# From Feature Selection to Instance-wise Feature Acquisition[1]

## Tutorial @ SDM 2024

**Daphney-Stavroula Zois**[1] **Charalampos Chelmis**[2]

[1] *Electrical and Computer Engineering Department*

[2] Computer Science Department

Saturday, April 20th, 2024

[1] This research is based upon work supported by the National Science Foundation grants ECCS-1737443 & CNS-1942330 and a Google AI for Social Good award.

# Instance–wise Feature Acquisition

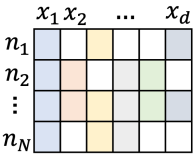
# Problem Definition

- Features may not be readily available (either during training or testing) because of acquisition costs (e.g., medical tests, energy constraints)
- Goal: balance tradeoff between accrued acquisition cost and accuracy
- Two major classes of methods:
  - Active feature acquisition [MSTPM05, AMPST11, GTN$^+$19]
  - Classification with costly features (also known as dynamic instance–wise feature selection or instance–wise feature acquisition) [DADPG12, JPL20, LZC21b, CQL$^+$23]

# Active Feature Acquisition

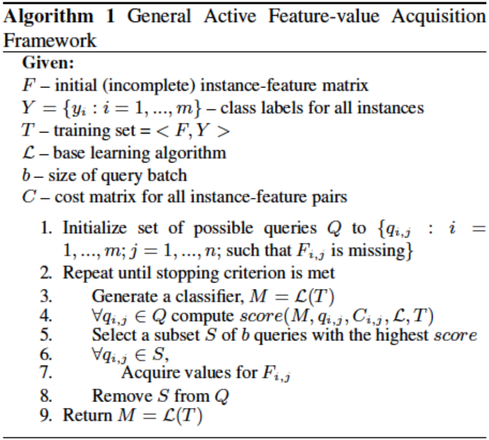
- Goal: acquire missing feature values at training time to improve classification model accuracy
- Learner may acquire value of $j$th feature $F_{i,j}$ of $i$th data instance at cost $C_{i,j}$

**Training:** subset of features are available upfront for some instances and all features are available upfront for others
**Testing:** all features are available and used for classification



# Active Feature Acquisition [MSTPM05]

- Goal: select instance–feature queries that will result in building most accurate model at lowest cost



**Algorithm 1** General Active Feature-value Acquisition Framework

```
Given:
F – initial (incomplete) instance-feature matrix
$Y = \{y_i : i = 1, ..., m\}$ – class labels for all instances
$T$ – training set = $< F, Y >$
$\mathcal{L}$ – base learning algorithm
$b$ – size of query batch
$C$ – cost matrix for all instance-feature pairs

1. Initialize set of possible queries $Q$ to $\{q_{i,j} : i = 1, ..., m; j = 1, ..., n$; such that $F_{i,j}$ is missing$\}$
2. Repeat until stopping criterion is met
3.     Generate a classifier, $M = \mathcal{L}(T)$
4.     $\forall q_{i,j} \in Q$ compute $score(M, q_{i,j}, C_{i,j}, \mathcal{L}, T)$
5.     Select a subset $S$ of $b$ queries with the highest $score$
6.     $\forall q_{i,j} \in S$,
7.         Acquire values for $F_{i,j}$
8.     Remove $S$ from $Q$
9. Return $M = \mathcal{L}(T)$
```

# Active Feature Acquisition [MSTPM05]

- Iterative procedure until stopping criterion is met, e.g., desirable accuracy has been obtained
- Expected utility of query: improvement in model accuracy per unit cost

$$E(q_{i,j}) = \sum_{k=1}^{K} P(F_{i,j} = V_k)\mathcal{U}(F_{i,j} = V_k)$$

$$\mathcal{U}(F_{i,j} = V_k) = \frac{\mathcal{A}(F, F_{i,j} = V_k) - \mathcal{A}(F)}{C_{i,j}}$$

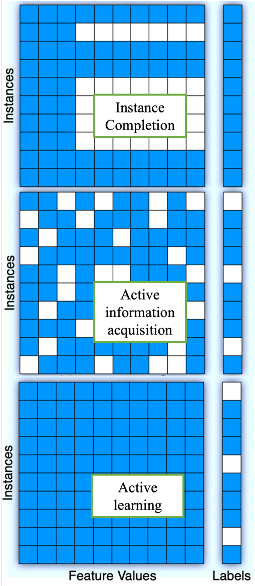
- $P(F_{i,j} = V_k)$ and $\mathcal{A}$ are unknown and estimated from training data
- Computing scores for all queries and identifying subset with highest score can be computationally expensive → randomly subsample queries to compute expected utility

# Selective Data Acquisition for ML [AMPST11]

- Other measures may be more effective, e.g., Log Gain promotes acquisitions which increase likelihood of correct class prediction

$$LG(x_i) = -\sum_{k=1}^{K} I(c_k, x_k) \log \hat{P}(c_k|x_i)$$

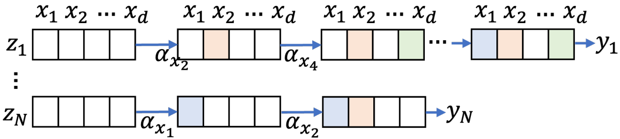
- Variations:
  - Instance completion: same subset of feature values are known for all instances, and subset of all remaining feature values can be acquired at fixed cost
  - Active information acquisition: both features and labels are missing at training
  - Active learning: all features are available, but labels are missing at training



# Instance–wise Feature Acquisition

- Goal: sequentially gather subset of features (unique for each data instance) during testing, before classifying it

**Training:** all candidate features are typically available upfront
**Testing:** features are acquired typically one at a time by expending cost



# Instance–wise Feature Acquisition Formulations

- Sequential decision–making mathematical frameworks (Markov Decision Process (MDP), Partially Observable MDP (POMDP)) [AZ04, BZD05, JC07, DADPG12, HDIE12, TS13, WTS14, SHY18, JPL20, CHAN21, LO21, LZC21b, GL23]
- Bayesian decision theoretical frameworks [CDYL04, CGD07]
- Algorithmic approaches (e.g., decision trees) [SL06, XKW$^{+}$14]

# Instance–wise Feature Acquisition

Based on above approaches, policies/decision mechanisms (i.e., which feature value to acquire next) have been derived

- AO$^*$ algorithm [AZ04, BZD05]
- Imitation learning [HDIE12]
- Empirical risk minimization [TS13] & linear programming [WTS14, WBTS14]
- Neural networks [CDA16]
- Greedy methods [JC07, MTP$^+$19, CQL$^+$23, GCL23]
- Reinforcement learning (RL) [DADPG12, SHY18, MOK$^+$19, ZHLZP19, JPL20, CHAN21, GL23]
- Bridging gap between greedy and RL [LO21]
- Probabilistic Circuits [KN23]

# Learning Datum–wise Sparse Representations [DADPG11, DADPG12]

- Goal: limit number of features per data instance to improve classification speed and prevent overfitting
  - Easy–to–classify data instances can be classified with looking at few features
  - More difficult data instances can be classified using more features
- In the context of supervised multi–class classification, learn datum–wise classification function $f_\theta(\mathbf{x}) = (y, \mathbf{z})$ of parameters $\theta$
  - $y$: predicted output
  - $\mathbf{z} = (z^1, \ldots, z^n)$: $z^i = 1$ implies that feature $i$ has been taken into consideration for computing label $y$ on datum $\mathbf{x}$

$$\theta^* = \arg\min_\theta \frac{1}{N} \sum_{i=1}^{N} \Delta(y_\theta(\mathbf{x}_i), y_i) + \lambda \frac{1}{N} \sum_{i=1}^{N} ||z_\theta(\mathbf{x}_i)||_0$$

→ combinatorial problem!

# Datum–wise Sparse Sequential Classification

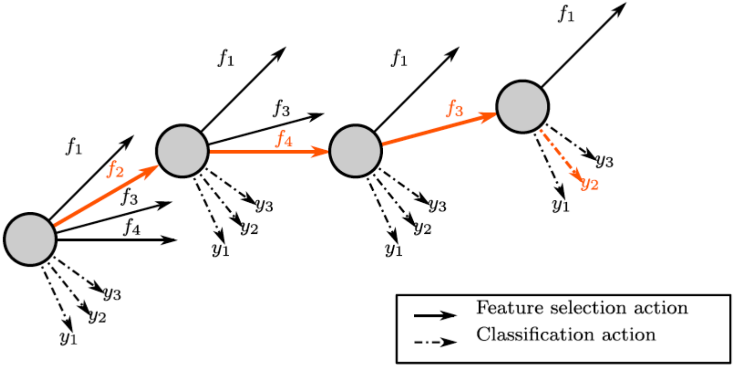


Figure 2: Sequential process for problem with $4$ features $(f_1, f_2, f_3, f_4)$ and $3$ possible labels $(y_1, y_2, y_3)$

# Markov Decision Process (MDP) formulation

- Markov decision process (MDP): mathematical framework $(S, A, P_a, R_a)$ for modeling decision making when outcomes are partly random and partly under control of decision maker
- Goal: MDP to classify data instance $\mathbf{x}$
  - Initially, we have no information about $\mathbf{x}$ (i.e., no features)
  - At each step, we can choose to acquire particular feature of $\mathbf{x}$ or to classify $\mathbf{x}$
- State $(S)$: features already selected
  Action $(A)$: feature selection or assign label
  Transition function $(P_a)$: only defined for feature selection actions
  Reward $(R_a)$: negative of $0-1$ loss for assigning label, and negative of feature selection fixed cost

# Markov Decision Process (MDP) formulation

- Find good policy for decision maker
  - Determine function $\pi_\theta$ that decision maker will choose when in state $s$ to maximize overall reward
- Optimal classifier $\theta^*$ corresponds to the optimal MDP policy

$$\begin{aligned}\theta^* &= \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N} \Delta(y_\theta(\mathbf{x}_i), y_i) + \lambda \frac{1}{N}\sum_{i=1}^{N} ||z_\theta(\mathbf{x}_i)||_0 \\ &= \arg\max_\theta \frac{1}{N}\sum_{i=1}^{N} \sum_{t=1}^{T_\theta(\mathbf{x}_i)+1} r(\mathbf{x}_i, z_\theta^{(t)}(\mathbf{x})_i, \pi_\theta(\mathbf{x}_i, z_\theta^{(t)}))\end{aligned}$$

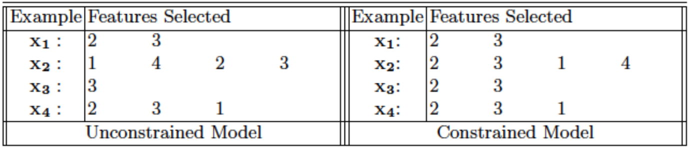
- Use of classical MDP solution algorithms to find best classifier $\theta^*$

# Proposed Solution

- Two main challenges:
  - Number of states is infinite
  - Reward function is only known for values of $\mathbf{x}$ in training set

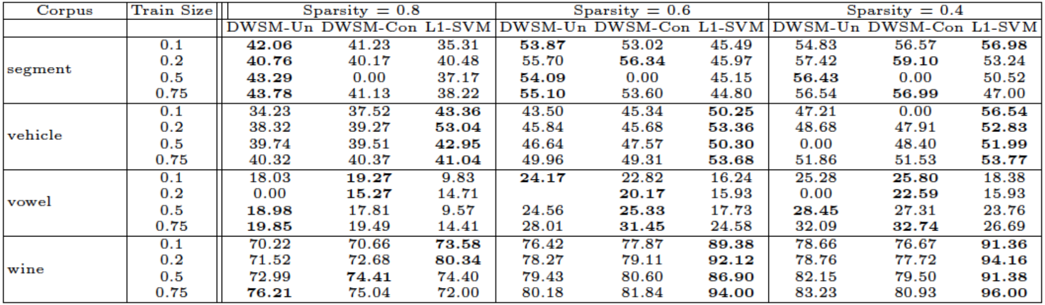
→ impossible to compute score function for all state–action pairs in tabular manner

- Outline:
  - Linear approximation of value function of MDP
  - Monte–Carlo approach to sample example states from learning space during training
  - Variant that considers feature selection in same order



| Example | Features Selected | | | | Example | Features Selected | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\mathbf{x_1}$ : | 2 | 3 | | | $\mathbf{x_1}$: | 2 | 3 | | |
| $\mathbf{x_2}$ : | 1 | 4 | 2 | 3 | $\mathbf{x_2}$: | 2 | 3 | 1 | 4 |
| $\mathbf{x_3}$ : | 3 | | | | $\mathbf{x_3}$: | 2 | 3 | | |
| $\mathbf{x_4}$ : | 2 | 3 | 1 | | $\mathbf{x_4}$: | 2 | 3 | 1 | |
| Unconstrained Model | | | | | Constrained Model | | | | |

# Some Results



| Corpus | Train Size | Sparsity = 0.8 | | | Sparsity = 0.6 | | | Sparsity = 0.4 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | DWSM-Un | DWSM-Con | L1-SVM | DWSM-Un | DWSM-Con | L1-SVM | DWSM-Un | DWSM-Con | L1-SVM |
| segment | 0.1 | **42.06** | 41.23 | 35.31 | **53.87** | 53.02 | 45.49 | 54.83 | 56.57 | **56.98** |
| | 0.2 | **40.76** | 40.17 | 40.48 | 55.70 | **56.34** | 45.97 | 57.42 | **59.10** | 53.24 |
| | 0.5 | **43.29** | 0.00 | 37.17 | **54.09** | 0.00 | 45.15 | **56.43** | 0.00 | 50.52 |
| | 0.75 | **43.78** | 41.13 | 38.22 | **55.10** | 53.60 | 44.80 | 56.54 | **56.99** | 47.00 |
| vehicle | 0.1 | 34.23 | 37.52 | **43.36** | 43.50 | 45.34 | **50.25** | 47.21 | 0.00 | **56.54** |
| | 0.2 | 38.32 | 39.27 | **53.04** | 45.84 | 45.68 | **53.36** | 48.68 | 47.91 | **52.83** |
| | 0.5 | 39.74 | 39.51 | **42.95** | 46.64 | 47.57 | **50.30** | 0.00 | 48.40 | **51.99** |
| | 0.75 | 40.32 | 40.37 | **41.04** | 49.96 | 49.31 | **53.68** | 51.86 | 51.53 | **53.77** |
| vowel | 0.1 | 18.03 | **19.27** | 9.83 | **24.17** | 22.82 | 16.24 | 25.28 | **25.80** | 18.38 |
| | 0.2 | 0.00 | **15.27** | 14.71 | | **20.17** | 15.93 | 0.00 | **22.59** | 15.93 |
| | 0.5 | **18.98** | 17.81 | 9.57 | 24.56 | **25.33** | 17.73 | **28.45** | 27.31 | 23.76 |
| | 0.75 | **19.85** | 19.49 | 14.41 | 28.01 | **31.45** | 24.58 | 32.09 | **32.74** | 26.69 |
| wine | 0.1 | 70.22 | 70.66 | **73.58** | 76.42 | 77.87 | **89.38** | 78.66 | 76.67 | **91.36** |
| | 0.2 | 71.52 | 72.68 | **80.34** | 78.27 | 79.11 | **92.12** | 78.76 | 77.72 | **94.16** |
| | 0.5 | 72.99 | **74.41** | 74.40 | 79.43 | 80.60 | **86.90** | 82.15 | 79.50 | **91.38** |
| | 0.75 | **76.21** | 75.04 | 72.00 | 80.18 | 81.84 | **94.00** | 83.23 | 80.93 | **96.00** |

Figure 3: Multi–class classification accuracy on three levels of sparsity for segment ($19$ features; $7$ classes), vehicle ($18$ features; $4$ classes), vowel ($19$ features; $11$ classes), and wine ($13$ features; $3$ classes) datasets.

- Experiments on datasets with maximum number of $60$ features since learning is quadratic wrt to number of features

# Extensions [DADPG12]

- Hard budget feature selection: fixed per–datum hard budget during inference

$$\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N}\Delta(y_\theta(\mathbf{x}_i), y_i) + \lambda\frac{1}{N}\sum_{i=1}^{N}||z_\theta(\mathbf{x}_i)||_0 \quad \text{subject to} \quad ||z_\theta(\mathbf{x}_i)||_0 \leqslant M$$

- Cost–sensitive feature acquisition and classification: fixed cost to each feature and misclassification cost depends on error made

$$\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N}\Delta(y_\theta(\mathbf{x}_i), y_i) + \lambda\frac{1}{N}\sum_{i=1}^{N}\langle\xi, z_\theta(\mathbf{x}_i)\rangle$$

# Extensions [DADPG12]

- Group feature selection: choose certain number of groups of features, but not individual features

$$\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N}\Delta(y_\theta(\mathbf{x}_i), y_i) + \lambda\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{g}\mathbb{1}(\mathcal{F}_t \subset \mathcal{Z}_\theta(\mathbf{x}_i))$$

  → minimize number $\mathcal{F}_t$ groups present in actual set of selected features

- Relational feature selection: features organized in complex structures, e.g., *subset of features to be selected depend on previously acquired features* (conditional features) or *cost of acquiring of subset of features depends on previously acquired features* (constrained features)

$$\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N}\Delta(y_\theta(\mathbf{x}_i), y_i) + \frac{1}{N}\sum_{i=1}^{N}\sum_{f,f'\in\mathcal{Z}_{\theta(\mathbf{x}_i)}}\text{Related}(f,f')(\lambda-\gamma)+\gamma$$

# Dropping Linear Approximation [JPL19, JPL20]

- Main idea: replace linear approximation of MDP value function with neural networks

$$Q^*(s,a) = \mathbb{E}_{s' \sim t(s,a)} \left[ r(s,a,s') + \gamma \max_{a'} Q^*(s',a') \right]$$

→ neural network estimates $Q^\theta(s,a)$ jointly for all actions by minimizing MSE

$$\ell_\theta(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{(s_t,a_t,r_t,s_{t+1}) \in \mathcal{B}} (q_t - Q^\theta(s_t,a_t))^2$$

- Deep reinforcement learning techniques to stabilize and speed–up learning
  - Deep $Q$–learning: separate target network with parameters $\phi$ and follows parameters $\theta$ with delay [LHP+15]

$$\phi = (1-\rho)\phi + \rho\theta$$

$$q_t = r_t + \max_a \gamma Q^\phi(s_{t+1}, a)$$

# Dropping Linear Approximation [JPL19, JPL20]

- Double $Q$–learning: combine estimates $Q^\theta$ and $Q^\phi$ to reduce bias due to max operator [VHGS16]

$$q_t = r_t + \max_a \gamma Q^\phi(s_{t+1}, \arg\max_a Q^\theta(s_{t+1}, a))$$

- Dueling architecture: decompose $Q$–function into value and advantage functions to accelerate and stabilize training [WSH+16]

$$Q^\theta(s,a) = V^\theta(s) + A^\theta(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A^\theta(s,a')$$

# Dropping Linear Approximation [JPL19, JPL20]

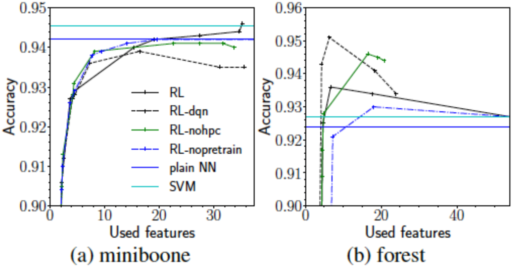
- Retrace: efficiently utilize long traces of experience with truncated importance sampling [MSHB16]

$$q_t = r_t + \gamma \mathbb{E}_{\alpha \sim \pi_\theta(s_t)} \left[ Q^\phi(s^{t+1}, a) \right] + \gamma \bar{\rho}_{t+1} \left[ q_{t+1} - Q^\phi(s_{t+1}, a_{t+1}) \right]$$

$$\bar{\rho}_{t+1} = \min \left( \frac{\pi(a_{t+1} | s_{t+1})}{\mu(a_{t+1} | s_{t+1})}, 1 \right),$$

where $\bar{\rho}_{t+1}$ is truncated importance sampling between exploration policy $\mu$ used when trajectory was sampled and current policy $\pi$

# Some Results



(a) miniboone

(b) forest

Figure 4: Performance of different versions of proposed algorithm for miniboone ($50$ features, $2$ classes) and forest ($54$ features, $7$ classes) datasets.

- Pretraining $Q$–values of classification actions and forwarding data instance to high–performance classifier that uses all features improves performance
- Experiments on datasets with maximum number of $784$ features

# Extensions [JPL20]

- Average budget with specific target $b$:

$$\min_{\theta} \mathbb{E}\left[\ell(y_\theta, y)\right] \quad \text{s.t. } \mathbb{E}\left[z_\theta(x)\right] \leq b \rightarrow$$

$$\max_{\lambda \geq 0} \min_{\theta} \mathbb{E}\left[\ell(y_\theta, y) + \lambda(z_\theta(x) - b)\right]$$

  Main idea: $\rightarrow$ iteratively perform gradient ascent in $\lambda$ and descend in $\theta$

  1. For fixed $\theta$, optimize $\lambda$ using gradient $\mathbb{E}\left[z_\theta(x) - b\right]$
  2. For fixed $\lambda$, apply reinforcement learning as before

- Missing features in training set: feature–selecting action is available only if corresponding feature is present and updates are made only with estimates of available actions

# Active Feature Acquisition w/ Generative Surrogate Models [LO21]

- Main idea: reformulate MDP as generative modeling task and optimize policy via model–based approach
- Outline:
  - Learn generative surrogate model (GSM) that captures dependencies among features $p(y, x_j | x_o)$
  - Use GSM to provide intermediate rewards

$$r_m(s, i) = H(y|x_o) - \gamma H(y|x_o, x_i) \quad \text{(information gain)}$$

  - Use GSM to provide side information, i.e.,
    → Confidence: current prediction $\hat{y}$ and likelihood $p(y|x_o)$
    → Imputed values & uncertainties of unobserved features to guide exploration
    → Utility of feature $i$: expected information gain

# Active Feature Acquisition w/ Generative Surrogate Models [LO21]

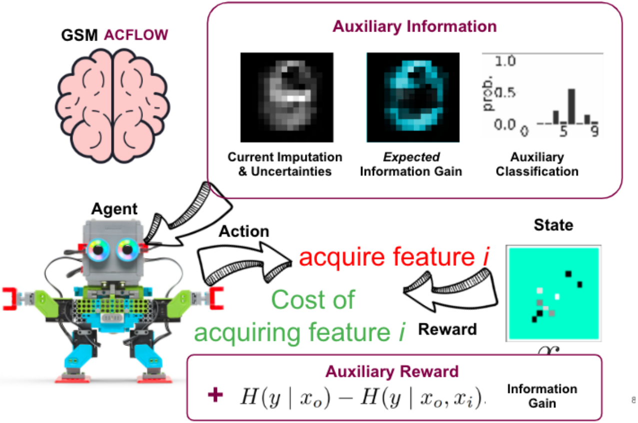
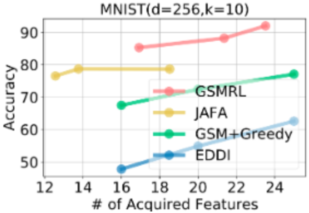


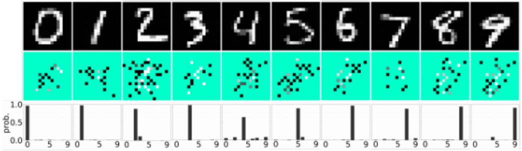
*Figure courtesy of Dr. Junier Oliva, Computer Science Department, University of North Carolina at Chapel Hill*

# Active Feature Acquisition w/ Generative Surrogate Models [LO21]

- Proposed approach is non–greedy
- Surrogate model is agnostic to feature acquisition policy → build prediction model $f_\theta(\cdot)$ that takes both current state $x_o$ and side information as inputs
- Prediction model is simultaneously trained with policy, and weight sharing between them learns better representations

# Some Results[2]





- Efficient Dynamic Discovery of High-Value Information with Partial VAE (EDDI) [MTP+19] ← greedy acquisition based on VAE
- Joint Active Feature Acquisition and Classification with Variable-Size Set Encoding (JAFA) [SHY18] ← plain reinforcement learning optimization with $Q$–learning

[2] *Slide courtesy of Dr. Junier Oliva, Computer Science Department, University of North Carolina at Chapel Hill*

# Enabling Instance–wise Feature Acquisition Orders [LZ21]

- Features are sequentially acquired one at a time
- Goal: jointly determine the order by which features should be acquired, the number of features to acquire, and the classification strategy to be used for each data instance during testing
- Our prior work [LZC21b, LZC21a] studied instance–wise feature acquisition when the order by which featured are reviewed is fixed and common to all data instances

# Problem Description

- $F \triangleq \{F_1, F_2, \dots, F_K\}$: set of features
- $K$: total number of features
- $C \in \{c_1, \dots, c_L\}$: class variable
- $e(F_k)$: cost of acquiring feature $F_k$
- $Q_{ij}$: cost of selecting class $c_j$ when class $c_i$ is true

# Optimization Setup

- Introduce random variables
  - $\sigma$: order by which features are acquired
    e.g., If $K = 3$, then $\sigma = (F_3, F_1, F_2)$ is a valid order
  - $\sigma(R) \in \{0, \ldots, K\}$: last feature acquired before classification decision
    e.g., $\sigma(R = 2) = F_2$ means stop after acquiring feature $F_2$
  - $D_{\sigma(R)} \in \{1, \ldots, L\}$: classification decision for data instance under consideration based on $\sigma(R)$ features
    e.g., $\{D_{\sigma(R=1)} = 1\}$ deciding in favor of class $c_1$ based on $\{F_3, F_1\}$

$$\text{min}_{\sigma,\sigma(R),D_{\sigma(R)}} J(\sigma, \sigma(R), D_{\sigma(R)})$$

$$J(\sigma, \sigma(R), D_{\sigma(R)}) = \mathbb{E}\left\{ \sum_{k=1}^{R} e(F_{\sigma(k)}) + \sum_{j=1}^{L} \sum_{i=1}^{L} Q_{ij} P(D_{\sigma(R)} = j, \mathcal{C} = c_i) \right\}$$

# Optimum Solution Outline

- $\pi_{\sigma(k)} \triangleq [\pi^1_{\sigma(k)}, \dots, \pi^L_{\sigma(k)}]$: posterior prob vector w/ $\pi^i_{\sigma(k)} \triangleq P(\mathcal{C} = c_i | F_{\sigma(1)}, \dots, F_{\sigma(k)})$
- Optimum feature acquisition strategy via dynamic programming
  - Last stage

$$\bar{J}_K(\pi_{\gamma_K}) = g(\pi_{\gamma_K}),$$

  - Any intermediate stage

$$\begin{aligned} \bar{J}_k(\pi_{\gamma_k}) &= \min\left[g(\pi_{\gamma_k}), \bar{\mathcal{A}}_k(\pi_{\gamma_k})\right] \\ g\big(\pi_{\gamma_k}\big) &= \min_{1 \leqslant j \leqslant L}\left[Q_j^T \pi_{\gamma_k}\right] \\ \bar{\mathcal{A}}_k(\pi_{\gamma_k}) &= \min_{F_{k+1} \in Z_k}\left[e(F_{k+1}) + \sum_{F_{k+1}} \Delta^T(F_{k+1} | F_{\gamma_1}, \dots, F_{\gamma_k}, \mathcal{C})\pi_{\gamma_k} \bar{J}_{k+1}(\pi_{\gamma_{k+1}})\right] \end{aligned}$$

- Optimum classification strategy

$$D^*_{\sigma(R)} = \arg\min_{1 \leqslant j \leqslant L}\left[Q_j^T \pi_{\sigma(R)}\right]$$

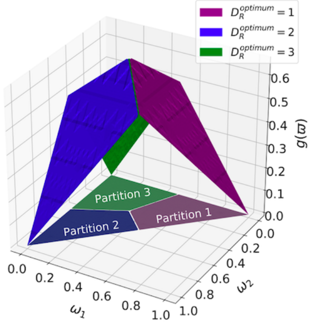
# Theoretical Results

- Function $g(\varpi)$ is continuous, concave, and piecewise linear, and represented by set $\{Q_j^T\}_{j=1}^L$ of $L$ vectors
- Functions $\bar{A}_k(\pi_{\gamma_k}), k = 0, \dots, K-1$, are continuous, concave, and piecewise linear
- At every stage $k \in \{0, \dots, K\}$, there exists a finite set $\{\alpha_k^i\}$ of vectors such that

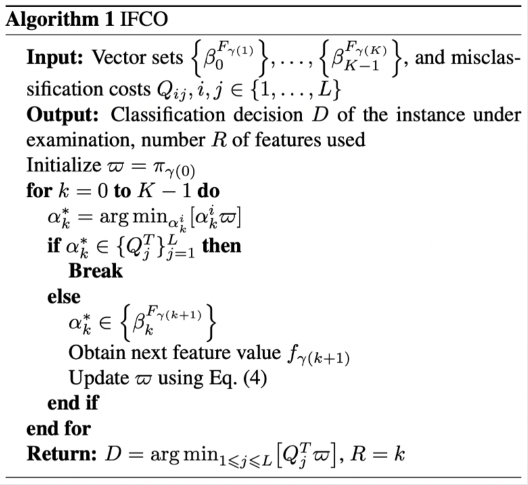
$$\bar{J}_k(\varpi) = \min_i [\alpha_k^i \varpi]$$

$$\{\alpha_k^i\} = \left\{\left\{\beta_k^{F_{\gamma_{k+1}}}\right\} \cup \{Q_j^T\}_{j=1}^L\right\}, k \in \{0, \dots, K-1\}$$

$$\{\alpha_K^i\} = \{Q_j^T\}_{j=1}^L$$



# IFCO Algorithm

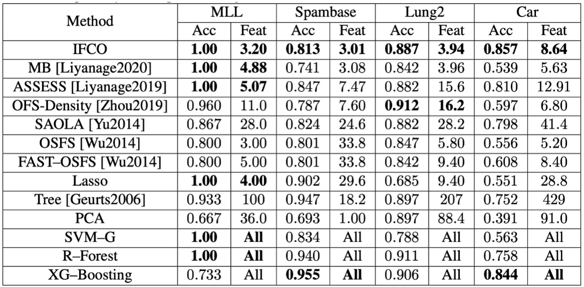


```
Algorithm 1 IFCO
  Input: Vector sets $\left\{\beta_0^{F_{\gamma(1)}}\right\}, \dots, \left\{\beta_{K-1}^{F_{\gamma(K)}}\right\}$, and misclassification costs $Q_{ij}, i, j \in \{1, \dots, L\}$
  Output: Classification decision $D$ of the instance under examination, number $R$ of features used
  Initialize $\varpi = \pi_{\gamma(0)}$
  for $k = 0$ to $K - 1$ do
    $\alpha_k^* = \arg\min_{\alpha_k^i} [\alpha_k^i \varpi]$
    if $\alpha_k^* \in \{Q_j^T\}_{j=1}^L$ then
      Break
    else
      $\alpha_k^* \in \left\{\beta_k^{F_{\gamma(k+1)}}\right\}$
      Obtain next feature value $f_{\gamma(k+1)}$
      Update $\varpi$ using Eq. (4)
    end if
  end for
  Return: $D = \arg\min_{1 \leqslant j \leqslant L} [Q_j^T \varpi]$, $R = k$
```

- The input vector sets $\{\beta_k^{F_{\gamma(k)}}\}$ can be computed using a standard point–based value iteration algorithm [KLC98]

# Some Results

- Three DNA microarray datasets: MLL ($5,848$ features), Lung2 ($3,312$ features), Car ($9,182$ features)
- One email dataset: Spambase ($57$ features)

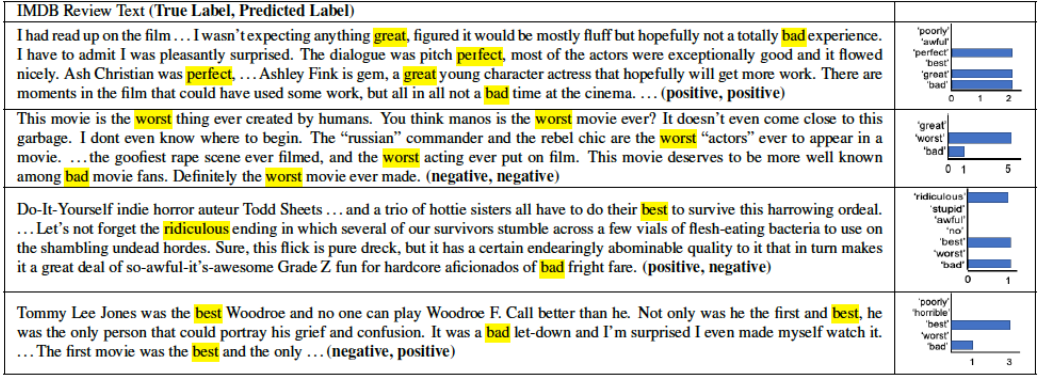


| Method | MLL | | Spambase | | Lung2 | | Car | |
|---|---|---|---|---|---|---|---|---|
| | Acc | Feat | Acc | Feat | Acc | Feat | Acc | Feat |
| IFCO | **1.00** | **3.20** | **0.813** | **3.01** | **0.887** | **3.94** | **0.857** | **8.64** |
| MB [Liyanage2020] | **1.00** | **4.88** | 0.741 | 3.08 | 0.842 | 3.96 | 0.539 | 5.63 |
| ASSESS [Liyanage2019] | **1.00** | **5.07** | 0.847 | 7.47 | 0.882 | 15.6 | 0.810 | 12.91 |
| OFS-Density [Zhou2019] | 0.960 | 11.0 | 0.787 | 7.60 | **0.912** | **16.2** | 0.597 | 6.80 |
| SAOLA [Yu2014] | 0.867 | 28.0 | 0.824 | 24.6 | 0.882 | 28.2 | 0.798 | 41.4 |
| OSFS [Wu2014] | 0.800 | 3.00 | 0.801 | 33.8 | 0.847 | 5.80 | 0.556 | 5.20 |
| FAST–OSFS [Wu2014] | 0.800 | 5.00 | 0.801 | 33.8 | 0.842 | 9.40 | 0.608 | 8.40 |
| Lasso | **1.00** | **4.00** | 0.902 | 29.6 | 0.685 | 9.40 | 0.551 | 28.8 |
| Tree [Geurts2006] | 0.933 | 100 | 0.947 | 18.2 | 0.897 | 207 | 0.752 | 429 |
| PCA | 0.667 | 36.0 | 0.693 | 1.00 | 0.897 | 88.4 | 0.391 | 91.0 |
| SVM–G | **1.00** | **All** | 0.834 | All | 0.788 | All | 0.563 | All |
| R–Forest | **1.00** | **All** | 0.940 | All | 0.911 | All | 0.758 | All |
| XG–Boosting | 0.733 | All | **0.955** | **All** | 0.906 | All | **0.844** | **All** |

# Some Results

- IMDB movie reviews dataset ($89,523$ features)

**Table 2.** Words (features) picked by IFCO are highlighted in yellow. The true/predicted label is given at the end of each review. The second column reports features selected in ascending order (Y–axis) versus feature value (X–axis).



| IMDB Review Text (**True Label, Predicted Label**) | |
|---|---|
| I had read up on the film . . . I wasn't expecting anything great, figured it would be mostly fluff but hopefully not a totally bad experience. I have to admit I was pleasantly surprised. The dialogue was pitch perfect, most of the actors were exceptionally good and it flowed nicely. Ash Christian was perfect, . . . Ashley Fink is gem, a great young character actress that hopefully will get more work. There are moments in the film that could have used some work, but all in all not a bad time at the cinema. . . . (**positive, positive**) | 'poorly'<br>'awful'<br>'perfect'<br>'best'<br>'great'<br>'bad'<br>0 1 2 |
| This movie is the worst thing ever created by humans. You think manos is the worst movie ever? It doesn't even come close to this garbage. I dont even know where to begin. The "russian" commander and the rebel chic are the worst "actors" ever to appear in a movie. . . . the goofiest rape scene ever filmed, and the worst acting ever put on film. This movie deserves to be more well known among bad movie fans. Definitely the worst movie ever made. (**negative, negative**) | 'great'<br>'worst'<br>'bad'<br>0 1 5 |
| Do-It-Yourself indie horror auteur Todd Sheets . . . and a trio of hottie sisters all have to do their best to survive this harrowing ordeal. . . . Let's not forget the ridiculous ending in which several of our survivors stumble across a few vials of flesh-eating bacteria to use on the shambling undead hordes. Sure, this flick is pure dreck, but it has a certain endearingly abominable quality to it that in turn makes it a great deal of so-awful-it's-awesome Grade Z fun for hardcore aficionados of bad fright fare. (**positive, negative**) | 'ridiculous'<br>'stupid'<br>'awful'<br>'no'<br>'best'<br>'worst'<br>'bad'<br>0 1 |
| Tommy Lee Jones was the best Woodroe and no one can play Woodroe F. Call better than he. Not only was he the first and best, he was the only person that could portray his grief and confusion. It was a bad let-down and I'm surprised I even made myself watch it. . . . The first movie was the best and the only . . . (**negative, positive**) | 'poorly'<br>'horrible'<br>'best'<br>'worst'<br>'bad'<br>1 3 |