Predicting Game Genres by Analyzing Code Structure

Authors: Spencer Frazier, Alex Morehead, Steven Prine, Emil Petersson
Faculty Mentor: Joseph Kendall-Morwick, Ph.D.
Department of Computer Science, Mathematics, & Physics, Missouri Western State University

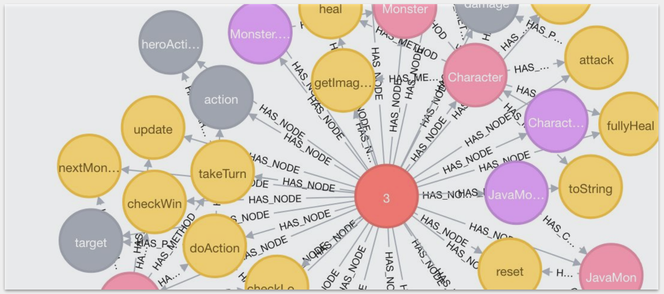
Introduction:

Recent research endeavors in bioinformatics and other machine learning fields have taken a topological approach to classifying data structured as graphs. One such study (Li et al. 2011) has accomplished this through the use of Support Vector Machines (SVM) to analyze the latent features of chemical compounds, proteins, and cell graphs.

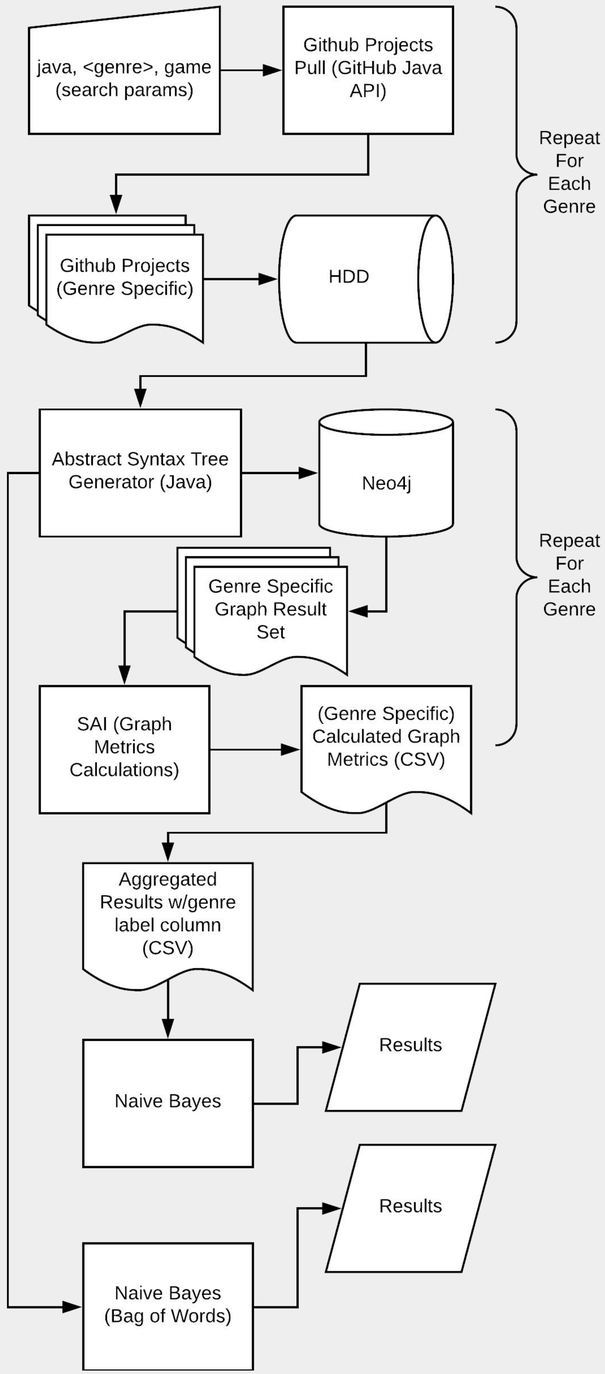
We are extending the techniques of this study to the domain of publicly available software projects to experiment with classifying a structured language codebase. To run this experiment, we developed an automated pipeline for extracting, transforming, and analyzing structural features among sets of genre-specific codebases. This pipeline consists of a service to execute specific queries for repositories on GitHub, a translation layer for exporting acquired software projects into a convenient format, and a set of graph-specific metrics to be performed comprehensively for each project. We specifically chose to study how well we can predict the genre of a specified Java game project available on GitHub and assess the utility of this technique relative to others that do not take into account structural features.

Domain:

The selection of data that was utilized for this research project included various representative game genres available on Github, which include: Adventure, Puzzle, Educational, RPG, and RTS.



Project Module Flow :



Results & Conclusions:

Using Our Own 10-fold cross-validation with Naive Bayes
Correctly Classified Instances: 41 (**32.2835%**).
Incorrectly Classified Instances: 86 (67.7165%).
Total Number of Instances: 127 (down-weighting our results).

Using A Bag of Words 10-fold cross-validation with Naive Bayes
Correctly Classified Instances: 49 (**38.8889%**).
Incorrectly Classified Instances: 77 (61.1111%).
Total Number of Instances: 126.

Confusion Matrix (Ours)

a	b	c	d	e	<-- classified as
10	1	10	0	5	a = Adventure
9	2	13	0	3	b = Educational
5	2	14	2	5	c = Puzzle
7	1	11	2	4	d = RPG
1	2	4	1	13	e = RTS

Confusion Matrix (B.O.W)

a	b	c	d	e	<-- classified as
10	8	4	0	4	a = RPG
4	16	1	1	3	b = Adventure
9	2	8	1	1	c = RTS
3	12	1	5	6	d = Puzzles
4	7	3	3	10	e = Educational

Both of the accuracies listed above are *significantly better* than average accuracy when guessing blindly. It is important to note there was not a significant difference between the two Naive Bayes results. Future research will be needed to determine whether or not primarily considering graphs' structural features will deliver more effective results, but we can confirm that their inclusion is beneficial in many cases.

References:

Bompetsis, Nikolas. (2016). AST-Generator. https://github.com/ElasticThree/ast-generator.
Li, Geng & Semerci, Murat & Yener, Bulent & Zaki, Mohammed. (2012). Graph Classification via Topological and Label Attributes.
Kendall-Morwick, Joseph & Leake, David. (2014). Facilitating Representation and Retrieval of Structured Cases: Principles and Toolkit.
Messner, B.T. & Bunke, H. (2000). The Problem of Graph Matching.
Mitchell, Thomas. (1997). Machine Learning.