

How Machines Learn to Code

Machine Learning on Source Code

Samuel Hopstock, Thomas Endres

Big Techday 11, 2018-05-18



Who we are



Samuel Hopstock

samuel.hopstock@tngtech.com



Thomas Endres

thomas.endres@tngtech.com

Classical approaches on program evaluation

Static analysis

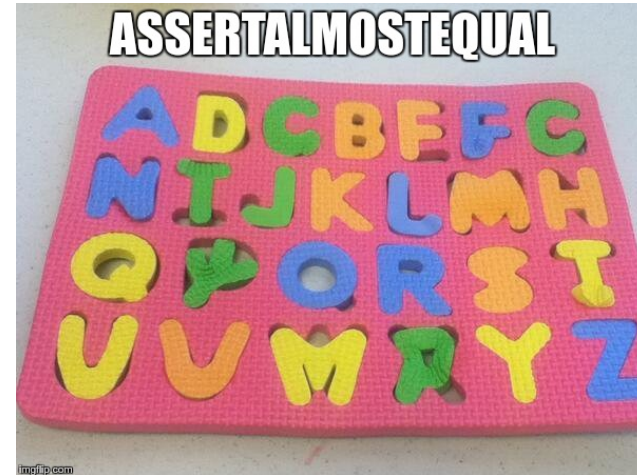


Classical approaches on program analysis

Dynamic analysis

A screenshot of a Java IDE's test results window. The title bar says 'All in java-monkey-runner_test'. The window shows a tree view of test results for the 'default package'. Each test is marked with a green checkmark and has a duration listed in milliseconds. The tests are grouped under several classes: JavaMonkeyRunnerApplicationTest, JavaMonkeyRunnerIntegrationTest, MonkeyRunnerTest, FileUtilsTest, and ProjectCopierTest.

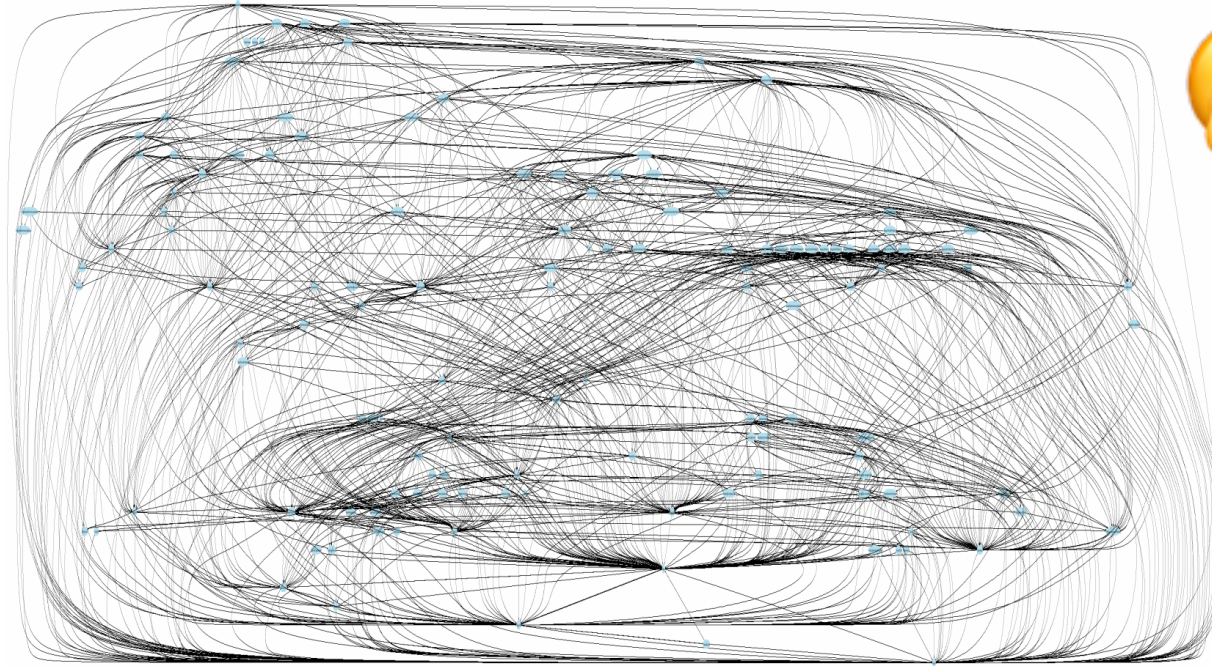
Test Class	Test Method	Duration (ms)	
JavaMonkeyRunnerApplicationTest	whenNotProvidingASourcePathAnErrorShouldBeThrown	536	
	whenNotProvidingATargetPathAnErrorShouldBeThrown	40	
	testRunApplication	144	
	testSimpleProjectModification	969	
JavaMonkeyRunnerIntegrationTest	testSimpleProjectModification	969	
	testSimpleProjectModification	969	
MonkeyRunnerTest	testSimpleProjectModification	47	
FileUtilsTest	testWrite	272	
	whenDirectoryDoesNotExistGetAllFilesWithExtensionShouldTI	116	
	whenDirectoryDoesNotExistCopyRecursivelyShouldThrowAnE	27	
	testCopyRecursively	20	
	testIsDirectory	46	
	whenDirectoryDoesNotExistDeleteRecursivelyShouldThrowAnE	3	
	testExists	9	
	testDeleteRecursively	2	
	whenDirectoryDoesNotExistWriteShouldThrowAnError	14	
	testGetAllFilesWithExtension	6	
	ProjectCopierTest	whenSourceDirectoryDoesNotExistAnErrorShouldBeThrown	29
	ProjectCopierTest	whenSourceDirectoryDoesNotExistAnErrorShouldBeThrown	46
	ProjectCopierTest	whenSourceDirectoryDoesNotExistAnErrorShouldBeThrown	31



<https://devrant.com/rants/64755/unit-test-is-close-enough>

Limitations

Complex
Dependencies

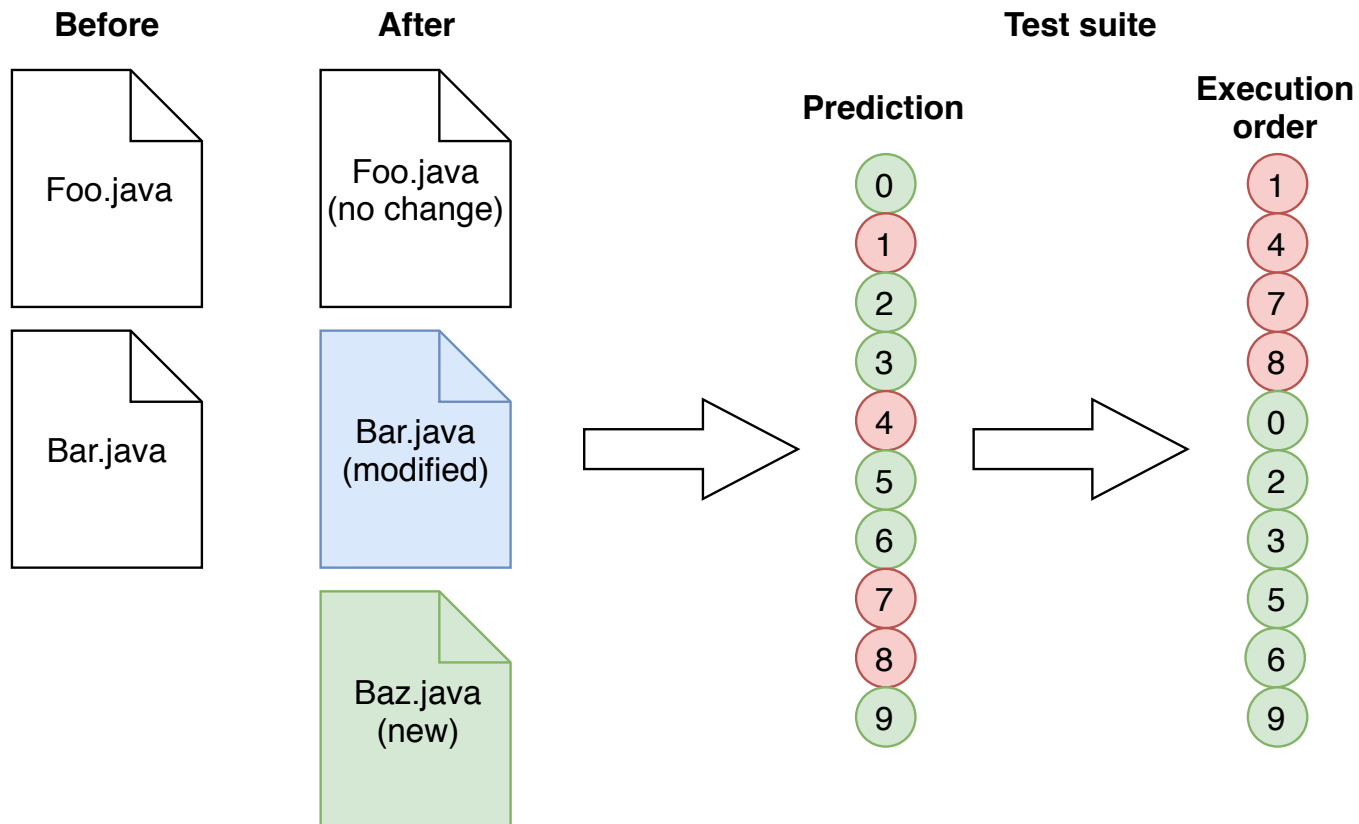


<https://thedailywtf.com/articles/The-Enterprise-Dependency>

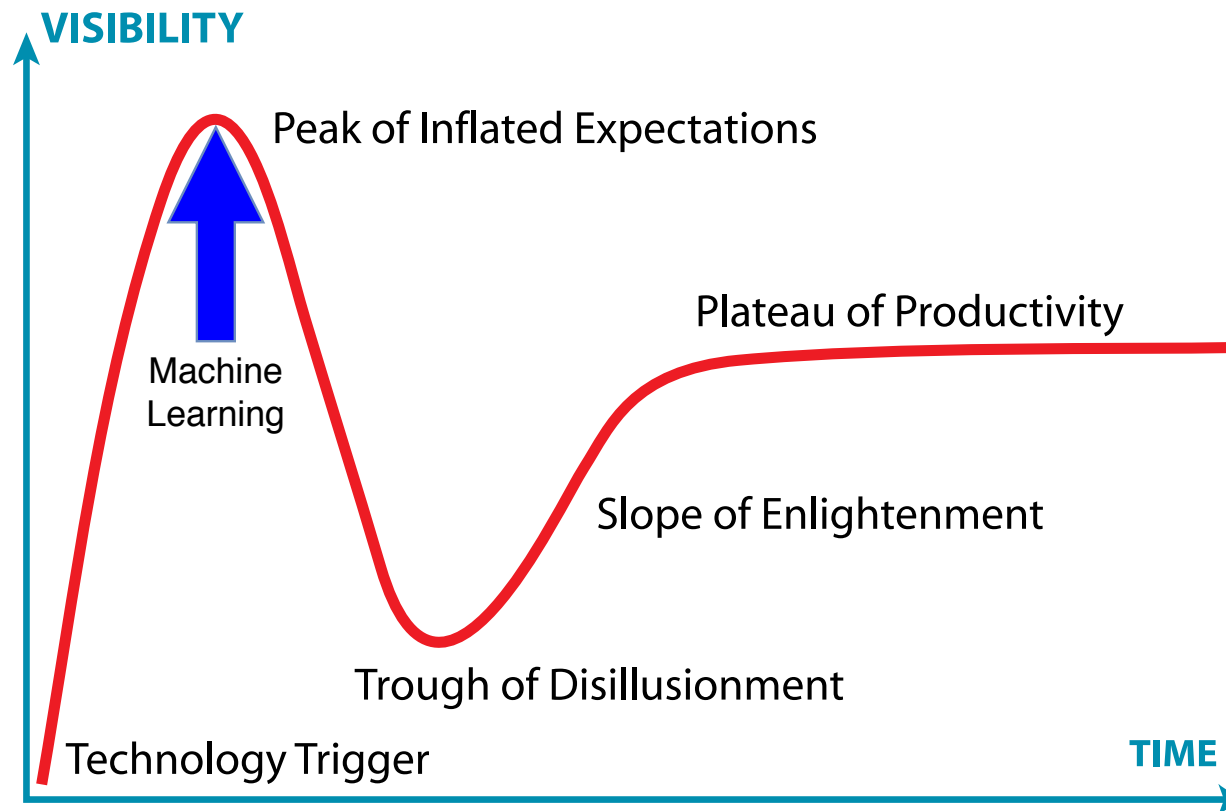
Big Test Suite



Our idea



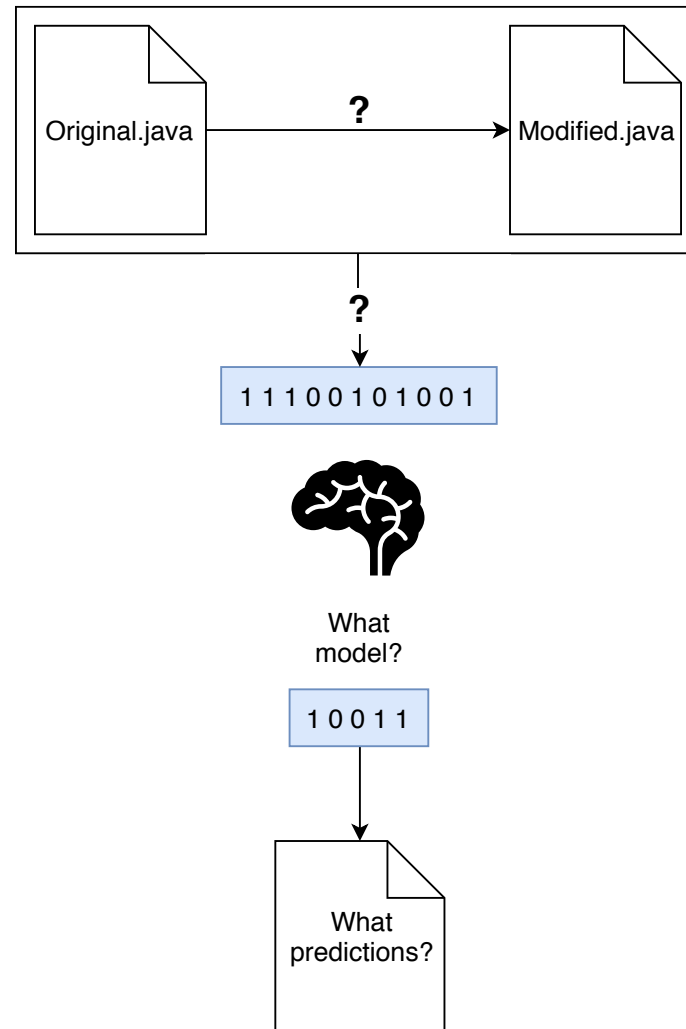
Why machine learning?



Jeremykemp, <https://commons.wikimedia.org/w/index.php?curid=10547051>

It seems good for finding complex relations in data

Overview: challenges to solve



Overview

Data

Vectors for machine learning

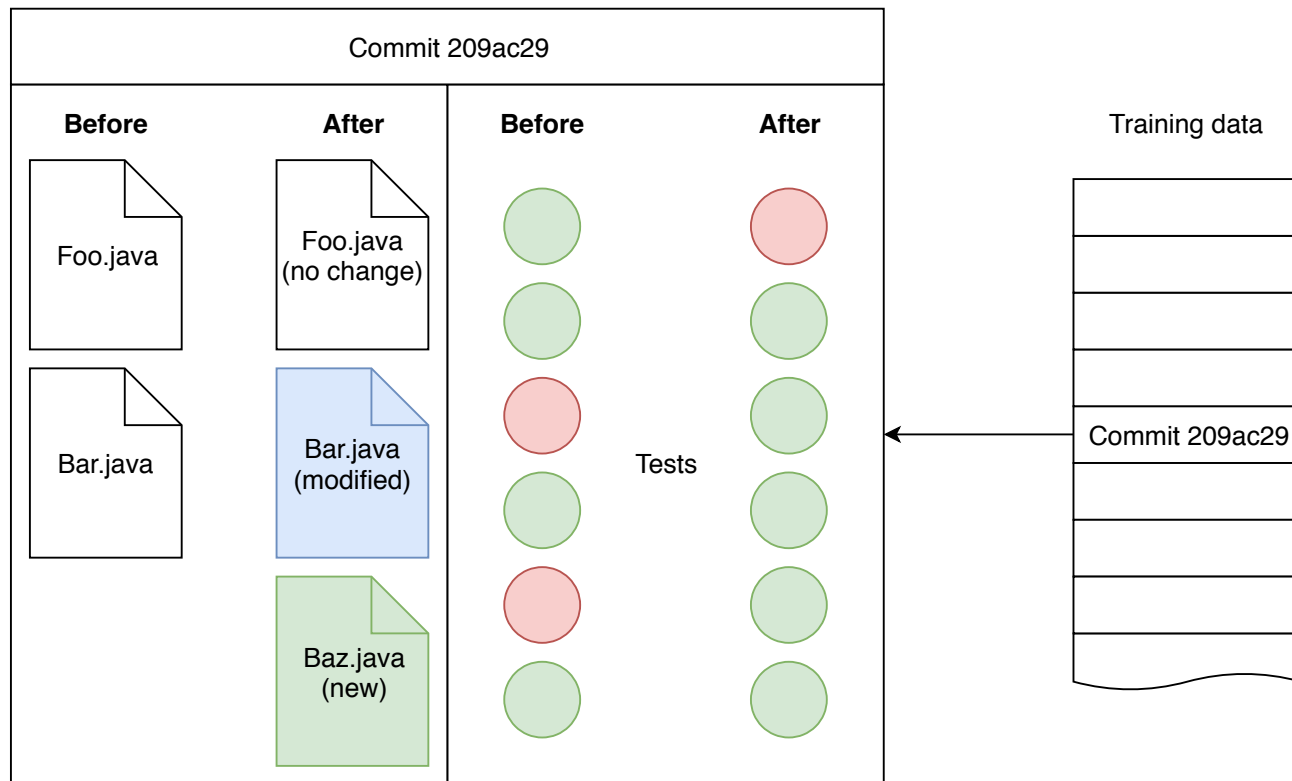
Machine learning techniques

Technical aspects

Conclusion

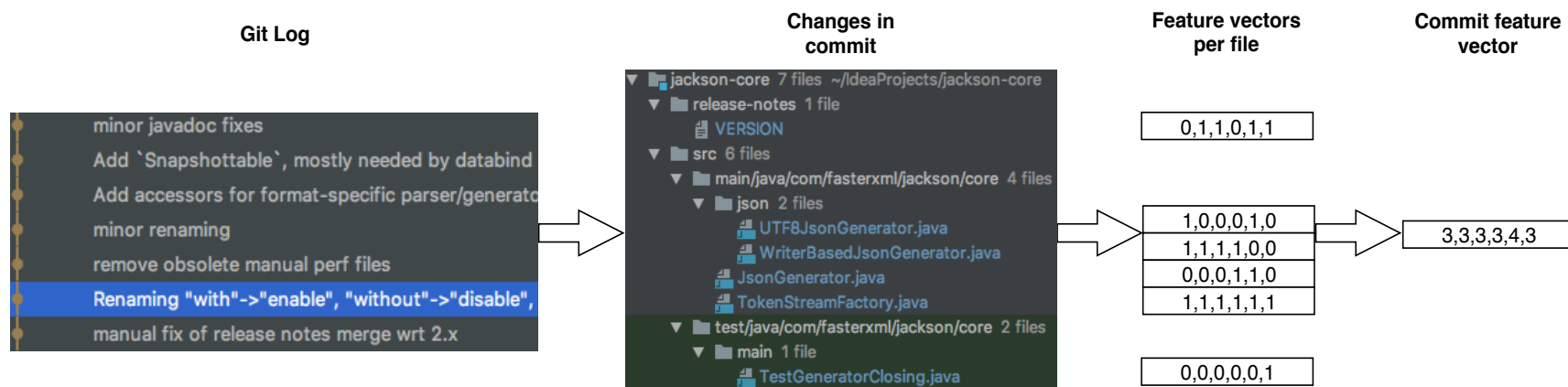
Related work

What data?



Data generation

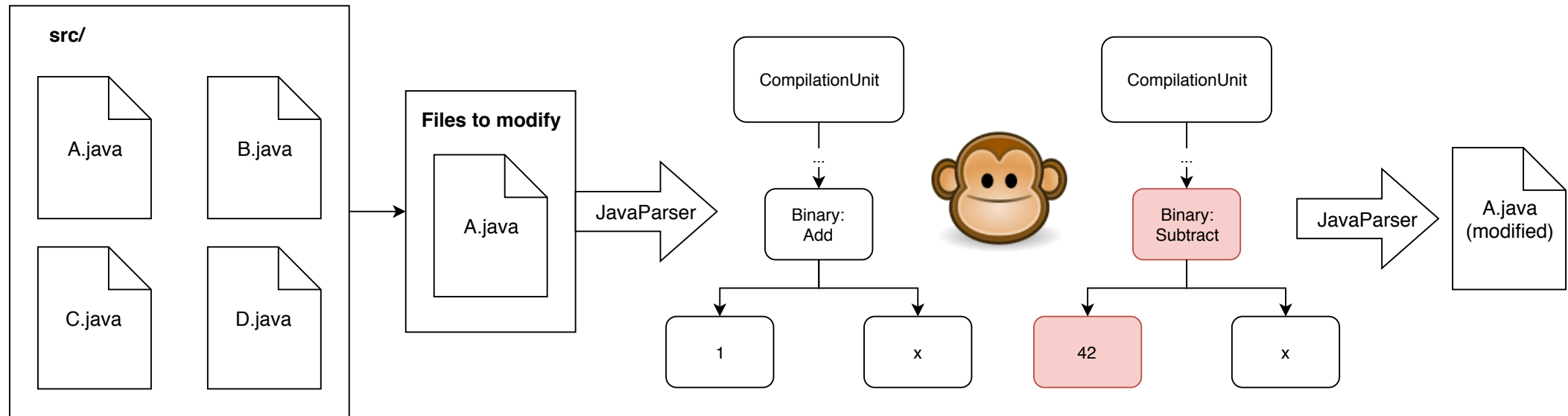
Historical data (e.g. Git log)



natural learning data → best prediction performance

MonkeyRunner

Randomly change source code semantics



Similar: randomly rotating image data

MonkeyRunner

original

```
public class Physics {  
    public int getEnergy(int m, int c) {  
        return m - Math.pow(c, 8);  
    }  
}
```

modified

```
public class Physics {  
    public int getEnergy(int m, int c) {  
        return m * Math.pow(c, 2);  
    }  
}
```

Overview

Data

Vectors for machine learning

Machine learning techniques

Technical aspects

Conclusion

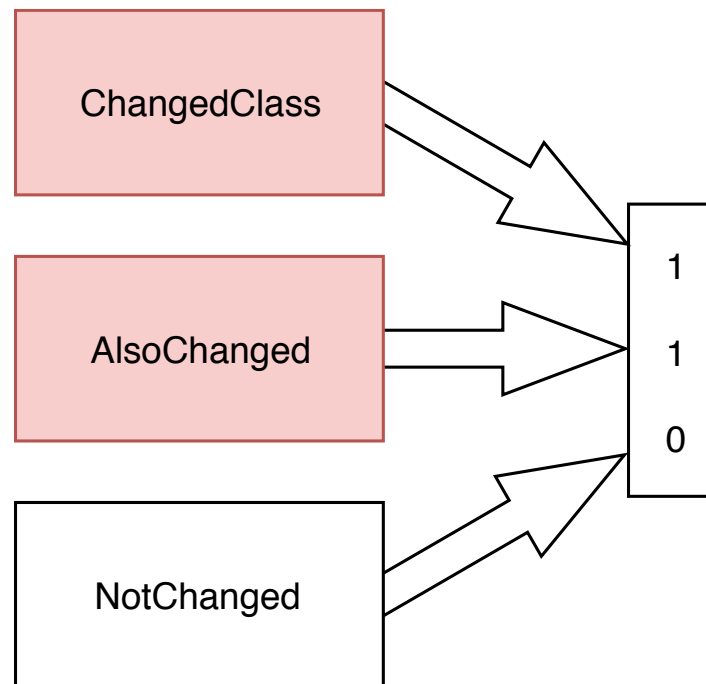
Related work

Vectors?



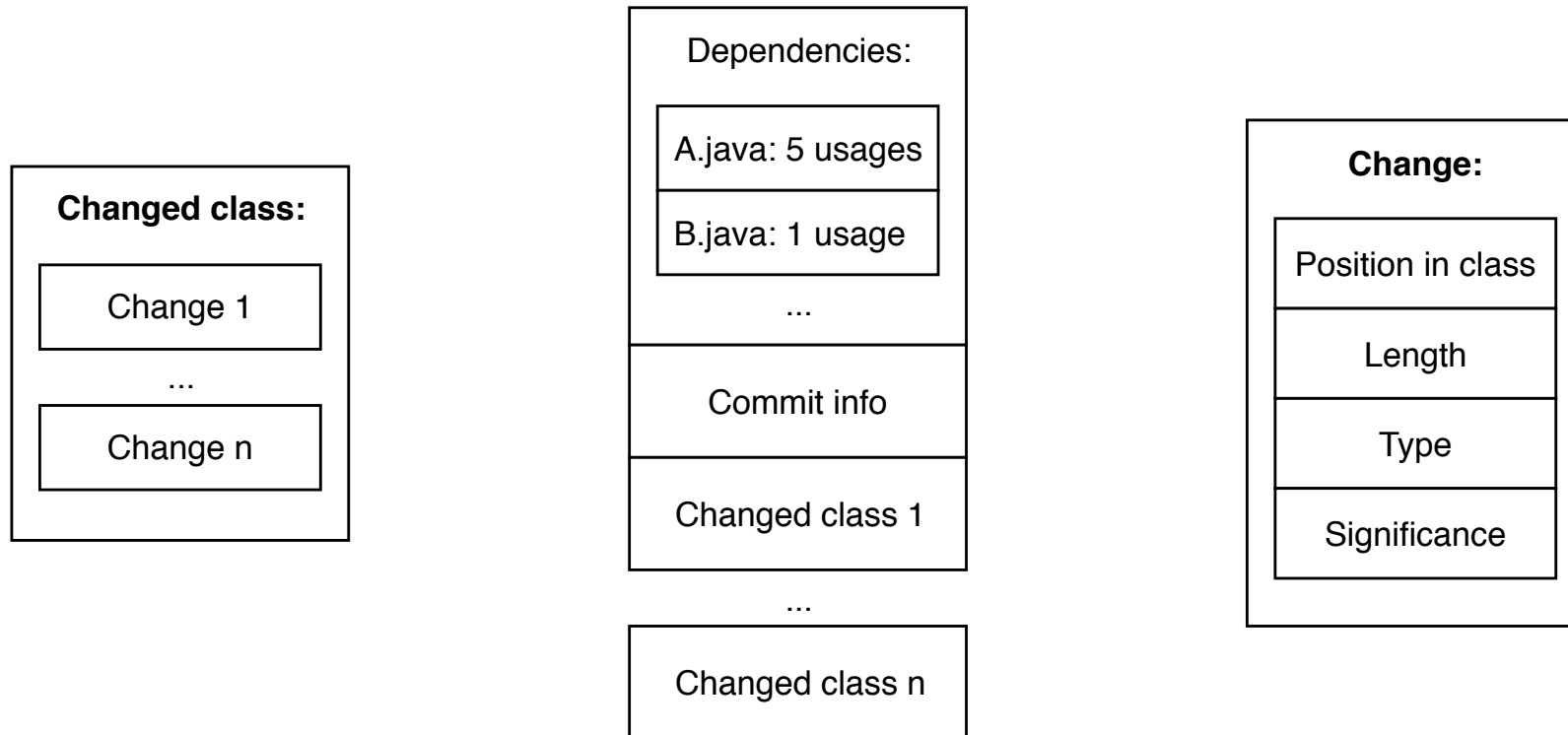
Input vectors

List of changed classes



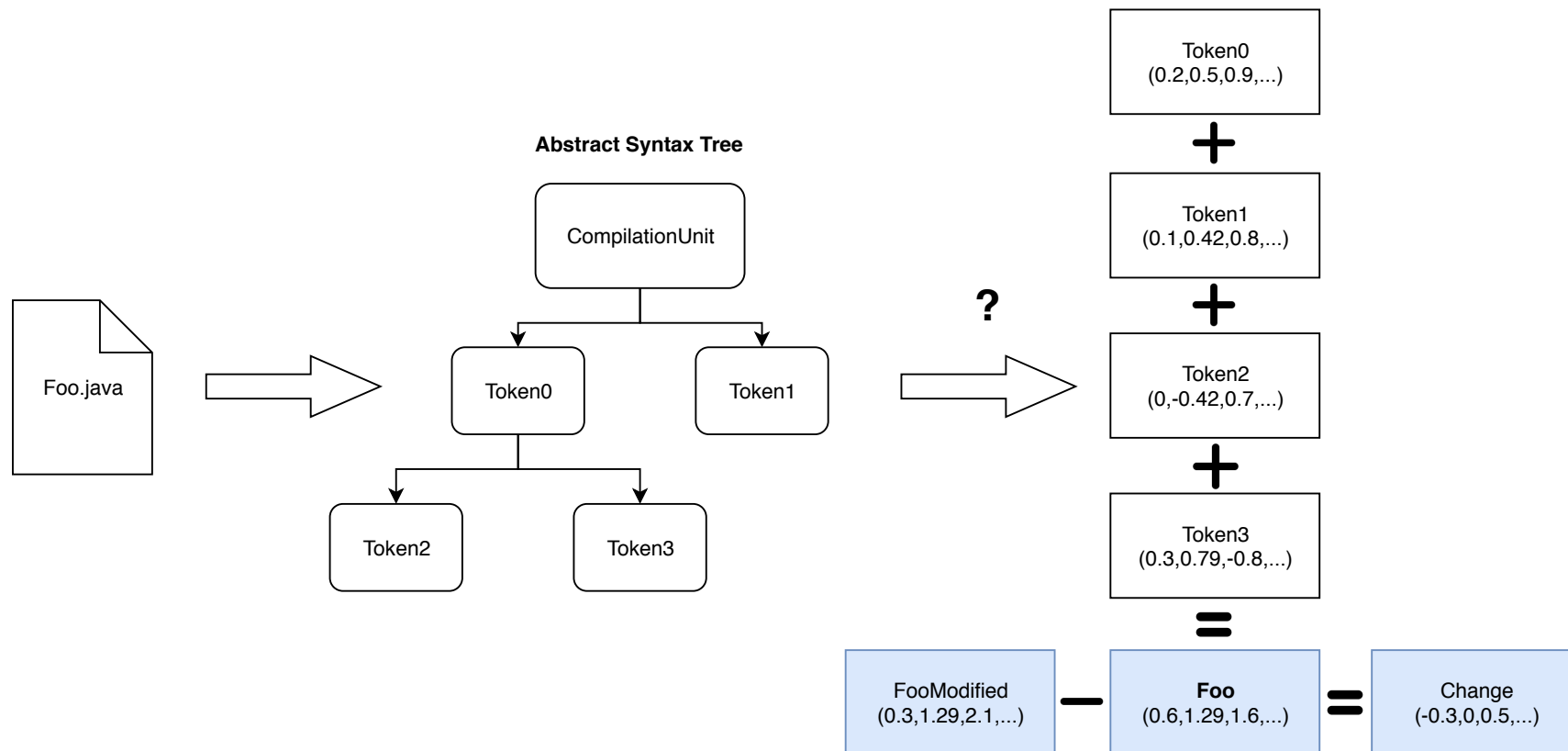
Input vectors

Change types encoded in class



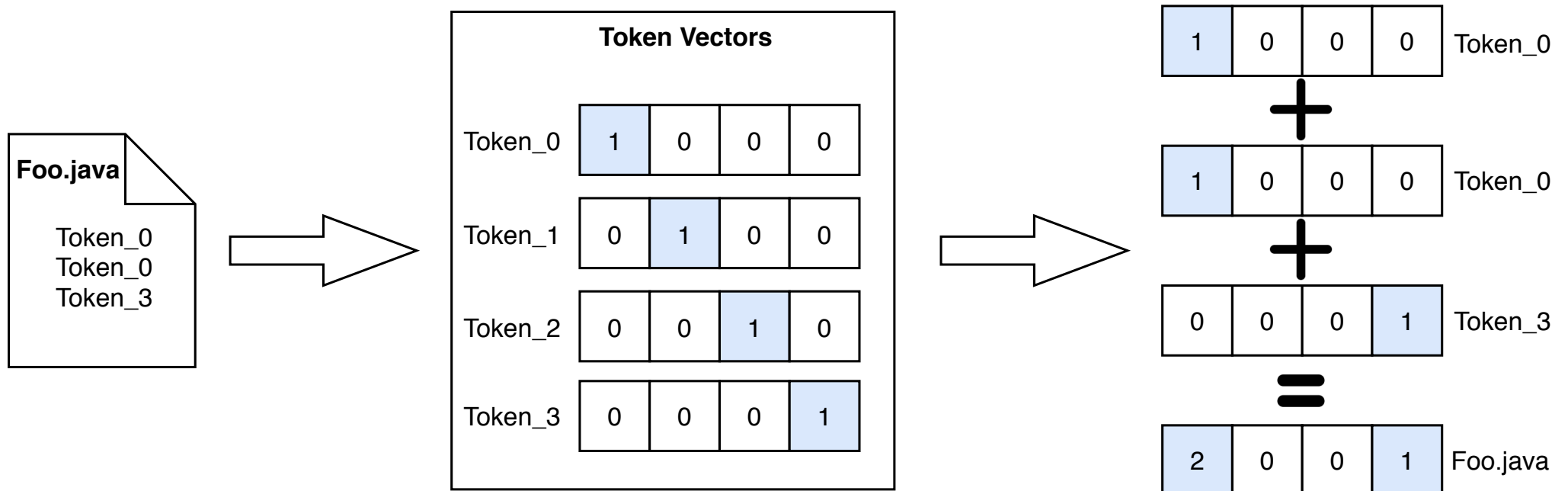
Problem: number of changes not constant

Input vectors based on AST



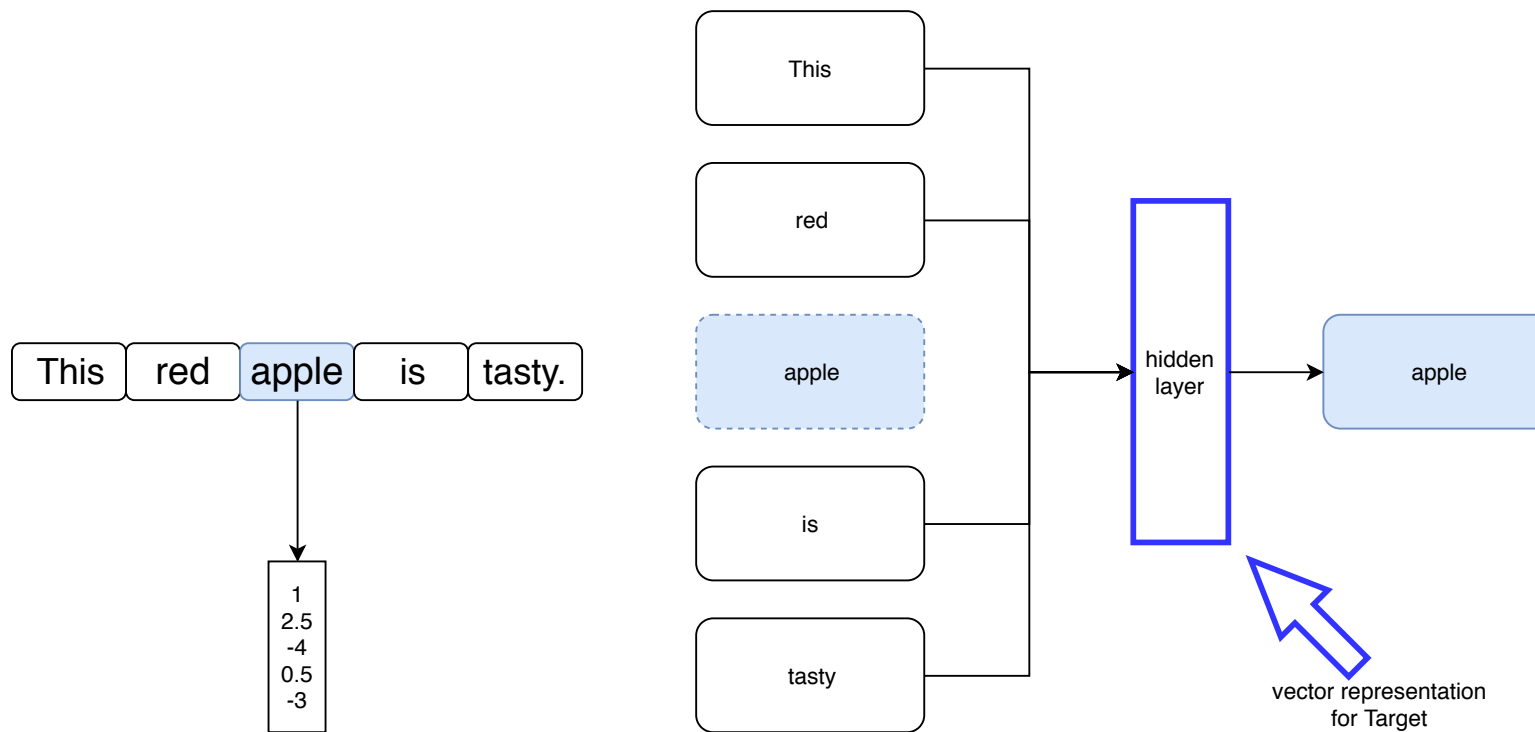
Input vectors based on AST

one-hot encoding



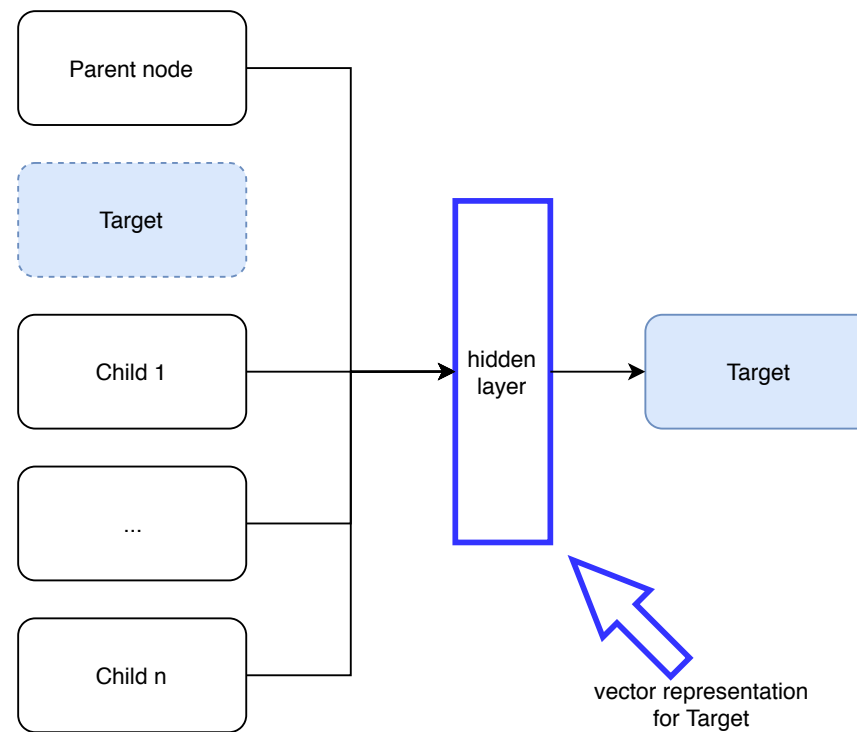
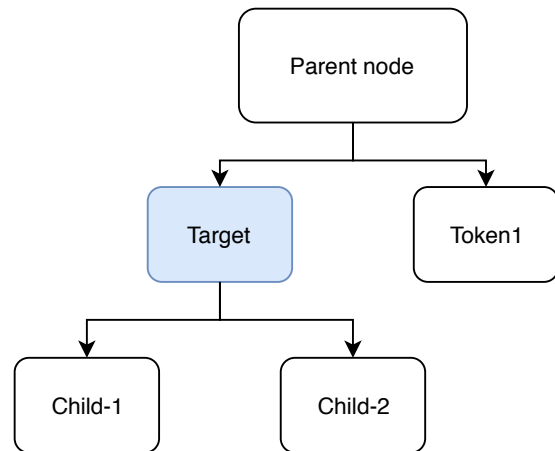
Input vectors based on AST

word2vec

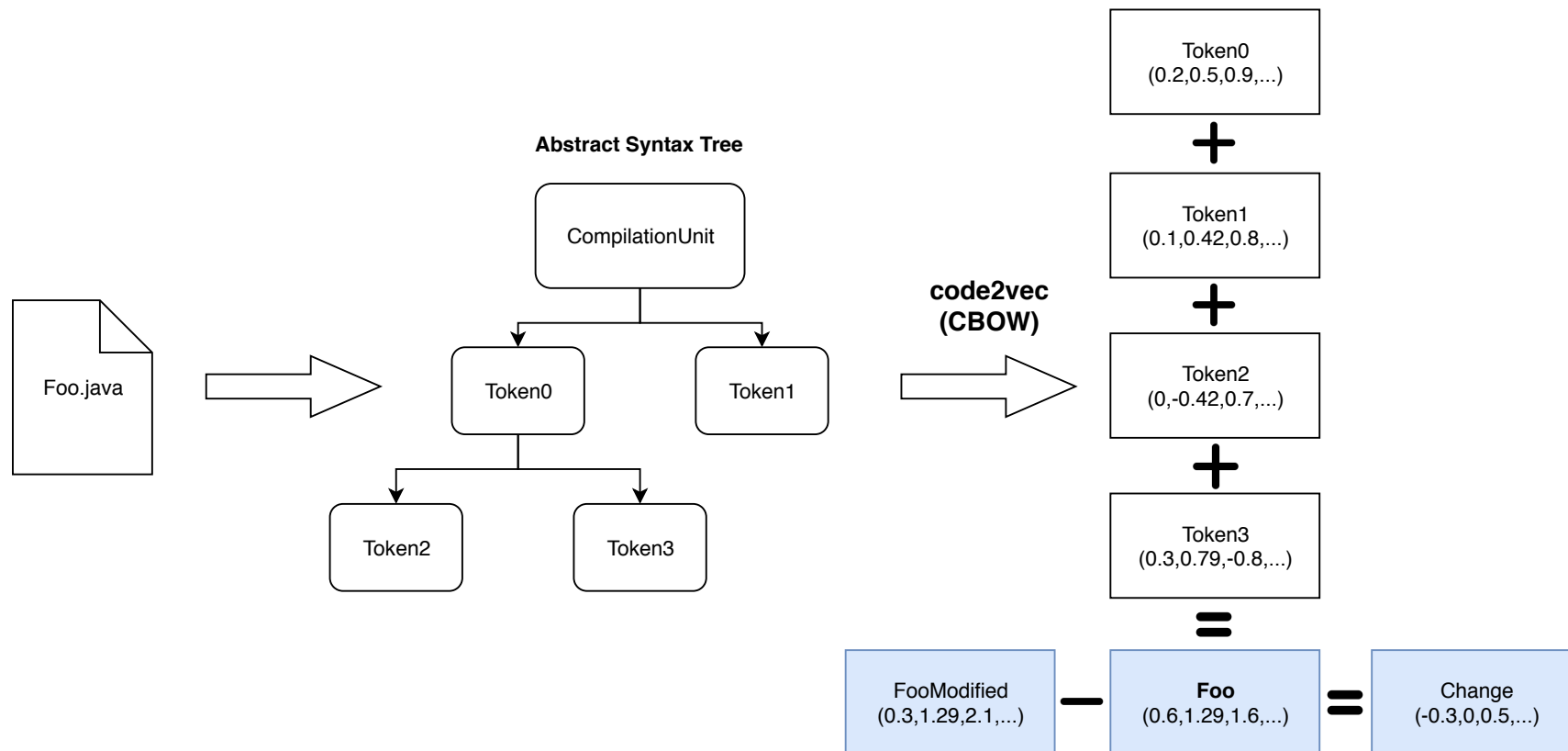


Input vectors based on AST

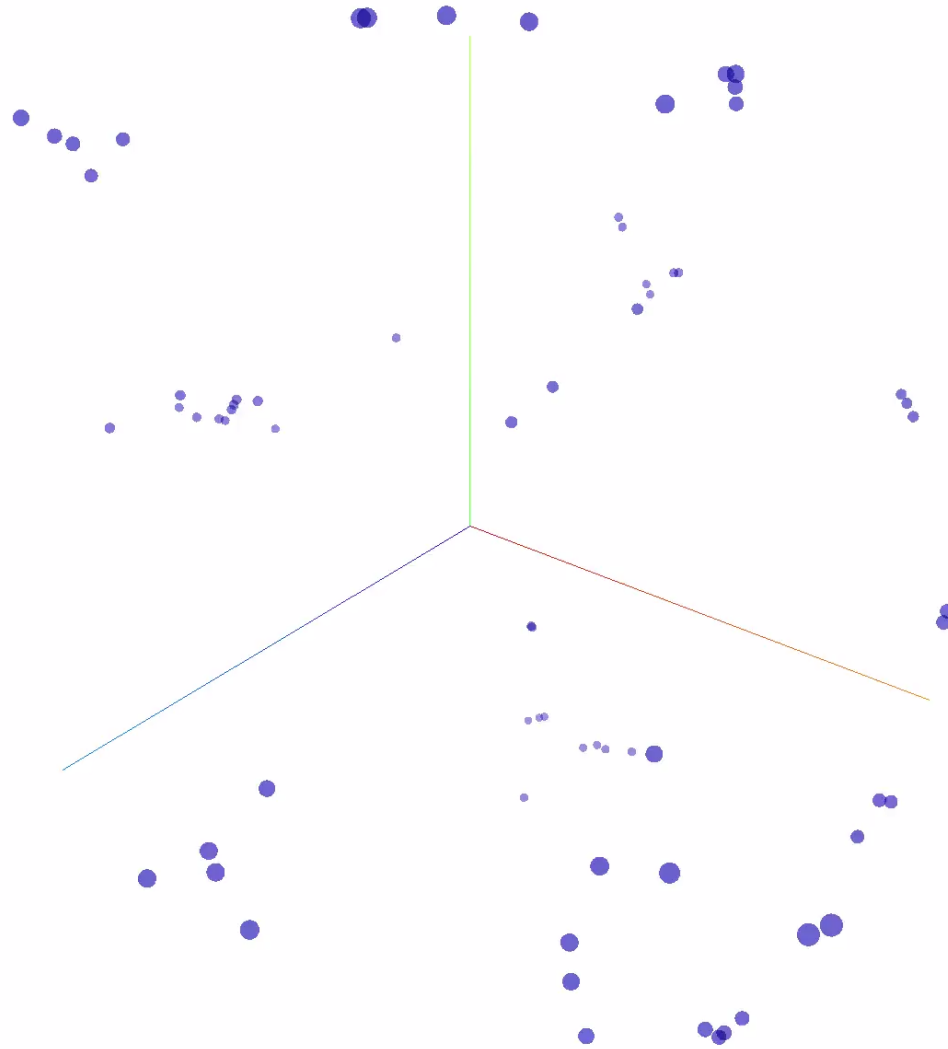
code2vec



Input vectors based on AST



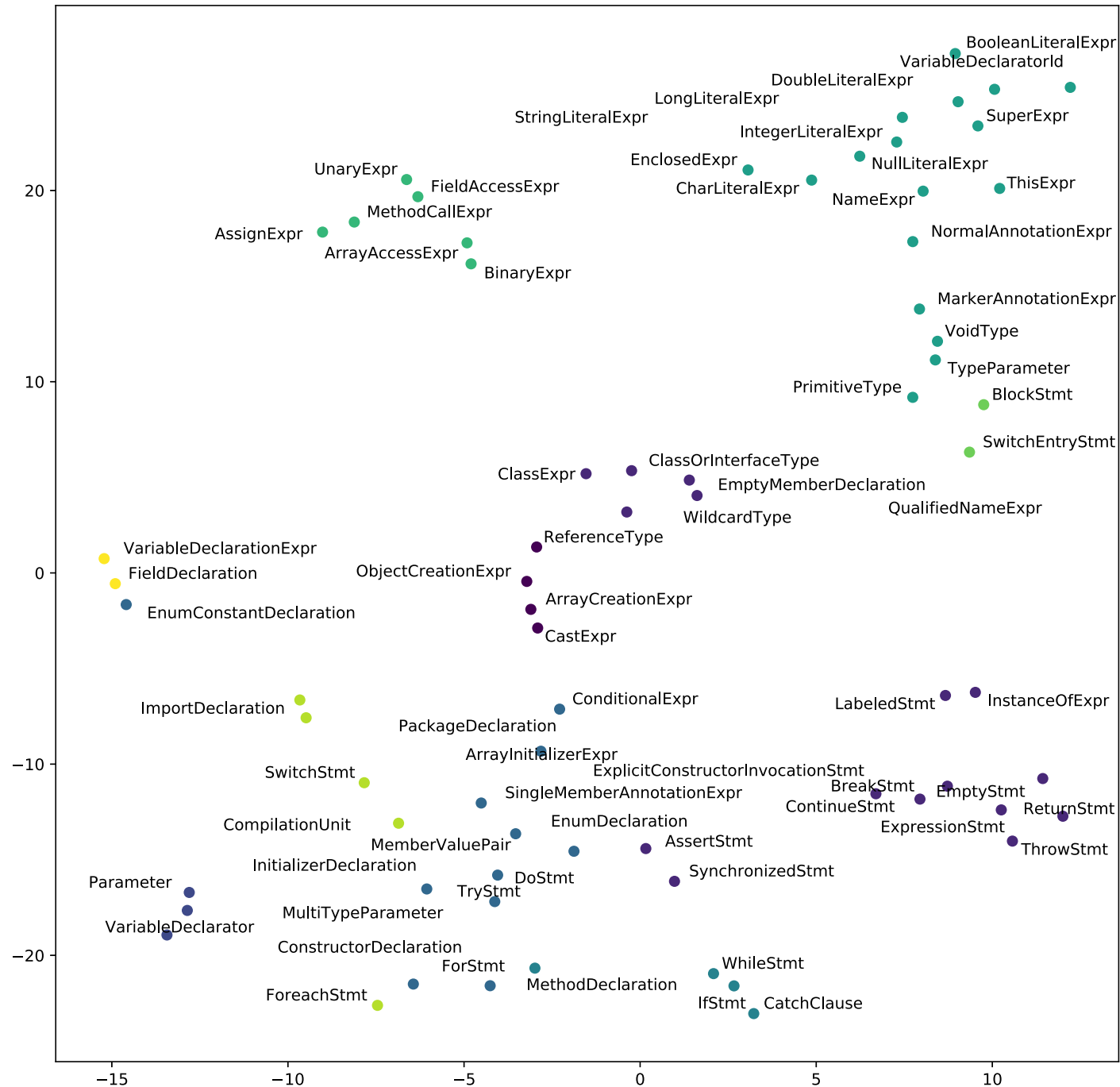
Token clustering with T-SNE



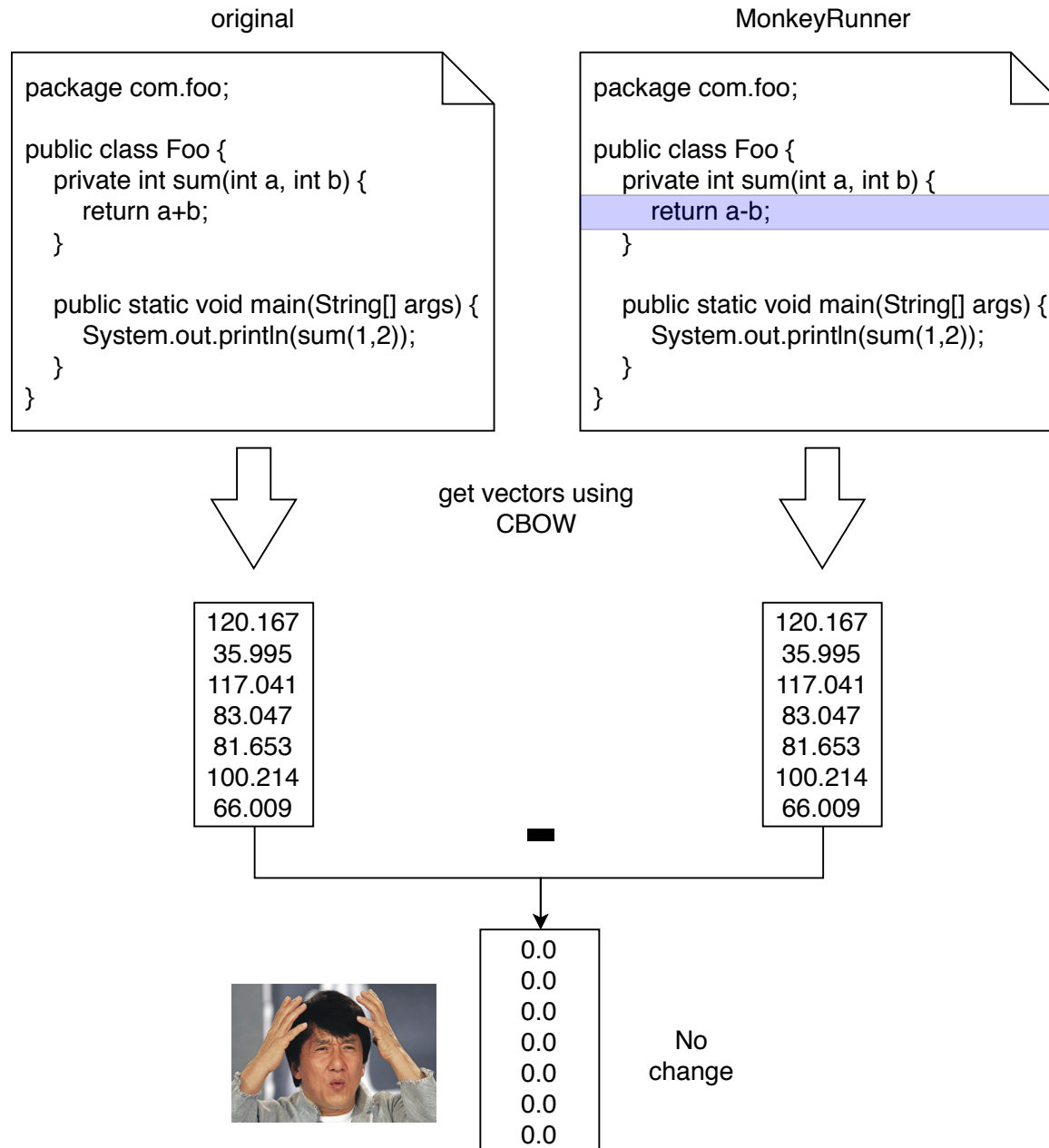
view in Tensorflow Projector:

<https://goo.gl/VUP454>

Token clustering with T-SNE

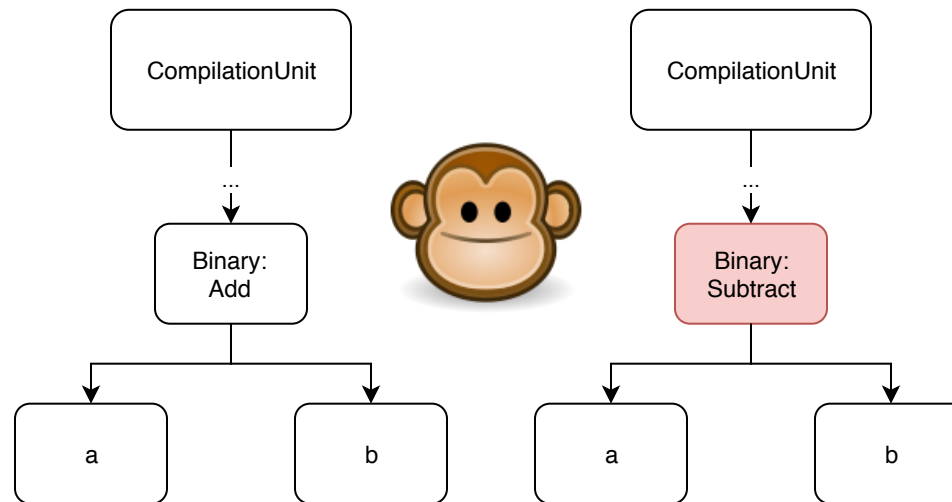


Difficulties



Difficulties

Why??



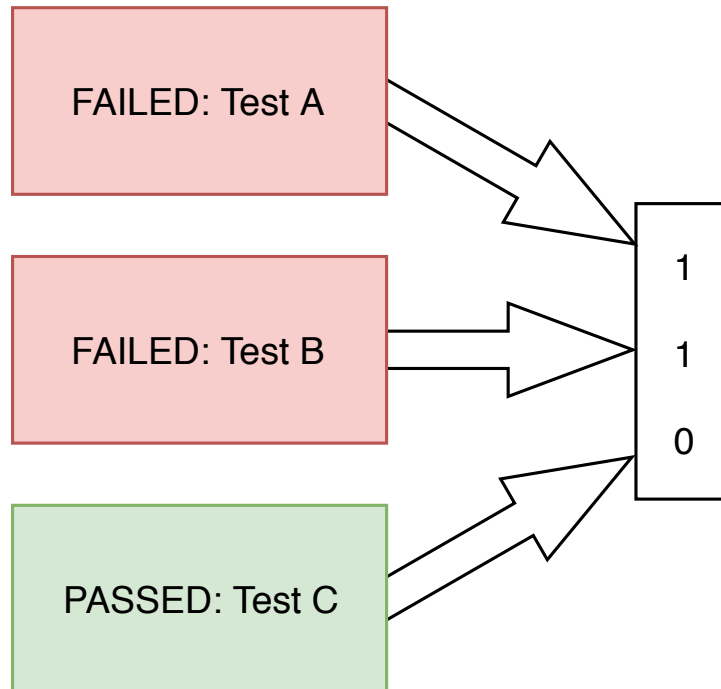
AST structure still the same!

Embed token parameters in vector?

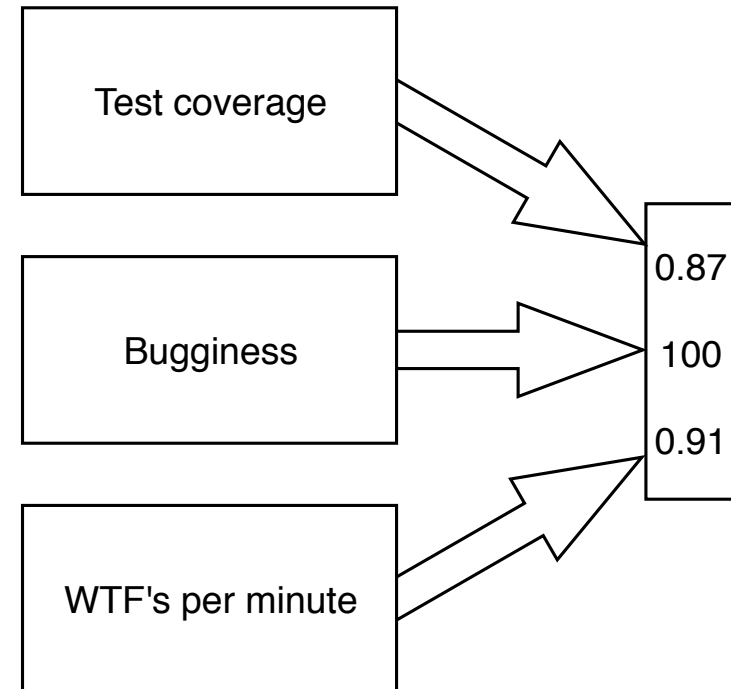
More complex MonkeyRunner changes?

Output vectors

test results

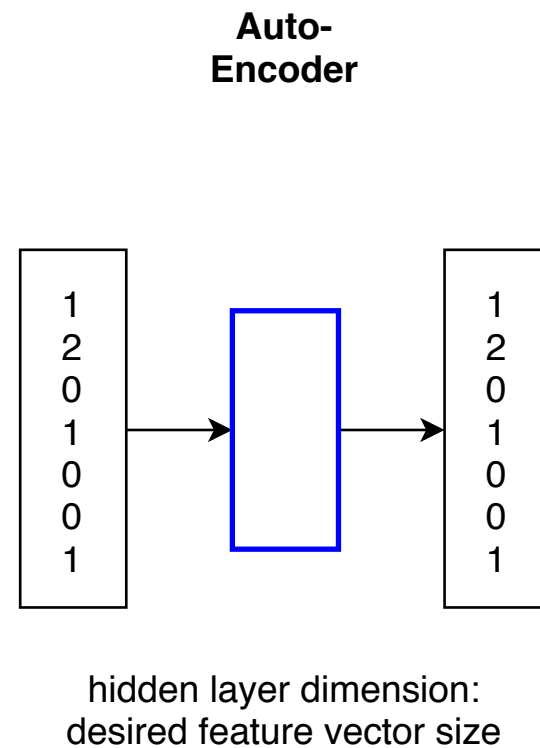
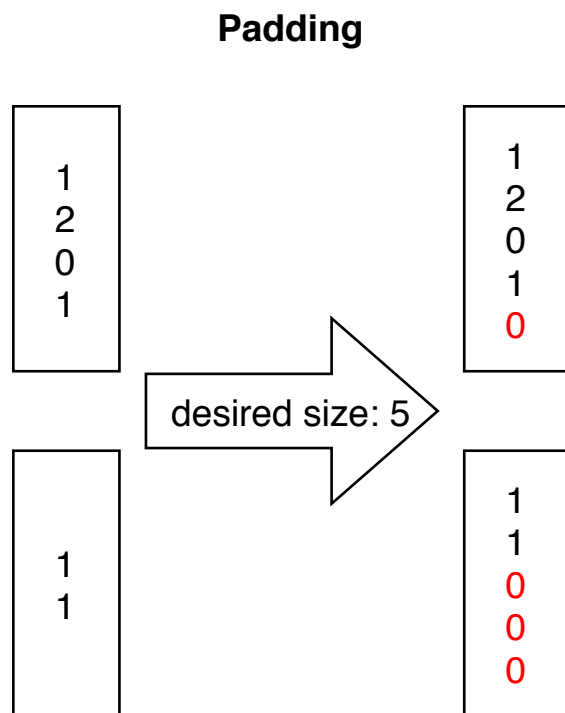


code quality/ other metrics



Difficulties

Vector dimensions not fixed?
(e.g. new classes/tests added)



Overview

Data

Vectors for machine learning

Machine learning techniques

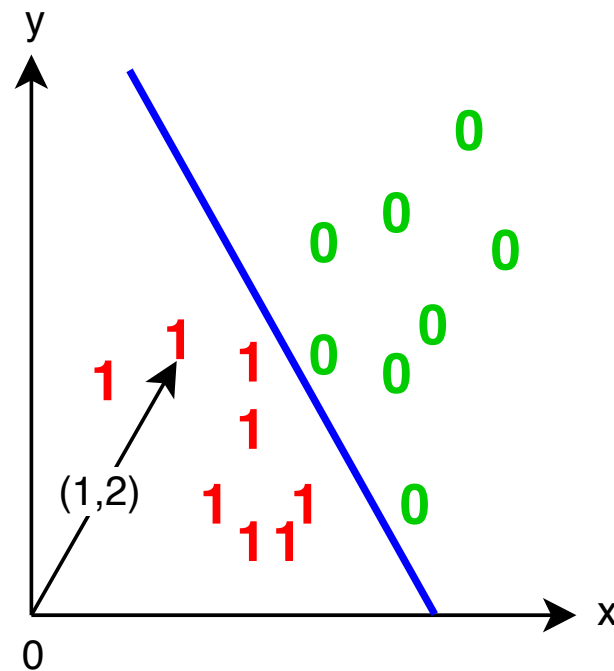
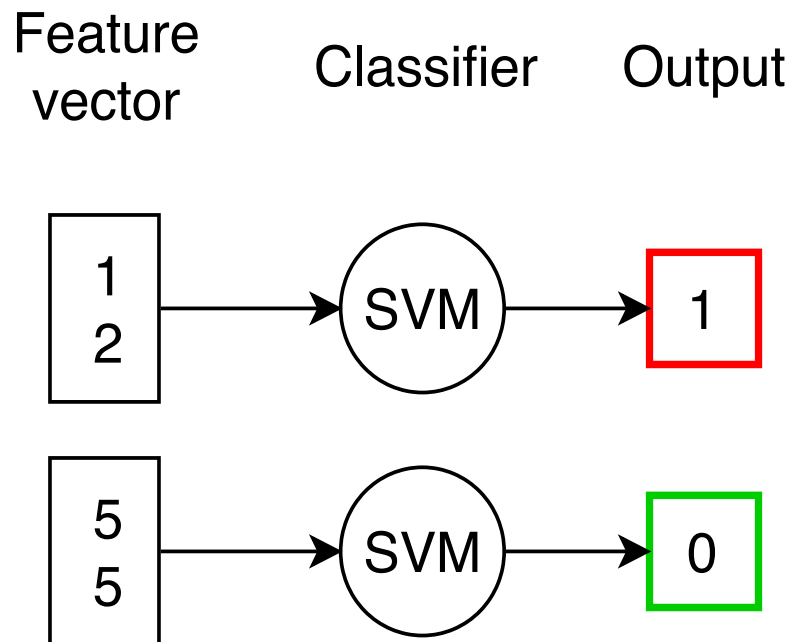
Technical aspects

Conclusion

Related work

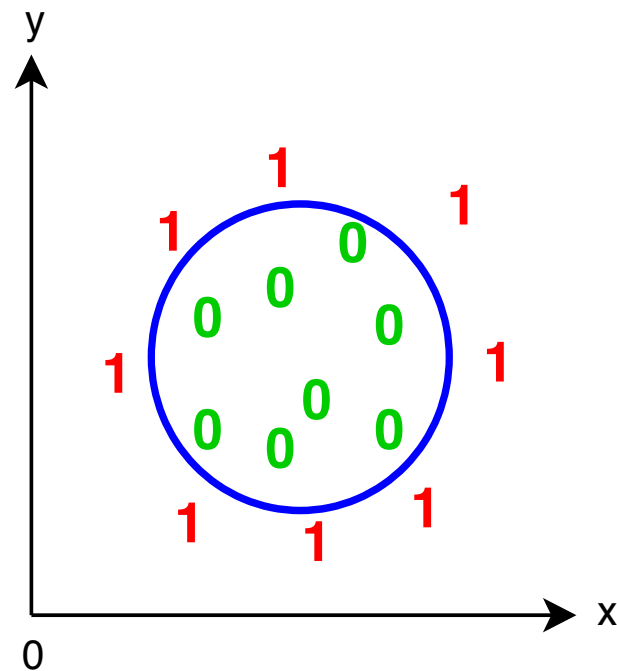
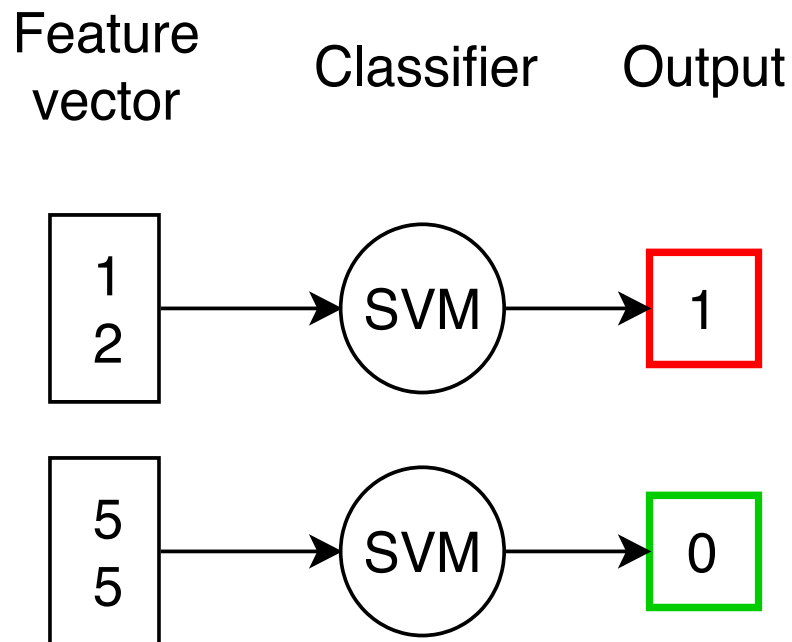
Statistical classification

Support vector machines (SVM)



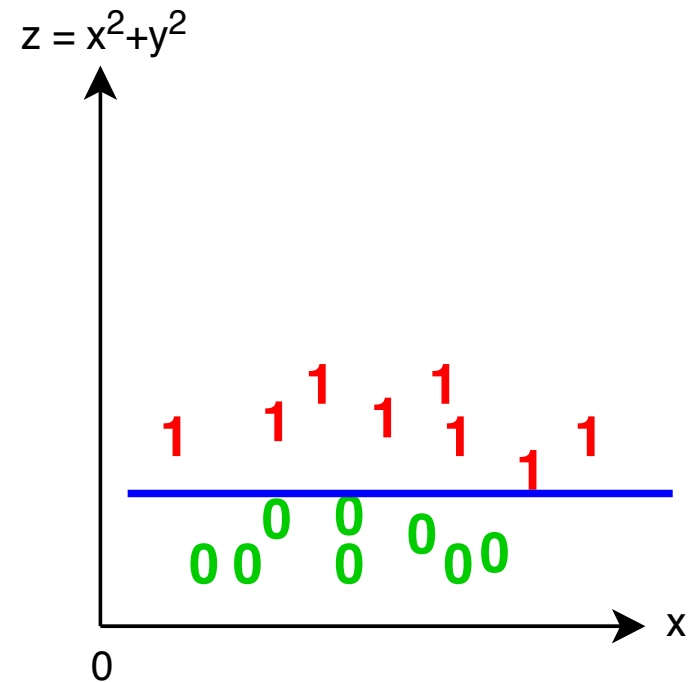
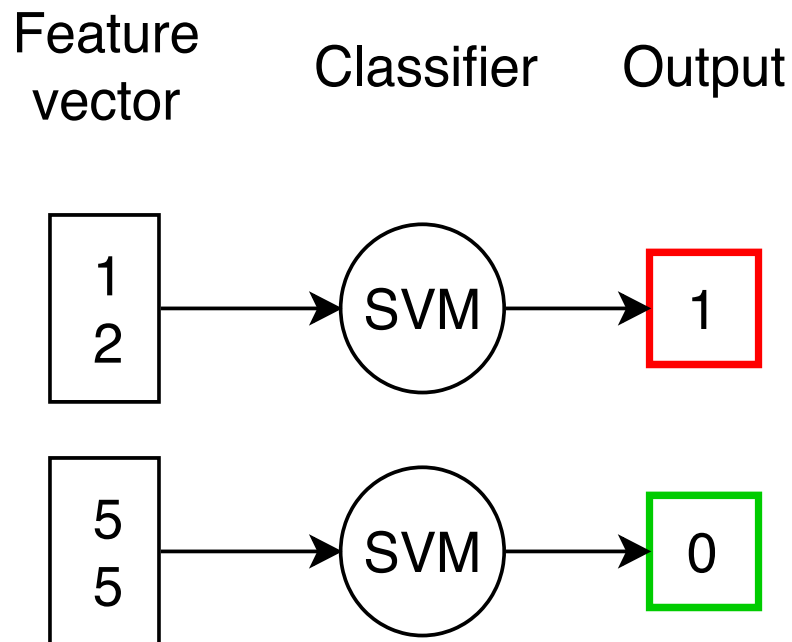
Statistical classification

Support vector machines (SVM)



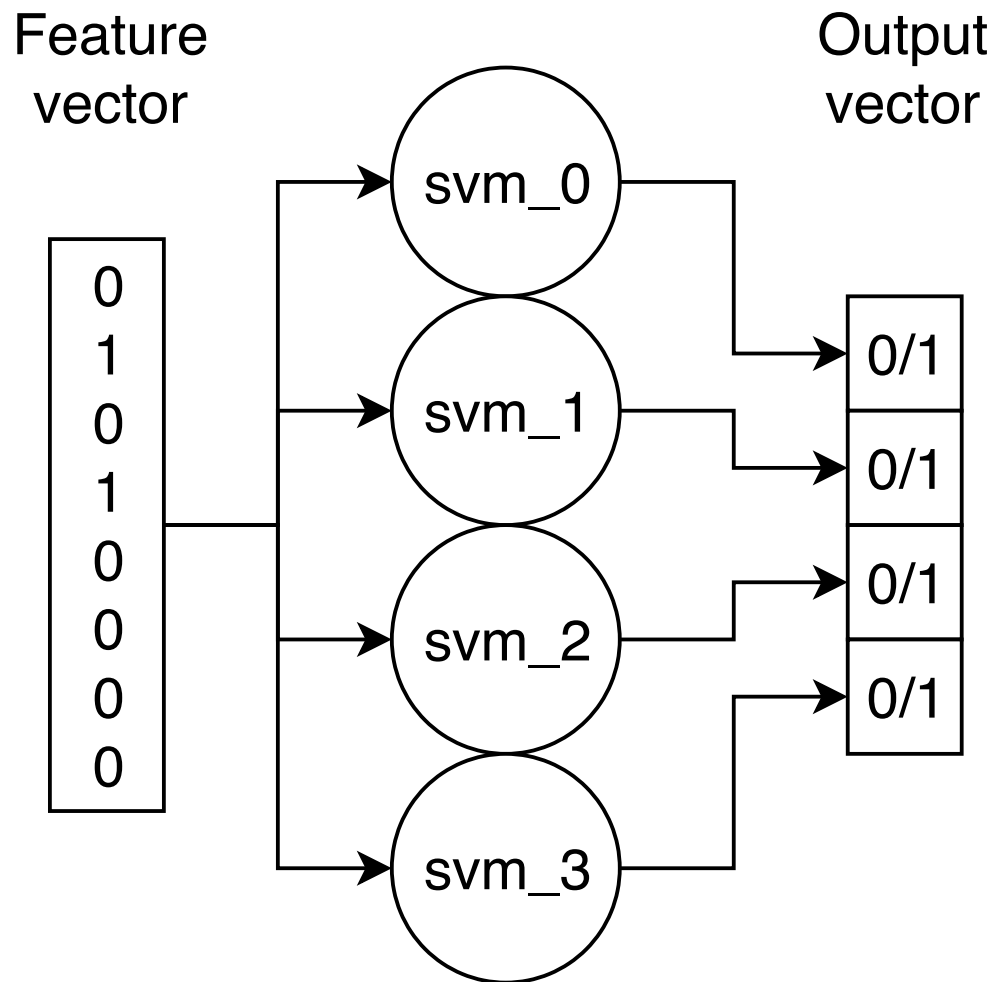
Statistical classification

Support vector machines (SVM)



Statistical classification

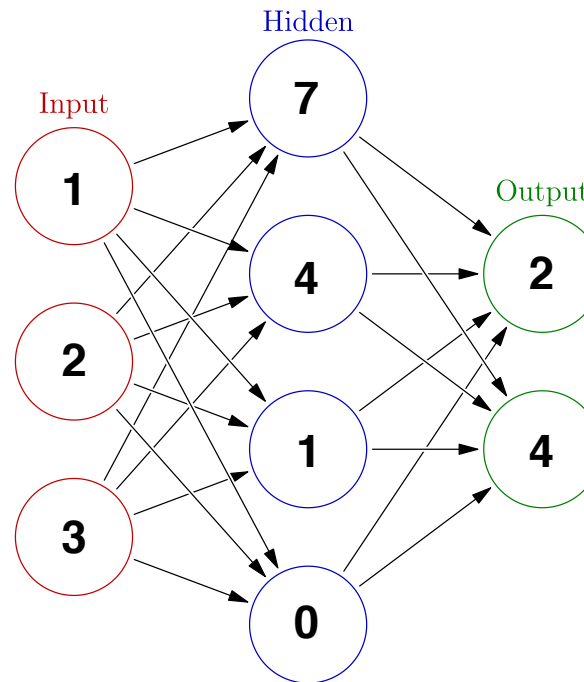
SVM with multilabel data



Neural network

Simple example: fully connected, one hidden layer

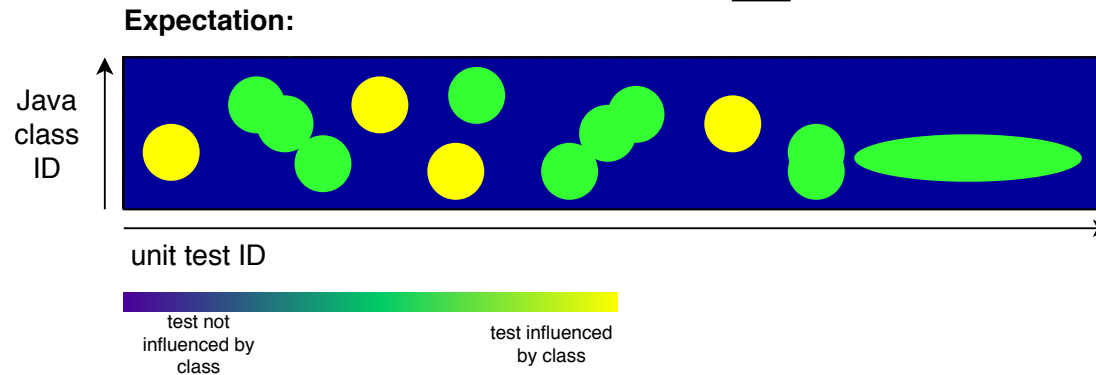
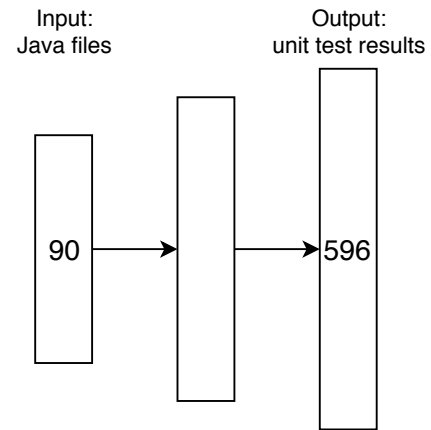
Training data	
<div>1 2 3</div>	<div>0 1</div>
Input	Output



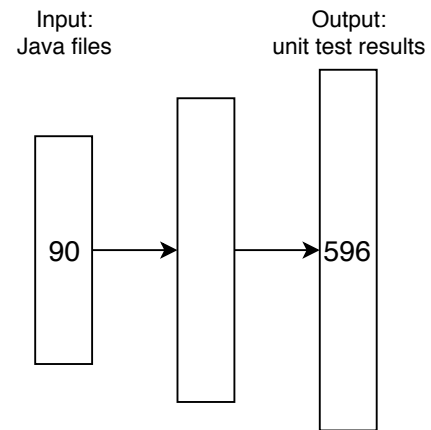
Predicted	Cost	Expected
<div>2 4</div>	$(2-0)^2$ + $(4-1)^2$	<div>0 1</div>

Goal: minimize cost
How: modify weights

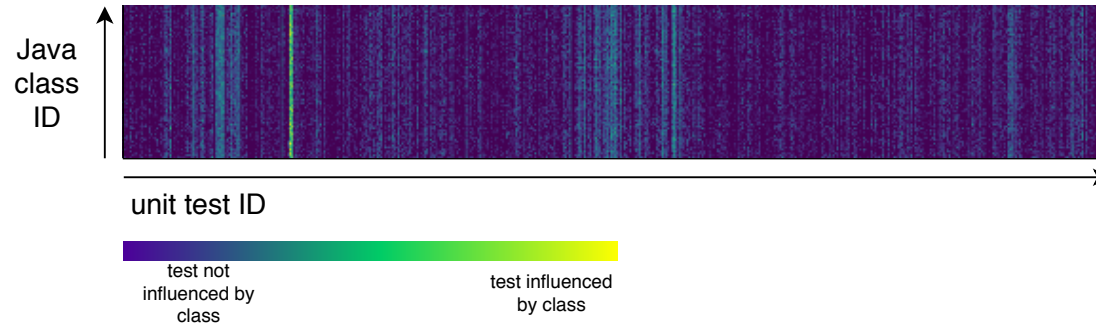
Neural network



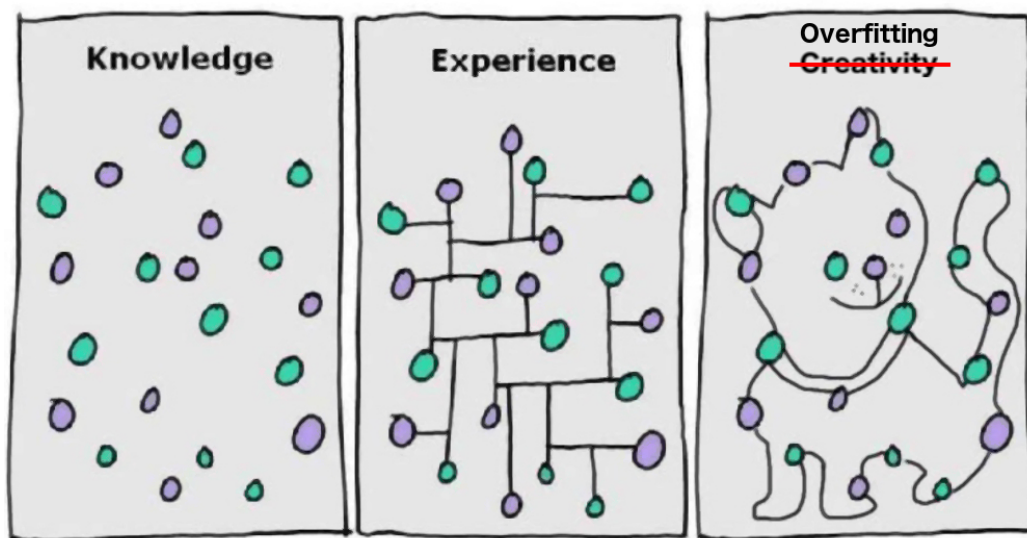
Neural network



Reality:



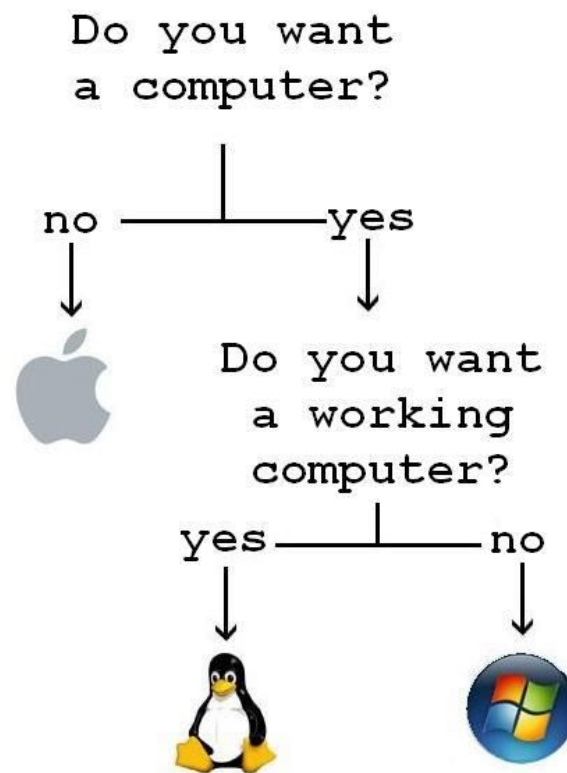
Overfitting



based on: Steve Wheeler, slideshare.net

Rule of thumb: ~10 samples per parameter
Our neural net: >120k weights → 1.2 million samples needed

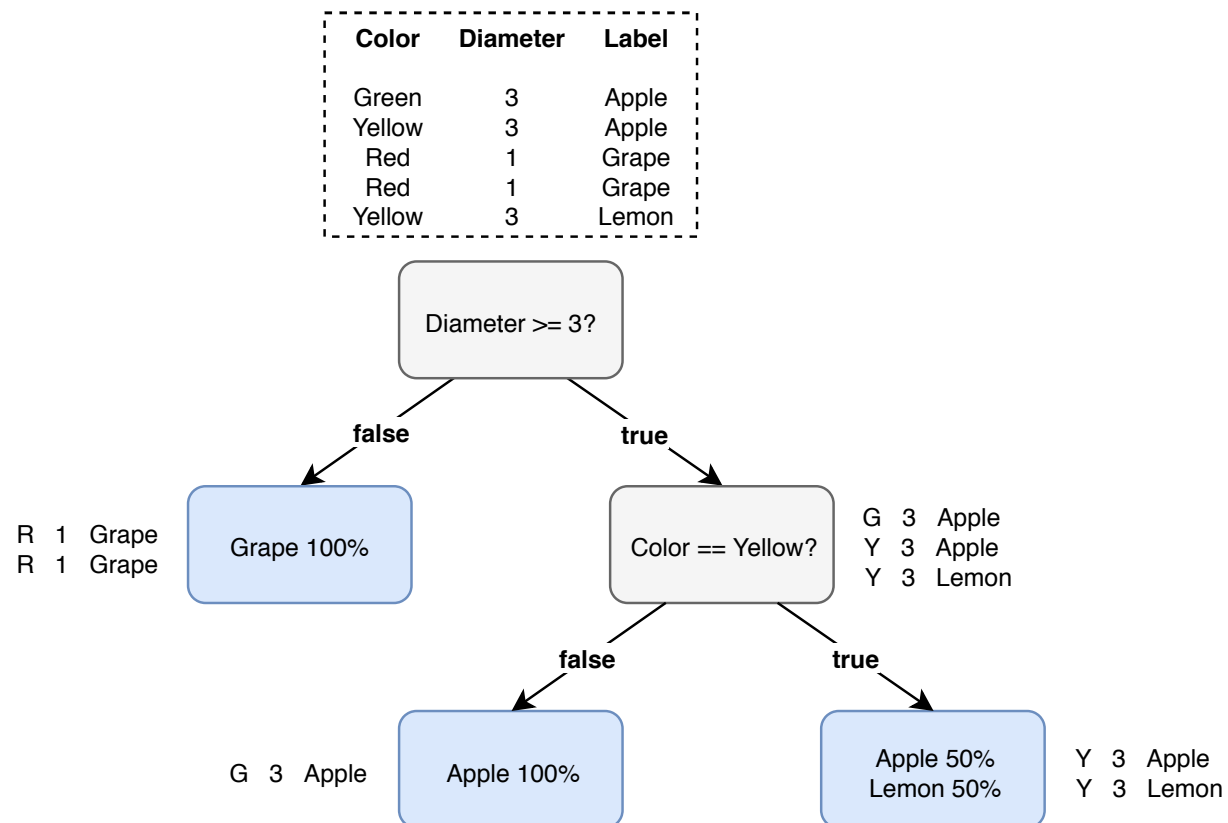
Decision trees



<https://www.pinterest.de/pin/80642649554451367>

Decision trees

Training



Overview

Data

Vectors for machine learning

Machine learning techniques

Technical aspects

Conclusion

Related work

Frameworks

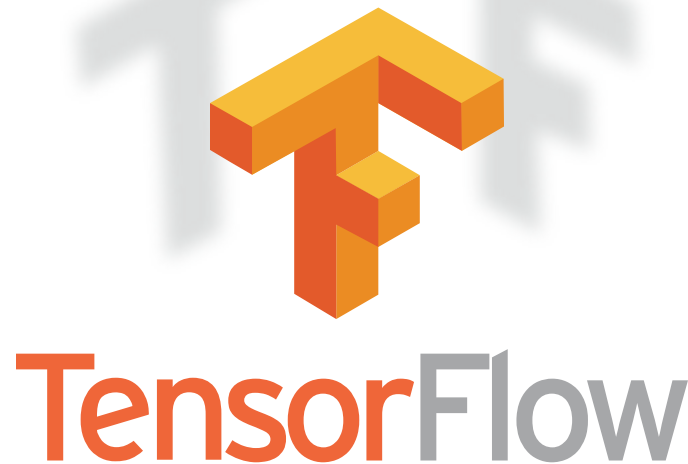


since 2007

Classification, regression and clustering algorithms

Language: Python with NumPy

Frameworks



since 2015

Calculations on tensors (more generalized form of vectors)

Popular for neural networks

Languages: Python, (Java, Go, C)

Simple neural network: Tensorflow

source: <https://gist.github.com/vinhkhuc/e53a70f9e5c3f55852b0>

```
def init_weights(shape):
    weights = tf.random_normal(shape, stddev=0.1)
    return tf.Variable(weights)

def forwardprop(X, w_1, w_2):
    h = tf.nn.sigmoid(tf.matmul(X, w_1)) # The \sigma function
    yhat = tf.matmul(h, w_2) # The \varphi function
    return yhat

def get_data():
    dataset = load_dataset()
    data = dataset["data"]
    target = dataset["target"]
    N, M = data.shape
    all_X = np.ones((N, M + 1))
    all_X[:, 1:] = data
    num_labels = len(np.unique(target))
    all_Y = np.eye(num_labels)[target] # One liner trick!
    return train_test_split(all_X, all_Y, test_size=0.33, random_state=RANDOM_SEED)

def main():
    train_X, test_X, train_y, test_y = get_data()
    x_size = train_X.shape[1] # Number of input nodes: 90 features
    h_size = 180 # Number of hidden nodes
    y_size = train_y.shape[1] # Number of outcomes (596 unit tests)
    X = tf.placeholder("float", shape=[None, x_size])
    y = tf.placeholder("float", shape=[None, y_size])
    w_1 = init_weights((x_size, h_size))
    w_2 = init_weights((h_size, y_size))
    yhat = forwardprop(X, w_1, w_2)
    predict = tf.argmax(yhat, axis=1)
    cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=y, logits=yhat))
    updates = tf.train.GradientDescentOptimizer(0.01).minimize(cost)
    sess = tf.Session()
    init = tf.global_variables_initializer()
    sess.run(init)
    for epoch in range(100):
        for i in range(len(train_X)):
            sess.run(updates, feed_dict={X: train_X[i: i + 1], y: train_y[i: i + 1]})
        train_accuracy = np.mean(np.argmax(train_y, axis=1) ==
                                sess.run(predict, feed_dict={X: train_X, y: train_y}))
        test_accuracy = np.mean(np.argmax(test_y, axis=1) ==
                                sess.run(predict, feed_dict={X: test_X, y: test_y}))
        print("Epoch = %d, train accuracy = %.2f%%, test accuracy = %.2f%%"
              % (epoch + 1, 100. * train_accuracy, 100. * test_accuracy))
    sess.close()
```



Frameworks



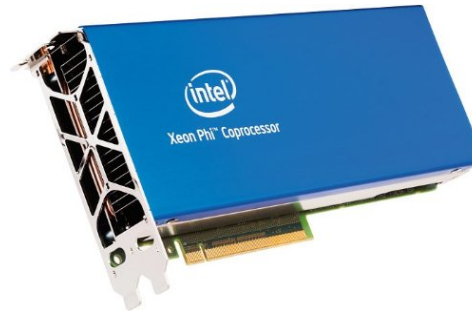
Keras (since 2015)

```
hidden_layer = Dense(180, input_shape=(90,))  
output_layer = Dense(596)  
model = Sequential([hidden_layer, output_layer])  
model.compile(loss='mean_squared_error', optimizer='sgd')  
model.fit(X_train, Y_train)
```

Specialized hardware



GPU learning



Intel Xeon + MKL



Movidius NCS
(e.g. for Raspberry Pi)



Clusters!

<https://www.nvidia.com/en-us/titan/titan-xp/>

<https://www.amazon.com/Intel-Xeon-Phi-7120P-Coprocessor/dp/B00FKG9R2Q>

<https://developer.movidius.com/buy>

Overview

Data

Vectors for machine learning

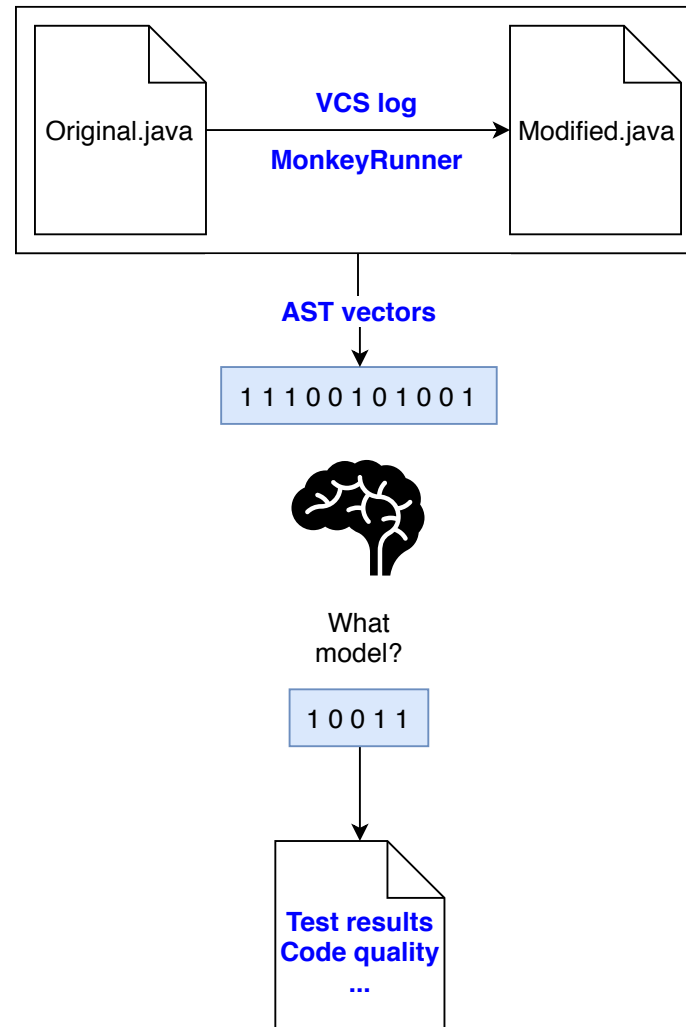
Machine learning techniques

Technical aspects

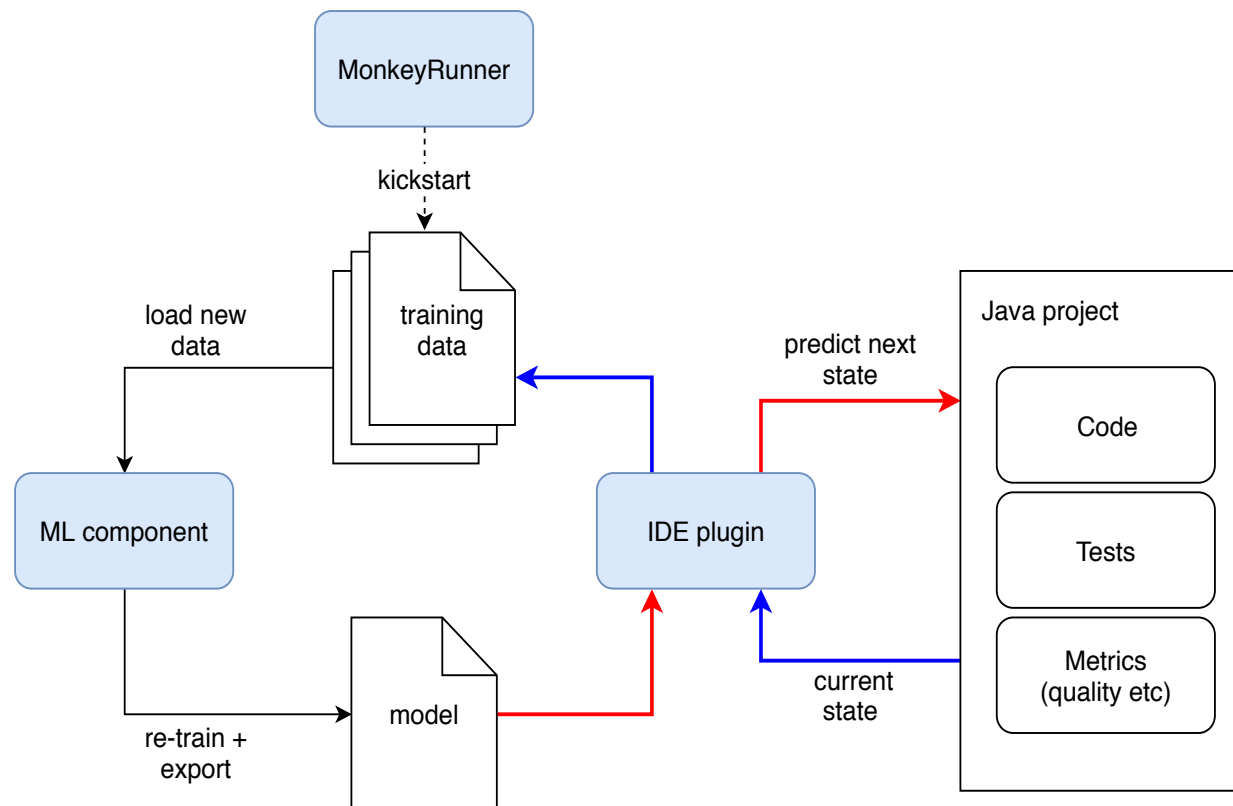
Conclusion

Related work

Recap: challenges



How could it be integrated?



Overview

Data

Vectors for machine learning

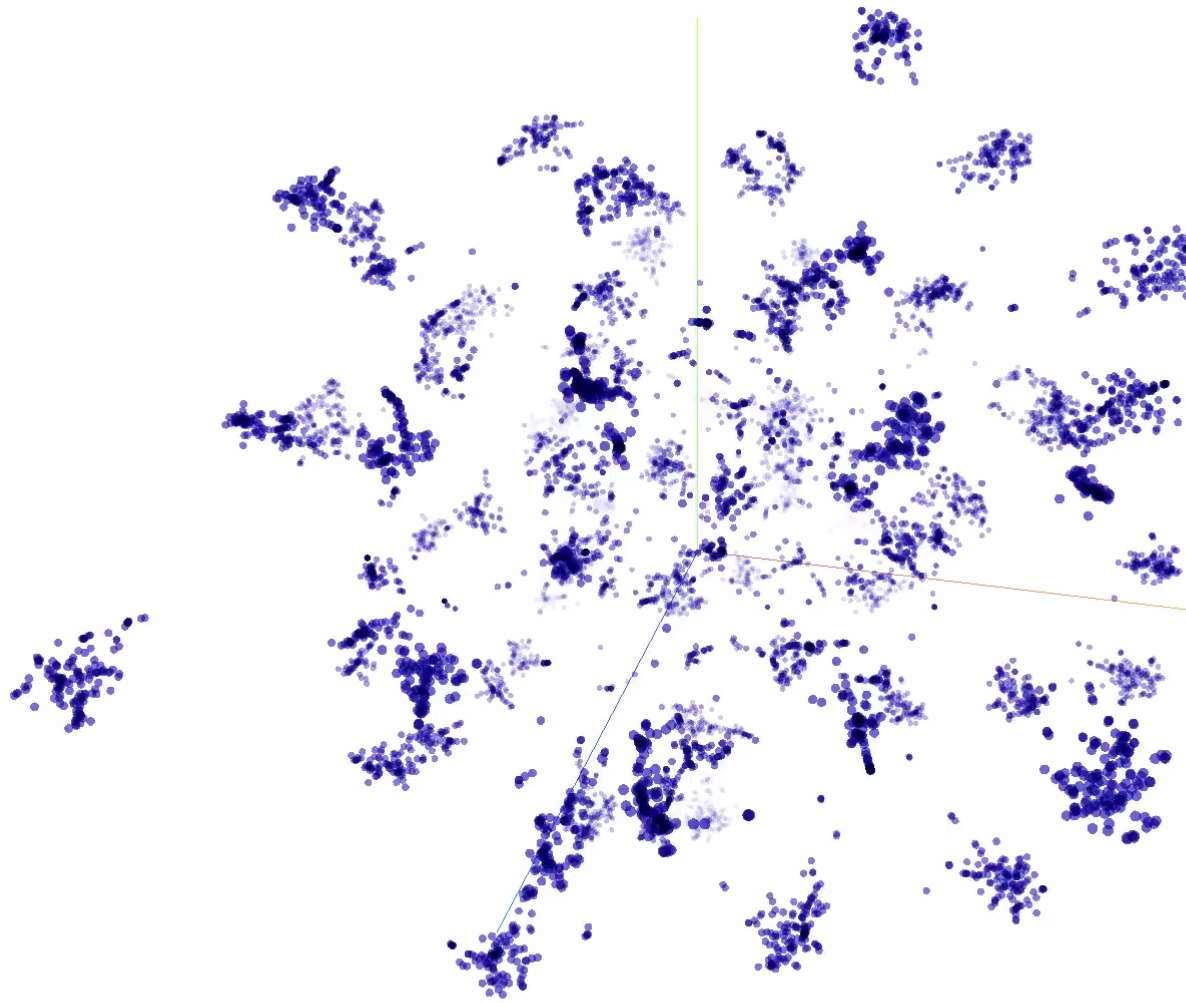
Machine learning techniques

Technical aspects

Conclusion

Related work

Clustering GitHub repos

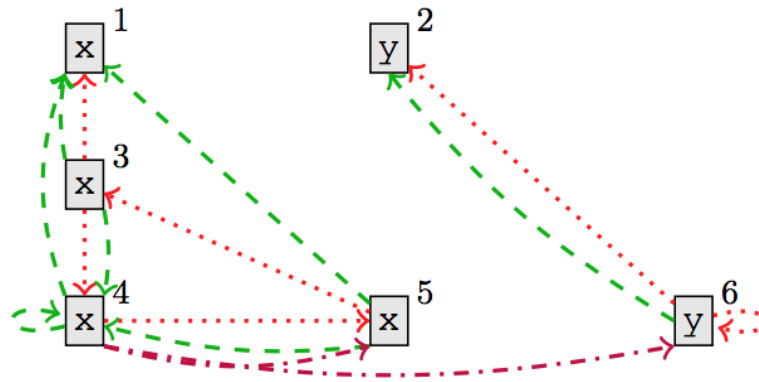


<http://vmarkovtsev.github.io/techtalks-2017-moscow>

VarNaming/VarMisuse

Source code as graph

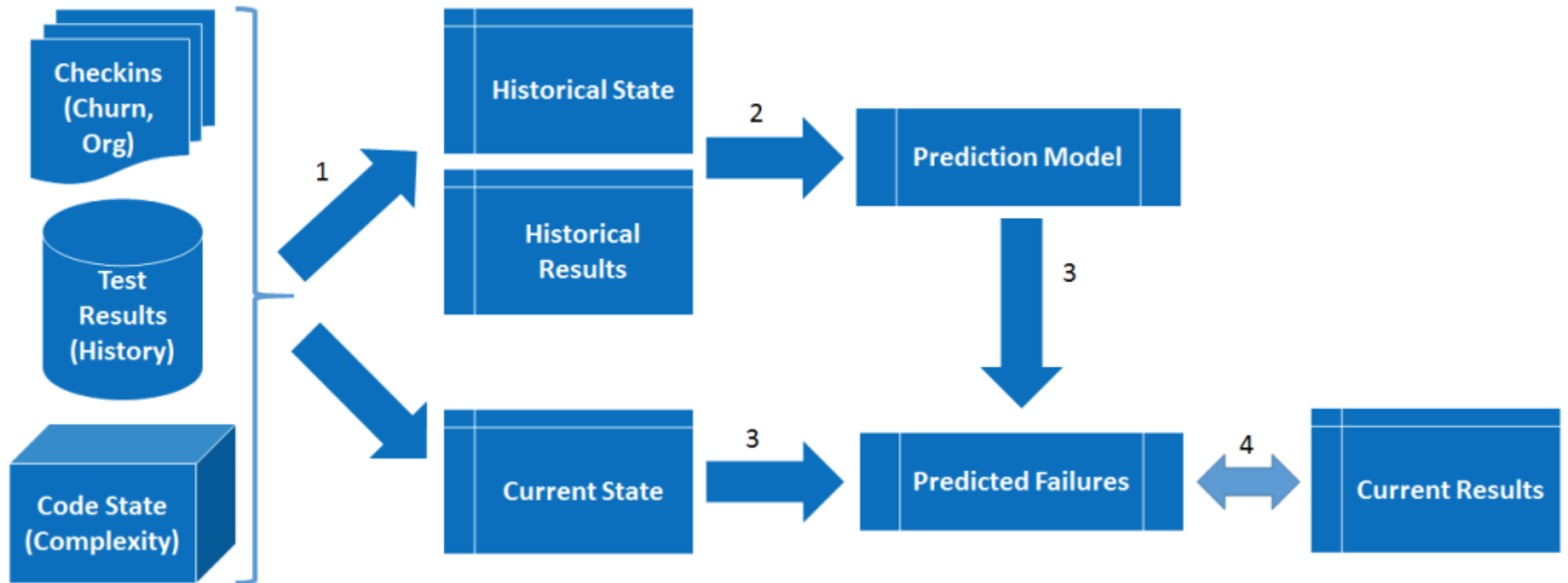
VarNaming/VarMisuse
(Source Code as Graph)



<https://arxiv.org/pdf/1711.00740.pdf>

Test failure prediction

Microsoft Dynamics AX 2012



Thank You!



Samuel Hopstock, samuel.hopstock@tngtech.com

Thomas Endres, thomas.endres@tngtech.com

