Transformer for Mathematical Programming

Jiasen Wang
Purple Mountain Laboratories, Nanjing, China
wangjiasen@pmlabs.com.cn

Abstract

This note records experimental results on using a transformer trained via supervised learning for solving optimization problems. The code is at https://github.com/jiasenwangs/transopt.

Index Terms

Transformer, optimizer, global optimization, deep learning.

I. Introduction

Transformer [1] is a universal input sequence to output sequence transduction model. Solving an optimization problem (mathematical program) is essentially a sequence2sequence task, with input being a mathematical program, and output being a (optimal/sub-optimal) solution.

It is interesting to use transformer to solve (global) optimization problems.

II. Problems

Problem (1) is a 2-dimension nonlinear program [2]:

$$
\begin{aligned}
&\min_{x}\ f(x) = a\cos(x_0)\sin(x_1) - x_0/(1+x_1^2) && \text{(1a)}\\
&\text{s.t.}\ -1 \le x_0 \le 2, && \text{(1b)}\\
&\qquad -1 \le x_1 \le 1, && \text{(1c)}
\end{aligned}
$$

where a is a parameter indicating that the problem varies with respect to a. When $a = 1$, $x^* = [2, 0.10578]^T$ is the global optimal solution.

Problem (2) is a 5-dimension global program [2]:

$$
\begin{aligned}
&\min_{x}\ f(x) = a(x_0-1)^2 + (x_0-x_1)^2 + (x_1-x_2)^3 + (x_2-x_3)^4 + (x_3-x_4)^4 && \text{(2a)}\\
&\text{s.t.}\ x_0 + x_1^2 + x_2^3 - 3\sqrt{2} - 2 = 0, && \text{(2b)}\\
&\qquad x_1 - x_2^2 + x_3 - 2\sqrt{2} + 2 = 0, && \text{(2c)}\\
&\qquad x_0x_4 - 2 = 0, && \text{(2d)}\\
&\qquad -5 \le x_i \le 5,\ i = 0, ..., 4. && \text{(2e)}
\end{aligned}
$$

When $a = 1$, $x^* = [1.1166, 1.2204, 1.5378, 1.9728, 1.7911]^T$ is the global optimal solution.

Problem (3) is a 6-dimension nonlinear program [2], [3]:

$$
\begin{aligned}
&\min_{x}\ f(x) = -ax_3 && \text{(3a)}\\
&\text{s.t.}\ x_0 + k_1x_0x_4 - 1 = 0, && \text{(3b)}\\
&\qquad x_1 - x_0 + k_2x_1x_5 = 0, && \text{(3c)}\\
&\qquad x_2 + x_0 + k_3x_2x_4 - 1 = 0, && \text{(3d)}\\
&\qquad x_3 - x_2 + x_1 - x_0 + k_4x_3x_5 = 0, && \text{(3e)}\\
&\qquad x_4^{0.5} + x_5^{0.5} - 4 \le 0, && \text{(3f)}\\
&\qquad 0 \le x_i \le 1,\ i = 0, ..., 3, && \text{(3g)}\\
&\qquad 0 \le x_i \le 16,\ i = 4, 5, && \text{(3h)}
\end{aligned}
$$

where $k_1 = 0.09755988$, $k_2 = 0.99k_1$, $k_3 = 0.0391908$, $k_4 = 0.9k_3$. When $a = 1$, $x^* = [0.772, 0.517, 0.204, 0.388, 3.036, 5.097]^T$ is the global optimal solution.

Problem (4) is a 5-dimension global program [4]:

$$\min_{x} \ f(x) = -20a\exp(-0.2\sqrt{\frac{\sum_{i=0}^{4} x_i^2}{5}}) - \exp(\frac{\sum_{i=0}^{4}\cos(2\pi x_i)}{5}) + 20 + \exp(1) \tag{4a}$$

$$\text{s.t. } x \in [0, 2.5]^5. \tag{4b}$$

When $a = 1$, $x^* = [0, 0, 0, 0, 0]^T$ is the global optimal solution.

Problem (5) is a 5-dimension mixed-integer nonlinear program [2]:

$$\min_{x} \ f(x) = 2ax_0 + 3x_1 + 1.5x_2 + 2x_3 - 0.5x_4 \tag{5a}$$

$$\text{s.t. } x_0^2 + x_2 - 1.25 = 0, \tag{5b}$$

$$x_1^{1.5} + 1.5x_3 - 3 = 0, \tag{5c}$$

$$x_0 + x_2 - 1.6 \leq 0, \tag{5d}$$

$$1.333x_1 + x_3 - 3 \leq 0, \tag{5e}$$

$$-x_2 - x_3 + x_4 \leq 0, \tag{5f}$$

$$x_i \geq 0, \ i = 0, 1, \tag{5g}$$

$$x_i \in \{0, 1\}, \ i = 2, 3, 4. \tag{5h}$$

When $a = 1$, $x^* = [1.12, 1.31, 0, 1, 1]^T$ is the global optimum.

Problem (6) is a 10-dimension combinatorial program [2]:

$$\min_{x} \ f(x) = ax^TQx \tag{6a}$$

$$\text{s.t. } x_i \in \{0, 1\}, \ i = 0, 1, ..., 9, \tag{6b}$$

where Q is a matrix:

$$Q = \begin{pmatrix} -1 & -2 & 2 & 8 & -5 & 1 & -4 & 0 & 0 & 8 \\ -2 & 2 & 0 & -5 & 4 & -4 & -4 & -5 & 0 & -5 \\ 2 & 0 & 2 & -3 & 7 & 0 & -3 & 7 & 5 & 0 \\ 8 & -5 & -3 & -1 & -3 & -1 & 7 & 1 & 7 & 2 \\ -5 & 4 & 7 & -3 & 1 & 0 & -4 & 2 & 4 & -2 \\ 1 & -4 & 0 & -1 & 0 & 1 & 9 & 5 & 2 & 0 \\ -4 & -4 & -3 & 7 & -4 & 9 & 3 & 1 & 2 & 0 \\ 0 & -5 & 7 & 1 & 2 & 5 & 1 & 0 & -3 & -2 \\ 0 & 0 & 5 & 7 & 4 & 2 & 2 & -3 & 2 & 3 \\ 8 & -5 & 0 & 2 & -2 & 0 & 0 & -2 & 3 & 3 \end{pmatrix}.$$

When $a = 1$, $x^* = [1, 1, 0, 0, 1, 0, 1, 1, 0, 0]^T$ is the global optimal solution.

Problem (7) is a one-dimension continuous program [2]:

$$\min_{x} \ f(x) = ax^4 - 3x^3 - 1.5x^2 + 10x \tag{7a}$$

$$\text{s.t. } -5 \leq x \leq 5. \tag{7b}$$

When $a = 1$, $x^* = -1$ is the global optimal solution.

Problem (8) is a 100-dimension global program [4]:

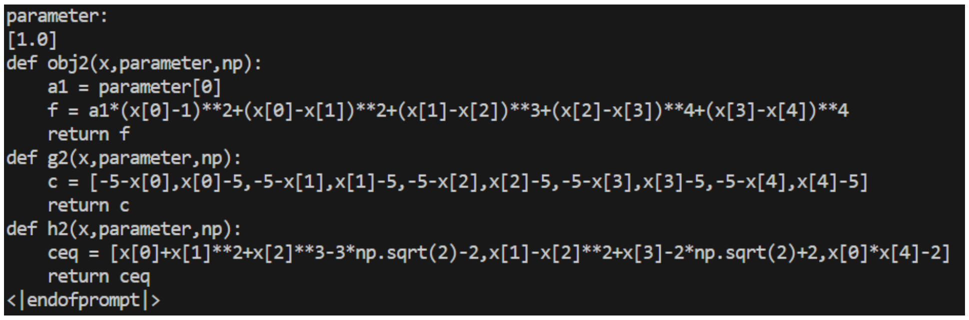
$$\min_{x} \ f(x) = -20a\exp(-0.2\sqrt{\frac{\sum_{i=0}^{99} x_i^2}{100}}) - \exp(\frac{\sum_{i=0}^{99}\cos(2\pi x_i)}{100}) + 20 + \exp(1) \tag{8a}$$

$$\text{s.t. } x \in [0, 2.5]^{100}. \tag{8b}$$

When $a = 1$, $x^* = [0, 0, ..., 0, 0]^T \in R^{100}$ is the global optimum.

III. Transformer

The adopted transformer model is from [5]. The input and output sequences are tokenized by using tiktoken [6].



```
parameter:
[1.0]
def obj2(x,parameter,np):
    a1 = parameter[0]
    f = a1*(x[0]-1)**2+(x[0]-x[1])**2+(x[1]-x[2])**3+(x[2]-x[3])**4+(x[3]-x[4])**4
    return f
def g2(x,parameter,np):
    c = [-5-x[0],x[0]-5,-5-x[1],x[1]-5,-5-x[2],x[2]-5,-5-x[3],x[3]-5,-5-x[4],x[4]-5]
    return c
def h2(x,parameter,np):
    ceq = [x[0]+x[1]**2+x[2]**3-3*np.sqrt(2)-2,x[1]-x[2]**2+x[3]-2*np.sqrt(2)+2,x[0]*x[4]-2]
    return ceq
<|endofprompt|>
```

Fig. 1. Input sequence (string) for Problem (2). The string "[1.0]" in the second line is for test, which means $a = 1$ in test phase. It will be some other string (e.g., "[1.034]") for training and validation. obj2 defines the objective function of Problem (2). g2 and h2 respectively define the inequality and equality constraints of the problem.

IV. Data

Training and validation data are obtained by sampling a from sets $[0.9, 0.99]$ and $[1.01, 1.1]$. For each problem, training and validation datasets include 20000 (sample a 20000 times) and 2000 (sample a 2000 times) samples, respectively. In total, there are $160,000$ training samples and $16,000$ validation samples.

Each training or validation sample is an (input sequence, output sequence) pair. For example, input sequence for Problem (2) looks like that in Fig. 1 except that the string "[1.0]" in the second line will be some other string (e.g., "[1.034]"). The number 1.034 is the exact sampled value of a. It is possible to use natural language prompts, rather than the python-style prompt (input sequence) as in Fig. 1.

Gurobi optimizer [7] is used to obtain the output sequence for every problem and individual a. For example, if Gurobi solves Problem (2) with solution $x =$[1.1147, 1.2204, 1.538, 1.973, 1.794]^T for $a = 1.034$, then "[1.1147, 1.2204, 1.538, 1.973, 1.794]" is used as the output sequence (string). If there is no solution of a problem, we can set output to "no solution". If one dimension of a solution of a problem reaches infinity, we can set that dimension to np.inf.

There are 8 test samples (8 problems). By setting $a = 1.0$ for each problem, we get a test sample. A test sample only includes an input sequence, see Fig. 1. The output sequence obtained by sending the input sequence to the transformer is cast as a list (vector) including only floating-point numbers. For example, if the output sequence is "[1.12,1.31,0,1,1]" for Problem (5), then it will be cast as output list [1.12,1.31,0,1,1].

V. Method

The transformer is set to 6 layers with 19,756,917 parameters. Supervised learning is used for training the transformer. The training algorithm is Adam [8]. The loss function is KL divergence on tokens. $160,000$ training samples are fed to the transformer with minibatch size 16. 100 training epochs are used.

The best validated model during training phase is used for testing (inference).

VI. Results

The training and validation losses are plotted in Fig. 2.

Error measure for a test problem is:

$$\text{error} = \frac{1}{n}\sum_{i=0}^{n-1} |x_i^* - \hat{x}_i|,$$

where vector x^* ($x^* = x_0^*$ if x^* is one dimension) is the global optimal solution of the test problem when $a = 1$, $n = \dim(x^*)$, $\hat{x}$ is the output vector (list) of the transformer for solving the problem. We take mean value of the 8 errors for 8 test problems as the total mean absolute error, which is 0.0029.

The exact output lists of the transformer in the test (inference) phase are recorded as follows.

For Problem (1), transformer output is [2.0, 0.1147001143900035], ground truth program solution is [2, 0.10578].

For Problem (2), transformer output is [1.1115980883554313, 1.218831433177214, 1.539050306708335, 1.9781908528462993, 1.7991810888798527], ground truth program solution is [1.1166, 1.2204, 1.5378, 1.9728, 1.7911].

For Problem (3), transformer output is [0.7717242192034419, 0.516976390621079, 0.2040315750767915, 0.388811654788413, 3.031980818413342, 5.101913614894], ground truth is [0.772, 0.517, 0.204, 0.388, 3.036, 5.097].

For Problem (4), transformer output is [0.0, 0.0, 0.0, 0.0, 0.0], ground truth is [0.0, 0.0, 0.0, 0.0, 0.0].

For Problem (5), transformer output is [1.118033988749895, 1.3103706971016593, 0, 1, 1], ground truth is [1.12, 1.31, 0, 1, 1].

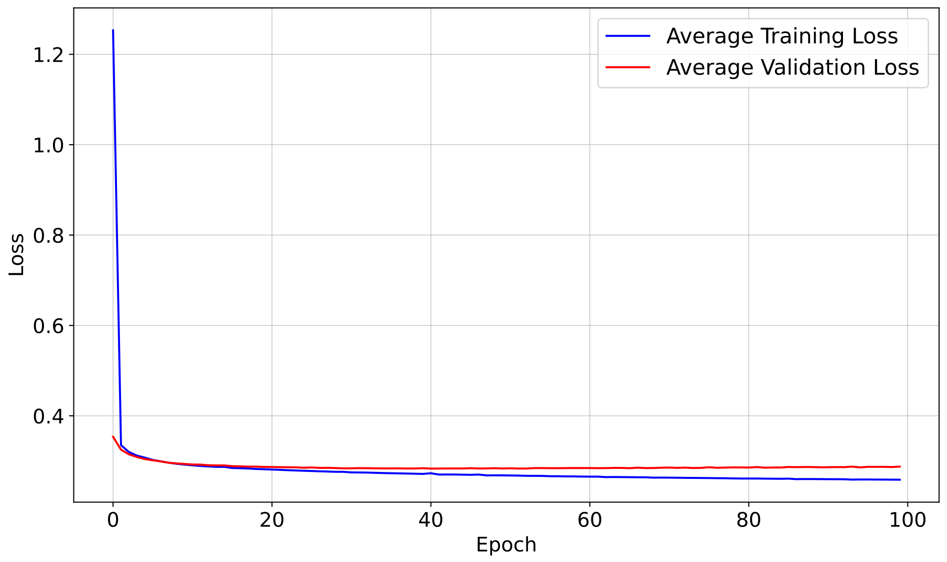


Fig. 2. Training and validation losses.

For Problem (6), transformer output is [1, 1, 0, 0, 1, 0, 1, 1, 0, 0], ground truth is [1, 1, 0, 0, 1, 0, 1, 1, 0, 0].

For Problem (7), transformer output is [-0.9876598638818813], ground truth is [-1].

For Problem (8), transformer output is [0.0, 0.0,..., 0.0] (100-dimension), ground truth is [0.0, 0.0,..., 0.0] (100-dimension).

Outputs of the transformer are proximal to the ground truths. So, solving optimization problems will be as simple as querying a neural network?

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gümüş, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger, *Handbook of Test Problems in Local and Global Optimization*. Springer New York, NY, 1999.

[3] H. Ryoo and N. Sahinidis, "Global optimization of nonconvex NLPs and MINLPs with applications in process design," *Computers & Chemical Engineering*, vol. 19, no. 5, pp. 551 – 566, 1995.

[4] T. Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.

[5] "The Annotated Transformer," https://nlp.seas.harvard.edu/annotated-transformer/, accessed: 2025-07-30.

[6] "Tiktoken," https://github.com/openai/tiktoken, accessed: 2025-07-30.

[7] "Gurobi," https://www.gurobi.com/ , accessed: 2025-07-30.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. [Online]. Available: https://arxiv.org/abs/1412.6980