Efficient Nonmyopic Batch Active Search

Shali Jiang  Gustavo Malkomes  Matthew Abbott  Benjamin Moseley  Roman Garnett

NeurIPS 2018
Many real problems involve searching for valuable items from a large pool of candidates in an iterative fashion

Drug discovery

Materials discovery
What is active search
What is active search
What is active search

inference model (gives probabilities)
Pr(•)
What is active search

inference model
\(Pr(\cdot)\) (gives probabilities)

select a point
What is active search

select a point

Is it positive?

Oracle

inference model
(gives probabilities)
Pr(·)
What is active search

Inference model (gives probabilities) $\text{Pr}(\cdot)$

Select a point

Is it positive?

Oracle

Yes/no $\$
What is active search

select a point

Is it positive?

Oracle

yes/no

inference model
(gives probabilities)

Pr(●)
What is active search

Inference model (gives probabilities) \( Pr(\cdot) \)

Select a point

Is it positive?

Oracle

Yes/no $
What is active search

Inference model (gives probabilities) $Pr(\cdot)$

Select a point

Is it positive?

Yes/no

Oracle
What is active search

Identify as many positives as possible in a given number of queries.

Is it positive?

select a point

Oracle

yes/no $
What is active search

Identify as many positives as possible in a given number of queries.

Oracle

How?

Is it positive?

select a point

inference model (gives probabilities)

Pr( )

yes/no $
What is active search

Identify as many positives as possible in a given number of queries.

Oracle

select a batch of points

Is it positive?

How?

inference model (gives probabilities) $Pr(\cdot)$

yes/no $\$\$
The most straightforward policy one might think of is greedy:

—always choose the points with highest probabilities
The most straightforward policy one might think of is *greedy*:

—always choose the points with highest probabilities

\[ X^* = \arg \max_X \text{“expected \#positives in } X \text{”} \]
The most straightforward policy one might think of is greedy:

—always choose the points with highest probabilities

\[ X^* = \arg \max_X \text{“expected \#positives in } X\text{”} \]

It’s efficient but myopic, ignoring what could happen in future
The most straightforward policy one might think of is greedy:

—always choose the points with highest probabilities

\[ X^* = \arg \max_X \text{“expected \#positives in } X\text{”} \]

It’s efficient but myopic, ignoring what could happen in future

How can we do better?
Nonmyopic: consider not only this batch, but also what could happen afterwards!

\[ X^* = \arg \max_X \left[ \text{expected } \# \text{positives in } X \right] + \]

[expected #positives in future conditioned on X].
Nonmyopic: consider not only this batch, but also what could happen afterwards!

\[ X^* = \arg \max_X \left[ \text{expected } \#\text{positives in } X \right] + \]

[expected \#positives in future conditioned on \( X \)].

Assume conditional independence after \( X \).
Nonmyopic: consider not only this batch, but also what could happen afterwards!

\[ X^* = \arg \max_X \left( \text{expected } \# \text{positives in } X \right) + \]

\[ \text{expected } \# \text{positives in future conditioned on } X \].

Assume conditional independence after \( X \)

Efficient for sequential setting (batch size 1) (Jiang et al. (ICML 2017)).
Nonmyopic: consider not only this batch, but also what could happen afterwards!

\[ X^* = \arg \max_X \left[ \text{expected } \#\text{positives in } X \right] + \]
\[ \text{[expected } \#\text{positives in future conditioned on } X \right] \].

Assume conditional independence after \( X \).

Efficient for sequential setting (batch size 1) (Jiang et al. (ICML 2017)).

Combinatorial search in batch setting → two approaches: greedy maximization and sequential simulation
Sequential simulation

assume the label by a fictional oracle

0 1 0 0

select 1st point with a sequential policy

select 2nd point, conditioned on the assumed label of the 1st
Sequential simulation

Assume the label by a fictional oracle

Select 1st point with a sequential policy

Select 2nd point, conditioned on the assumed label of the 1st

How?
Sequential simulation

- Assume the label by a fictional oracle

- How?
  - Sample a label?
  - Most likely label?
  - Always positive?
  - Always negative?

- Select 1st point with a sequential policy
- Select 2nd point, conditioned on the assumed label of the 1st
Sequential simulation

How?

- sample a label?
- most likely label?
- always positive?
- always negative?

Best performing due to encouraging diversity

select 1st point with a sequential policy
select 2nd point, conditioned on the assumed label of the 1st
assume the label by a fictional oracle
Empirical results

Averaged over 1600 experiments (10 drug discovery datasets, 8 batch sizes, and 20 repetitions each)
Batching is more efficient, but what are we compromising?
Batching is more efficient, but what are we compromising?

$T = 20$
Batching is more efficient, but what are we compromising?

\[ T = 20 \]

\[ (1 \text{ point / iter}) \times (20 \text{ iters}) = b = 1 \]

every point is chosen after observing the outcomes of all previous points!
Batching is more efficient, but what are we compromising?

\[ T = 20 \]

\( (1 \text{ point} / \text{iter}) \times (20 \text{ iters}) = 1 \]

\( (5 \text{ points} / \text{iter}) \times (4 \text{ iters}) = 5 \)

Every point is chosen after observing the outcomes of all previous points!

Points are chosen without observing the outcomes of previously added points in this batch.
Batching is more efficient, but what are we compromising?

\[ T = 20 \]

- (1 point / iter) * (20 iters)
  - \( b = 1 \)
  - Every point is chosen after observing the outcomes of all previous points!

- (5 points / iter) * (4 iters)
  - \( b = 5 \)
  - Points are chosen without observing the outcomes of previously added points in this batch

Less adaptive decisions could lead to worse performance!
Batching is more efficient, but what are we compromising?

\[ T = 20 \]

(1 point / iter) * (20 iters) \[ b = 1 \]

(5 points / iter) * (4 iters) \[ b = 5 \]

Less adaptive decisions could lead to worse performance!

But how much worse?
Adaptivity gap

We prove that the performance ratio between optimal sequential and batch policies is at least linear in the batch size!

\[
\frac{\text{OPT}_1}{\text{OPT}_b} = \Omega \left( \frac{b}{\log T} \right)
\]
Adaptivity gap

We prove that the performance ratio between optimal sequential and batch policies is at least linear in the batch size!

\[ \frac{\text{OPT}_1}{\text{OPT}_b} = \Omega \left( \frac{b}{\log T} \right) \]

matching empirical results
Adaptivity gap

We prove that the performance ratio between optimal sequential and batch policies is at least linear in the batch size!

\[ \frac{\text{OPT}_1}{\text{OPT}_b} = \Omega \left( \frac{b}{\log T} \right) \]

This insight could help us choose the batch size in cases where we have many options.

matching empirical results
Thanks for your attention!
Poster: #131

Shali Jiang  Gustavo Malkomes  Matthew Abbott  Benjamin Moseley  Roman Garnett

NeurIPS 2018