Design of an algebraic recognition engine

cleverlearn

CentraleSupélec

Group 5 Composition

CentraleSupélec

- Abderrahmane Dkour
- Lucas Fernandes Martins
- João Pedro Regazzi Ferreira Da Silva
- Gabriel Souza Lima
- Salma Maazoum

cleverlearn

- Ibrahim Rakib (Founder/Tech)
- Baptiste Vélon (Co-founder/Finance)
- Romain Beutel (Product)

Table of content

1. Introduction

Who is Cleverlearn?

Make learning more **effective**, **engaging** and **tailored** to individual needs

The project

Problem

- Subjectivity
- Bias
- Time-consuming
- Error-prone

Solution

The aim of this project is to develop a platform for assessing the accuracy of students' answers in a automatic way.

- Faster
- Consistent
- More objective
- More efficient.

Quantification of the proximity of two algebraic expressions

Simple recognition
Simplified expressions

$$x^2 + 2x - 1 = 2 \times x + x^2 - 1$$

$$\frac{hk}{cm} = \frac{kh}{m \times c} = \frac{k}{m} \times \frac{h}{c}$$

Semantic recognition
Equivalent expressions

$$3x = 2x + x$$

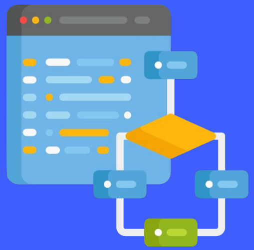
$$\frac{Ee^2}{er} = \frac{Ee}{r}$$

Simple recognition
Simplified expressions

$$\sqrt{\frac{gh}{m}} \sim \sqrt{\frac{h}{mg}}$$

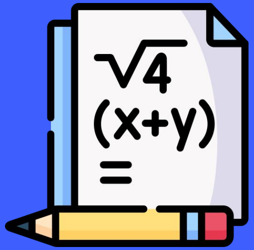
Steps

Algorithmic approach



Dataset generation

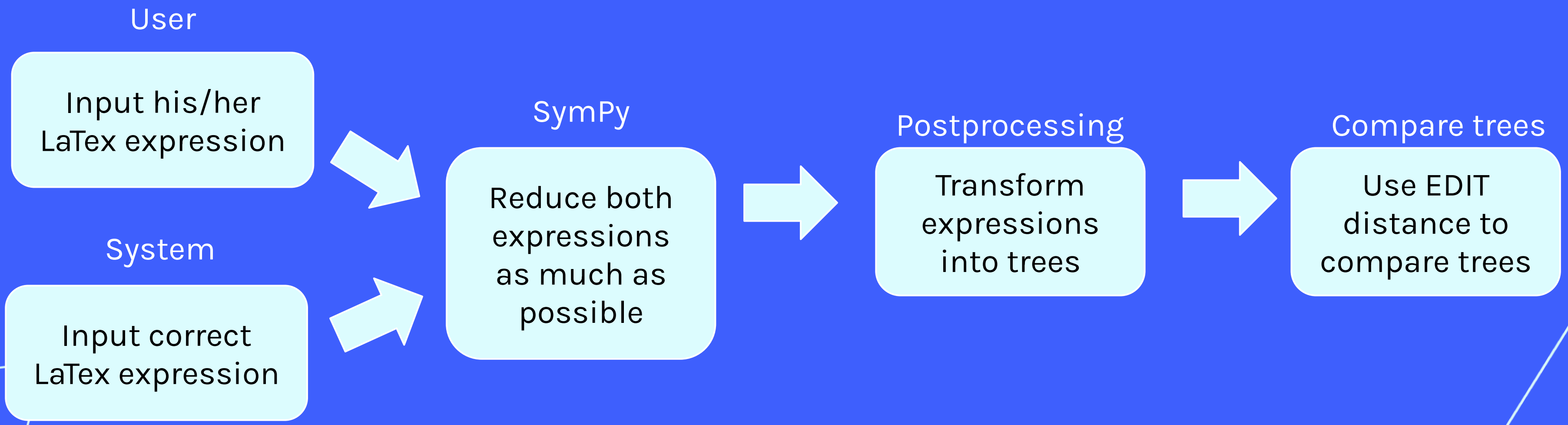
FormulaNet



2. Progress

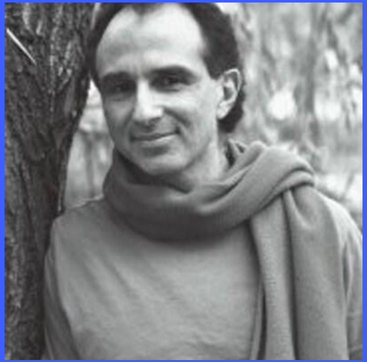
2.1 Algorithmic Approach

Core Workflow



How to compare the trees?

- Quantify how many **insertions**, **deletions**, and **editions** are necessary to transform one graph into another



Dennis Sasha
New York University

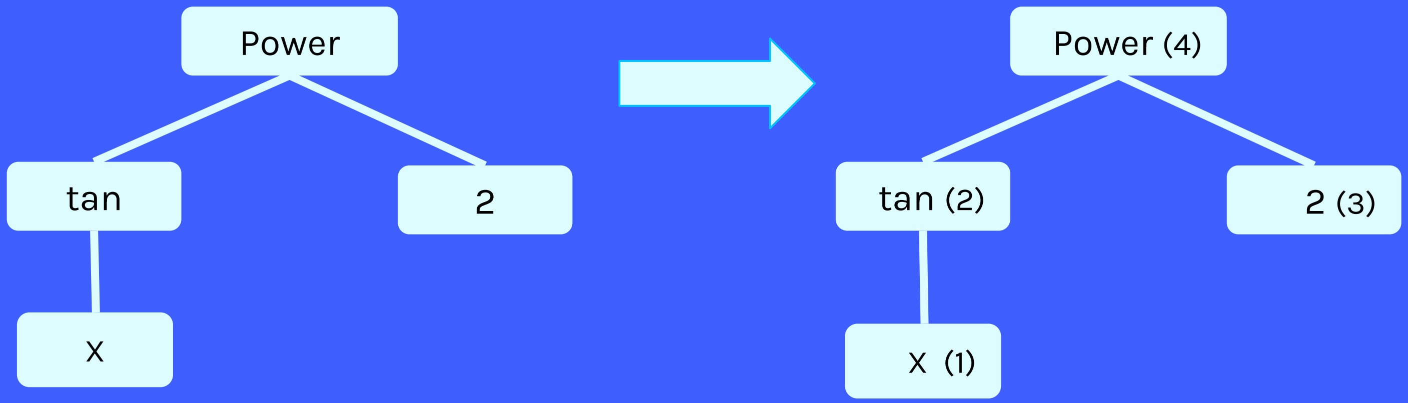
Kaizhong Zhang
University of Ontario

On the editing distance between unordered labeled trees
K. Zhang, D. Sasha (1989)

Revisiting the tree edit distance and its backtracing: A tutorial
Benjamin Paaßen (2018)

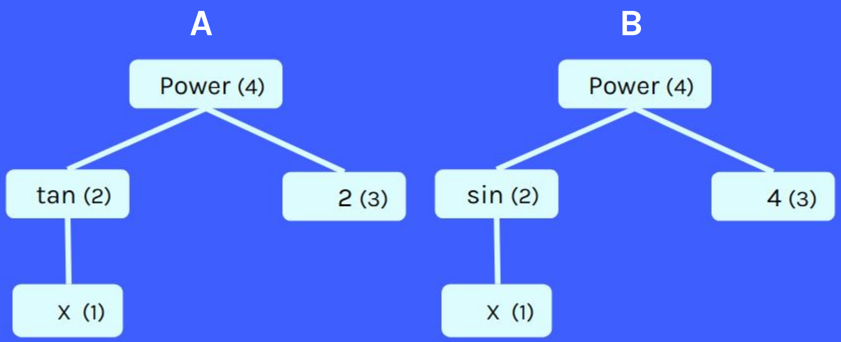
Zhang–Shasha tree edit distance

1. Order assignment



Zhang–Shasha tree edit distance

1. Order assignment
2. Distance Matrix

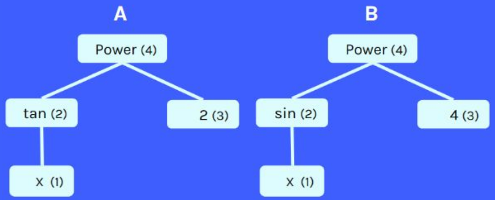


B

A		Λ	1	2	3	4
	Λ	0	1	2	3	4
	1	1				
	2	2				
	3	3				
	4	4				

Zhang–Shasha tree edit distance

1. Order assignment
2. Distance Matrix
3. Forest Distance Calculation

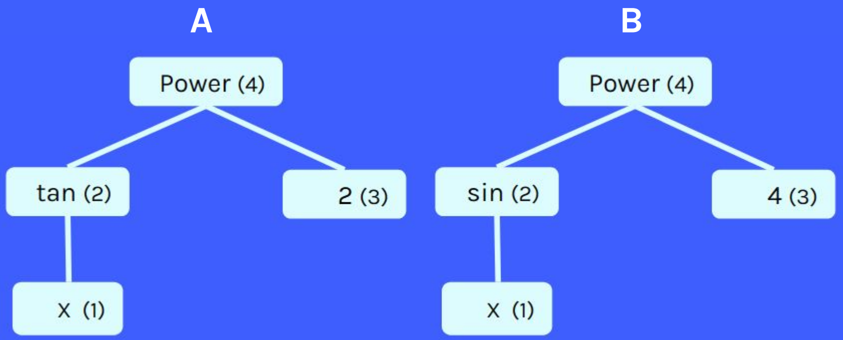


		B				
		Λ	1	2	3	4
	Λ	0	1	2	3	4
	1	1				
A	2	2				
	3	3				
	4	4				

$$D[i,j] = \min\{D[i-1,j]+1, D[i,j-1]+1, D[i-1,j-1]+(1 \text{ if } A_i \neq B_j, \text{ else } 0)\}$$

Zhang–Shasha tree edit distance

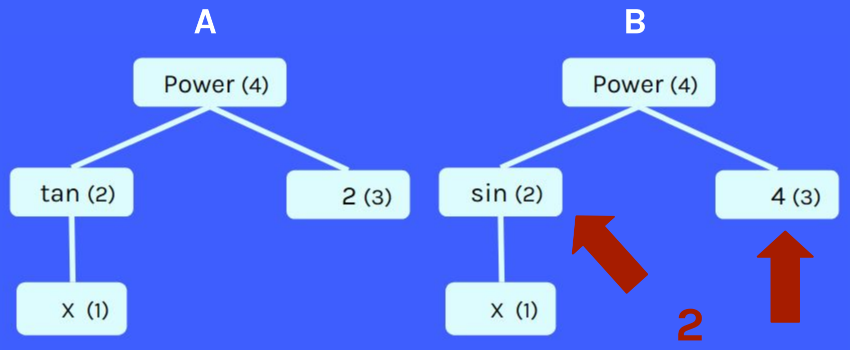
1. Order assignment
2. Distance Matrix
3. Forest Distance Calculation



		B				
		Λ	1	2	3	4
	Λ	0	1	2	3	4
	1	1	0	1	2	3
A	2	2	1	1	2	3
	3	3	2	2	2	3
	4	4	3	3	3	2

Zhang–Shasha tree edit distance

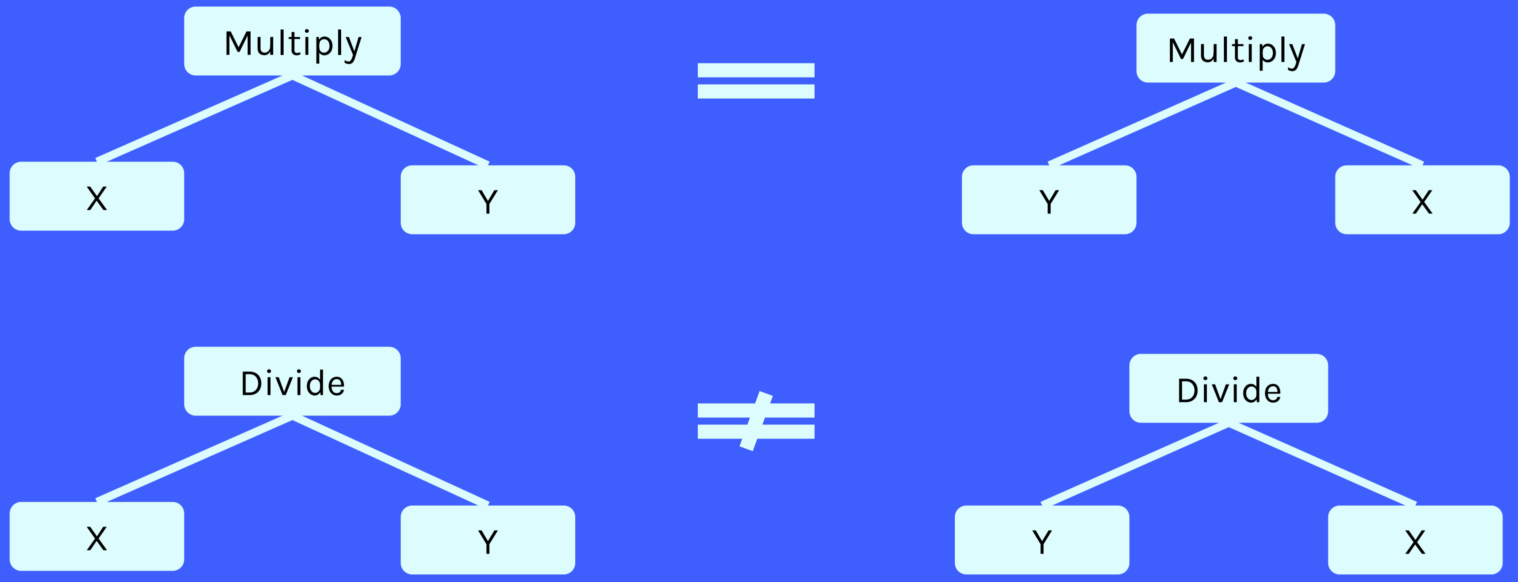
1. Order assignment
2. Distance Matrix
3. Forest Distance Calculation
4. Return tree edit distance



B

		Λ	1	2	3	4
	Λ	0	1	2	3	4
	1	1	0	1	2	3
A	2	2	1	1	2	3
	3	3	2	2	2	4
	4	4	3	3	3	2

Consistency



Other Features

Patterned Tree Construction

Text Alignment Preprocessing

Custom feedback

Handling Undefined Limits

Application User Interface

Feedback generation

Quantitative feedback:

=

$$\mathrm{SCORE} = 1 - \frac{D_{EDIT}}{max(N1, N2)}$$

- D_{EDIT}: Tree Edit Distance, the metric used to measure differences between trees.
- N1: Number of nodes in the student's answer.
- N2: Number of nodes in the expected answer.

- Normalization: Adjusts DEDIT by dividing by N1 and N2 to ensure each mistake has an equal impact on the final score.

Feedback generation

Qualitative feedback

Main idea: given student’s input, detect different types of **mistakes**

- **Detect different types of mistakes:**
- **Sign error** (+ or -)
- **Wrong variable** (‘x’ instead of ‘y’ for instance)
- **Wrong unary expression**

For each type of mistake, a dedicated function was developed

2.2 Dataset generation

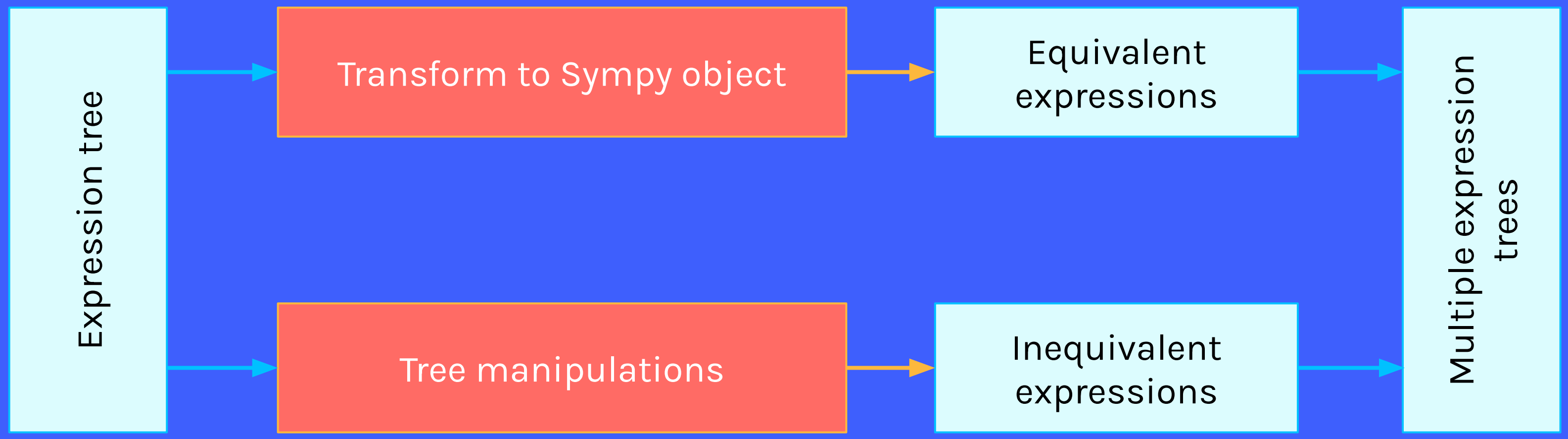
While depth not equal to 1

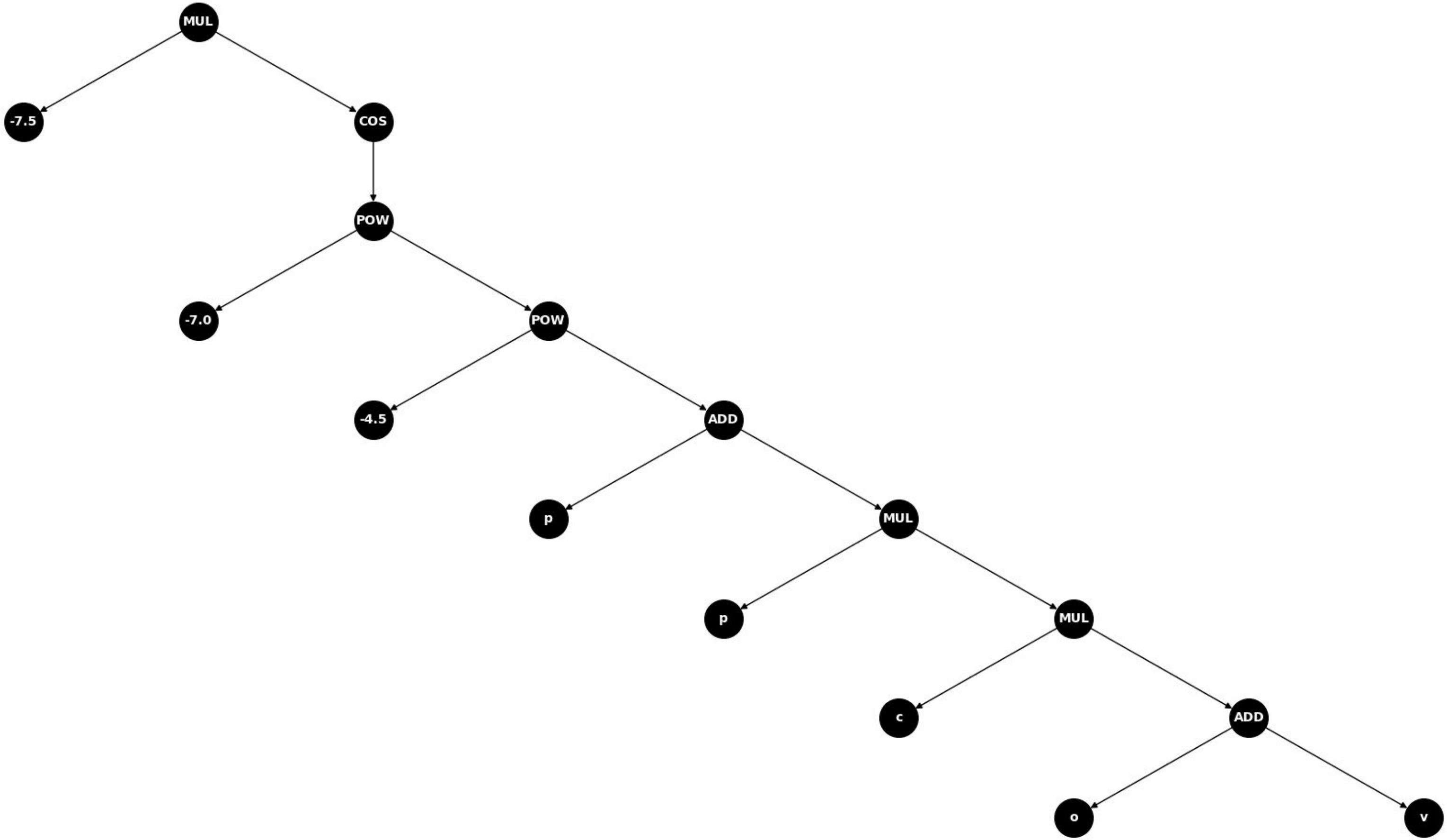
Choose random function/operator/…

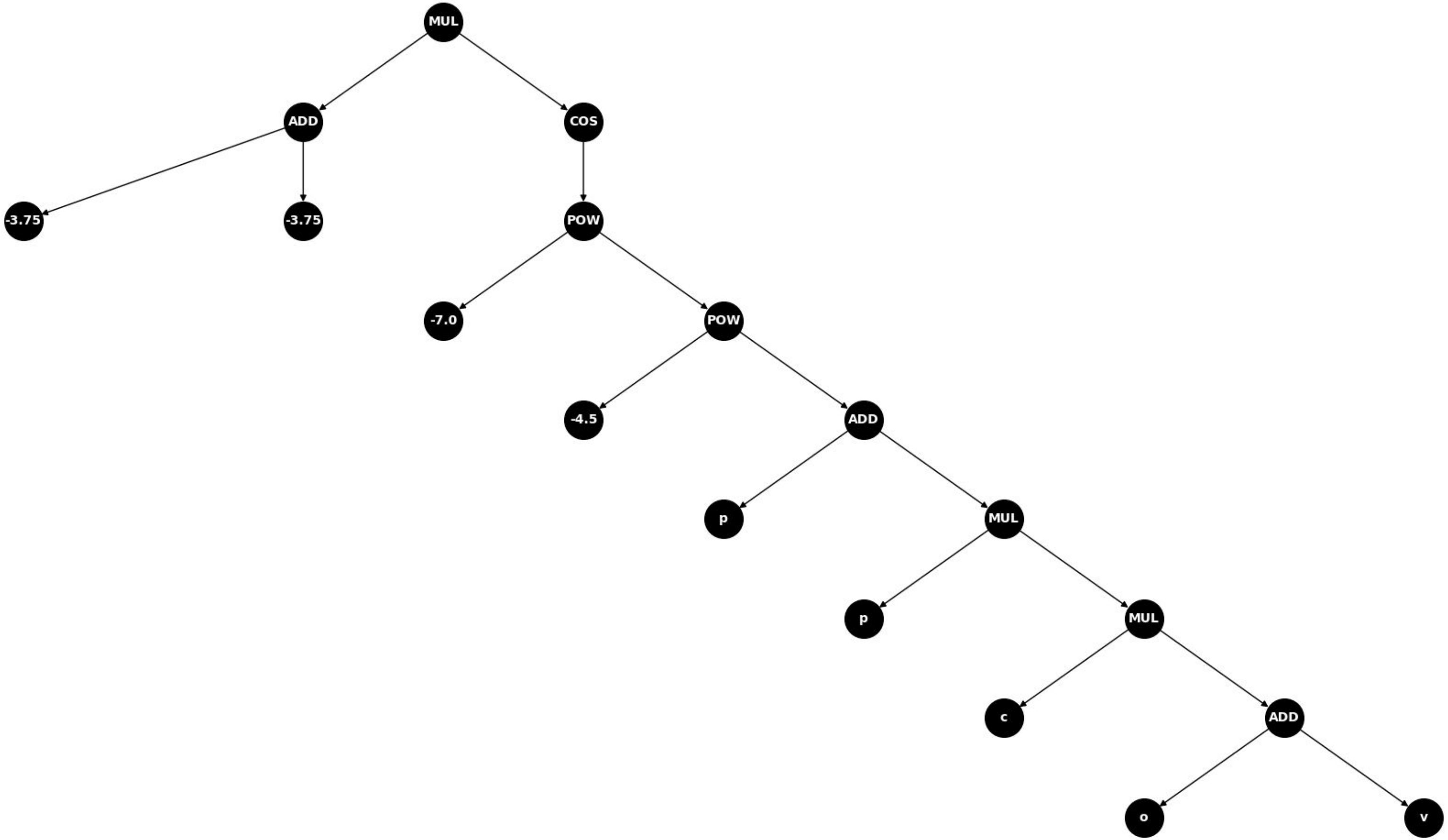
When at depth 1

Choose an atomic (literal/variable/…)

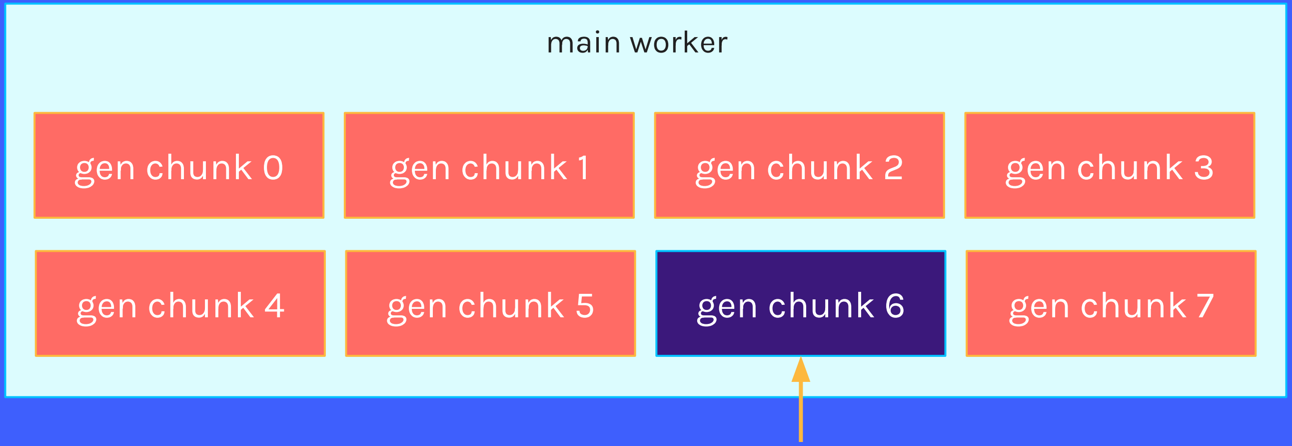
2.2 Dataset generation




MUL
-7.5
COS
POW
-7.0
POW
-4.5
ADD
p
MUL
p
MUL
c
ADD
o
v


MUL
ADD
COS
-3.75
-3.75
POW
-7.0
POW
-4.5
ADD
p
MUL
p
MUL
c
ADD
o
v

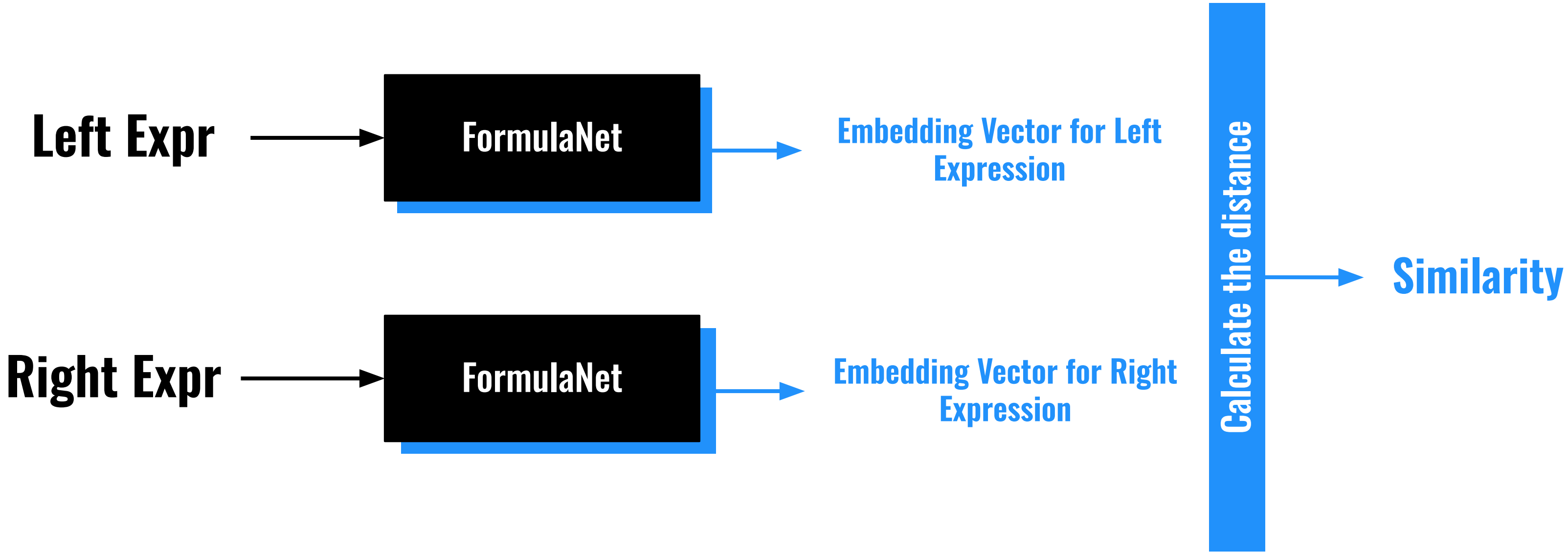
2.2 Dataset generation



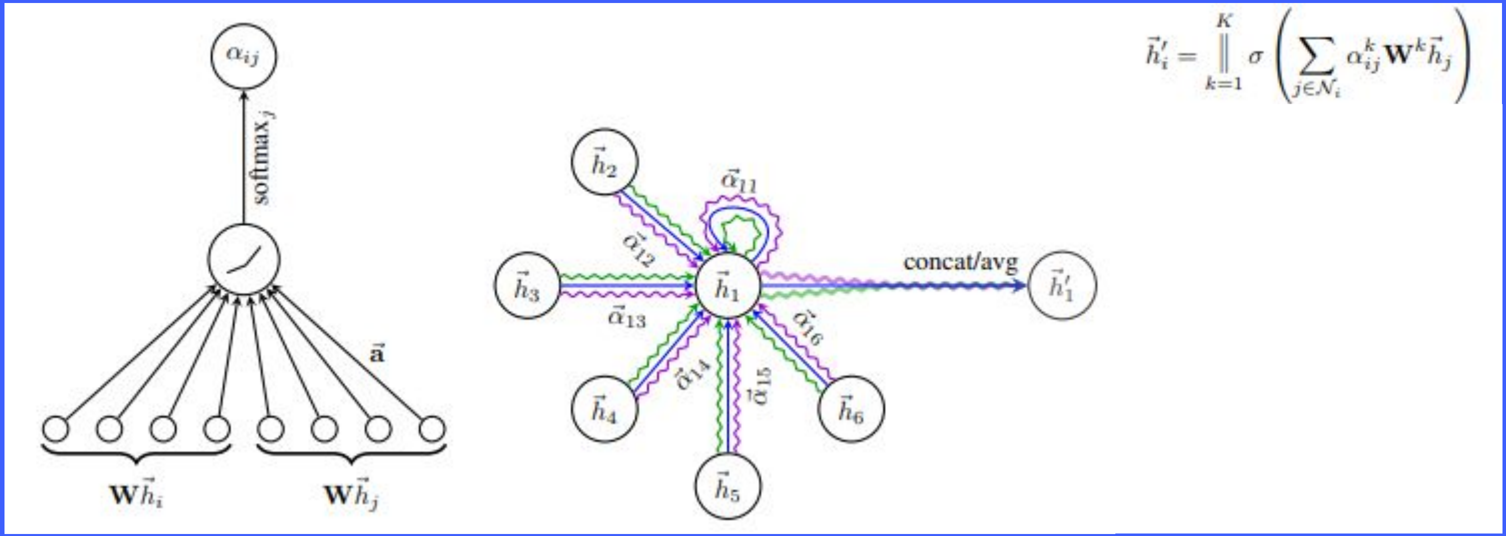
We destroy and recreate the worker when Sympy bricks at an expression

(happens with a probability of ~8% for randomly generated data, this is the best way to handle it)

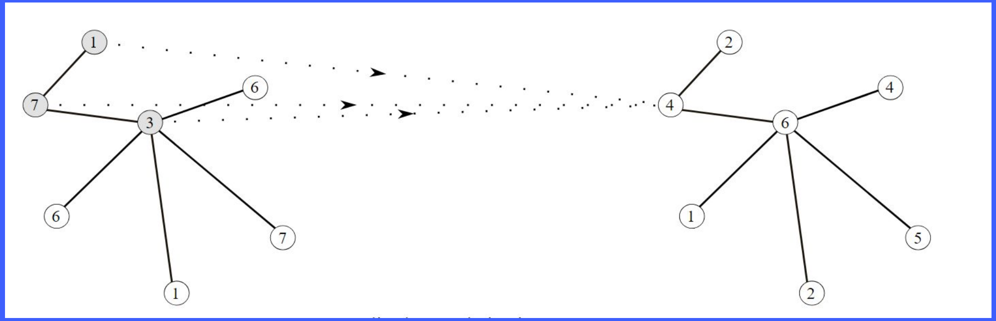
The idea is to use a siamese architecture to learn embeddings for the math expressions. Then, we can use the distance to understand the similarity between two expressions


Left Expr
FormulaNet
Embedding Vector for Left Expression
Right Expr
FormulaNet
Embedding Vector for Right Expression
Calculate the distance
Similarity

Graph Attention Networks



Graph Convolutional Networks



$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$

Node v's initial embedding.

... is just node v's original features.

and for $k = 1, 2, \dots$ upto K:

$$h_v^{(k)} = f^{(k)}\left(W^{(k)} \cdot \frac{\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)}\right) \quad \text{for all } v \in V.$$

Node v's embedding at step k.

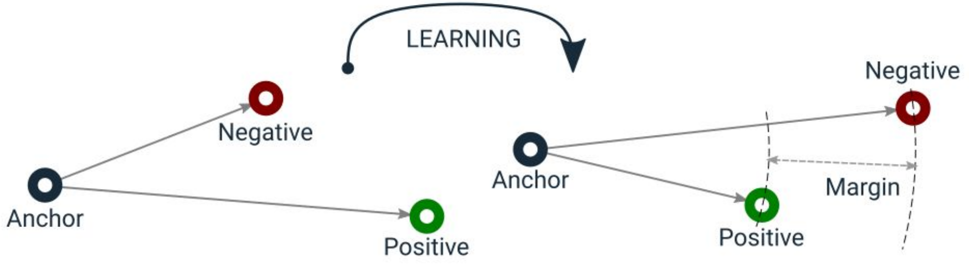
Mean of v's neighbour's embeddings at step $k-1$.

Node v's embedding at step $k-1$.

Color Codes:

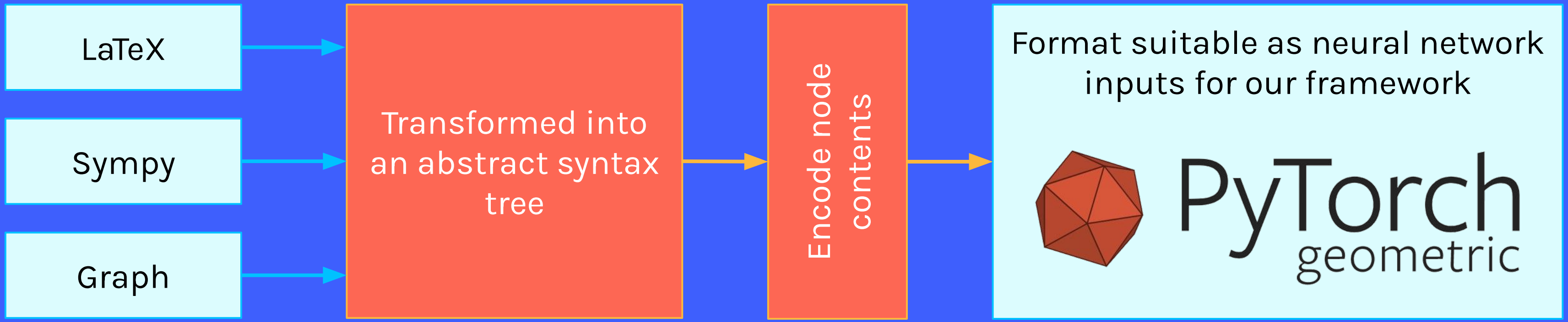
- Embedding of node v.
- Embedding of a neighbour of node v.
- (Potentially) Learnable parameters.

Triplet Loss

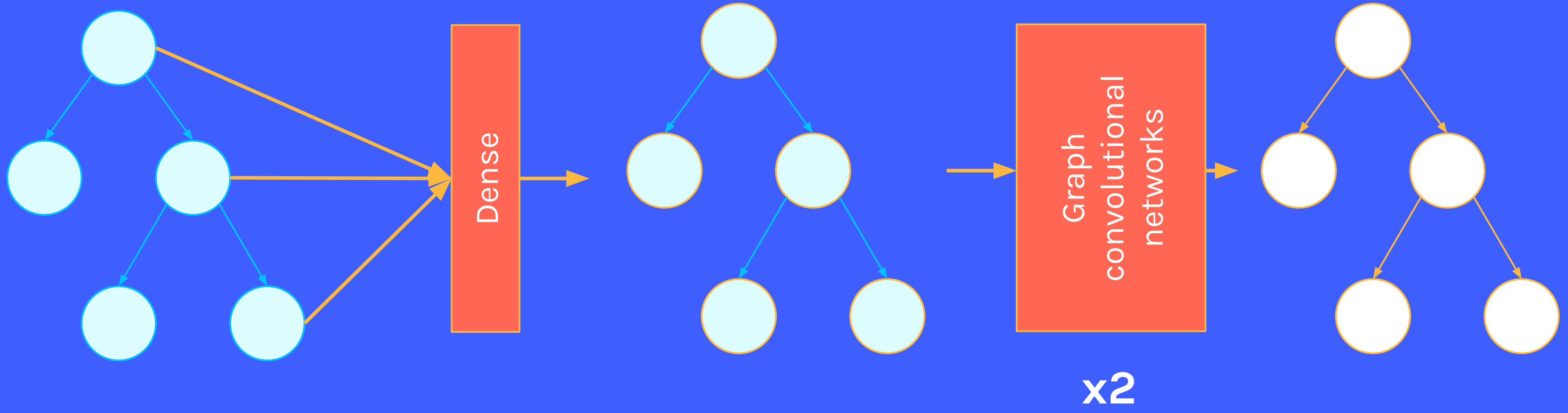


$$L(P, A, N) = max(d(P, A) - d(N, A) + m, 0)$$

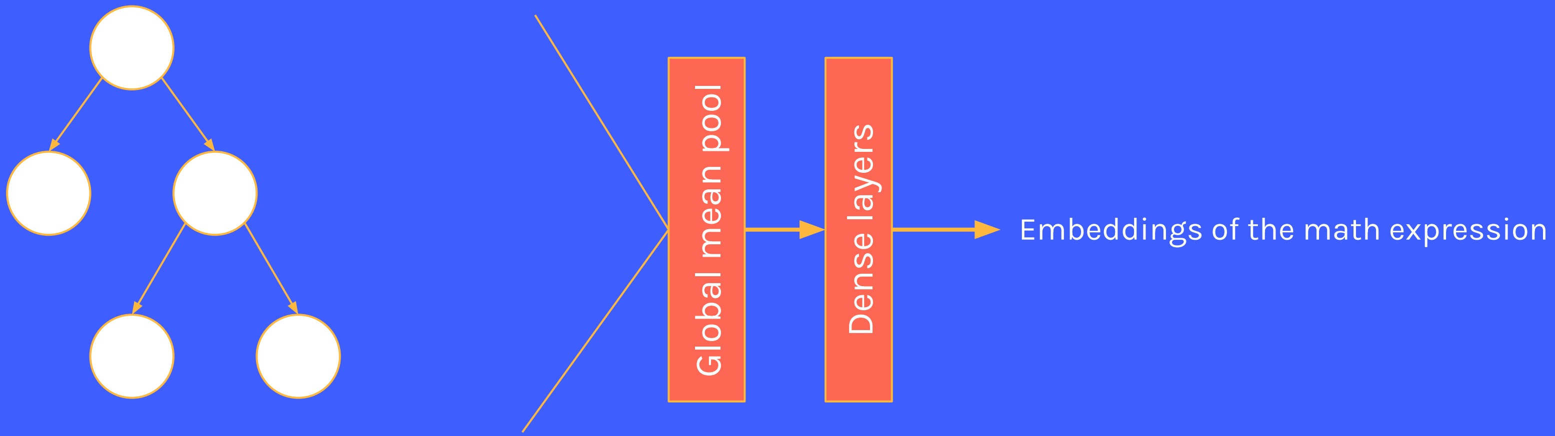
2.3 FormulaNet



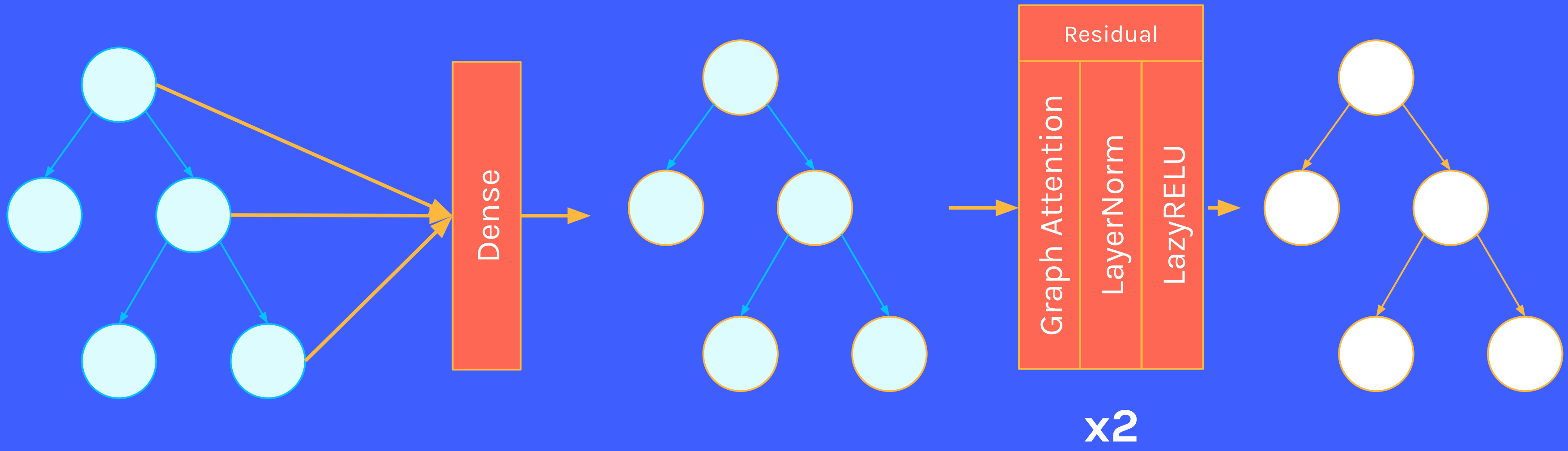
2.3.1 ConvFormulaNet



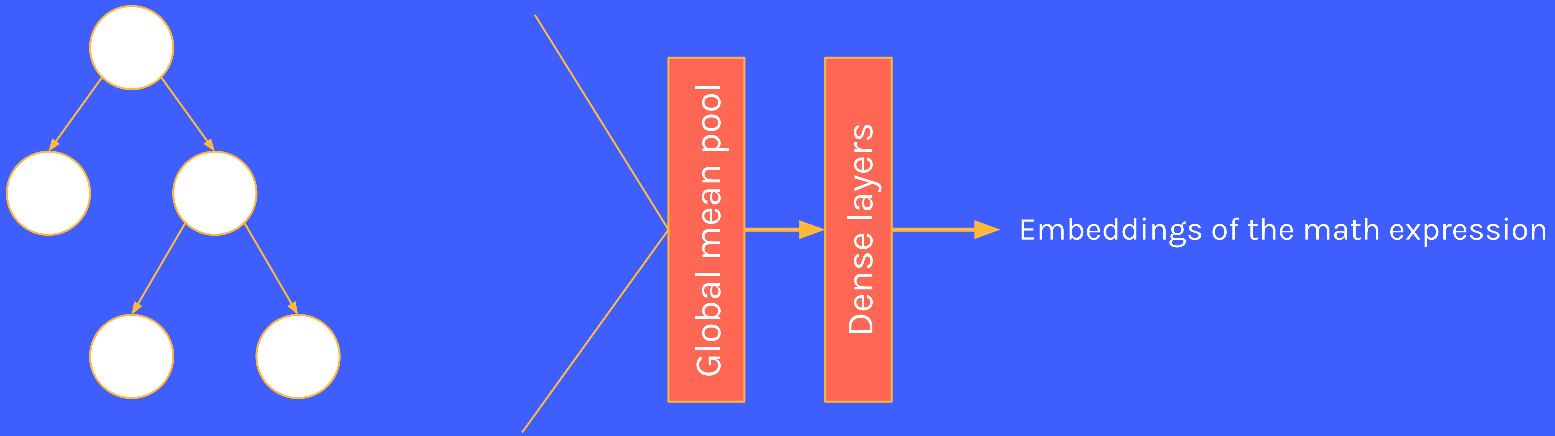
2.3.1 ConvFormulaNet



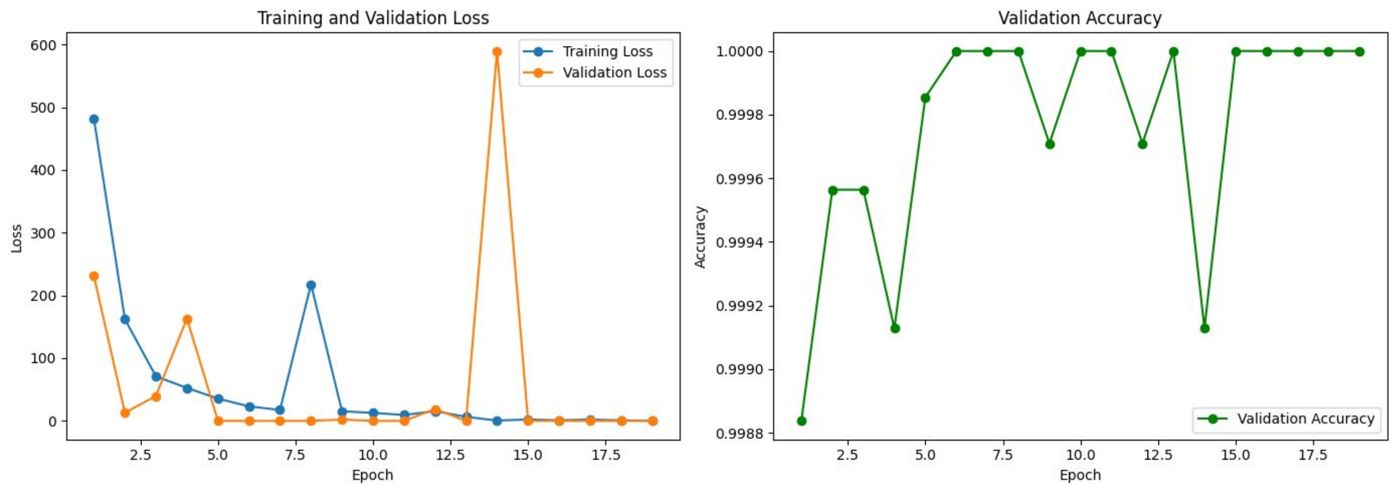
2.3.2 AttFormulaNet



2.3.2 AttFormulaNet



2.3.2 AttFormulaNet



2.4 Error messages

MathBERT

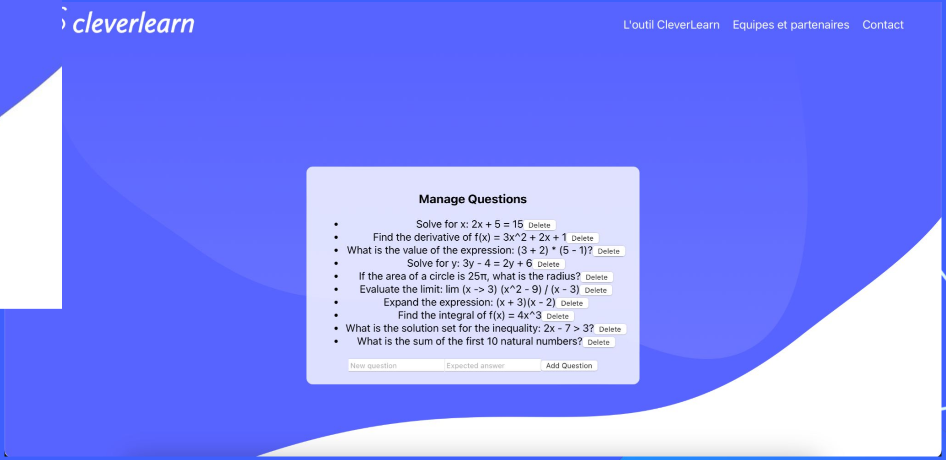
Regular word embeddings

Embeddings for math expressions obtained from FormulaNet

Finetune using pre-generated error messages (enhanced dataset generation)

3. Web site

To facilitate the presentation of results, an interactive web interface was created.



3. Added value

3.2 Added value

- **Automate** the correction platform, eliminating the need of human labor for correcting every single question

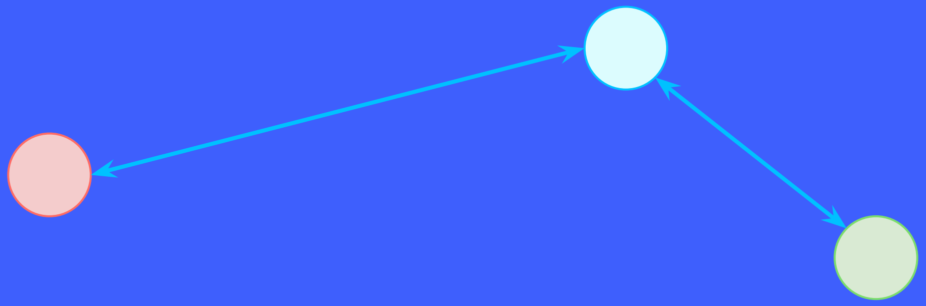
- Provide **instant feedback** to the student, allowing for a more interactive and dynamical way to learn

- Explored and experimented with the possibilities of using deep neural networks for expression comparison (**research assets** that can later on be utilized by the client)

- Conceived **front end platform** that allows for simple and convenient presentation

4. What we learned

4.1 Lessons learned



~22000000

~475000000

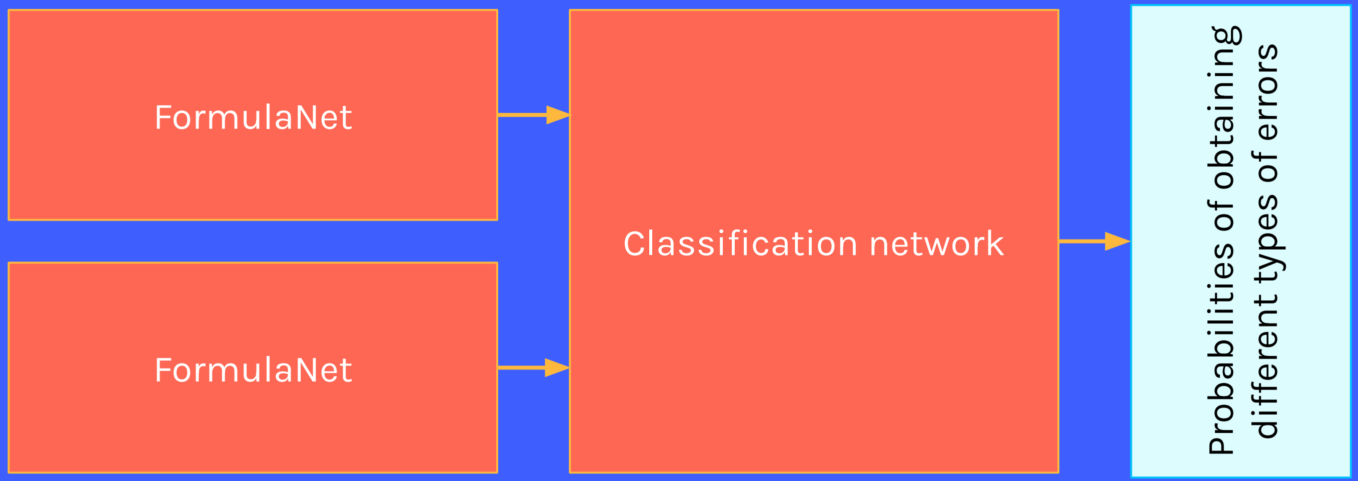
4.1 Lessons learned

Really difficult approach

As difficult as trying to learn “image embeddings” or “text embeddings” for a language model. Mathematical formulas can be really expressive

We ought to limit the scope of our problem

4.1 Multi-label classification



4.1 Lessons learned

Euclidean distance is the best when it comes to training, the model converges better with it

It takes 14 epochs on average to reach 60% accuracy (from 8 different runs)

Transformers/Attention yields great results when it comes to learning how to properly separate similar and dissimilar examples.

Convolution was better at learning symmetry and structure (even with low accuracy it still handled *some* examples well)

4.1 Lessons Learned

Soft Skills	Hard Skills
Communication with the client and within the group	Graphs for numerical expressions
Team organization to separate tasks	Graph neural networks
Adaptability to change approaches and to propose new solutions	React and Frontend software engineering best practices
	Graphs manipulation

5. Conclusion

Demo time

A demo is worth a thousand words…

https://drive.google.com/file/d/1XLt3TL0A0QbNRIZ1w-XhLm7o8YVRRzmn/view?usp=sharing

Thank you!

CentraleSupélec

cleverlearn