

Using Program Synthesis for Social Recommendations

Alvin Cheung

Armando Solar-Lezama

Samuel Madden

MIT

Building social networking apps using phone data


- Mobile phones now come with various sensors
 - Accelerometer
 - GPS
 - Proximity
- Data collected from sensors can be used to automatically generate events
 - Enables new social networking apps

LifeJoin: Automatic generation of interesting events

LifeJoin Newsfeed


Learned model


Inferred events

We think that you like events that are about  and 


Event Ratings


event score: -0.6862915

 Pacific Street
 Oct 26, 19:31
Oct 27, 12:02

 Broadway @ Columbia St
 Oct 26, 18:40
Oct 27, 10:31

event score: $-5.0761978e-10$

 Bitter National Magnet Laboratory
 Oct 27, 12:02
Oct 27, 12:05

 Red Line
 Oct 27, 11:24
Oct 27, 11:25

Like Dislike

Like Dislike

Learning example

- Given labeled data:

User	Location	Time	Interested?
Joe	Office	10am	N
Bill	Home	3pm	N
Joe	Office	11pm	Y
Joe	Bar	6am	Y

- Possible classifier:
(User = Joe) and
(location = Office or location = Bar) and
(time < 7 am or time > 10pm)

Applying machine learning to LifeJoin

- Does not create decomposable models

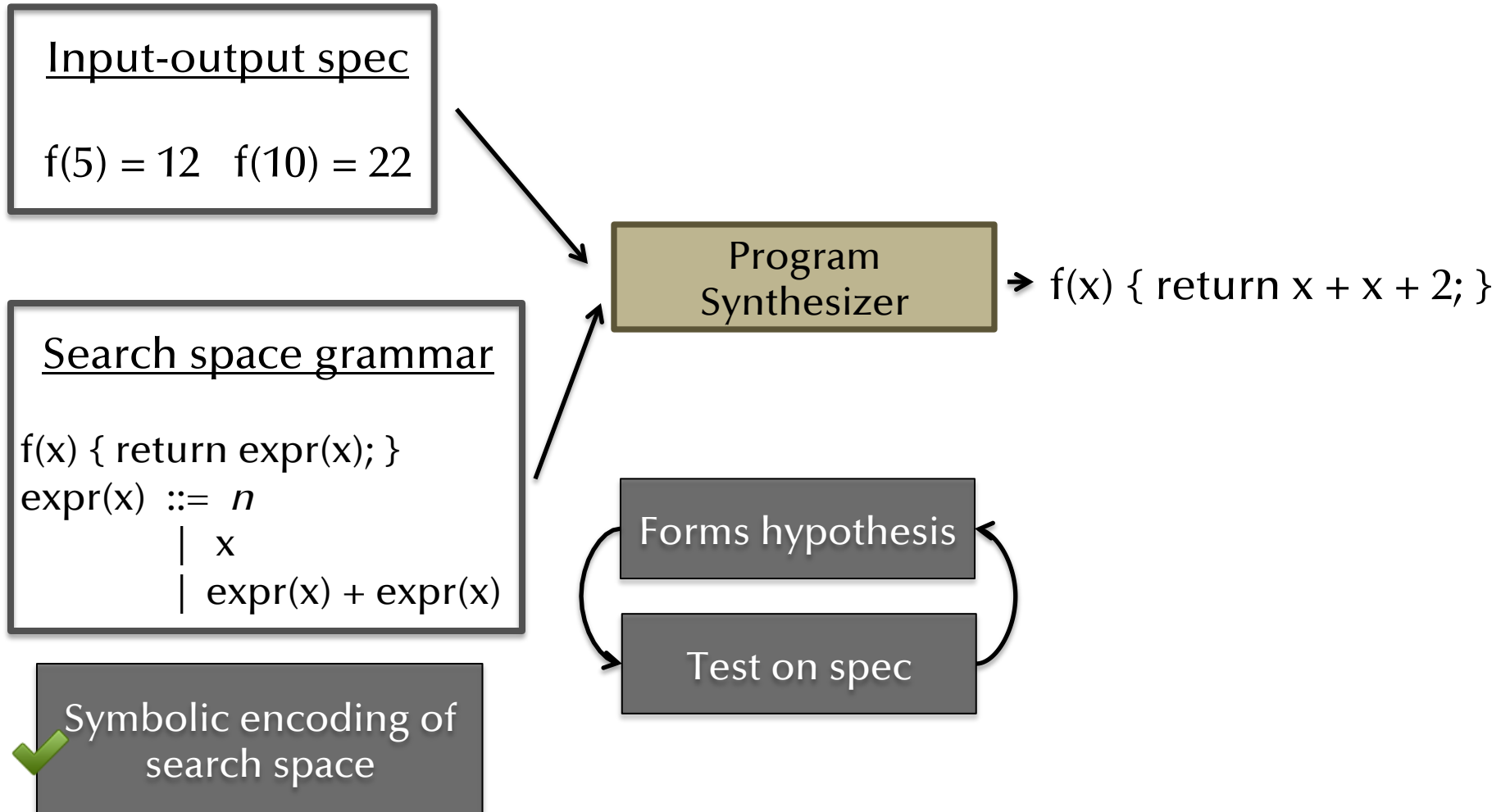
(User = Joe) and
(location = Office or location = Bar) and
(time < 7 am or time > 10pm)

vs.

$$0.25x_1 + 0.65x_2 > 0$$

- Need to encode inputs into feature space representation
 - Create substantial time and space overhead
- Events are highly personalized
 - Hard to leverage labeled data from others

Intro to program synthesis



Applying synthesis to LifeJoin

Input-output spec
Interest([Peter, jog, Charles, 5PM]) = Y
Interest([Mary, office, 9AM]) = N
...

Program
Synthesizer

Search space grammar
interest(e) { act(e) | loc(e) |
 act(e) & loc(e) | ... }

act(e) ::= e.user = *u*
 | e.activity = *a*
 | e.time = *t*
 | act(e) & act(e)
 ...

interest(e)
{ e.user = Peter and
 e.activity = jog
}

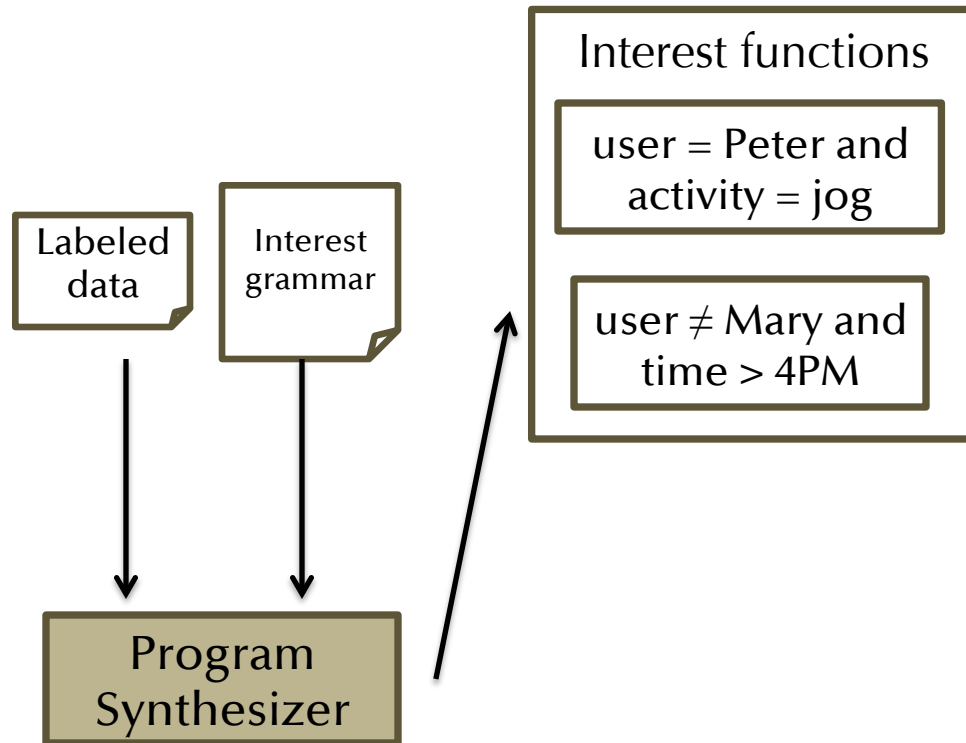
interest(e)
{ e.user ≠ Mary and
 e.time > 4PM
}

Shortcomings:

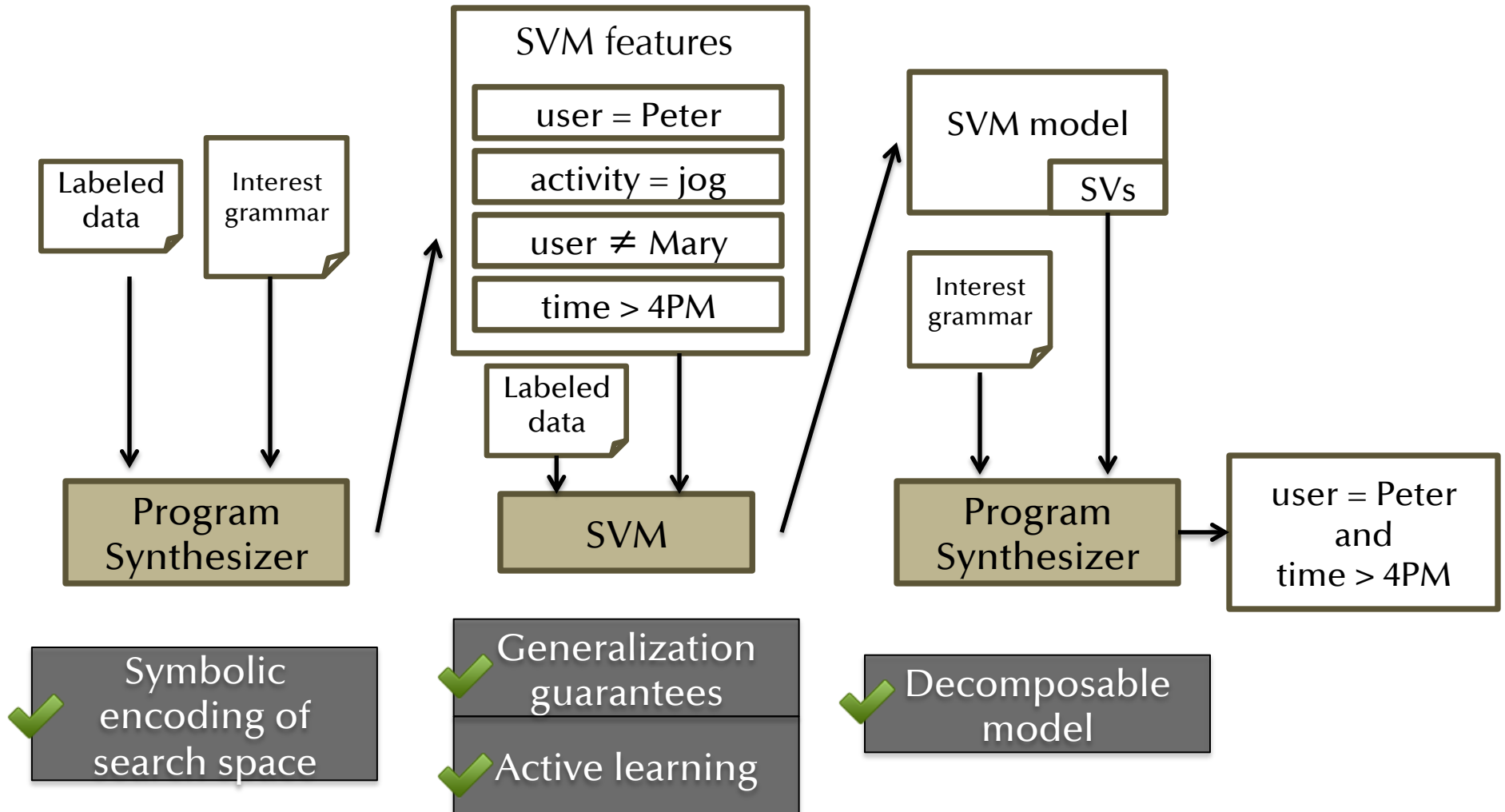
✗ Generalization
guarantees

✗ Active learning

Our new hybrid approach



Our new hybrid approach

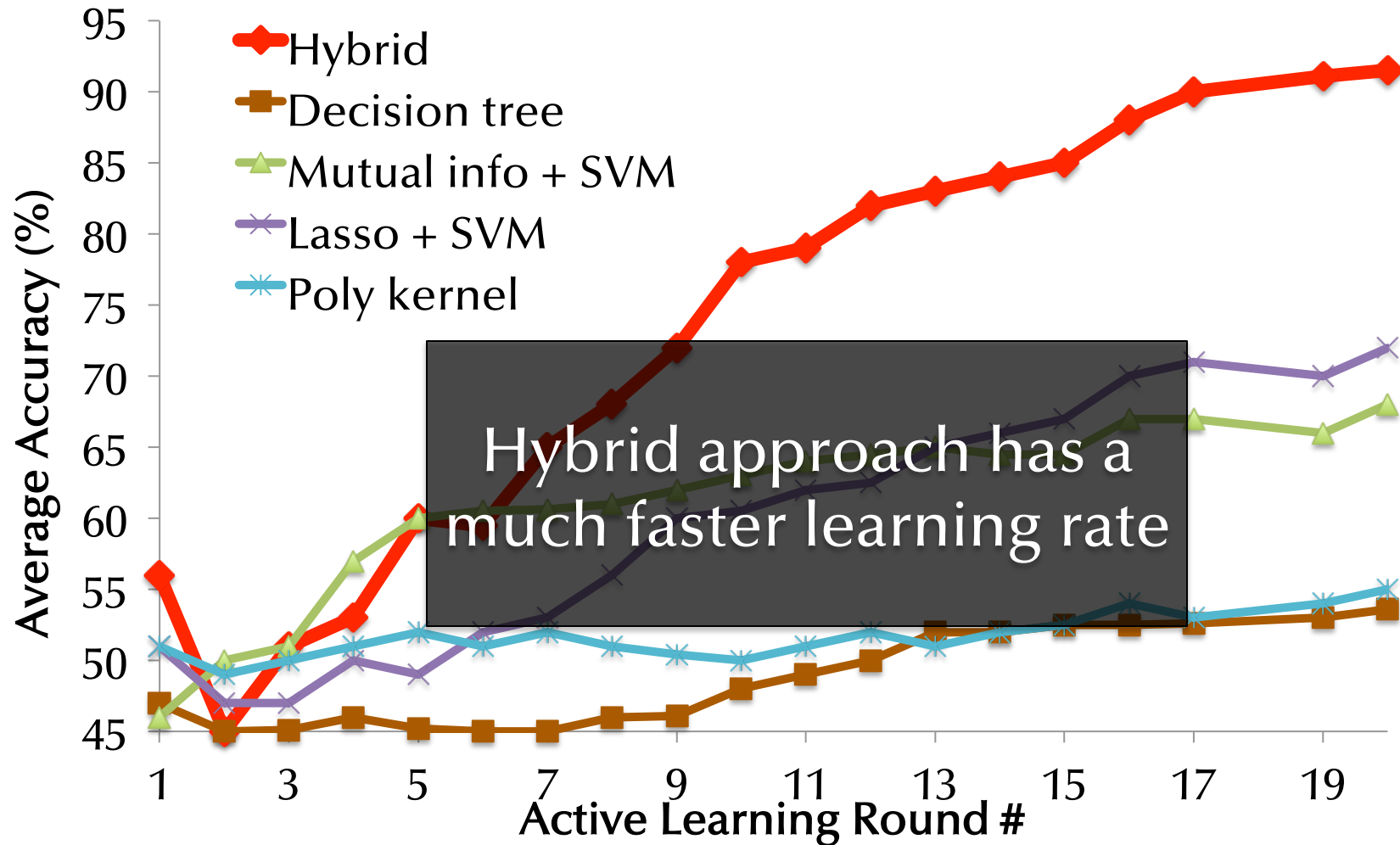


Experimental results

- Implemented different feature selection algorithms and classifiers

Name	Feature selection	Classification
Linear	None	Linear SVM
Poly	Unary features only	Poly. kernel SVM
L1	LASSO	Linear SVM
MI	Mutual information	Linear SVM
Tree	C4.5 on unary features	Linear SVM
Hybrid	Features extracted from 10 synthesized functions	Linear SVM

Experimental results: active learning



Learning user interests in LifeJoin

- Learning user interests from phone data poses new challenges
- Combine machine learning algorithms and programming synthesis techniques to solve the learning problem

<http://people.csail.mit.edu/akcheung>