Comp Photography (Spring 2016) Final Project

Karthiksundar sankaran
ksankaran6@gatech.edu

https://cs6475.wordpress.com/2016-spring/

Captcha Stregth Tester

Test Captcha’s resistance to being read by OCR software using computational Photography techniques

Goal : Captcha Stregth Testing

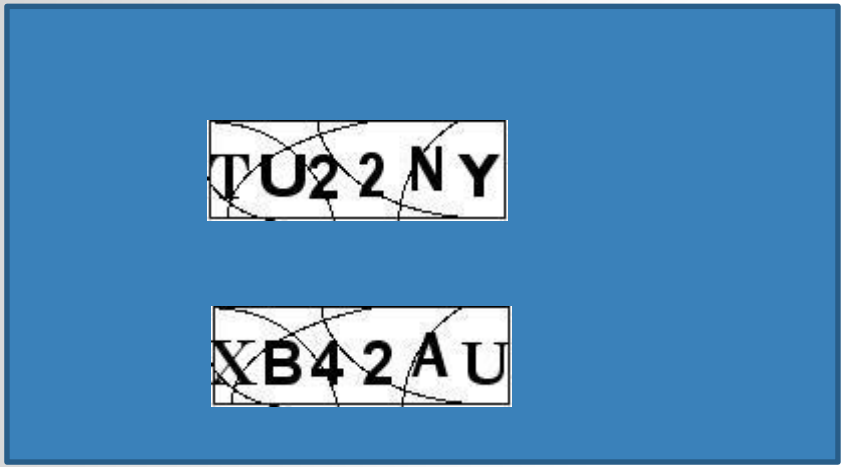
When providing services online one of the biggest problems facing web service providers is the use of automated robots to submit forms. In many cases this is done by unwelcome users for nefarious purposes.

One of the commonly used methods to distinguish between human and robot visitors is the use of Captchas, the goal of this project is to test the strength of captchas against attacks using computational photography techniques to improve OCR accuracy.

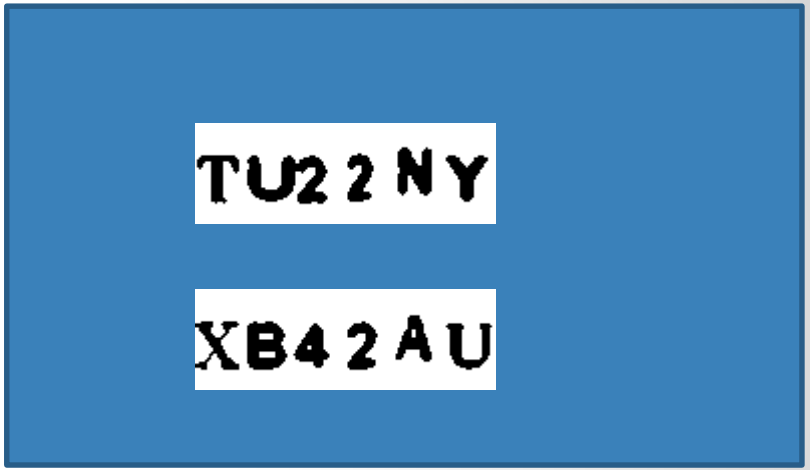
The ultimate goal is to build Captchas that are highly resistant to common computational photography techniques and make OCR difficult. This project is motivated by real need at work where automated robots are causing great deal of financial loss.

Captcha manipulation to make easier reading by OCR sofware

Input

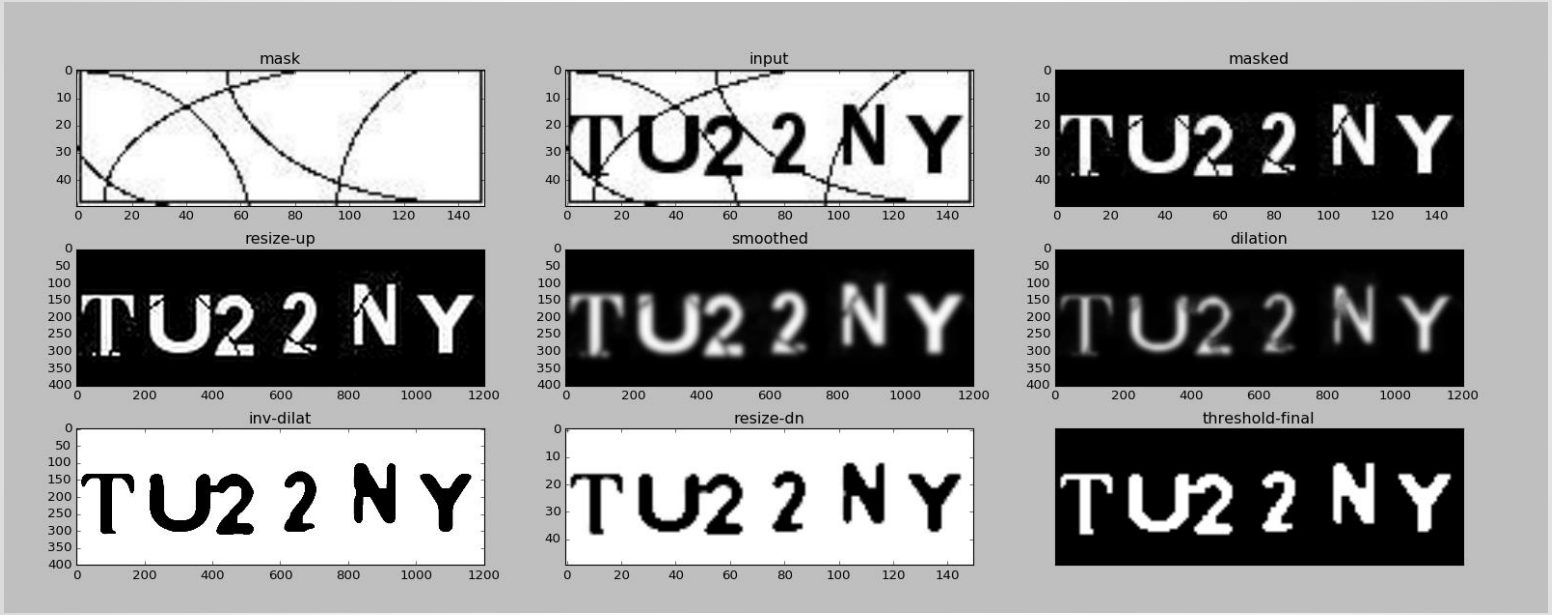


Output



Current pipeline (generic) produces these images and has been overall better with an average success rate of 62% correctly identified letters and 16% success rate on exact match. Which is quiet good considering websites allow multiple tries for Captchas.
Output can be tailored to be much less jagged in the edges by adjusting parameters but the settings used now produced the best success rate overall for all the 50 images I tested with. Some pipeline improvements need to be made to make the image edges smoother while increasing OCR accuracy.

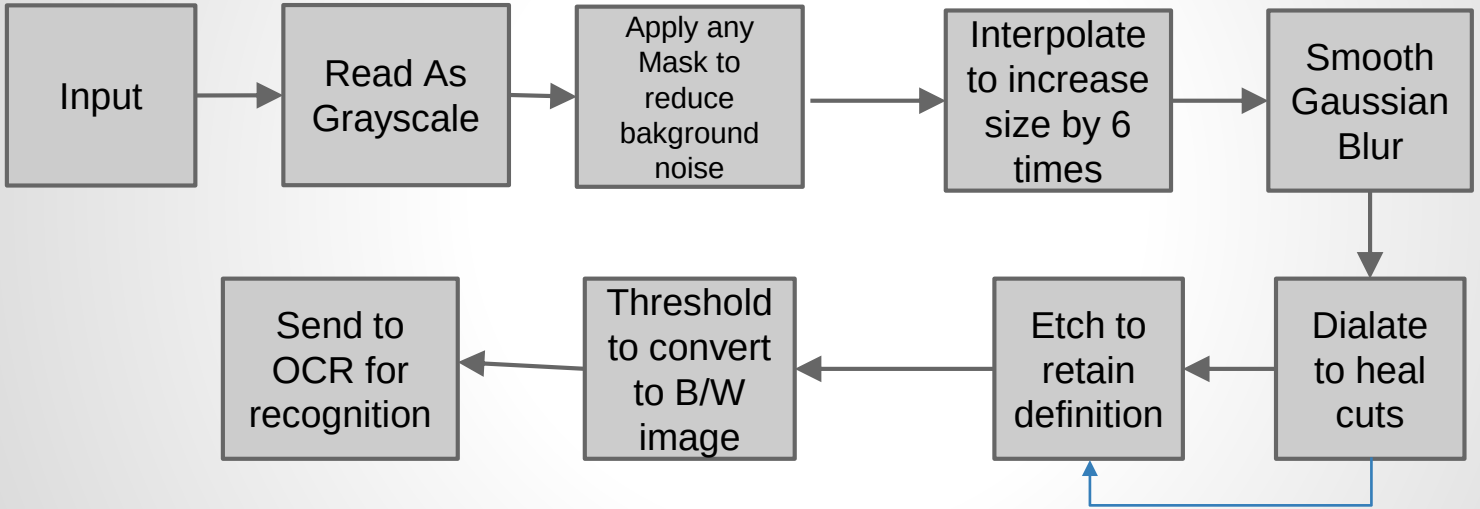
Generic transformations to clarify text



The Challenge

- As part of this project I found that individual captchas can be easily broken by adjusting the pipeline or the different parameters used for masking, smoothing, dilating or etching.
- The Major challenge is to come up with one generic pipeline that can break all captchas.
- Depending on the complexity of the captcha generator what works for one image may not work for another, Visualization using pyplot of the steps helped coming up with a better pipeline.

Showcase your pipeline



What is the best way to see your project?

Code is here:

https://github.com/ksankaran6/captcha

Rest of the details are in the powerpoint.

What worked

- Breaking individual captcha’s was easily done using the techniques learned in this course. In most cases if a single captcha was attacked by specifically tailoring the pipeline – I was able to crack the captcha.
- OCR errors due to background noise introduced were easily eliminated using masking techiques learn in the class.
- Thresholding and other mechanisms eliminated issues due to color variations.
- Small cuts in the images were cured using smoothing (Gaussian and other) techniques to heal the cuts. OpenCV Morphology was also helpful in filling the gaps leading to better OCR results.

What did not work? Why?

1. Coming with a generic pipeline for breaking captchas is still a work in progress.
2. Dialation and errosion tends to smooth out sharp edges leading to issues like S getting identified as 5 etc., I am looking at options like edge detection using something like Canny edge detector to preseve the contours while infilling (healing broken parts).

Captchas Used

Processed Captchas

Results comparison

	A	B	C	D	E
1	image	Direct	PostProcessing - Try1	Manual	Match- Try1
2	captcha1.jpg	---	XB42AU	XB42AU	100%
3	captcha10.jpg	---	ZUJGXW	ZU36XW	67%
4	captcha11.jpg	---	UMZ 6N0	UM26NQ	67%
5	captcha12.jpg	---	GZSJJ A	Gz52JA	67%
6	captcha13.jpg	---	NR33TX	NR88TX	67%
7	captcha14.jpg	---	AYS GZQ	AY86ZQ	67%
8	captcha15.jpg	---	3137121	sj37VJ	0%
9	captcha16.jpg	---	QZBJUT	Qz63UT	67%
10	captcha17.jpg	---	CT34 TP	CT34TP	100%
11	captcha18.jpg	---	GFBSUD	GF88UD	67%
12	captcha19.jpg	---	ZZSBIB	ZZ98MB	50%
13	captcha2.jpg	---	UZB 82 E	UZ88ZE	67%
14	captcha20.jpg	---	Zwsjfis	ZW53AG	33%
15	captcha21.jpg	---	PESJTK	PE53TK	67%
16	captcha22.jpg	---	FJ69NE	FJ69NE	100%
17	captcha23.jpg	---	TU22NY	TU22NY	100%
18	captcha24.jpg	---	DZZTW'H	DZ27Wu	33%
19	captcha25.jpg	---	"(ZQRD	MK29MD	0%

Other captchas tried

A second type of captcha was also tried with similar results as the one shown above.

References / Pointers

tesseract-ocr - https://github.com/tesseract-ocr
PyTesseract - https://pypi.python.org/pypi/pytesseract/0.1
Captchas - From different att.com properties
Robin Davids blog - http://www.robindavid.fr/opencv-tutorial/cracking-basic-captchas-with-opencv.html
https://cdn.elie.net/publications/text-based-captcha-strengths-and-weaknesses.pdf

Team

Karthik Sankaran – Individual project.