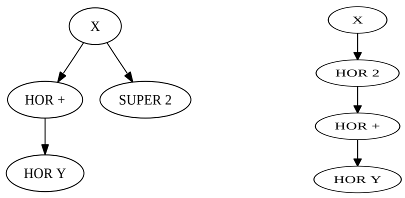
# Intelligent Combination of Structural Analysis Algorithms: Application to Mathematical Expression Recognition

Amit Pillay and Richard Zanibbi
Document and Pattern Recognition Lab, Rochester Institute of Technology, NY, USA
{ aap2731, rlaz}@cs.rit.edu

Although many researchers develop several different approaches for mathematical expression recognition[2], there is a limit in improving any single approach again and again. Hence combination of recognition algorithms [1] can be adopted for possible further improvement in recognition.

The goal of the research is to develop a machine learning algorithm for combining structure recognition algorithms to improve recognition performance and apply this algorithm on mathematical expressions. As part of this effort Recognition Strategy Language (RSL) [4] will be modified to support algorithm combination and evaluation.



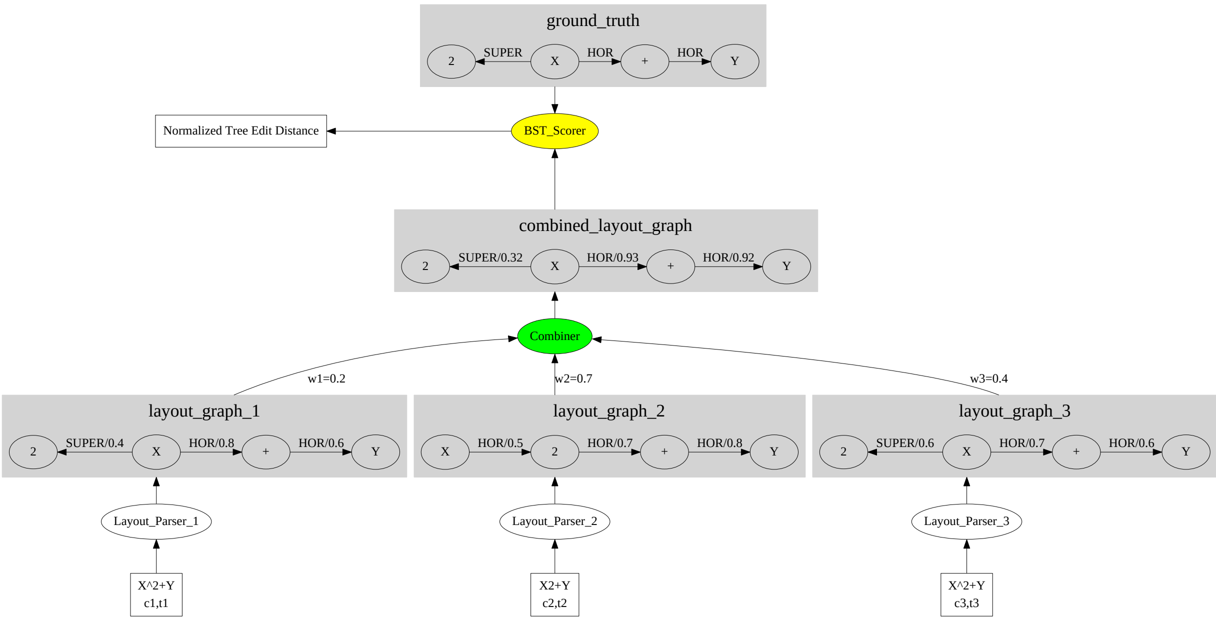

(a) Tree1 (b) Tree2

Figure 1: Two Baseline Structure Trees Representing Interpretations of $X^2 + Y$

We need to compute the tree edit distance between two Baseline Structure Trees (BST) [5] as shown in Figure 1(a) and Figure 1(b) representing two interpretations of a mathematical expression having matching symbols but different spatial relationships between them. The expressions are converted into directed trees having nodes with values as a symbol in given expression concatenated with the type of spatial relationship which the symbol have with previous symbol as shown in Figures 1(a) and 1(b). To cope with the repeated symbols in the expression symbols needs to be numbered to identify them uniquely. Tree edit distance is then given as the number of spatial relationship mismatches between nodes of two trees plus the number of mismatching parents of nodes in two trees. In the given example, Tree2 shows the incorrect interpretation of the expression $X^2 + Y$. The tree edit distance between Figures 1(a) and 1(b) is 2. One is due to symbol "2" in Tree2 having a mismatched spatial association with its parent "X" (i.e HOR instead of SUPER as in Tree1) and one due to symbol "+" having mismatching parent (have parent as "2" instead of "X" as in Tree1). Edit distance is then normalized as number of edits / 2n(where n is the number of symbols); 2n is the maximum edit distance, where every symbol has the wrong parent, and in the wrong spatial relationship.

Our machine learning architecture uses a Graph Transformer Network (GTN) [6] consisting of modules that communicate their states and outputs in the form of trees whose arcs carry numerical information(see Figure 2). Learning in our GTN takes places using standard back-propagation for neural networks [3].

- Inputs in the form of list of symbols in a mathematical expression and their bounding boxes coordinates are given to different layout parsers (current research uses DRACULAE parser [5]). In addition, DRACULAE takes two parameters i.e. centroid parameter which is a ratio used to define the vertical location of symbol centroid and region threshold for different regions around the symbol. Each of these parsers performs the Baseline Structure Tree (BST) extraction from given set of symbols and their bounding boxes' coordinates. As shown in Figure 2, three parsers generates BSTs for the expression $X^2 + Y$. Each symbol is represented by a node with a spatial association with the parent symbol such as HOR for horizontal adjacency or other spatial associations (SUPER, SUBSC, ABOVE, BELOW, ULIMIT, LLIMIT and CONTAINS). Spatial associations are accompanied with a penalty for that association.




Combining Parsers Using Graph Transformer Network

Figure 2: GTN. Tree edit distance between ground truth and combined BST is 0

- Each of the BSTs generated is given to another module called the Combiner module. The connections between parsers and the Combiner module are accompanied by a weight as can be seen in Figure 2. The weights are defined for the respective parser output as a whole. The Combiner performs the linear combination of weighted penalties of spatial relationships between two symbols produced by different parsers. For example, in the given Figure 2, linear combination of weighted penalties is performed for SUPER association between symbols X and 2 produced by parser1 and parser3. The minimum penalty BST is produced by applying Prim's algorithm for finding minimum spanning trees.

- BST generated by the Combiner module is given to another module called BST Scorer module which computes an error in the form of normalized tree edit distance between the combined BST and the ground truth as described above. This distance act as the loss function for the current pattern. The gradient at the Combiner output is back propagated to calculate the gradient of the loss function w.r.t the weights for the parsers. Then for each parser we obtain gradient of the loss function with respect to the parser's parameter vector.

For the experiment number of typeset and handwritten ground truth expressions will be generated. The GTN will be trained for combining 2-3 DRACULAE parsers and the recognition results will be analyzed.

# References


[1] Brunet-Manquat and Francis. Syntactic Parser Combination For Improved Dependency Analysis. *Workshop On Robust Methods In Analysis Of Natural Language Data ROMAND*, 2004.

[2] K.F. Chan and D.Y. Yeung. Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition*, 3(1):3–15, 2000.

[3] Rumelhart D.E, Hinton G.E, and Williams R.J. Learning internal representations by error propagation. *Parallel Distributed Processing: Exploring in the Microstructure of Cognition*, 1:318–362, 1986.

[4] Zanibbi R, Dorothea Blostein, and Cordy J. R. Decision-Based Specification and Comparison of Table Recognition Algorithms. *Machine Learning in Document Analysis and Recognition, Eds. S. Marinai and H. Fujisawa*, 90:71–103, 2008.

[5] Zanibbi. R, Blostein. D., and J.R Cordy. Recognizing mathematical expressions using tree transformation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1455–1467, 2002.

[6] Lecun Y, Bottou L, Bengio Y, and Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.