UNIVERSITÀ DI PAVIA

Recursive Recognition of Offline Handwritten Mathematical Expressions

Marco Cotogni Claudio Cusano Antonino Nocera

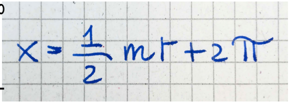
marco.cotogni01@universitadipavia.it, claudio.cusano@unipv.it, antonino.nocera@unipv.it

25th International Conference on Pattern Recognition

Recognition of Offline Handwritten Math Expressions

Given an image depicting a mathematical expression decode it in a symbolic representation (e.g. in the LaTeX language)

Input image



`x = \frac{1}{2} mt + 2 \pi`

Output symbols

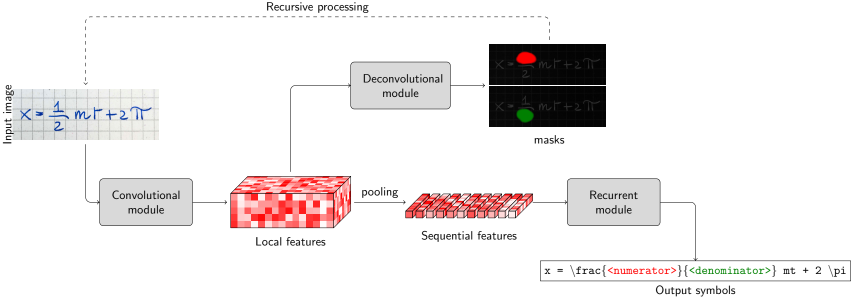
The trajectory of the pen is not available (offline recognition)

More challenging that conventional OCR!

- non-sequential spatial layout (e.g. fractions)
- little prior information encoded in language models, and dictionaries

Recursive recognition

Our solution is designed to enable fast processing even on devices with limited computational resources (e.g. mobile devices)



- a CNN extracts image features
- a RNN translates features into symbols
- a deconvolutional module is used to identify complex subexpressions that are processed recursively

Convolutional module

Operation	Param.	Output size
Input	-	$W \times 160 \times 1$
Conv. 7×7, st. 1×1	1600	$W \times 160 \times 32$
Batch Norm. + ReLU	64	$W \times 160 \times 32$
Max Pooling 4×4	-	$(W/4) \times 40 \times 32$
Conv. 3×3, st. 1×1	18 496	$(W/4) \times 40 \times 64$
Batch Norm. + ReLU	128	$(W/4) \times 40 \times 64$
Conv. 3×3, st. 1×1	36 928	$(W/4) \times 40 \times 64$
Batch Norm. + ReLU	128	$(W/4) \times 40 \times 64$
Conv. 3×3, st. 2×2	36 928	$(W/8) \times 20 \times 64$
Batch Norm. + ReLU	128	$(W/8) \times 20 \times 64$
Conv. 3×3, st. 1×1	73 856	$(W/8) \times 20 \times 128$
Batch Norm. + ReLU	256	$(W/8) \times 20 \times 128$
Conv. 3×3, st. 1×1	147 584	$(W/8) \times 20 \times 128$
Batch Norm. + ReLU	256	$(W/8) \times 20 \times 128$
Conv. 3×3, st. 1×2	147 584	$(W/8) \times 10 \times 128$
Batch Norm. + ReLU	256	$(W/8) \times 10 \times 128$
Conv. 3×3, st. 1×1	295 168	$(W/8) \times 10 \times 256$
Batch Norm. + ReLU	512	$(W/8) \times 10 \times 256$
Conv. 3×3, st. 1×1	590 080	$(W/8) \times 10 \times 256$
Batch Norm. + ReLU	512	$(W/8) \times 10 \times 256$
Conv. 3×3, st. 1×2	590 080	$(W/8) \times 5 \times 256$
Batch Norm. + ReLU	512	$(W/8) \times 5 \times 256$

Recurrent module

Operation	Param.	Output size
Input	-	$(W/8) \times 5 \times 256$
Max pool 1×5	-	$(W/8) \times 256$
LSTM cell	5 251 072	$(W/8) \times 1024$
Linear	10 250	$(W/8) \times 100$
Softmax	-	$(W/8) \times 100$

Deconvolutional module

Operation	Param.	Output size
Input	-	$(W/8) \times 5 \times 256$
Conv. 3×3, st. 1×1	295 040	$(W/8) \times 5 \times 128$
Batch Norm. + ReLU	256	$(W/8) \times 5 \times 128$
Upsampling 2×8	-	$(W/4) \times 40 \times 128$
Conv. 3×3, st. 1×1	2306	$(W/4) \times 40 \times 2$
Sigmoid	-	$(W/4) \times 40 \times 2$
Upsampling 4×4	-	$W \times 160 \times 2$

End-to-end trainable with off-the-shelf algorithms for the minimization of the Connectionist Temporal Classification loss (CTC)

Data

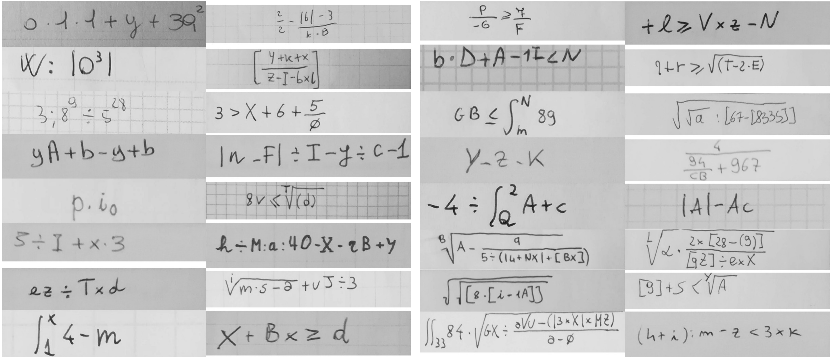
We collected images of 9100 expressions including 99 different symbols and tokens

- the digits $0 \ldots 9$
- the English letters a…z and A…Z
- the Greek letters α, β, γ, ϵ, ϕ, λ, μ and π;
- the arithmetic operators $+$, $-$, $\times$, $\cdot$ and $\div$
- the relational operators $<$, $>$, $\leqslant$, $\geqslant$, $=$
- the parenthesis $($, $)$, $[$, $]$ and $|$
- the punctuation symbols . , ; and :
- the integral symbol $\int$
- the LaTeX tokens `\frac`, `\sqrt`, _ and ^

8300 training images, 400 validation, 400 test

Data

Expressions were randomly generated according to a grammar and handwritten by more than 100 volunteers



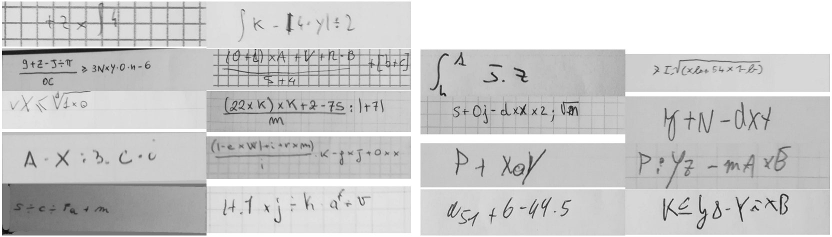
Results

We obtained an average Levenshtein distance of 0.691 between expected and output sequences of symbols

Category	Accuracy (%)
Digits	90.7
English letters	95.2
Greek letters	95.0
Operators	96.7
Parentheses	94.6
Punctuation	96.4
Fractions	98.4
Roots	94.7
Integrals	100.0
Subscripts	96.2
Superscripts	84.2

Results

Errors are most common for bad-quality images and for expressions written in a style that is generally hard to read



Conclusions

We presented a method for the recognition of handwritten mathematical expressions

Thanks to its recursive definition, the method is fast and accurate

It allows to recognize complex expressions in a resonable time even when computational resources are scarce, which is the case of smartphones and other mobile devices