Keyboard Acoustic Emanations

(or why you shouldn't leave a mic near your keyboard)

Scott Leishman

ML Tea Talk - May 3rd, 2006

Paper Details

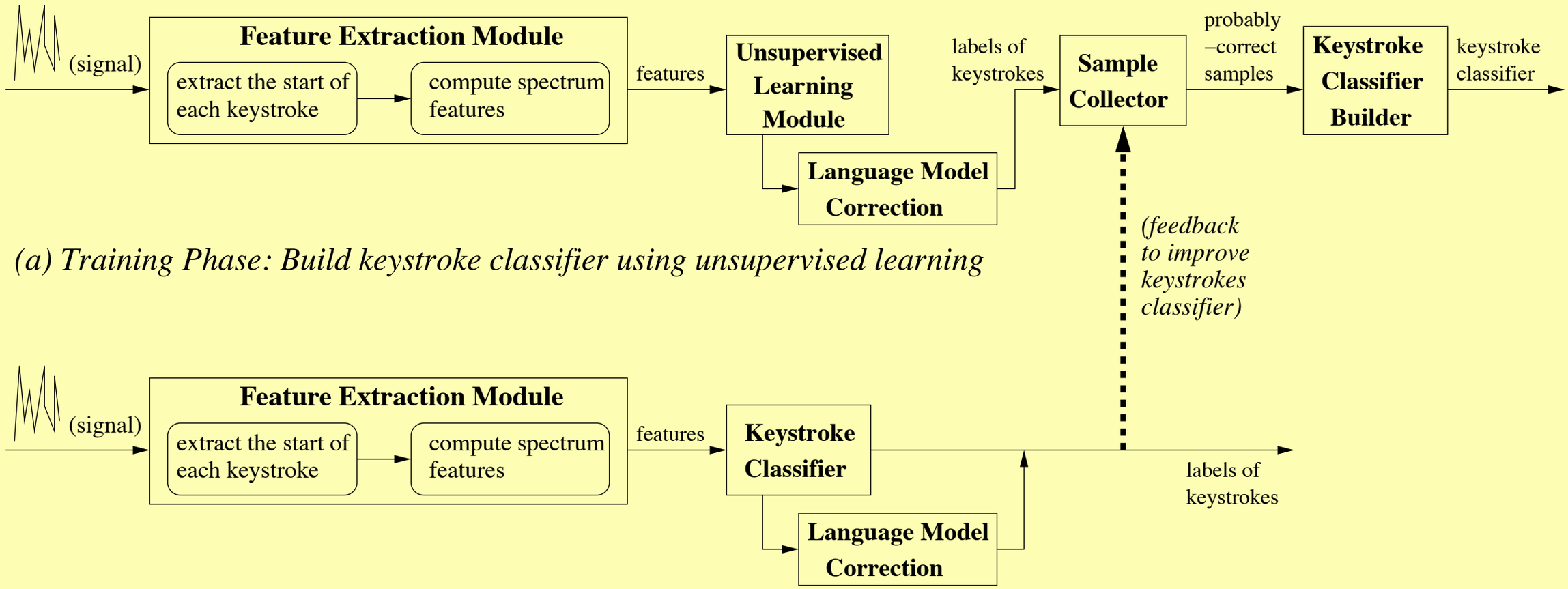
- **Title:** *Keyboard Acoustic Emanations Revisited*
- **Authors:** *Li Zhuang, Feng Zhou, J. D. Tygar (Berkeley)*
- **Presented:** *12th ACM Conference on Computer and Communication Security, November 2005*
- **URL:** *http://www.cs.berkeley.edu/~tygar/keyboard.htm*

2

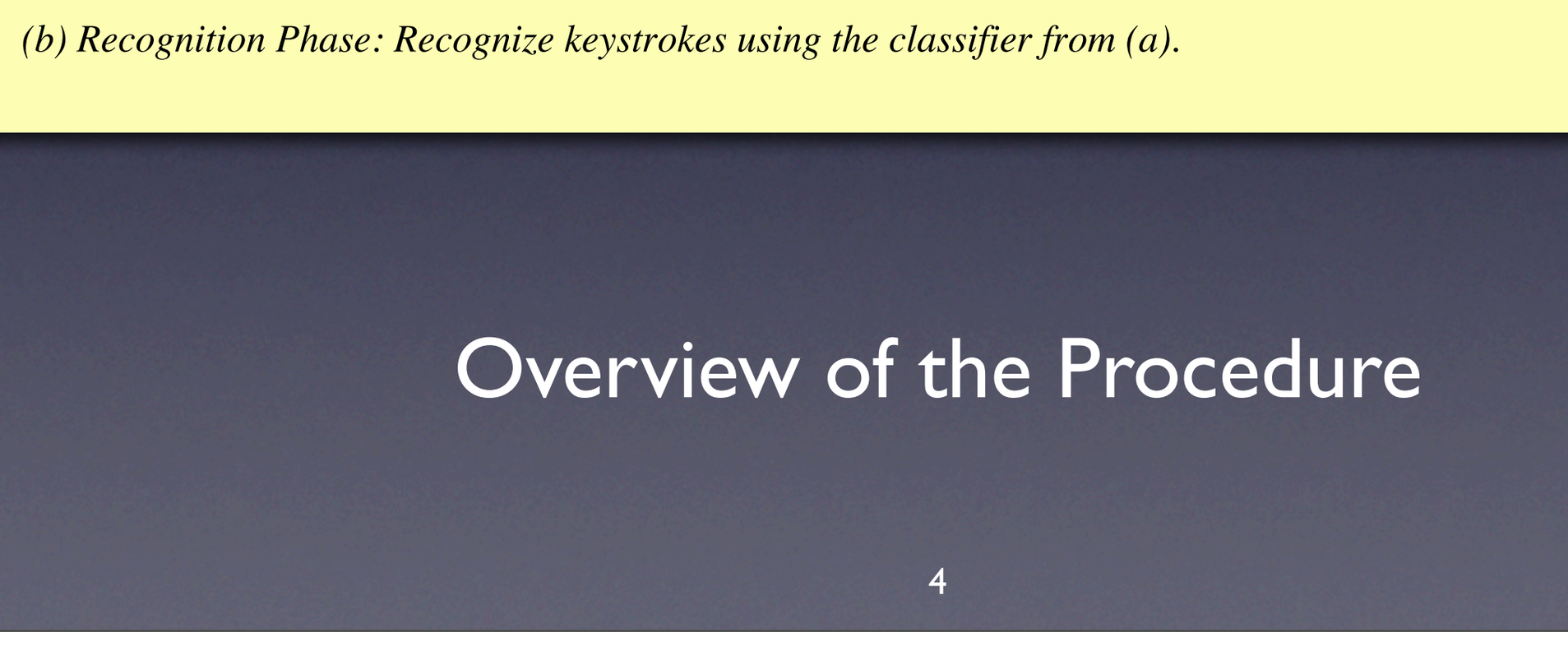
The Basics

- Can we use the sounds a keyboard makes as the user types to infer the sequence of keys pressed?
- Obvious (nefarious) application: eavesdropping and password theft
- Asonov and Agrawal (IBM, 2004) used a similar approach, but they required labeled training data
- Other types of computer emanations previously explored:
  - CRT/LCD electromagnetic radiation (Cryptonomicon!)
  - Acoustics from printers and even CPU's!

3



(a) Training Phase: Build keystroke classifier using unsupervised learning



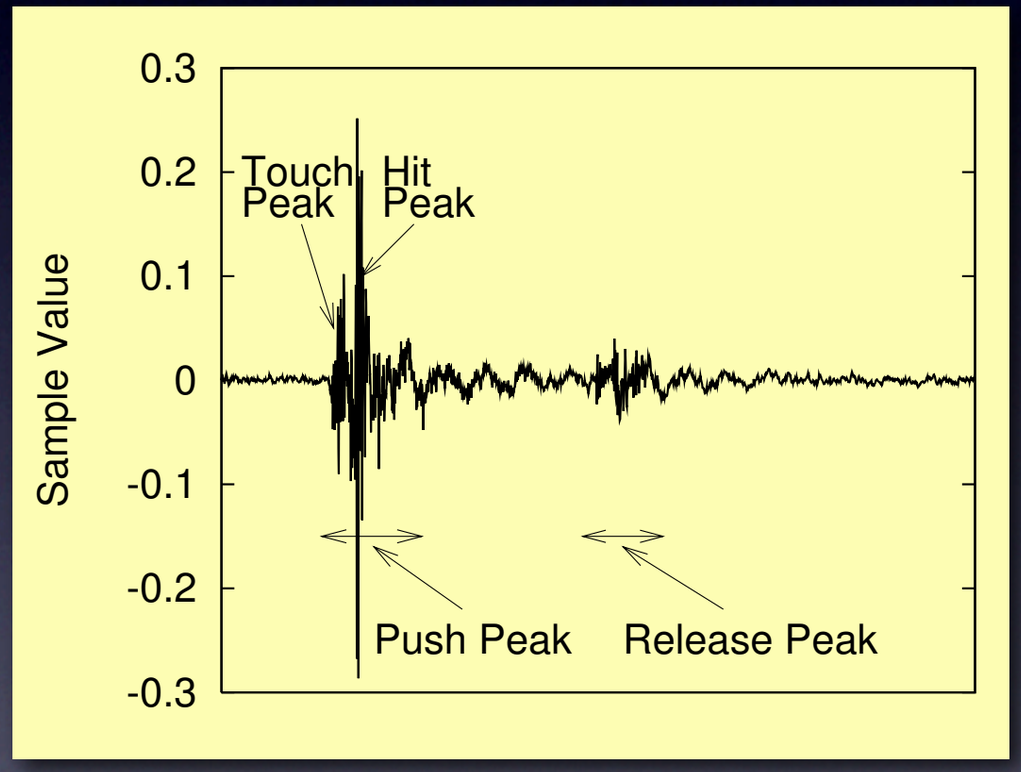
(b) Recognition Phase: Recognize keystrokes using the classifier from (a).

Overview of the Procedure

4

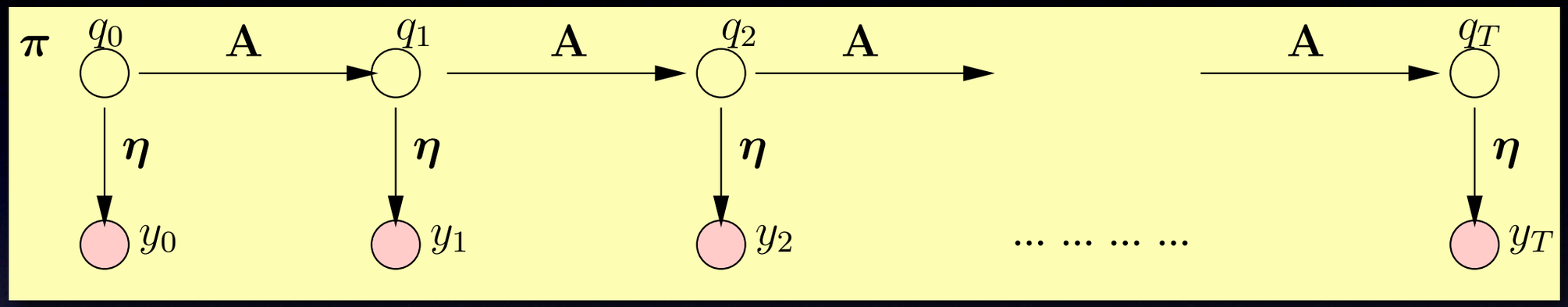
Keystroke Feature Extraction

- Key push to release period: ~100 ms, larger period between consecutive key presses
- Determine start time of key presses by thresholding signal
- Data over 12KHz ignored
- Mel-Frequency Cepstral Coefficients of push peak part of signal determined (10ms sliding window)
- Only consider 30 keys: a-z, space, enter, comma, period



5

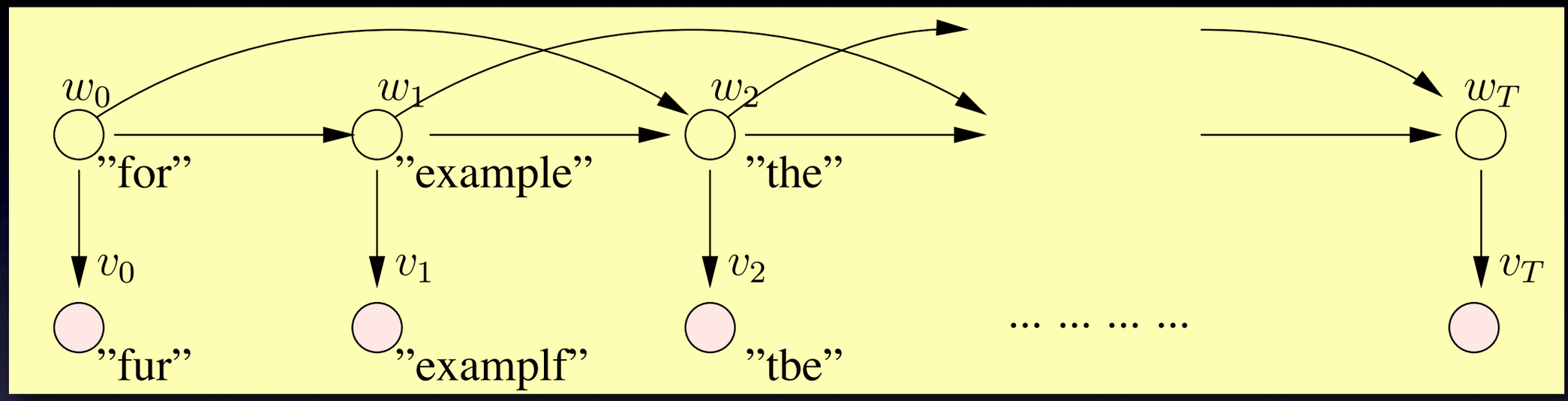
Unsupervised Learning Model



q_i = key label, y_i = cluster class, A_{q_{i-1},q_i} = bigram character matrix

- Use soft K-Means to initially cluster MFCC's over 50 classes
- Use a simple bigram HMM model where the key label transition matrix A_{q_{i-1},q_i} is based on a corpus of english text
- $\eta_{q_i,y_i} = p(y_i|q_i)$ is learned using EM
- Viterbi decoding to infer sequence of key labels

Error Correction Using a Language Model



v_i = recognized word, w_i = corrected word, A_{w_{i-2},w_{i-1},w_i} = trigram word matrx

- To improve performance, recognized key label sequence is split into words and refined using a trigram HMM model
- Emission probabilities are defined by the (regularized) confusion matrix:

$$p(v_i|w_i) = \prod_{j=1}^{|w_i|} E_{v_j,w_j}$$

$$E_{x,y} = \frac{N_{x,y}}{N_x}, \text{ where } x = \text{typed key}, \; y = \text{recognized key}$$

7

Supervised Training and Recognition

- Language corrected results are used as labeled training samples to build a supervised classifier
- Only words with fewer than 1/4 of their characters “corrected” are used.
- 3 classifiers explored: 2-layer Neural Net, LDA classifier, Mixture of Gaussians
- Original (unlabeled) input features are fed back through the classifier, language corrected, then used as further labeled training data to improve the classifier

8

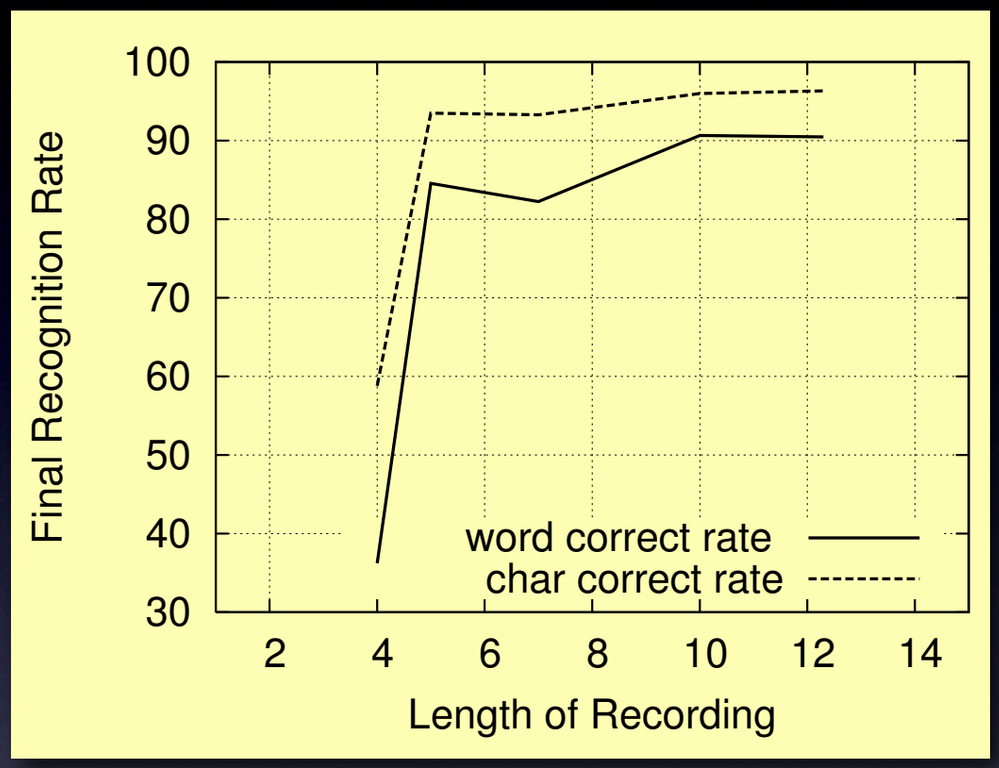
Results

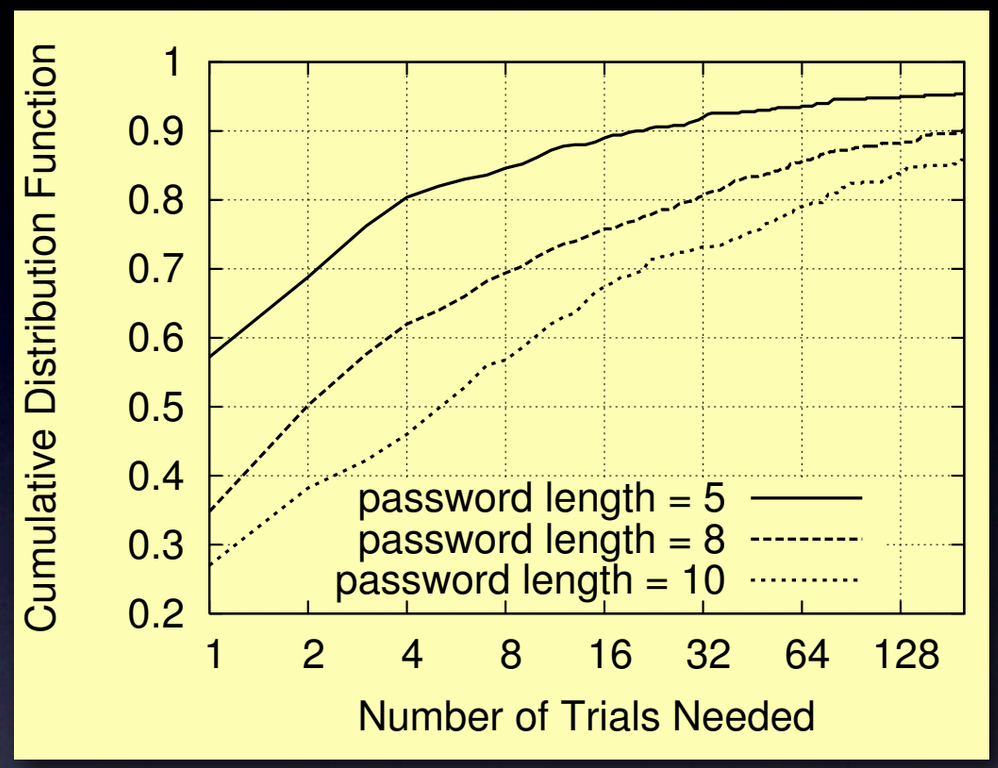
		Set 1		*Set 2*		*Set 3*		*Set 4*	
		words	chars	words	chars	words	chars	words	chars
unsupervised learning	keystrokes	34.72	76.17	38.50	79.60	31.61	72.99	23.22	67.67
	language	74.57	87.19	71.30	87.05	56.57	80.37	51.23	75.07
1st supervised feedback	keystrokes	58.19	89.02	58.20	89.86	51.53	87.37	37.84	82.02
	language	89.73	95.94	88.10	95.64	78.75	92.55	73.22	88.60
2nd supervised feedback	keystrokes	65.28	91.81	62.80	91.07	61.75	90.76	45.36	85.98
	language	90.95	96.46	88.70	95.93	82.74	94.48	78.42	91.49
3rd supervised feedback	keystrokes	66.01	**92.04**	62.70	**91.20**	63.35	**91.21**	48.22	**86.58**
	language	**90.46**	**96.34**	**89.30**	**96.09**	**83.13**	**94.72**	**79.51**	**92.49**

		Neural Network		*Linear Classification*		*Gaussian Mixtures*	
		words	chars	words	chars	words	chars
1st supervised feedback	keystrokes	59.17	87.07	58.19	89.02	59.66	87.03
	language	80.20	90.85	89.73	95.94	78.97	90.45
2nd supervised feedback	keystrokes	70.42	90.33	65.28	91.81	66.99	90.25
	language	81.17	91.21	90.95	96.46	80.20	90.73
3rd supervised feedback	keystrokes	71.39	**90.81**	66.01	**92.04**	69.68	**91.57**
	language	**81.42**	**91.93**	**90.46**	**96.34**	**83.86**	**93.60**

9

Results Continued

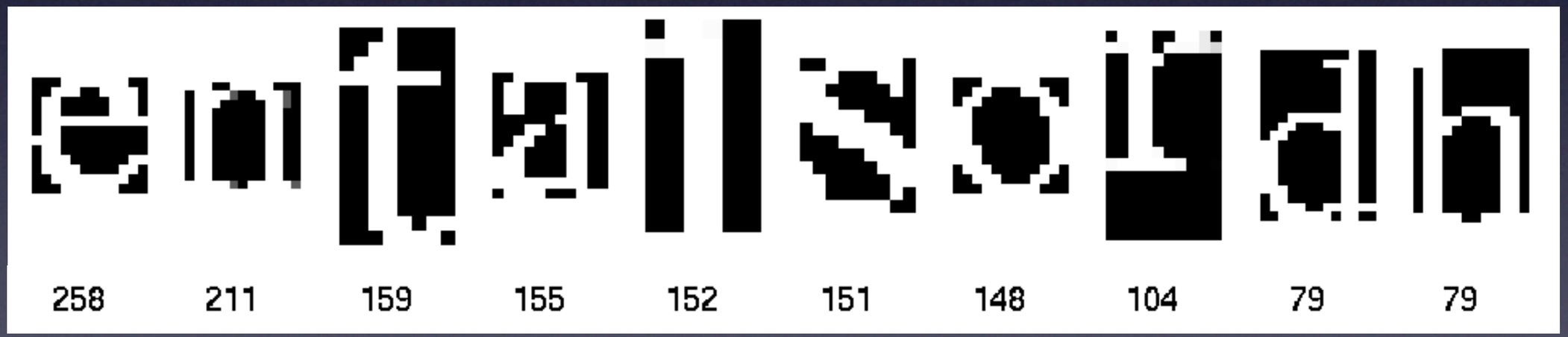




- Results shown using set 1, LDA, 3 feedback iterations
- 90% of 5 character passwords (random text), can be guessed in < 20 attempts, 80% of 10 character passwords in < 75 attempts!

Related Work

- OCR: cluster blobs of ink and combine the number of elements in the cluster with an english language character model to infer the character or symbol belonging to that blob



10 most frequent cluster averages, from a single page of a NIPS 2001 paper