Unsupervised Learning Scenario: Clustering

Insurance claim -> Event Processor

- Contextual event handling
- Gathering experience

Event Processor -> Claims
- Prediction
- Training

Claims -> Clusters

Alert if anomalous
Supervised Learning Scenario: Classification

Event Processor

Patient checks in

Patient checks out

Features → Department

Classifier

Route to proper department

contextual event handling

gathering experience

prediction

training
Supervised Learning Scenario: Collaborative Filtering

Supervised Learning Scenario: Collaborative Filtering

You might like:
META: Middleware for Events, Transactions, and Analytics [IBMRD’16]
Model Drift

This paper:
• *When to retrain?*
• *On what training data?*
Retraining Workflow (Attempt 1)

Step 1: cache example $\Delta$

Step 3: should retrain? (next decision point)

- **no**
  - Step 4: select retraining data
  - Step 5: retrain model

- **yes**
**Fix**ed-Size Retraining Strategy (Quality-Oblivious)

![Graph showing quality over time with retraining points marked as too early and too late.](image)

- **Too early**: wasted effort
- **Too late**: poor quality
Retraining Workflow (Attempt 2)

Step 1: cache example $\Delta$

Step 2: inc-eval model

Step 3: should retrain?  
(no)

Step 4: select retraining data

Step 5: retrain model

Training data $D$
Test data $T$
Evaluate error (inverse of quality):

$$\text{evaluate}(\text{Model}(D), T) = \sqrt{\frac{\sum_{t \in T} \text{error}(t)^2}{|T|}}$$

New test data $\Delta$
Incrementally update $\text{sum}$ and $\text{count}$

$$\text{evaluate}(\text{Model}(D), T \pm \Delta) = \sqrt{\text{sum}/\text{count}}$$
Retrain When Quality falls Below Threshold

![Graph showing quality over time with retraining points.]

- Quality
- Threshold
- Start (initial model)
- Time
Training Data Selection for Quality-Directed Strategies

- **All**: Data since beginning of time
- **Gen**: Data since last retraining
- **Sw**: Sliding window of width $w$
Retraining Strategies

Step 1: cache example $\Delta$

Step 2: inc-eval model

Step 3: should retrain?
  - yes: Step 4: select retraining data
  - no: Step 5: retrain model

Step 4: select retraining data

Step 5: retrain model

<table>
<thead>
<tr>
<th>Quality-oblivious strategies</th>
<th>Quality-directed strategies</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Never</strong>: Never retrain model</td>
<td><strong>All</strong>: Data since beginning of time</td>
</tr>
<tr>
<td>✷ <strong>Fix</strong>: Fixed-size retrain interval</td>
<td>△ <strong>Gen</strong>: Data since last retraining</td>
</tr>
<tr>
<td></td>
<td>✷ <strong>Sw</strong>: Sliding window of width $w$</td>
</tr>
</tbody>
</table>
AQuA Store Interface

Instance of AquaAlgo subclass

AQuA Store Interface

Examples cache

Meta-data
  • Generations
  • Statistics

Mahout (or other analytics framework)

HDFS (or other store layer)

add  flush  selectTrainingData  retrain
Gradual vs. Abrupt Model Drift

Quality

Time

Start
(initial model)

Retrain

Retrain

Retrain

Model based on
stale or
insufficient data

Abrupt
drift
Retraining Workflow (Attempt 3)

Step A1: cache example ∆

Step A2: is abrupt-flag set?  
- yes: Step A5: incrementally update overall quality and EMA quality  
- no: Step A3: hold enough data?  
  - no: Step A6: abrupt drift?  
    - no: Step A4: retrain on hold-data and reset abrupt-flag  
    - yes: Step A7: set abrupt-flag  
  - yes: Step 3: should retrain?  
    - no: Step 4: select retraining data  
    - yes: Step 5: retrain model
Performance: Collaborative Filtering
(Algorithm: ALSWR, dataset: Netflix)

1/Quality (lower is better)

Cost, measured in examples trained (lower is better)

$S_w$ (sliding window strategies) are on Pareto frontier
Performance: Clustering


1/Quality (lower is better)

$S_w$ (sliding window strategies) are on Pareto frontier

Cost, measured in examples trained (lower is better)
Performance: Classification
(Algorithm: Complementary Naïve Bayes, dataset: Wikipedia)

1/Quality (lower is better)

$S_w$ (sliding window strategies) are on Pareto frontier

Cost, measured in examples trained (lower is better)
Related Work

\[ \text{evaluate}(\text{Model}(D), T) \rightarrow \text{incremental evaluation (cheap!)} \rightarrow \text{evaluate}(\text{Model}(D), T \pm \Delta) \]

\[ \text{Model}(D) \rightarrow \text{incremental training (expensive!)} \rightarrow \text{Model}(D \pm \Delta) \]

\[ D \rightarrow \text{training} \rightarrow \text{update (±Δ)} \rightarrow D \pm \Delta \]
Conclusions

- Incremental evaluation of model quality
- Quality-directed retraining
- Strategies for gradual and abrupt model drift
- Sliding window strategies are on Pareto frontier