Precise Piecewise Affine Models from Input-Output Data

Rajeev Alur
Nimit Singhania
University of Pennsylvania

EMSOFT’14, New Delhi
Real World Systems | Models
---|---
Complex | Easy to Analyze

Motivation
Motivation

Real World Systems

Data Driven

Input-Output Data

Models
Motivation

Real World Systems

Input-Output Data

Data Driven

Piecewise Affine Models
1D Piecewise Affine Model

\[
\begin{align*}
\text{if } (x < 2.5) & \quad 5 - x \\
\text{else if } (x < 7.5) & \quad x \\
\text{else} & \quad 15 - x
\end{align*}
\]
2D Piecewise Affine Model

\[ l_1: \ 0 \]

\[ l_2: \ y - x \]

\[ l_3: \ (y - x - 16)/5 \]

Linear Functions
2D Piecewise Affine Model

- \( l_1: 0 \)
- \( l_2: y - x \)
- \( l_3: (y - x - 16)/5 \)

Guard Predicates

- \((x < -1 \text{ and } y < -1) \text{ or } (x > 1 \text{ and } y > 1)\)
- \(y < x - 4\)

Linear Functions
2D Piecewise Affine Model

if ((x < -1 and y < -1) or (x > -1 and y > -1))
  0
else if (y < x - 4)
  (y - x + 16)/5
else
  y-x
Problem

Input-Output Data $\mathbf{D}$

Piecewise Affine Model $f$

$f$ closely approximates $\mathbf{D}$

$f$ is simple
Problem
Problem

• Given input-output data $\mathbf{D}: \mathbb{R}^d \times \mathbb{R}$ and error bound $\delta$, learn a piecewise affine model $f: \mathbb{R}^d \rightarrow \mathbb{R}$

• $|f(a) - b| \leq \delta$ for all points $(a, b) \in \mathbf{D}$

• Minimize size of model
  • Minimize number of linear functions and size of predicates
Problem

• Given input-output data $D: \mathbb{R}^d \times \mathbb{R}$ and error bound $\delta$, learn a piecewise affine model $f: \mathbb{R}^d \to \mathbb{R}$

  • $|f(a) - b| \leq \delta$ for all points $(a, b) \in D$

• Minimize size of model

  • Minimize number of linear functions and size of predicates

• Two problems

  • Learn linear functions

  • Learn guard predicates for associated regions
Problem

• Given input-output data $D: R^d \times R$ and error bound $\delta$, learn a piecewise affine model $f: R^d \rightarrow R$
  
  • $|f(a) - b| \leq \delta$ for all points $(a, b) \in D$

• Minimize size of model
  
  • Minimize number of linear functions and size of predicates

• Two problems
  
  • Learn linear functions
  
  • Learn guard predicates for associated regions

• Minimizing the model size - NP-Hard [1, 2]
  
  • Best effort solution
Solution
Solution

• Learn set of linear functions that covers all points in D
  • \((a, b)\) covered by linear function \(g\) if, \(|g(a) - b| \leq \delta\)
  • Greedy heuristic
Solution

- Learn set of linear functions that covers all points in D
  - \((a, b)\) covered by linear function \(g\) if, \(|g(a) - b| \leq \delta\)
  - Greedy heuristic

- Learn precise guard predicates for each linear function \(g\)
  - Identifies region for \(g\)
    - True on points in \(D\) covered by \(g\)
    - False on points in \(D\) not covered by \(g\)
  - Problem of classification
Learning Linear Functions

- Select random point $p$
Learning Linear Functions

- Select random point $p$
- Fit a linear function that covers points in neighborhood of $p$
Learning Linear Functions

- Select random point $\mathbf{p}$
- Fit a linear function that covers points in neighborhood of $\mathbf{p}$
Learning Linear Functions

- Select random point $p$
- Fit a linear function that covers points in neighborhood of $p$
Learning Linear Functions

- Select random point $p$
- Fit a linear function that covers points in neighborhood of $p$
Learning Linear Functions

- Select random point $p$
- Fit a linear function that covers points in neighborhood of $p$
- Refine it to cover as many points as possible
Learning Linear Functions

- Select random point \( p \)
- Fit a linear function that covers points in neighborhood of \( p \)
- Refine it to cover as many points as possible
Learning Linear Functions

• Select random point $p$

• Fit a linear function that covers points in neighborhood of $p$

• Refine it to cover as many points as possible
Learning Linear Functions

\[ l_1: 0 \]
\[ l_2: y - x \]
\[ l_3: \frac{y - x + 16}{5} \]
Learning Linear Functions

l1: 0

l2: y - x

l3: (y - x + 16)/5
Learning Linear Functions

\[ l_1 : 0 \]

\[ l_2 : y - x \]

\[ l_3 : (y - x + 16)/5 \]
Learning Linear Functions

l1: 0
l2: y - x
l3: (y - x + 16)/5
Learning Guard Predicates

\[ l_1: 0 \]
\[ l_2: y - x \]
\[ l_3: \frac{(y - x + 16)}{5} \]
Learning Guard Predicates

l1: 0
l2: y - x
l3: (y - x + 16)/5
Learning Guard Predicates

\[ l_1: 0 \]
\[ l_2: y - x \]
\[ l_3: \frac{y - x + 16}{5} \]
Learning Guard Predicates

• Propose a counterexample guided approach
• Learns **precise** classifier
• Learns **small** predicate
• Inspired by “Beautiful Interpolants” by Alborghouthi et. al. [3]
• Not possible with existing techniques for *non-convex* regions
Learning Guard Predicates
Learning Guard Predicates

\[ c_1 \cdot x + c_2 \cdot y + c_3 > 0 \]

\[ c_1 \cdot -3 + c_2 \cdot -4 + c_3 > 0 \]

\[ c_1 \cdot -3 + c_2 \cdot 3 + c_3 < 0 \]
Learning Guard Predicates

\[ c_1 x + c_2 y + c_3 > 0 \]
\[ c_1 -3 + c_2 -4 + c_3 > 0 \]
\[ c_1 -3 + c_2 3 + c_3 < 0 \]
\[ c_1 = 0, \ c_2 = -2, \ c_3 = -1 \]

\[ 2y + 1 < 0 \]
Learning Guard Predicates

c1.x + c2.y + c3 > 0

c1.-3 + c2.-4 + c3 > 0

c1.-3 + c2.3 + c3 < 0

c1 = 0, c2 = -2, c3 = -1

2y + 1 < 0
Learning Guard Predicates

c1 \cdot x + c2 \cdot y + c3 > 0

c1 \cdot -3 + c2 \cdot -4 + c3 > 0

c1 \cdot -3 + c2 \cdot 3 + c3 < 0

c1 = 0, c2 = -2, c3 = -1

2y + 1 < 0
Learning Guard Predicates

\[ c_1 \cdot x + c_2 \cdot y + c_3 > 0 \]

\[ c_1 \cdot -3 + c_2 \cdot -4 + c_3 > 0 \]

\[ c_1 \cdot -3 + c_2 \cdot 3 + c_3 < 0 \]

\[ c_1 = 0, \ c_2 = -2, \ c_3 = -1 \]

\[ 2y + 1 < 0 \]
Learning Guard Predicates

\[
\begin{align*}
3y - 4x - 10 &< 0 \\
c_1 - 3 + c_2 - 4 + c_3 &< 0 \\
c_1 + c_2 + c_3 &> 0 \\
c_1 + 2 + c_2 + 3 + c_3 &> 0 \\
c_1 - 3 + c_2 - 3 + c_3 &> 0 \\
3y - 4x - 10 &< 0
\end{align*}
\]
Learning Guard Predicates

\[ 3y - 4x - 10 < 0 \]

\[ c_1.x + c_2.y + c_3 > 0 \]

\[ c_1.-3 + c_2.-4 + c_3 > 0 \]

\[ c_1.2 + c_2.3 + c_3 > 0 \]

\[ c_1.-3 + c_2.3 + c_3 < 0 \]

\[ 3y - 4x - 10 < 0 \]
Learning Guard Predicates

\[ 3y - 4x - 10 < 0 \]

\[ c_1 x + c_2 y + c_3 > 0 \]

\[ c_1 -3 + c_2 -4 + c_3 > 0 \]

\[ c_1 2 + c_2 3 + c_3 > 0 \]

\[ c_1 -3 + c_2 3 + c_3 < 0 \]

\[ 3y - 4x - 10 < 0 \]
Learning Guard Predicates

4y - 3x - 16 < 0
Learning Guard Predicates

$4y - 3x - 16 < 0$
Learning Guard Predicates

\[4y - 3x - 16 < 0\]
Learning Guard Predicates

\[ 4y - 3x - 16 < 0 \text{ and } 2y - 3x + 6 > 0 \]
Learning Guard Predicates

\[(x < -1.5 \text{ and } y < -1.5) \text{ or } (x > 1.5 \text{ and } y > 1.5)\]
Learning Guard Predicates

\[(x < -1.5 \text{ and } y < -1.5) \lor (x > 1.5 \text{ and } y > 1.5)\]

\(O(N^3)\) calls to Linear Constraint Solver
Evaluation

• **MOSAIC**: Tool implemented in Matlab

• Synthetic Data
  
  • Quality of learnt models
  
  • Manually constructed piecewise affine functions and non-linear functions
  
  • Sample training data points, learn model and evaluate on test data points
Synthetic Data

Error %: % of test points with error > δ
Synthetic Data

Size of model vs Input Dimensions
Evaluation

- **MOSAIC**: Tool implemented in Matlab

- Data from Pick and Place Machines
  - Input Voltages and Head Position collected at regular intervals of time
  - Learn discrete time update function for Head Position
  - Compare against existing technique
    - **LINEAR**: replace guard learning technique with SVM
Train Error: Mean Error on points used to learn model
Pick and Place Machine Data

Test Error: Mean Error on points *not* used to learn model
Conclusion

- Presented a technique to learn Piecewise Affine Models from Input-Output Data
- Guarantees on bound on error
- Small model size in practice
- Counterexample guided technique for learning guard predicates
References


Questions?