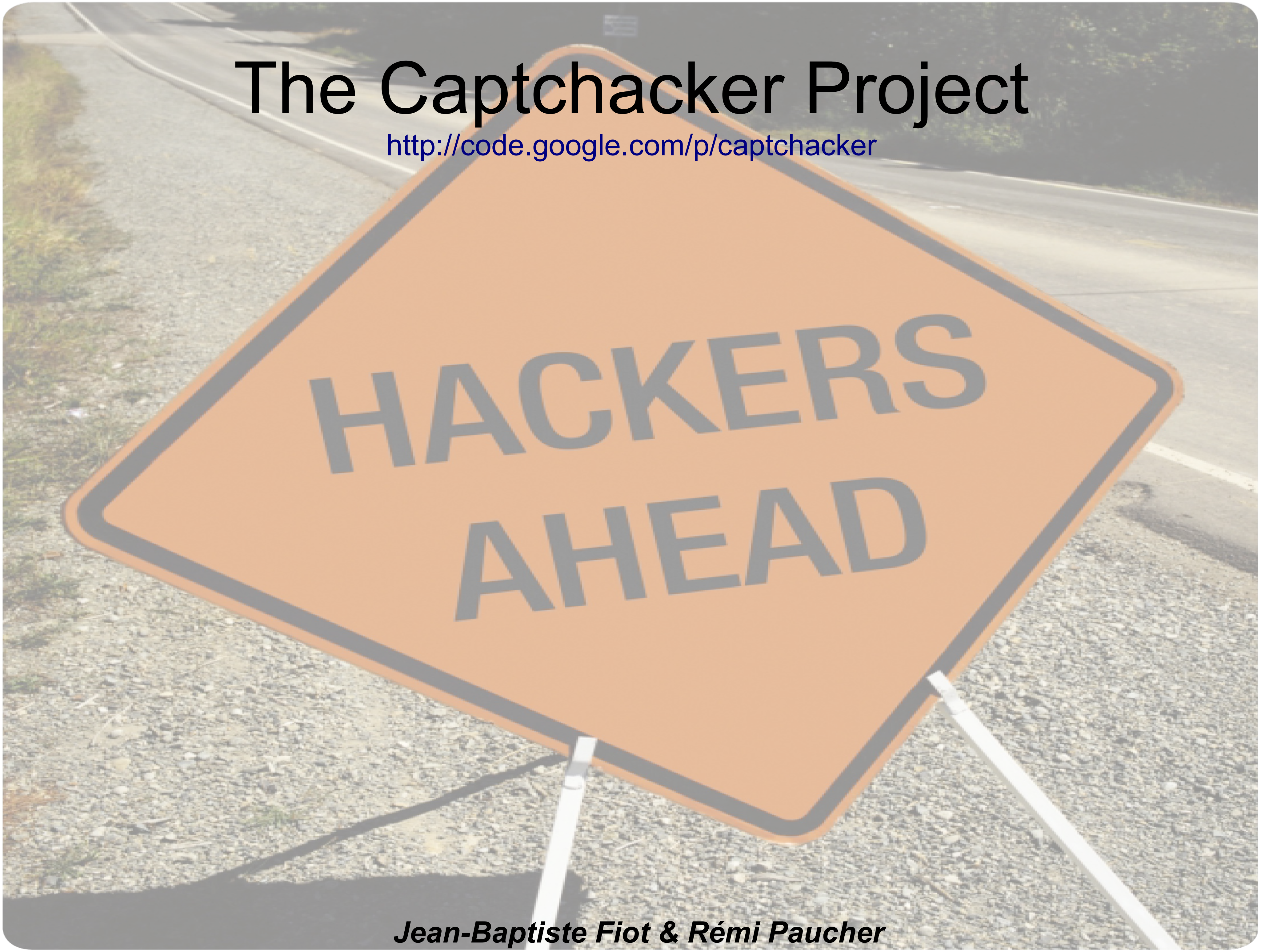
The Captchacker Project

http://code.google.com/p/captchacker



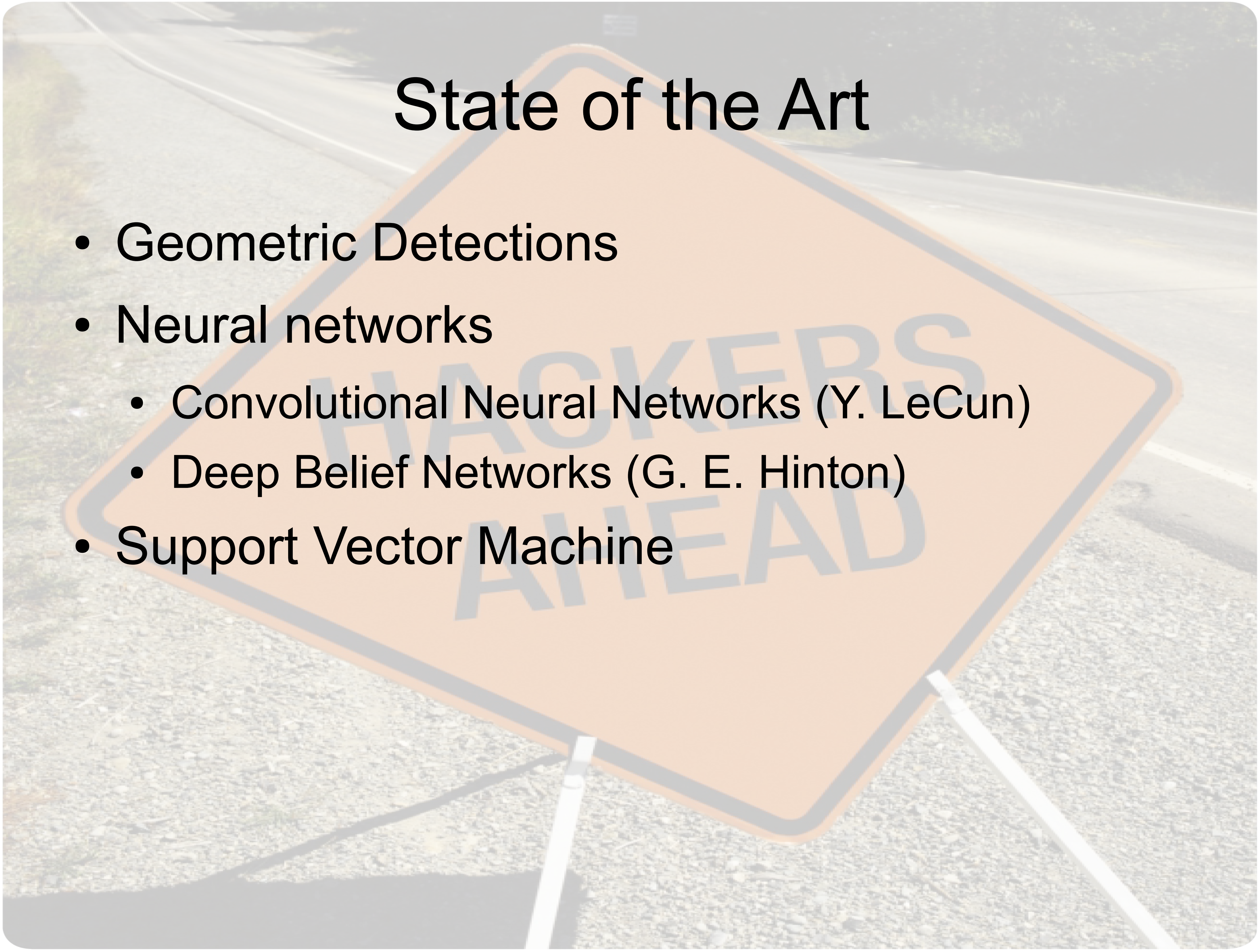
Jean-Baptiste Fiot & Rémi Paucher

You said captcha???

- C.A.P.T.C.H.A = Completely Automated Program To Tell Computers and Humans Apart

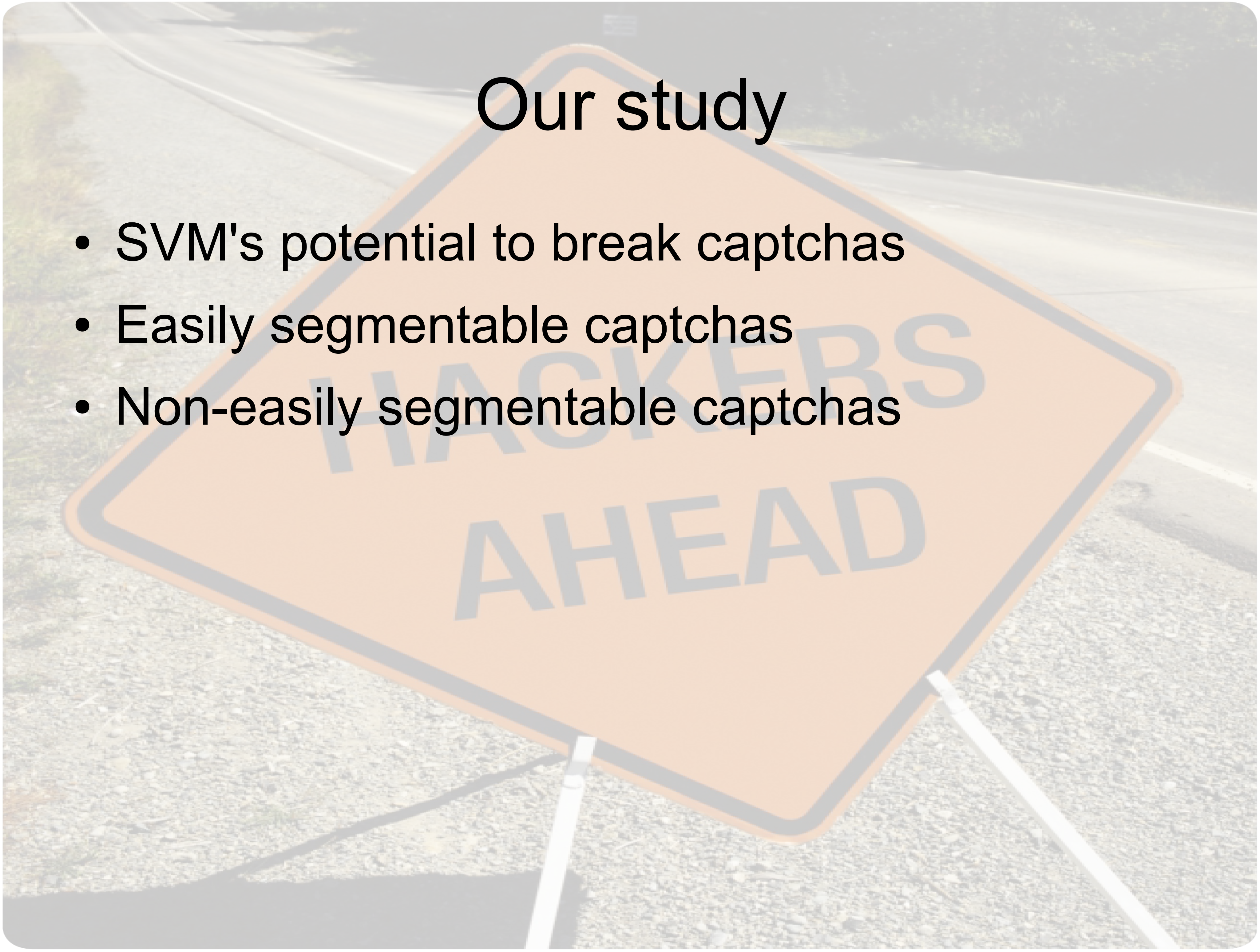
Motivations

- Do automatically these delightful activities:
  - Spam via email address creation
  - Spam comment on blogs
  - Website registration
  - Bias online polls
  - Brute force attack your sista or gf's MSN account



State of the Art

- Geometric Detections
- Neural networks
  - Convolutional Neural Networks (Y. LeCun)
  - Deep Belief Networks (G. E. Hinton)
- Support Vector Machine

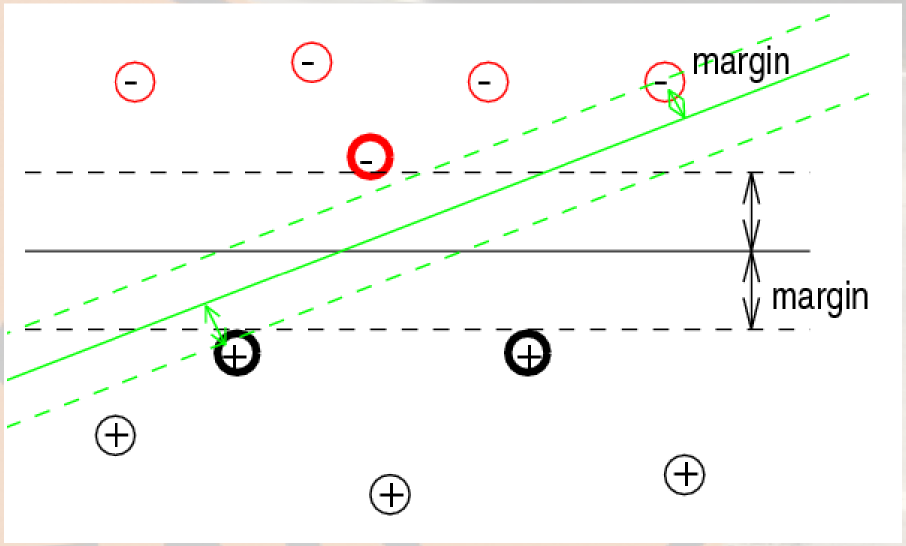


Our study

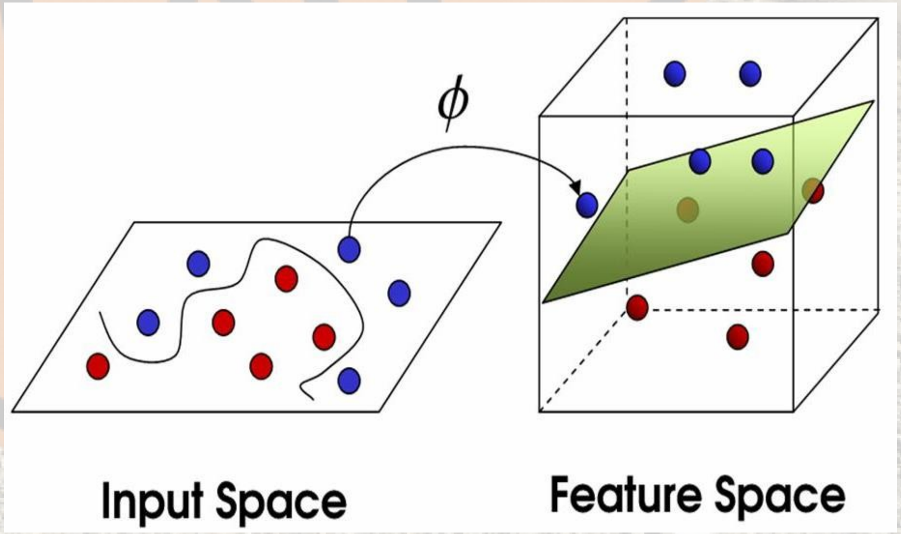
- SVM's potential to break captchas
- Easily segmentable captchas
- Non-easily segmentable captchas

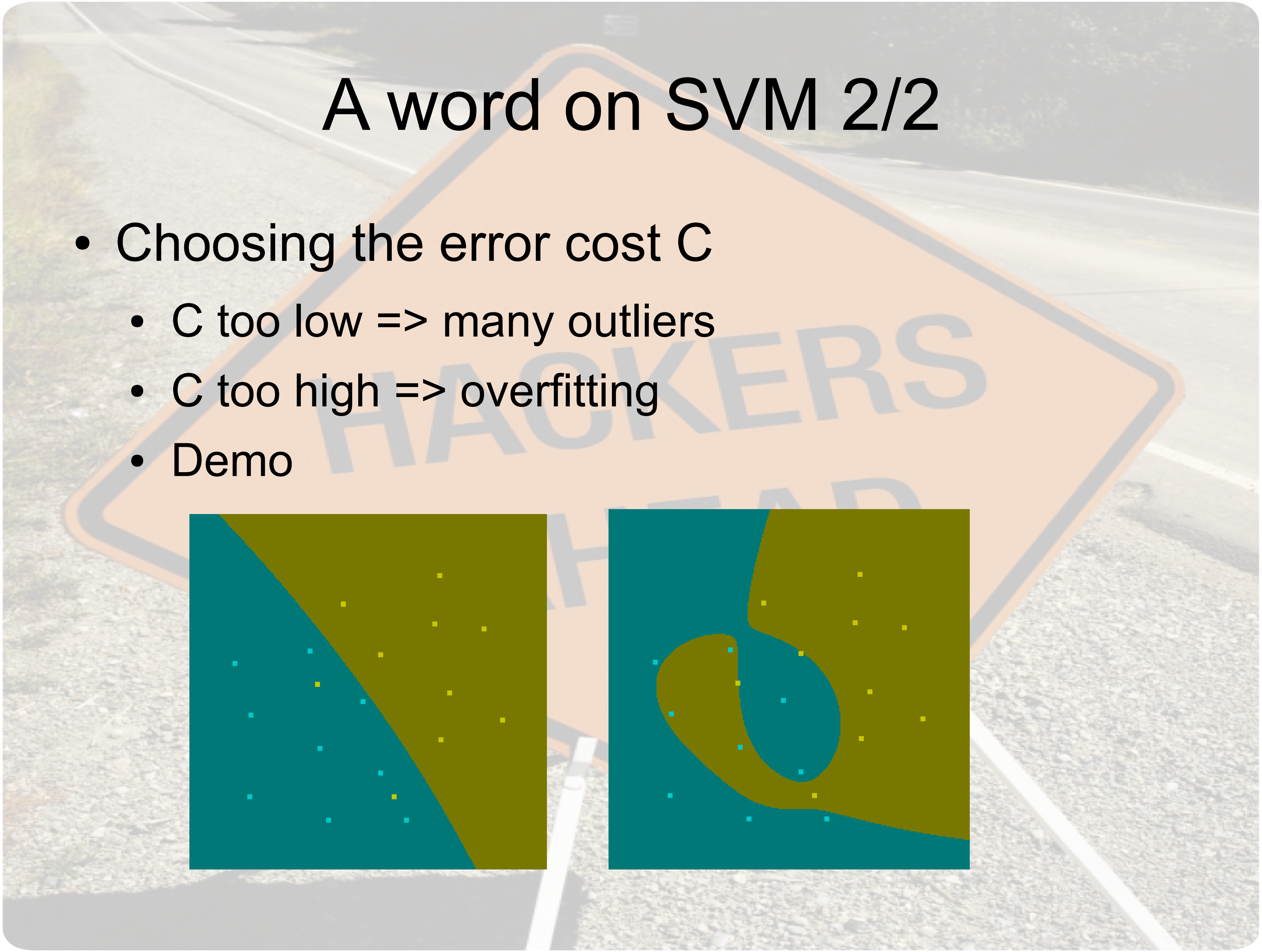
A word on SVM 1/2

- Separate data via a hyperplane maximizing the margin



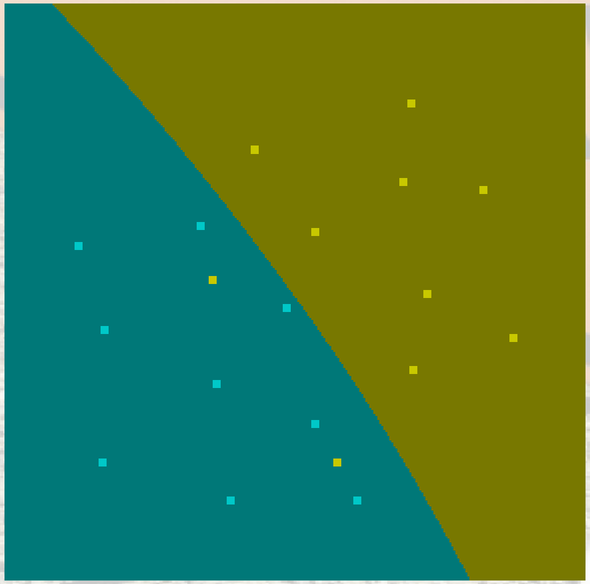
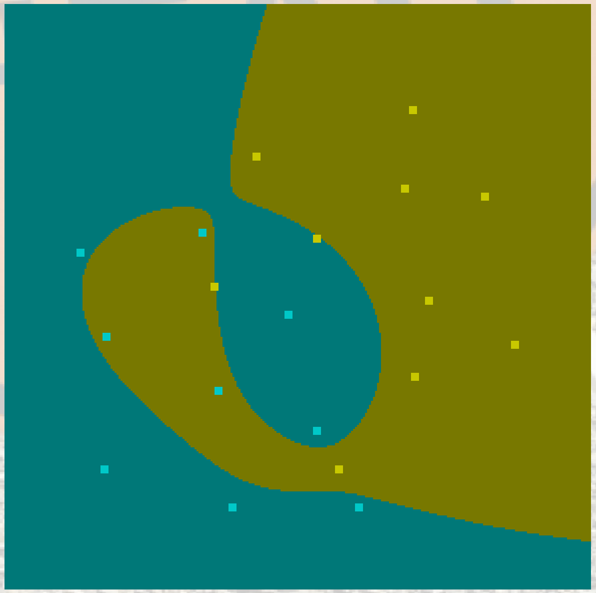
- When the data are not linearly separable
  - Higher dimension space AND/OR
  - Allow outliers

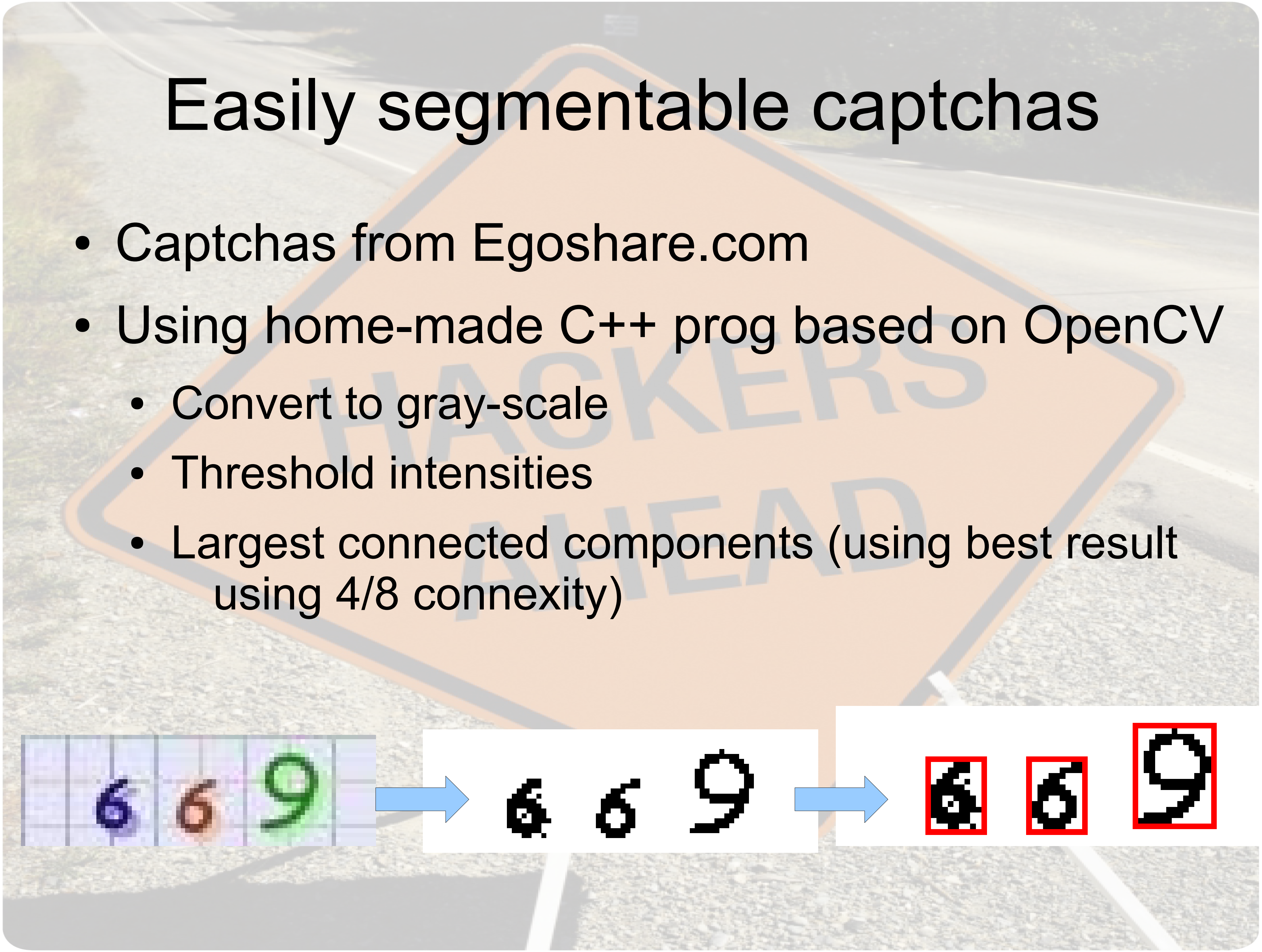




A word on SVM 2/2

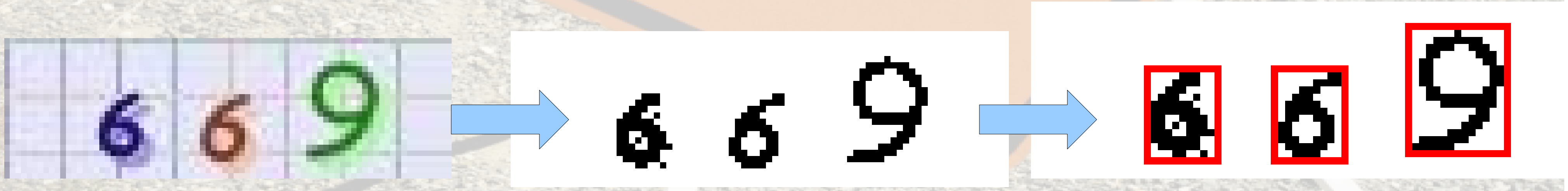
- Choosing the error cost C
  - C too low => many outliers
  - C too high => overfitting
  - Demo

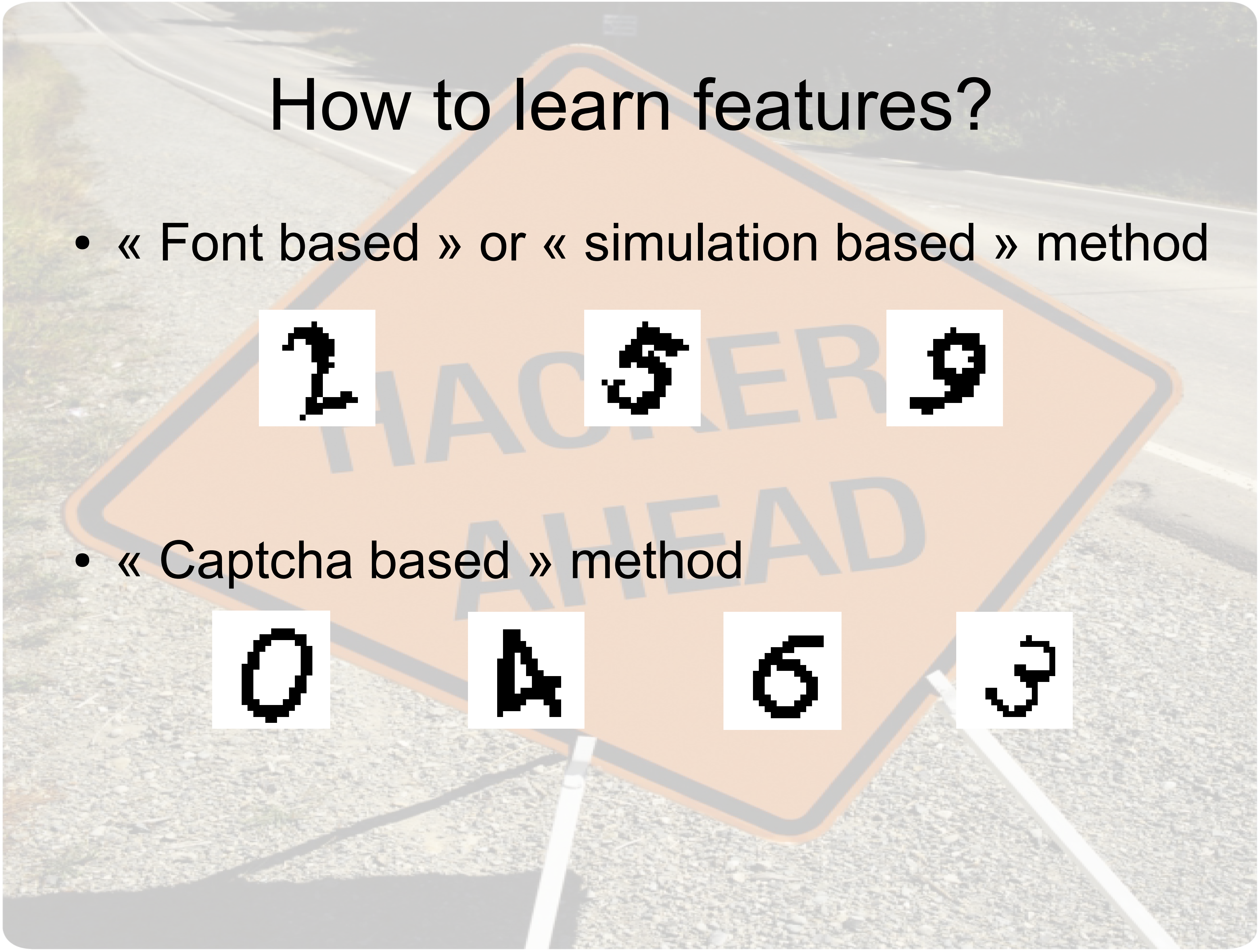





Easily segmentable captchas

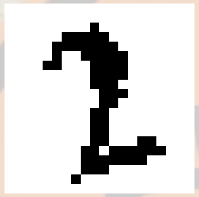
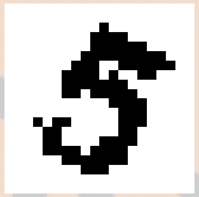
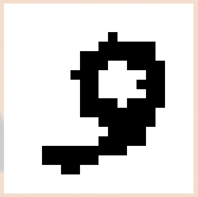
- Captchas from Egoshare.com
- Using home-made C++ prog based on OpenCV
  - Convert to gray-scale
  - Threshold intensities
  - Largest connected components (using best result using 4/8 connexity)



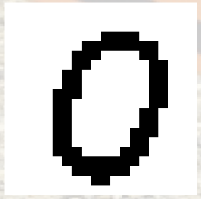
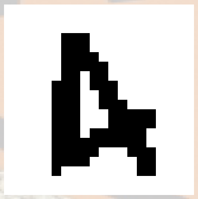
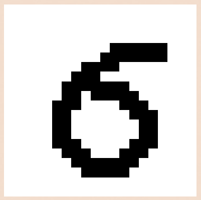
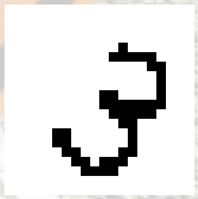


How to learn features?

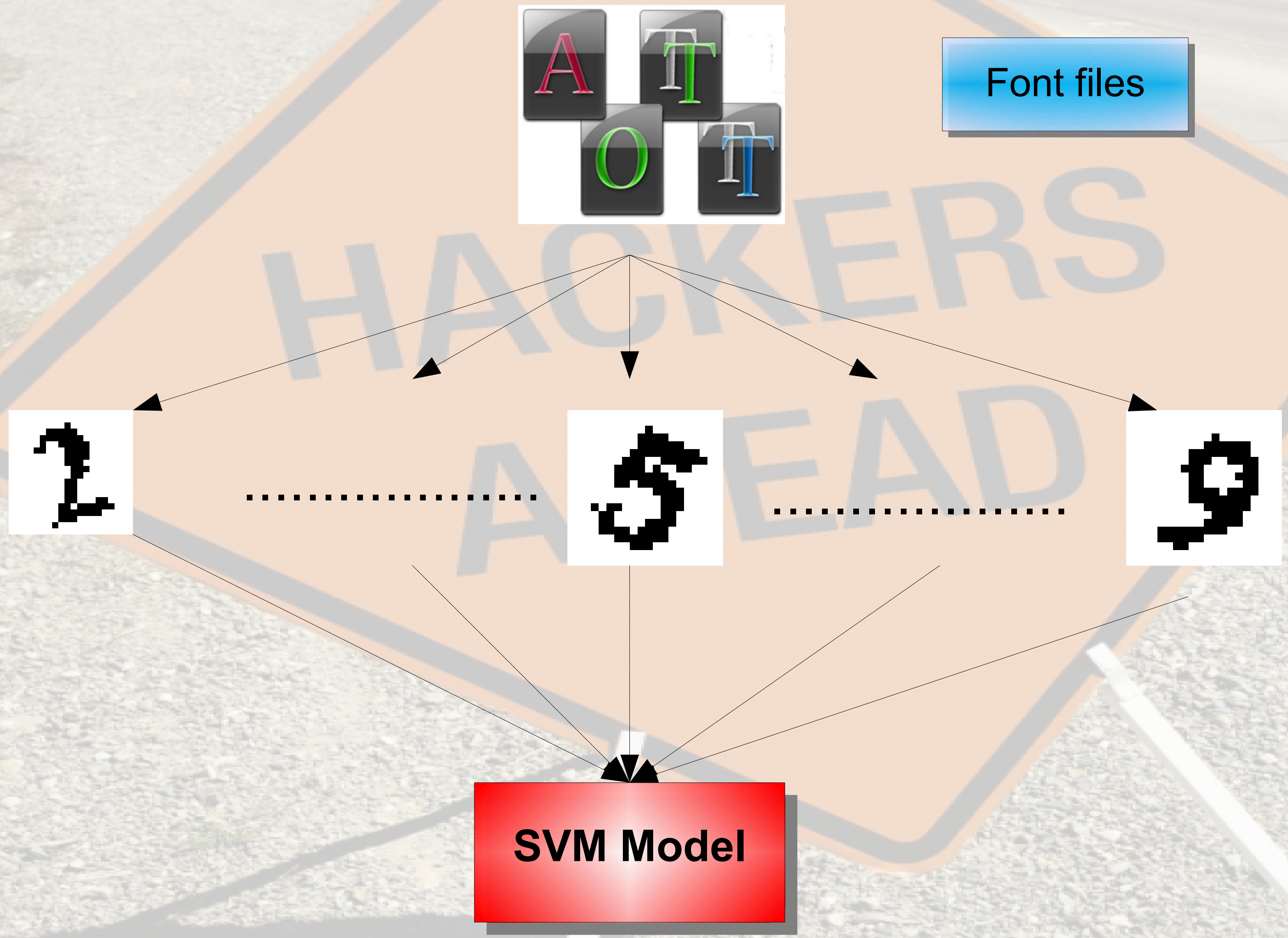
- « Font based » or « simulation based » method

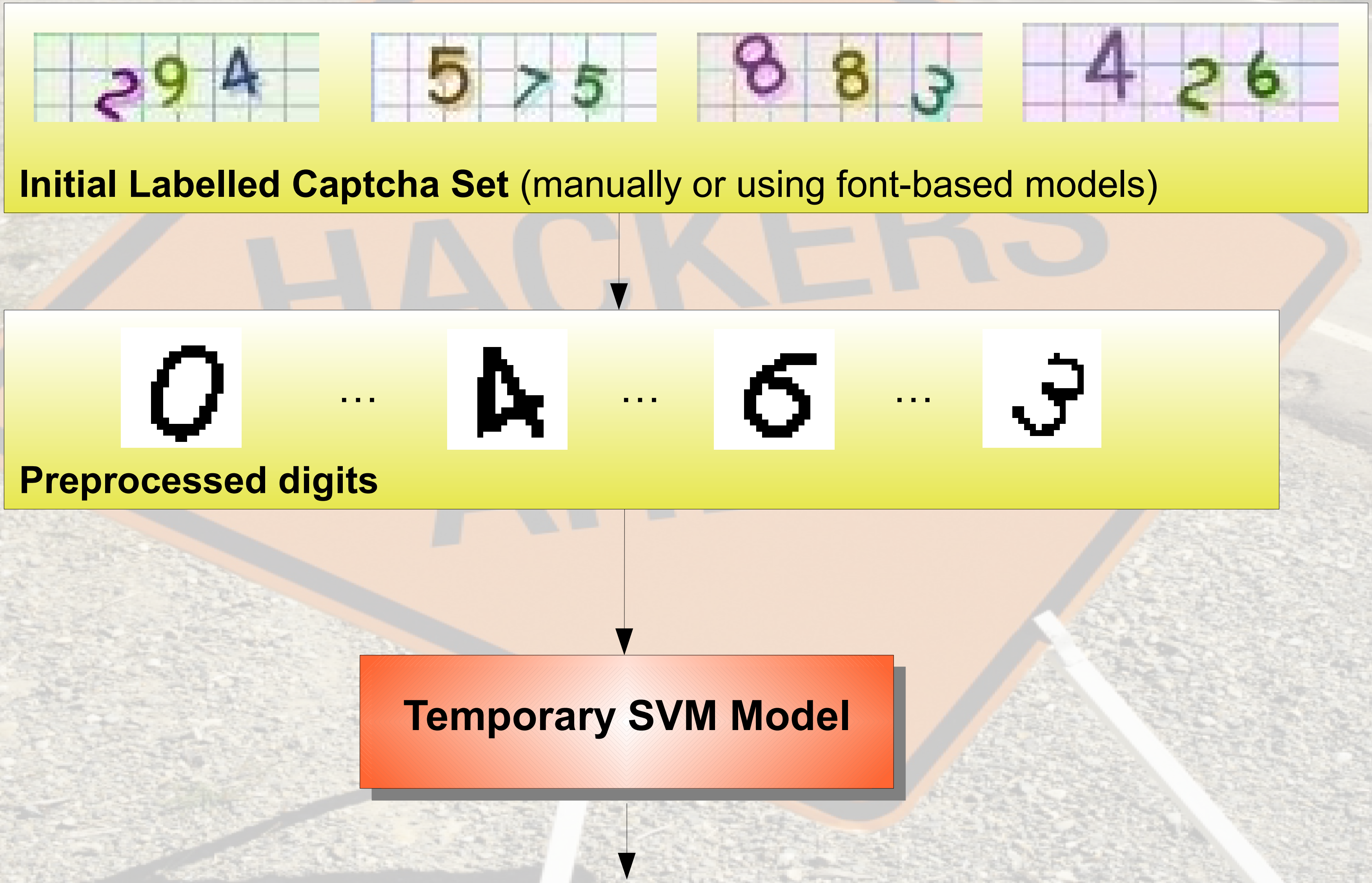
- « Captcha based » method

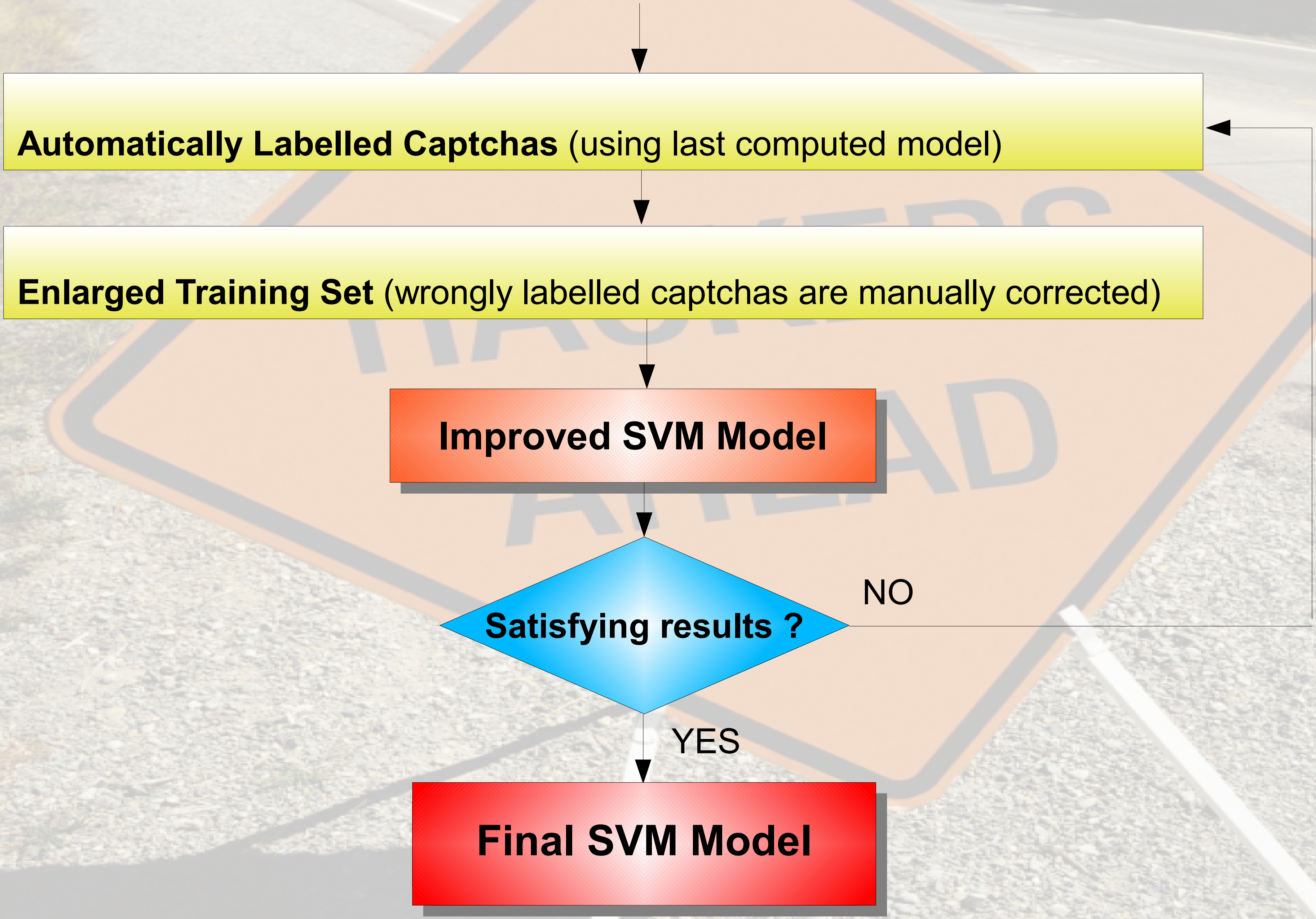
« Simulation-based » method



« Captcha based » method 1/2

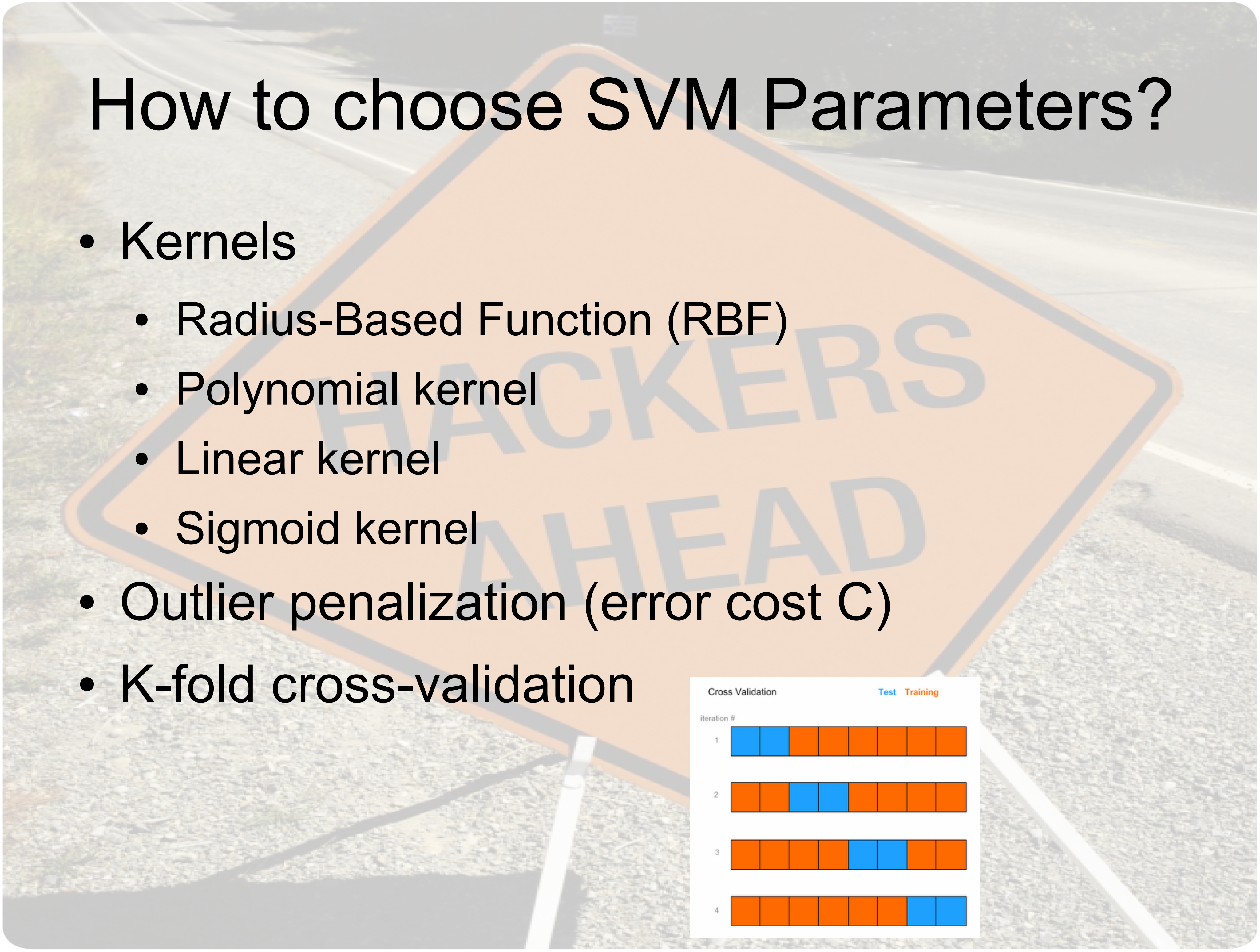


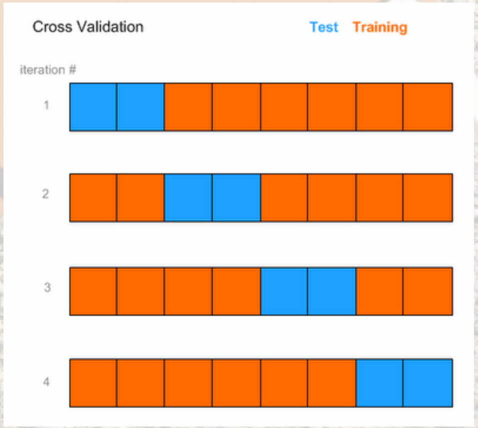
« Captcha based » method 2/2


Automatically Labelled Captchas (using last computed model)
Enlarged Training Set (wrongly labelled captchas are manually corrected)
Improved SVM Model
Satisfying results ?
NO
YES
Final SVM Model

How to choose SVM Parameters?

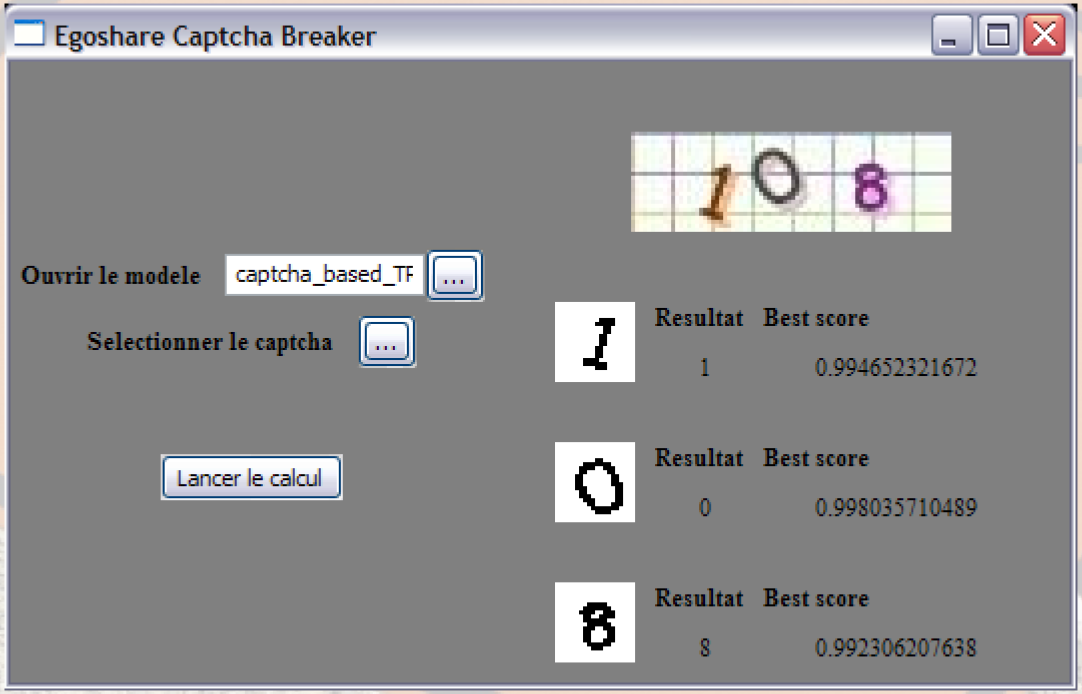
- Kernels
  - Radius-Based Function (RBF)
  - Polynomial kernel
  - Linear kernel
  - Sigmoid kernel
- Outlier penalization (error cost C)
- K-fold cross-validation





So u hack 'hem all or what???

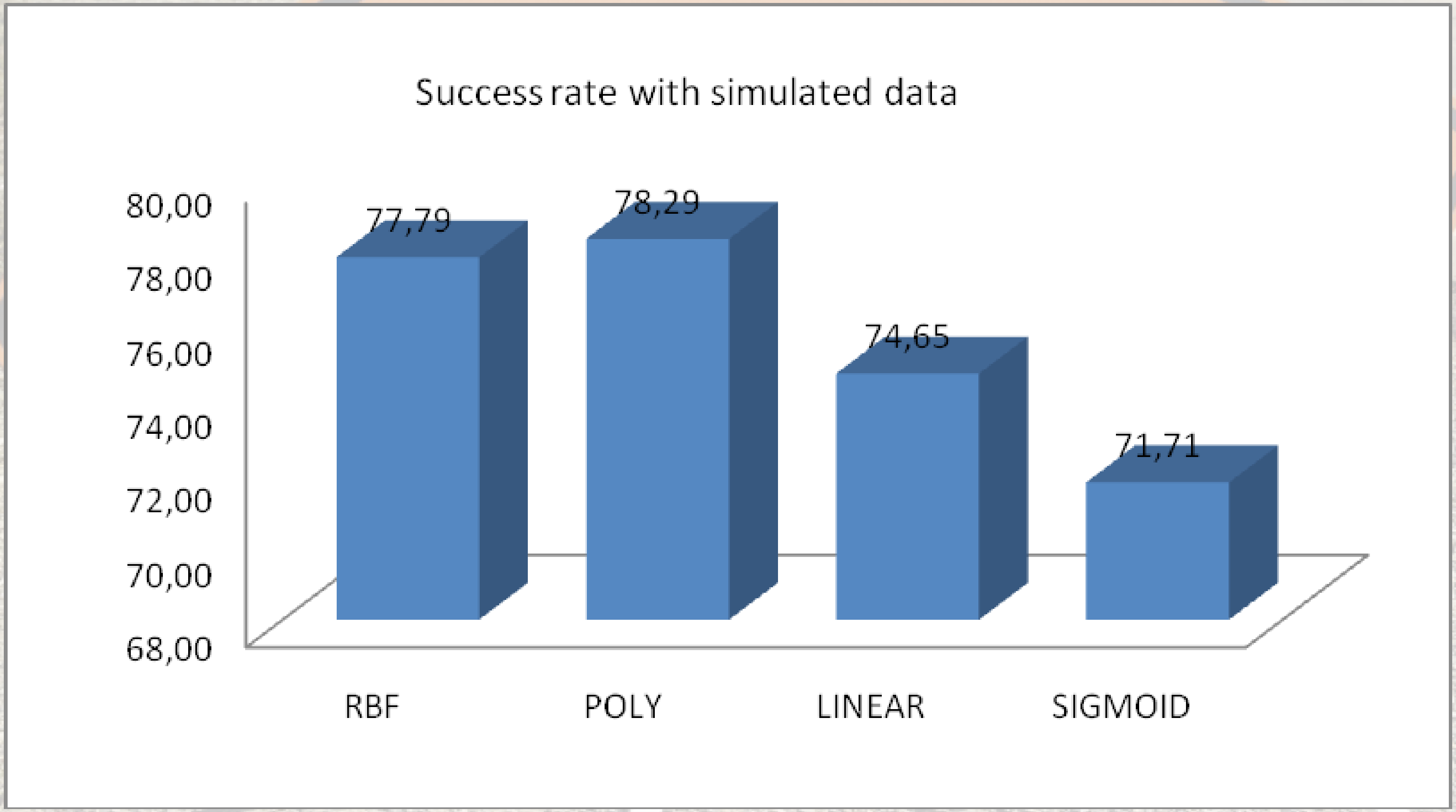
- Check this demo out!



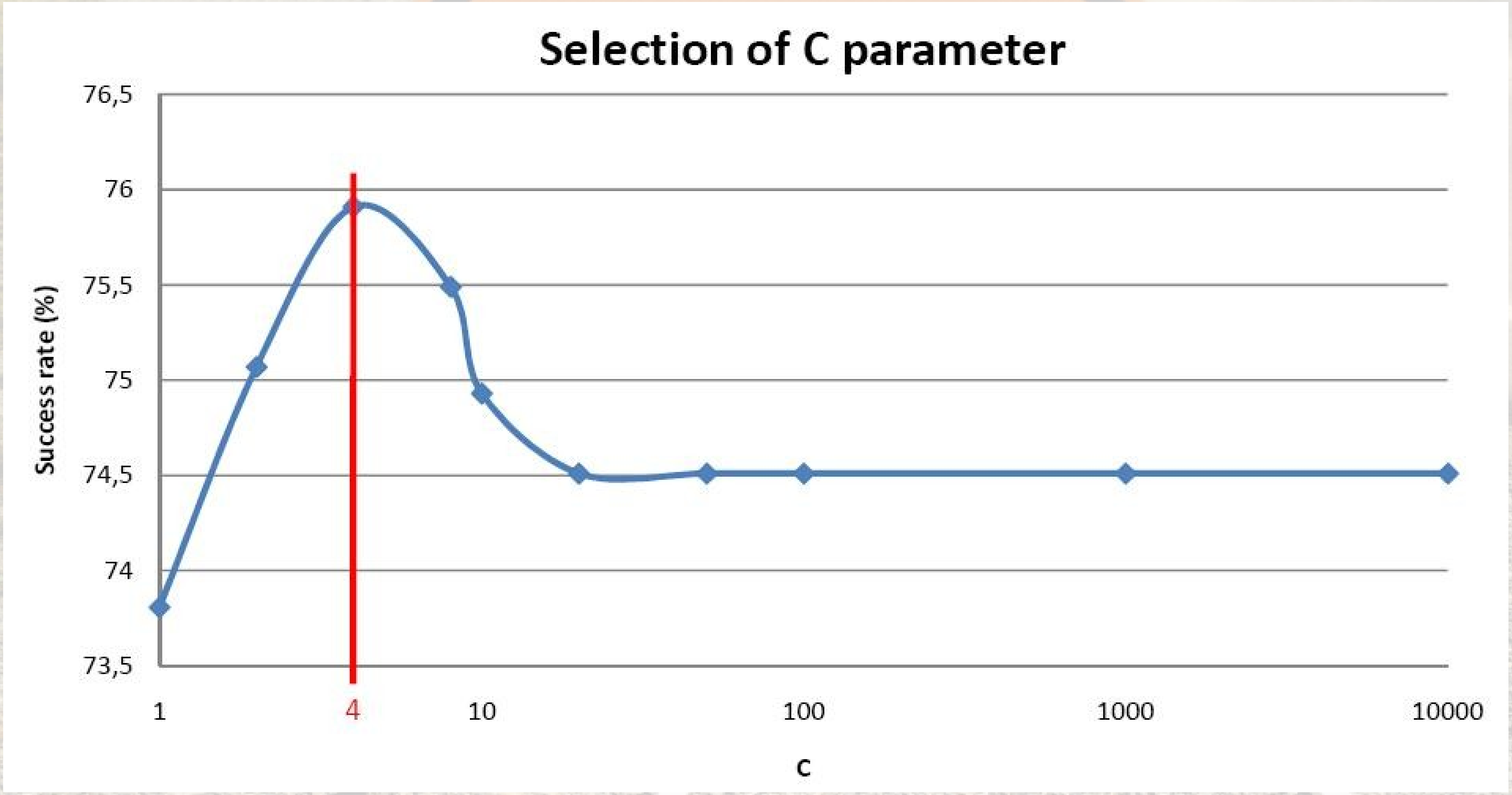
```
Resultat   Best score
1          0.994652321672

Resultat   Best score
0          0.998035710489

Resultat   Best score
8          0.992306207638
```

Performance 1/4

- Simulation based method:

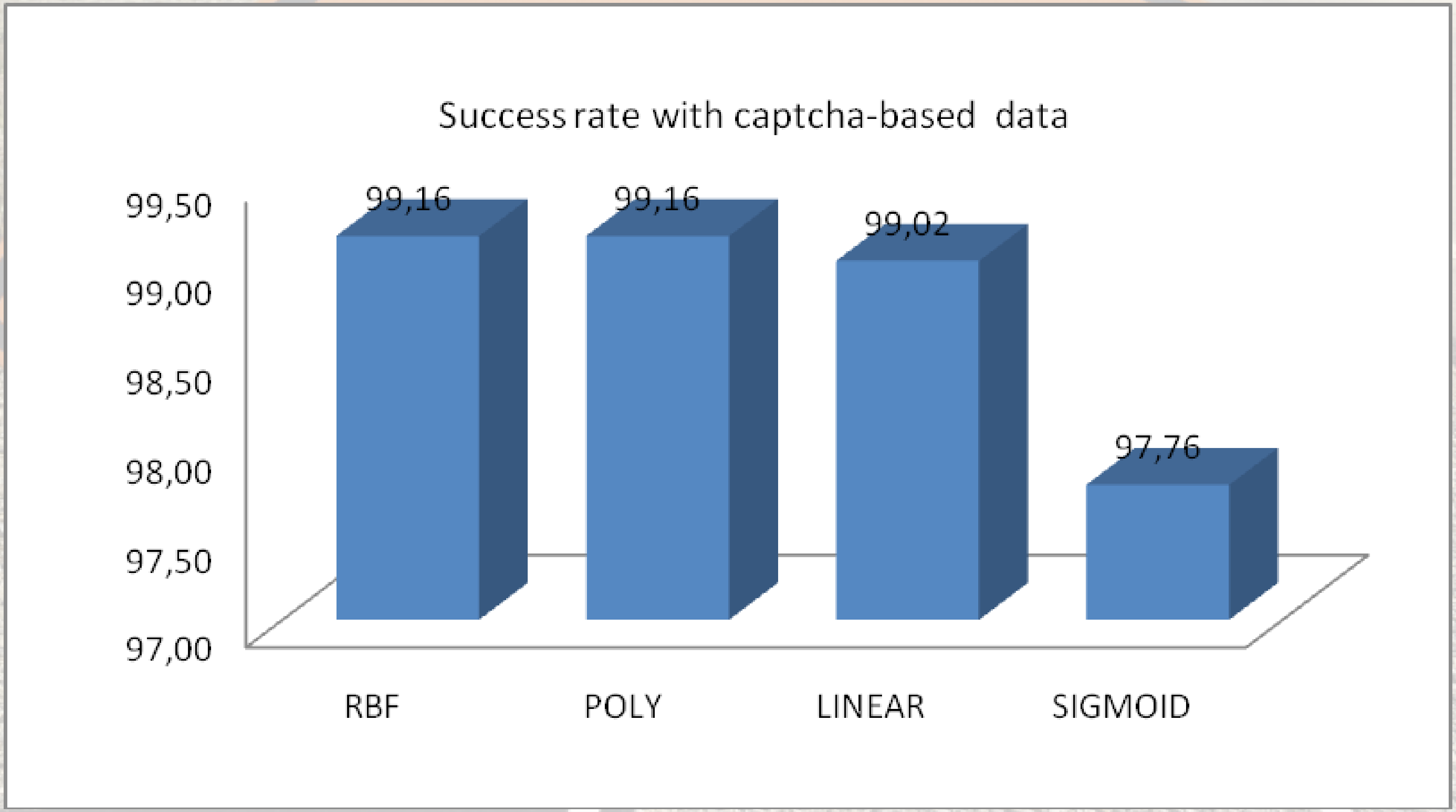


Performance 2/4

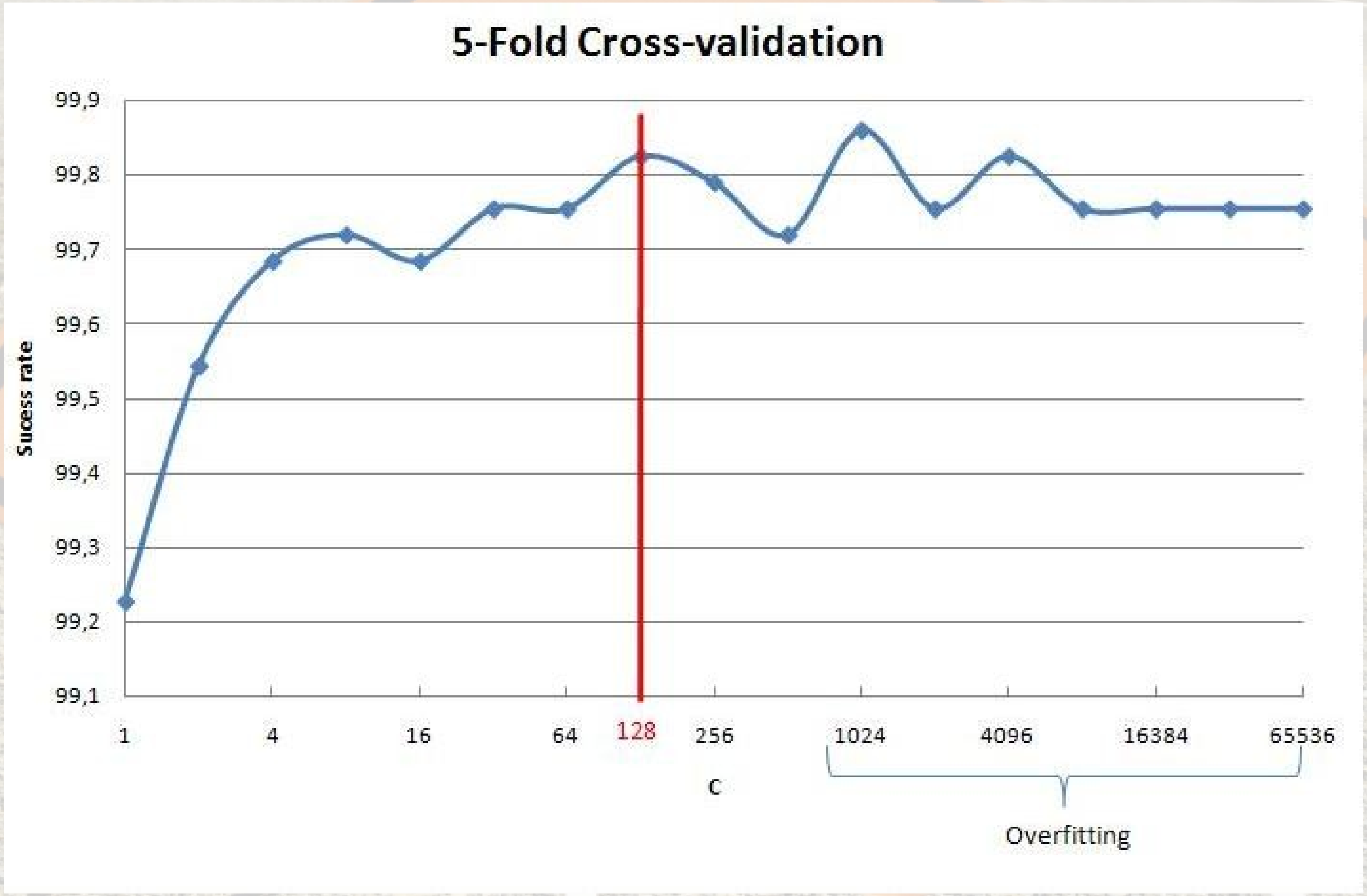


Performance 3/4

- Captcha-based method:

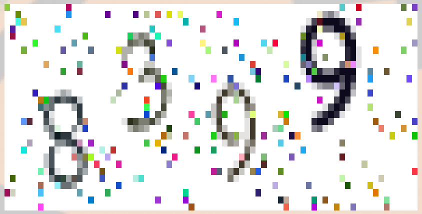


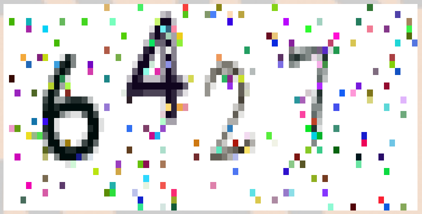
Performance 4/4

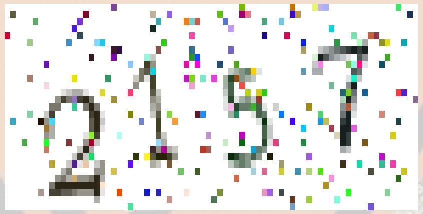


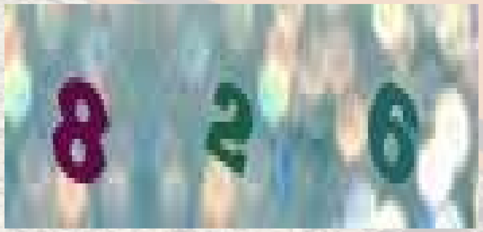
Other easily segmentable captchas

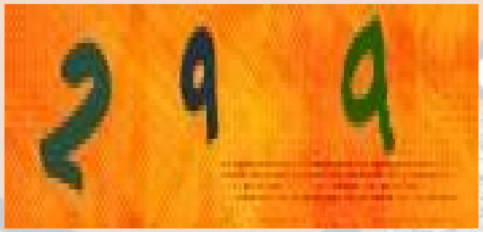
- Nice results with a specific model
  - Portable method :)



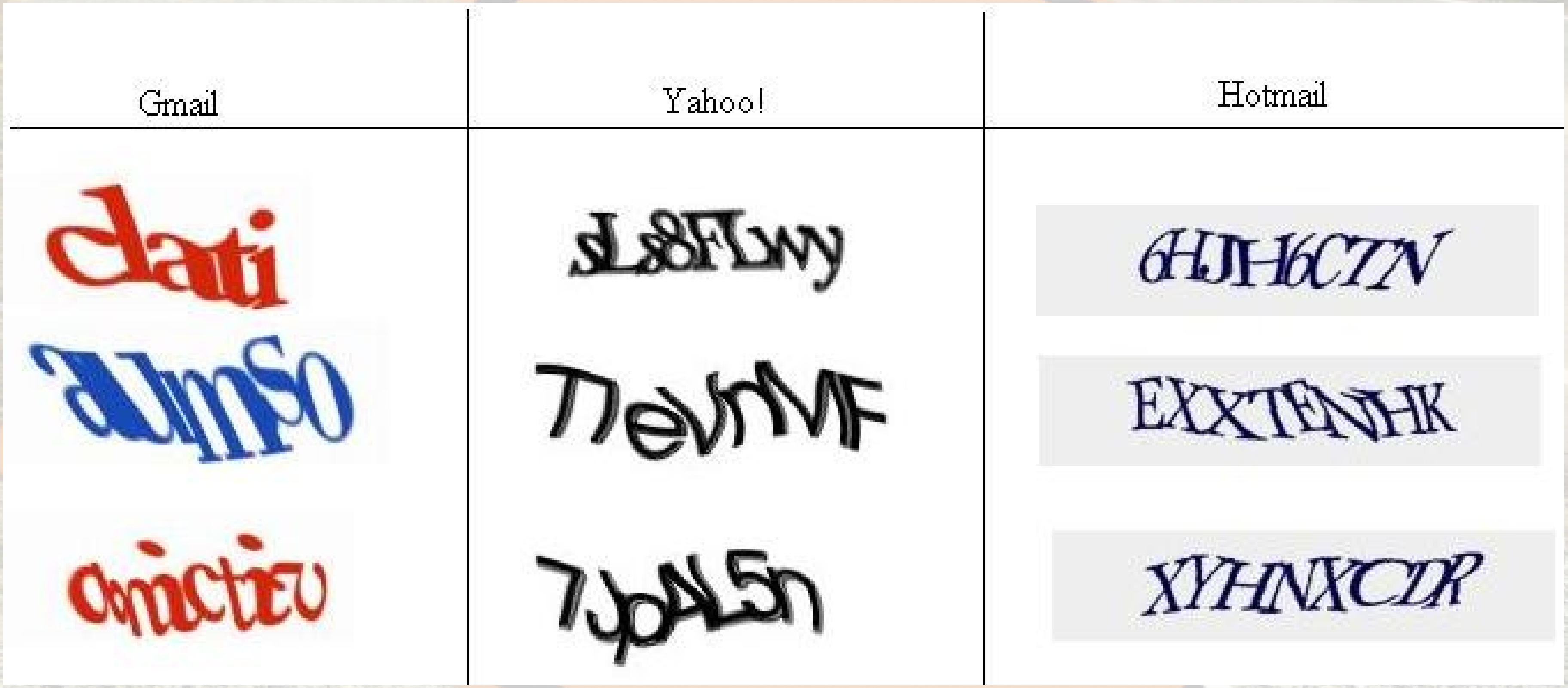






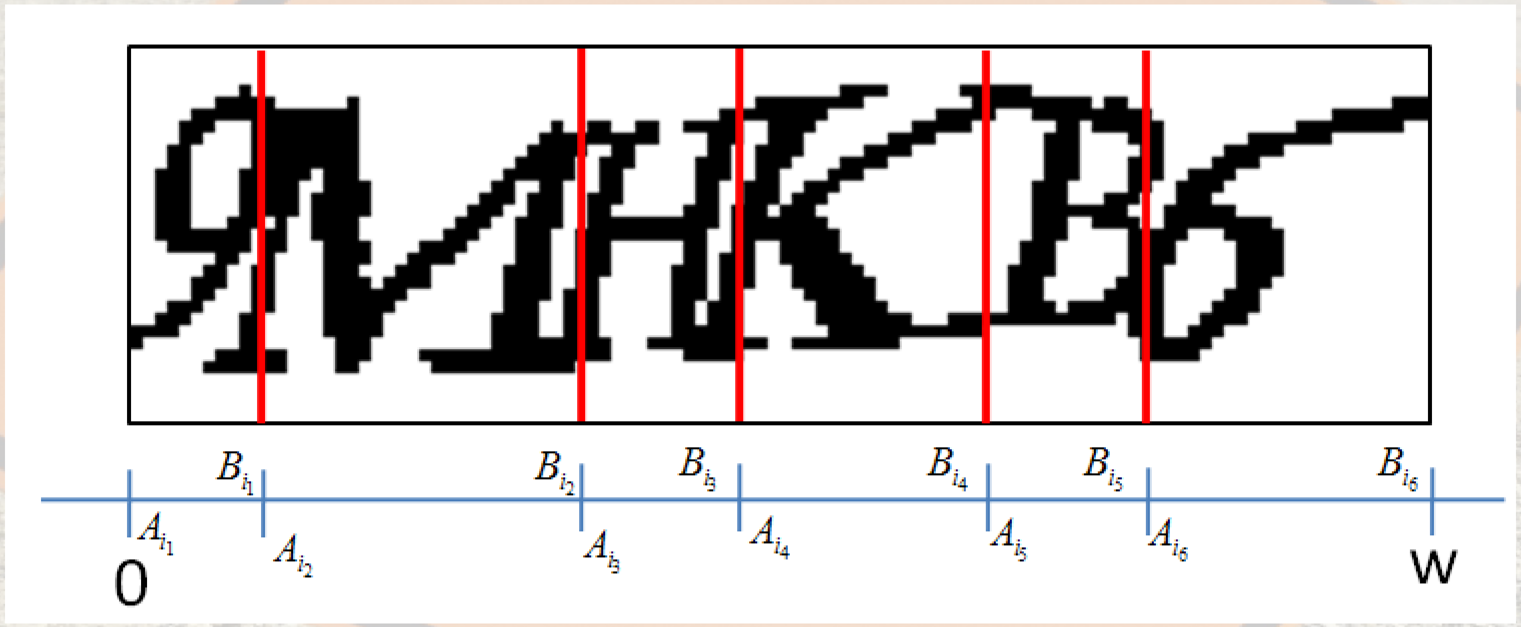


Non-easily segmentable captchas


Gmail
Yahoo!
Hotmail
6HJH6C7N
EXXTENHK
XYHNXCDR

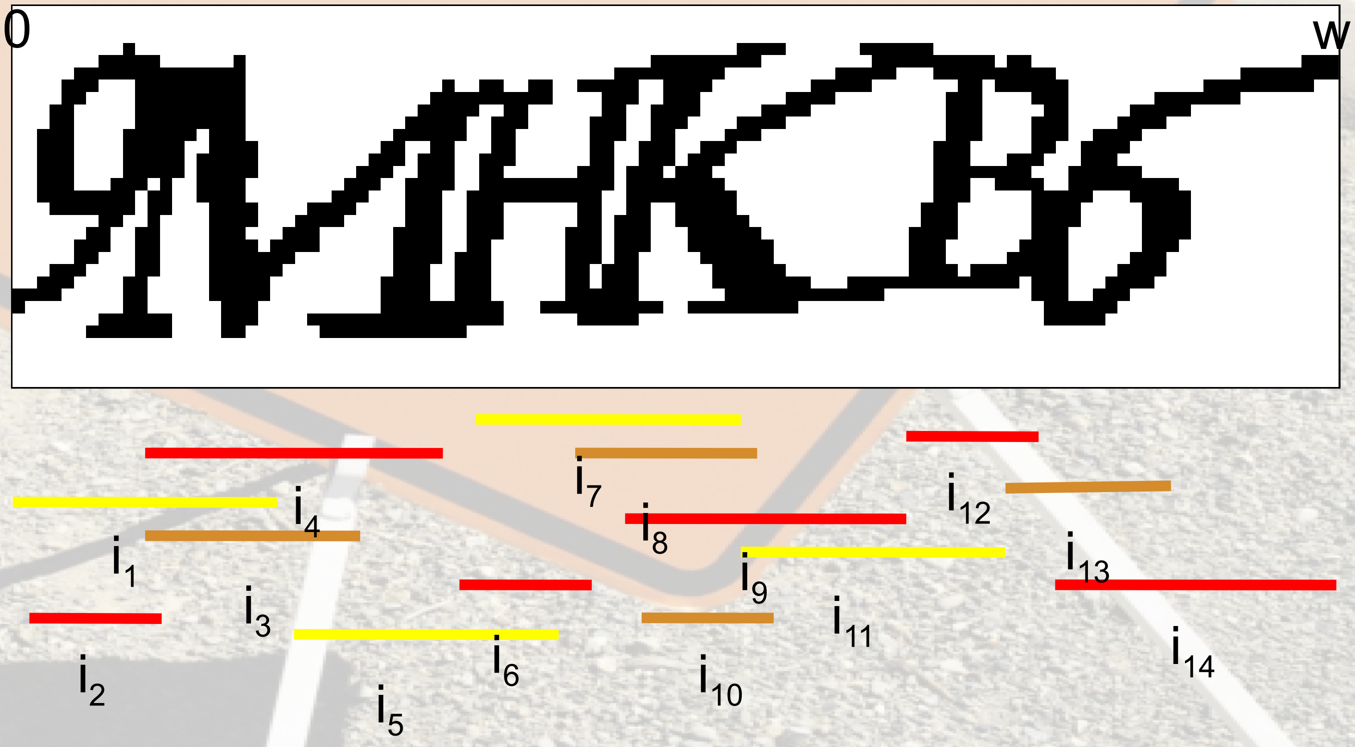
Automatic segmentation

- We would like our algo to detect this:



What we've got…

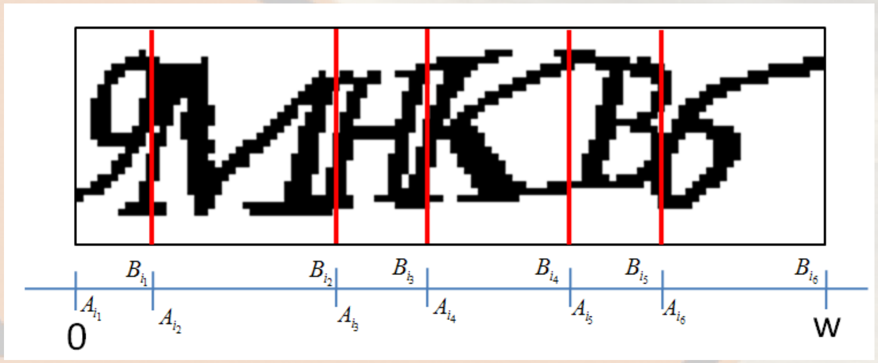
- Preprocessed captcha
- Classifier
  - Prediction on a subwindow of any width
  - Score telling how sure the prediction is



🟥 Good score
🟧 Medium score
🟨 Low score

Formalization

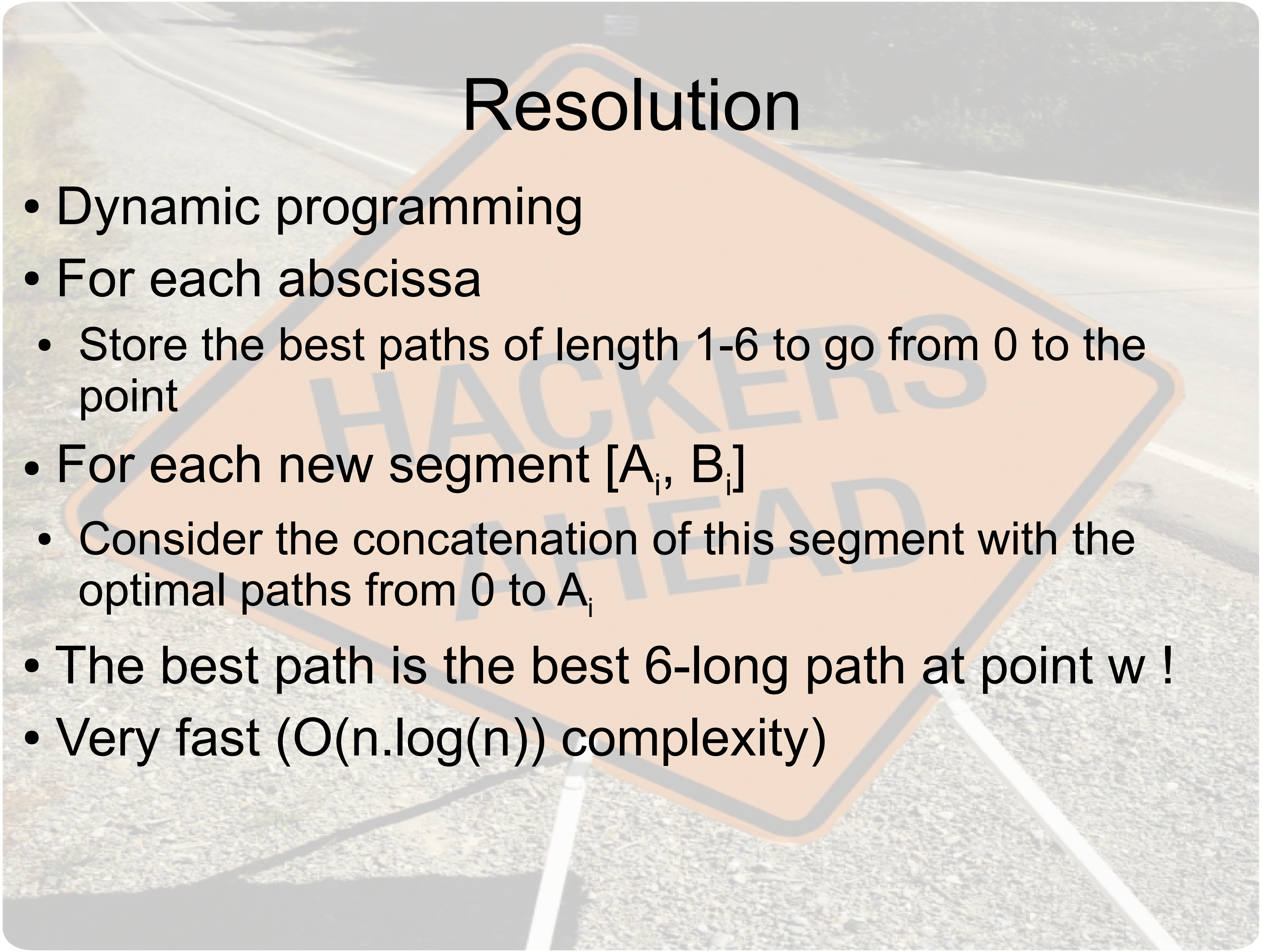
- Optimization of sum (or product) of scores over all letters:



$$\max_{(i_1,i_2,i_3,i_4,i_5,i_6)} S_{i_1} + S_{i_2} + S_{i_3} + S_{i_4} + S_{i_5} + S_{i_6}$$

Given the constraints:

$$\begin{cases} A_{i_1} = 0 \\ B_{i_6} = w \\ \forall\, k \in [1,5]\ B_{i_k} = A_{i_{k+1}} \end{cases}$$

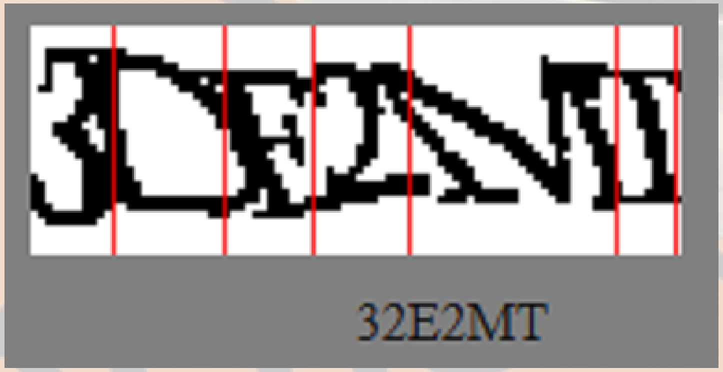


Resolution

- Dynamic programming
- For each abscissa
  - Store the best paths of length 1-6 to go from 0 to the point
- For each new segment $[A_i, B_i]$
  - Consider the concatenation of this segment with the optimal paths from 0 to A_i
- The best path is the best 6-long path at point w !
- Very fast (O(n.log(n)) complexity)

Results

- Not so good ☹



- Very difficult to build a good model with many classes (high score ⇔ clear character)
- Model has to be built with similar data than in the captchas
- Some characters are not well recognized
- Slow (score computation on each subwindow)

Conclusion

- Fun project, with nice applications
- Special thanks to Iasonas Kokkinos
- http://code.google.com/p/captchacker