Manjari Maheshwari

16 December 2020

Classification using Support Vector Machines & Hidden Markov Models

This final project was an extension of the work completed for the midterm, which consisted of developing 3 Hidden Markov Models (HMMs) for 3 Malware Families, including Winwebsec, Zbot, and ZeroAccess. Using these three models, trained on one family each, I ran 6 experiments of each model against one other family. Some of the same parts of the project had to be redone for this project, including counting the number of occurrences of each opcode in each family. Based on the tuning parameter, M= 17, which was chosen in midterm 2, I converted the top 16 opcodes into ascii characters, and all remaining opcodes into "Q". Using this parameter, I translated 1,000 files from each family into these ascii characters. Then, I translated the ascii characters to numbers that could be used within HMM. These pre-processing steps can be better understood from the following table.

Int (#)	Symbol	Winwebsec Opcodes	Zbot Opcodes	ZeroAccess Opcodes
1	A	mov	mov	mov
2	B	push	add	xor
3	C	call	push	push
4	D	cmp	jmp	add
5	E	pop	nop	cmp
6	F	jz	call	jmp
7	G	jnz	pop	or
8	H	add	cmp	sub
9	I	test	retn	call

10	J	lea	xor	and
11	K	jmp	jz	jnz
12	L	retn	sub	lea
13	M	xor	inc	adc
14	N	and	lea	pop
15	O	inc	test	inc
16	P	sub	jnz	jz
17	Q	remaining	remaining	remaining

Beyond the tuning parameter of M=17, some other parameters to consider include the number of states per model, which I kept at N=2 since that is considered a good baseline for Hidden Markov Models. After creating the three trained HMM Models for midterm 2, I used the resulting pi, A, and B values to test 100 files from each family against the model. This resulted in 100 logProbs which were passed into a scatter chart and then scored based on various thresholds to calculate the True Positive Rate. The goal of checking all thresholds is to ensure the ROC curve is independent of location and based solely on the data. The end results of my HMM models can be seen in the following table, which suggests that the parameters N=2 and M=17 were very good for Webwinsec vs Zbot, and somewhat average for the others. I chose to keep these tuning parameters for this project to see how different kernels for an SVM model would boost the accuracy score.

Winwebsec Model		Zbot Model		ZeroAccess Model	
vs Zbot	vs ZeroAccess	vs Winwebsec	vs ZeroAccess	vs Winwebsec	vs Zbot
0.94	0.50	0.505	0.505	0.51	0.659999

For this project, we had to make at least 9 HMM models (3 for each family) to train Support Vector Machines (SVMs) with various tuning parameters. The number of HMM models used to train an SVM model is considered the number of dimensions used. I restricted mine to 3 dimensions. I kept the same N=2 and M=17. The original midterm 2 had a training observation length of 50,000 characters, and a maximum of 5,000 characters for testing. I increased the testing maximum to 50,000 characters since my computer could handle the computation. Another tuning parameter includes the kernels used to test the SVM model, whose results can be seen in the experiments section of this report.

# Dimensions per SVM model	# kernels & which ones	M (# Symbols)	N (# States)	T (# Observations)
3	(4) linear, polynomial, radial basis, sigmoid	17	2	50,000

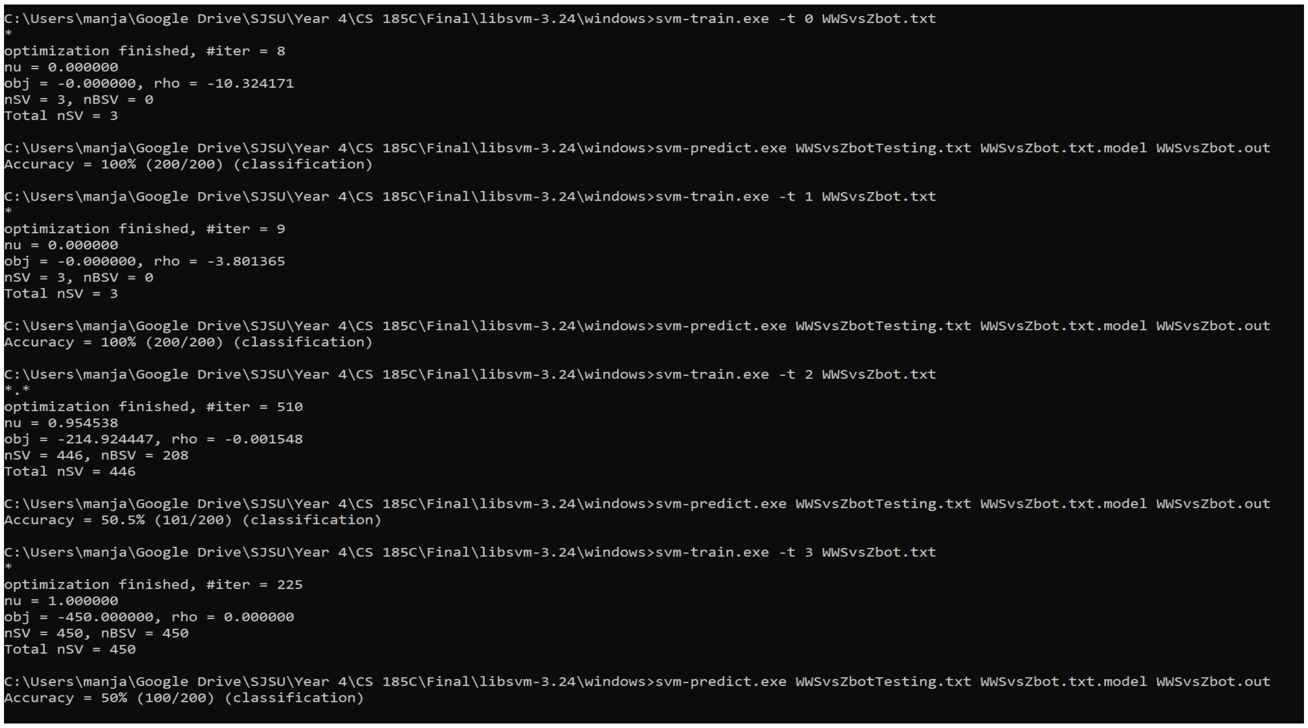
For this final project, these models had to be created in a slightly different way. Firstly, I made HMM models abstract objects that contained pi, A, B, and O as features within the class and randomized the creation of pi, A, and B to allow for much easier and faster computation. I also had to change how the HMM models were being trained. To build an SVM feature vector, I split the 1,000 files per family into 5 sets: 4 sets of 225 files, and 1 set of 100 files. The first 3 sets of 225 files were used to train 3 distinct HMM models to evaluate accurate pi, A, and B values.

Once I had trained the HMM models, I scored the last set of 225 files against each model and compiled all of the logProbs into a 2D-array made of 225 rows and 4 columns. The first column consisted of a label for the family, so if I was training an SVM model for a specific family, that family would get a +1 label, while the opposing family would get a -1 label. The next 3 columns were the scores for that file against each HMM Model, respectively. These feature vectors were compiled into a file to be used to train an SVM model, with the format: "-1 1:logProbFromModel1, 2:logProbFromModel2, 3:logProbFromModel3" for each row. Since the SVM library I used for this project, libSVM, build multi-class models, all 6 experiments were actually made on 6 models trained on two families. The experiments were conducted using feature vectors of the scores of the last set of 100 files and ran on Windows command line. The following table and pictures show the results of these experiments.

Kernel Used		WWS vs Zbot	WWS vs ZA	Zbot vs WWS	Zbot vs ZA	ZA vs WWS	ZA vs Zbot
Linear	# iter for Training	8	3	9	12	3	10
	Testing Accuracy	100%	98.5%	100%	100%	98.5%	100%
Polynomial	# iter for Training	9	3	7	10	3	8
	Testing Accuracy	100%	98.5%	100%	100%	98.5%	100%
Radial Basis	# iter for Training	510	655	513	567	654	600
	Testing Accuracy	50.5%	52%	50.5%	50.5%	52%	50.5%
Sigmoid	# iter for Training	225	225	225	225	225	225
	Testing Accuracy	50%	50%	50%	50%	50%	50%

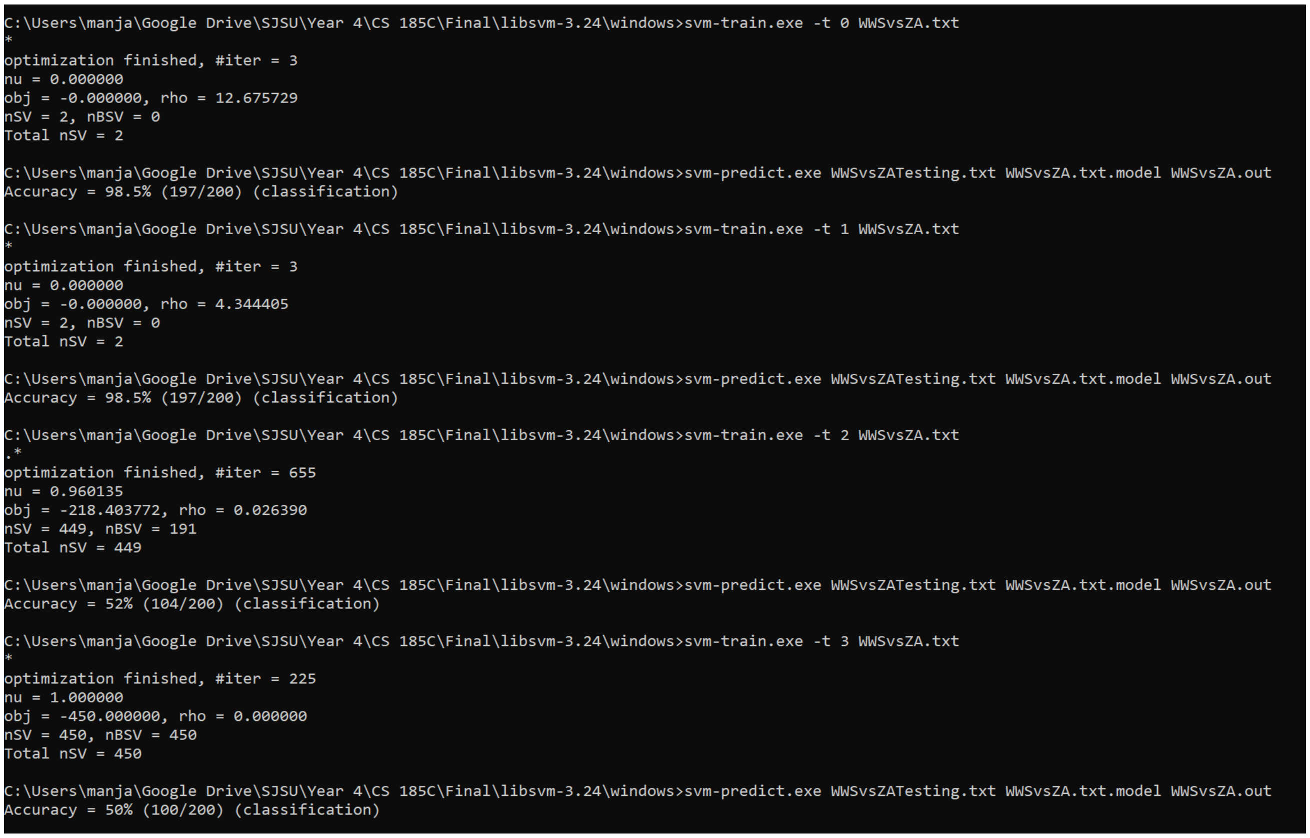
Here are the command lines used. The kernels are represented by -t, where 0 is linear, 1 is polynomial, 2 is Radial Basis, and 3 is Sigmoid.

Winwebsec vs Zbot:



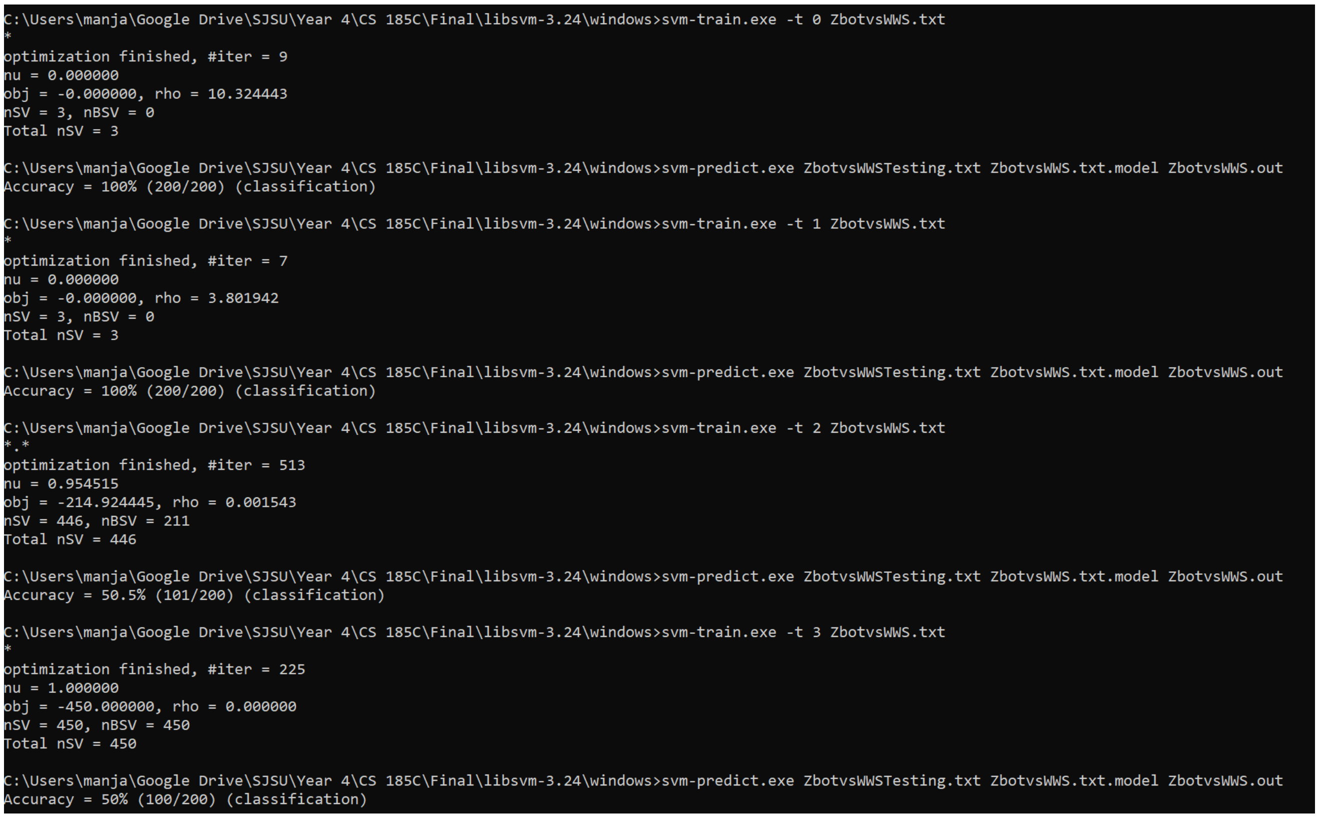
```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 WWSvsZbot.txt
*
optimization finished, #iter = 8
nu = 0.000000
obj = -0.000000, rho = -10.324171
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZbotTesting.txt WWSvsZbot.txt.model WWSvsZbot.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 WWSvsZbot.txt
*
optimization finished, #iter = 9
nu = 0.000000
obj = -0.000000, rho = -3.801365
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZbotTesting.txt WWSvsZbot.txt.model WWSvsZbot.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 WWSvsZbot.txt
*.*
optimization finished, #iter = 510
nu = 0.954538
obj = -214.924447, rho = -0.001548
nSV = 446, nBSV = 208
Total nSV = 446

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZbotTesting.txt WWSvsZbot.txt.model WWSvsZbot.out
Accuracy = 50.5% (101/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 WWSvsZbot.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZbotTesting.txt WWSvsZbot.txt.model WWSvsZbot.out
Accuracy = 50% (100/200) (classification)
```

Winwebsec vs. ZeroAccess



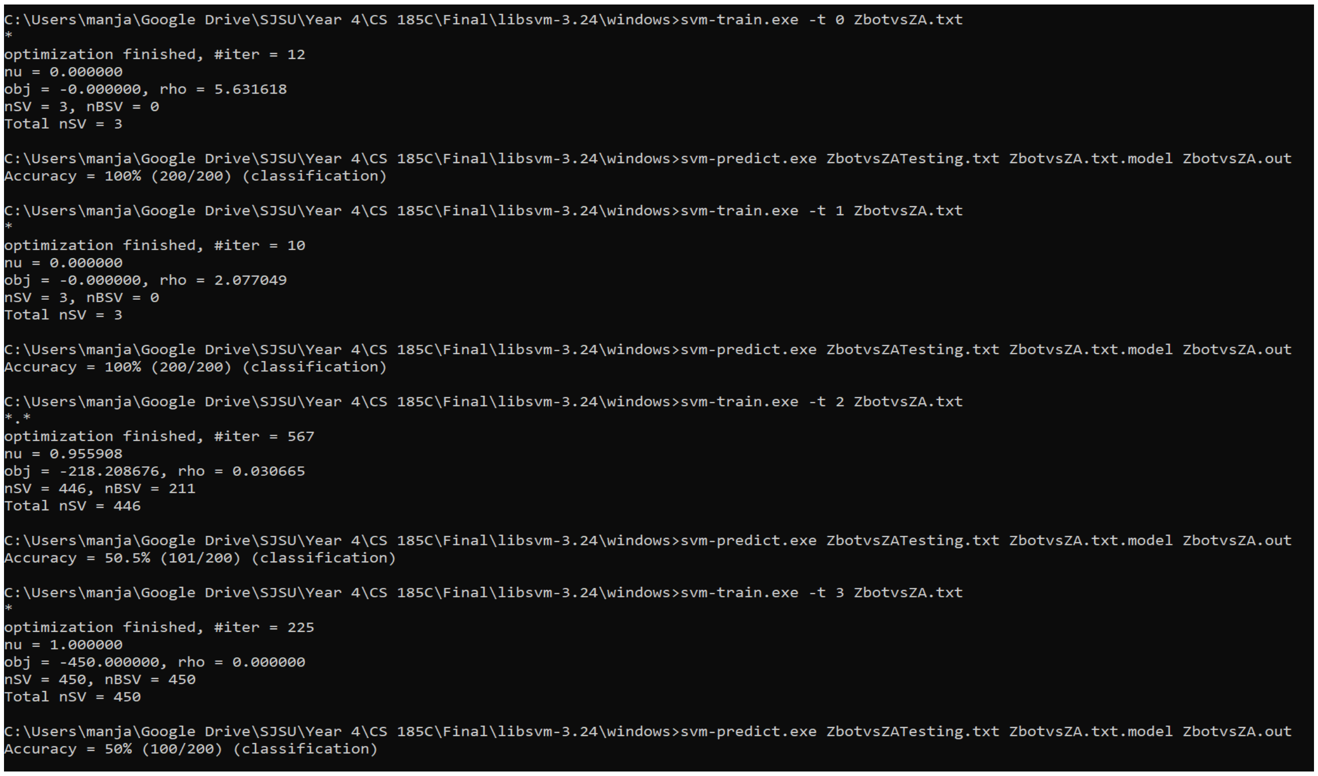
```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 WWSvsZA.txt
*
optimization finished, #iter = 3
nu = 0.000000
obj = -0.000000, rho = 12.675729
nSV = 2, nBSV = 0
Total nSV = 2

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZATesting.txt WWSvsZA.txt.model WWSvsZA.out
Accuracy = 98.5% (197/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 WWSvsZA.txt
*
optimization finished, #iter = 3
nu = 0.000000
obj = -0.000000, rho = 4.344405
nSV = 2, nBSV = 0
Total nSV = 2

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZATesting.txt WWSvsZA.txt.model WWSvsZA.out
Accuracy = 98.5% (197/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 WWSvsZA.txt
.*
optimization finished, #iter = 655
nu = 0.960135
obj = -218.403772, rho = 0.026390
nSV = 449, nBSV = 191
Total nSV = 449

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZATesting.txt WWSvsZA.txt.model WWSvsZA.out
Accuracy = 52% (104/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 WWSvsZA.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe WWSvsZATesting.txt WWSvsZA.txt.model WWSvsZA.out
Accuracy = 50% (100/200) (classification)
```

Zbot vs. Winwebsec:



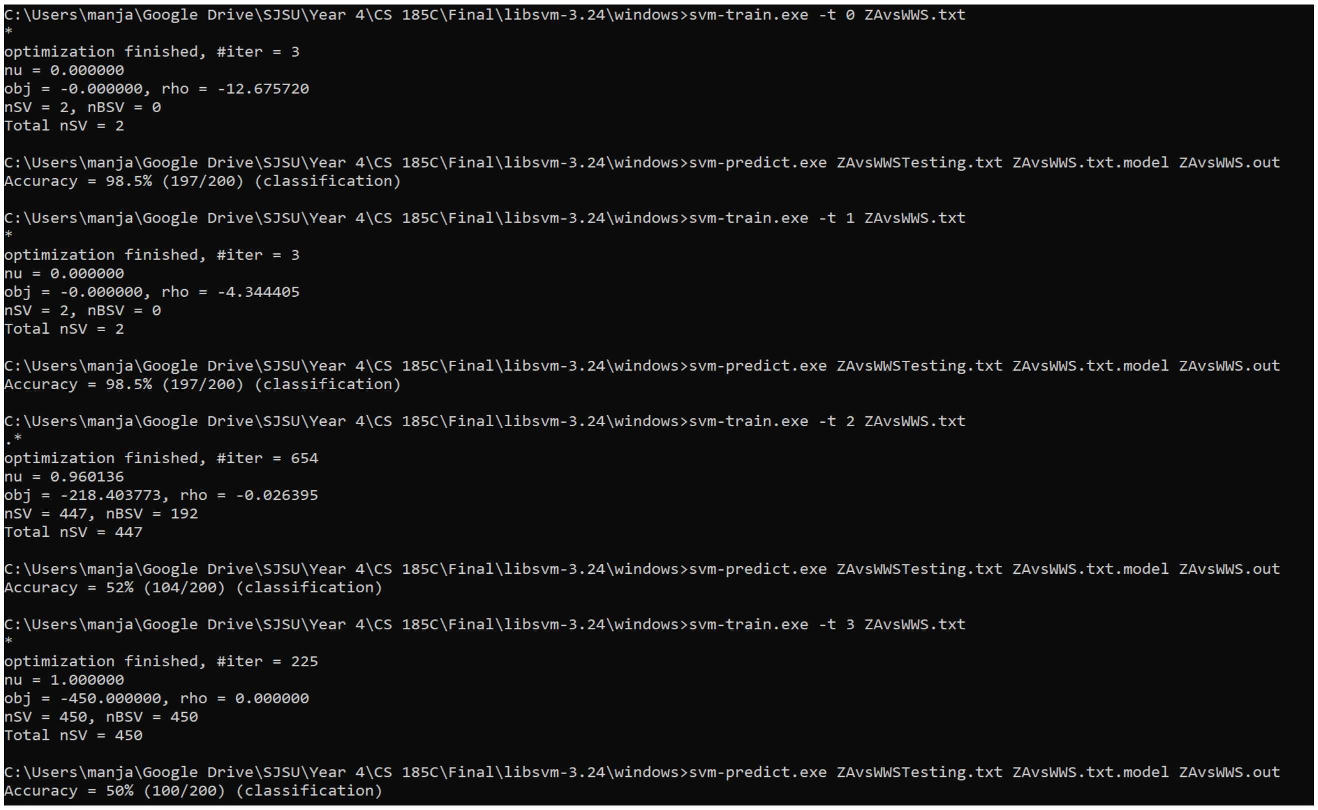
```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 ZbotvsWWS.txt
*
optimization finished, #iter = 9
nu = 0.000000
obj = -0.000000, rho = 10.324443
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsWWSTesting.txt ZbotvsWWS.txt.model ZbotvsWWS.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 ZbotvsWWS.txt
*
optimization finished, #iter = 7
nu = 0.000000
obj = -0.000000, rho = 3.801942
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsWWSTesting.txt ZbotvsWWS.txt.model ZbotvsWWS.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 ZbotvsWWS.txt
*.*
optimization finished, #iter = 513
nu = 0.954515
obj = -214.924445, rho = 0.001543
nSV = 446, nBSV = 211
Total nSV = 446

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsWWSTesting.txt ZbotvsWWS.txt.model ZbotvsWWS.out
Accuracy = 50.5% (101/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 ZbotvsWWS.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsWWSTesting.txt ZbotvsWWS.txt.model ZbotvsWWS.out
Accuracy = 50% (100/200) (classification)
```

Zbot vs. ZeroAccess:



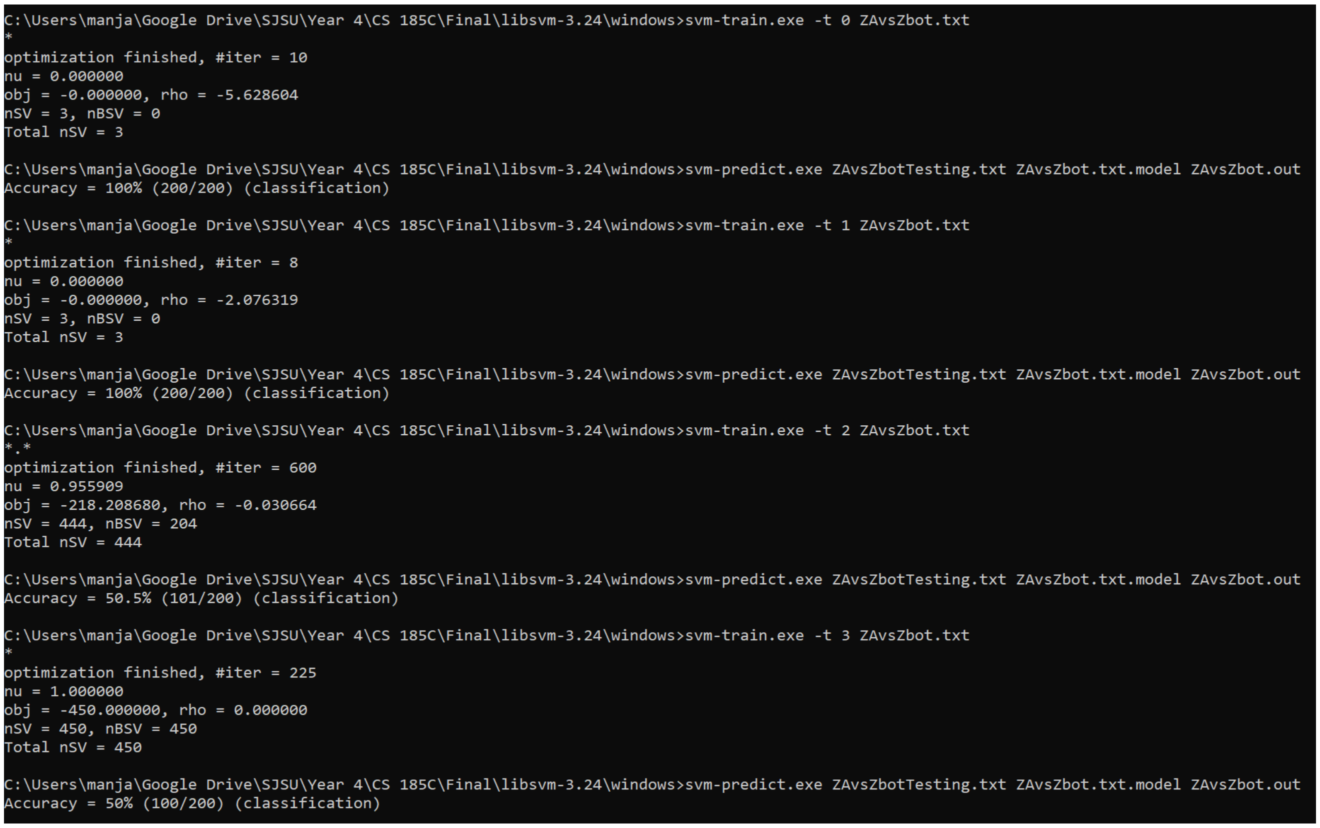
```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 ZbotvsZA.txt
*
optimization finished, #iter = 12
nu = 0.000000
obj = -0.000000, rho = 5.631618
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsZATesting.txt ZbotvsZA.txt.model ZbotvsZA.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 ZbotvsZA.txt
*
optimization finished, #iter = 10
nu = 0.000000
obj = -0.000000, rho = 2.077049
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsZATesting.txt ZbotvsZA.txt.model ZbotvsZA.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 ZbotvsZA.txt
*.*
optimization finished, #iter = 567
nu = 0.955908
obj = -218.208676, rho = 0.030665
nSV = 446, nBSV = 211
Total nSV = 446

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsZATesting.txt ZbotvsZA.txt.model ZbotvsZA.out
Accuracy = 50.5% (101/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 ZbotvsZA.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZbotvsZATesting.txt ZbotvsZA.txt.model ZbotvsZA.out
Accuracy = 50% (100/200) (classification)
```

ZeroAccess vs Winwebsec:



```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 ZAvsWWS.txt
*
optimization finished, #iter = 3
nu = 0.000000
obj = -0.000000, rho = -12.675720
nSV = 2, nBSV = 0
Total nSV = 2

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsWWSTesting.txt ZAvsWWS.txt.model ZAvsWWS.out
Accuracy = 98.5% (197/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 ZAvsWWS.txt
*
optimization finished, #iter = 3
nu = 0.000000
obj = -0.000000, rho = -4.344405
nSV = 2, nBSV = 0
Total nSV = 2

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsWWSTesting.txt ZAvsWWS.txt.model ZAvsWWS.out
Accuracy = 98.5% (197/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 ZAvsWWS.txt
.*
optimization finished, #iter = 654
nu = 0.960136
obj = -218.403773, rho = -0.026395
nSV = 447, nBSV = 192
Total nSV = 447

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsWWSTesting.txt ZAvsWWS.txt.model ZAvsWWS.out
Accuracy = 52% (104/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 ZAvsWWS.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsWWSTesting.txt ZAvsWWS.txt.model ZAvsWWS.out
Accuracy = 50% (100/200) (classification)
```

ZeroAccess vs. Zbot:



```
C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 0 ZAvsZbot.txt
*
optimization finished, #iter = 10
nu = 0.000000
obj = -0.000000, rho = -5.628604
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsZbotTesting.txt ZAvsZbot.txt.model ZAvsZbot.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 1 ZAvsZbot.txt
*
optimization finished, #iter = 8
nu = 0.000000
obj = -0.000000, rho = -2.076319
nSV = 3, nBSV = 0
Total nSV = 3

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsZbotTesting.txt ZAvsZbot.txt.model ZAvsZbot.out
Accuracy = 100% (200/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 2 ZAvsZbot.txt
*.*
optimization finished, #iter = 600
nu = 0.955909
obj = -218.208680, rho = -0.030664
nSV = 444, nBSV = 204
Total nSV = 444

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsZbotTesting.txt ZAvsZbot.txt.model ZAvsZbot.out
Accuracy = 50.5% (101/200) (classification)

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-train.exe -t 3 ZAvsZbot.txt
*
optimization finished, #iter = 225
nu = 1.000000
obj = -450.000000, rho = 0.000000
nSV = 450, nBSV = 450
Total nSV = 450

C:\Users\manja\Google Drive\SJSU\Year 4\CS 185C\Final\libsvm-3.24\windows>svm-predict.exe ZAvsZbotTesting.txt ZAvsZbot.txt.model ZAvsZbot.out
Accuracy = 50% (100/200) (classification)
```

The methodology for this project included similar preprocessing from midterm 2, with a more significant breakdown of files for training and testing due to the extra components. The original midterm only broke the family into 900 files for training and 100 for testing. For the final project, I broke the files into a string array with the first 3 indices filled with 50,000 characters from 225 files each to be used to train each HMM model, and then 325 filenames, for a total of 328 indices. The first 225 files out of the 325 were scored against each HMM model and then collected into an SVM feature vector for training. My SVM models were trained on two families, so a total of 500 feature vectors. Once the SVM models were correctly trained, I used the remaining 100 out of 325 files of two families to build a testing dataset of 200 feature vectors. My SVM model did really well with the linear and polynomial kernels, hitting either 98.5% and 100% each time. I do believe that this was partly due to the fact that my SVM models were trained on two families with a large data set. The best outcomes were WWS vs Zbot, Zbot vs WWS, Zbot vs ZA, and ZA vs Zbot at 100% for both linear and polynomial kernels. On the other hand, the sigmoid and radial basis kernels hit relatively low accuracies closer to 50%.

Overall, this project was such an interesting way to explicitly see the difference between SVM trained on HMMs versus only HMMs as I maintained most of the parameters from earlier. It was also really interesting to see the differences between various kernels within SVM models. As can be seen from the above data and tables, the SVM model was significantly better than individual HMM models at classifying the different malware families, especially on the linear and polynomial kernels. For future work, I'd like to try different dimensions, including up to 5 HMM models, for training the SVM models. I'd also like to use a different SVM library to build models trained on a single family rather than two at once, and attempt to build visual representations to better understand the classification process. I'd also like to learn more about boosting on top of stacking and try different HMM parameters since the original HMM models for Zbot and ZeroAccess did not score as well on N=2 and M=17, but did really well for SVM. I would also like to try increasing the number of characters used for HMM training to hopefully produce more accurate HMM scores for the feature vector. However, this is largely dependent on hardware capabilities, so maybe with further resources, larger observation strings can be used.