Handwritten Mathematical Expression Recognition

Heqing Ya, Yulan Huang

Carnegie Mellon University, Language Technology Institute

ABSTRACT

Recognizing handwritten mathematical expression is a challenging scientific task. It requires to combine both studies of character recognition and structure learning, facilitating one result based on another. In this project, we define our study in a controlled and increasing scope, enhancing our algorithm accordingly.

Motivation: To offer convenience for homework grading, CMU students are required to waste a lot of time convert formula derived by hand into typesetting format. To reduce this meaningless and arduous task, this project develops an algorithm that automatically convert the handwritten mathematical expression into LaTeX format.

Limitations: **(1)** sparse data (no existing large handwritten mathematical corpus), **(2)** no clear scope of mathematical expression

Objectives: **(1)** Figure out how to separate each character from an image of math expression **(2)** Figure out how to retrieve structural information from the separated image **(3)** Figure out how to recognize each character in a separated image **(4)** Figure out how to jointly utilize the structural information and character recognition information

SCOPE DEFINITION

Scope I: Single Character Recognition

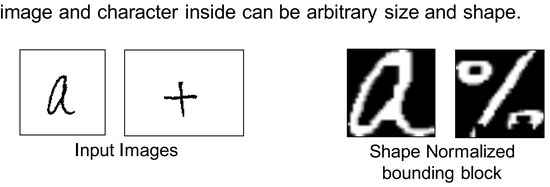
•**Input:** Image with only one character (Left 2 Images bellow)

•**Output:** **(1)** Bounding block of the character (Right 2 Images bellow) **(2)** Recognized character or character probability distribution

•**Requirements:**

The algorithm should handle the 127 characters defined as our target vocabulary.

The input image and character inside can be arbitrary size and shape.



Input Images

Shape Normalized bounding block

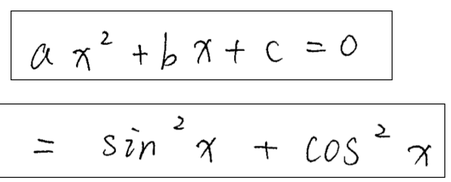
Scope II: Single Line Math Expression Recognition

•**Input:** Image with single line expression

•**Output:** **(1)** Bounding block of each characters and the expression structure **(2)** Recognized expression in LaTeX form **Requirements:**

The algorithm should cover scope I.

The input should cover subscript and superscript.



Input Images

Scope II: Multi-Line Math Expression Recognition

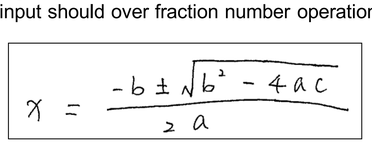
•**Input:** Image with multi-line expression

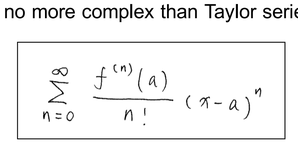
•**Output:** **(1)** Bounding block of the characters **(2)** Recognized expression

•**Requirements:**

The algorithm should cover scope II.

The input should over fraction number operation but no more complex than Taylor series.





Input Images

CHARACTER GROUPING

- **Target**: Group separate parts belonging to the same symbols , such as '%', '=', 'I', '≥', '≤', etc.
- **Input:** A list of connected components with their bounding boxes
- **Approach: (1)** Sort all the connected components according to Z-order **(2)** Iterate through all connected components, combine potentially related components, put them together and predict with CNN, group them if it the largest confidence value is higher than a threshold.

CHARACTER RECOGNITION

Convolutional Neural Net

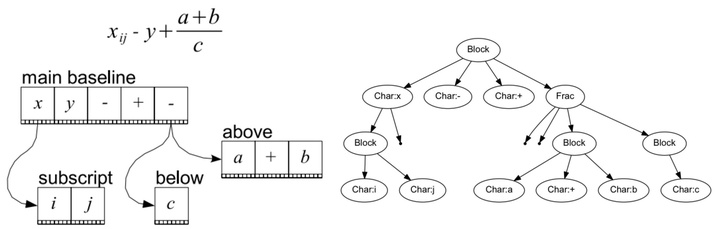
- **Model:** A CNN with 2 alternate convolution and pooling layers, 1 hidden fully connected sigmoid layer as output.

K-Nearest Neighbor

- **Model:** A KNN classifier with image intensity as input (k=127), used as baseline.

STRUCTURE ORGANIZATION

- **Input:** A list of symbols with their bounding boxes
- **Approach:** Parse structure recursively, reducing all the expressions into Single Line Math Expression with only subscript and superscript: **(1)** Find all vinculums, parse all symbols above as numerator, and symbols below as denominator. **(2)** Find all radical signs, for each radical sign, parse all symbols whose bounding boxes are overlapped with the radical sign as radicand. **(3)** Find all Σ and π, parse all the symbols above as superscript and all the symbols below as subscript. **(4)** Calculate an baseline, determine superscripts and subscripts.



Baseline Separation Method and Expression Tree Data Structure

CHARACTER STRUCTURE FUSION

- **Input: (1)** Probability distributions for each characters from CNN output; **(2)** Character sequences organized along Baselines.

Voting Approach

- **Approach: (1)** Define evidence rule for each character. **(2)** Determine candidate characters based on the input probability distribution. **(3)** Iteratively count the evidence rule, determine and update the candidate characters.

Bayesian Approach

- **Approach: (1)** Define 7 classes of the characters **(2)** Define an HMM-like model by assuming each character is based on previous characters. Manually define the transition and defined the emission based on CNN-output. **(3)** Decode the sequence using Viterbi-like algorithm and get the most-likely character in the decoded states.

DATA COLLECTION

- **Mathematical Characters:**
- 127 Characters, 911 hand-written samples for each, scanned and extracted based on the bounding box, and reshaped into the same size 32px x32px

- **Mathematical Expressions:**
- 20 Single-Line Math Expressions and 20 Multi-line Math Expressions

EXPERIMENT RESULTS

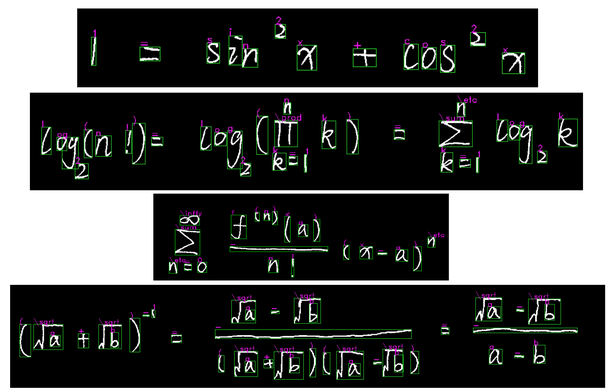
Stage I: Character Recognition Result

	KNN	CNN
Average error	0.2594	0.1245

Truth	Predict	Error Rate	Truth	Predict	Error Rate
C	c	0.66	W	W	0.32
1	|	0.44	0	O	0.30
S	s	0.42	z	Z	0.30
O	0	0.36	X	×	0.28
o	0	0.36	Z	z	0.27

TOP 10 Error Case from CNN

Stage II, III: Character Recognition in Math Expression Result



Test Images

- **Results with Voting:**
- 1=sin^{2}x+cos^{2}x
- log_{2}(n!)=log_{2}(\prod_{k=1}^{n}k)=\sum_{k=1}^{\eta}log_{2}k
- \sum^{\infty}_{\eta=0}\frac{f^{(n)}<a)}{n!}(x-a)^{\eta}
- (\sqrt{a}+\sqrt{b})^{-1}=\frac{\sqrt{a}-\sqrt{b}}{(\sqrt{a}+\sqrt{b})(\sqrt{a}-\sqrt{b})}=\frac{\sqrt{a}-\sqrt{b}}{a-b}

Discussion

(1) CNN is able to produce high recognition accuracy with errors for similar looking characters **(2)** We can further increase the prediction accuracy by considering context of the character in the math expression **(3)** Without large amount of data, Bayesian model is hard to estimate ☹