SCEADAN

Systematic Classification Engine for Advanced Data ANalysis

Nicole Beebe, Ph.D.
The University of Texas at San Antonio
Dept. of Info. Systems & Cyber Security

Simson Garfinkel, Ph.D.
The Naval Postgraduate School
Dept. of Computer Science

Caveats

- UTSA funding sources
  - NPS Grant No. N00244-11-1-0011
  - NPS Grant No. N00244-13-1-0027
  - UTSA Provost's Summer Research Mentorship Program

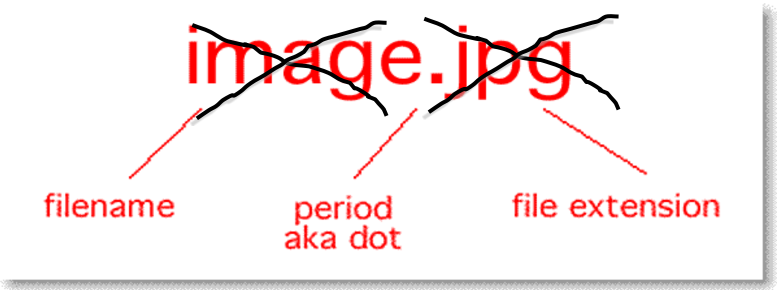
- Disclaimer
  - Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the Naval Postgraduate School

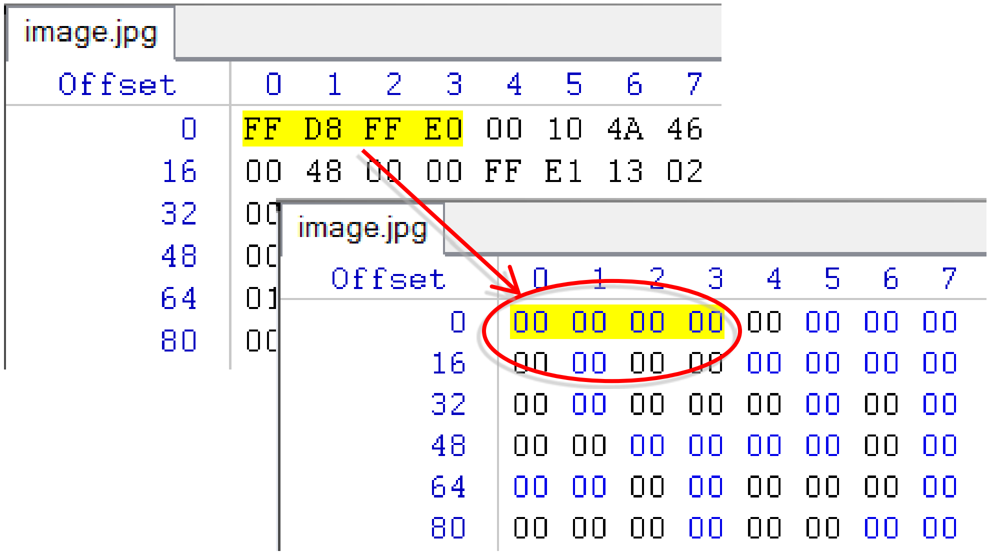
Motivation, Use Cases, Literature Review

BACKGROUND

Intro: Data/File Type Determination

- Challenging on fragments (non-header) and files without accurate signatures/extensions
  - Poor prediction accuracies in large multi-class situations
  - No useful tools available after 10 years of research by field
  - Significant scalability and speed issues





Primary Use Cases

- Digital forensics
  - Focus investigative efforts (e.g. search hit ranking feature)
  - Triage and disk profiling
  - Fragment identification/isolation, file recovery/reassembly
- Intrusion detection
  - Overcome signature-based obfuscation techniques
  - Anomaly detection
    - Context triggered indication & warning (e.g. payload vs. protocol), as opposed to traffic-based, "bursty" anomaly detection
  - Exfiltration, extrusion detection and profiling

Other Possible Use Cases

- Firewalls
  - Content based blocking
- Malware
  - Detection: Content based malware, virus detection
  - Analysis: Mapping binary objects
- Steganalysis
  - Detecting statistically abnormal file types due to content
  - Isolating “stegged” data within binary objects

- CAVEAT: Use traditional methods when you have trusted file signatures!

Literature Review

- Penrose et al. (2013)
- Patel and Singh (2013)
- Alherbawi et al. (2013)
- Roussev and Quates (2013)
- Xie et al. (2013)
- Fitzgerald et al. (2012)
- Gopal et al. (2011)
- Axelsson (2010)
- Conti et al. (2010)
- Li et al. (2010, 2005)
- Ahmed et al. (2010, 2009)
- Cao et al. (2010)
- Amirani et al. (2012, 2011, 2008)
- Calhoun and Coles (2008)
- Moody and Erbacher (2008)
- Zhang and White (2007)
- Veenman (2007)
- Erbacher and Mulholland (2007)
- Karresand and Shahmehri (2006)
- Hall and Davis (2006)
- McDaniel and Heydari (2003)
- Shannon (2004)

Two Fundamental Approaches

- Naïve statistical classification
  - Machine learning or metrics based approaches
    - Byte frequency distribution (n-gram analysis)
    - Complexity measures (entropy, Kolmogrov complexity, etc.)

- Specialized, semantic based approaches
  - Utilize knowledge of internal file structures

    Ex: JPEG sections; ZIP "local file headers"
  - Look for predictive, string based indicators

    Ex: `>>stream` preceding a Deflate compressed stream = .PDF
  - Not signature based in traditional sense

Byte Frequency Distributions

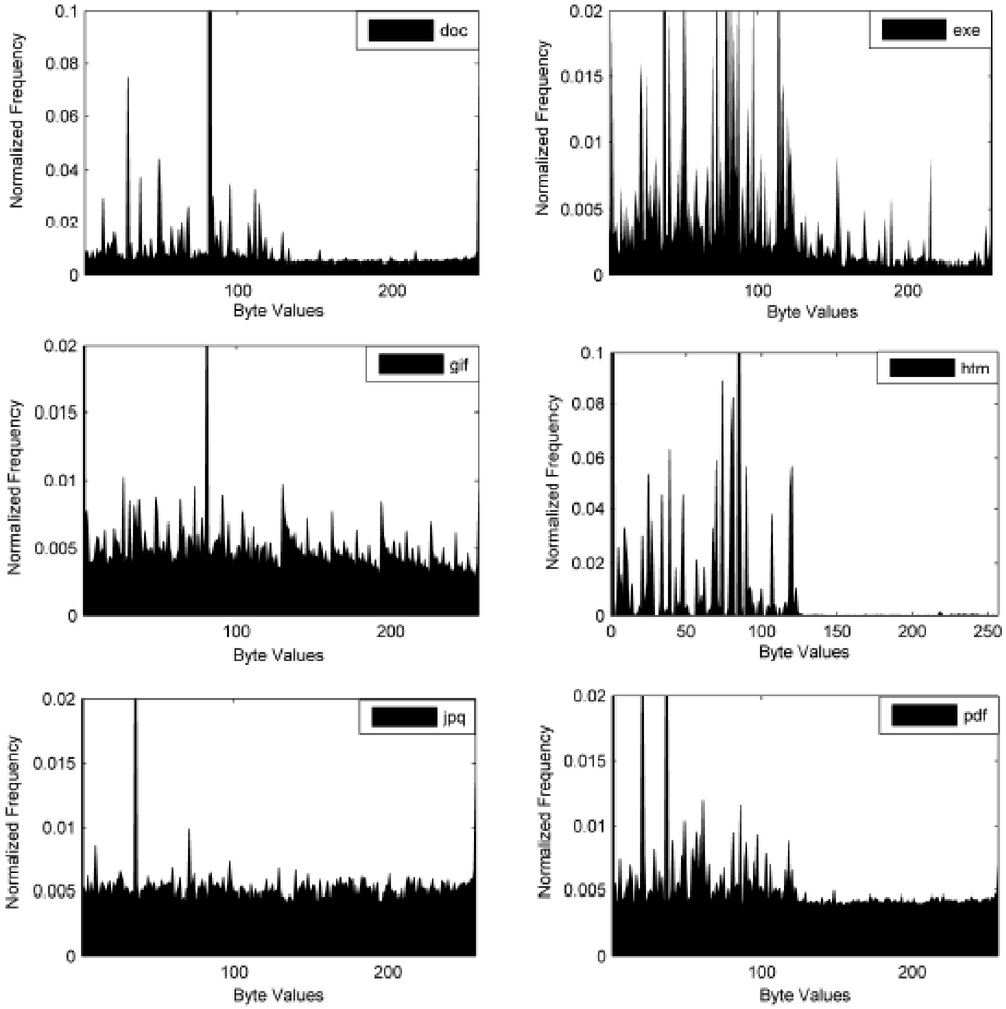


Figure 3. Byte frequency distribution of different file types.

n-gram Analysis:

Unigram = 1 byte

256 “features”

\x00, \x01 … \xFF

Bi-gram = 2 bytes

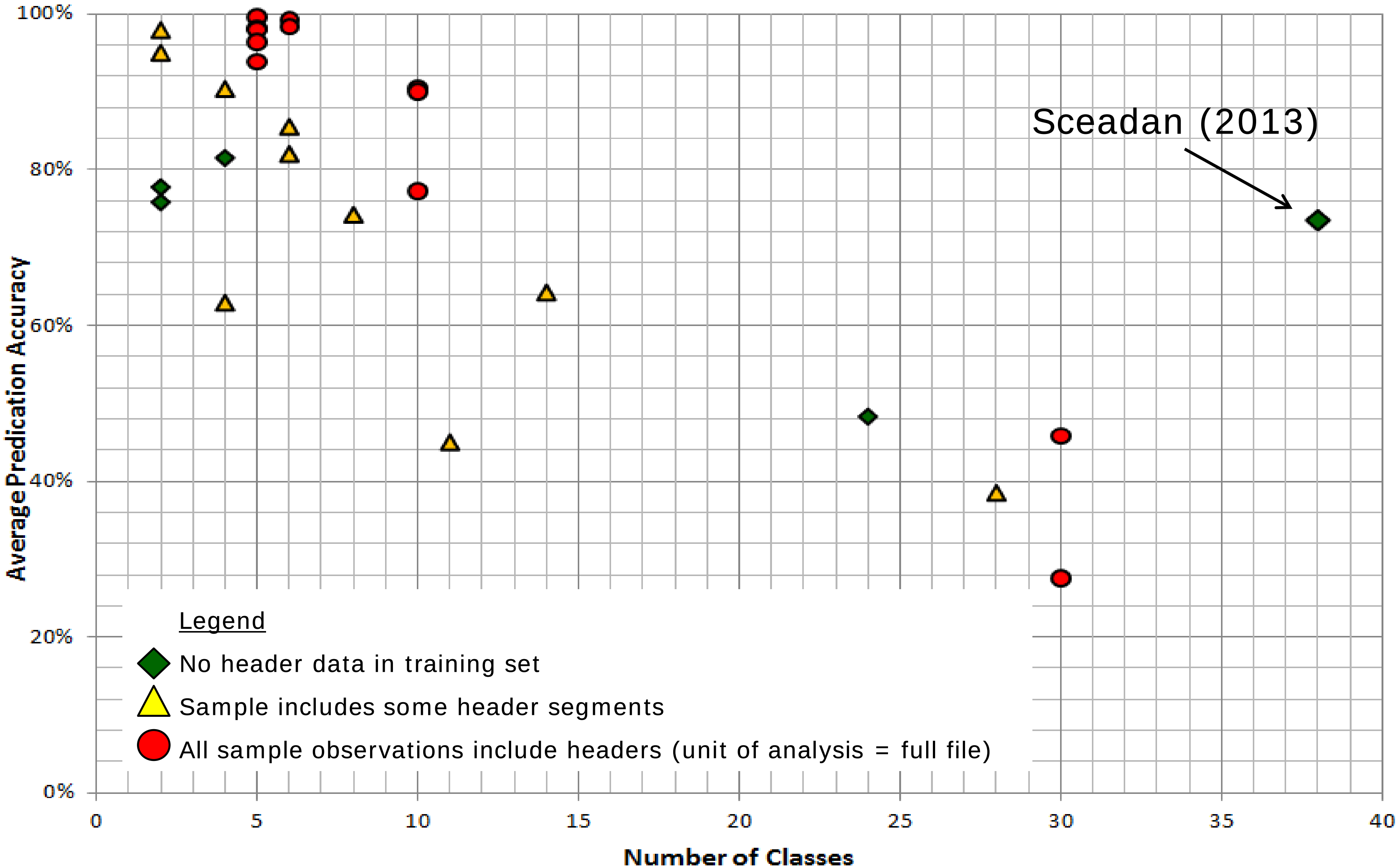
65,536 “features”

\x0000, \x0001, … \xFFFF

A few non-ngram features
(e.g. entropy, kurtosis, …)

Images from Amirani et al. (2012)

Including Headers Biases Results



Sceadan (model/tool) does *not* rely on header segments (but *can* use them)

Tool Developed

SCEADAN

Tool Developed: Sceadan

- The name
  - **S**ystematic **C**lassification **E**ngine for **A**dvanced **D**ata **AN**alysis
  - Old English / Proto-Germanic for "To Classify"

- Naïve statistical classifier
  - Classifies independent of signatures, extension, file system data
  - Uses N-gram features (concatenated unigram+bigram vectors)
  - Built-in training and prediction modes
  - Leverages `LIBLINEAR` (http://www.csie.ntu.edu.tw/~cjlin/liblinear/)
  - Uses support vector machines (SVMs)
  - Built-in model (≈50MB), but you can train/use your own

True Positive Prediction Rates in our Experiments

Type	Ext	Rate %
Delimited	.csv	100
JSON records	.json	100
Base64 encoding	.b64	100
Base85 encoding	.a85	100
Hex encoding	.urlenc	100
Postscript	.ps	100
Log files	.log	99
CSS	.css	99
Plain text	.text, .txt	98
XML	.xml	98
FS-EXT	.ext3	97
Java Source Code	.java	97
JavaScript code	.js	95
Bi-tonal images	.tif, .tiff	95
HTML	.html	91
GIF	.gif	86
MS-XLS	.xls	84
MP3	.mp3	84
Bitmap	.bmp	83
AVI	.avi	78
JPG	.jpg	76
BZ2	.bz2	72
H264	.mp4	72
FS-NTFS	.ntfs	71

Type	Ext	Rate %
AAC	.m4a	69
MS-DOCX	.docx	62
WMV	.wmv	59
PDF	.pdf	54
MS-DOC	.doc	53
MS-XLSX	.xlsx	50
FLV	.flv, .FLV	44
ZLIB – DEFLATE	.gz	29
Portable Network Graphic	.png	28
FS-FAT	.fat	25
MS-PPTX	.pptx	21
ZLIB - DEFLATE	.zip	20
MS-PPT	.ppt	14
ENCRYPTED	N/A	13

Average Sceadan prediction accuracy:
71.5%*

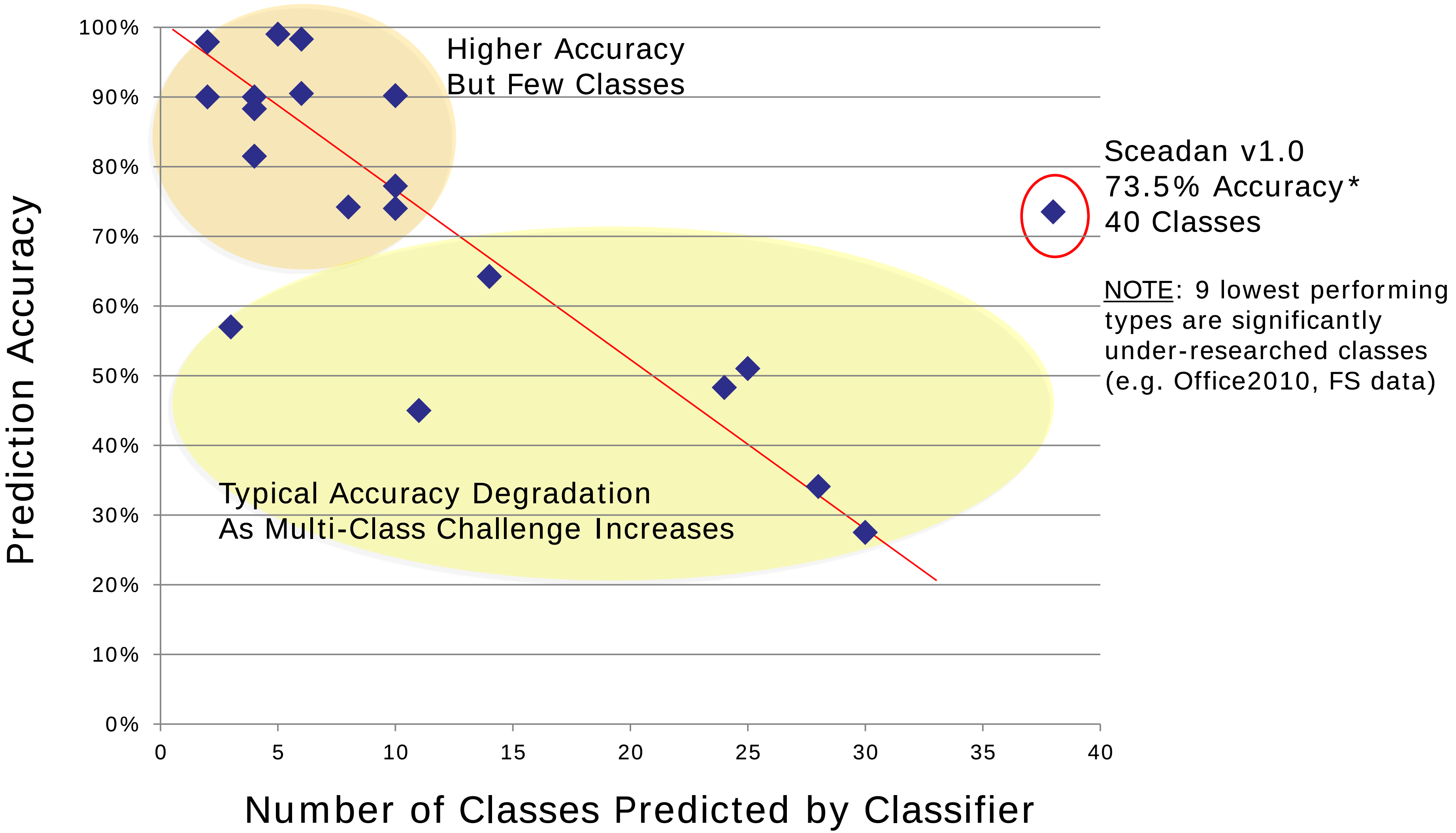
(NOTE: Later modeling netted 73.5% accuracy)

Random chance classification:
1/40 = 2.5%

Train/Test: 60%:40% (million fragment sample)

* Beebe et al. (2013) "Sceadan: Using Concatenated N-Gram Vectors for Improved Data/File Type Classification," *IEEE Transactions on Information Forensics and Security*, (8:9), pp. 1519-1530.

Sceadan Performance Relative to Others (2013)



*Additional model training has improved classifier accuracy from 71.5% to 73.5%

Built-In Model Details (Sceadan v1.2.1)

- Model file
  - model.ucv-bcv.20130509.c256.s2.e005
  - MD5 D068034D64329ECCA25DD76F515656A7
- Model trained using digitalcorpora.org files
  - Random subset of GOVDOCS1 files (types: see Beebe et al. 2013)
  - FILETYPES1 data set (UTSA open source file collection)
- 512B training samples (n=1,800 samples of each type)
  - Segmented all files into 512 byte blocks
  - Removed header sector
- Model parameters & solver function
  - C=256, e=.005, gamma=N/A (linear kernel)
  - L2 regularized L2 loss function, primal solver

Model Building (Training)

- Motivation
  - Train on different data types
  - High number of classes degrades model performance
  - Experiment with different block sizes, features, etc.
- Training data must be verified, prepared, cleansed
- Sceadan optimizes model parameters
  - Runs `grid.py`
  - Optimizes C (surface smoothing parameter)
  - Optimizes gamma (single training point influence)
- Sceadan randomly splits sample into train:test sets

Other Capabilities

- Ability to write LibSVM compliant vectors to output
- Multi-threaded, in-memory/compiled model
- Creates confusion matrices automatically
- Sub-block classification capability
  - Can specify sub-block size within files to classify
- Can classify individual files, or in directory mode:

```
student@ubuntu:~/sceadan/src$ ./sceadan_app ./sceadan_test_slg
0           ELF # ./sceadan_test_slg/elf.txt
0           FLV # ./sceadan_test_slg/flv.txt
0           WMV # ./sceadan_test_slg/wmv.txt
```

Predicted Type

Input path/filename
These samples were named
<<true type>>.txt

Block offset
(0=full file classified)

Miscellaneous

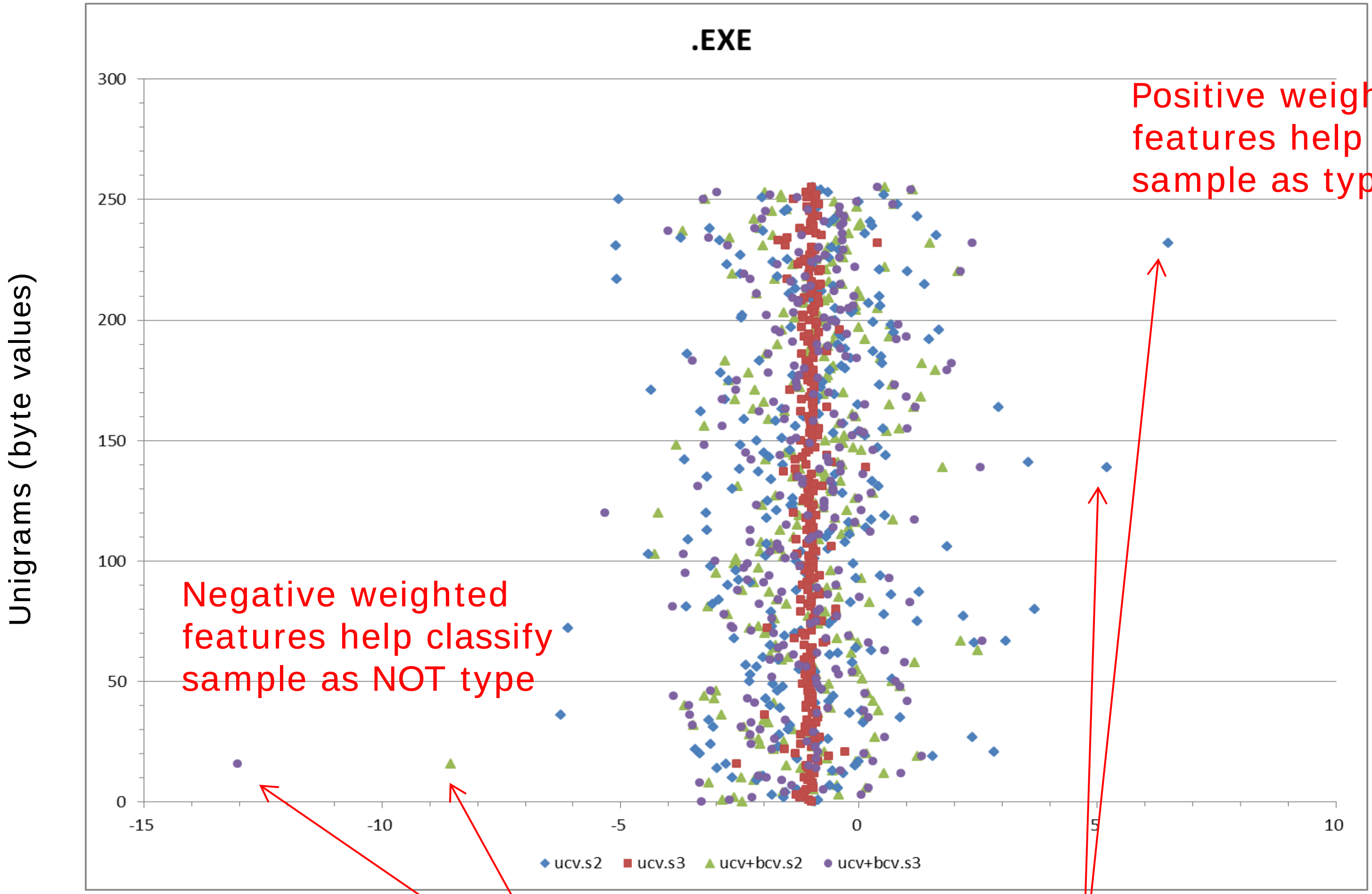
- Copyright: University of Texas at San Antonio
- License: GPLv2
- Written in C
- Linux CLI based
- Code rewritten/improved in 2014 by NPS
- To obtain: github.com/nbeebe/sceadan

Future Code Development Plans

LATEST RESEARCH

UTSA THE UNIVERSITY OF TEXAS AT SAN ANTONIO

Scatter Plot of Unigram Weights (4 Models) for .EXE Files



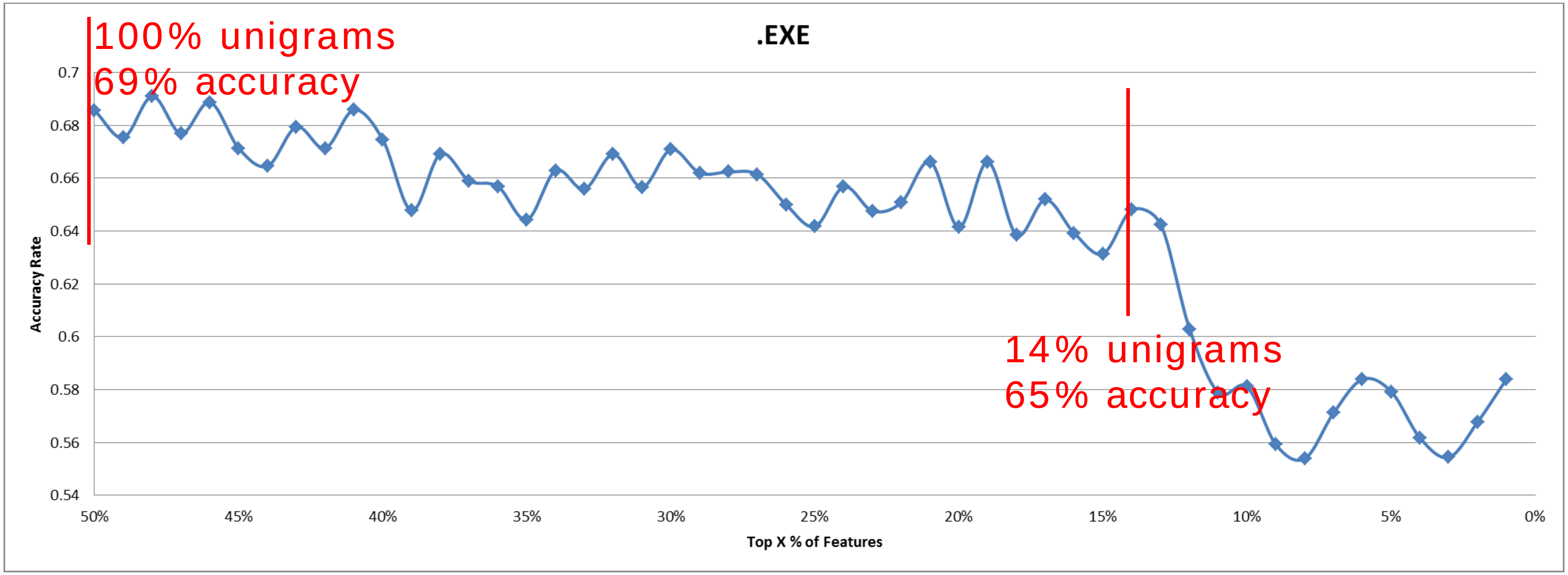
Positive weighted features help classify sample as type

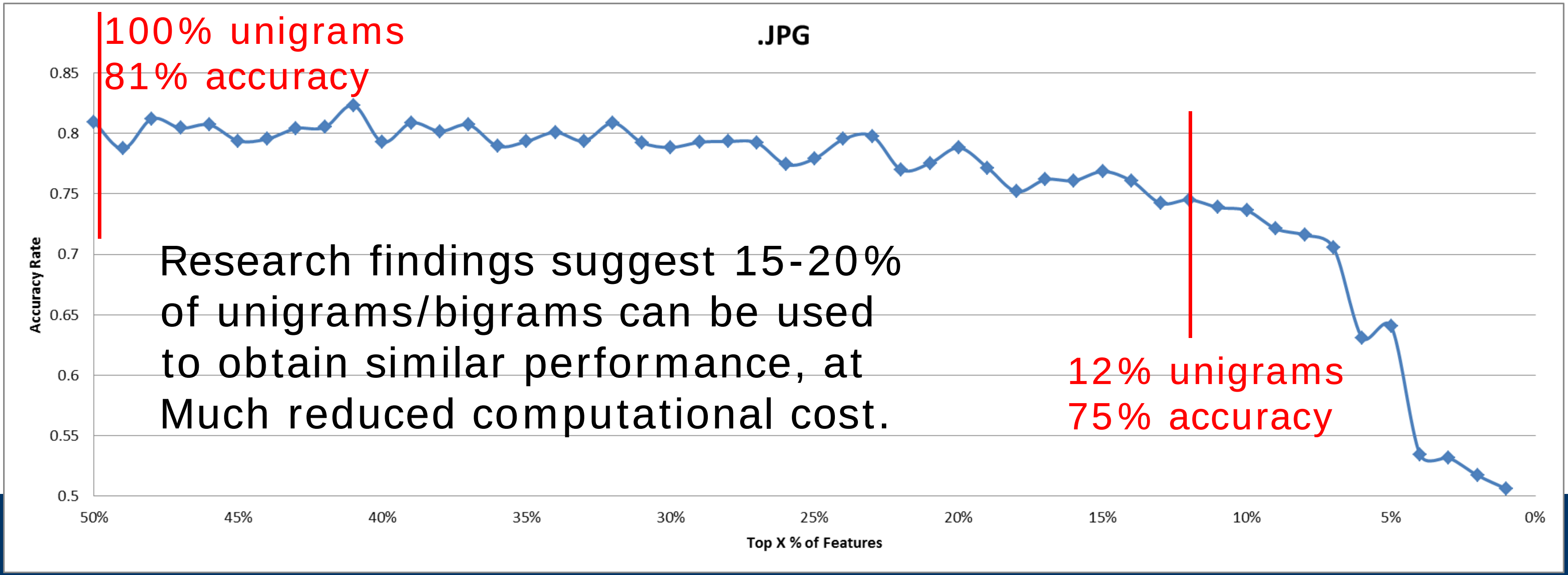
Negative weighted features help classify sample as NOT type

High absolute value features are good discriminators

Not all n-grams are equally discriminatory

Prediction Accuracy as % Unigrams Decline

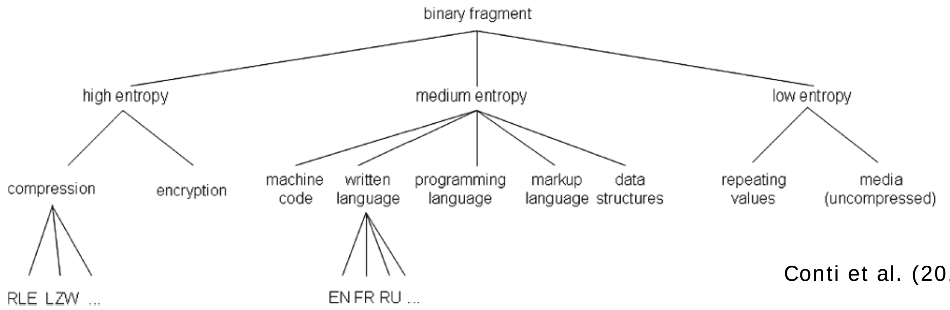




Research findings suggest 15-20% of unigrams/bigrams can be used to obtain similar performance, at Much reduced computational cost.

Hierarchical Classification Modeling

- Advanced, hierarchical classification design
  - Improve accuracy via hierarchical data *class* classification, followed by data *type* classification



Conti et al. (2010)

  - In practice, *class* classification
    - Becomes triage mechanism to focus classification efforts
    - Improves prediction accuracy
      - Reduces multi-class size in tough classes
      - Enables better feature selection within classes
      - Reduces problem of over-fitting

UTSA THE UNIVERSITY OF TEXAS AT SAN ANTONIO

Nicole.Beebe@utsa.edu

(210) 458-8040

(210) 269-5647 (Cell)

COMMENTS / QUESTIONS ?