Predicting the Types of File Fragments

By

William Calhoun, Drue Coles

Presented At

The Digital Forensic Research Conference

DFRWS 2008 USA Baltimore, MD (Aug 11th - 13th)

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment. As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

http:/dfrws.org

Predicting the Types of File Fragments

William C. Calhoun and Drue Coles

Department of Mathematics, Computer Science and Statistics

Bloomsburg, University of Pennsylvania

Bloomsburg, PA 17815

Thanks to

- David Reichert (student) Bloomsburg U.
- Stephen Rhein (student) Bloomsburg U.
- Ron Bosch (statistician) Harvard S. of P.H.
- Reza Noubary (statistician) Bloomsburg U.
- David Bennet (statistician) Bloomsburg U.
- The DFRWS referees

The Problem

- To identify the types of fragments of computer files (without headers and footers).
- File type prediction software could be used to quickly find fragments for further investigation and to improve the performance of file-carvers, virus scanners, firewalls and search engines.

File Carvers

- The file header and footer can be used to identify the type of the file and to locate the first and last fragment.
- If the file was stored contiguously and was not overwritten, “file carving” programs can easily recover the whole file.
- File fragmentation can confuse file carvers.

File fragmentation can confuse file carvers

Our Goal

- To develop effective methods to identify the type of a file fragment automatically.
- The method must work even on file fragments from the middle of the file with no header or footer.

McDaniel and Heydari (2003)

- Used file “fingerprints”(byte frequency distribution), byte frequency correlation and file headers and trailers.
- The best results use file headers and trailers.

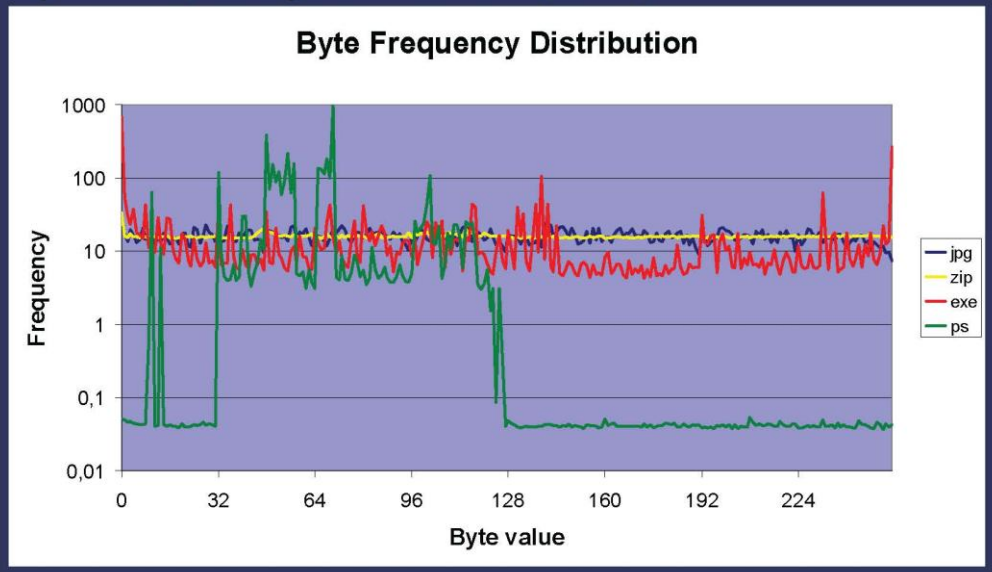
Li, Wang, Stolfo and Herzog (2005)

- Used “fileprints” -- the histograms of byte frequencies and variance.
- Centroid fileprints are found for each type of file. File types are predicted by finding the nearest centroid fileprint.
- The file header is included in their tests.

From *File Type Identification of File Fragments by Their Binary Structure* by Karresand and Shahmehri

Principle (Byte Frequency Distr.)

- Byte Frequency Distribution (BFD)



FOI

LINKÖPINGS UNIVERSITET

Karresand and Shahmehri (2006)

- The “Oscar” method – also uses centroid fileprints, but a different metric.
- Also uses the rate of change of byte values.
- Optimized for JPG files and uses special information about JPG file structure. Works very well in this context (99.2% detection with no false positives.)

Veenman(2007)

- Applies Fisher's linear discriminant to the fileprint, entropy and Kolmogorov complexity.
- Similar to our approach, but we used other statistics and the longest common subsequence algorithm.
- It is difficult to compare performance since the tests were not the same.

Our Approaches

1. Use Fisher's classification theory from statistics.
2. Use the longest common subsequence algorithm.

12,878 entries
Last updated: Tue Aug 5 2008

Created, developed, and nurtured by Eric Weisstein at Wolfram Research

Linear Discriminant Analysis

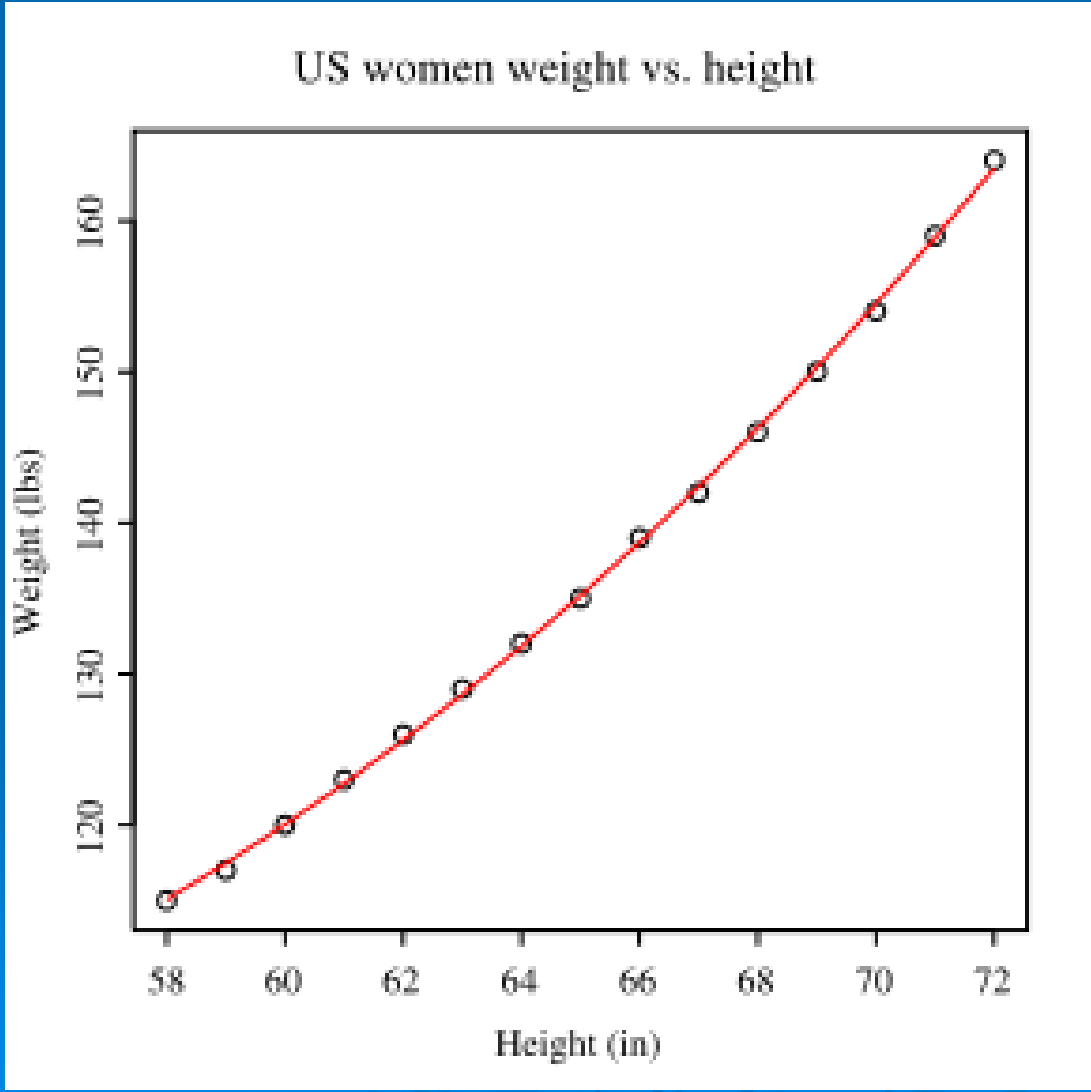
CONTRIBUTE This Entry

The Season 2 episode "Soft Target" (2006) of the television crime drama *NUMB3RS* features linear discriminant analysis.

CITE THIS AS:

Weisstein, Eric W. "Linear Discriminant Analysis." From *MathWorld*--A Wolfram Web Resource.
http://mathworld.wolfram.com/LinearDiscriminantAnalysis.html

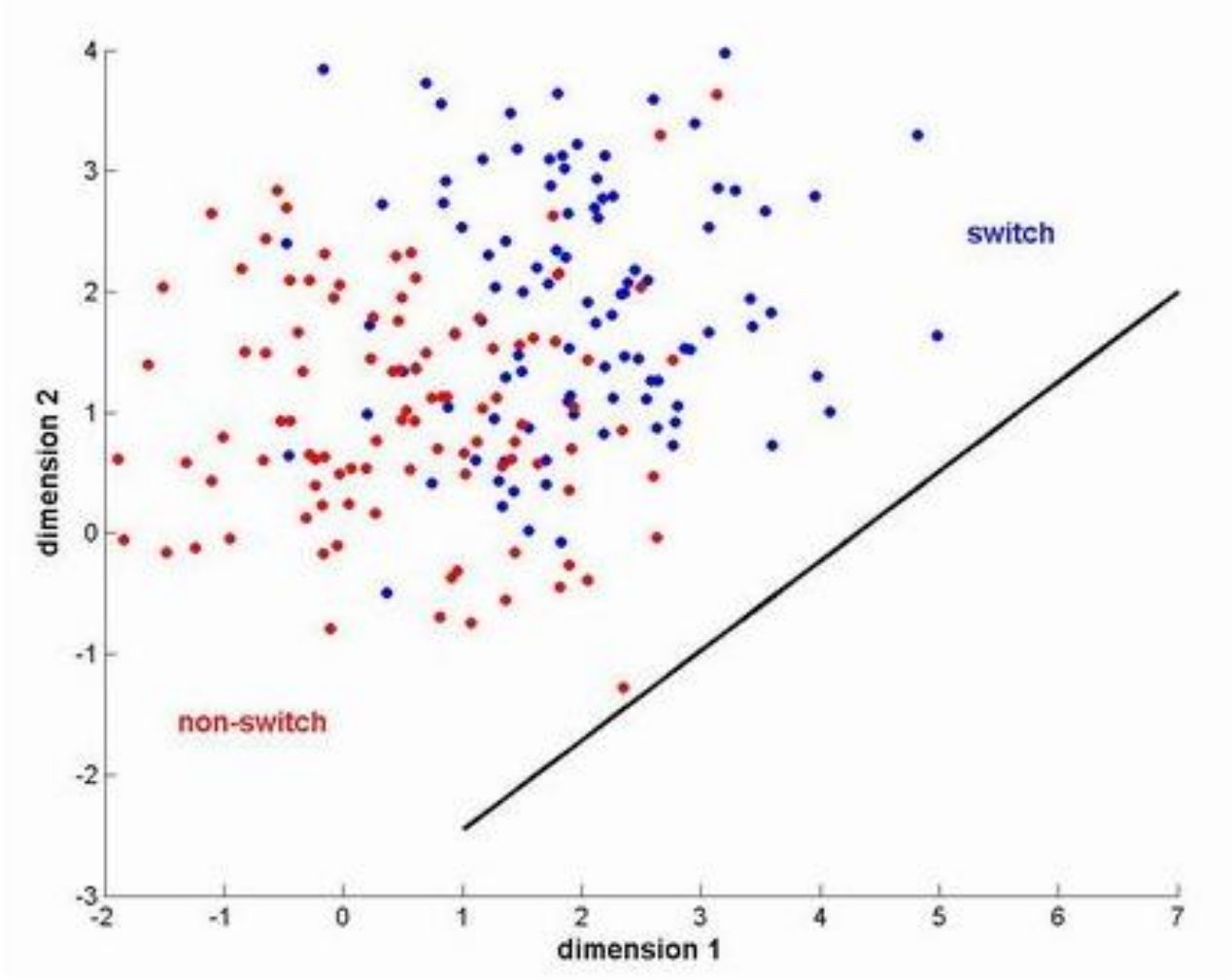
Regression: Finding the best curve to fit the data



Classification Theory

- Similar to regression. Provides a model that fits the data as well as possible.
- Classification theory becomes computationally infeasible with too many variables.
- Statistics we used: mean, modes, standard deviation, entropy, correlation between successive bytes and the frequencies of special codes such as ASCII codes.
- (We have also used a genetic algorithm to search for a good model.)

The linear discriminant projects the data on a line



Longest Common Substring

X = “DABCABCA”

Y = “ACABADAB”

Longest Common Substring

X = “DABCABCA”

Y = “ACABADAB”

Longest Common *Subsequence*

X = “DABCABCA”

Y = “ACABADAB”

Longest Common *Subsequence*

$X =$ "DABCABCA"

$Y =$ "ACABADAB"

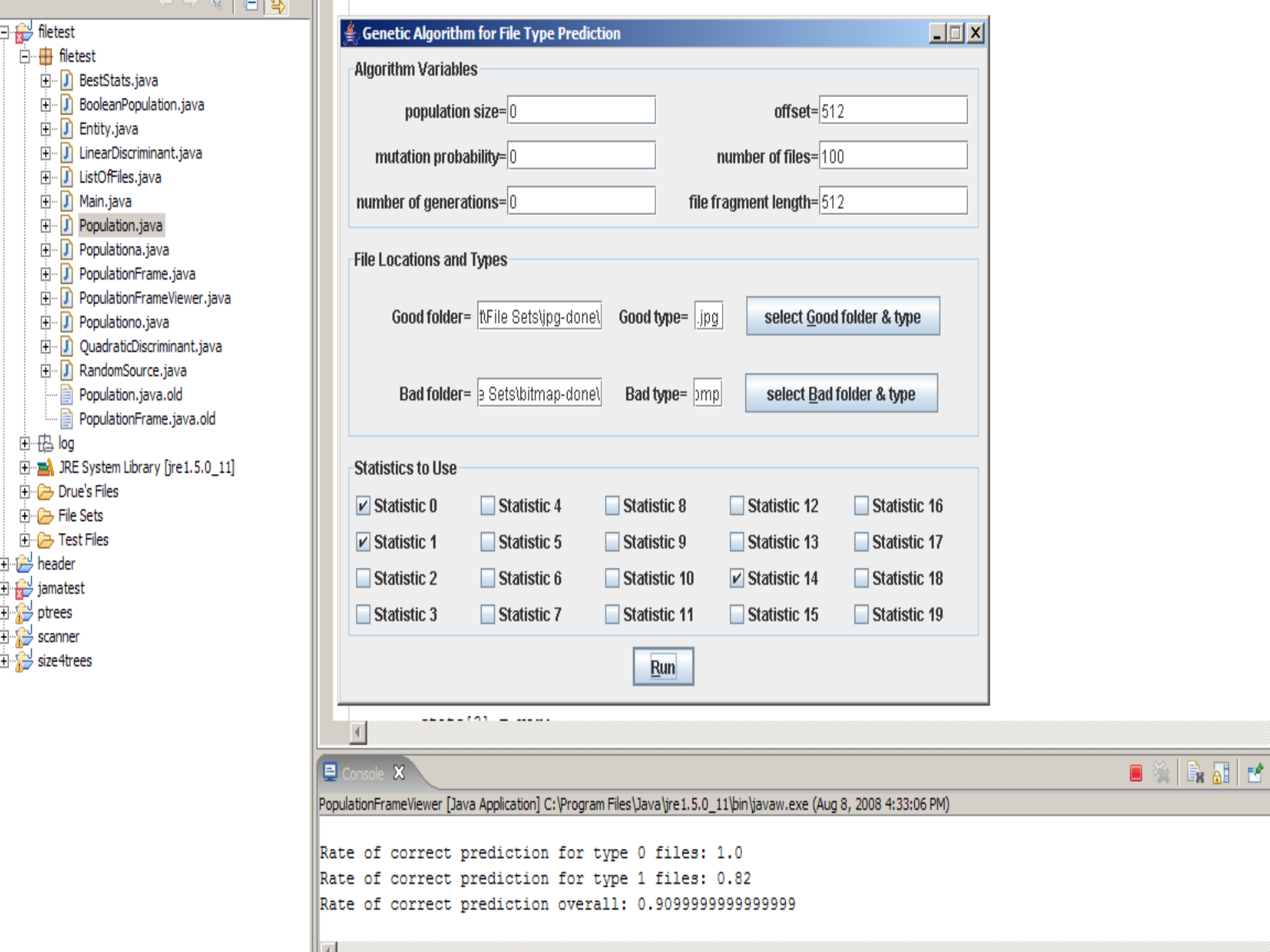
There is an obvious way to compute the LCS of two strings.

But the time-complexity is $O(n2^m)$ for two strings of length m and n. If the strings have 100 characters it would take millions of years to find the LCS.

Using dynamic programming the LCS can be computed in time roughly mn.

The Test

- Generate a model for predicting file type using 50 sample files of type 1 (good files) and 50 sample files of type 2 (bad files).
- Test the model on 50 good “mystery” files and 50 bad “mystery” files.
- See how often the predictions were correct.


filetest
filetest
BestStats.java
BooleanPopulation.java
Entity.java
LinearDiscriminant.java
ListOfFiles.java
Main.java
Population.java
Populationa.java
PopulationFrame.java
PopulationFrameViewer.java
Populationo.java
QuadraticDiscriminant.java
RandomSource.java
Population.java.old
PopulationFrame.java.old
log
JRE System Library [jre1.5.0_11]
Drue's Files
File Sets
Test Files
header
jamatest
ptrees
scanner
size4trees
Genetic Algorithm for File Type Prediction
Algorithm Variables
population size=0
offset=512
mutation probability=0
number of files=100
number of generations=0
file fragment length=512
File Locations and Types
Good folder=t\File Sets\jpg-done\
Good type=.jpg
select Good folder & type
Bad folder=e Sets\bitmap-done\
Bad type=pmp
select Bad folder & type
Statistics to Use
Statistic 0
Statistic 1
Statistic 2
Statistic 3
Statistic 4
Statistic 5
Statistic 6
Statistic 7
Statistic 8
Statistic 9
Statistic 10
Statistic 11
Statistic 12
Statistic 13
Statistic 14
Statistic 15
Statistic 16
Statistic 17
Statistic 18
Statistic 19
Run
Console
PopulationFrameViewer [Java Application] C:\Program Files\Java\jre1.5.0_11\bin\javaw.exe (Aug 8, 2008 4:33:06 PM)
Rate of correct prediction for type 0 files: 1.0
Rate of correct prediction for type 1 files: 0.82
Rate of correct prediction overall: 0.9099999999999999

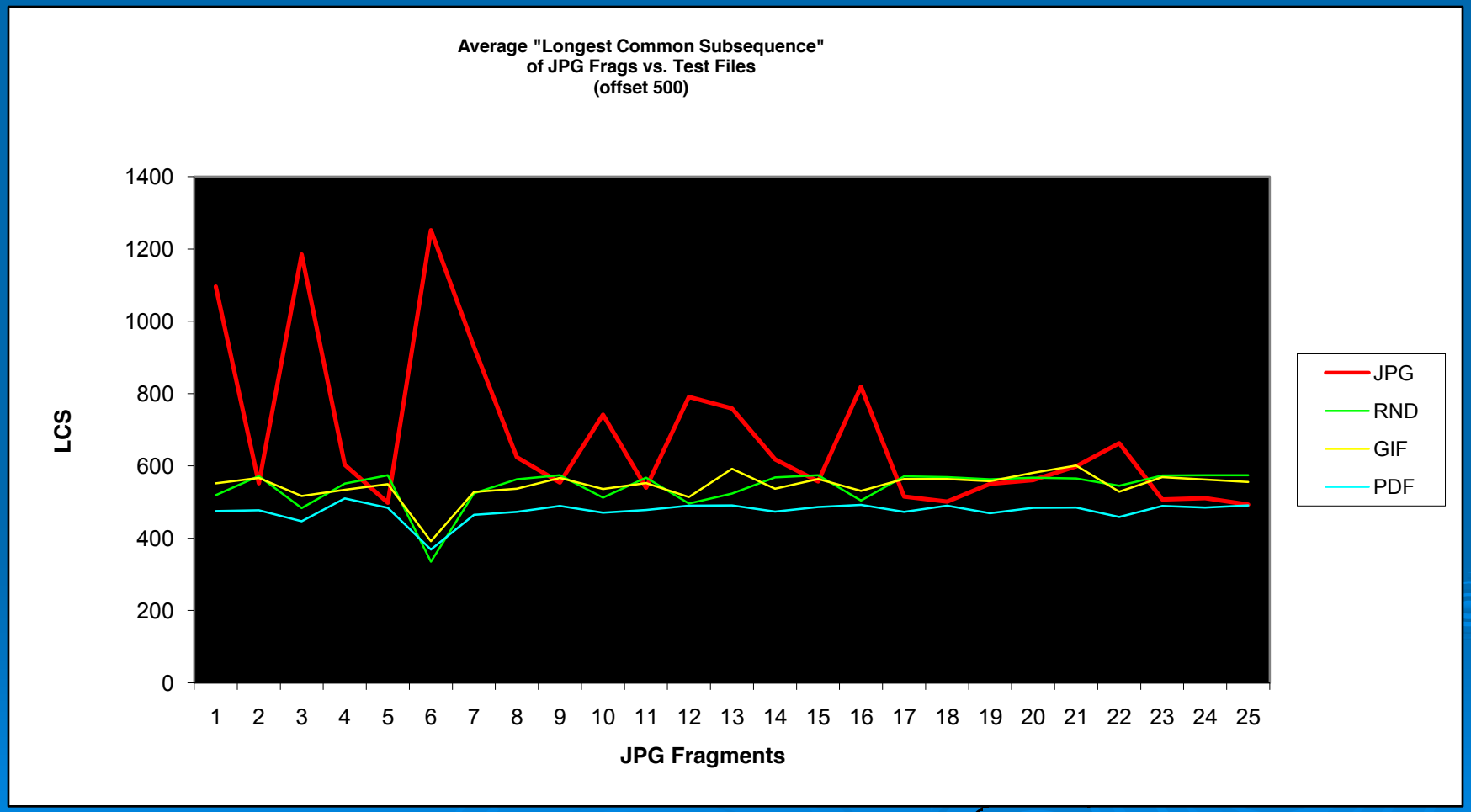
Some File Types We Used

- Jpeg - Graphics file. Fourier-type transform applied, low amplitude waves removed (lossy compression), compressed.
- Gif - Graphics file. Lossless compression.
- Bitmap – Graphics file. No compression
- Pdf – Code for text and graphics. May be compressed
- Excel - Spreadsheet data. Text and numbers.
- Random - random sequence of bytes
- Executable - Machine code for a program.

The best statistics in our experiments

- A-E-L-H-MF-SF: Ascii, Entropy, Low, High, ModesFreq (Sum of the top 4 frequencies), SdFreq (Sd of frequencies).
- LongerSSeqA: The file type is predicted to be type A if the average length of the longest common subsequence with type A sample files is longer than with type B sample files.
- (LongerSSeqA is accurate, but slower then the linear discriminant.)

Most JPG fragments have longer Average LCS with JPGs



How well did it work? (header lost) offset=128, fragment length=896

Types	Rate correct A-E-L-H-MF-SF	Rate Correct LongerSSeqA
JPG v. PDF	.98	.94
JPG v. GIF	.93	.87
JPG v. BMP	.85	.74
PDF v. GIF	.89	.92
PDF v. BMP	.82	.79
GIF v. BMP	.83	.81

How well did it work? (sector lost) Offset=512 fragment length=512

Types	Rate correct A-E-L-H-MF-SF	Rate correct LongerSseqA
JPG v. PDF	.80	.91
JPG v. GIF	.79	.85
JPG v. BMP	.87	.75
PDF v. GIF	.84	.90
PDF v. BMP	.90	.92
GIF v. BMP	.85	.83

Other Experiments

- Used longest substring common to all files of a given type (with David Reichert). Works well for finding headers but not well for fragments without headers.
- The Longest Common Subsequence method appears to work well in tests with 4 file types and with randomly corrupted files

To do list

- Test the linear discriminant on multiple file types.
- Use frequencies of two-bytes (or longer)
- Use a genetic algorithm or Analysis of Variance to determine the best combination of statistics to use.
- Try the quadratic discriminant.
- Use JAMA for matrix inversion.

HEX: The Collegiate Journal of Computer Forensics

- Call for Papers
- A new peer-reviewed on-line journal for undergraduate computer forensics students and their instructors. The first issue will appear in Fall 2008.
- Submissions in the form of Word or .pdf files may be sent by e-mail to me:
- wcalhoun@bloomu.edu (use HEX in the subject line.)

THANK YOU!

- Questions?