# Combining Symbolic Expressions and Black-box Function Evaluations in Neural Programs

Forough Arabshahi* Sameer Singh* Animashree Anandkumar†

*University of California, Irvine † California Institute of Technology

## Neural Programming

- Learning black-box functions
- Observations:
  - black-box function evaluations (*fEval*)
  - program execution traces (*eTrace*)
- Challenges: Lack of generalization due to:
  - *fEval*: Insufficient structural information
  - *eTrace*: Computational issues affecting the domain coverage
- Solution:
  - Most problems have access to symbolic representations (*sym*)
  - Combine *sym* and *fEval* data:
    - *sym*: preserve problem's structure
    - *fEval*: enable function evaluation

---

- Case study: Modeling mathematical equations
- Summary of contributions:
  - Combine symbolic representation and function evaluation
  - Equation verification and equation completion using TreeLSTMs
  - Balanced dataset generation method

## Mathematical Equation Modeling

- Grammar rules:

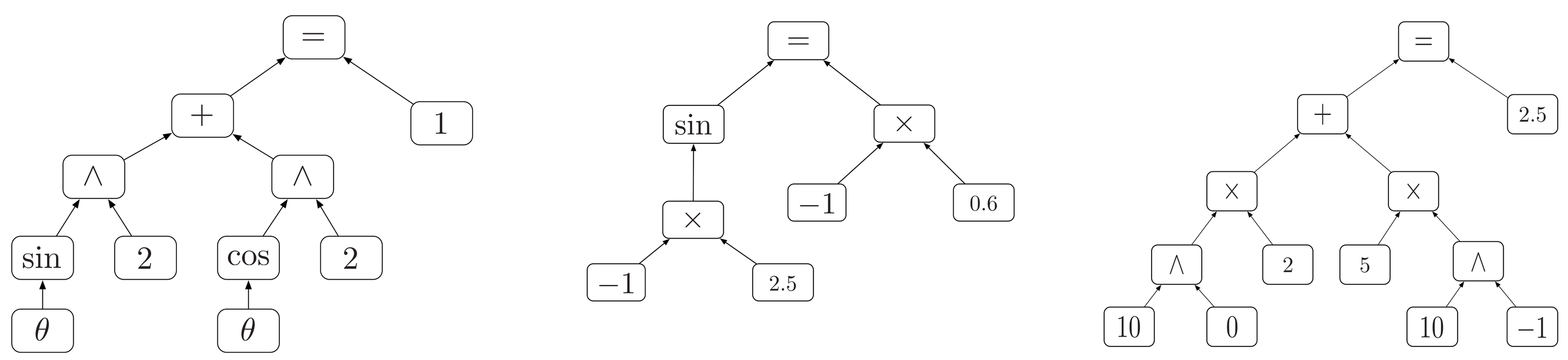
$$
\begin{aligned}
&I \to =(E,E),\ \neq(E,E)\\
&E \to T, F_1(E), F_2(E,E)\\
&F_1 \to \sin, \cos, \tan, \ldots\\
&F_2 \to +, \wedge, \times, \ldots\\
&T \to -1, 0, 1, 2, \pi, x, y, \ldots, \text{any number in [-3.14,+3.14]}
\end{aligned}
$$

- Covered domain:

Table: Symbols in our grammar, i.e. the functions, variables, and constants

| Unary functions, $F_1$ | | | | | Terminal, $T$ | | Binary, $F_2$ |
|---|---|---|---|---|---|---|---|
| sin | cos | csc | sec | tan | 0 | 1 | $+$ |
| cot | arcsin | arccos | arccsc | arcsec | 2 | 3 | $\times$ |
| arctan | arccot | sinh | cosh | csch | 4 | 10 | $\wedge$ |
| sech | tanh | coth | arsinh | arcosh | 0.5 | $-1$ | |
| arcsch | arsech | artanh | arcoth | exp | 0.4 | 0.7 | |
| | | | | | $\pi$ | $x$ | |

- Examples of equation trees:




$\sin^2(\theta)+\cos^2(\theta)=1$ $\sin(-2.5)=-0.6$ decimal tree for 2.5

## Dataset Generation Scheme:

### Generating Symbolic Equations

- Generate possible equations valid in the grammar
  - Start from a small initial set of axioms
  - For each axiom, choose a random tree node
  - Make local random changes to the node:
  - Problem: More incorrect equations than correct
  - Solution: Sub-tree matching
- Generate additional correct equations
  - *mathDictionary*: A dictionary of valid mathematical statements.
    - E.g. $(x+y : y+x)$ forms a key-value pair
  - For each correct equation in the dataset, chose a random tree node
  - Find a dictionary key whose pattern matches the chosen sub-tree
  - Replace the sub-tree with the value's pattern, e.g:
    - Equation: $\sin^2\theta+\cos^2\theta=1$
    - Chosen node: $+$
    - Key-value pair: $(x+y : y+x)$
    - output: $\cos^2\theta+\sin^2\theta=1$

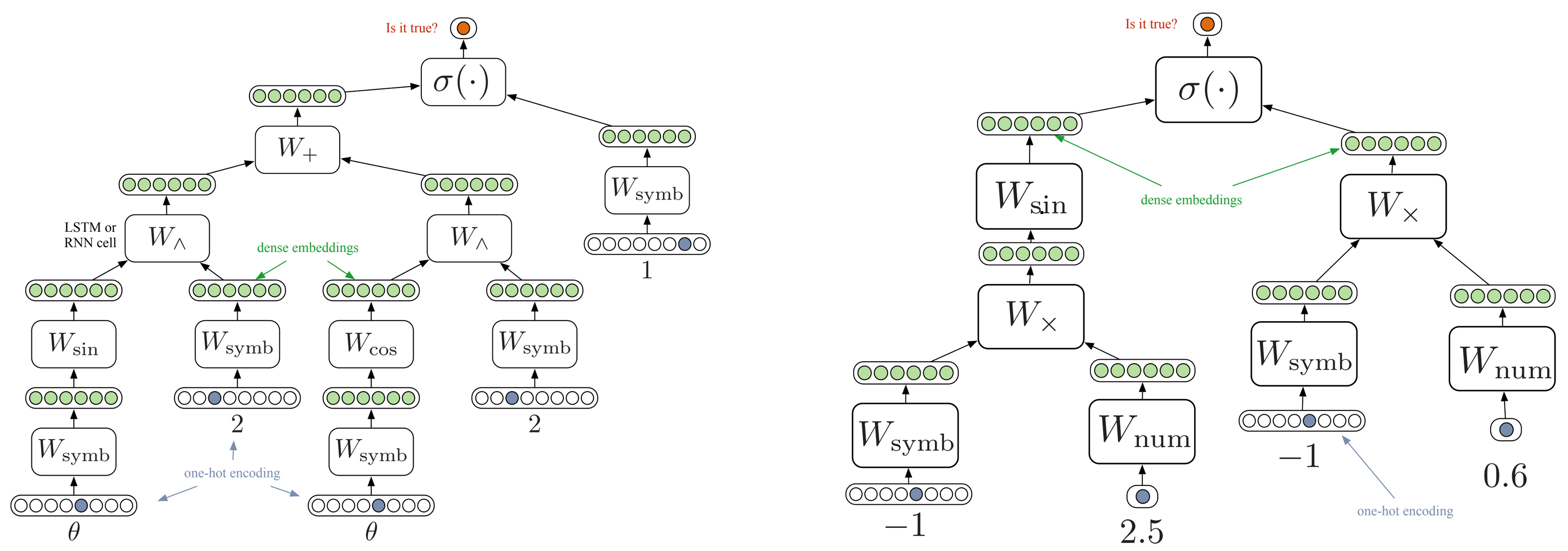
---

### Generating function evaluation equations

- Function Evaluation
  - Range of floating point numbers of precision 2: [-3.14, 3.14]
  - For each unary function: draw a random number and evaluate
  - For each binary function: draw two random numbers and evaluate
- Representation of numbers
  - For all numbers in the dataset, form the decimal tree expansion
  - E.g. $2.5 = 2\times 10^0 + 5\times 10^{-1}$

## Tree LSTMs for Modeling Equations

- Tree LSTM whose structure mirrors the input equation
  - Function blocks are LSTM cells
    - Weight sharing between occurrences of the same function
  - Symbol block is a 1-layer feed-forward net for embedding terminals
  - Number block is a 2-layer feed-forward net for embedding numbers




---

- Baselines:
  - Sequential Recurrent Neural Networks
  - Sequential LSTMs
  - Tree-structured RNNs without function evaluation data
  - Tree-LSTMs without function evaluation data
  - Tree-structured RNNs with function evaluation data

## Experiments and Results

Complexity of an equation: its expression tree depth

- Equation Verification: Generalization to unseen identities

Table: **Generalization Evaluation:** the train and the test contain equations of the same depth [1,2,3,4]. Results are on unseen equations. *Sym* and *F Eval* refer to accuracy on Symbolic and function evaluation expressions, respectively. Test set sizes shown as the counts in (Sym + F Eval) data.

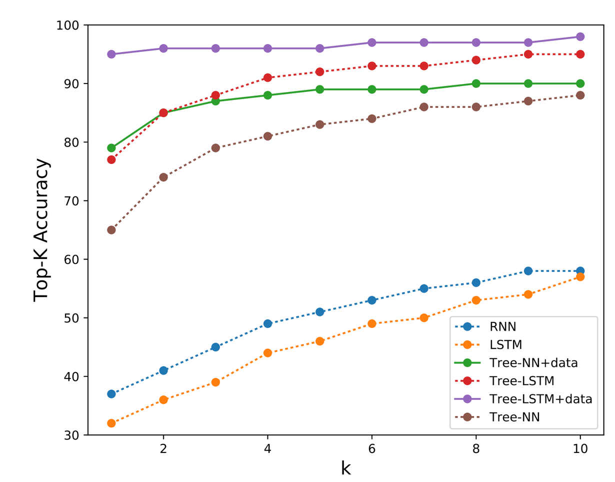
| Approach | Sym | F Eval | depth 1 | depth 2 | depth 3 | depth 4 |
|---|---|---|---|---|---|---|
| Test set size | 3527 | 401 | 5+2 | 542+158 | 2416+228 | 563+13 |
| Majority Class | 50.24 | 50.00 | 20.00 | 45.75 | 52.85 | 43.69 |
| Sympy | 81.74 | - | 80.00 | 89.11 | 82.98 | 69.44 |
| RNN | 66.37 | - | 50 | 62.93 | 65.13 | 72.32 |
| LSTM | 81.71 | - | 80.00 | 79.49 | 80.81 | 83.86 |
| TreeNN | 92.06 | - | **100** | 95.37 | 94.16 | 87.45 |
| TreeLSTM | 95.18 | - | 80.00 | 96.50 | 95.07 | 94.50 |
| TreeNN + data | 93.38 | 92.81 | 87.5 | 94.43 | 92.32 | 93.58 |
| TreeLSTM + data | **97.11** | **97.17** | 75.00 | **98.14** | **97.01** | **97.05** |

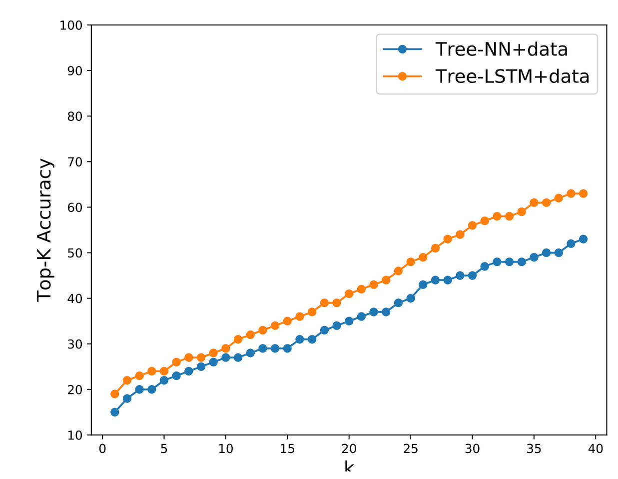
- Equation Verification: Extrapolation to unseen depths

Table: **Extrapolation Evaluation** to measure capability of the model to generalize to unseen depth. Acc: Accuracy, Prec: Precision, Rec: Recall

| Approach | Train:1,2,3; Test on 4 | | | Train:1,3,4; Test on 2 | | |
|---|---|---|---|---|---|---|
| | Acc | Prec | Rec | Acc | Prec | Rec |
| Majority Class | 55.22 | 0 | 0 | 56.21 | 0 | 0 |
| RNN | 65.15 | 68.61 | 75.51 | 71.27 | 82.98 | 43.27 |
| LSTM | 76.40 | 71.62 | 78.35 | 79.31 | 75.27 | 79.31 |
| TreeNN | 88.36 | 87.87 | 85.86 | 92.58 | 89.04 | 94.71 |
| TreeLSTM | 93.27 | 90.20 | 95.33 | 94.78 | 94.15 | 93.90 |
| TreeNN + data | 92.71 | 88.07 | 93.66 | 94.09 | 91.06 | 93.19 |
| TreeLSTM + data | **96.17** | **92.97** | **97.15** | **97.37** | **96.08** | **96.86** |

- Equation Completion




| $4^{\tanh(0)} = \blacksquare^x$ | |
|---|---|
| $-2^0$ | 0.9999 |
| $1^0$ | 0.9999 |
| $7^0$ | 0.9999 |
| $-3^0$ | 0.999 |
| $8^0$ | 0.999 |

| $\tan(\blacksquare) = 0.29$ | |
|---|---|
| 0.28 | 0.9977 |
| 0.27 | 0.9977 |
| 0.29 | 0.9977 |
| 0.26 | 0.9977 |
| 0.25 | 0.9977 |