

Some thoughts about arranging the locations to generate math expression tree

The segmentation code (with classifier) returns the following data:

1. x (it may also be recognized as a “times” \* symbol)
2. dash (it does not know whether it should be a minus sign(-), or fraction line, or even part of a equal sign(=))
3. y

The above example shows that even a well-trained classifier(or human eyes) CANNOT recognize each single part of this equation solely. Another problem to solve is that we got 3 bounding rects, but how is their order? How to use all of these 3 rects to generate math expression tree?

A RNN approach

(\*\*\*\*First get a description of RNN and LSTM)

What input and output we finally want using RNN (let’s first take a look at the ideal results of rnn on the above equation):

We start from the upper corner rect (which contains x)

x1,y1,x2,y2 is the bounding rect

let [bounding rect] be a vector: (x1, y1, x2, y2)

|  |  |
| --- | --- |
| X (input) | Y (output) |
| [bounding rect: 0] literal: x | [bounding rect: 1] operator: frac |
| [bounding rect: 1] operator: frac | [bounding rect: 2] literal: y |

Then we got a sequence x->frac->y

Writing a parser of the above sequence, we want get frac(x,y) (x, y are the children of frac)