Budgeted stream-based active learning via adaptive submodular maximization

Kaito Fujii (UTokyo)
(joint work with Hisashi Kashima (KyotoU))

IBIS Workshop
2017.11.8
Table of Contents

1  Application: Pool-/Stream-based Active Learning

2  Previous Work: Adaptive Submodular Maximization

3  Previous Work: Submodular Secretary Problem

4  Proposed Framework

5  Experiments
Supervised Classification

**Input**
A set of labeled instances \( \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1,...,n} \)

**Output**
A classifier \( \hat{f} : \mathcal{X} \rightarrow \mathcal{Y} \)

\( \mathcal{X} = \mathbb{R}^2, \)
\( \mathcal{Y} = \{\text{red, blue}\} \)
Supervised Classification

**Input**
A set of labeled instances \( \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1,\ldots,n} \)

**Output**
A classifier \( \hat{f} : \mathcal{X} \rightarrow \mathcal{Y} \)

\[
\mathcal{X} = \mathbb{R}^2, \\
\mathcal{Y} = \{ \text{red, blue} \}
\]
Motivation for Active Learning

In some real world scenarios,

- there are a lot of *unlabeled* instances, but
- labeling needs a large *cost* (money or time).

**Active Learning**

The learner selects which instances to label and can reduce the labeling cost.
All unlabeled instances are given in advance

Unlabeled instances

\[ V = \{ x_i \}_{i=1}^{\ldots,n} \subset X \]

Labeling oracle

\[ \phi : V \rightarrow Y \]
Pool-based Active Learning

All unlabeled instances are given in advance

Unlabeled instances

\[ V = \{x_i\}_{i=1,\ldots,n} \subset \mathcal{X} \]

Labeling oracle

\[ \phi : V \rightarrow \mathcal{Y} \]
Pool-based Active Learning

All unlabeled instances are given in advance

Unlabeled instances
\[ V = \{ x_i \}_{i=1,\ldots,n} \subset \mathcal{X} \]

Labeling oracle
\[ \phi : V \rightarrow \mathcal{Y} \]
Pool-based Active Learning

All unlabeled instances are given in advance

Unlabeled instances
\[ V = \{ x_i \}_{i=1,\ldots,n} \subset \mathcal{X} \]

Labeling oracle
\[ \phi : V \rightarrow \mathcal{Y} \]

Unlabeled instances

Labeling oracle
Pool-based Active Learning

All unlabeled instances are given in advance

Unlabeled instances

\[ V = \{ x_i \}_{i=1,...,n} \subset \mathcal{X} \]

Labeling oracle

\[ \phi : V \rightarrow \mathcal{Y} \]
Pool-based Active Learning

All unlabeled instances are given in advance

Unlabeled instances

\[ V = \{ x_i \}_{i=1,\ldots,n} \subset X \]

Labeling oracle

\[ \phi : V \rightarrow Y \]
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$

$(n, k$ known in advance)

Unlabeled instances
$V = \{x_i\}_{i=1,...,n} \subset \mathcal{X}$

Labeling oracle
$\phi: V \rightarrow \mathcal{Y}$
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$ 
($n, k$ known in advance)

Unlabeled instances 
$V = \{x_i\}_{i=1,\ldots,n} \subseteq \mathcal{X}$

Labeling oracle 
$\phi : V \rightarrow \mathcal{Y}$
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$ ($n, k$ known in advance)

Unlabeled instances
$V = \{x_i\}_{i=1,...,n} \subset \mathcal{X}$

Labeling oracle
$\phi: V \rightarrow \mathcal{Y}$
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$
($n, k$ known in advance)

Unlabeled instances
$V = \{x_i\}_{i=1, \ldots, n} \subset \mathcal{X}$

Labeling oracle
$\phi: V \rightarrow \mathcal{Y}$
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$
($n, k$ known in advance)

Unlabeled instances
$V = \{x_i\}_{i=1,...,n} \subset \mathcal{X}$

Labeling oracle
$\phi : V \rightarrow \mathcal{Y}$
Unlabeled instances arrive sequentially
We consider the case of selecting $k$ instances out of $n$
($n, k$ known in advance)

Unlabeled instances
$V = \{x_i\}_{i=1,\ldots,n} \subset \mathcal{X}$

Labeling oracle
$\phi : V \rightarrow \mathcal{Y}$
Stream-based Active Learning

Unlabeled instances arrive sequentially
We consider the case of selecting \( k \) instances out of \( n \)
\( (n, k \) known in advance)
Overview

A new framework for stream-based active learning

**Pool-based active learning**
- All unlabeled instances are given in advance

**Stream-based active learning**
- Unlabeled instances appear one by one

- **Adaptive submodular maximization**
  - [Golovin–Krause’11]

- **Proposed framework**

- **Submodular secretary problem**
  - [Bateni–Hajiaghayi–Zadimoghaddam’13]
Table of Contents

1. Application: Pool-/Stream-based Active Learning
2. Previous Work: Adaptive Submodular Maximization
3. Previous Work: Submodular Secretary Problem
4. Proposed Framework
5. Experiments
**Submodular Maximization**

Selection of a “good” subset of given finite set $V$

Maximize $f(S)$

subject to $|S| \leq k$

$f : 2^V \rightarrow \mathbb{R}$ submodular

**Data Summarization** [Badanidiyuru+’14]

Select a small summary for given large dataset $V$

$$f\left(\begin{array}{c}
\circ \\
\circ \\
\circ \\
\end{array}\right) > f\left(\begin{array}{c}
\circ \\
\circ \\
\circ \\
\end{array}\right)$$
Adaptive Submodular Maximization

[Adaptive Submodularity]

An extension of submodularity to this adaptive setting

The learner can select the next instance to label according to the labels observed so far.
Table of Contents

1. Application: Pool-/Stream-based Active Learning
2. Previous Work: Adaptive Submodular Maximization
3. Previous Work: Submodular Secretary Problem
4. Proposed Framework
5. Experiments
Classical Secretary Problem [folklore’60s]

**Problem**

$n$ candidates arrive **in random order** ($n$ is given), and decide whether to hire at each arrival

**Classical Secretary Algorithm**

pass the first $\lfloor n/e \rfloor$ ones, and after that, if the coming one is the best so far, hire him

→ the best one can be hired with prob. $\geq 1/e$
Classical Secretary Problem [folklore’60s]

Problem

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

Classical Secretary Algorithm

pass the first $\lfloor n/e \rfloor$ ones, and after that, if the coming one is the best so far, hire him.

→ the best one can be hired with prob. $\geq 1/e$

$n = 10$

3
**Classical Secretary Problem** [folklore’60s]

**Problem**

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

**Classical Secretary Algorithm**

pass the first \( \lfloor n/e \rfloor \) ones, and after that, if the coming one is the best so far, hire him.

→ the best one can be hired with prob. \( \geq 1/e \)

\[ n = 10 \]
Classical Secretary Problem [folklore’60s]

**Problem**

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

**Classical Secretary Algorithm**

pass the first $\left\lfloor \frac{n}{e} \right\rfloor$ ones, and after that, if the coming one is the best so far, hire him.

the best one can be hired with prob. $\geq \frac{1}{e}$

$n = 10$

<table>
<thead>
<tr>
<th>3</th>
<th>4</th>
<th>7</th>
</tr>
</thead>
</table>

Classical Secretary Problem [folklore’60s]

**Problem**

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

**Classical Secretary Algorithm**

Pass the first $\lfloor n/e \rfloor$ ones, and after that, if the coming one is the best so far, hire him. The best one can be hired with prob. $\geq 1/e$.

$n = 10$

3 4 7 6
Classical Secretary Problem [folklore’60s]

**Problem**

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

**Classical Secretary Algorithm**

pass the first $\lceil n/e \rceil$ ones, and after that, if the coming one is the best so far, hire him.

→ the best one can be hired with prob. $\geq 1/e$

$n = 10$

3 4 7 6 9
Classical Secretary Problem [folklore'60s]

Problem

$n$ candidates arrive in random order ($n$ is given), and decide whether to hire at each arrival.

Classical Secretary Algorithm

pass the first $\lfloor n/e \rfloor$ ones, and after that, if the coming one is the best so far, hire him. The best one can be hired with prob. $\geq 1/e$

$n = 10$

3 4 7 6 9 4 5 8 0 2
A generalization of the classical secretary problem

1. multiple candidates can be selected
2. the objective function $f : 2^V \rightarrow \mathbb{R}_{\geq 0}$ is submodular

$n = 10, k = 3$
The competitive ratio of an algorithm is $\alpha \in [0, 1]$. For any problem instance, the output $S$ satisfies:

$$\mathbb{E}[f(S)] \geq \alpha \max_{S^* \subseteq V} f(S^*)$$

the optimal achieved by the clairvoyant
Table of Contents

1 Application: Pool-/Stream-based Active Learning
2 Previous Work: Adaptive Submodular Maximization
3 Previous Work: Submodular Secretary Problem
4 Proposed Framework
5 Experiments
Proposed Framework

The proposed framework is a combination of previous frameworks, but it is not straightforward.

- Adaptive Submodular Maximization
- Submodular Secretary Problem

+ New property: Policy-Adaptive Submodularity
Policy-adaptive submodularity is also a natural extension of submodularity to the adaptive setting.
Policy-adaptive submodularity is also a natural extension of submodularity to the adaptive setting.

Submodularity

Diminishing return of each instance \[\rightarrow\] Diminishing return of each set

Adaptive Submodularity
Policy-adaptive submodularity is also a natural extension of submodularity to the adaptive setting.
Adaptive Stream Algorithm

**Stream setting** A limited memory can be used

Partition the whole stream into $k$ segments, and select the “best” instance from each segment

\[
\frac{n}{k} \text{ instances} \quad \frac{n}{k} \text{ instances} \quad \frac{n}{k} \text{ instances} \quad \cdots
\]

Theorem [Fujii–Kashima’16]

The competitive ratio is \((2 - \sqrt{3})(1 - 1/e) \approx 0.16\)
Adaptive Secretary Algorithm

Secretary setting immediate decision at each arrival

Apply the classical secretary algo. to each segment, and select the “best” instance with probability $1/e$

\[
\begin{align*}
\max \Delta (v|\psi_0) & \text{ with prob. } \frac{1}{e} \\
\max \Delta (v|\psi_1) & \text{ with prob. } \frac{1}{e} \\
\max \Delta (v|\psi_2) & \text{ with prob. } \frac{1}{e}
\end{align*}
\]

Theorem [Fujii–Kashima’16]
The competitive ratio is $\frac{1 - 1/e}{2e \sqrt{1 + 2/e}} \approx 0.08$
Table of Contents

1 Application: Pool-/Stream-based Active Learning
2 Previous Work: Adaptive Submodular Maximization
3 Previous Work: Submodular Secretary Problem
4 Proposed Framework
5 Experiments
Experimental Settings

Datasets

- WDBC ($n = 596$, 32-d)
- MNIST ($n = 14780$, reduced to 10-d by PCA)

Benchmarks

- Uncertainty sampling
- Random

The proposed method is based on ALuMA [Gonen+13]
Experimental Results

The proposed method outperforms uncertainty sampling in each setting.
Overview

A new framework for stream-based active learning

Pool-based active learning
All unlabeled instances are given in advance

Adaptive submodular maximization [Golovin–Krause’11]

Stream-based active learning
Unlabeled instances appear one by one

Submodular secretary problem [Bateni–Hajiaghayi–Zadimoghaddam’13]

Proposed framework