Learning to Understand Source Code with Machine Learning

Miltos Allamanis

Microsoft Research, Cambridge
Code as...

Data → Software Engineering (SE) Tools

Machine Learning (ML) component → Artificial Intelligence (AI) Tool
Code Understanding

Code Synthesis
Rule-based Expert Systems → Statistical NLP with hand-coded features → Deep Learning
Source Code is Bimodal
Research in ML+Code

- Infer latent intent
- Ambiguous information
- Learned heuristics

ml4code.github.io

A Survey of Machine Learning for Big Code and Naturalness

MILTIADES ALLAMANIS, Microsoft Research
EARL T. BARR, University College London
PREMKUMAR DEVANBU, University of California, Davis
CHARLES SUTTON, University of Edinburgh and The Alan Turing Institute

Research at the intersection of machine learning, programming languages, and software engineering has recently taken important steps in proposing learnable probabilistic models of source code that exploit code's abundance of patterns. In this article, we survey this work. We contrast programming languages against natural languages and discuss how these similarities and differences drive the design of probabilistic models. We present a taxonomy based on the underlying design principles of each model and use it to navigate the literature. Then, we review how researchers have adapted these models to application areas and discuss cross-cutting and application-specific challenges and opportunities.
Vector Space Representations

Local representation

Distributed representation
Detecting Variable Misuses

with Graph Neural Networks

Joint work with
Marc Brockschmidt, Mahmoud Khademi
Target Task

```csharp
var clazz = classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.IsNotNull(clazz);

var first = classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.IsNotNull(first);

Assert.AreEqual("string", first.Properties["Name"].Name);
Assert.IsFalse(clazz.Properties["Name"].IsArray);
```

Possible type-correct options: clazz, first

⚠️ Not easy to catch with static analysis tools.
int SumPositive(int[] arr, int lim) {
    int sum = 0;
    for (int i = 0; i < lim; i++)
        if (arr[i] > 0)
            sum += arr[i];

    return sum;
}
Representing Program Structure as a Graph

```
var clazz = classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first = classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(first);
Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```
(x^1, y^2) = Foo();
while (x^3 > 0) x^4 = x^5 + y^6
Representing Program Structure as a Graph

(x^1, y^2) = Foo();
while (x^3 > 0) x^4 = x^5 + y^6
Representing Program Structure as a Graph

\[(x^1, y^2) = \text{Foo}();\]
\[\text{while } (x^3 > 0) \quad x^4 = x^5 + y^6\]
Representing Program Structure as a Graph

\[(x^1, y^2) = \text{Foo}();\]
\[\text{while } (x^3 > 0) \ x^4 = x^5 + y^6\]
Representing Program Structure as a Graph

Additional Edge Types:
• ReturnsTo

```c
int foo(int sum)
{
    ...
    return x;
}
```
Representing Program Structure as a Graph

Additional Edge Types:
- ReturnsTo
- FormalArgName

```java
define function foo:
  parameter int sum
  return b = foo(result);
```

```java
void foo(int sum) {
  ... 
}
```
Graph Neural Networks

Graph Representation of Problem

Initial Representation of each node

Li et al (2015). Gated Graph Sequence Neural Networks.
Graph Neural Networks: Message Propagation

\[ x = \sum_{n' \in \text{neig}(n)} E_{\tau(n' \rightarrow n)} h_{t-1}^{n'} + b_{\tau(n' \rightarrow n)} \]

\[ h_t^n = \text{GRU}(h_{t-1}^n, x) \]

Graph Neural Networks: Unrolling

Graph Neural Networks: Unrolling

Node Classification: $y_i = \text{MLP}(h_T^n)$

Node Selection: $y = \text{softmax}(f(\{h_T^n\}))$

Graph Classification: $y = g(\{h_T^n\})$

https://github.com/Microsoft/gated-graph-neural-network-samples

Li et al (2015). Gated Graph Sequence Neural Networks.
Backwards Edges

Graph Representation of Problem

Li et al (2015). Gated Graph Sequence Neural Networks.
Representing Variable Type Information

\[ \tau^*(v) = \{ \tau_{\text{List}<\text{string}>}, \tau_{\text{IList}}, \tau_{\text{object}}, \ldots \} \]

\[ r_{\text{List}<\text{string}>} \]

\[ r_{\text{IList}} \]

\[ r_{\text{object}} \]

Elementwise Max

\[ r_{\tau^*(v)} \]
Representing Nodes

classTypes → class, types

Avg

Type Representation $r_{\tau^*(v)}$
Graph Representation for Variable Misuse

```csharp
var clazz = classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.IsNotNull(clazz);

var first = classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
```

Possible type-correct options: `clazz`, `first`
Graph Representation for Variable Misuse

```csharp
var clazz = classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first = classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(first);
Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

Goal: make the representation of SLOT as close as possible to the representation of the correct candidate node
int SumPositive(int[] arr, int limit) {
    int sum = 0;
    for (int i = 0; i < limit; ++i)
        if (arr[i] > 0)
            sum += arr[i];
    return sum;
}
Implementation

- Sparse TensorFlow implementation
- 16 edge types (forward + backward)

Stats

- 65 graphs/s during training
- 272 graphs/s during testing

https://github.com/Microsoft/gated-graph-neural-network-samples
<table>
<thead>
<tr>
<th>Name</th>
<th>Git SHA</th>
<th>kLOCs</th>
<th>Slots</th>
<th>Vars</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akka.NET</td>
<td>719335a1</td>
<td>240</td>
<td>51.3k</td>
<td>51.2k</td>
<td>Actor-based Concurrent &amp; Distributed Framework</td>
</tr>
<tr>
<td>AutoMapper</td>
<td>2ca7c2b5</td>
<td>46</td>
<td>3.7k</td>
<td>10.7k</td>
<td>Object-to-Object Mapping Library</td>
</tr>
<tr>
<td>BenchmarkDotNet</td>
<td>1670ca34</td>
<td>28</td>
<td>5.1k</td>
<td>6.1k</td>
<td>Benchmarking Library</td>
</tr>
<tr>
<td>BotBuilder</td>
<td>190117c3</td>
<td>44</td>
<td>6.4k</td>
<td>8.7k</td>
<td>SDK for Building Bots</td>
</tr>
<tr>
<td>choco</td>
<td>93985688</td>
<td>36</td>
<td>3.8k</td>
<td>5.2k</td>
<td>Windows Package Manager</td>
</tr>
<tr>
<td>commandline†</td>
<td>09677b16</td>
<td>11</td>
<td>1.1k</td>
<td>2.3k</td>
<td>Command Line Parser</td>
</tr>
<tr>
<td>CommonMark.NETDev</td>
<td>f3d54530</td>
<td>14</td>
<td>2.6k</td>
<td>1.4k</td>
<td>Markdown Parser</td>
</tr>
<tr>
<td>Dapper</td>
<td>931c700d</td>
<td>18</td>
<td>3.3k</td>
<td>4.7k</td>
<td>Object Mapper Library</td>
</tr>
<tr>
<td>EntityFramework</td>
<td>fa0b7ec8</td>
<td>263</td>
<td>33.4k</td>
<td>39.3k</td>
<td>Object-Relational Mapper</td>
</tr>
<tr>
<td>Hangfire</td>
<td>ffc4912f</td>
<td>33</td>
<td>3.6k</td>
<td>6.1k</td>
<td>Background Job Processing Library</td>
</tr>
<tr>
<td>Humanizer†</td>
<td>cc11a77e</td>
<td>27</td>
<td>2.4k</td>
<td>4.4k</td>
<td>String Manipulation and Formatting</td>
</tr>
<tr>
<td>Lean†</td>
<td>f574bfd7</td>
<td>190</td>
<td>26.4k</td>
<td>28.3k</td>
<td>Algorithmic Trading Engine</td>
</tr>
<tr>
<td>Nancy</td>
<td>72e1f614</td>
<td>70</td>
<td>7.5k</td>
<td>15.7</td>
<td>HTTP Service Framework</td>
</tr>
<tr>
<td>Newtonsoft.Json</td>
<td>6057d9b8</td>
<td>123</td>
<td>14.9k</td>
<td>16.1k</td>
<td>JSON Library</td>
</tr>
<tr>
<td>Ninject</td>
<td>7006297f</td>
<td>13</td>
<td>0.7k</td>
<td>2.1k</td>
<td>Code Injection Library</td>
</tr>
<tr>
<td>NLog</td>
<td>643e326a</td>
<td>75</td>
<td>8.3k</td>
<td>11.0k</td>
<td>Logging Library</td>
</tr>
<tr>
<td>Opsrver</td>
<td>51b032e7</td>
<td>24</td>
<td>3.7k</td>
<td>4.5k</td>
<td>Monitoring System</td>
</tr>
<tr>
<td>OptiKey</td>
<td>7d35c718</td>
<td>34</td>
<td>6.1k</td>
<td>3.9k</td>
<td>Assistive On-Screen Keyboard</td>
</tr>
<tr>
<td>Orleans</td>
<td>e0d6a150</td>
<td>300</td>
<td>30.7k</td>
<td>35.6k</td>
<td>Distributed Virtual Actor Model</td>
</tr>
<tr>
<td>Polly</td>
<td>0afdb32</td>
<td>32</td>
<td>3.8k</td>
<td>9.1k</td>
<td>Resilience &amp; Transient Fault Handling Library</td>
</tr>
<tr>
<td>quartznet</td>
<td>b33e6f86</td>
<td>49</td>
<td>9.6k</td>
<td>9.8k</td>
<td>Scheduler</td>
</tr>
<tr>
<td>raven.dbDev</td>
<td>55230922</td>
<td>647</td>
<td>78.0k</td>
<td>82.7k</td>
<td>Document Database</td>
</tr>
<tr>
<td>RestSharp</td>
<td>70de357b</td>
<td>20</td>
<td>4.0k</td>
<td>4.5k</td>
<td>REST and HTTP API Client Library</td>
</tr>
<tr>
<td>Rx.NET</td>
<td>2d146fe5</td>
<td>180</td>
<td>14.0k</td>
<td>21.9k</td>
<td>Reactive Language Extensions</td>
</tr>
<tr>
<td>scripts</td>
<td>f3cc8bcb</td>
<td>18</td>
<td>2.7k</td>
<td>4.3k</td>
<td>C# Text Editor</td>
</tr>
<tr>
<td>ServiceStack</td>
<td>6d59da75</td>
<td>231</td>
<td>38.0k</td>
<td>46.2k</td>
<td>Web Framework</td>
</tr>
<tr>
<td>ShareX</td>
<td>718d7111</td>
<td>125</td>
<td>22.3k</td>
<td>18.1k</td>
<td>Sharing Application</td>
</tr>
<tr>
<td>SignalR</td>
<td>fa88089e</td>
<td>53</td>
<td>6.5k</td>
<td>10.5k</td>
<td>Push Notification Framework</td>
</tr>
<tr>
<td>Wox</td>
<td>cda6272</td>
<td>13</td>
<td>2.0k</td>
<td>2.1k</td>
<td>Application Launcher</td>
</tr>
</tbody>
</table>
### Quantitative Results

<table>
<thead>
<tr>
<th>Accuracy (%)</th>
<th>Syntax Only</th>
<th>Avg BiRNN</th>
<th>GGNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seen Projects</td>
<td>49.6</td>
<td>73.5</td>
<td>82.6</td>
</tr>
</tbody>
</table>

3.8 type-correct alternative variables per slot (median 3, $\sigma = 2.6$)
## Quantitative Results

<table>
<thead>
<tr>
<th>Accuracy (%)</th>
<th>Syntax Only</th>
<th>Avg BiRNN</th>
<th>GGNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seen Projects</td>
<td>49.6</td>
<td>73.5</td>
<td>82.6</td>
</tr>
<tr>
<td>Unseen Projects</td>
<td>36.2</td>
<td>59.7</td>
<td>76.0</td>
</tr>
</tbody>
</table>

3.8 type-correct alternative variables per slot (median 3, \(\sigma = 2.6\))
bool TryFindGlobalDirectivesFile(string baseDirectory, string fullPath, out string path) {
    baseDirectory = baseDirectory.TrimEnd(Path.DirectorySeparatorChar);
    var directivesDirectory = Path.GetDirectoryName(baseDirectory)
        .TrimEnd(Path.DirectorySeparatorChar);
    while (directivesDirectory != null && directivesDirectory.Length >= baseDirectory.Length) {
        path = Path.Combine(directivesDirectory, GlobalDirectivesFileFileName);
        if (File.Exists(path)) return true;
        directivesDirectory = Path.GetDirectoryName(directivesDirectory)
            .TrimEnd(Path.DirectorySeparatorChar);
    }
    path = null;
    return false;
}

What the model sees...
bool TryFindGlobalDirectivesFile(string baseDirectory, string fullPath, out string path) {
    baseDirectory = baseDirectory.TrimEnd(Path.DirectorySeparatorChar);
    var directivesDirectory = Path.GetDirectoryName(fullPath)
        .TrimEnd(Path.DirectorySeparatorChar);
    while (directivesDirectory != null && directivesDirectory.Length >= baseDirectory.Length) {
        path = Path.Combine(directivesDirectory, GlobalDirectivesFileName);
        if (File.Exists(path)) return true;

        directivesDirectory = Path.GetDirectoryName(directivesDirectory)
            .TrimEnd(Path.DirectorySeparatorChar);
    }
    path = null;
    return false;
}
Qualitative Results

public void MergeFrom(pb::CodedInputStream input) {
    uint tag;
    while ((tag = input.ReadTag()) != 0) {
        switch (tag) {
            default:
                input.SkipLastField();
                break;
            case 10: {
                .AddEntriesFrom(input, _repeated_payload_codec);
                break;
            }
        }
    }
}

Payload: 66%, payload_: 44%
Qualitative Results

```csharp
public override bool IsDisposed
{
    get
    {
        lock (_gate)
        {
            return _isDisposed;
        }
    }
}
```

<table>
<thead>
<tr>
<th>#1</th>
<th>_gate: 99%, _observers: 1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>#2</td>
<td>_isDisposed: 90%, _isStopped: 8%, HasObservers: 2%</td>
</tr>
</tbody>
</table>
protected void ValidateRestorePreconditions(string backupFilename) {
    if (IsValidBackup(backupFilename) == false) {
        output("Error:" + backupLocation + " doesn’t look like a valid backup");
        output("Error: Restore Canceled");
        throw new InvalidOperationException(
            backupLocation + " doesn’t look like a valid backup");
    }
    ...
}
protected LinePosition GetTrackingLineColumn(Workspace workspace, DocumentId doc)
{
    var currentSnapshot = textBuffer.CurrentSnapshot;

    return GetLinePosition(snapshot, trackingPoint);
}

private LinePosition GetLinePosition(ITextSnapshot snapshot, ITrackingPoint trackingPoint)
```csharp
private string GetTestOutputFilePath(string filePath)
{
    string outputPath = @"Z:\";

    try
    {
        outputPath = Path.GetDirectoryName(filePath);
    }
    catch (ArgumentException)
    {
    }

    if (string.IsNullOrEmpty(outputPath))
    {
        outputPath = @"Z:\";
    }

    return this.CompilationOptions == null ? "" : Path.Combine(outputPath, this.AssemblyId);
}
```
Learning to Name Source Code

We must choose now or the client might cancel the project...

Well, `arrayListCompletedFromFormWithoutDuplicate`?

Too long...
Learning to Name Source Code

A name reflects important aspects of code functionality.

Learning to name source code is a first step in understanding code through machine learning.
Encode Context, Predict Name

Fuse sources of information:

- Tokens around variable uses:
  
  ```
  var ??? = user.GetData();
  ```

- Name of variable type:
  
  ```
  InputStream ??? = FooBar();
  ```

- Formal parameters it is matched to:
  
  ```
  foo(???);
  ...
  void foo(int userNum) { ... }
  ```

⇒ Encode Context, Predict Name
Model: Decoder

- Linear
- Copy Attention
- Attention
- LSTM

Variables:
- $o_1, o_2, ..., o_V$
- $t_c^1, t_a^1, t_1^1, t_f$
- $r_c^1, r_a^1, r_t^1$
- $h_c, h_t, h_a$
- $h_{0}^d$
catch (ErrorResponseException e)
{
    if (e.StatusCode == HttpStatusCode.NotFound)
    {
        var text = e.ResponseString;
        if (text.Contains("maxQueryStringEncoding"))
            throw new ErrorResponseException(e, text);
        throw new ErrorResponseException(e, "There is no index named: " + index);
    }
}

if (HandleException(responseException))
    return null;
if (field == null)
{
    fieldsToFetch = fieldsToFetch.CloneWith(document.GetFields().Select(x => x.Name).ToArray()).
    return base.RetrieveDocument(document, fieldsToFetch, score);
}

var projection = RavenJObject.Parse(field.StringValue);

if (fieldsToFetch.FetchAllStoredFields)
{
    var fields = new HashSet<string>(document.GetFields().Select(x => x.Name));
    fields.Remove(Constants.ReduceKeyFieldName);
    var documentFromFields = new RavenJObject();
    AddFieldsToDocument(document, fields, documentFromFields);

    Model Suggestions
    ------------------
    Suggestions fieldNames (64.59%),

    return new IndexQueryResult
    {
        Projection = projection,
        Score = score.Score,
Practical Considerations

- Fuse all sources of information
- Train-Use & Feedback Loop
- Train/Use Costs
- Metrics
- Low-Resource Environments
- Confidentiality
This is your machine learning system?

Yup! You pour the data into this big pile of linear algebra, then collect the answers on the other side.

What if the answers are wrong?

Just stir the pile until they start looking right.
Practical(?) Tips on Debugging Machine Learning Models

Model Capacity (*what can the model learn?*)
- Overtrain on a small dataset
- Synthetic data

Optimization Issues (*can we make the model learn?*)
- Look at training curves
- Monitor gradient update ratios
- Hand-pick parameters for synthetic data

Other model “bugs” (*is the model doing what I want it to do?*)
- Generate samples from your model (if you can)
- Visualize learned representations (*e.g.* embeddings, nearest neighbors)
- Error analysis (examples where the model is failing, most “confident” errors)
- Simplify the problem/model
- Increase capacity, sweep hyperparameters (*e.g.* increase size of $h$ in LSTM)

https://youtu.be/oMB24_ao05A
Learning Semantic Continuous Representations of Symbolic Expressions

(a-b)*(b+c)+(b-b)

a*b+a*c-b*(b+c)

a*c+b*(a-b-c)

ICML 2017. Joint work with Charles Sutton, Pushmeet Kohli, Pankajan Chanthirasegaran
Semantic Continuous Representations

Syntax-driven but **small** changes in syntax can lead in **large** changes in semantics

SemVecs

$$a \land b$$

$$b \land a$$

$$\neg((\neg a) \lor (\neg b))$$

$$a \implies b$$

$$\neg a \lor b$$

$$((\neg a) \lor b) \land (c \lor (\neg c))$$
Sequence RNNs


"Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets" by Armand Joulin and Tomas Mikolov, 2015
Recursive Neural Networks (TreeNN)

Socher et al. 2011, 2013
TreeNNs

Parent

TreeNN

Child 1

Child 2
Learning to Discover Efficient Math Identities

Zaremba et al. 2014
EqNet Architecture Overview
SemVecs in EqNet

Symbolic Expression Parse Tree

Unit Hypersphere

$\mathbb{R}^D$
**Combining Expressions**

\[ \text{COMBINE} (r_{c_0}, \ldots, r_{c_k}, \tau_n) \]

\[ \bar{l}_0 \leftarrow \begin{bmatrix} r_{c_0}, & \ldots, & r_{c_k} \end{bmatrix} \]

\[ \bar{l}_1 \leftarrow \sigma \left( W_{i,\tau_n} \cdot \bar{l}_0 \right) \]

\[ \bar{l}_{out} \leftarrow W_{o0,\tau_n} \cdot \bar{l}_0 + W_{o1,\tau_n} \cdot \bar{l}_1 \]

**Return** \[ \bar{l}_{out} / \| \bar{l}_{out} \|_2 \]
Subexpression Forcing

Parents are of same Equivalence Class

Same Equivalence Class, Different Representations
Subexpression Forcing

Parents are of same Equivalence Class

Same Equivalence Class, Different Representations
\[ \text{SUBEXPFORCE} \left( r_{c_0}, \ldots, r_{c_k}, r_n, \tau_n \right) \]

\[
x \leftarrow \begin{bmatrix} r_{c_0}, \ldots, r_{c_k} \end{bmatrix}
\]

\[
\tilde{x} \leftarrow \tanh \left( W_d \cdot \tanh \left( W_e, \tau_n \cdot \begin{bmatrix} r_n, x \end{bmatrix} \cdot n \right) \right)
\]

\[
\tilde{x} \leftarrow \tilde{x} \cdot \|x\|_2 / \|\tilde{x}\|_2
\]

\[
\tilde{r}_n \leftarrow \text{COMBINE}(\tilde{x}, \tau_n)
\]

\[
\text{return } - \left( \tilde{x}^T x + \tilde{r}_n^T r_n \right)
\]
Training Objective

\[ P(e_i | T) = \frac{\exp \left( \text{TREENN}(T)^\top q_{e_i} + b_i \right)}{\sum_j \exp \left( \text{TREENN}(T)^\top q_{e_j} + b_j \right)} \]
Training Objective

\[ \mathcal{L}_{\text{ACC}}(T, e_i) = \max \left( 0, \arg \max_{e_j \neq e_i, e_j \in \mathcal{E}} \log P(e_j | T) - \log P(e_i | T) + m \right) \]

\[ \mathcal{L}(T, e_i) = \mathcal{L}_{\text{ACC}}(T, e_i) + \frac{\mu}{|Q|} \sum_{n \in Q} \text{SUBEXPFORCE}(\text{ch}(n), n) \]

scheduled introduction
<table>
<thead>
<tr>
<th>Dataset</th>
<th># Vars</th>
<th># Equiv Classes</th>
<th># Exprs</th>
<th>$H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMPBOOL8</td>
<td>3</td>
<td>120</td>
<td>39,048</td>
<td>5.6</td>
</tr>
<tr>
<td>SIMPBOOL10$^S$</td>
<td>3</td>
<td>191</td>
<td>26,304</td>
<td>7.2</td>
</tr>
<tr>
<td>BOOL5</td>
<td>3</td>
<td>95</td>
<td>1,239</td>
<td>5.6</td>
</tr>
<tr>
<td>BOOL8</td>
<td>3</td>
<td>232</td>
<td>257,784</td>
<td>6.2</td>
</tr>
<tr>
<td>BOOL10$^S$</td>
<td>10</td>
<td>256</td>
<td>51,299</td>
<td>8.0</td>
</tr>
<tr>
<td>SIMPBOOLL5</td>
<td>10</td>
<td>1,342</td>
<td>10,050</td>
<td>9.9</td>
</tr>
<tr>
<td>BOOLL5</td>
<td>10</td>
<td>7,312</td>
<td>36,050</td>
<td>11.8</td>
</tr>
<tr>
<td>SIMPPOLY5</td>
<td>3</td>
<td>47</td>
<td>237</td>
<td>5.0</td>
</tr>
<tr>
<td>SIMPPOLY8</td>
<td>3</td>
<td>104</td>
<td>3,477</td>
<td>5.8</td>
</tr>
<tr>
<td>SIMPPOLY10</td>
<td>3</td>
<td>195</td>
<td>57,909</td>
<td>6.3</td>
</tr>
<tr>
<td>ONEV-POLY10</td>
<td>1</td>
<td>83</td>
<td>1,291</td>
<td>5.4</td>
</tr>
<tr>
<td>ONEV-POLY13</td>
<td>1</td>
<td>677</td>
<td>107,725</td>
<td>7.1</td>
</tr>
<tr>
<td>POLY5</td>
<td>3</td>
<td>150</td>
<td>516</td>
<td>6.7</td>
</tr>
<tr>
<td>POLY8</td>
<td>3</td>
<td>1,102</td>
<td>11,451</td>
<td>9.0</td>
</tr>
</tbody>
</table>
### BOOL8

<table>
<thead>
<tr>
<th>$(\neg a) \land (\neg b)$</th>
<th>$(\neg a \land \neg c) \lor (\neg b \land a \land c) \lor (\neg c \land b)$</th>
<th>$(\neg a) \land b \land c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \neg((\neg a) \Rightarrow ((\neg a) \land b))$</td>
<td>$c \oplus (((\neg a) \Rightarrow a) \Rightarrow b)$</td>
<td>$\neg(\neg(b \lor ((\neg c) \lor a))$</td>
</tr>
<tr>
<td>$\neg((b \lor ((\neg a))) \lor b)$</td>
<td>$\neg((b \lor (b \lor a)) \lor c)$</td>
<td>$((a \lor b) \land c) \land (\neg a)$</td>
</tr>
<tr>
<td>$(\neg a) \oplus ((a \lor b) \lor a)$</td>
<td>$\neg((\neg(b \lor ((\neg a))) \lor c)$</td>
<td>$\neg((\neg(b) \Rightarrow a)) \land c$</td>
</tr>
<tr>
<td>$(b \Rightarrow (b \Rightarrow a)) \land (\neg a)$</td>
<td>$((b \lor a) \lor (\neg b)) \lor c$</td>
<td>$(c \land (c \Rightarrow (\neg a))) \land b$</td>
</tr>
<tr>
<td>$((\neg a) \Rightarrow b) \Rightarrow (a \lor a)$</td>
<td>$\neg(((b \lor a) \land a)) \lor c$</td>
<td>$b \land (\neg(b \land (c \Rightarrow a)))$</td>
</tr>
</tbody>
</table>

### False

<table>
<thead>
<tr>
<th>$(a \lor a) \land (c \Rightarrow c)$</th>
<th>$(a \Rightarrow (\neg c)) \lor (a \lor b)$</th>
<th>$a \Rightarrow ((b \land (\neg c)) \lor b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(b \land (\neg (b \Rightarrow a)))$</td>
<td>$(a \Rightarrow (c \lor b)) \lor b$</td>
<td>$\neg((\neg(b \lor a) \Rightarrow b))$</td>
</tr>
<tr>
<td>$b \land ((a \lor a) \lor a)$</td>
<td>$b \lor (a \Rightarrow (b \lor c))$</td>
<td>$(\neg a) \lor (\neg(b \Rightarrow (\neg a)))$</td>
</tr>
<tr>
<td>$((\neg b) \land b) \land (a \lor a)$</td>
<td>$(b \lor a) \lor (x \Rightarrow (\neg a))$</td>
<td>$b \lor ((\neg(b \land (\neg a)))$</td>
</tr>
<tr>
<td>$c \land (x \Rightarrow (\neg a)) \land c$</td>
<td>$b \lor (((\neg a) \lor (c \lor b))$</td>
<td>$\neg((a \Rightarrow (a \lor b)) \land a)$</td>
</tr>
<tr>
<td>Eq Class 1</td>
<td>Eq Class 2</td>
<td>Eq Class 3</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
</tbody>
</table>

Test Datasets
Test Datasets

UnseenEqClass Testset

Eq Class 1
Eq Class 2
Eq Class 3
Eq Class 4
Eq Class 5
Eq Class 6
Eq Class 7
Eq Class 8

20%
Test Datasets

- **SeenEqClass Testset**
  - Eq Class 1
  - Eq Class 2
  - Eq Class 3
  - Eq Class 4
  - Eq Class 5

- **UnseenEqClass Testset**
  - Eq Class 6
  - Eq Class 7
  - Eq Class 8

20% of each class is used for testing.
$score_k(q) = \frac{|N_k(q) \cap c|}{\min(k, |c|)}$
Evaluation

SeenEqClass Testset

UnseenEqClass Testset

\[ \text{score}_k \]

\[ 10^{-1} \quad 10^{0} \]

\[ k \]

\[ 5 \quad 10 \]

Evaluation

- tf-idf
- GRU
- StackRNN
- TreeNN-1Layer
- TreeNN-2Layer
- EqNet
EqNet Performance vs Dataset Characteristics

Operator Types

Num of Variables

score_k

Simple
All

1 Var
3 Vars
10 Vars
Evaluation of Compositionality

SeenEqClass Testset

UnseenEqClass Testset

Evaluation of Compositionality
Visualizing Polynomials and their Negatives
Visualizing Polynomials and their Negatives
Visualizing Polynomials and their Negatives
Visualizing Polynomials and their Negatives
t-SNE Visualization
SimpPoly8

http://groups.inf.ed.ac.uk/cup/semvec/
t-SNE Visualization
Bool10-UnseenTest
Source Code is Bimodal

Practical Considerations
- Fuse all sources of information
- Metrics
- Train-Use & Feedback Loop
- Low-Resource Environments
- Train/Use Costs
- Confidentiality

From Code to Graphs
```
int SumPositive(int[] arr, int line) {
    int sum = 0;
    for (int i = 0; i < line; ++i) {
        if (arr[i] > 0) {
            return sum;
        }
    }
    return sum;
}
```

~900 nodes/graph ~8k edges/graph

What the model sees...

Closing Thoughts