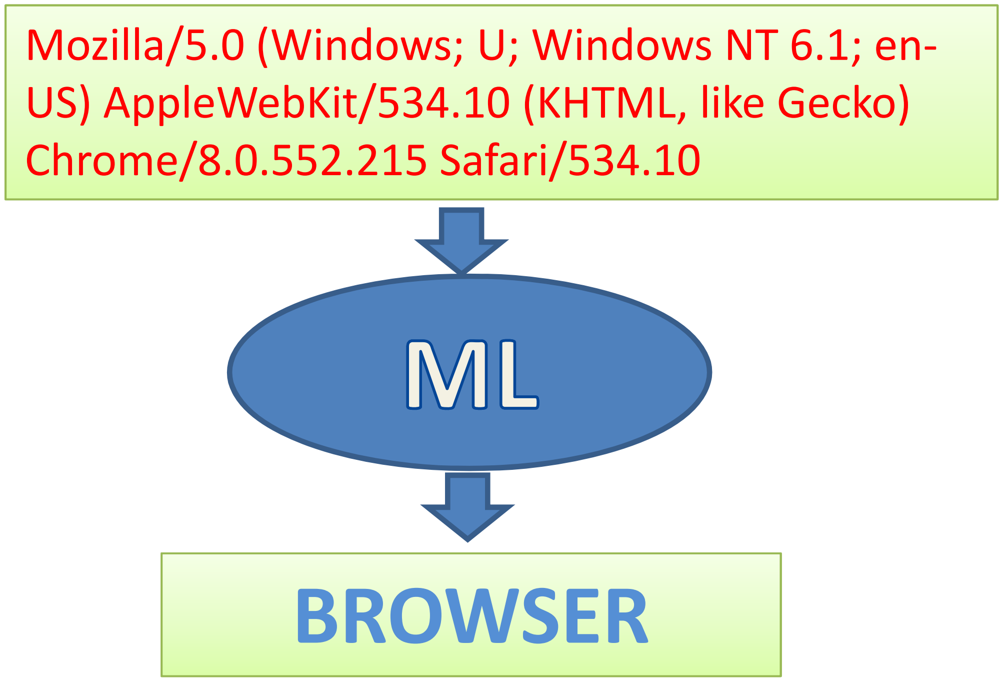
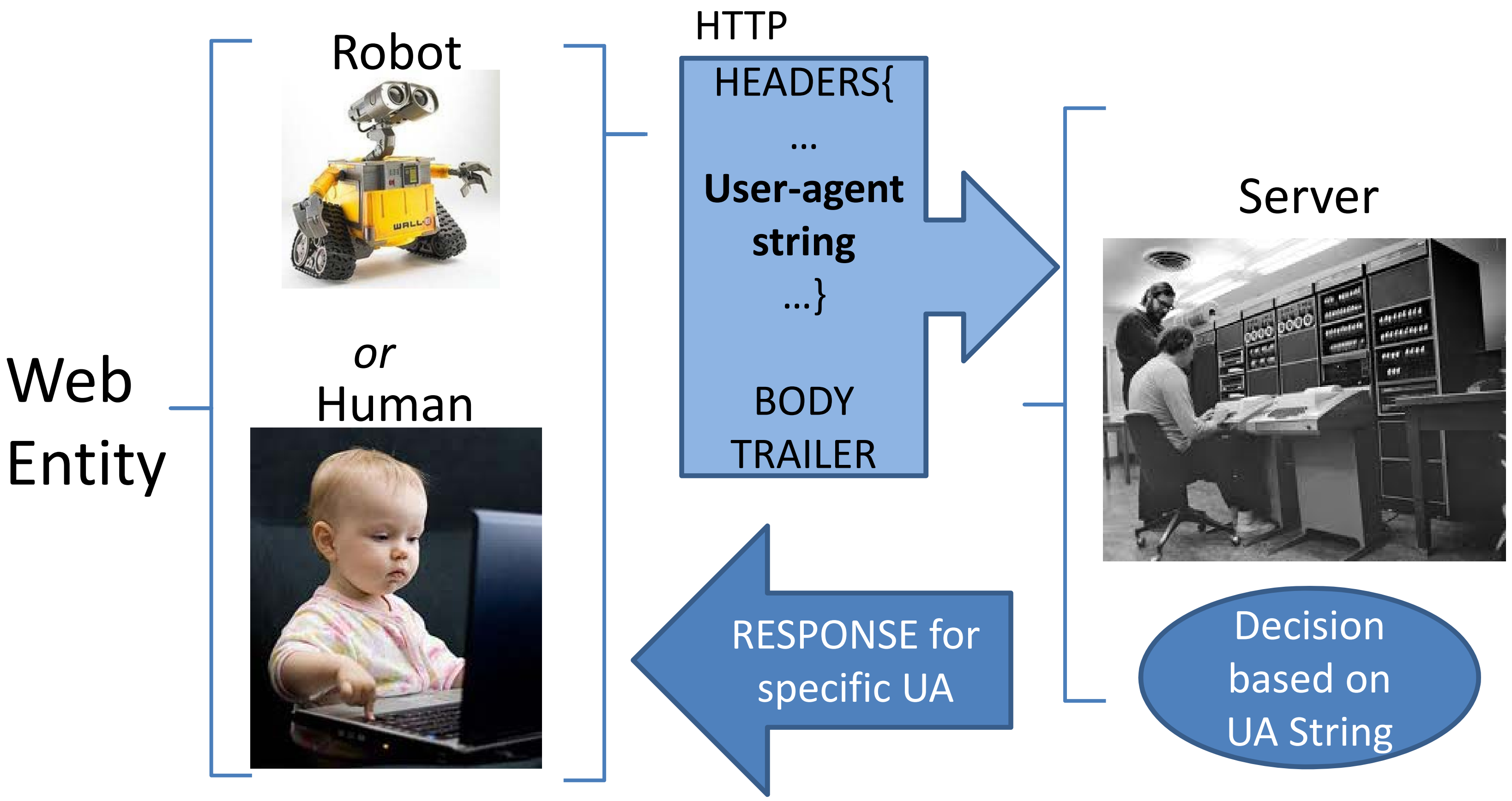
MAUL: Machine Agent User Learning

(some unscrambling necessary)

Robert Holley
Daniel Rosenfeld
CS229 Group Project
December 7, 2010


Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/534.10 (KHTML, like Gecko) Chrome/8.0.552.215 Safari/534.10
ML
BROWSER

The User-agent String

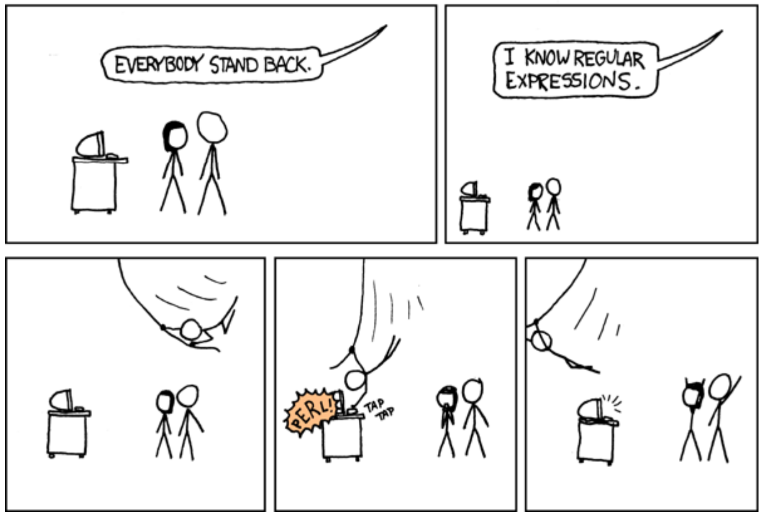


Typical User Agent Strings:

Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0; SV1; .NET CLR 1.1.4322; .NET CLR 1.0.3705; .NET CLR 2.0.50727)
Mozilla/5.0 (compatible; Googlebot/2.1; http://www.google.com/bot.html)
Mozilla/5.0 (Windows; U; Windows NT 5.1; de; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.3 (.NET CLR 3.5. 30729) (Prevx 3.0.5)
BaiduImagespider+(+http://help.baidu.jp/system/05.html)

Classifying User-agent Strings

- Previous Work:
  - Brittle parsing
  - Regular expressions
- Efforts to maintain UA classifiers since early days of internet.
  - Browscap.dll in early MS web servers (since replaced).
- 10-20 *new* UA strings per week.
  - New UA strings break parsers.
  - Parsing engine must be updated by human.
- No attempts at applying text classification ML algorithms to UA strings.



Problems with Std. Approaches:
Many user agents attempt to deceive the server parsing engine in order to get specific content, i.e. pages optimized for GoogleBot, by adding specific tokens to the UA string. A hierarchal regex engine will be confused by such additions. A discriminative algorithm may still classify the modified string correctly.

UA strings are frequently mangled due to plugins and other modifications. For example many UA strings in our dataset have an entirely different UA string inserted midstream. These UA strings sometimes do not have properly terminated parentheses and are hard to parse for robustly.

SVM Based Text Classification

- SVMs can be used as robust text classifiers due to overfitting protection property of maximal margin classifier.
- Dictionary/Feature Vector with Linear/Poly/RBF Kernels
  - Tokenization of UA strings (for our data set) creates 10^4 features. Tokens are generated by breaking the string apart at any of following characters: **[],./\()** or whitespace. This method is robust, and does not depend on parsing for paired parentheses. For numbers, all numbers of length *d* are coalesced into a single token.

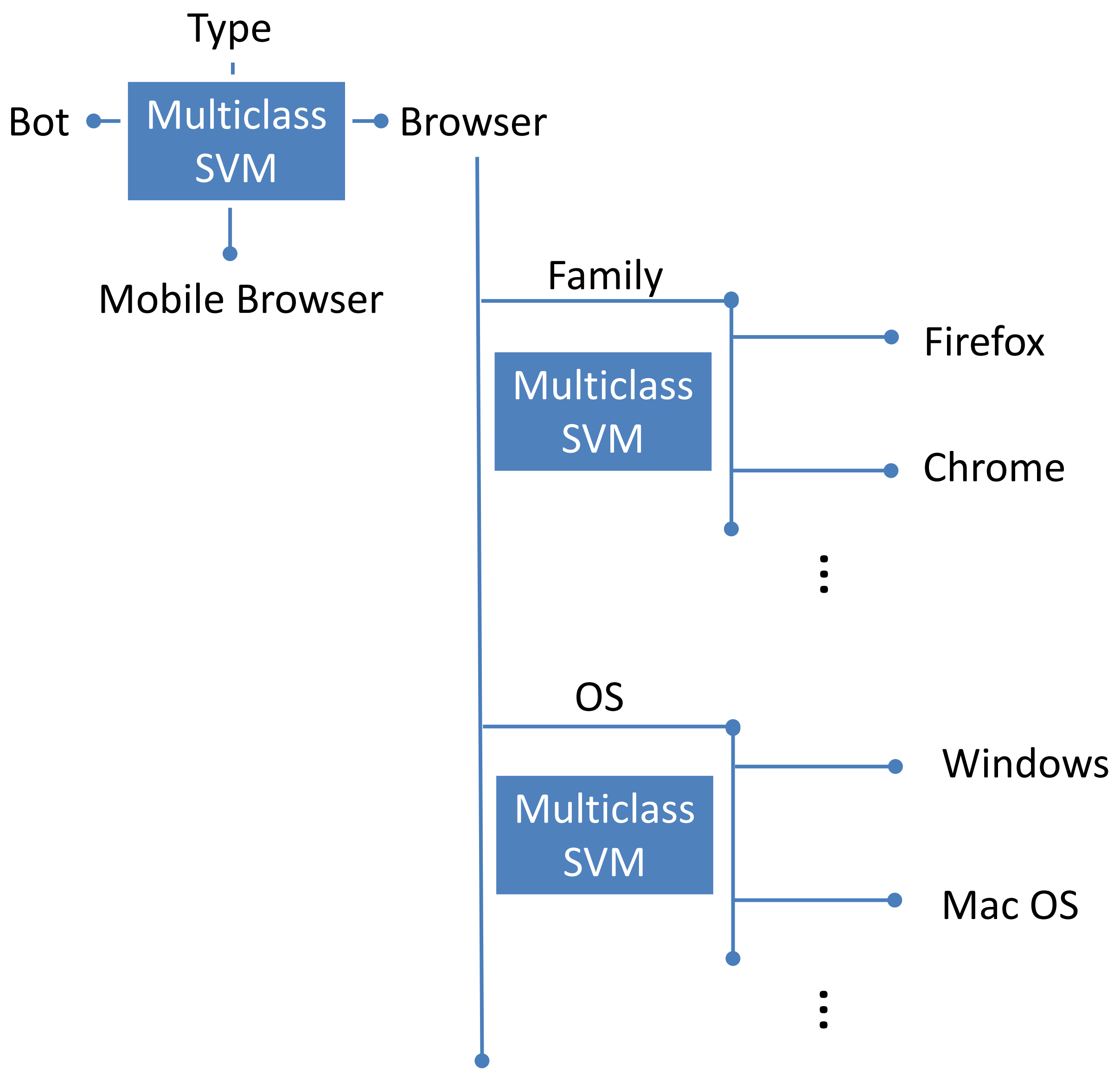
Tokenization Example:

AmigaVoyager/2.95 (compatible; MC680x0; AmigaOS)

AmigaVoyager	2	95	compatible	MC680x0	AmigaOS
2353	0	10	8	3348	1045

- Levenshtein Edit Distance Kernel
  - Edit distance is defined as minimum number of single character edits to change one string into another.
  - Edit distance kernel: $K_{edit}(x,z) = \exp\left[-\gamma \cdot \text{edit}\left(x,z\right)\right]$
  - This kernel scales as $(\textit{string length})^2$, therefore we implement a tokenized version where the string is instead a list of tokens (see above).
- Subsequence Kernel:
  - This kernel compares all shared subsequences of two strings. Subsequences do not have to be contiguous. The longer each string is, the lower its weight in the Kernel. {See: Lodhi et al., Journal of ML Research, Vol. 2, 2002, 419-444 for more details. }
  - This Kernel was also tokenized. Our goal was to include order and position information in the Kernel, which could be important in UA string classification.

Multi-class Classification



Multiclass SVM

With *k* classes, form *k*(*k*-1)/2 SVMs where each SVM is 1 class vs. 1 class.

For prediction on new input, all 1-vs.-1 classifiers are run. The class with the most *wins* is the final prediction.

Results

- For testing purposes, each category under test (chosen from our dataset of >53,000 UA strings) was split, where 80% was used for training and 20% used for testing.

Kernel\Accuracy	Browser v. MB v. Bot	Family	OS
Linear			
RBF			
Edit String			
Subsequence			

Conclusions:

Robust classification possible. Robots easy to tell from browsers, even with attempts at deception. Browser and OS are easily classified by Linear classifier. Edit string and Subsequence Kernels, despite additional complexity over Linear Kernel and inclusion of position and order information, do not outperform the linear classifier.