Symbolic Regression via Neural network weights

Singh Dilpreet, Barke Saiprasad

Project goals

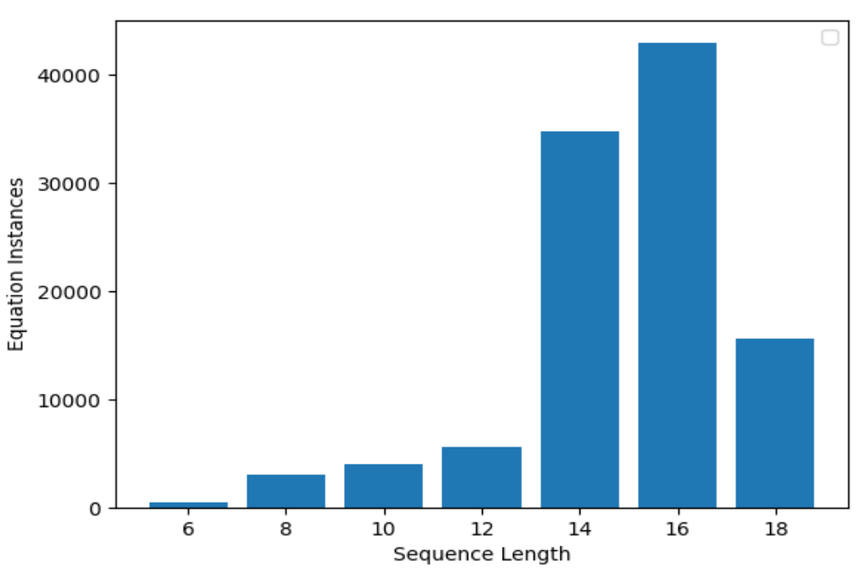
- Can we use neural network weights to represent equations?
- Generate equations in the form of string tokens given only sample data i.e., weights of a network which “represent” an equation.
- The network in the previous point is an MLP which is used to perform regression on sampled x, y values of the said equation. **E.g., “$y=ax^2+bx+c$”.**

Methodology

- **Phase 1:**
  - Randomly generate polynomial equations up to the 2nd order with the form **(+/-) a*x*x (+/-) b*x (+/-) c** with **a, b, c ∈ [-5, 5]** and sample 5000 (x, y) pairs from these equations for uniformly sampled values of x **∈ [-1, 1]**
  - Train an MLP on these samples (input x, output y) with an MSE loss
  - Store the equations and weights of the trained MLPs as samples for the dataset to be used in the next phase.
- **Phase 2:**
  - The input is the weights of the previously trained MLP. The output is the equation as a string.
  - We use a sequential model e.g., Transformer Decoder to predict the equation string.
  - The objective is the difference between the true equation and the predicted one.

Datasets

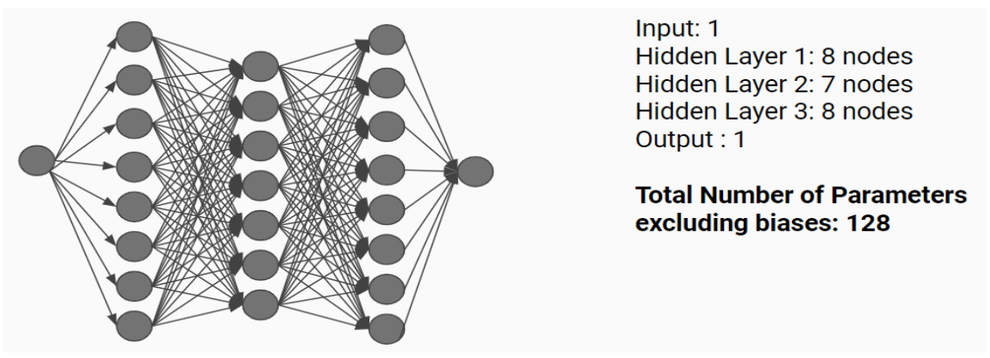
- **Dataset 1:** 100K polynomial equations and (x, y) pairs and weights of the MLPs trained from on these pairs.



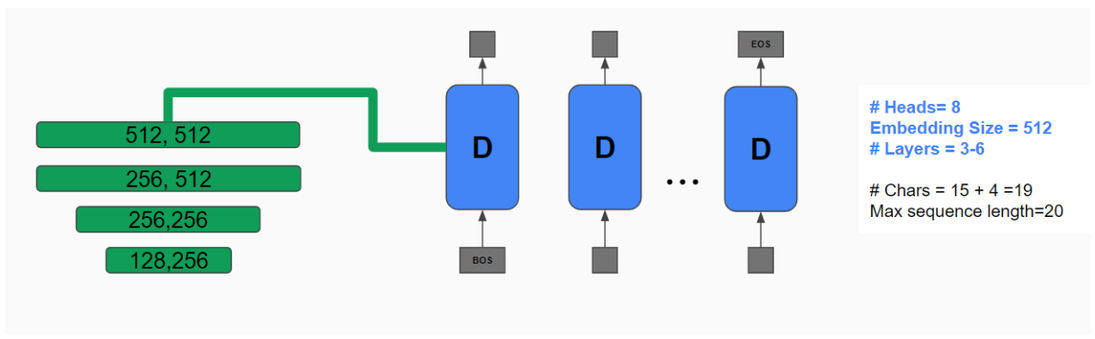
- **Dataset 2:** 50k quadratic equations with 16 tokens only with (x, y) pairs and weights of MLPs trained on these pairs

Implementation

- **Phase 1:** The following MLP was trained for 300 epochs on 100K equations. Final validation losses in all MLPs were very close to 0. The MLP weights are flattened and together with the equations stored as key value pairs as part of a larger dataset.

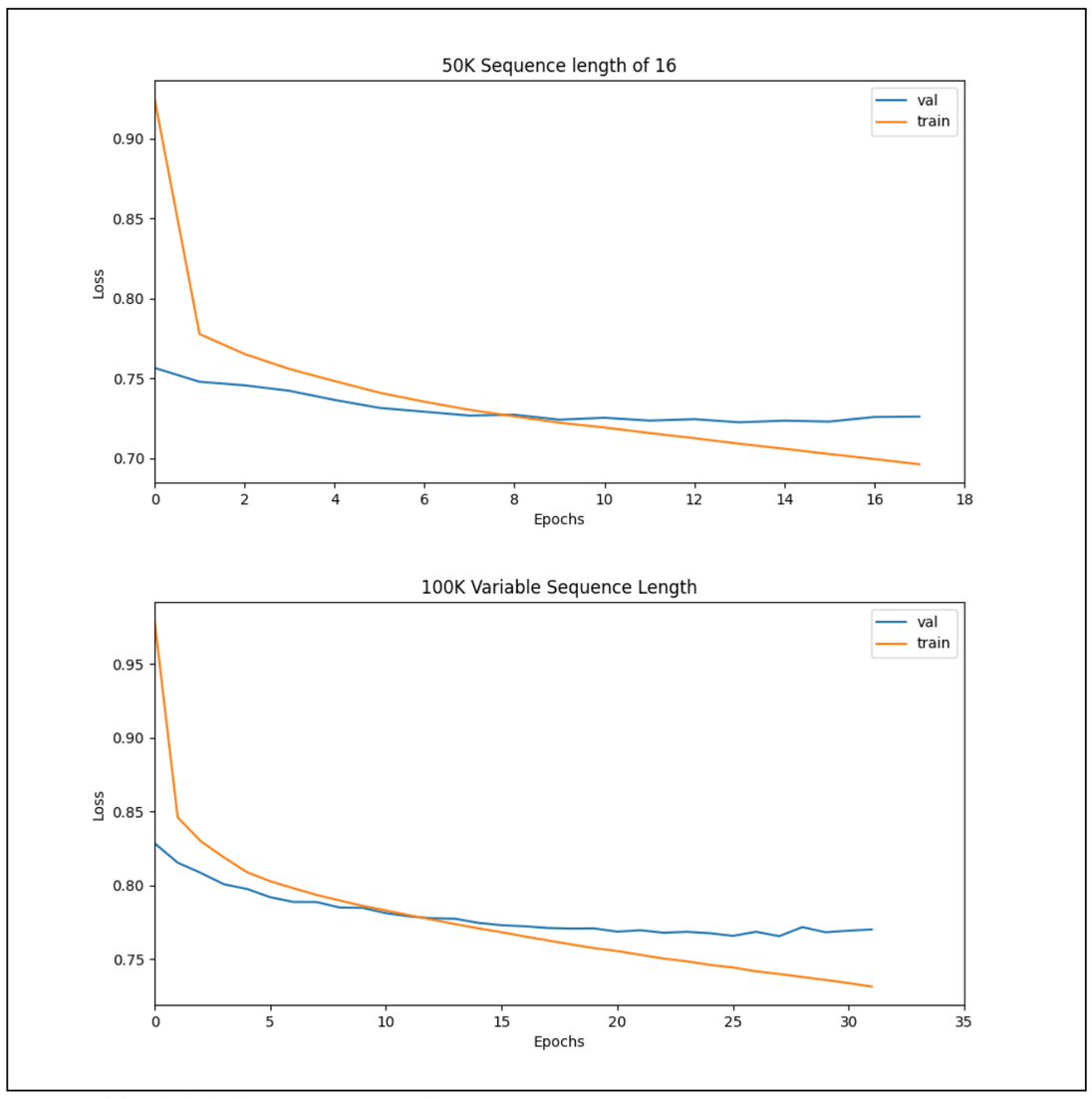


- **Phase 2:**
  - The weights from the previous phase are inputs to an MLP that expands the weight vector from 128 to 512. This new feature vector is given as memory to a Transformer Decoder. The transformer decoder output is given to a generator which outputs a class vector with length equal to vocabulary size which in our case is 19. i.e. Vocab = {0-9, +, *, -, x, .}
  - We use cross entropy loss during training-validation and n-BLEU scores on the test set.
  - Greedy search and Beam search are applied to the outputs of the generator during inference.



Experiments and Results

- **Training Validation Curve for our model**



- **Model inference results**
  - **Biased dataset**

1. **Ground Truth**: +4 . 6 * x * x + 3 . 3 * x + 5
   Predicted equation: + 3 . 6 * x * x + 1 . 6 * x + 4. 7
2. **Ground Truth:** - 2 . 8 * x * x – 2 . 0 * x – 2 . 9
   Predicted equation: - 3 . 7 * x * x – 4 . 7 * x - 4
3. **Ground Truth:** + 4 * x – 3 . 4
   Predicted equation: + 1. 7 * x * x + 3. 7 * x – 4
4. **Ground Truth:** + 1 * x + 1 . 3
   Predicted equation: - 0. 6 * x * x + 3. 3 * x + 3. 0

  - **Uniform length dataset**

5. **Ground Truth:** - 4 . 0 * x * x - 4 .0 * x - 5
   Predicted equation : - 4 . 9 * x * x - 5 * x - 4 . 0
6. **Ground Truth:** - 3 . 4 * x * x + 2 . 3 * x + 3
   Predicted equation : - 3 . 3 * x * x + 3 . 3 * x + 3

Observations and inferences

- The model fails to learn the exact numeric values of coefficients of the equation.
- In certain cases, as shown above it predicts numerical values close to the ground truth.
- The model correctly learns the positions of each of the individual “types” of tokens relative to each other.
- In certain cases, the model also predicts the sign of the numerical values correctly as seen above.
- The 100k dataset is heavily skewed towards sequences with higher number of tokens i.e., 14, 16. This causes an inherent bias in the network towards longer sequence lengths and results in incorrect predictions for targets with shorter length.
- *This behavior is fixed when we train the network on a dataset with nearly uniform distribution of sequence lengths*

Future scope

- The equation weights in a particular order do not mean anything. Augment the dataset by manufacturing more data points by permuting the elements of the weight vectors OR using positional encoding before feeding the weight vector into the feature expanding MLP.
- Simplifying the problem further by using whole numbers for equation coefficients and restricting the order of the polynomials to one. This will be a good feasibility study for the larger problem.
- Since x lies in [-1, 1] and equation coefficients lie in [-5, 5], it is possible that in certain (x, y) pairs the c term dominates because it's not being multiplied by a value less than 1. Retrying the exercise with larger values of x and coefficients might give further insights.