820
Direct policy	N/A	0.731
Location predict	0	0.723
ORIGAMI (5)	**67.4**	0.707
ORIGAMI (10)	54.2	**0.701**

Table 5.1: Benchmarking ORIGAMI on wildfire management. Predicting sets of regions by quantizing space via ORIGAMI folding induces higher quality downstream policy (as measured by the decision loss ℓ). in ORIGAMI (n), n refers to number of sets left at termination of the algorithm (we perform $C - n$ folds).

Results Summary results are provided in Table 5.1. We observe policies based on predictions over ORIGAMI sets to achieve lower decision losses on our test data. Notably, *Location predict* fails to correctly predict any wildfire location during testing, suggesting generalization at the fine-grained scale with the amount of data available is not possible. Predicting ORIGAMI set membership (5 and 10 sets) reaches a considerably higher accuracy. Fig.5.1 provides an example of the sets produced by ORIGAMI: the regions (in shades of grey) are indicative of features the decision loss is based on.

Acquisition	All classes	bot-50	bot-20
Random	17.6%	6.5%	2.8%
Worst-1	30.3%	12.4%	6.1%
Worst-3	30.4%	14.3%	7.0%
ORIGAMI	**35.7%**	**19.4%**	**10.6%**

Table 5.2: Performance of different acquisition methods in the deep active learning experiments. We report average test accuracy across: all 100 classes, worst 20 classes, and worst 50 classes. Worst-classes are identified by ordering based on marginal test accuracy.

5.2 Active Learning

We apply ORIGAMI to large neural network supervised training with limited data. In particular, we use the average H-entropy of sets generated by folding the simplex of classes as guidance to acquire additional data. The *active learning* setting typically involves two interleaved stages: a training stage, where the network is optimized given available data, and an *acquisition* stage, where a new batch is acquired[3].

Experimental details In each run, we optimize an ensemble of 3 ViT (Dosovitskiy et al., 2020) models for image classification on the standard CIFAR100 dataset. We start with a single batch of images, and each epoch we extend the dataset with an additional batch of 128 images constructed following a particular procedure. We compare three different acquisition strategies: (1) random, in which we sample a new batch of images uniformly from all classes, (2) worst-n class, which constructs a new sample of images from the n classes with lowest marginal accuracy (3) ORIGAMI, where we sample uniformly from the top set in the partition generated by ORIGAMI, ranked by highest average H-entropy. To build ORIGAMI sets, we use a decision loss where each model is an action, such that $|\mathbb{A}| = 3$, $C = 100$, and each entry in L is the average loss of each model on all instances of a given class.

Results We provide results in Table 5.2. With 100 epochs and training and a total dataset of $\approx$ 10k images, we reach 35.7% accuracy when ORIGAMI is used as the acquisition method. We observe a quick drop off when inspecting test performance on the worst classes ordered by marginal accuracy, with ORIGAMI having an overall higher worst-case accuracy. Sampling according to highest H-entropy ensures marginal accuracy across classes is balanced, with new data acquired for classes on which the ensemble is struggling.

[3]See (Wang et al., 2016) for other acquisition strategies in deep active learning.

5.3 Amortized ORIGAMI

The vertex objective introduced in §4.3 considerably improves the computation cost of obtaining good abstractions by means of iterative folding when compared to the other methods discussed. An alternative solution is to instead amortize the cost of computing the Bayes optimal objective (3) by pre-training a neural network approximator to match it on a dataset of loss matrices. In the following, we discuss preliminary results and observations, emerging from training a simple neural network to fit the map $\boldsymbol{L} \to \mathcal{L}(\boldsymbol{L}, i, j)$ on synthetic loss matrices $\boldsymbol{L}$.

With $\mathbb{S}(C)$ the space of all $C \times C$ upper triangular matrices ($\mathbb{S}(C) \equiv \mathbb{R}^{C(C-1)/2}$), we define the neural network $a_\theta^C : \mathbb{R}^{|\mathbb{A}| \times C} \to \mathbb{S}(C)$ with parameters θ.

We perform training by providing supervision to the model in the form of tuples $(\boldsymbol{L}, \boldsymbol{M}_{ij})$, where the $\boldsymbol{M}_{ij}$ are produced *offline* by the Monte Carlo approximation of the integral objective feeding $\boldsymbol{L} \sim p(\boldsymbol{L})$ with $p(\boldsymbol{L}) = \mathcal{U}([0,1]^{|\mathbb{A}| \times C})$.

The neural network parameters are then optimized via standard gradient methods to minimize a relative mean-square error objective between the model's predictions and target Bayes optimal folding costs. Such a model can be then invoked during iterative folding as a surrogate for other ORIGAMI variants. Time and compute resources to build a dataset and train the model are thus traded for speedups at inference time when fast evaluation of the folding algorithm is prioritized.

Experimental Details We test the amortized procedure on a dataset of 10^4 uniformly sampled loss matrices $\boldsymbol{L}$. The number of actions $|\mathbb{A}|$ is fixed to 2 while C ranges from 3 (the minimum significant number of classes) to 64. This choice is due to the fact that all ORIGAMI algorithms scale linearly with the number of actions and we are mainly interested in amortizing the quadratic scaling with C. The ground-truth integral folding objectives in the form of the upper triangular matrices $\boldsymbol{M}_{ij}$ have then computed with the Monte Carlo procedure detailed in §4.1 using $N = 10^3$ particles. The neural network a_θ^C comprises four layers with 64 neurons each. The loss matrices are flattened and passed to a_θ^C which returns vectors of dimension $C(C-1)/2$, corresponding to the predicted non-zero entries of $\boldsymbol{M}$.

Results The model, trained for 500 epochs for all values of C is evaluated via a test set of 10^3 additional tuples $(\boldsymbol{L}, \boldsymbol{M})$ in terms of RMSE loss and accuracy in predicting the optimal folding indices. We observe that the prediction accuracy rapidly decreases with the number of classes C while the test RMSE loss increases, as reported by Fig. 5.2. This indicates that the learning problem becomes increasingly difficult with C and the ORIGAMI folding cannot be

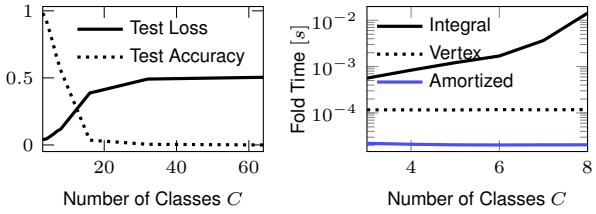

Figure 5.2: **[Left]**: Test RMSE loss between the output of $a_\theta^C(\boldsymbol{L})$ and $\boldsymbol{M}(\boldsymbol{L})$; accuracy in recovering (i^*, j^*) using a_θ^C as surrogate objective for different numbers of classes. **[Right]**: Average (CPU) time required to obtain the optimal folding indexes (i^*, j^*) for the integral (Monte Carlo), vertex and amortized ORIGAMI.

amortized by a simple neural architecture. Nonetheless, in the region where the amortized model is accurate, i.e. for $C < 8$, we report a significant speedup (several orders of magnitude) compared to ORIGAMI equipped with integral and vertex objectives.

6 RELATED WORK

Multiple studies have taken a utility-theoretic perspective on learning and inference. The utility structure of problems has been leveraged in procedures for formulating abstractions of classes and actions as disjunctions (Horvitz and Klein, 1993). A decision-making perspective has also been used to guide abstraction for simplifying probabilistic inference (Poh et al., 1994). Work includes efforts to drive the heterogeneous costs of misclassification into the objective functions and machine learning training procedures (Bach et al., 2006b). Recent work has explored the end-to-end consideration of the quality of decisions in combinatorial optimization (Wilder et al., 2019) and in human-AI collaboration (Wilder et al., 2020). Dubois et al. (2021) propose to leverage knowledge of downstream tasks for compression, improving compression rates over task-agnostic methods. Zhao et al. (2021) formalize a new family of divergences, where discrepancy between distributions is measured through the optimal decision loss induced by each. H-entropy has seen use in Bayesian optimization (Neiswanger et al., 2022), where a new family of acquisition functions is developed.

7 DISCUSSION

We identify and outline several extensions related to the introduction of ORIGAMI algorithms in dynamic decision-making problems and numerical simulation.

Dynamic ORIGAMI We have so far discussed static abstractions synthesized by ORIGAMI as a fixed set of sets of outcomes. However, as decision losses can change in time e.g., if decision matrix L_t has an explicit dependence on time, the abstractions should track these new preferences. This can take place by applying a modified ORIGAMI algorithm able to unfold and fold, instead of starting anew each time.

Folding for simulation The process of quantization and creation of abstractions via ORIGAMI can be loosely connected to meshing and discretization techniques ubiquitous in graphics and numerical simulation of differential equations (Plewa et al., 2005). Instead of standard metrics to guide discretization, ORIGAMI is driven by utilities and is not constrained to sets that are local in space or time. As shown in our experiments, geographical regions found via ORIGAMI can involve disjoint subregions. Locality can be enforced or promoted via minimal changes to the method.

8 CONCLUSION

We presented methods that guide the formulation of abstractions to simplify learning problems based on a careful consideration of downstream decisions. The distillation of abstractions enables data-efficient learning of predictive models. We derive a class of iterative algorithms we refer to as ORIGAMI that work to reduce the dimensionality of the probability simplex while preserving information useful for downstream decisions. In doing so, the method progressively hides information that is not necessary to implement optimal policies, allowing predictive models to learn over sets rather than fine-grained outcomes without loss in decision quality.

References

R. Azuma, M. Daily, and C. Furmanski. A review of time critical decision making models and human cognitive processes. In *2006 IEEE aerospace conference*, pages 9–pp. IEEE, 2006.

F. R. Bach, D. Heckerman, and E. Horvitz. Considering cost asymmetry in learning classifiers. *The Journal of Machine Learning Research*, 7:1713–1741, 2006a.

F. R. Bach, D. Heckerman, and E. Horvitz. Considering cost asymmetry in learning classifiers. *The Journal of Machine Learning Research*, 7:1713–1741, 2006b.

R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

M. H. DeGroot. Uncertainty, information, and sequential experiments. *The Annals of Mathematical Statistics*, 33 (2):404–419, 1962.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16

words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Y. Dubois, B. Bloem-Reddy, K. Ullrich, and C. J. Maddison. Lossy compression for lossless prediction. *Advances in Neural Information Processing Systems*, 34: 14014–14028, 2021.

N. W. C. Group. Wildland fire suppression tactics reference guide, 1996.

P. Hartman. On functions representable as a difference of convex functions. *Pacific Journal of Mathematics*, 9(3): 707–713, 1959.

H. Hersbach, B. Bell, P. Berrisford, S. Hirahara, A. Horányi, J. Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, D. Schepers, et al. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

M. K. Ho. The value of abstraction. *Current opinion in behavioral sciences*, 29, 2019.

E. Horvitz and A. C. Klein. Utility-based abstraction and categorization. In *Proceedings of Uncertainty in Artificial Intelligence*, pages 128–135, 1993.

E. J. Horvitz and M. Barry. Display of information for time-critical decision making. In *Proceedings of Uncertainty in Artificial Intelligence*, pages 296—-305, 1995.

P. Jain, S. C. Coogan, S. G. Subramanian, M. Crowley, S. Taylor, and M. D. Flannigan. A review of machine learning applications in wildfire science and management. *Environmental Reviews*, 28(4):478–505, 2020.

A. Kapoor and E. Horvitz. Breaking boundaries: Active information acquisition across learning and diagnosis. *Advances in neural information processing systems*, 2009.

C. Lindig-León, S. Gottwald, and D. A. Braun. Analyzing abstraction and hierarchical decision-making in absolute identification by information-theoretic bounded rationality. *Frontiers in neuroscience*, 13:1230, 2019.

T. Lipp and S. Boyd. Variations and extension of the convex–concave procedure. *Optimization and Engineering*, 17(2):263–287, 2016.

G. Litjens, F. Ciompi, and J. van der Laak. A decade of gigascience: The challenges of gigapixel pathology images. *GigaScience*, 11, 2022.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

W. Neiswanger, L. Yu, S. Zhao, C. Meng, and S. Ermon. Generalizing bayesian optimization with decision-theoretic entropies. *arXiv preprint arXiv:2210.01383*, 2022.

T. Plewa, T. Linde, V. G. Weirs, et al. Adaptive mesh refinement-theory and applications. 2005.

K. L. Poh, M. Fehling, and E. Horvitz. Dynamic construction and refinement of utility-based categorization models. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(11):1653–1663, 1994.

M. G. Rollins. Landfire: a nationally consistent vegetation, wildland fire, and fuel assessment. *International Journal of Wildland Fire*, 18(3):235–249, 2009.

K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, 2016.

R. Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

B. Wilder, B. N. Dilkina, and M. Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *AAAI*, 2019.

B. Wilder, E. Horvitz, and E. Kamar. Learning to complement humans. *arXiv preprint arXiv:2005.00582*, 2020.

S. Zhao, A. Sinha, Y. He, A. Perreault, J. Song, and S. Ermon. Comparing distributions by measuring differences that affect decision making. In *International Conference on Learning Representations*, 2021.

D. Zheng, C. Ma, M. Wang, J. Zhou, Q. Su, X. Song, Q. Gan, Z. Zhang, and G. Karypis. Distdgl: distributed graph neural network training for billion-scale graphs. In *2020 IEEE/ACM 10th Workshop on Irregular Applications: Architectures and Algorithms (IA3)*, pages 36–44. IEEE, 2020.

Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: Redesigning skip connections to exploit multi-scale features in image segmentation. *IEEE transactions on medical imaging*, 39(6):1856–1867, 2019.

Ideal Abstractions for Decision-Focused Learning

Table of Contents

Notation We report here a reference for notation used in main text and supplementary.

Symbol	Description
$\mathbb{R}$	Set of reals
Δ^C	C-dimensional probability simplex
$\mathcal{U}$	Uniform distribution
$\mathbb{E}[x]$	Expected value of random variable x
$\circ$	Composition of functions $f \circ g(x) = f(g(x))$

A Derivations

Proposition 4 (Folding increases H-entropy)**.** *Let $\boldsymbol{p} \in \Delta^C$ and $\boldsymbol{q} = f \circ f \cdots \circ f(\boldsymbol{p})$ be any sequence of folds. Then,*

$$H_\ell(\boldsymbol{p}) \leq H_{\tilde{\ell}}(\boldsymbol{q}).$$

Proof. We show the result for each row $k = 1, \ldots, C$ of $\boldsymbol{L}\boldsymbol{p}$ and a single fold:

$$\begin{aligned}
\sum_{m=1}^{C} \boldsymbol{L}_m \boldsymbol{p}_m &\leq \sum_{n=1}^{C-1} \tilde{\boldsymbol{L}}_n \boldsymbol{q}_n \Leftrightarrow \\
\boldsymbol{L}_{ki}\boldsymbol{p}_i + \boldsymbol{L}_{kj}\boldsymbol{p}_j &\leq \max\{\boldsymbol{L}_{ki}, \boldsymbol{L}_{kj}\}\boldsymbol{q}_j \Leftrightarrow \\
\boldsymbol{L}_{ki}\boldsymbol{p}_i + \boldsymbol{L}_{kj}\boldsymbol{p}_j &\leq \max\{\boldsymbol{L}_{ki}, \boldsymbol{L}_{kj}\}\boldsymbol{p}_i + \\
&\quad \max\{\boldsymbol{L}_{ki}, \boldsymbol{L}_{kj}\}\boldsymbol{p}_j.
\end{aligned}$$

□

Proposition 5 (H–entropy is minimized on vertices)**.** *The minimizer*

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p} \in \Delta^C} H_\ell(\boldsymbol{p})$$

is a vertex of Δ^C

Proof. Let $\mathbb{V}_\Delta$ be the set of vertices of the simplex, i.e. the canonical basis $\mathbf{e}_1, \ldots, \mathbf{e}_C$ of $\mathbb{R}^C$. We need to show that $\min_{\boldsymbol{p} \in \Delta^C} H_\ell(\boldsymbol{p}) = \min_{i=1,\ldots,C} H_\ell(\mathbf{e}_i)$. Due to convexity of the simplex Δ^C, the minimizer $\boldsymbol{p}^*$ can be expresses as a convex combination of the vertices, i.e.

$$\boldsymbol{p}^* = \sum_{i=1}^{C} \alpha_i \mathbf{e}_i, \quad \alpha_i \geq 0, \ \sum_{i=1}^{C} \alpha_i = 1.$$

By Jensen's inequality we have

$$H_\ell(\boldsymbol{p}^*) = \inf_a \boldsymbol{L}\boldsymbol{p}^* = \inf_a \boldsymbol{L} \sum_{i=1}^{C} \alpha_i \mathbf{e}_i \geq \sum_{i=1}^{C} \alpha_i \inf_a \boldsymbol{L}\mathbf{e}_i$$

and

$$\sum_{i=1}^{C} \alpha_i \inf_a \boldsymbol{L}\mathbf{e}_i \geq \min_{i=1,\ldots,C} \inf_a \boldsymbol{L}\mathbf{e}_i$$

so the minimum of H_ℓ over $\boldsymbol{p} \in \Delta^C$ is bounded below by the minimum over the vertices. Since the vertices belong to Δ^C, the result is proved. □

B Experiments

B.1 Folding for Decision Problems

Dataset curation We design and build a new dataset named `FIRE!` that contains active fire information from *Visible Infrared Imaging Radiometer Suite* (VIIRS) on a spatial resolution of 375 meters, as well as climate and vegetation data.

Fires: We consider $1,339,234$ fire instances collected over the years 2020 and 2021. Each instance contains fire radiative power, location (latitude and longitude) and auxiliary information such as time of day and confidence for the measurement. We select the region spanned by latitude $(36, 39)$ and longitude $(-121.6, -118.6)$.

Vegetation: We collect data from the LANDFIRE program. In particular, we add the following features: existing vegetation height (EVH), existing vegetation cover (EVC), existing vegetation type (EVT), slope degrees (SlpD), slope percent rise (slpP), roads, aspect (Asp). As these databases are updates at lower frequencies than VIIRs and climate, we have access to 2019 and 2020 snapshots which we use as additional context for the model. The region is aligned with VIIRS spatial coordinates.

Climate: We extract a set of climate and weather features from the large-scale ERA5 dataset. Weather and climate variables describe a larger region than latitude $(36, 39)$ and longitude $(-121.6, -118.6)$ to provide context for the predictive model. All data slices are aligned in time.

Predictive task The model takes as input a snapshot x (1 week, aggregated as described above) and is tasked with predicting the location of the largest wildfire (measured in radiative power) in the following week. When using `ORIGAMI`, spatial locations are clustered according to the decision loss, and thus the dimension of y is smaller than the dimension of x. We optimize the parameters of all models using a standard binary cross entropy loss.

Decision making and decision loss We design a simple, deterministic closed-loop policy reliant on predictions made by a deep learning model. Our goal is to investigate whether the quality of a decision policy can be improved by performing upstream prediction on a "simplified" space of locations found as a decision–optimal clustering with `ORIGAMI`.

Wildfire management actions are: (1) *land intervention*, (2) *aircraft intervention* (3) *indirect containment*, according to the location predicted by the wildfire location model. The decision loss is crafted according to insights extracted from (Group, 1996). The following factors are used: fire radiative power, existing vegetation height, roads, temperature, magnitude of wind. In particular, action (1) incurs in high cost when slope and terrain height are larger, (2) when wind is strong, and (3) when vegetation is dense. Since all features are on different scales, we normalized the decision loss to obtain values in a comparable range. We note that the decision loss ℓ is not meant to encode *all* factors one may want to consider for wildfire management, as the experiment is meant to showcase potential applications of `ORIGAMI`.

Training details We train the wildfire location predictive model for 100 epochs using the AdamW optimizer (Loshchilov and Hutter, 2017), with a learning rate of 0.001 and a cosine decay schedule to 0.0001. We set the batch size to 8 (each element of the batch contains a temporal slice of context data, with the model asked to predict the location of the largest wildfire in the following week). All models are standard `ResNets` with 18 layers.

When evaluating the policy, we pick the Bayes optimal wildfire strategy between: (1) *land intervention*, (2) *aircraft intervention* (3) *indirect containment*, according to the location predicted by the wildfire location model.

B.2 Active Learning

Training details We train ensembles of 3 vision transformers `ViT` (Dosovitskiy et al., 2020) on the CIFAR100 dataset, starting from a single random batch of data. Each epoch, we increase dataset size by sampling a new batch based on different acquisition methods:

- *random*: a new batch of images is obtained by sampling uniformly from all 100 classes
- worst-n: the new batch is obtained by sampling data uniformly if the corresponding label belongs to the n classes with lowest marginal accuracy
- `ORIGAMI`: we apply `ORIGAMI` to generate a partition of all classes. We select the set of classes in the partition with highest average H-entropy, and sample uniformly.

We use `ViT`-base from the `timm` library (Wightman, 2019) with patch size 16 and latent dimension 224, and add the logits of each model in the ensemble before computing the cross-entropy loss. We train all models 100 epochs using AdamW optimizer (Loshchilov and Hutter, 2017), a learning rate schedule with 20 epochs of linear warmup (0.0001 to 0.001) followed by cosine decay down to 0.0001. We use batch size 128.