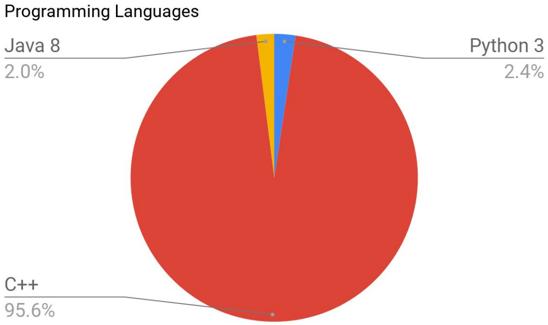
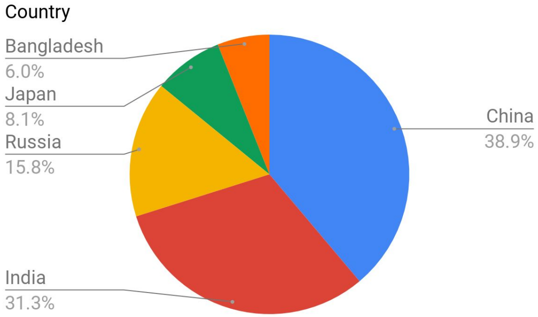
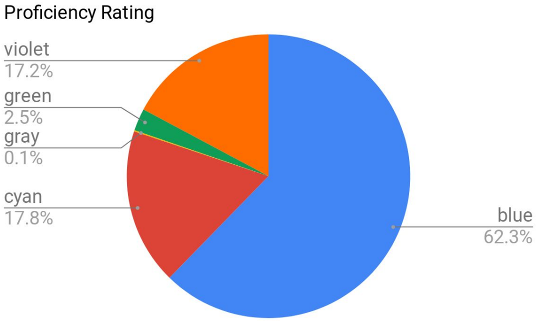
# Analysis of Competitive Codebases

Ashish Kumar Jayant
Animesh Baranawal
Shikhar Bharadwaj

# Data + Tools







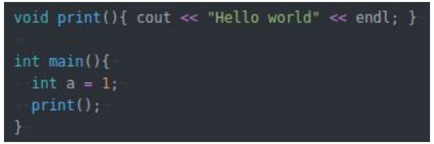
*srcML* : generates XML representation of **abstract syntax tree (AST)** for C, C++, Java

**doc2vec** : vector embedding for documents ( https://github.com/jhlau/doc2vec )

**zss** : edit distance between two tree structures ( https://github.com/timtadh/zhang-shasha )

python NumPy scikit learn Keras

# Starting Simple...



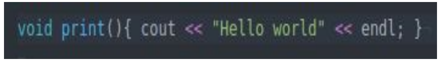
```
void print(){ cout << "Hello world" << endl; }

int main(){
  int a = 1;
  print();
}
```

**Primitive Feature Extraction**

# variable: 1
# function call: 1
# function decl: 2
(other primitive features include
# Array declarations,
# Pointer declarations,
# Reference declarations,
# Typdefs etc.)

**Train Random Forest**

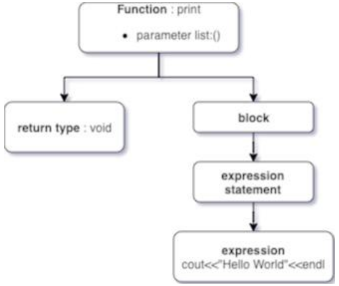
**DOES NOT WORK**



```
void print(){ cout << "Hello world" << endl; }
```

**Generate AST**

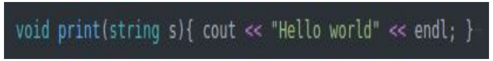
*srcML*



**Calculate Similarity Score**

using ZSS module

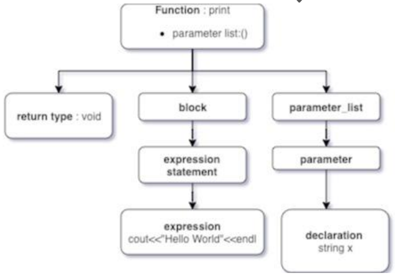
**Use Clustering??**



```
void print(string s){ cout << "Hello world" << endl; }
```

**Generate AST**

*srcML*



# Going complex..

tokenise

```
void, print, (, ),
{, }, cout, <<,
“Hello World”,
endl, ;, int, main,
a, =, 1
```

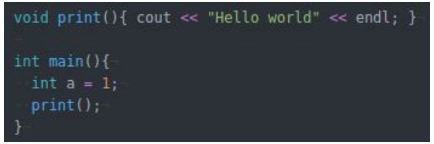
Remove blacklist

tokens

```
void, print,
cout, <<, “Hello
World”, endl,
int, main, a, =,
1
```

count

**Token Feature Vector**



```
void print(){ cout << "Hello world" << endl; }

int main(){
  int a = 1;
  print();
}
```

Generate AST

**AST**

Pre-order

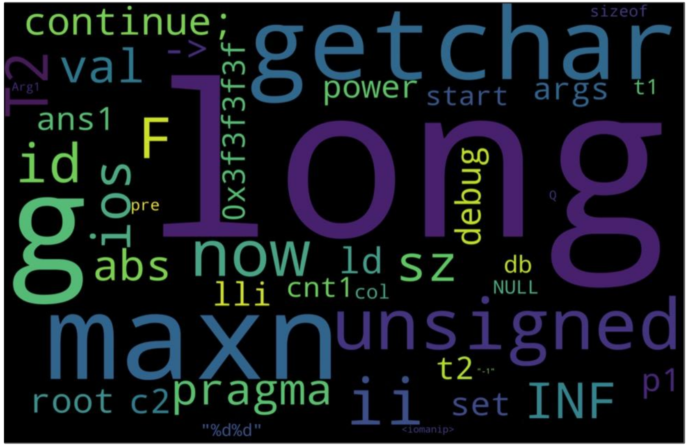
traversal

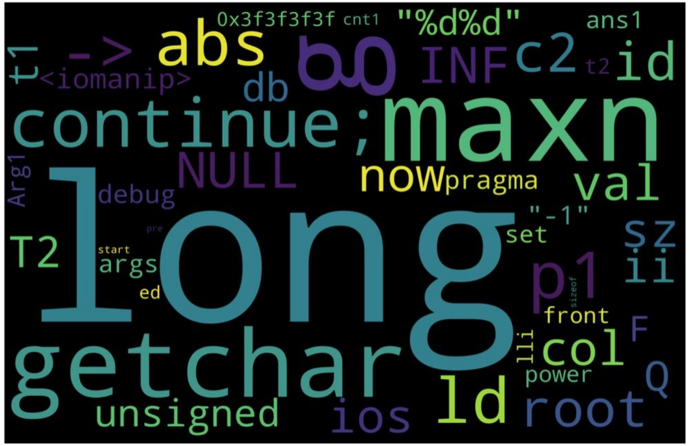
include directive file using namespace name function type name name parameter_list block expr_stmt expr name operator literal type string operator name function type name name parameter_list block decl_stmt decl type name name init expr literal type number expr_stmt expr call name argument_list

doc2vec

**AST Vector Embedding** a.k.a. **Code Embedding**

# What do we get?





**India** vs **China** using **Light GBM**

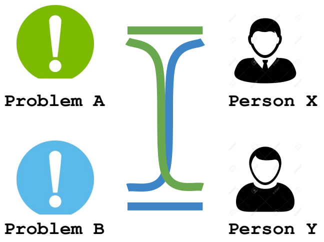
`Accuracy = 93%, F-score = 0.9`

**Coding style ~ Coding Proficiency??**

| Features | Model | Accuracy | Precision | Recall | f_Score |
|---|---|---|---|---|---|
| Doc2Vec of ASTs | Logistic Regression | **0.68** | **0.46** | **0.58** | **0.51** |
| Token vectors | Light GBM | **0.71** | **0.28** | **0.18** | **0.22** |
| Doc2Vec of ASTs + Token vector | Logistic Regression | **0.61** | **0.57** | **0.68** | **0.62** |

`Primitive Neural Network : Accuracy = 67%, F-score = 0.68`

# Diving into Code Embeddings



$A_X$ = **doc2vec**(ProblemA by PersonX)

$B_X$ = **doc2vec**(ProblemB by PersonX)

$A_Y$ = **doc2vec**(ProblemA by PersonY)

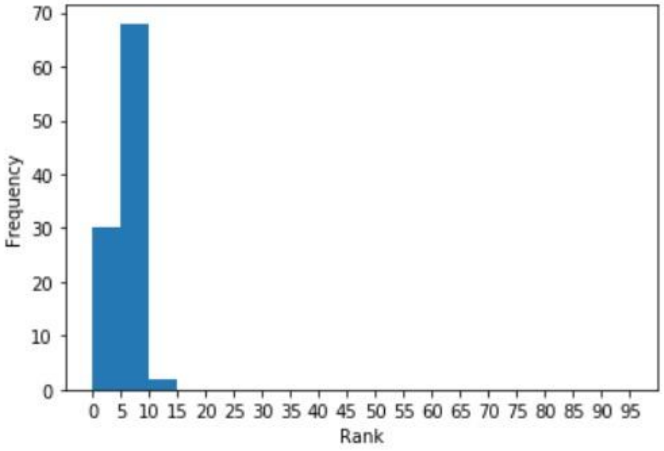
$B_Y$ = **doc2vec**(ProblemB by PersonY)

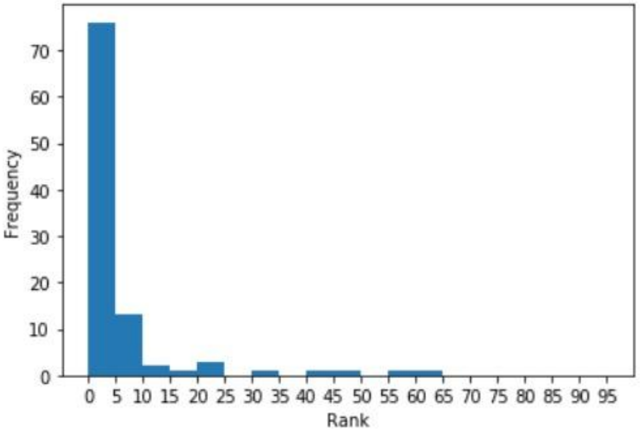
Can we break embedding into **person component** and **problem component**?

$$A_x \sim A + X \; ?$$

If so,

$$A_X - A_Y + B_Y \approx B_X$$


Frequency
Rank


Frequency
Rank