



Automatic recognition of handwritten mathematical expressions

Presentazione tesi di laurea triennale in: Ingegneria Informatica

Federico Michelotto



CROHME

Competition on Recognition of Handwritten Mathematical Expressions

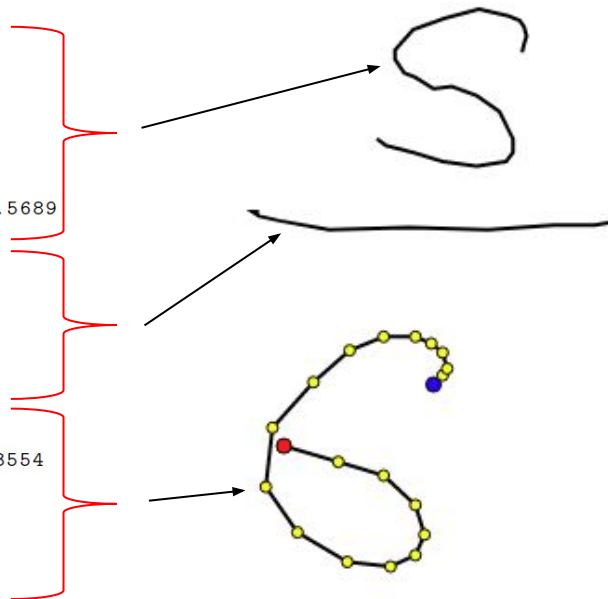
Three main tasks:

- Task 1. Online Handwritten Formula Recognition
- **Task 2. Offline Handwritten Formula Recognition**
- Task 3. Detection of Formulas in Document Pages

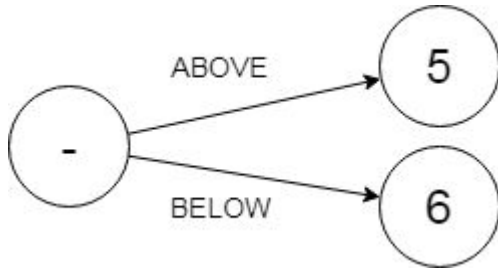
Online vs offline

(x,y)

```
<trace id="0">
11.6603 15.4164, 11.6643 15.4004, 11.6683 15.3883, 11.6643
15.3723, 11.6563 15.3603, 11.6362 15.3522, 11.5841 15.3402,
11.5239 15.3562, 11.4677 15.3763, 11.4356 15.4124, 11.4356
15.4285, 11.4517 15.4525, 11.4717 15.4606, 11.5038 15.4806,
11.5359 15.4766, 11.5801 15.4927, 11.6202 15.5208, 11.6443
15.5689, 11.6443 15.593, 11.6322 15.609, 11.5801 15.6171,
11.5199 15.609, 11.4677 15.593, 11.4196 15.5809, 11.4035 15.5689
</trace>
<trace id="1">
11.8248 15.7174, 11.8128 15.7174, 11.7887 15.7214, 11.7165
15.7214, 11.6001 15.7294, 11.4597 15.7254, 11.3192 15.7294,
11.231 15.7134, 11.1949 15.7053, 11.1828 15.6973, 11.1949
15.6973
</trace>
<trace id="2">
11.5038 16.0023, 11.5199 15.9862, 11.5279 15.9742, 11.5199
15.9461, 11.4998 15.93, 11.4717 15.918, 11.4155 15.918, 11.3554
15.9421, 11.2912 15.9982, 11.2189 16.0785, 11.2069 16.1828,
11.2631 16.2631, 11.3513 16.3152, 11.4276 16.3233, 11.4717
16.3032, 11.4878 16.2671, 11.4717 16.2149, 11.4155 16.1628,
11.3353 16.1387, 11.239 16.1106
</trace>
```



Label graph representation of $\frac{5}{6}$



Nodes :

N, 0, 5, 1.0

N, 1, -, 1.0

N, 2, 6, 1.0

Edges :

E, 1, 0, A, 1.0

E, 1, 2, B, 1.0

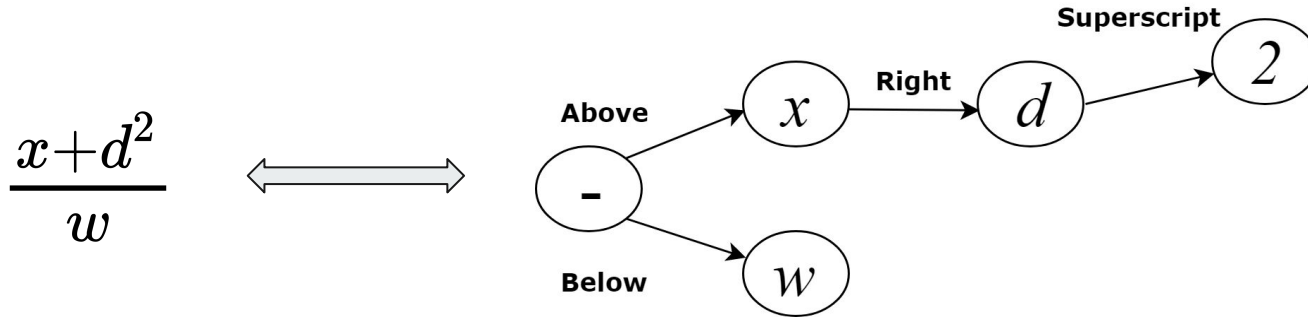
Above


Below

To represent the label graph is used the .lg (label graph)format.

Six spatial relationships are defined in the CROHME competition: *Right*, *Above*, *Below*, *Inside* (for square root) , *Superscript* and *Subscript*.

Every LaTeX expression can be translated into a label graph format and vice versa.





The goal of this project is to create a system using deep learning techniques, able to recognize offline handwritten mathematical expressions (MEs) and translate them into LaTeX.

(LaTeX expression)

$$\sin^2(x) + \cos^2(x) = 1 \quad \Longrightarrow \quad \backslash sin^{2}(x) + \backslash cos^{2}(x) = 1$$

↓

$$\sin^2(x) + \cos^2(x) = 1$$



Dataset

- 9993 handwritten MEs in the training set
- 986 in the validation set
- 1199 in the test set

In addition to the complete expressions, the competition provides a dataset of individual symbols, in this set there are also 'junk' symbols (symbols that do not belongs to any class):

- 180440 (82150 junk) symbols in the training set
- 18435 (8416 junk) symbols in the validation set
- 15483 in the test set



First (naive) approach

- Train a CNN (convolutional neural network) classifier to recognize if any particular image is a symbol or not, and in case identify which symbol is related to the image
- Apply a sliding window algorithm to pass the whole image through a “*window*” with different aspect ratio and size. Classify each candidate region
- Build an automatic system that translate the symbols detected in a LaTeX equation

Initial results

- 92% without junk
- 83% with junk

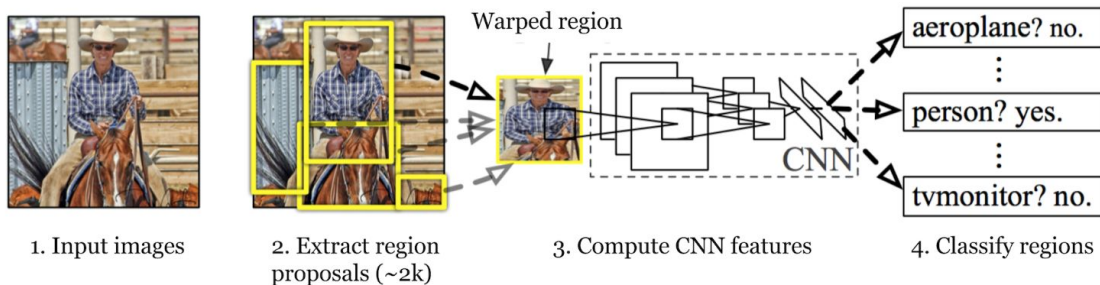


Problems

- The sliding window approach is slow: to “cover” the whole image thousands of forward pass are required for each image.
- Complex to reconstruct all the symbols correctly.
- Special problems for the square root symbol: in the training set there are no symbols inside...

Alternative to sliding window: R-CNN

R-CNN (Region Convolutional Neural Networks) proposed by Girshick et al. [1], 2013, uses an external algorithm (Selective Search) that select the regions with some relevance instead to classify all the regions provided by a brute force sliding window algorithm.



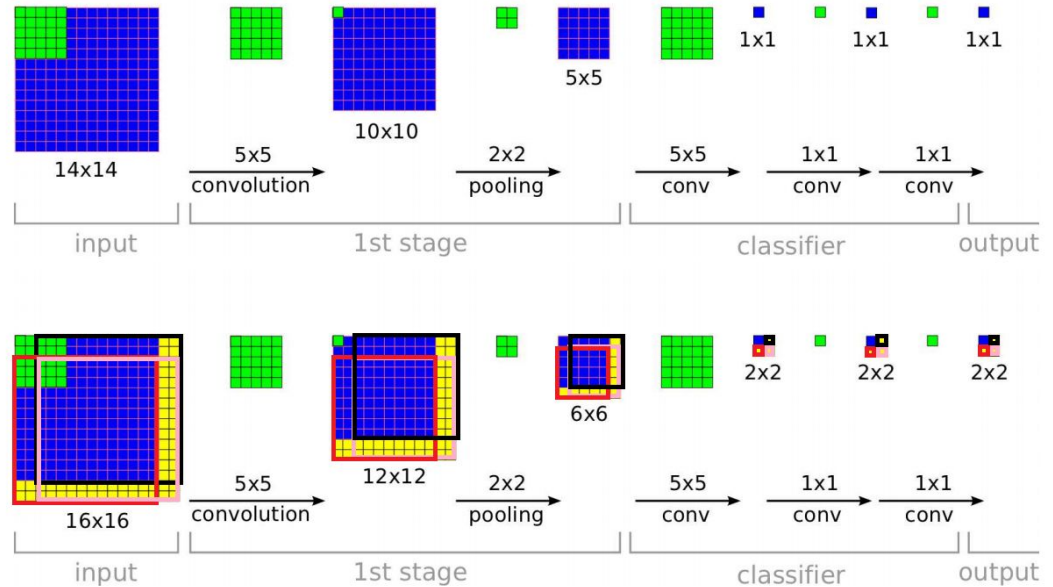


Problems of R-CNN

- Slow: still a lot of proposals (~2000)
- Inefficient: a full forward pass for each region
- The images must be resized to pass through the CNN.

Overfeat

Sermanet et al. [2], 2013, show how and why ConvNets can be used efficiently to implement both a multiscale and a sliding window network.



Consider the following CNN, and a fixed window of 14x14, the image below show how there is no need to compute the forward pass for every window, but it's enough to compute the forward pass only once on the entire image. The size of the output in this case is equal to 4, because 4 windows fits in the input image.



Fast R-CNN

Evolution of the first version published by Girshick et .al [3] in 2015 that solves the problem of the “one forward pass for each proposal” exploiting the properties of the convolutional neural network shown in the Overfeat paper.

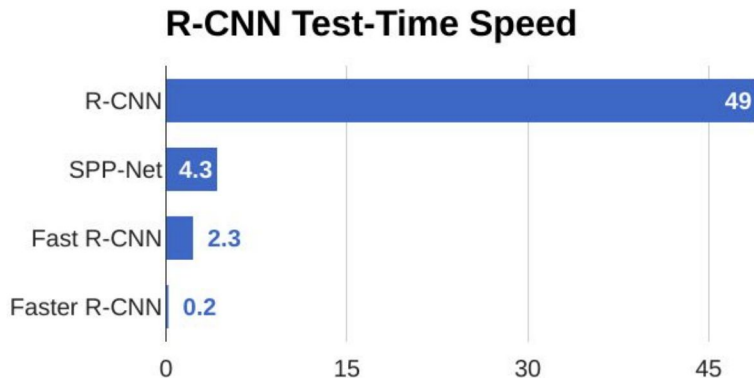
	R-CNN	Fast R-CNN
Test time per image	47 seconds	0.32 seconds
(Speedup)	1x	146x
Test time per image with Selective Search	50 seconds	2 seconds
(Speedup)	1x	25x

Bottleneck:
Selective Search



Faster R-CNN

In Faster R-CNN (Ren et al. [4], 2015) has been inserted a region proposal network (RPN) to substitute the external proposal algorithm used in the previous versions. The RPN predicts regions from feature map (one per image). Then is the same as Fast R-CNN.





New approach

- Use the Faster R-CNN model to train the detector
- Build an automatic system that translate the symbols detected in a LaTeX equation

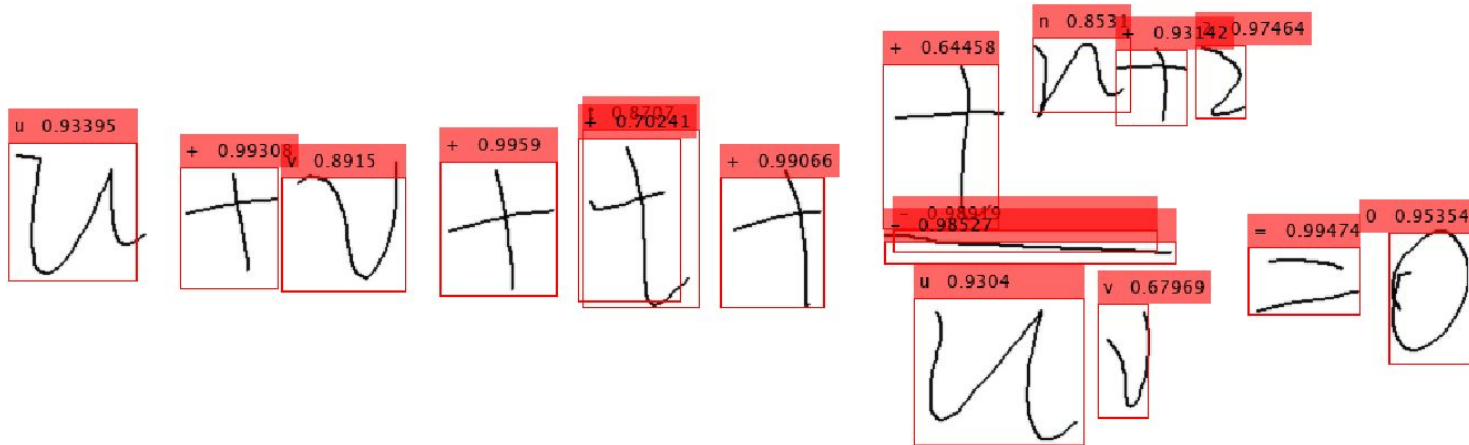
To train the Faster R-CNN model, ground-truth bounding boxes with the corresponding labels must be extracted from the images of the complete MEs.

This can be done parsing the online data of every ME.

Detector

The detector has been trained on the ConvNet ResNet-50 using the 9993 images in the training set.

A first glimpse of how the detector works:





LaTeX translation

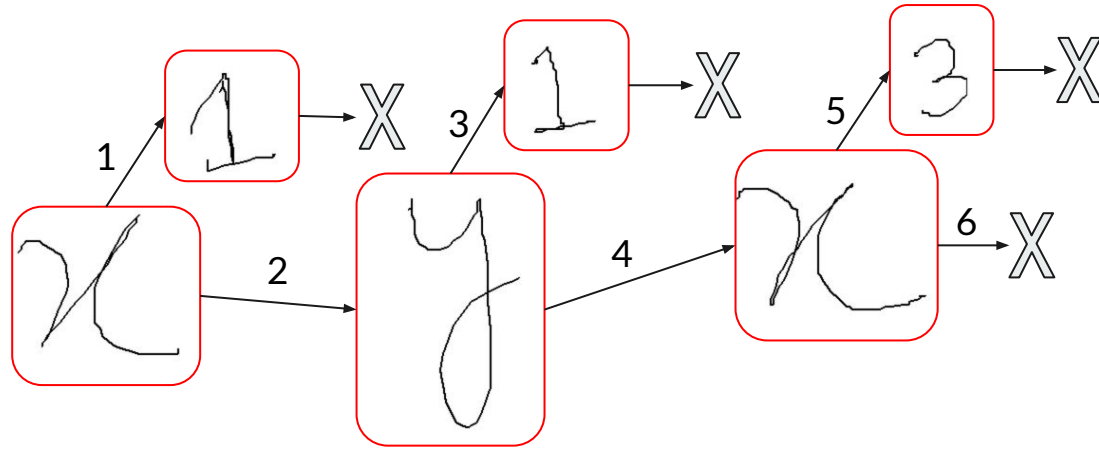
As a first approach I decided to develop an heuristic method based on a recursive algorithm.

The method consists to parse the symbols detected from left to right, giving precedence to the other edge relationships (above, below, subscript, superscript, inside).

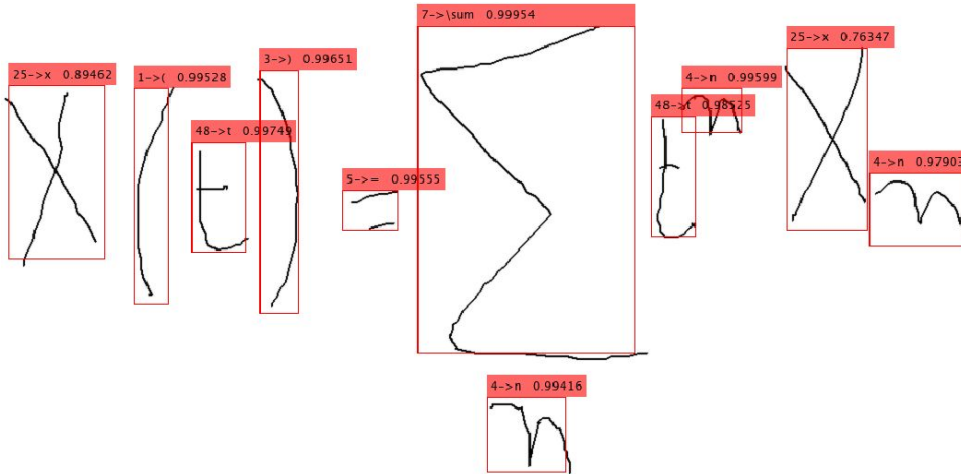
The recursive algorithm exit when there are no others right symbols to parse, this condition is limited by a fixed distance between the symbols when the symbol in question is not at level 0 of the expression.

The symbols used are deleted from the list of available symbols, this list will be returned together with the result of the partial expression.

example

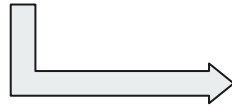
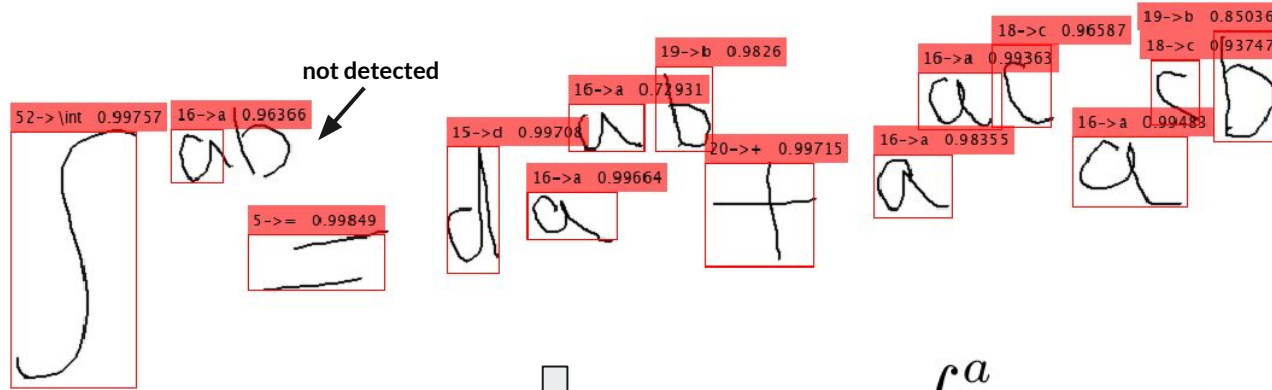


Final system examples (1)



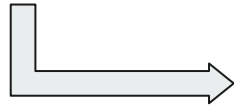
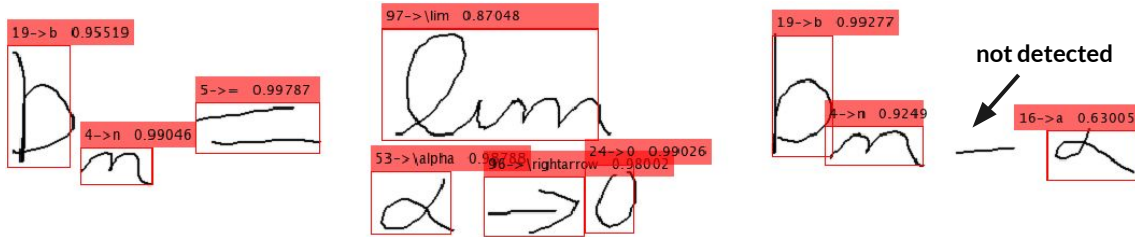
$$x(t) = \sum_n t^n x_n$$

Final system examples (2)



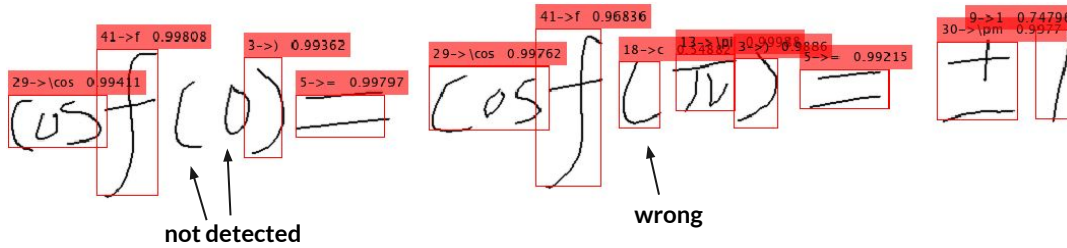
$$\int^a = da^{ab} + a^{ac}a^{cb}$$

Final system examples (3)



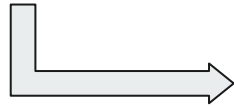
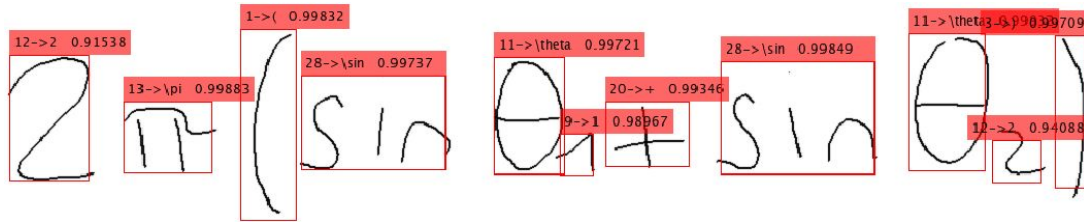
$$b_n = \lim_{\alpha \rightarrow 0} b_n$$

Final system examples (4)



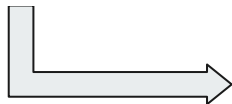
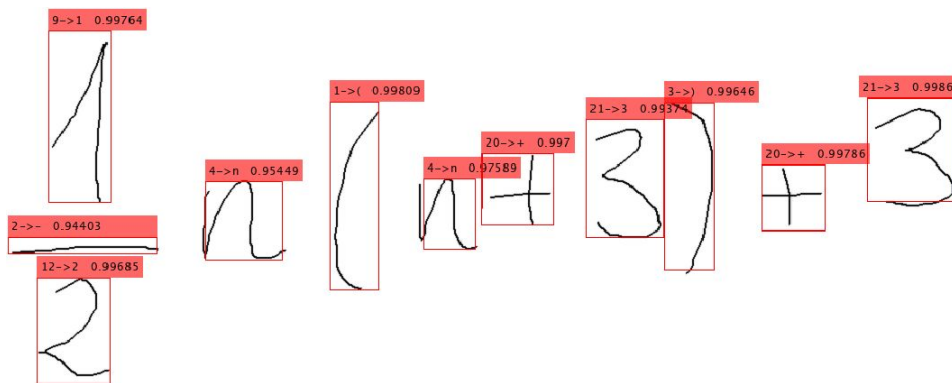
$$\cos f) = \cos f(c\pi) = \pm 1$$

Final system examples (5)



$$2\pi(\sin \theta_1 + \sin \theta_2)$$

Final system examples (6)



$$\frac{1}{2}n(n+3)+3$$



Future work

- Test the accuracy of the system
- Build a translation system using deep learning techniques



Let's try