



CHALMERS
UNIVERSITY OF TECHNOLOGY



UNIVERSITY OF GOTHENBURG

Deep-EC2: A Non-Myopic Active Feature Acquisition Framework

Master's thesis in Computer science – algorithms, languages and logic

Haozhe Sun

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

MASTER'S THESIS 2026

Deep-EC2: A Non-Myopic Active Feature Acquisition Framework

Haozhe Sun



UNIVERSITY OF
GOTHENBURG



CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Deep-EC2: A Non-Myopic Active Feature Acquisition Framework
Haozhe Sun

© Haozhe Sun, 2026.

Supervisor: Linus Aronsson, Department of Computer Science and Engineering
Examiner: Morteza Haghir Chehreghani, Department of Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in L^AT_EX
Gothenburg, Sweden 2026

Abstract

Active Feature Acquisition (AFA) addresses the problem of sequentially deciding which features to observe in order to maximise predictive accuracy while minimising acquisition cost. Non-myopic methods, which account for how an acquisition changes the value of later ones, typically rely on reinforcement learning (RL), with its reward design, rollouts, and hyperparameter sweeps.

This thesis proposes *Deep-EC2*, an AFA framework that replaces reinforcement learning with supervised regression onto analytical targets derived from the Equivalence Class Edge Cutting (EC2) criterion, the only AFA objective known to be adaptive submodular. A value network learns to rank candidate features by their expected EC2 gain, estimated with a masked k -nearest-neighbour estimator over the training set, and a third training phase adapts the predictor to the subsets that the learned policy acquires. At inference, acquisition requires only greedy forward passes, with no rollout or planning.

Experiments on eleven benchmarks—three synthetic gated tasks, the CUBE-NM context-feature benchmark, and seven real-world tabular datasets—compare Deep-EC2 with seven published AFA methods, the RL method Opportunistic Learning (OL), and a random reference, under identical splits, three seeds, and seed-paired significance tests. On CUBE-NM, Deep-EC2 acquires a context feature first on every instance, significantly outperforms every baseline including OL, and reaches perfect accuracy after 7–8 acquisitions. On the synthetic task Syn3 it acquires the gate first on every instance and is on par with the strongest non-RL baselines. On both benchmarks a budget-by-budget analysis reveals an *invest-then-exploit* signature—an early deficit incurred by acquiring an individually uninformative context feature or gate, repaid by significant mid-budget gains—the hallmark of non-myopic acquisition. On the real-world tabular data Deep-EC2 remains competitive, and adapting the predictor to the policy improves all eleven benchmarks. Training a model takes 8–18 minutes on the synthetic benchmarks, faster than OL on each of them and about six to eighty times cheaper once OL’s hyperparameter sweep is counted.

Overall, the thesis shows that non-myopic acquisition can be obtained from a supervised value network trained on EC2 targets, at myopic inference cost and without reinforcement learning.

Keywords: active feature acquisition, non-myopic planning, EC2, value network, k -nearest neighbours

Acknowledgements

First and foremost, I would like to thank my supervisor, Linus Aronsson, for the guidance throughout this project. The patience, the careful feedback on both the method and the writing, and the willingness to discuss every half-finished idea have shaped this thesis in many ways, and I have learned a great deal from our work together.

I am also grateful to my examiner, Morteza Haghiri Chehreghani, for making this project possible, for valuable comments along the way, and for the time spent evaluating this work.

Finally, a special thanks to the four members of our family who contributed nothing to the methodology but a great deal to my well-being: our Border Collie Ganmaoling, our Chinese rural dog Banlangen, and our two Chinese rural cats Sanjiu and Keli. Named after cold remedies, they have proven to be the most effective cure for a stressful day.

Haozhe Sun, Gothenburg, 2026-10-20

Contents

List of Figures	xi
List of Tables	xiii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Current Paradigm and Problem Statement	1
1.3 Proposed Method	3
1.4 Thesis Outline	3
2 Related Work	5
2.1 Active Feature Acquisition	5
2.2 Myopic Methods in AFA	5
2.3 Non-Myopic Methods and Reinforcement Learning	6
2.3.1 AFA as an MDP and Computational Intractability	6
2.4 Equivalence Class Edge Cutting (EC2) and kNN-Based Surrogates	7
3 Theoretical Foundations	11
3.1 Active Feature Acquisition under a Hard Budget	11
3.1.1 Predictor under Partial Observability	11
3.1.2 Acquisition Process	12
3.1.3 Optimisation Objective	12
3.2 The Equivalence Class Edge Cutting (EC2) Objective	12
3.2.1 Decision Regions, the Version Space, and the Conflict Graph	12
3.2.2 The Acquisition Objective	13
3.2.3 The Combined Limitations of CMI and GBS	14
3.3 k -Nearest-Neighbour Estimation	15
4 The Deep-EC2 Framework	17
4.1 The EC2 Objective in Closed Form	17
4.1.1 Remaining Edge Weight and the Gini Form	17
4.1.2 The Realizable Case	18
4.1.3 One-Step Marginal Gain	19
4.1.4 Adaptive Monotonicity and Adaptive Submodularity	19
4.1.5 Comparison of Acquisition Objectives	20
4.2 Partial Observation Encoding	21

4.2.1	Predictor Pretraining under Random Masks	21
4.3	EC2 Target Computation	22
4.3.1	Threshold Binarization and the Cubic Target	23
4.3.2	The Initial State	24
4.4	Value Network Training and the Acquisition Policy	25
4.4.1	Supervised Regression against kNN Targets	25
4.5	Phase 3: Adapting the Predictor to the Policy	26
4.6	Inference and the Complete Pipeline	27
5	Experiments	29
5.1	Experimental Setup	29
5.1.1	Benchmark Datasets	29
5.1.2	Feature Groups for Categorical Attributes	30
5.1.3	Baselines	31
5.1.4	Evaluation Protocol and Metrics	31
5.1.5	Hyperparameters and Implementation	32
5.2	CUBE-NM: Context-First Acquisition Beats All Baselines	33
5.3	Syn1–Syn3: Gate-First Acquisition on Syn3	34
5.4	Real-World Benchmarks	37
5.5	Ablation: Predictor Adaptation	39
5.6	Training Cost, Robustness, and Limitations	40
6	Conclusion and Future Work	43
6.1	Conclusions	43
6.2	Limitations	44
6.3	Future Work	44
	Bibliography	45

List of Figures

5.1	Feature-selection frequency per acquisition step on CUBE-NM (seed 0). The first acquisition is a context feature on every instance, and the first steps are dominated by context features (features 0–4). . . .	34
5.2	Test accuracy versus acquisition budget on CUBE-NM (three-seed mean $\pm$ seed spread).	34
5.3	Test accuracy versus acquisition budget on the synthetic benchmarks (three-seed mean $\pm$ seed spread).	35
5.4	Feature-selection frequency per acquisition step on Syn3 (three seeds aggregated), overall and split by the active branch. The gate x_{11} is acquired first on every instance, after which the policy turns to the features of the active branch.	35
5.5	Accuracy difference between Deep-EC2 and each baseline as a function of the budget (three-seed means). Thin lines: Deep-EC2 minus the named baseline; thick line: Deep-EC2 minus the strongest baseline at each budget; filled markers: the difference to all baselines is significant (paired t -test, $p < 0.05$).	37
5.6	Test accuracy versus acquisition budget on the seven real-world datasets (three-seed mean $\pm$ seed spread).	38
5.7	Test AUROC versus acquisition budget on the seven real-world datasets (three-seed mean $\pm$ seed spread).	39

List of Tables

5.1	Summary of the benchmark datasets. “Dim.” is the acquisition dimension (for Bank Marketing, the number of one-hot columns and, in parentheses, the number of acquisition groups after grouping categorical dummy columns, Section 5.1.2). “Metric” lists the metrics reported at every budget on the test split.	30
5.2	Hyperparameters of Deep-EC2 and dataset-specific deviations. . . .	33
5.5	Accuracy difference (Deep-EC2 – baseline) at budgets 1–8 on CUBE-NM and Syn3, three-seed means. Superscripts */** mark differences that are significant at $p < 0.05/p < 0.01$ (paired t -test, 2 d.f.). Negative values at the first budgets reflect the acquisition of the context feature or the gate; the positive values that follow reflect its exploitation.	36
5.6	Ablation of the predictor adaptation (accuracy–AUC, mean $\pm$ sample standard deviation over three seeds). Δ is the gain of the complete method over the two-phase variant. The last columns give the best baseline mean and the number of the eight baselines (seven published methods and, on the synthetic benchmarks, OL) that the complete method is significantly better / worse than (paired t -test, $p < 0.05$); bold marks a value above the best baseline.	40
5.3	Normalised area under the budget–accuracy curve (Equation (5.2)), mean $\pm$ sample standard deviation over the three seeds. Bold marks the best mean per row. A superscript */** on a baseline cell indicates that Deep-EC2 is significantly better than that method (paired t -test, $p < 0.05/p < 0.01$, 2 d.f.); a subscript +/++ indicates that Deep-EC2 is significantly worse. OL was run on the synthetic benchmarks only.	42
5.4	Normalised area under the budget–AUROC curve, same conventions as Table 5.3. CUBE-NM is omitted because our evaluation pipeline logs accuracy only for this benchmark.	42

1

Introduction

1.1 Background and Motivation

In many real-world decision problems, collecting data is costly, time consuming, and sometimes even risky. Consider healthcare: a treatment plan may be decided based on a small subset of medical tests, and performing every available test on every patient would be infeasible in terms of cost, time, and patient risk. Instead, tests should be selected sequentially, guided by the information gathered so far, acquiring only what is necessary to reach a reliable decision. In this setting, the features of an instance are acquired adaptively, one at a time, and every acquisition incurs a cost. This process is formalised in the machine learning literature as active feature acquisition (AFA) [1]. Its roots reach back to early work on sequential diagnosis [2] and decision-theoretic troubleshooting [3], and it connects directly to classical measures of the information provided by an experiment [4] and to the notion of the *value of information*: an experiment (or feature measurement) is only worth performing if its expected benefit in decision quality outweighs its cost [5], [6]. Within machine learning, AFA extends static feature selection [7] to an adaptive, per-instance setting, and recent benchmarks evaluate it systematically as a paradigm for cost-sensitive prediction [8].

The defining challenge of AFA is the trade-off between *acquisition cost* and *predictive performance*. Acquiring more features generally improves the accuracy of the resulting prediction, but each measurement consumes resources, and different instances may require different features to be predicted reliably. A successful AFA system must therefore decide, for every instance and at every step, whether the expected improvement in prediction quality from the next measurement justifies its cost, or whether it is better to stop and predict using only the features acquired so far. The sequential nature of this process is essential: the informativeness of a feature typically depends on which features have already been observed, so acquisition decisions must be made adaptively, based on the evidence accumulated up to that point.

1.2 Current Paradigm and Problem Statement

Existing AFA methods can be categorised along a fundamental axis: *myopic* methods, which evaluate each candidate feature with a one-step lookahead, versus *non-myopic* methods, which plan over longer acquisition horizons [1]. Myopic methods

score a candidate feature by the expected immediate gain it would deliver, most commonly the conditional mutual information between the candidate feature and the target label given the current partial observation [1]. Representative modern examples include EDDI [9], GDFS [10], and DIME [11], together with a long line of classic cost-sensitive classifiers [12], [13], [14], [15]. Because they require no planning beyond the next step, myopic methods are computationally cheap, and they often perform remarkably well in practice [1], [8]. However, one-step lookahead is fundamentally limited: it cannot account for interactions between unobserved features, where two features that are individually uninformative are jointly decisive, and it generally produces suboptimal acquisition sequences.

Non-myopic methods aim to overcome these limitations by optimising the long-term value of an acquisition strategy. AFA is naturally formalised as a partially observable Markov decision process, the AFA-POMDP [1], in which the agent’s information state records which features have been acquired and their realised values, and the optimal policy balances immediate cost against future information gain. Because the resulting decision problem is a Markov decision process over this information state, it can in principle be solved by reinforcement learning (RL), and indeed the vast majority of non-myopic AFA methods are RL-based, ranging from early approximate policy iteration and fitted-Q approaches [16], [17] to deep RL systems such as Jafa [18], Opportunistic Learning [19], and the deep RL formulation of Janisch et al. [20], [21], as well as model-based RL and search-based planners such as ODIN [22] and MCTS-based acquisition [23].

Despite their principled foundations, the non-myopic paradigm as currently implemented suffers from a severe bottleneck: it is dominated by reinforcement learning. The AFA-POMDP is computationally intractable in general: optimal planning remains hard even when all probabilistic components are known [6], and the state space over feature subsets grows combinatorially with the dimensionality of the data [1]. In practice, RL-based AFA methods inherit the well-known difficulties of deep RL: they require extensive simulated rollouts to estimate long-term returns and stabilisation techniques such as target networks and double Q-learning [21], rely on reward shaping to stabilise training [1], and can take days of training for a single cost setting [21]. These issues make RL-based non-myopic methods difficult to train and deploy efficiently, which limits their practical utility in exactly the cost-sensitive applications for which they are designed.

Problem statement. There is a need for a non-myopic AFA approach that retains the long-term planning capability of RL-based methods while avoiding their prohibitive training complexity and unstable convergence. In particular, the acquisition objective should (i) capture the value of future acquisitions, not just the immediate step, (ii) admit efficient and stable training without the machinery of RL, and (iii) scale to high-dimensional feature spaces.

1.3 Proposed Method

To address this problem, we propose *Deep-EC2*, a hybrid AFA framework that combines the Equivalence Class Edge Cutting (EC2) objective with a k -nearest-neighbour (kNN) approximation, trained as a supervised regression problem rather than through reinforcement learning.

The EC2 objective, introduced in the context of Bayesian active learning [24] and later applied to AFA [25], is the only objective suited to AFA, together with its direct extensions, that is known to be *adaptive submodular* [1], [24]. This property guarantees that a simple greedy acquisition policy, which at each step selects the feature that maximises the expected reduction in total edge weight, achieves a $(1 - 1/e)$ -approximation to the optimal adaptive policy under the EC2 utility (for uniform acquisition costs). EC2 thus allows forward-looking acquisition through a greedy rule, without explicit multi-step planning. Its design is motivated by the combined limitations of the two classical alternatives: the widely used conditional mutual information objective is not adaptive submodular, even under a Naive Bayes assumption, whereas the adaptive submodular generalized binary search objective targets the identification of the entire feature realisation rather than the label, and can therefore be far from optimal for prediction [24], [26].

The remaining challenge is computational: evaluating the EC2 gain requires expectations over the unobserved feature values, which are intractable to compute exactly for continuous, high-dimensional data. Deep-EC2 approximates the conditional distributions needed for the EC2 gain with a kNN Monte Carlo estimator over the training set, following the neighbourhood-based estimation paradigm that has proven effective in several AFA methods, from early margin-based approaches to recent ones [1], [27], [28], [29]. The kNN estimates serve as supervised regression targets for a *value network*: a neural network that, given the currently acquired subset of features, predicts the EC2 gain of every remaining candidate feature. A third training phase completes the framework: after the value network is trained, the predictor is adapted to the subsets that the learned policy acquires. Training is thus a sequence of supervised problems that needs no reward design, rollouts, or return estimates, and at inference the acquisition policy reduces to a sequence of greedy forward passes, as cheap as a myopic rule.

In summary, Deep-EC2 obtains forward-looking acquisition from the structure of the EC2 objective rather than from reward-driven optimisation. Chapter 5 shows that it acquires non-myopically on the CUBE-NM benchmark and on the gated synthetic task Syn3, where it outperforms an RL baseline and is on par with the strongest non-RL baselines respectively, at a fraction of the RL training cost.

1.4 Thesis Outline

The remainder of this thesis is organised as follows. Chapter 2 surveys the AFA literature, Chapter 3 develops the theoretical foundations of the AFA problem, the EC2 objective, and kNN estimation, Chapter 4 presents the Deep-EC2 framework

in detail, Chapter 5 reports the experimental evaluation, and Chapter 6 concludes and discusses directions for future work.

2

Related Work

2.1 Active Feature Acquisition

AFA addresses the problem of sequentially acquiring a subset of an instance’s features so as to optimise the trade-off between predictive performance and acquisition cost [1]. Each acquisition step reveals the value of one feature at a known cost, and the process stops when a prediction is made based on the features gathered so far. The problem is a special case of sequential decision-making under test-time budgets [19], [21], and its study spans both classic cost-sensitive classification [12], [13] and modern learning-based approaches [1].

Existing AFA methods are most commonly categorised along the axis between myopic and non-myopic methods: *myopic* methods select each feature with a one-step lookahead, while *non-myopic* methods account for the long-term consequences of an acquisition decision [1]. Since AFA is naturally formulated as a (partially observable) Markov decision process, a recent taxonomy also organises methods by how they approximate the resulting optimal policy, mirroring standard POMDP solution paradigms: embedded cost-aware predictors (e.g., cost-sensitive decision trees), model-based methods that plan using learned probabilistic components, model-free methods that learn policies directly from simulated episodes, and hybrid methods combining both [1]. Recent benchmark studies have begun to evaluate AFA methods systematically under static feature settings, confirming both the relevance of the field and the considerable variance in performance across methods and datasets [8].

2.2 Myopic Methods in AFA

Myopic methods evaluate each candidate feature by the immediate information gain it is expected to deliver, without considering future acquisition steps. Under logarithmic loss, the optimal one-step rule reduces to selecting the feature with the largest conditional mutual information with the target label, $I(\mathbf{y}; \mathbf{x}_a \mid x_S)$, possibly normalised by the acquisition cost; it is one of the most common selection heuristics in the AFA literature [1]. This criterion has a long history, tracing back to classical information-theoretic and decision-theoretic measures of information and value [4], [5], and to early active-testing and diagnostic models [3], [30].

Within machine learning, myopic AFA encompasses a broad family of methods. Classic cost-sensitive classifiers acquire features greedily within the model itself: cost-sensitive decision trees [12], [13], test-cost-sensitive Naive Bayes classifiers [31], and budget-constrained tree ensembles (boosted trees and random forests) that operate under a test-time budget [14], [15]. Modern myopic methods estimate the required conditional quantities with learned models: EDDI [9] uses a partial variational autoencoder to estimate conditional mutual information, GDFS [10] amortises the greedy CMI policy, training a policy network jointly with the predictor, and DIME [11] estimates conditional mutual information discriminatively. Neighbourhood-based myopic heuristics also exist, such as the fast margin-based method of Nan et al. [27], which scores features by their expected effect on the fraction of positive-margin neighbours.

The appeal of myopic methods lies in their computational efficiency and training stability: no lookahead beyond the immediate step is required, and each selection reduces to scoring a candidate set under the current observation. However, one-step lookahead is inherently limited. Non-myopic planning must account for interactions among unobserved features: two features that are individually uninformative can be jointly decisive, an effect that no myopic rule can capture [1]. Moreover, the common myopic objective $-H(\mathbf{y} \mid x_S)$ is *not* adaptive submodular, even under a Naive Bayes assumption, so no near-optimality guarantee holds for greedy selection in general; near-optimality is only recovered under restrictive assumptions on the data distribution [1], [26]. Consequently, myopic acquisition sequences are typically suboptimal for the long-term trade-off between cost and accuracy.

2.3 Non-Myopic Methods and Reinforcement Learning

Non-myopic methods plan beyond the immediate step, aiming to optimise the long-term value of the acquisition policy. This section develops the MDP view of AFA that motivates them, and then surveys the reinforcement-learning-based methods that currently dominate the non-myopic paradigm.

2.3.1 AFA as an MDP and Computational Intractability

A distinctive property of AFA is that the observed information (S, x_S) , namely the set of acquired features together with their realised values, is a sufficient statistic for the full observation history: it records exactly which features have been acquired and what was observed. Consequently, each (S, x_S) induces a unique posterior belief over the unobserved quantities $(\mathbf{x}_U, \mathbf{y})$, and the optimal acquisition policy can be written as a function of (S, x_S) alone [1]. This defines a fully observable Markov decision process—the *MDP view* of the AFA-POMDP—whose state space is the set $\{(S, x_S) : S \subseteq [n], x_S \in \mathcal{X}_S\}$ and whose value function satisfies the corresponding Bellman recursion [1].

The main difficulty is computational. The number of possible states (S, x_S) grows

combinatorially with the dimensionality n , and becomes effectively uncountable for continuous features, so exact dynamic programming is feasible only for very small problems or under strong structural assumptions. More importantly, even if the probabilistic components required for planning (the predictive distribution and the observation model) were given, optimal planning remains intractable in general: for value-of-information problems in graphical models, of which AFA is a special case, optimal adaptive acquisition is provably intractable except for chain-structured models and a few closely related low-complexity cases [6]. Practical AFA therefore couples two challenges: learning the conditional quantities induced by the data distribution, and approximately solving the resulting MDP [1].

Because the problem is an MDP, reinforcement learning provides a natural way to learn acquisition policies from simulated episodes, without explicitly modelling the transition dynamics. Dulac-Arnold et al. were among the first to make this connection explicit, treating AFA as reward maximisation in the corresponding MDP and learning the policy with approximate policy iteration [16], an idea pursued further with fitted Q-iteration over the class-posterior state representation [17], [32]. Subsequent work extended this line to deep RL, using algorithms such as DQN and PPO [33], [34] within the standard RL framework [35]: JAFA [18] learns a joint acquisition-and-classification model with variable-size set encoders, Opportunistic Learning [19] trains a deep Q-network-style acquisition policy under a per-instance budget, mitigating the sparse terminal reward with a cost-normalised intermediate reward that quantifies the shift in the predictive distribution, and Janisch et al. formulate classification with costly features as a sequential decision-making problem solved with deep RL [20], [21]. Model-based variants plan with learned surrogate models or search: ODIN [22] combines a generative model with model-based deep RL, and Chen et al. apply Monte Carlo tree search to costly-feature classification [23].

The strengths of this paradigm are clear: RL-based methods optimise the long-term reward directly, and can in principle discover acquisition strategies that myopic rules cannot. In practice, however, they inherit the well-known difficulties of RL. Learning from delayed returns requires extensive simulated rollouts and makes credit assignment difficult, and reward shaping is used to stabilise training [1]; deep Q-learning in addition needs stabilisation techniques such as target networks and double Q-learning, and training can take days for a single cost setting [21]. Combined with the combinatorial size of the state space, this makes RL-based AFA notoriously difficult to train and deploy. These limitations motivate acquisition objectives that retain non-myopic behaviour while avoiding the RL machinery, as discussed next.

2.4 Equivalence Class Edge Cutting (EC2) and kNN-Based Surrogates

A promising route toward principled non-myopic acquisition without reinforcement learning is provided by submodular surrogate objectives. The central notion is *adaptive submodularity* [36]: for an adaptive monotone and adaptive submodular utility

function, a greedy policy that at each step selects the action maximising the expected marginal gain achieves a $(1 - 1/e)$ -approximation to the optimal policy with the same budget [36], [37]; for non-uniform costs, the analogous guarantee holds for the gain per unit cost [36]. Since AFA can be viewed as a special case of adaptive stochastic optimisation, any acquisition objective with these two properties immediately yields a greedy acquisition strategy with a formal near-optimality guarantee [1], [36].

The EC2 objective, introduced by Golovin et al. in the context of Bayesian active learning [24], is the only objective suited to AFA, together with its direct extensions, that is known to be adaptive submodular [1]. EC2 assigns each instance to a decision region for every possible label, constructs a graph over instances whose edges connect pairs that lie in different decision regions while remaining consistent with the current partial observation, and weighs each edge by the joint probability of its endpoints. The acquisition objective is the expected reduction in total edge weight: acquiring a feature that cuts many high-weight edges removes much of the remaining disagreement between conflicting hypotheses [1], [24]. This design responds directly to the complementary limitations of the two classical objectives discussed above: information-gain criteria such as CMI are not adaptive submodular due to complementarities among features, whereas generalized binary search (GBS) *is* adaptive submodular but shrinks the version space as a whole, which is not aligned with distinguishing the decision regions and can be linearly suboptimal for that purpose [24]. Because the EC2 objective is adaptive submodular, greedy selection is near-optimal with respect to it, and the resulting policy is non-myopic in the sense that it approximates the multi-step lookahead value [1]. EC2 was subsequently applied to AFA in online decision tree learning [25], to interactive troubleshooting [38], and, in a combinatorial multi-armed bandit variant, to cost-efficient online decision making [39]. One caveat should be noted: EC2 is a *surrogate* objective that may not be fully aligned with the AFA objective, so greedy optimisation of EC2 can in principle be suboptimal for the original trade-off between cost and accuracy [1].

On the computational side, the EC2 gain, like other non-myopic quantities, requires expectations over the unobserved features, which are intractable to compute exactly. A practical solution, increasingly common in recent AFA methods, is *non-parametric, neighbourhood-based estimation*: instead of fitting deep generative models, which are computationally expensive and difficult to train [1], [28], the required conditional distributions are approximated empirically from a k -nearest-neighbour neighbourhood of the current partial observation in the training set. Early work in this direction includes the margin-based kNN method of Nan et al. [27]. More recently, AACO [28] approximates a non-greedy acquisition-conditioned oracle, which scores candidate future feature subsets by their expected loss plus cost, estimating the expectation over unobserved features with kNN, and Information Templates [29] learns a library of jointly informative feature subsets (templates) whose expected loss is evaluated with k -nearest-neighbour estimates, acquiring the cheapest unobserved feature of the best template, and further variants guide acquisition with local feature explanations [40] or with ANOVA F-scores recomputed on a distance-filtered training neighbourhood [41].

The Deep-EC2 framework proposed in this thesis sits at the intersection of these two strands. It adopts the EC2 objective for its formal near-optimality guarantees, computes the required conditional expectations with a kNN Monte Carlo estimator, and trains a value network by supervised regression on the resulting targets, combining the non-myopic guarantees of EC2 with the stability and efficiency of neighbourhood-based estimation, without any reinforcement learning. Chapter 3 develops the underlying theory, and Chapter 4 describes the framework in detail.

3

Theoretical Foundations

This chapter develops the theoretical foundations underlying the proposed Deep-EC2 framework. Section 3.1 formalises the AFA problem under a hard acquisition budget, following the formulation of Aronsson et al. [1]. Section 3.2 presents the EC2 objective that drives feature selection and its near-optimality guarantees. Section 3.3 introduces the kNN estimation technique used to make the EC2 objective computable in practice. The concrete method that combines these components into a trainable value network is described in Chapter 4.

3.1 Active Feature Acquisition under a Hard Budget

Let $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_n$ denote the collective feature space of n dimensions, and let $\mathcal{Y} = \{1, \dots, K\}$ denote the target label space. We assume that data instances $(\mathbf{x}, y)$ are distributed according to an unknown joint distribution $p(\mathbf{x}, y)$. For any index subset of acquired features $S \subseteq [n] \triangleq \{1, \dots, n\}$, we write $\mathbf{x}_S \in \mathcal{X}_S$ for the restriction of $\mathbf{x}$ to the features indexed by S , and $U \triangleq [n] \setminus S$ for the set of remaining unobserved features. Acquiring an unobserved feature $i \in U$ reveals its realised value x_i and incurs a known acquisition cost $c_i \in \mathbb{R}_+$. The total cost of acquiring the features in S is:

$$c(S) \triangleq \sum_{j \in S} c_j. \quad (3.1)$$

3.1.1 Predictor under Partial Observability

A predictor in the AFA setting must be able to classify from *arbitrary* subsets of observed features. It is therefore defined as a mapping from partial observations to labels [1]:

$$f : \{(S, \mathbf{x}_S) \mid S \subseteq [n], \mathbf{x}_S \in \mathcal{X}_S\} \rightarrow \mathcal{Y}. \quad (3.2)$$

In practice, f is typically applied to a fixed-dimensional representation of the partial observation via *masking*: a binary mask $\mathbf{m} \in \{0, 1\}^n$ with $m_i = 1$ if and only if feature i has been acquired, together with a full-length feature vector in which the unobserved entries are replaced by a placeholder value (e.g., zero). The input to f is then the concatenation of the masked feature vector and the mask itself. This

is the representation adopted throughout this thesis, as detailed in Section 4.2 of Chapter 4.

3.1.2 Acquisition Process

The features to be acquired for an instance $\mathbf{x}$ are determined adaptively by a policy π . Given the current partial observation $\mathbf{x}_S$ and a per-instance hard budget $B \geq 0$, the available actions are restricted to those acquisitions that fit within the remaining budget, together with the option to stop:

$$\mathcal{A}_B(S) \triangleq \{a \in U : c(S) + c_a \leq B\} \cup \{\text{STOP}\}. \quad (3.3)$$

If π selects $a \in U$, the cost c_a is paid and the value x_a is observed; if it selects **STOP**, the process terminates with the prediction $\hat{y} = f(S, \mathbf{x}_S)$. We write $\pi[\mathbf{x}] \subseteq [n]$ for the final set of features acquired by the policy on instance $\mathbf{x}$.

3.1.3 Optimisation Objective

The unified goal of AFA is to jointly optimise a policy π and a classifier f to balance predictive accuracy against cumulative acquisition expenditure [1]:

$$\min_{f, \pi} \mathbb{E}_{(\mathbf{x}, y) \sim p(\mathbf{x}, y)} \left[\ell \left(f \left(\pi[\mathbf{x}], \mathbf{x}_{\pi[\mathbf{x}]} \right), y \right) \right] \quad (3.4)$$

$$\text{subject to } c(\pi[\mathbf{x}]) \leq B \quad \forall \mathbf{x} \in \mathcal{X}, \quad (3.5)$$

where $\ell(\cdot, \cdot)$ is the classification loss.

The constraint $c(\pi[\mathbf{x}]) \leq B$ imposes a *hard budget* that no instance may exceed [1]. A hard budget is natural when resources are strictly limited per instance and is often easier to interpret, although its drawback is that some instances may require features whose total cost exceeds B to be predicted accurately. Throughout this thesis we adopt this setting, so that acquisition is constrained solely by the hard budget.

3.2 The Equivalence Class Edge Cutting (EC2) Objective

To guide feature selection without relying on the sample-inefficient optimisation loops of reinforcement learning, we adopt the Equivalence Class Edge Cutting (EC2) objective [24], [25]. We present it in the instance-based form used in the AFA literature [1], [25].

3.2.1 Decision Regions, the Version Space, and the Conflict Graph

For each label $y \in \mathcal{Y}$, define the *decision region*

$$\mathcal{R}_y \triangleq \{x \in \mathcal{X} : y = \arg \max_{y' \in \mathcal{Y}} p(y' | x)\}, \quad (3.6)$$

i.e., the region of the feature space where label y is the most probable given the full feature vector, which is the Bayes-optimal prediction under 0–1 loss. More generally, the regions can be defined through any loss ℓ by replacing the $\arg \max$ with $\arg \min_{y'} \mathbb{E}_{\mathbf{y}|x}[\ell(\mathbf{y}, y')]$. We fix an arbitrary tie-breaking rule, so that $\{\mathcal{R}_y\}_{y \in \mathcal{Y}}$ forms a partition of $\mathcal{X}$.

Central to EC2 is the *version space* of a partial observation: the set of full feature vectors consistent with the evidence acquired so far,

$$V(x_S) \triangleq \{x \in \mathcal{X} : x_j = (x_S)_j \text{ for all } j \in S\}, \quad (3.7)$$

whose probability mass satisfies $p(V(x_S)) = p(x_S)$, since the event $\{\mathbf{x} \in V(x_S)\}$ is exactly the event $\{\mathbf{x}_S = x_S\}$.

Two instances that fall into *different* decision regions are in conflict: no single prediction can be correct for both. EC2 captures this conflict in a weighted *conflict graph* over the feature space. The edge set contains exactly the conflicting pairs,

$$\mathcal{E} \triangleq \bigcup_{y \neq y'} \{\{x, x'\} : x \in \mathcal{R}_y, x' \in \mathcal{R}_{y'}\}, \quad (3.8)$$

and each edge carries the weight

$$w(x, x') = p(x)p(x'), \quad (3.9)$$

the joint probability mass of the conflicting pair under the data distribution. An edge is *consistent* with x_S if both of its endpoints lie in the version space $V(x_S)$; acquiring a feature eliminates (“cuts”) exactly the edges that become inconsistent. Formally, observing feature i with outcome v cuts the edges

$$\mathcal{E}(i, v) \triangleq \{\{x, x'\} \in \mathcal{E} : x_i \neq v \text{ or } x'_i \neq v\}, \quad (3.10)$$

where x_i denotes the i -th coordinate of x . Acquisition is sufficient once all remaining conflict is resolved, i.e., once $V(x_S) \subseteq \mathcal{R}_y$ for some y : all feature vectors still consistent with the evidence then agree on the optimal prediction. Notably, this stopping condition is *self-certifying*, since the policy can detect it from the observed evidence alone [24]. The total weight of edges still consistent with x_S measures the amount of unresolved disagreement, and the EC2 utility $u_{\text{EC}^2}(x_S)$ is defined as the total weight of edges *cut* by the observed outcomes.

3.2.2 The Acquisition Objective

The EC2 acquisition rule selects, at each step, the feature that maximises the *expected one-step gain in cut edge weight*, normalised by the feature cost [1], [24]:

$$a^* = \arg \max_{a \in \mathcal{A}_B(S) \setminus \{\text{STOP}\}} \frac{\mathbb{E}_{\mathbf{x}_a|x_S} [u_{\text{EC}^2}(x_S, \mathbf{x}_a)] - u_{\text{EC}^2}(x_S)}{c_a}, \quad (3.11)$$

where x_S is the currently observed evidence; equivalently, the rule maximises the expected reduction in the weight of edges still consistent with x_S . Because edges

represent pairs of instances that are simultaneously probable and in disagreement, cutting a high-weight edge requires resolving a conflict that affects a large fraction of the probability mass, precisely the disagreements that matter for classification.

Crucially, u_{EC2} satisfies the *adaptive submodularity* property introduced by Golovin and Krause [36]: informally, the expected marginal gain of an acquisition does not increase as more features are observed, and the objective is adaptive monotone. For any adaptive submodular and adaptive monotone objective, greedy selection achieves a $(1 - 1/e)$ -approximation to the globally optimal (non-myopic) policy when acquisition costs are uniform [36], [37], with extensions to non-uniform costs given by Golovin and Krause [36]. In fact, EC2 is the only objective suited to AFA, together with its direct extensions, that is known to be adaptive submodular, whereas common alternatives such as the negative conditional entropy are not, even under a Naive Bayes assumption [1], [26]. Greedy acquisition with respect to EC2 therefore carries a formal near-optimality guarantee, and induces non-myopic behaviour in the sense that it approximates the multi-step lookahead value [1]. It should be noted, however, that EC2 is a surrogate objective: it is connected to, but not fully aligned with, the AFA objective of Equation (3.4), so greedy optimisation of EC2 can in principle be suboptimal for the original problem [1].

3.2.3 The Combined Limitations of CMI and GBS

The design of EC2 is best understood as a response to the complementary shortcomings of the two classical acquisition objectives: conditional mutual information (CMI) and generalized binary search (GBS).

Conditional mutual information. The most common myopic rule selects the feature maximising the conditional mutual information $I(\mathbf{y}; \mathbf{x}_i \mid x_S)$ with the label [1]. CMI is label-directed, but it suffers from two problems. First, it is myopic and therefore blind to *complementarities* among features: consider independent, uniformly distributed binary features with $y = x_1 \oplus x_2$ (XOR), where $p(y \mid x_1) = p(y) = p(y \mid x_2)$; each feature then has zero mutual information with the label, although observing both determines the label exactly. Second, the underlying utility $-H(\mathbf{y} \mid x_S)$ is known *not* to be adaptive submodular, even under a Naive Bayes assumption [1], [26], so no near-optimality guarantee holds for greedy CMI-based selection in general; indeed, greedy selection by information gain or by the value of information under 0–1 loss can be $\Omega(n/\log n)$ times costlier than optimal [24].

Generalized binary search. GBS greedily acquires the feature whose outcome distribution given x_S is most impure, thereby splitting the version space as evenly as possible; equivalently, it maximises the eliminated version-space probability mass, with utility $u_{\text{GBS}}(x_S) = 1 - p(x_S)$ (up to an additive term independent of the acquired features) [24]. GBS *is* adaptive submodular, so greedy selection is near-optimal [24], [36], but only for the objective of identifying the *full feature realisation*, terminating only when the version space is a singleton. For prediction this target is unnecessarily strict: the label is typically determined long before the entire realisation is. Worse, the two objectives are not aligned, and the misalignment can be severe: Golovin et

al. construct an instance with n features on which GBS acquires, in expectation, $n/2$ times as many features as the optimal policy for determining the decision region, which needs only one, precisely because *reducing the version-space mass does not necessarily facilitate differentiation among the decision regions* [24].

EC2. EC2 retains from GBS the version-space perspective that yields adaptive submodularity, but redefines the objective around *decision regions*: edges connect only instances in conflicting regions, so that progress is measured by the elimination of label conflict rather than by the shrinkage of the version space as a whole, and acquisition stops as soon as the version space is contained in a single region. From the CMI perspective, EC2 likewise keeps the focus on the label. It is this combination, namely label-directed termination with an adaptively submodular objective, that distinguishes EC2, which is the only objective suited to AFA, together with its direct extensions, that is known to be adaptive submodular [1], [24]. Chapter 4 derives a closed form of the EC2 utility and its marginal gain, states its adaptive monotonicity and adaptive submodularity (Theorem 4.1), and relates the resulting gain to the myopic CMI and GBS objectives of this section (Section 4.1.5).

3.3 k -Nearest-Neighbour Estimation

The EC2 marginal gain of the acquisition objective requires the conditional distributions $p(x_i \mid \mathbf{x}_S)$ and $p(y \mid \mathbf{x}_S, x_i)$ for arbitrary partial observations. In general these are intractable to compute exactly, and fitting a deep generative model to provide them is itself computationally expensive and difficult to train [1], [28]. A practical alternative, used by a growing number of AFA methods [27], [28], [29], is *non-parametric neighbourhood estimation* with k -nearest neighbours (kNN) [1].

The k -nearest-neighbour method estimates a local conditional distribution from the training set. Given a query point $\mathbf{x}$, a reference dataset $\mathcal{D}$, and a distance function $d(\cdot, \cdot)$, the k -nearest-neighbour set $\mathcal{N}_k(\mathbf{x}) \subseteq \mathcal{D}$ consists of the k training samples closest to $\mathbf{x}$ under d . Empirical statistics computed over $\mathcal{N}_k(\mathbf{x})$ provide consistent non-parametric estimates of local conditional distributions; in particular, the class posterior is approximated by the empirical label frequencies in the neighbourhood:

$$\hat{p}(y = c \mid \mathbf{x}) = \frac{|\{(\mathbf{x}', y') \in \mathcal{N}_k(\mathbf{x}) : y' = c\}|}{k}. \quad (3.12)$$

In the AFA setting, the relevant conditional is $p(y \mid \mathbf{x}_S)$, the label distribution given only the currently observed feature subset S . Standard Euclidean distance treats all coordinates equally and is therefore inappropriate when only a subset $S \subseteq [n]$ of features is available. Replacing it with a *masked distance* that restricts the comparison to observed coordinates,

$$d(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \|\mathbf{m} \odot (\mathbf{x}^{(i)} - \mathbf{x}^{(j)})\|_2, \quad (3.13)$$

where $\mathbf{m}$ is the acquisition mask and $\odot$ denotes element-wise multiplication, yields neighbours that are locally similar under $\mathbf{x}_S$ alone. The empirical statistics over this

neighbourhood then serve as a Monte Carlo approximation of the true posterior $p(y \mid \mathbf{x}_S)$ [1], [28]. In the same way, the expectation over the unobserved feature x_i in the EC2 marginal gain can be approximated by averaging over the neighbourhood; for continuous features, this is done by first discretising the candidate feature through a set of threshold splits. This yields tractable, kNN-based EC2 gain targets for the value network of the proposed framework, as detailed in Section 4.3 of Chapter 4.

4

The Deep-EC2 Framework

This chapter presents the proposed Deep-EC2 framework. Section 4.1 develops the EC2 objective of Section 3.2 into a closed form that avoids explicit enumeration of the (intractably large) edge set, derives its one-step marginal gain, the quantity that the framework must compute, and states the adaptive monotonicity and adaptive submodularity underlying its near-optimality guarantee. The remaining sections describe the components of the framework. Training proceeds in three phases: the predictor is pretrained on random masks (Section 4.2); a value network is trained by supervised regression onto kNN estimates of the EC2 gain (Sections 4.3 and 4.4); and the predictor is finally adapted to the subsets that the learned policy acquires (Section 4.5). Algorithm 4.5 summarises the complete pipeline.

4.1 The EC2 Objective in Closed Form

The derivation of the EC2 objective presented in this section is based on the exposition of the EC2 objective in the survey of Aronsson et al. [1], which we develop into a closed form that avoids explicit enumeration of the (intractably large) edge set.

Throughout this section we write $p(x_S) \triangleq \mathbb{P}(\mathbf{x}_S = x_S)$ and, for a subset $\mathcal{A} \subseteq \mathcal{X}$, $p(\mathcal{A}) \triangleq \mathbb{P}(\mathbf{x} \in \mathcal{A})$, so that in particular $p(V(x_S)) = p(x_S)$ for the version space of Equation (3.7). We also abbreviate $p(\mathcal{R}_y \mid x_S) \triangleq \mathbb{P}(\mathbf{x} \in \mathcal{R}_y \mid \mathbf{x}_S = x_S)$ and let $w(\mathcal{E}') \triangleq \sum_{\{x, x'\} \in \mathcal{E}'} w(x, x')$ denote the total weight of an edge set. With these conventions, the EC2 utility of Section 3.2 is the total weight of edges cut by the observed outcomes,

$$u_{\text{EC}^2}(x_S) = w\left(\bigcup_{i \in S} \mathcal{E}(i, x_i)\right), \quad (4.1)$$

which depends on x_S only through its coordinates and is therefore well defined.

4.1.1 Remaining Edge Weight and the Gini Form

Direct evaluation of Equation (4.1) requires enumerating all edges, which is intractable for large or continuous feature spaces. We therefore derive an equivalent expression in terms of the conditional distributions that the framework can estimate. The weight of the edges between two distinct regions $\mathcal{R}_y$ and $\mathcal{R}_{y'}$ that are

still consistent with x_S (i.e., whose both endpoints lie in the version space) is

$$\begin{aligned}
 w(y, y' \mid x_S) &\triangleq w((\mathcal{R}_y \cap V(x_S)) \times (\mathcal{R}_{y'} \cap V(x_S))) \\
 &= \sum_{x' \in \mathcal{R}_y \cap V(x_S)} \sum_{x'' \in \mathcal{R}_{y'} \cap V(x_S)} p(x') p(x'') \\
 &= p(\mathcal{R}_y \cap V(x_S)) p(\mathcal{R}_{y'} \cap V(x_S)) \\
 &= p(x_S)^2 p(\mathcal{R}_y \mid x_S) p(\mathcal{R}_{y'} \mid x_S),
 \end{aligned} \tag{4.2}$$

where the last step uses $p(\mathcal{R}_y \cap V(x_S)) = p(V(x_S)) p(\mathcal{R}_y \mid x_S)$ together with $p(V(x_S)) = p(x_S)$. In particular, before any feature is observed, $w(y, y' \mid \emptyset) = p(\mathcal{R}_y) p(\mathcal{R}_{y'})$. Summing over all pairs of distinct regions, and halving because the sum over ordered pairs (y, y') counts every unordered edge $\{x, x'\}$ twice, yields the total weight of edges remaining given x_S :

$$\begin{aligned}
 w(x_S) &\triangleq \frac{1}{2} \sum_{y \neq y'} w(y, y' \mid x_S) \\
 &= \frac{1}{2} p(x_S)^2 \sum_{y \neq y'} p(\mathcal{R}_y \mid x_S) p(\mathcal{R}_{y'} \mid x_S) \\
 &= \frac{1}{2} p(x_S)^2 \left(1 - \sum_{y \in \mathcal{Y}} p(\mathcal{R}_y \mid x_S)^2 \right),
 \end{aligned} \tag{4.3}$$

where the last step uses that $\{\mathcal{R}_y\}_{y \in \mathcal{Y}}$ partitions $\mathcal{X}$, so that $\sum_y p(\mathcal{R}_y \mid x_S) = 1$ whenever $p(x_S) > 0$. Note that $w(\emptyset) = \frac{1}{2}(1 - \sum_y p(\mathcal{R}_y)^2)$. Since every edge is either cut by x_S or still consistent with x_S , and no cut edge can become consistent again, the EC2 utility of Equation (4.1) equals

$$u_{\text{EC}^2}(x_S) = w(\emptyset) - w(x_S) = \frac{1}{2} \left[1 - \sum_{y \in \mathcal{Y}} p(\mathcal{R}_y)^2 - p(x_S)^2 \left(1 - \sum_{y \in \mathcal{Y}} p(\mathcal{R}_y \mid x_S)^2 \right) \right]. \tag{4.4}$$

4.1.2 The Realizable Case

In general, $p(\mathcal{R}_y \mid x_S)$ cannot be simplified further. It collapses to the ordinary label posterior, however, under the *realizability* assumption that the label is fully determined by the features, i.e., $p(y \mid x) \in \{0, 1\}$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Under this assumption, $\mathbf{1}[x \in \mathcal{R}_y] = p(y \mid x)$, and hence

$$\begin{aligned}
 p(\mathcal{R}_y \mid x_S) &= \frac{p(\mathcal{R}_y \cap V(x_S))}{p(V(x_S))} = \frac{\sum_{x \in V(x_S)} \mathbf{1}[x \in \mathcal{R}_y] p(x)}{p(x_S)} \\
 &= \frac{\sum_{x \in V(x_S)} p(y \mid x) p(x)}{p(x_S)} = \frac{\sum_{x \in V(x_S)} p(y, x)}{p(x_S)} \\
 &= \frac{\sum_{x_U} p(y, x_S, x_U)}{p(x_S)} = \frac{p(y, x_S)}{p(x_S)} = p(y \mid x_S).
 \end{aligned} \tag{4.5}$$

Specialising Equation (4.4) to the realizable case therefore amounts to replacing every occurrence of $p(\mathcal{R}_y \mid \cdot)$ by $p(y \mid \cdot)$.

4.1.3 One-Step Marginal Gain

The greedy rule of Equation (3.11) requires, for every candidate feature $i \in U$, the expected one-step gain

$$\begin{aligned}\Delta_{\text{EC}^2}(i \mid x_S) &\triangleq \mathbb{E}_{\mathbf{x}_i \mid x_S} [u_{\text{EC}^2}(x_S, \mathbf{x}_i)] - u_{\text{EC}^2}(x_S) \\ &= w(x_S) - \mathbb{E}_{\mathbf{x}_i \mid x_S} [w(x_S, \mathbf{x}_i)],\end{aligned}\tag{4.6}$$

where the constant $w(\emptyset)$ cancels. Substituting Equation (4.3) and using $p(x_S, x_i) = p(x_S)p(x_i \mid x_S)$,

$$\begin{aligned}\Delta_{\text{EC}^2}(i \mid x_S) &= \frac{1}{2} p(x_S)^2 \left(1 - \sum_y p(\mathcal{R}_y \mid x_S)^2 \right) - \frac{1}{2} \mathbb{E}_{\mathbf{x}_i \mid x_S} \left[p(x_S, \mathbf{x}_i)^2 \left(1 - \sum_y p(\mathcal{R}_y \mid x_S, \mathbf{x}_i)^2 \right) \right] \\ &= \frac{1}{2} p(x_S)^2 \left(\text{Gini}(p(\mathcal{R} \mid x_S)) - \mathbb{E}_{\mathbf{x}_i \mid x_S} \left[p(\mathbf{x}_i \mid x_S)^2 \text{Gini}(p(\mathcal{R} \mid x_S, \mathbf{x}_i)) \right] \right),\end{aligned}\tag{4.7}$$

where $\text{Gini}(q) \triangleq 1 - \sum_z q(z)^2$ denotes the Gini impurity of a distribution q , here applied to the distribution of decision regions (or labels in the realizable case of Section 4.1.2).

Equation (4.7) has two consequences. First, since the term $\text{Gini}(p(\mathcal{R} \mid x_S))$ does not depend on i and $p(x_S)^2 > 0$, the greedy selection $\arg \max_i \Delta_{\text{EC}^2}(i \mid x_S)$ —which coincides with the cost-normalised rule of Equation (3.11) when acquisition costs are uniform, as in all experiments of this thesis—is equivalently

$$\arg \min_{i \in U} \mathbb{E}_{\mathbf{x}_i \mid x_S} \left[p(\mathbf{x}_i \mid x_S)^2 \text{Gini}(p(\mathcal{R} \mid x_S, \mathbf{x}_i)) \right] : \tag{4.8}$$

the EC2 rule prefers the feature whose realised value is expected to leave the *least* version-space-mass-weighted impurity over decision regions, a splitting criterion directed specifically at label conflicts, in contrast to GBS, which splits the version space irrespective of the regions. Second, the gain depends on the data distribution only through the conditionals $p(x_i \mid x_S)$ and $p(\mathcal{R}_y \mid x_S, x_i)$, which is precisely what makes a non-parametric, neighbourhood-based approximation possible (Section 4.3).

4.1.4 Adaptive Monotonicity and Adaptive Submodularity

We first recall the two properties underlying the near-optimality guarantee of greedy selection [36]. For a utility function u defined on partial observations and a feature $i \notin S$, the *expected marginal gain* is $\Delta(i \mid x_S) = \mathbb{E}_{\mathbf{x}_i \mid x_S} [u(x_S, \mathbf{x}_i)] - u(x_S)$. The function u is *adaptive monotone* if $\Delta(i \mid x_S) \geq 0$ for every x_S with $p(x_S) > 0$ and every $i \notin S$; it is *adaptive submodular* if, for every pair of partial observations $x_S \subseteq x_{S'}$ such that $x_{S'}$ extends x_S (i.e., $S \subseteq S'$ and the two agree on S) and $p(x_{S'}) > 0$, and for every feature $i \notin S'$,

$$\Delta(i \mid x_S) \geq \Delta(i \mid x_{S'}).\tag{4.9}$$

Adaptive submodularity thus requires that the expected marginal gain of a feature, where the expectation is over its own outcome, never increases as more evidence is observed, the adaptive analogue of diminishing returns. Note that the inequality must hold for *every* extension $x_{S'}$, not merely on average over extensions.

Theorem 4.1 (Golovin et al. [24]). *The EC2 utility u_{EC^2} of Equation (4.1) is adaptive monotone and adaptive submodular with respect to any prior $p(\mathbf{x})$ over a finite set of feature realisations (proved by Golovin et al. for rational-valued priors).*

Together with the adaptive monotonicity/submodularity framework of Golovin and Krause [36], [37], the theorem yields the near-optimality guarantee quoted in Section 3.2: for uniform acquisition costs, the greedy policy of Equation (3.11) obtains at least a $(1 - 1/e)$ fraction of the optimal policy’s expected utility.

The intuition behind the theorem is that edges, once cut, remain cut: acquiring a feature can only resolve conflict, never create it, which makes the utility adaptive monotone. The gain of a feature is the weight of the conflict it eliminates among instances still consistent with the evidence, and as features accumulate, less conflict remains to be resolved, so expected marginal gains shrink—the adaptive analogue of diminishing returns, i.e., adaptive submodularity. Structurally, for every full realisation the utility is a weighted coverage objective over the edges cut by the observed outcomes, the canonical setting in which greedy selection is near-optimal [24], [36].

For the framework of this thesis, the theorem is the decisive property: the one-step greedy rule of Equation (3.11) is non-myopic in effect while remaining purely greedy in mechanism—no lookahead, planning, or simulated rollouts are required, because the near-optimal quality is inherited from the objective rather than from an optimisation procedure. It is precisely this structure that allows the greedy rule to be replaced by a value network trained by supervised regression (Sections 4.3 and 4.4). Two qualifications are in order. First, the guarantee bounds the loss in EC2 utility, not in the AFA objective of Equation (3.4), to which EC2 is a surrogate [1]. Second, it concerns exact greedy selection on the exact utility, whereas the concrete framework of this chapter approximates both; the consequences for the deployed policy are made precise in Section 4.6.

4.1.5 Comparison of Acquisition Objectives

The closed form of Equation (4.7) makes precise how EC2 relates to the two classical acquisition objectives discussed in Section 3.2.3. The myopic CMI rule selects the feature maximising $I(\mathbf{y}; \mathbf{x}_i \mid x_S)$, thereby reducing the impurity of the label posterior; its underlying utility $-H(\mathbf{y} \mid x_S)$ is not adaptive submodular, however, so greedy selection carries no near-optimality guarantee, and the one-step lookahead is blind to complementarities among unobserved features. GBS, by contrast, *is* adaptive submodular, but its gain

$$\Delta_{\text{GBS}}(i \mid x_S) = p(x_S) \text{Gini}(p(\mathbf{x}_i \mid x_S))$$

depends only on the outcome distribution of the candidate feature: the split is directed at the version space as a whole, irrespective of the decision regions, which

can be linearly suboptimal for prediction [24].

The EC2 gain of Equation (4.7), equivalently the selection rule of Equation (4.8), combines the two perspectives. Like CMI, it is label-directed: the impurity is evaluated over the decision-region posterior $p(\mathcal{R}_y \mid x_S, \mathbf{x}_i)$, so a split is valued only insofar as it separates conflicting regions. Like GBS, it is version-space-aware: each candidate outcome is weighted by $p(\mathbf{x}_i \mid x_S)^2$, the mass of the version space that the outcome leaves intact, mirroring the version-space factor $p(x_S)$ of the GBS gain. A feature is therefore preferred precisely when both its outcome is decisive for the version space and the resulting posterior over decision regions is pure. This coupling yields acquisition that is simultaneously label-directed and adaptive submodular—EC2 remains the only objective suited to AFA, together with its direct extensions, that is known to be adaptive submodular [1], [24]. The kNN-based target of Section 4.3, on which the framework is built, retains the same functional form, i.e., the same label-directed, version-space-weighted structure; whether it inherits the adaptive submodularity itself is a separate question, discussed in Section 4.6. Moreover, like both alternatives, the gain depends on the data distribution only through the conditionals $p(\mathbf{x}_i \mid x_S)$ and $p(\mathcal{R}_y \mid x_S, \mathbf{x}_i)$, the latter reducing to $p(y \mid x_S, \mathbf{x}_i)$ in the realizable case (GBS requires only the former), so the single neighbourhood-based estimator of Section 3.3 suffices to evaluate any of them.

4.2 Partial Observation Encoding

The predictor and the value network must both process an *arbitrary* subset of the available features. We adopt the masked representation introduced in Section 3.1: a partial observation (S, x_S) is encoded as the pair

$$\xi(S, x_S) = (x \odot m, m) \in \mathbb{R}^{2n}, \quad (4.10)$$

where $m \in \{0, 1\}^n$ is the acquisition mask with $m_i = 1$ if and only if $i \in S$, $x \in \mathcal{X}$ denotes the full-length instance vector in which the unobserved coordinates are set to a placeholder value (zero), and $\odot$ denotes element-wise multiplication. Appending the mask serves two purposes: it distinguishes a genuinely observed zero from an unobserved placeholder, and it lets the network infer exactly which features have been acquired, which the masked vector alone does not reveal. Both the predictor f of Section 3.1 and the value network of Section 4.4 consume the resulting $2n$ -dimensional vector.

4.2.1 Predictor Pretraining under Random Masks

Feature selection is only meaningful if the classifier can exploit whichever subset happens to have been acquired. In the first training phase, the framework therefore pretrains the predictor to classify from *any* partial observation. In each training step an observation pattern is sampled by drawing a scalar threshold $r \sim \text{U}(0, 1)$ and, for every feature, an independent uniform variate $\nu_i \sim \text{U}(0, 1)$, and setting $m_i = \mathbf{1}[\nu_i > r]$. Conditionally on r , each feature is observed independently with probability $1 - r$; marginalising over r , every feature is observed with probability

1/2, but the features are no longer independent: the number of observed features is uniformly distributed on $\{0, 1, \dots, n\}$, since $\mathbb{P}(|S| = k) = \int_0^1 \binom{n}{k} (1-r)^k r^{n-k} dr = 1/(n+1)$, and given its size the subset is uniform. Every subset is therefore realised, and small and large subsets are visited equally often. The predictor is trained with cross-entropy on the masked encodings,

$$\mathcal{L}_{\text{pre}}(\theta_f) = \mathbb{E}[\ell_{\text{CE}}(f(\xi(S, x_S)), y)], \quad (4.11)$$

where the expectation is over instances, masks, and the resulting subsets. Exposing the predictor to the full spectrum of subsets, and in particular to the low-budget states that acquisition visits, is what allows a single network to serve as the classifier at every point of the acquisition trajectory. Once pretraining converges, the predictor weights are frozen while the value network is trained; they are adapted to the learned policy only in the third phase (Section 4.5). Algorithm 4.1 summarises the procedure.

Algorithm 4.1 Predictor pretraining under random masks

Require: Training set $\mathcal{D}$, predictor f_{θ_f}

Ensure: Pretrained parameters θ_f (frozen thereafter)

- 1: **for** each mini-batch $(x, y) \sim \mathcal{D}$ **do**
 - 2: draw $r \sim \text{U}(0, 1)$ and $\nu_i \sim \text{U}(0, 1)$ independently for $i = 1, \dots, n$;
 - 3: set $m_i \leftarrow \mathbf{1}[\nu_i > r]$ and $S \leftarrow \{i : m_i = 1\}$ ▷ random mask
 - 4: $\hat{y} \leftarrow f_{\theta_f}(\xi(S, x_S))$ ▷ Equation (4.10)
 - 5: update θ_f by one gradient step on $\ell_{\text{CE}}(\hat{y}, y)$ ▷ Equation (4.11)
 - 6: **end for**
-

4.3 EC2 Target Computation

By Equation (4.8), under the unit acquisition costs used throughout this thesis, greedy selection of the EC2 objective reduces to picking the feature with the smallest expected version-space-mass-weighted impurity

$$Q_i(x_S) = \mathbb{E}_{\mathbf{x}_i | x_S} \left[p(\mathbf{x}_i | x_S)^2 \text{Gini}(p(\mathcal{R} \cdot | x_S, \mathbf{x}_i)) \right], \quad (4.12)$$

for every candidate feature $i \in U$, where $\text{Gini}(q) = 1 - \sum_z q(z)^2$ (Section 4.1.3) and, in the realizable case of Section 4.1.2, the decision-region distribution $p(\mathcal{R} \cdot | \cdot)$ coincides with the label posterior. Deep-EC2 trains a *value network* (Section 4.4) to regress onto $Q_i(x_S)$, so the quantities that must be computed during training are $Q_i(x_S)$ for every unobserved feature and every state the policy visits. Computing them amounts to estimating the two conditionals $\hat{p}(\mathbf{x}_i | x_S)$ and $\hat{p}(y | x_S, \mathbf{x}_i)$. Using the label posterior in place of the decision-region posterior is exact only under realizability; on datasets with label noise, including the synthetic benchmarks of Chapter 5, whose labels are sampled from a Bernoulli distribution, it is an additional approximation. Following Section 3.3, both are estimated non-parametrically from a neighbourhood of training instances under the masked distance.

4.3.1 Threshold Binarization and the Cubic Target

For a discrete feature, $Q_i(x_S)$ could be evaluated directly over its finite outcomes; for continuous features the expectation in Equation (4.12) is over a continuum and must be discretised. Deep-EC2 divides the range of each candidate feature into M bins. The bin boundaries are the $M - 1$ equi-spaced quantiles of the training values of feature i ,

$$\tau_{i,b} = \hat{q}_{b/M}(x_i^{(N)}), \quad b = 1, \dots, M - 1, \quad (4.13)$$

so that each bin carries approximately the same training-set mass ($M = 10$ in all experiments). Conditioning on the current partial observation x_S , let $\mathcal{N}_k(x_S)$ denote the k nearest training instances under the masked distance of Section 3.3, and let $\hat{p}_b(x_S)$ be the fraction of $\mathcal{N}_k(x_S)$ whose i -th coordinate falls into bin b . To evaluate the expectation we adopt a piecewise-constant approximation in which the conditional density $p(\mathbf{x}_i | x_S)$ is represented, within each bin, by its probability mass $\hat{p}_b(x_S)$; its square—the version-space-mass factor of Equation (4.12)—is then approximated by $\hat{p}_b(x_S)^2$, and the expectation collapses to a sum over bins. This yields the supervised regression target

$$T_i(x_S) = \sum_{b=1}^M \hat{p}_b(x_S)^3 \text{Gini}(\hat{p}(\cdot | \text{bin}_b, x_S)), \quad (4.14)$$

i.e., with the empirical bin posteriors made explicit,

$$T_i(x_S) = \sum_{b=1}^M \hat{p}_b(x_S)^3 \left(1 - \sum_{c \in \mathcal{Y}} \hat{p}(c | \text{bin}_b, x_S)^2 \right),$$

where $\hat{p}(c | \text{bin}_b, x_S)$ is the empirical frequency, within bin b of the neighbourhood, of label c . The cubic weight has a transparent reading: $\hat{p}_b$ is the probability mass of the candidate outcome—the “splitting” factor, in the spirit of GBS—and the additional $\hat{p}_b^2$ is the discrete analogue of the version-space-mass factor $p(\mathbf{x}_i | x_S)^2$ of Equation (4.12). A bin therefore contributes to the target in proportion to both how likely the outcome is and how much of the version space that outcome leaves intact. Since factors independent of i have been discarded throughout (Equation (4.8)), the greedy EC2 rule is recovered by selecting the feature with the smallest target,

$$a^* = \arg \min_{i \in U} T_i(x_S), \quad (4.15)$$

with previously acquired features excluded. Algorithm 4.2 summarises the computation; bins containing no neighbour contribute zero to the target.

Algorithm 4.2 kNN-based EC2 target computation

Require: Partial observation x_S with mask m ; training set $\mathcal{D}$; neighbourhood size k ; quantile bins $\{\text{bin}_b\}_{b=1}^M$ per feature (Equation (4.13))

Ensure: Target vector $T(x_S) \in \mathbb{R}^n$

```

1: if  $S = \emptyset$  then
2:   return cached global target  $T(\emptyset)$  ▷ Equation (4.16)
3: end if
4:  $\mathcal{N}_k(x_S) \leftarrow$  the  $k$  instances of  $\mathcal{D}$  nearest to  $x_S$  under the masked distance of Equation (3.13)
5: for each unobserved feature  $i \in U$  do
6:   for  $b = 1, \dots, M$  do
7:      $\mathcal{N}_b \leftarrow \{(x', y') \in \mathcal{N}_k(x_S) : x'_i \in \text{bin}_b\}$ ;  $\hat{p}_b \leftarrow |\mathcal{N}_b| / k$ 
8:      $\hat{p}(c \mid \text{bin}_b, x_S) \leftarrow |\{(x', y') \in \mathcal{N}_b : y' = c\}| / |\mathcal{N}_b|$  for each  $c \in \mathcal{Y}$ 
9:      $G_b \leftarrow 1 - \sum_{c \in \mathcal{Y}} \hat{p}(c \mid \text{bin}_b, x_S)^2$ 
10:   end for
11:    $T_i(x_S) \leftarrow \sum_{b=1}^M \hat{p}_b^3 G_b$  ▷ Equation (4.14)
12: end for
13: return  $T(x_S)$ 

```

4.3.2 The Initial State

When nothing has been observed ($S = \emptyset$), the masked distance of Equation (3.13) is zero for every pair, the neighbourhood degenerates to the whole training set, and no conditioning is in force. In that case the target of Equation (4.14) is evaluated over all N training instances rather than over a neighbourhood: it becomes the exact global statistic of the training set,

$$T_i(\emptyset) = \sum_{b=1}^M \hat{p}_b^3 \left(1 - \sum_{c \in \mathcal{Y}} \hat{p}(c \mid \text{bin}_b)^2 \right), \quad (4.16)$$

where $\hat{p}_b$ and $\hat{p}(c \mid \text{bin}_b)$ now denote global bin and class frequencies. This quantity is independent of the instance, is computed once and cached, and has zero estimation variance. It supplies the regression target for the first acquisition decision, which is shared by every instance.

Implementation note. The masked distance is evaluated in matrix form, $\sum_j m_j (x_j - x'_j)^2 = m \cdot (x' \odot x') - 2(m \odot x) \cdot x' + \sum_j m_j x_j^2$, which avoids materialising a $B \times N \times n$ difference tensor at every step; because the expanded form loses precision for small distances, it only shortlists $2k$ candidates, whose exact distances then select the k nearest neighbours, so the targets are identical to those of the direct computation. Large training sets use a random subsample of 10 000 instances as kNN reference (Table 5.2); the target at the empty state is always computed on the full training set.

4.4 Value Network Training and the Acquisition Policy

The value network v_θ is a multi-layer perceptron that maps the masked encoding of Equation (4.10) to a vector of EC2 target estimates,

$$v_\theta : \mathbb{R}^{2n} \rightarrow \mathbb{R}^n, \quad v_\theta(\xi(S, x_S))_i \simeq T_i(x_S), \quad (4.17)$$

for all features i ; in the experiments it has two hidden layers of 512 units with ReLU activations and dropout (0.3). The coordinate of an already acquired feature is never used: it is excluded from the training loss and set to $+\infty$ at selection time.

4.4.1 Supervised Regression against kNN Targets

In the second training phase, the pretrained predictor is frozen and only the value network is trained, as a pure regression problem. For every training batch an acquisition episode starts from the empty observation ($S = \emptyset$) and proceeds for up to $B_{\max}$ steps. At each step, given the current partial observation x_S :

1. the value network is evaluated at $\xi(S, x_S)$;
2. an action is drawn ε -greedily: with probability ε uniformly at random among the unobserved features, otherwise the greedy selection of Equation (4.15) evaluated on the value-network estimate;
3. the targets $T_i(x_S)$ of Equation (4.14) are computed for *all* unobserved features at the current state, and the regression loss

$$\mathcal{L}(\theta)_t = \frac{1}{|U|} \sum_{i \in U} (v_\theta(\xi(S, x_S))_i - T_i(x_S))^2 \quad (4.18)$$

is accumulated;

4. the mask is updated with the chosen feature, and the step advances.

Because the targets are computed for every unobserved feature rather than only for the acquired one, the loss is not tied to any single action: the value network must reproduce the EC2 score function over the states the trajectory visits, and the ε -greedy actions serve purely as a state exploration mechanism. This is what turns value estimation into plain regression—there are no accumulated rewards, no bootstrapping, and no reward-to-go; each target is a target in the supervised sense. Gradients are accumulated over the $B_{\max}$ steps of the episode and a single optimiser step is taken per batch, so the loss is averaged over the whole trajectory. On datasets with grouped features, all columns of the selected group are revealed at once, which leaves the formulation unchanged if each group is treated as a single feature. Algorithm 4.3 summarises the training procedure.

Exploration is annealed while training. The trajectory-mean validation metric (Section 5.1) is monitored, and whenever it fails to improve for a patience period, ε is multiplied by a decay factor and the learning rate is reset; training terminates after

a fixed number of such annealings. The random actions during this phase prevent the sampled trajectories from collapsing onto a single fixed acquisition order, and thereby expose every state—and in particular the initial state—to the regression loss.

Algorithm 4.3 Value-network training by simulated acquisition episodes

Require: Frozen predictor f ; value network v_θ ; training set $\mathcal{D}$; budget $B_{\max}$; exploration rate ε

- 1: **for** each mini-batch $(x, y) \sim \mathcal{D}$ **do**
- 2: $S \leftarrow \emptyset$; $\mathcal{L} \leftarrow 0$ ▷ episode starts from the empty observation
- 3: **for** $t = 1, \dots, B_{\max}$ **do**
- 4: $s \leftarrow v_\theta(\xi(S, x_S))$ ▷ estimates of $T_i(x_S)$ for all i (Equation (4.17))
- 5: with probability ε draw a uniformly from U ; otherwise $a \leftarrow \arg \min_{i \in U} s_i$
- 6: $\mathcal{L} \leftarrow \mathcal{L} + \frac{1}{|U|} \sum_{i \in U} (s_i - T_i(x_S))^2$ ▷ Equation (4.18); $T(x_S)$ by Algorithm 4.2
- 7: $S \leftarrow S \cup \{a\}$ ▷ acquire feature a and update the mask
- 8: **end for**
- 9: update θ by one gradient step on $\mathcal{L}/B_{\max}$
- 10: **end for**

4.5 Phase 3: Adapting the Predictor to the Policy

The predictor of Section 4.2.1 is trained on random masks, whose distribution differs from that of the masks the learned policy actually produces: the policy visits a small, structured family of subsets, whereas random masking spreads the training signal over all 2^n of them. Baselines that train their selector jointly with the predictor, such as GDFS and DIME, adapt the classifier to their own policy’s subsets as a by-product. Deep-EC2 does the same in a third training phase that leaves the acquisition policy untouched. With the value network frozen at its selected checkpoint, each training instance is rolled out greedily under Equation (4.20) for $B_{\max}$ steps, and the predictor is fine-tuned with cross-entropy on all masks of the trajectory,

$$\mathcal{L}_{\text{adapt}}(\theta_f) = \mathbb{E} \left[\frac{1}{B_{\max} + 1 + R} \left(\sum_{t=0}^{B_{\max}} \ell_{\text{CE}}(f(\xi(S_t, x_{S_t})), y) + \sum_{r=1}^R \ell_{\text{CE}}(f(\xi(S'_r, x_{S'_r})), y) \right) \right], \quad (4.19)$$

where S_t is the set acquired by the policy after t steps and $S'_1, \dots, S'_R$ are random masks drawn as in Algorithm 4.1 ($R \approx B_{\max}/4$), which keep budgets beyond the training horizon covered. Fine-tuning uses Adam with learning rate 10^{-4} , and the predictor weights with the best trajectory-mean validation metric are retained (patience 5, at most 50 epochs). Because the value network is not updated, the acquisition order—and therefore every statement about the learned policy in Chapter 5—is unchanged; only the classifier evaluated along the trajectory improves. Its contribution is quantified in the ablation of Section 5.5.

4.6 Inference and the Complete Pipeline

At inference a fully trained model acquires features greedily. Starting from an empty observation, at each step the acquired features are excluded (their estimates are set to $+\infty$) and the feature minimising the value-network estimate is selected,

$$\pi(S, x_S) = \arg \min_{i \in U} v_\theta(\xi(S, x_S))_i. \quad (4.20)$$

This repeats until the budget is exhausted, and at any point the prediction is $f(\xi(S, x_S))$, obtained from the adapted predictor. Because the value network feeds directly on the masked encoding and returns one score per feature, each acquisition step costs one forward pass of each network: no rollout, planning, or search is involved. Algorithm 4.4 summarises the evaluation procedure, which reports metrics at every budget level.

Algorithm 4.4 Greedy acquisition and evaluation

Require: Adapted predictor f ; trained value network v_θ ; test set $\mathcal{D}_{\text{test}}$; budget B_{max}

Ensure: Accuracy and AUROC at each budget $t = 0, \dots, B_{\text{max}}$

- 1: **for** each test instance $(x, y) \in \mathcal{D}_{\text{test}}$ **do**
 - 2: $S \leftarrow \emptyset$; $\hat{y}_0 \leftarrow f(\xi(S, x_S))$ ▷ zero-budget prediction
 - 3: **for** $t = 1, \dots, B_{\text{max}}$ **do**
 - 4: $s \leftarrow v_\theta(\xi(S, x_S))$; set $s_i \leftarrow +\infty$ for all $i \in S$
 - 5: $a \leftarrow \arg \min_{i \in [n]} s_i$; $S \leftarrow S \cup \{a\}$ ▷ Equation (4.20)
 - 6: $\hat{y}_t \leftarrow f(\xi(S, x_S))$
 - 7: **end for**
 - 8: **end for**
 - 9: **return** accuracy and AUROC of $\{\hat{y}_t\}_{t=0}^{B_{\text{max}}}$ against the true labels
-

Algorithm 4.5 summarises the complete training pipeline.

Algorithm 4.5 Deep-EC2: complete training pipeline

Require: Training and validation sets; budget B_{max}

- 1: pretrain the predictor f on random masks ▷ phase 1, Algorithm 4.1
 - 2: compute the quantile bins and the cached empty-state target $T(\emptyset)$ ▷ Equations (4.13), (4.16)
 - 3: train the value network v_θ by regression onto kNN targets, with f frozen ▷ phase 2, Algorithms 4.2, 4.3
 - 4: fine-tune f on the trajectories of the greedy policy, with v_θ frozen ▷ phase 3, Equation (4.19)
 - 5: **return** (f, v_θ) ; acquire and predict with Algorithm 4.4
-

Two simplifications relative to the general formulation of Section 3.1 deserve emphasis. First, features are assumed to have unit acquisition cost, so the hard budget of

Equation (3.5) reduces to a limit on the number of acquired features, and the framework is evaluated at every budget $t = 0, \dots, B_{\max}$ rather than at a single learned stopping point. Second, the policy performs no learned stopping: it always acquires until the budget is exhausted and termination is enforced externally.

The relation between the deployed policy and Theorem 4.1 also deserves a precise statement. The $(1 - 1/e)$ guarantee attaches to exact greedy selection on the exact EC2 utility, and the deployed policy satisfies neither premise: the targets $T_i(x_S)$ of Equation (4.14) approximate the true EC2 gain through the neighbourhood estimate and the quantile-bin discretisation, and the value network in turn approximates the targets, so the policy is greedy with respect to an estimate of an estimate. Neither source of error is bounded—the kNN estimator carries no finite-sample error control, the binned objective is not known to be the marginal gain of any adaptive submodular utility, and the regression error of v_θ is uncontrolled—so the near-optimality guarantee does not transfer to Deep-EC2, even though the unit-cost setting matches the uniform-cost premise of the theorem. (If every selected feature were guaranteed to attain at least a fraction $1/\alpha$, $\alpha \geq 1$, of the largest available expected gain, the approximate-greedy analysis of Golovin and Krause [36] would merely recover the weaker $(1 - e^{-1/\alpha})$ bound.) The theorem therefore certifies the design choice—greedy maximisation of an objective that is provably adaptive monotone and adaptive submodular, in contrast to the myopic CMI rule, for which no such property holds (Section 4.1.5)—rather than the deployed policy.

5

Experiments

This chapter reports the experimental evaluation of Deep-EC2. Section 5.1 describes the eleven benchmarks, the baselines, and the evaluation protocol. Section 5.2 then examines CUBE-NM, where Deep-EC2 acquires the context features first and is significantly ahead of every baseline, including the reinforcement-learning method OL; a feature-order heatmap makes the learned policy visible. Section 5.3 turns to the synthetic gated benchmarks, where the same non-myopic behaviour appears on Syn3 as gate-first acquisition and an *invest-then-exploit* signature. Section 5.4 reports the real-world benchmarks, Section 5.5 the contribution of the predictor adaptation, and Section 5.6 training cost, sensitivity analyses, and limitations of the evaluation.

5.1 Experimental Setup

5.1.1 Benchmark Datasets

The evaluation covers eleven benchmarks. The three synthetic benchmarks Syn1–Syn3 follow the gated synthetic tasks of the SEFA benchmark suite [42], adapted from the INVASE benchmark [43]. Each instance has eleven independent features drawn from a standard normal distribution, and the label is sampled from a Bernoulli distribution whose logit is a *gated* combination of two nonlinear functions over disjoint feature subsets: the sign of the gating feature x_{11} selects which of the two logits determines the sampling probability,

$$\text{logit}(p(y=0)) = \mathbf{1}[x_{11} < 0] h_1(x_{S_1}) + \mathbf{1}[x_{11} \geq 0] h_2(x_{S_2}), \quad (5.1)$$

where the three logits are $h(x) \in \{4x_1x_2, \sum_{i=3}^6 1.2x_i^2 - 4.2, -10 \sin(0.2x_7) + |x_8| + x_9 + e^{-x_{10}} - 2.4\}$ (a product, a quadratic form, and a sine/absolute/exponential combination), and Syn1–Syn3 pick the pairs (h_1, h_2) from them; the logits are the versions adapted in [42] from the originals in [43] to produce probabilities closer to 0 or 1, and the remaining features are pure noise. The gating structure gives these datasets a pronounced *forward-looking* component: observing x_{11} alone almost fully determines which subset of the remaining features matters, so a non-myopic policy has a measurable structural advantage over any myopic rule. All three datasets have 11 features, a binary label, and a 60 000/10 000/10 000 train/validation/test split.

CUBE-NM [8] is a synthetic benchmark in which an acquirable context feature (one of five values, uninformative about the label by itself) determines which block of ten

features carries the class signal; the dataset has 55 features and 8 classes. The design includes cheap *context features* that reveal which context applies to an instance; each such feature is individually uninformative about the label yet jointly decisive, again giving non-myopic acquisition a structural advantage. We use the noise-free variant of the generator; 6 250 instances are generated for training and validation in an 80/20 split (5 000/1 250), and the test set is a freshly generated sample of 1 000 instances under a fixed seed shared by all methods.

The remaining seven datasets are standard real-world classification problems: Metabric (breast-cancer gene-expression and survival), Chronic Kidney Disease (CKD, UCI 336), Bank Marketing (UCI 222), MNIST reduced to the 20 most predictive pixels, California Housing with the regression target quantised into 4 equal-frequency bins, MiniBooNE (UCI 199) balanced and reduced to 20 features, and the Electrical Grid Stability dataset. None of them has an explicit feature hierarchy of the kind built into Syn1–Syn3 and CUBE-NM. Pre-processing follows the SEFA and AFA-Benchmark suites where applicable: continuous attributes are standardised (binary/one-hot columns are left untouched), missing values are imputed from the training split only, and the imbalance of MiniBooNE is removed by downsampling. Table 5.1 summarises all datasets.

The synthetic datasets are generated with a single seed and the splits are exported to disk, so every method trains and evaluates on exactly the same instances. For Bank Marketing, acquisitions operate on feature groups rather than individual one-hot columns (Section 5.1.2). For every dataset both accuracy and AUROC are reported at each budget; the trajectory-mean value of the primary metric drives validation checkpoints and early stopping—accuracy for the synthetic, CUBE-NM, and Grid Stability datasets, and AUROC for the tabular datasets.

Dataset	Dim.	Classes	Split (Tr/Va/Te)	Metric	Notes
CUBE-NM	55	8	5 000 / 1 250 / 1 000	Acc & AUROC	synthetic, 5 contexts, noise-free
Syn1–Syn3	11	2	60 k / 10 k / 10 k	Acc & AUROC	synthetic, gated rules
Metabric	12	2	1 518 / 189 / 191	Acc & AUROC	survival (SEFA processing)
CKD	24	2	280 / 60 / 60	Acc & AUROC	UCI 336; median/mode imputation
Bank Marketing	63 (20 groups)	2	70% / 15% / 15%	Acc & AUROC	UCI 222; categorical groups
MNIST	20	10	50 k / 10 k / 10 k	Acc & AUROC	20 STG-selected pixels
California Housing	8	4	16 512 / 2 064 / 2 064	Acc & AUROC	target into 4 bins
MiniBooNE	20	2	56 499 / 10 k / 10 k	Acc & AUROC	20 STG features, balanced
Grid Stability	12	2	8 000 / 1 000 / 1 000	Acc & AUROC	electrical grid simulation

Table 5.1: Summary of the benchmark datasets. “Dim.” is the acquisition dimension (for Bank Marketing, the number of one-hot columns and, in parentheses, the number of acquisition groups after grouping categorical dummy columns, Section 5.1.2). “Metric” lists the metrics reported at every budget on the test split.

5.1.2 Feature Groups for Categorical Attributes

For the Bank Marketing dataset, the ten categorical attributes are one-hot encoded into 53 dummy columns. Exposing the policy to individual dummy columns would split the acquisition of a single categorical attribute—such as the client’s job—across several semantically empty steps. Deep-EC2 therefore pools, for each categorical attribute, all of its dummy columns into one acquisition *group*: the value network

scores each group by the minimum target over its columns, so that a partially acquired group is never selected, and one acquisition reveals all columns of the selected group at once. Together with the ten continuous attributes, taken as single-column groups, this yields 20 acquisition groups, and both training and evaluation operate on groups rather than on columns. The remaining datasets have no categorical attributes requiring grouping.

5.1.3 Baselines

Deep-EC2 is compared with eight references. The first is *random acquisition*: the same predictor as Deep-EC2, with features acquired in uniformly random order. Sharing the classifier isolates the value added by the acquisition policy; any gap between Deep-EC2 and this baseline is attributable to feature selection alone. The remaining seven are published AFA methods run from their public implementations: EDDI [9], GDFS [10], DIME [11], AACO [28], and, from the dynamic-selection benchmark, iterative imputation, greedy ablation, and the concrete-autoencoder baseline CAE, which selects a fixed, instance-independent feature set. All seven run on every benchmark of Table 5.1 under the same protocol, on the same splits, with the same three seeds (0, 1, 2) as Deep-EC2. As a reinforcement-learning reference, we add Opportunistic Learning (OL) [19], a deep Q-learning method in which the value network builds on the representations learned by the predictor network, using the PyTorch implementation of the SEFA suite [42] without modification. OL is trained on exactly the data Deep-EC2 sees; its hyperparameters are chosen among the nine SEFA configurations by the trajectory-mean validation metric (accuracy, as for Deep-EC2), and the selected configuration is trained and evaluated under the same three seeds. Because every OL run requires such a sweep, OL was run on the four synthetic benchmarks, where look-ahead matters most; it is marked “–” elsewhere. Seed s of every method therefore shares the data splits of seed s of Deep-EC2, which is what makes the seed-level paired comparisons of Section 5.1.4 meaningful.

5.1.4 Evaluation Protocol and Metrics

For each trained model, acquisition is simulated once on the held-out test set. Starting from an empty observation, the greedy policy of Equation (4.20) is applied repeatedly; at every budget $t = 0, \dots, B_{\max}$ the (adapted) predictor is evaluated under the currently acquired mask, yielding the accuracy and the AUROC of the partial-observation prediction. The random baseline follows the same procedure with uniformly random feature order. Two quantities are reported for each dataset:

- the *per-budget curves*, accuracy and AUROC as functions of the number of acquired features;
- the *normalised area under each curve*,

$$\text{AUC}(\text{metric}) = \frac{1}{B_{\max}} \int_0^{B_{\max}} \text{metric}(t) dt, \quad (5.2)$$

computed by trapezoidal integration, which summarises the whole budget–performance trade-off in a single scalar.

Statistical reporting. Because all methods share the same three seeds, comparisons are made with *paired* tests: for every dataset the normalised area of Equation (5.2) is computed per seed, giving three paired observations per method, and the differences Deep-EC2 minus each baseline are tested with a two-sided paired t -test (2 degrees of freedom). With three seeds the Wilcoxon signed-rank test cannot reach significance at all (its smallest possible two-sided p -value is 0.25), so the paired t -test is the most powerful option available; p -values below 0.05 are therefore read as indicative rather than confirmatory, and the per-seed spreads are always reported alongside. Individual budgets are tested with the same procedure in Sections 5.2 and 5.3.

AUROC is computed from the predicted probability of the positive class for binary problems and as one-vs-rest macro-averaged AUROC for multiclass problems. At the maximum budget all features are observed, so the acquisition policy no longer matters there: the remaining differences between methods at that point reflect only the quality of each method’s own predictor, not of its acquisition order. The differences attributable to acquisition appear at low and intermediate budgets, which is precisely the regime the area of Equation (5.2) summarises; since the area also integrates the predictor quality, the full-budget accuracy is reported alongside where it explains a gap.

5.1.5 Hyperparameters and Implementation

All results in this chapter are obtained with the complete Deep-EC2 pipeline of Algorithm 4.5: predictor pretraining, value-network regression onto the kNN targets, and adaptation of the predictor to the learned policy. The common configuration is as follows, with dataset-specific deviations in Table 5.2. The predictor and the value network each have two hidden layers of 512 units with ReLU activations, followed by batch normalisation in the predictor and by dropout 0.3 in the value network; the kNN targets use $k = 50$ neighbours and $M = 10$ quantile bins per feature. Training uses Adam and a ReduceLROnPlateau scheduler (factor 0.5, minimum 10^{-6} , patience 10 for the predictor and 20 for the value network); the predictor is pretrained for at most 100 epochs with early stopping on the validation cross-entropy, the value network for at most 250 epochs (Metabric: 300) with the exploration ε initialised at 0.1 and multiplied by 0.2 whenever the trajectory-mean validation metric plateaus (at most 8 annealings), and the checkpoint is selected by the trajectory-mean validation metric. The predictor adaptation uses Adam with learning rate 10^{-4} , at most 50 epochs, and patience 5. On the synthetic benchmarks and CUBE-NM the value network is trained in an efficient configuration—a kNN reference of at most 10 000 instances, batch size 256, and learning rate 1.4×10^{-3} —which is also the configuration whose training cost is reported in Section 5.6; the real-world datasets use batch size 128 and learning rate 10^{-3} . For Bank Marketing the training episodes acquire at most 10 of the 20 groups, while evaluation always sweeps to the full budget.

Setting	Value
Predictor hidden size / depth	512 / 2 layers, batch normalisation
Value network hidden size / depth	512 / 2 layers, dropout 0.3 (Metabric 0.0)
kNN neighbours k / bins M	50 / 10 (CKD $k=20$; Metabric $k=15$)
kNN reference set	$\leq 10\,000$ instances (synthetic, MNIST, MiniBooNE); 5 000 (Bank); full otherwise
Batch size / value-network learning rate	256 / 1.4×10^{-3} (synthetic, CUBE-NM); 128 / 10^{-3} (CKD 64)
Optimiser	Adam, ReduceLROnPlateau (0.5, pat. 10 / 20, min 10^{-6})
Predictor / value network epochs	≤ 100 (early stop) / ≤ 250 (Metabric 300)
Exploration ε	0.1, $\times 0.2/\text{plateau}$, ≤ 8 annealings; random first step on Metabric
Predictor adaptation	Adam 10^{-4} , ≤ 50 epochs, patience 5, $R \approx B_{\max}/4$ random masks
Checkpoint monitor	trajectory-mean Acc (synthetic, CUBE-NM, Grid); AUROC (tabular)
Seeds	0–2

Table 5.2: Hyperparameters of Deep-EC2 and dataset-specific deviations.

Models are implemented in PyTorch Lightning and trained on a single NVIDIA GPU with 32-bit precision. The synthetic datasets are generated in the repository; the remaining datasets are downloaded from their public sources and pre-processed as described above. All experiments share the same three seeds.

5.2 CUBE-NM: Context-First Acquisition Beats All Baselines

Tables 5.3 and 5.4 report the normalised area under the budget–accuracy and budget–AUROC curves of Deep-EC2, the random reference, and the seven published baselines on all eleven benchmarks, as well as the RL baseline OL on the synthetic benchmarks, together with seed-level paired t -tests against Deep-EC2. The clearest result appears on CUBE-NM, the benchmark with the highest dimension (55 features) and the most explicit forward-looking structure.

Deep-EC2 attains an accuracy–AUC of 0.927 on CUBE-NM, against 0.921 for the RL baseline OL and 0.889 for the best non-RL baseline (DIME), and the paired tests find it significantly ahead of *every* baseline, by 0.006 over OL ($p = 0.010$) and by 0.038–0.147 over the non-RL methods (Table 5.3). The learned policy explains the margin (Figure 5.1): on every test instance and every seed the first acquisition is a context feature, and the first three steps are dominated by context features, after which the policy reads the feature block of the identified context, while the non-informative features are deferred to the end. A myopic rule that scores candidates by their single-step information gain ranks the context features low, because each is individually uninformative about the label. The per-budget curves (Figure 5.2) show the consequence: Deep-EC2 reaches perfect test accuracy after 7–8 acquisitions, OL after 8–9, the fastest non-RL baselines after 20 (GDFS, AACO), and the slowest after 34–48 (DIME, EDDI, iterative imputation). Budget by budget (Table 5.5), Deep-EC2 trails all seven non-RL baselines significantly at budget 2, where the context has been bought but not yet exploited, and leads all seven significantly by 0.17–0.57 at budgets 5–8; against OL, which also acquires a context feature early, it is behind by 0.01 at budget 2 (not significant) and significantly ahead by 0.06–0.17 at budgets 5–7.

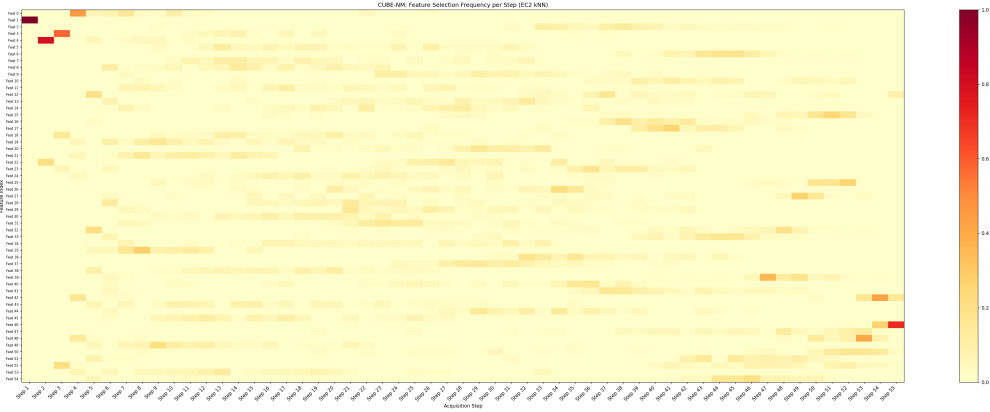


Figure 5.1: Feature-selection frequency per acquisition step on CUBE-NM (seed 0). The first acquisition is a context feature on every instance, and the first steps are dominated by context features (features 0–4).

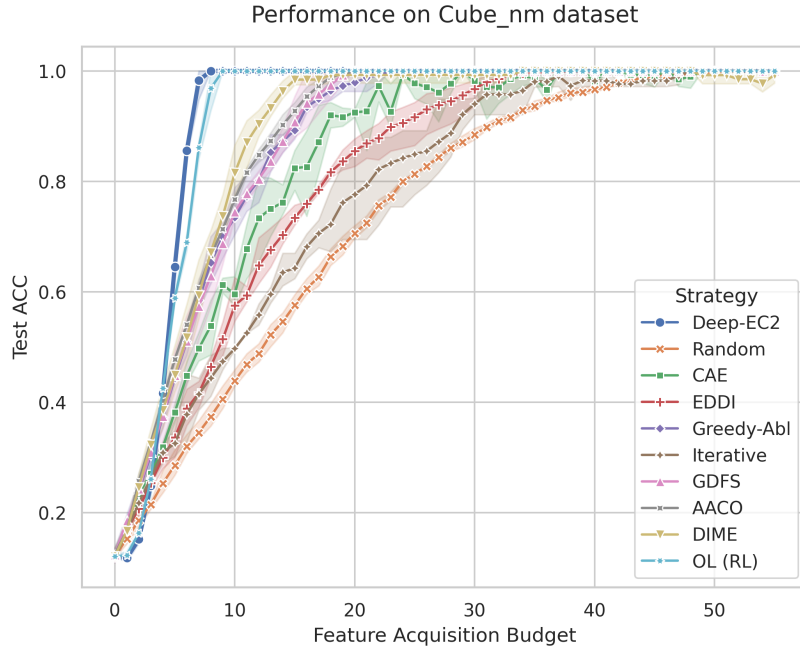


Figure 5.2: Test accuracy versus acquisition budget on CUBE-NM (three-seed mean $\pm$ seed spread).

5.3 Syn1–Syn3: Gate-First Acquisition on Syn3

On the synthetic benchmarks the forward-looking feature is the gate x_{11} , which carries no label information on its own but determines which of two feature blocks generates the label. Figure 5.3 shows the accuracy curves, Figure 5.4 the learned feature order on Syn3, and Table 5.5 the budget-by-budget comparisons on Syn3.

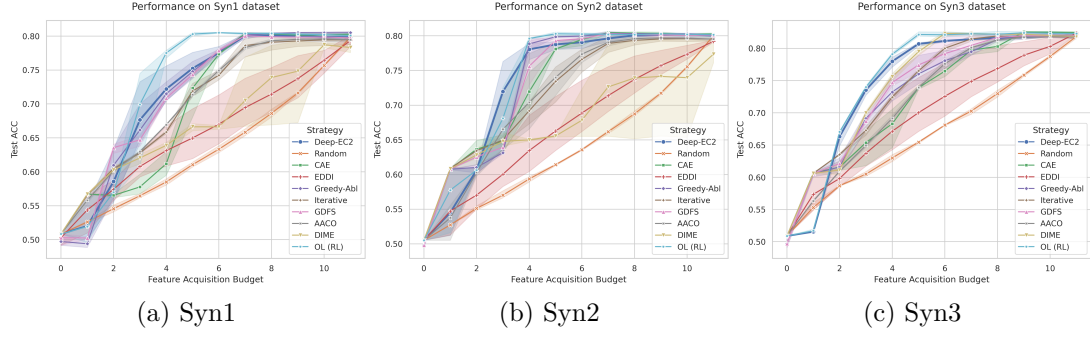


Figure 5.3: Test accuracy versus acquisition budget on the synthetic benchmarks (three-seed mean $\pm$ seed spread).

Overall accuracy. Deep-EC2 reaches an accuracy–AUC of 0.717, 0.735, and 0.749 on Syn1–Syn3 (Table 5.3). On all three tasks it is on par with the strongest non-RL baselines: on Syn3 it ties DIME (0.749) and is ahead of the other six non-RL methods, significantly so for CAE, iterative imputation, and AACO; on Syn1 it is narrowly the best non-RL method and significantly ahead of CAE and DIME, and on Syn2 it is within 0.001 of the best non-RL method (greedy ablation) and significantly ahead of CAE. The RL baseline OL is the best method on all three tasks, by 0.004 on Syn2 (not significant) and by 0.014 and 0.007 on Syn1 and Syn3.

Learned feature order on Syn3. On Syn3 the policy acquires the gate first on every test instance and every seed, and then turns to the features of the active branch (Figure 5.4): branch-1 instances ($x_{11} < 0$) continue with x_3 – x_6 , and branch-2 instances ($x_{11} \geq 0$) with x_7 and x_{10} . A myopic rule would instead start with the strongest single feature, since the gate alone says nothing about the label. On Syn2 the gate is acquired first on two of the three seeds and within the first two acquisitions on every instance, and on Syn1 it is acquired first on one of the three seeds.

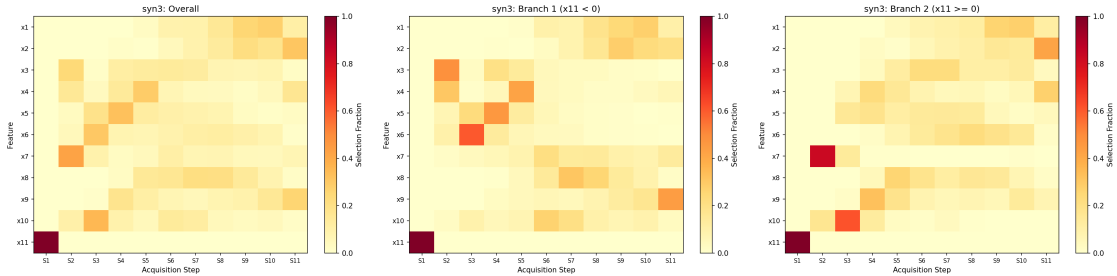


Figure 5.4: Feature-selection frequency per acquisition step on Syn3 (three seeds aggregated), overall and split by the active branch. The gate x_{11} is acquired first on every instance, after which the policy turns to the features of the active branch.

The invest-then-exploit signature. The budget-by-budget comparison on Syn3 (Table 5.5, Figure 5.5) shows the same shape as on CUBE-NM. At budget 1, where

only the gate has been acquired, Deep-EC2 is significantly behind six of the seven non-RL baselines, by up to 0.09. As soon as the first feature of the active branch has been added, the investment pays: at budget 2 Deep-EC2 is ahead of all seven by 0.03–0.07, significantly so for five of them, and at budget 3 it is ahead of all seven by 0.04–0.10, again significantly for five, remaining ahead of most baselines through the mid-budget regime before the curves converge. Against OL the differences are small at every budget (at most 0.015).

<i>CUBE-NM</i>									
Budget	Random	CAE	EDDI	Greedy-Abl	Iterative	GDFS	AACO	DIME	OL
1	−0.035*	−0.049*	−0.050*	−0.062*	−0.049	−0.068**	−0.050**	−0.049*	−0.005
2	−0.034	−0.082**	−0.055*	−0.088**	−0.066*	−0.091**	−0.105**	−0.094*	−0.012
3	+0.034*	−0.022	−0.005	−0.055	−0.014	−0.061	−0.079*	−0.074	−0.012
4	+0.163**	+0.098**	+0.117*	+0.044*	+0.107**	+0.041	+0.014	+0.030	−0.009
5	+0.359**	+0.263**	+0.309**	+0.201**	+0.320**	+0.197*	+0.167**	+0.196**	+0.057*
6	+0.535**	+0.407**	+0.467**	+0.346**	+0.477**	+0.344**	+0.315**	+0.338**	+0.166**
7	+0.637**	+0.485**	+0.566**	+0.398**	+0.568**	+0.409**	+0.375**	+0.389*	+0.121**
8	+0.626**	+0.462**	+0.535**	+0.347**	+0.556**	+0.371**	+0.334**	+0.328**	+0.031
<i>Syn3</i>									
Budget	Random	CAE	EDDI	Greedy-Abl	Iterative	GDFS	AACO	DIME	OL
1	−0.037*	−0.091**	−0.058	−0.090**	−0.090**	−0.090**	−0.048**	−0.090**	−0.001
2	+0.076*	+0.048	+0.065*	+0.047*	+0.028	+0.042*	+0.054*	+0.052*	−0.007
3	+0.131**	+0.082	+0.098*	+0.044	+0.063**	+0.048**	+0.086**	+0.036*	−0.005
4	+0.150**	+0.097*	+0.108	+0.047	+0.056**	+0.033*	+0.088**	+0.024	−0.011
5	+0.153**	+0.068**	+0.107	+0.047	+0.042**	+0.033	+0.068**	+0.012	−0.015*
6	+0.131**	+0.047*	+0.086	+0.031	+0.011	+0.020	+0.035**	−0.012*	−0.010
7	+0.111**	+0.017*	+0.065	+0.019*	−0.000	+0.010	+0.013**	−0.009**	−0.008*
8	+0.086**	+0.013	+0.048	+0.002	−0.003	−0.001	+0.003	−0.006*	−0.006

Table 5.5: Accuracy difference (Deep-EC2 – baseline) at budgets 1–8 on CUBE-NM and Syn3, three-seed means. Superscripts */** mark differences that are significant at $p < 0.05/p < 0.01$ (paired t -test, 2 d.f.). Negative values at the first budgets reflect the acquisition of the context feature or the gate; the positive values that follow reflect its exploitation.

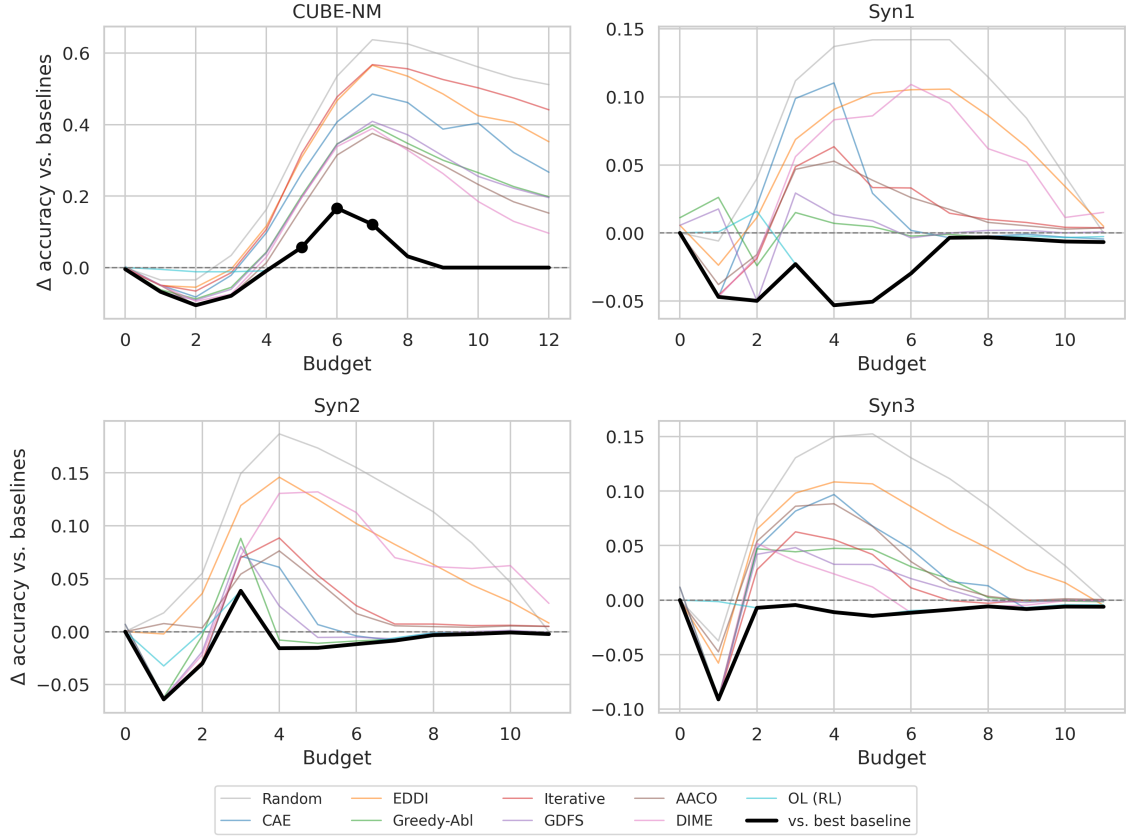


Figure 5.5: Accuracy difference between Deep-EC2 and each baseline as a function of the budget (three-seed means). Thin lines: Deep-EC2 minus the named baseline; thick line: Deep-EC2 minus the strongest baseline at each budget; filled markers: the difference to all baselines is significant (paired t -test, $p < 0.05$).

5.4 Real-World Benchmarks

On the seven real-world datasets Deep-EC2 is competitive but holds no systematic edge (Table 5.3; Figures 5.6 and 5.7). On CKD it is statistically tied with every published baseline, on Metabric it trails only greedy ablation significantly, and on Bank Marketing it is significantly ahead of EDDI, iterative imputation, and AACO and behind GDFS by 0.004. On Grid Stability it is significantly ahead of CAE and behind GDFS and AACO by 0.009–0.016. On MNIST, MiniBooNE, and California Housing five or six baselines are significantly ahead: by 0.03–0.08 on MNIST, 0.007–0.014 on MiniBooNE, and 0.03–0.07 on California Housing. In all cases Deep-EC2 is well above random acquisition with the same predictor, and its acquisition cost at inference remains that of a myopic method: a sequence of greedy forward passes of the value network, with no rollout, search, or generative-model sampling.

5. Experiments

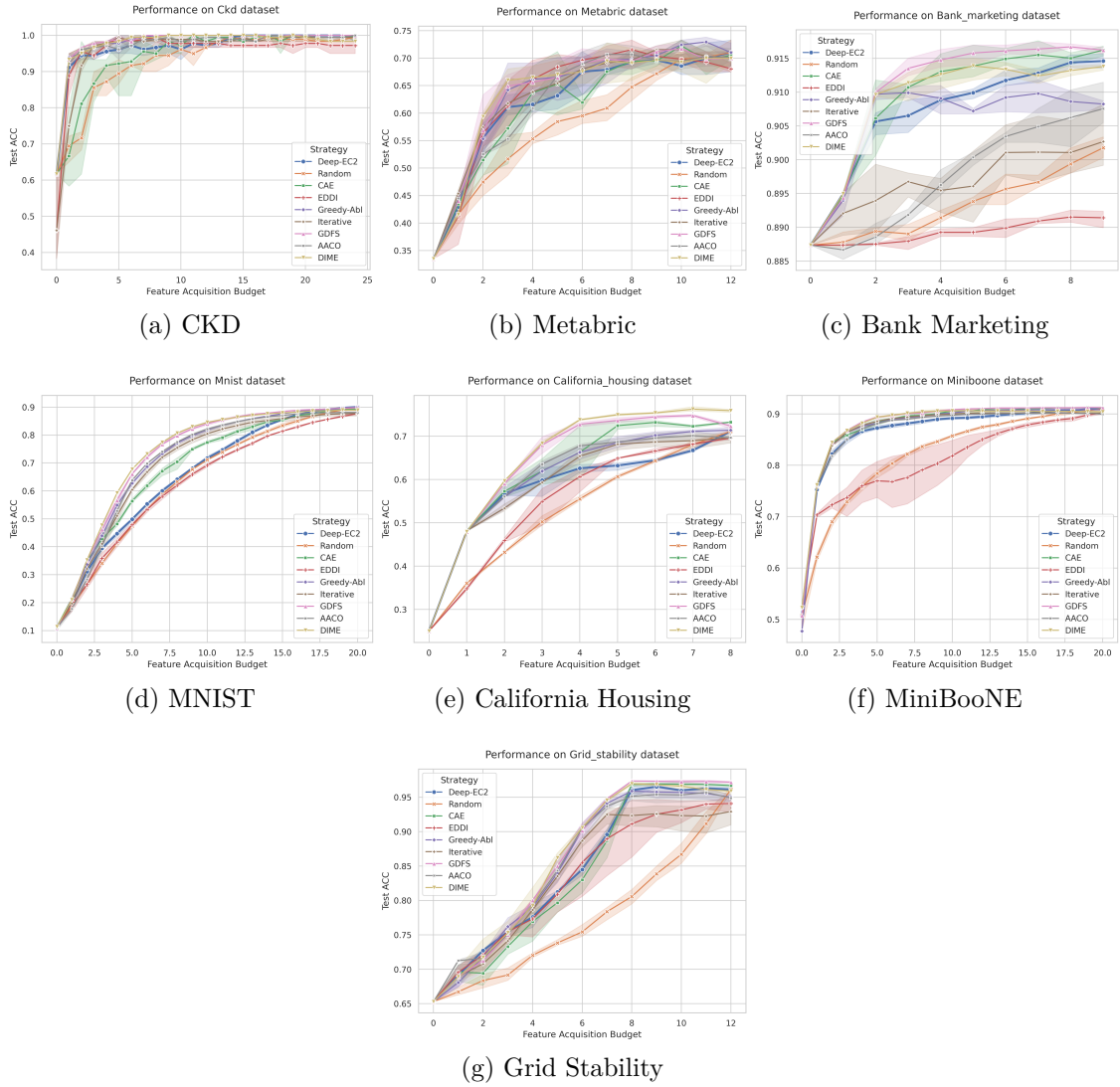


Figure 5.6: Test accuracy versus acquisition budget on the seven real-world datasets (three-seed mean $\pm$ seed spread).

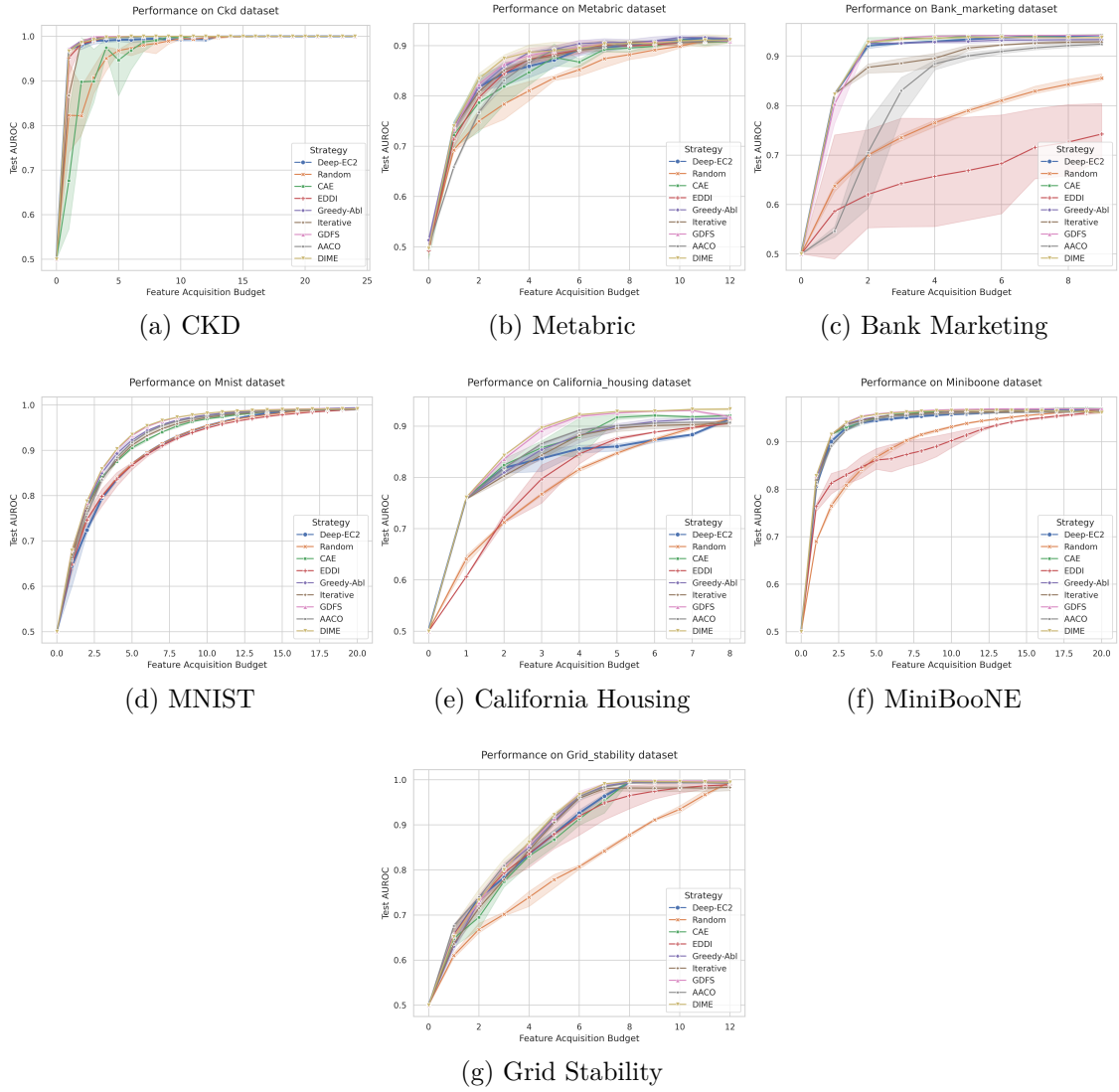


Figure 5.7: Test AUROC versus acquisition budget on the seven real-world datasets (three-seed mean $\pm$ seed spread).

5.5 Ablation: Predictor Adaptation

Table 5.6 compares Deep-EC2 with its *two-phase* variant, which omits the third training phase and evaluates the policy with the predictor pretrained on random masks.

Dataset	Two-phase (no adaptation)	Deep-EC2 (complete)	Δ	Best baseline	Sig. better / worse (complete)
Syn1	0.713 $\pm$ 0.006	0.717 $\pm$ 0.006	+0.004	0.731	2 / 1
Syn2	0.730 $\pm$ 0.001	0.735 $\pm$ 0.003	+0.005	0.739	1 / 0
Syn3	0.746 $\pm$ 0.001	0.749 $\pm$ 0.002	+0.004	0.756	3 / 1
CUBE-NM	0.926 $\pm$ 0.001	0.927$\pm$0.000	+0.001	0.921	8 / 0
Grid Stability	0.820 $\pm$ 0.007	0.847 $\pm$ 0.003	+0.027	0.863	1 / 2
CKD	0.964 $\pm$ 0.006	0.970 $\pm$ 0.007	+0.006	0.985	0 / 0
Metabric	0.618 $\pm$ 0.016	0.625 $\pm$ 0.013	+0.006	0.643	0 / 1
Bank Marketing	0.895 $\pm$ 0.002	0.908 $\pm$ 0.001	+0.013	0.912	3 / 1
MNIST	0.640 $\pm$ 0.006	0.656 $\pm$ 0.004	+0.016	0.734	1 / 6
MiniBooNE	0.867 $\pm$ 0.000	0.870 $\pm$ 0.002	+0.004	0.884	1 / 5
California Housing	0.558 $\pm$ 0.017	0.587 $\pm$ 0.007	+0.029	0.658	0 / 5

Table 5.6: Ablation of the predictor adaptation (accuracy–AUC, mean $\pm$ sample standard deviation over three seeds). Δ is the gain of the complete method over the two-phase variant. The last columns give the best baseline mean and the number of the eight baselines (seven published methods and, on the synthetic benchmarks, OL) that the complete method is significantly better / worse than (paired t -test, $p < 0.05$); bold marks a value above the best baseline.

Adapting the predictor to the policy’s trajectories improves the accuracy–AUC on all eleven benchmarks. The gains are largest where the frozen, randomly pretrained predictor was weakest even at the full budget—where all features are observed and the acquisition order is irrelevant: +0.027 on Grid Stability and +0.029 on California Housing, followed by MNIST (+0.016) and Bank Marketing (+0.013). Baselines such as GDFS and DIME train the predictor jointly with their selector and enjoy this adaptation as a by-product; for Deep-EC2 it is a separate phase that leaves the acquisition order unchanged.

5.6 Training Cost, Robustness, and Limitations

Training cost. The comparison with OL permits a direct comparison of training cost on the same cluster (one GPU per run, wall-clock time per trained model, including predictor pretraining, predictor adaptation, and evaluation for Deep-EC2; the nodes were shared, so the figures are indicative). One Deep-EC2 model took about 18 minutes on the synthetic tasks and 8 minutes on CUBE-NM, against 30 and 150 minutes for one OL model. OL additionally requires a sweep over nine hyperparameter configurations, which Deep-EC2 does not, so the total cost of producing the three-seed results was about 0.9 against 5.9 GPU-hours per synthetic task and 0.4 against 34 GPU-hours on CUBE-NM. Profiling showed that the neighbour search for the kNN targets accounts for almost 90% of the training time. Two measures address it: computing the masked distance in matrix form (Section 4.3) makes the target about 7 times faster with identical results, and bounding the kNN reference set removes the quadratic growth of the cost with the training-set size. Without these measures a synthetic-task model took about 3.5 hours, more than ten times longer, for an accuracy–AUC higher by at most 0.004; acquisition at inference is unaffected.

Sensitivity to the kNN hyperparameters. The kNN target introduces two hyperparameters, the neighbourhood size k and the number of discretisation bins M . A grid search over $k \in \{5, 20, 100\}$ and $M \in \{4, 20, 40\}$ on six tabular datasets, carried out for the two-phase variant, shows that the final-budget accuracy and AUROC are insensitive to both: all 54 configurations match the default configuration at the maximum budget, and the differences are confined to the low-budget region of the curves.

Limitations of the evaluation. Three caveats bound the conclusions. First, all comparisons rest on three seeds; the paired t -test is the strongest test available at that sample size (Section 5.1.4), but its power is low, so non-significant differences should be read as unresolved rather than as true ties. Second, CUBE-NM is evaluated in its noise-free variant; the published benchmark configuration adds observation noise, so the CUBE-NM numbers are not directly comparable to results reported elsewhere in the literature. Third, the published baselines on Bank Marketing were evaluated only up to a budget of 10 groups, so the Bank Marketing comparison covers the lower half of the Deep-EC2 budget sweep, and the RL baseline OL was run on the synthetic benchmarks only.

5. Experiments

	Deep-EC2	Random	CAE	EDDI	Greedy-Abl	Iterative	GDFS	AACO	DIME	OL (RL)
Syn1	0.717±0.006	0.631±0.003**	0.699±0.001*	0.658±0.020	0.716±0.001	0.703±0.003	0.715±0.002	0.704±0.001	0.672±0.015*	0.731±0.005 ₊
Syn2	0.735±0.003	0.633±0.001**	0.732±0.003*	0.667±0.031	0.736±0.001	0.719±0.008	0.734±0.002	0.714±0.014	0.678±0.031	0.739±0.001
Syn3	0.749±0.002	0.668±0.002**	0.726±0.003**	0.698±0.028	0.736±0.007	0.740±0.003**	0.741±0.006	0.722±0.002**	0.749±0.001	0.756±0.002 ₊
CUBE-NM	0.927±0.000	0.743±0.008**	0.842±0.005**	0.814±0.003**	0.877±0.003**	0.780±0.013**	0.879±0.007**	0.887±0.003**	0.889±0.011*	0.921±0.001 ₊
Grid Stability	0.847±0.003	0.772±0.004**	0.841±0.001*	0.834±0.019	0.855±0.003	0.838±0.008	0.863±0.005 ₊₊	0.856±0.003 ₊	0.861±0.006	–
CKD	0.970±0.007	0.930±0.003*	0.946±0.017	0.961±0.005	0.983±0.002	0.964±0.004	0.985±0.001	0.977±0.004	0.981±0.004	–
Metabric	0.625±0.013	0.583±0.007*	0.620±0.006	0.638±0.014	0.643±0.017 ₊	0.638±0.012	0.642±0.012	0.626±0.013	0.639±0.016	–
Bank Marketing	0.908±0.001	0.894±0.000**	0.910±0.000	0.889±0.000**	0.906±0.001	0.898±0.003*	0.912±0.000 ₊	0.898±0.001*	0.910±0.000	–
MNIST	0.656±0.004	0.638±0.002*	0.686±0.003 ₊	0.627±0.005**	0.716±0.004 ₊₊	0.698±0.010 ₊	0.732±0.001 ₊₊	0.708±0.001 ₊₊	0.734±0.001 ₊₊	–
MiniBooNE	0.870±0.002	0.823±0.004**	0.881±0.000 ₊	0.811±0.019*	0.878±0.000 ₊	0.877±0.000 ₊	0.884±0.000 ₊₊	0.873±0.001	0.881±0.001 ₊₊	–
California Housing	0.587±0.007	0.533±0.002**	0.626±0.002 ₊	0.554±0.010	0.613±0.006 ₊	0.599±0.007	0.649±0.004 ₊₊	0.614±0.005 ₊₊	0.658±0.000 ₊₊	–

Table 5.3: Normalised area under the budget–accuracy curve (Equation (5.2)), mean $\pm$ sample standard deviation over the three seeds. Bold marks the best mean per row. A superscript */** on a baseline cell indicates that Deep-EC2 is significantly better than that method (paired t -test, $p < 0.05/p < 0.01$, 2 d.f.); a subscript +/++ indicates that Deep-EC2 is significantly worse. OL was run on the synthetic benchmarks only.

	Deep-EC2	Random	CAE	EDDI	Greedy-Abl	Iterative	GDFS	AACO	DIME	OL (RL)
Syn1	0.789±0.008	0.690±0.003**	0.769±0.003	0.721±0.026	0.790±0.001	0.778±0.002	0.788±0.002	0.778±0.001	0.745±0.013	0.804±0.004
Syn2	0.812±0.002	0.695±0.000**	0.810±0.003	0.738±0.043	0.817±0.001 ₊	0.801±0.008	0.814±0.002 ₊	0.792±0.016	0.762±0.024	0.818±0.001 ₊
Syn3	0.818±0.002	0.728±0.002**	0.795±0.004**	0.763±0.037	0.811±0.006	0.814±0.001*	0.816±0.004	0.787±0.002**	0.823±0.001 ₊	0.826±0.002 ₊
Grid Stability	0.876±0.002	0.799±0.003**	0.868±0.005	0.868±0.016	0.883±0.000 ₊	0.874±0.003	0.888±0.005 ₊	0.889±0.001 ₊₊	0.888±0.008	–
CKD	0.985±0.001	0.964±0.002**	0.962±0.011	0.986±0.002	0.988±0.000 ₊	0.983±0.000	0.988±0.000 ₊	0.987±0.000	0.987±0.001	–
Metabric	0.855±0.006	0.824±0.010	0.844±0.004	0.852±0.004	0.864±0.003	0.856±0.003	0.861±0.007	0.845±0.003	0.863±0.007	–
Bank Marketing	0.902±0.001	0.765±0.004**	0.905±0.002	0.667±0.088*	0.898±0.002	0.882±0.004*	0.904±0.004	0.825±0.017*	0.903±0.000 ₊	–
MNIST	0.905±0.001	0.907±0.000	0.922±0.000 ₊	0.902±0.004	0.929±0.002 ₊₊	0.923±0.003 ₊	0.933±0.000 ₊	0.924±0.001 ₊₊	0.933±0.000 ₊₊	–
MiniBooNE	0.935±0.002	0.893±0.003**	0.941±0.000 ₊	0.887±0.009**	0.941±0.001	0.938±0.001	0.944±0.000 ₊	0.934±0.001	0.941±0.000 ₊	–
California Housing	0.825±0.007	0.783±0.001**	0.848±0.002 ₊	0.792±0.009*	0.842±0.005	0.836±0.005	0.863±0.003 ₊	0.844±0.001 ₊	0.866±0.000 ₊₊	–

Table 5.4: Normalised area under the budget–AUROC curve, same conventions as Table 5.3. CUBE-NM is omitted because our evaluation pipeline logs accuracy only for this benchmark.

6

Conclusion and Future Work

6.1 Conclusions

This thesis set out to obtain non-myopic active feature acquisition without reinforcement learning. The proposed framework, *Deep-EC2*, builds the acquisition objective on the EC2 criterion—the only objective suited to AFA, together with its direct extensions, that is known to be adaptive submodular—and makes it computable on continuous, high-dimensional data through a masked k -nearest-neighbour estimator with per-feature threshold discretisation. A value network regresses onto the resulting EC2 gains, and the predictor is finally adapted to the subsets that the learned policy acquires. At inference, acquisition reduces to a sequence of greedy forward passes, with no rollouts, search, or generative sampling.

The experimental evaluation, conducted on eleven benchmarks against seven published methods and the RL method Opportunistic Learning under a seed-paired protocol with significance testing, supports three conclusions.

First, Deep-EC2 acquires non-myopically. On CUBE-NM it acquires a context feature first on every instance, attains a budget–accuracy area of 0.927, is significantly ahead of every baseline including OL, and reaches perfect accuracy after 7–8 acquisitions where the non-RL baselines need 20–48. On the synthetic task Syn3 it acquires the gate first on every instance and then turns to the features of the active branch, and is on par with the strongest non-RL baselines. On both benchmarks the budget-by-budget comparison shows the invest-then-exploit signature—a significant early deficit incurred by acquiring an individually uninformative context feature or gate, followed by significant mid-budget gains.

Second, Deep-EC2 remains competitive beyond these benchmarks: on the seven real-world tabular datasets it is well above random acquisition and close to the published baselines on most of them, and adapting the predictor to the policy improved all eleven benchmarks.

Third, the framework is practical. Training is a sequence of supervised problems that converged stably on all eleven datasets with a largely shared configuration and requires no reward design or hyperparameter sweep. With an efficient implementation of the neighbour search and a bounded kNN reference set, training a model took about 18 minutes on the synthetic tasks and 8 minutes on CUBE-NM, against 30 and 150 minutes for OL, and the total cost including OL’s hyperparameter sweep

was six to eighty times lower. Acquisitions at deployment cost no more than those of a myopic method.

6.2 Limitations

Five limitations qualify these conclusions. The evaluation rests on three seeds per method; paired t -tests are the strongest statistics available at that sample size, but their power is low, and several small differences remain unresolved rather than settled. The CUBE-NM benchmark was evaluated in its noise-free variant, so those results do not transfer directly to the noisy configuration used in the published benchmark suite. The empirical study assumed uniform acquisition costs with a hard budget; the cost-normalised EC2 objective extends naturally to non-uniform costs, but this extension was not exercised experimentally. The near-optimality guarantee of Theorem 4.1 attaches to exact greedy selection on the exact EC2 utility, whereas the deployed policy is greedy with respect to value-network estimates of k NN-discretised targets; since neither approximation is bounded, the guarantee does not transfer to Deep-EC2 itself (Section 4.6); it certifies the design choice rather than the deployed policy, whose support is empirical. Finally, the RL comparison covers one method, OL, on the synthetic benchmarks only.

6.3 Future Work

Several directions follow from this work. The most direct is to extend the target beyond one step: multi-step EC2 targets, computed exactly where the training set still populates the cells of the partition or approximately elsewhere, would let the value network look further ahead than a single acquisition. Target computation is a second lever: finer or adaptive discretisation, transformations that amplify weak label signal in the neighbourhood Gini estimate, and explicit redundancy handling. A third direction is observation noise: the noisy-context configuration of CUBE-NM proved unstable for the current target estimator, and neighbourhood estimators that are robust to corrupted observations would broaden the framework’s applicability. Fourth, scale: the exact k NN search over the training set bounds the approach on very large or very high-dimensional problems, and approximate nearest-neighbour structures or learned target estimators could lift this limit. Beyond the current protocol, natural extensions include non-uniform acquisition costs with a learned stopping rule, richer predictor classes, comparisons with further RL methods, and evaluations with more seeds and per-instance tests to convert the indicative significance analysis of this thesis into confirmatory evidence.

Bibliography

- [1] L. Aronsson, A. Rahbar, and M. H. Chehreghani, “A survey on active feature acquisition strategies,” *arXiv preprint arXiv:2502.11067*, 2026, arXiv:2502.11067v2. [Online]. Available: <https://arxiv.org/abs/2502.11067v2>.
- [2] G. A. Gorry and G. O. Barnett, “Experience with a model of sequential diagnosis,” *Computers and Biomedical Research*, vol. 1, no. 5, pp. 490–507, 1968.
- [3] D. Heckerman, J. S. Breese, and K. Rommelse, “Decision-theoretic troubleshooting,” *Communications of the ACM*, vol. 38, no. 3, pp. 49–57, 1995.
- [4] D. V. Lindley, “On a measure of the information provided by an experiment,” *The Annals of Mathematical Statistics*, vol. 27, no. 4, pp. 986–1005, 1956.
- [5] R. A. Howard, “Information value theory,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 2, no. 1, pp. 22–26, 1966.
- [6] A. Krause and C. Guestrin, “Optimal value of information in graphical models,” *Journal of Artificial Intelligence Research*, vol. 35, pp. 557–591, 2009.
- [7] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [8] V. Schütz, H. Wu, R. Rezvan, L. Aronsson, and M. H. Chehreghani, “AFABench: A generic framework for benchmarking active feature acquisition,” *arXiv preprint arXiv:2508.14734*, 2025. [Online]. Available: <https://arxiv.org/abs/2508.14734>.
- [9] C. Ma, S. Tschitschek, K. Palla, J. M. Hernandez-Lobato, S. Nowozin, and C. Zhang, “EDDI: Efficient dynamic discovery of high-value information with partial VAE,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 4234–4243. [Online]. Available: <https://proceedings.mlr.press/v97/ma19c.html>.
- [10] I. Covert, W. Qiu, M. Lu, N. Y. Kim, N. White, and S.-I. Lee, “Learning to maximize mutual information for dynamic feature selection,” in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 202, PMLR, 2023, pp. 6424–6447. [Online]. Available: <https://proceedings.mlr.press/v202/covert23a.html>.
- [11] S. Gadgil, I. Covert, and S.-I. Lee, “Estimating conditional mutual information for dynamic feature selection,” in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: <https://openreview.net/forum?id=0ju2Qu9jvn>.

- [12] P. D. Turney, “Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm,” *Journal of Artificial Intelligence Research*, vol. 2, pp. 369–409, 1995.
- [13] C. X. Ling, V. S. Sheng, and Q. Yang, “Test strategies for cost-sensitive decision trees,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 8, pp. 1055–1067, 2006.
- [14] Z. Xu, K. Q. Weinberger, and O. Chapelle, “The greedy miser: Learning under test-time budgets,” in *Proceedings of the 29th International Conference on Machine Learning*, 2012.
- [15] F. Nan, J. Wang, and V. Saligrama, “Feature-budgeted random forest,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37, PMLR, 2015, pp. 1983–1991. [Online]. Available: <https://proceedings.mlr.press/v37/nan15.html>.
- [16] G. Dulac-Arnold, L. Denoyer, P. Preux, and P. Gallinari, “Datum-wise classification: A sequential approach to sparsity,” in *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, ser. Lecture Notes in Computer Science, vol. 6911, Springer, 2011, pp. 375–390.
- [17] T. Rückstieß, C. Osendorfer, and P. van der Smagt, “Sequential feature selection for classification,” in *AI 2011: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, vol. 7106, Springer, 2011, pp. 132–141.
- [18] H. Shim, S. J. Hwang, and E. Yang, “Joint active feature acquisition and classification with variable-size set encoding,” in *Advances in Neural Information Processing Systems*, vol. 31, 2018. [Online]. Available: <https://proceedings.neurips.cc/paper/2018/hash/e5841df2166dd424a57127423d276bbe-Abstract.html>.
- [19] M. Kachuee, O. Goldstein, K. Kärkkäinen, S. Darabi, and M. Sarrafzadeh, “Opportunistic learning: Budgeted cost-sensitive learning from data streams,” in *International Conference on Learning Representations*, 2019. [Online]. Available: <https://openreview.net/forum?id=S1e0Ho09KX>.
- [20] J. Janisch, T. Pevný, and V. Lisý, “Classification with costly features using deep reinforcement learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3959–3966. DOI: 10.1609/aaai.v33i01.33013959. [Online]. Available: <https://arxiv.org/abs/1711.07364>.
- [21] J. Janisch, T. Pevný, and V. Lisý, “Classification with costly features as a sequential decision-making problem,” *Machine Learning*, vol. 109, no. 8, pp. 1587–1615, 2020. [Online]. Available: <https://arxiv.org/abs/1909.02564>.
- [22] S. Zannone, J. M. Hernández-Lobato, C. Zhang, and K. Palla, “ODIN: Optimal discovery of high-value INformation using model-based deep reinforcement learning,” in *ICML 2019 Workshop on Real-world Sequential Decision Making: Reinforcement Learning and Beyond*, 2019.
- [23] Z. Chen, J. Huang, H. Ahn, and X. Ning, “Costly features classification using monte carlo tree search,” in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.
- [24] D. Golovin, A. Krause, and D. Ray, “Near-optimal bayesian active learning with noisy observations,” in *Advances in Neural Information Processing Sys-*

- tems, Extended version: arXiv:1010.3091v2, vol. 23, 2010, pp. 766–774. [Online]. Available: <https://arxiv.org/abs/1010.3091>.
- [25] A. Rahbar, Z. Ye, Y. Chen, and M. H. Chehreghani, “Efficient online decision tree learning with active feature acquisition,” in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI-23)*, 2023, pp. 4163–4171. DOI: 10.24963/ijcai.2023/463. [Online]. Available: <https://arxiv.org/abs/2305.02093>.
 - [26] Y. Chen, S. H. Hassani, A. Karbasi, and A. Krause, “Sequential information maximization: When is greedy near-optimal?” In *Proceedings of the 28th Conference on Learning Theory*, ser. Proceedings of Machine Learning Research, vol. 40, PMLR, 2015, pp. 338–363. [Online]. Available: <https://proceedings.mlr.press/v40/Chen15b.html>.
 - [27] F. Nan, J. Wang, K. Trapeznikov, and V. Saligrama, “Fast margin-based cost-sensitive classification,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 2952–2956. DOI: 10.1109/icassp.2014.6854141.
 - [28] M. Valancius, M. Lennon, and J. B. Oliva, “Acquisition conditioned oracle for nongreedy active feature acquisition,” in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235, PMLR, 2024, pp. 48957–48975. [Online]. Available: <https://proceedings.mlr.press/v235/valancius24a.html>.
 - [29] H.-T. Huang, D. Dinh, and J. B. Oliva, “Information templates: A new paradigm for intelligent active feature acquisition,” *arXiv preprint arXiv:2508.18380*, 2025.
 - [30] D. Geman and B. Jedynak, “An active testing model for tracking roads in satellite images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 1, pp. 1–14, 1996.
 - [31] X. Chai, L. Deng, Q. Yang, and C. X. Ling, “Test-cost sensitive naive bayes classification,” in *Fourth IEEE International Conference on Data Mining (ICDM’04)*, 2004, pp. 51–58.
 - [32] T. Rückstieß, C. Osendorfer, and P. van der Smagt, “Minimizing data consumption with sequential online feature selection,” *International Journal of Machine Learning and Cybernetics*, vol. 4, no. 3, pp. 235–243, 2013.
 - [33] V. Mnih et al., “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
 - [34] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
 - [35] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (Adaptive Computation and Machine Learning), 2nd. The MIT Press, 2018.
 - [36] D. Golovin and A. Krause, “Adaptive submodularity: Theory and applications in active learning and stochastic optimization,” *Journal of Artificial Intelligence Research*, vol. 42, pp. 427–486, 2011.
 - [37] A. Krause and D. Golovin, “Submodular function maximization,” in *Tractability: Practical Approaches to Hard Problems*, Cambridge University Press, 2014, pp. 71–104.

- [38] Y. Chen, J.-M. Renders, M. H. Chehreghani, and A. Krause, “Efficient on-line learning for optimizing value of information: Theory and application to interactive troubleshooting,” in *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, ser. Proceedings of Machine Learning Research, vol. R15, 2017, pp. 311–320. [Online]. Available: <https://arxiv.org/abs/1703.05452>.
- [39] A. Rahbar, N. Åkerblom, and M. H. Chehreghani, “Cost-efficient online decision making: A combinatorial multi-armed bandit approach,” *Transactions on Machine Learning Research*, 2025. [Online]. Available: <https://arxiv.org/abs/2308.10699>.
- [40] N. Beebe-Wang, W. Qiu, and S.-I. Lee, “Explanation-guided dynamic feature selection for medical risk prediction,” in *ICML 3rd Workshop on Interpretable Machine Learning in Healthcare (IMLH)*, 2023.
- [41] A. Mirzaei, V. Pourahmadi, H. Sheikhzadeh, and A. Abdollahpourroostam, “Fast classification with sequential feature selection in test phase,” *arXiv preprint arXiv:2306.14347*, 2023.
- [42] A. Norcliffe, C. Lee, F. Imrie, M. van der Schaar, and P. Liò, “Stochastic encodings for active feature acquisition,” in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267, PMLR, 2025, pp. 46 784–46 814. [Online]. Available: <https://proceedings.mlr.press/v267/norcliffe25a.html>.
- [43] J. Yoon, J. Jordon, and M. van der Schaar, “INVASE: Instance-wise variable selection using neural networks,” in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=BJg_roAcK7.