Using RNN to fix syntax errors in JavaScript files

Jordan Siaw, Justin Lew, and Luiz Peres
SFU School of Computing Science
`['jsiaw', 'jylew', 'lperesde'].map(f => f + '@sfu.ca')`

The Problem

Parsers often fail miserably in helping to find syntax errors. For example:

```
1 if (process.argv.length > 3)  // Missing '{'
2    console.error('Not enough args!');
3    process.exit(1);
4 }
```

Error message from Mozilla SpiderMonkey (2016):

```
wrong.js:4: SyntaxError: syntax error:
wrong.js:4: }
wrong.js:4: ^
```

Error message from Node.js with Google's V8 engine (2016):

```
/path/to/wrong.js:6  // Line 6 doesn't even exist!
});
^

SyntaxError: Unexpected token }
```

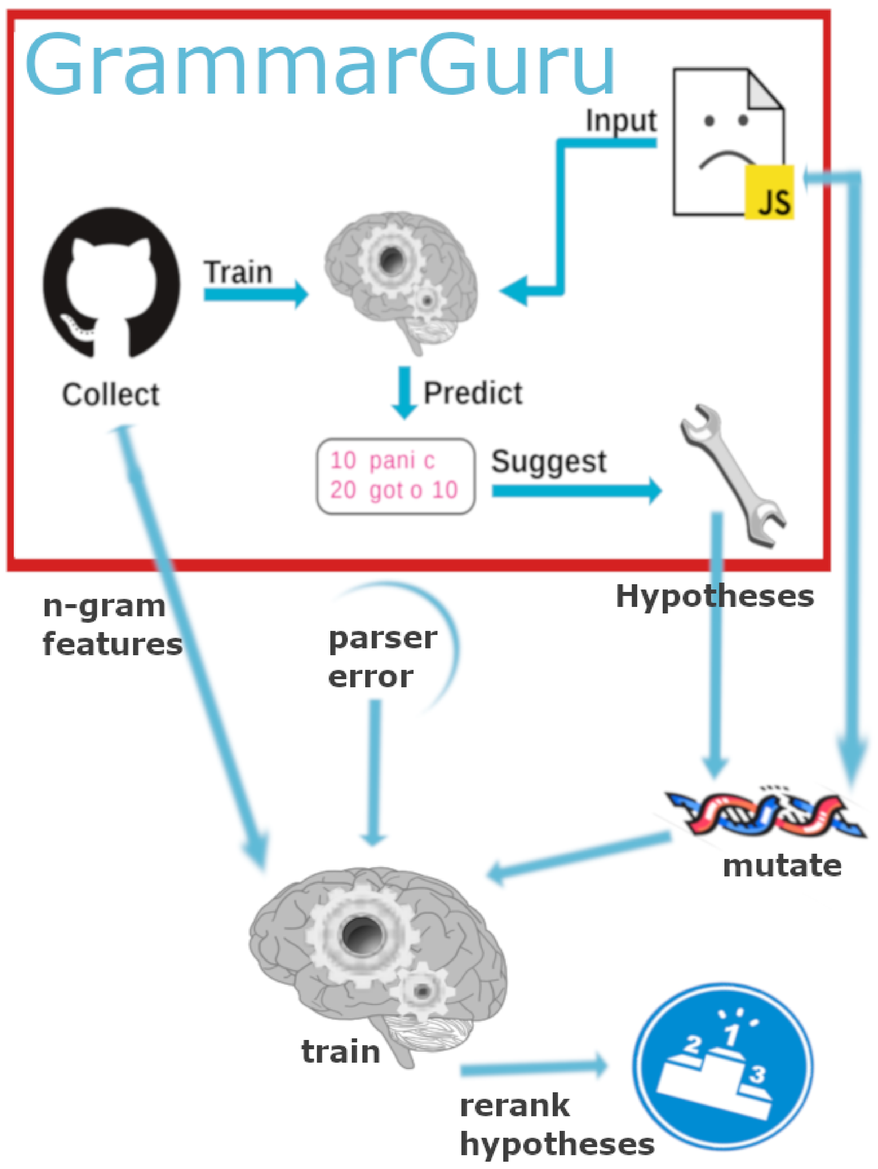
GrammarGuru

GrammarGuru is a tool created by Santos et al. (Santos EA et al., 2017) that attempts to address this problem by finding and fixing single token syntax errors, using LSTM language models on 10,000 GitHub repos. GrammarGuru works great for locating token insertions and deletions. However, it doesn't perform so well when a token is substituted, e.g. when a keyword is mistyped: `functions` instead of `function`.

Improving GrammarGuru suggestions

We improve GrammarGuru by training a neural network that re-evaluates and re-ranks GrammarGuru's scores by using a modification of the features of the ROSE ranking system (Song and Cohn, 2011):

ID	Description
1-4	n-gram precision, n=1...4
5-8	n-gram recall, n=1...4
9-12	n-gram f-measure, n=1...4
13	average n-gram precision per code line
14	n-gram score at the document level
15-18	n-gram precision excluding common tokens, n=1...4
19-22	n-gram recall excluding common tokens, n=1...4
23-26	n-gram f-measure excluding stopwords, n=1...4
27	average n-gram precision excluding stopwords, n=1...4
28	pos distance of hyp_1, hyp_2 against parser error $\{-1, 0, 1\}$
29	which of hyp_1, hyp_2 equals parser error token $\{-1, 0, 1\}$
30	hyp_1 and/or hyp_2 fix the syntax error $\{-1, 0, 1\}$
31	which of hyp_1, hyp_2 has best score $\{-1, 0, 1\}$



Re-ranking Results

15 of the most unnatural tokens in the erroneous JS file are fed to our neural network and re-ranked using the ROSE approach.

```
1 $form.submit(functions(evt) {  // mistyped keyword
2    ...
3 });
```

Error message from GrammarGuru (2017):

```
I don't know how to fix it :C
```

Result after re-ranking:

```
wrong.js:1:14 try substituting with 'function'
$form.submit(functions(evt) {
             ^

          function
```

Mutation	GrammarGuru MRR	Our Solution MRR
Insertion	0.52	
Deletion	0.50	
Substitution	0.30	

References

[1] **Santos EA, Campbell JC, Hindle A, Amaral JN. 2017**. Finding and correcting syntax errors using recurrent neural networks. PeerJ Preprints 5:e3123v1 https://doi.org/10.7287/peerj.preprints.3123v1

[2] **Xingyi Song and Trevor Cohn. 2011**. Regression and ranking based optimisation for sentence level machine translation evaluation. In Proceedings of the Sixth Workshop on Statistical Machine Translation, pages 123–129. Association for Computational Linguistics