tnGPS: Discovering Unknown Tensor Network Structure Search Algorithms via Large Language Models (LLMs)

Junhua Zeng[1]* {jh.zenggdut@gmail.com}, **Chao Li[2]*** {chao.li@riken.jp}, **Zhun Sun[3]** {zhun.sun@tohoku.ac.jp}, **Qibin Zhao[2]** {qibin.zhao@riken.jp}, and **Guoxu Zhou[1,4]** {gx.zhou@gdut.edu.cn}

*Equal contribution [1]Guangdong University of Technology, China; [2]RIKEN-AIP, Japan; [3]Tencent Inc., China; [4]Key Laboratory of Intelligent Detection and the Internet of Things in Manufacturing, China.

Vision and Contributions

A **tensor network** (TN) is a mathematical representation of a large, complex object that can be *decomposed* into simpler, interconnected parts.

Representation for complex **quantum systems**
(Orus, Nature Phys. '19)

Tensor-network **learning models**

Discovering faster **matrix multi...**
(AlphaTensor, Fawzi et al., Nat...)

Tensor Network Structure Search (TN-SS):

Various TN models

(Oseledets, 2011)

(Zhao et al, 2016)   *(Tucker, 1966)*

Search for the most suitable TN model for the task

Motivation:

Exploring the **enormous language space** of LLMs for **autonomous TN-SS algorithm discovery**, saving human experts from the labor-intensive algorithm design process and letting them focus on more challenging problems.

"The limits of my language mean the limits of my world."
— from "Tractatus Logico-Philosophicus"
(1889-1951)

In this work, the **main contributions** are:

1. We **propose t̲ensor-n̲etwork-purposed G̲PT-driven s̲tructure search (tnGPS),** a large language model (LLM)- driven automation framework designed to automatically generate novel and effective TN-SS algorithms tailored to specific downstream tasks;

2. Experimental results demonstrate that the algorithms discovered by tnGPS outperform existing TN-SS algorithms on benchmark data.

How Human Experts

Idea Pool: Gather information throug... literature reviews and paper retrieval.

Knowledge Categorization (KC): Refine ideas into knowledge clusters.

Idea Dropout (ID): Filter out less...

```
Function description:
def GenerateSample(history_populations, fitness_scores, best_individual,
    new_individuals_numbers, current_iteration, maximum_iteration, hyper...

    # GenerateSample: Function takes in integer vectors and output integer...

    # Inputs:
    # history_populations: Dictionary. Keys are integer strings from '1' to so...
    # larger value. Each key contains a list of integer numpy vectors.
    # fitness_scores: Dictionary. Keys are same as history_populations.

        : (omitted)

    # hyperparameters: Dictionary. Keys are strings contain the constants
    # used for computation. Default values should be provided using .get().

    # Output:
    # new_individuals: List, len new_individuals_numbers, contains integer
    # numpy vectors. Each vector's len is the same as the len of the vectors
    # history_populations. Furthermore, The elements are within range
    # [1,hyperparameters['code_upperbound']].

    return new_individuals
```

Algorithm 1: . . .
* * *
Algorithm 1 score:
. . . (omitted)
Algorithm N: . . .
* * *
Algorithm N score:
Algorithms 1 to N are implementations of the 'Ger...
implies better performance.

Learning from their results, think about what works
methods with lower scores. You are encouraged to be creative to incorporate novel
ideas but do not simply stack methods together.

```
Function description:
def GenerateSample(history_populations, fitness_scores, best_individual,
    new_individuals_numbers, current_iteration, maximum_iteration, hyper...

    # GenerateSample:

    # Inputs:
    # history_population
    # larger value. Each
    # fitness_scores: Di...

    # hyperparameters:
    # used for computati...

    # Output:
    # new_individuals: L...
    # numpy vectors. E...
```

Algorithm 1: . . .
Algorithm 1 score:

Algorithm 1: # centroid
  : (omitted)
Algorithm M: # centroid

Give me a novel 'GenerateSample' that is methodologically different from the above algorithms. You are encouraged to be creative to incorporate novel ideas but do not simply stack methods together.

...are implementations of the 'GenerateSample' function. A lower score
...rmance.
...esults, think about what works and what doesn't, provide M novel
...scores. You are encouraged to be creative to incorporate novel
...ply stack methods together.

Knowledge Recombination (KR) Prompt

...ncrease their
... parallel
...eas.

...centroid
... (omitted)
...centroid

Give me a novel 'GenerateSample' that is methodologically different from the above algorithms. You are encouraged to be creative to incorporate novel ideas but do not simply stack methods together.

Diversity Injection (DI) Prompt

```
Function description:
def GenerateSample(history_populations, fitness_scores, best_individual,
    new_individuals_numbers, current_iteration, maximum_iteration, hyper...

    # Inputs:
    # history_populations: Dictionary. Keys are integer strings from '1' to so...
    # larger value. Each key contains a list of integer numpy vectors.
    # fitness_scores: Dictionary. Keys are same as history_populations.

        : (omitted)

    # hyperparameters: Dictionary. Keys are strings contain the constants
    # used for computation. Default values should be provided using .get().

    # Output:
    # new_individuals: List, len new_individuals_numbers, contains integer
    # numpy vectors. Each vector's len is the same as the len of...
```

Provide runnable code that has implemented all your ideas (If any part
requires choice, use choices from random). Do not leave any placeholder for
me, all the functionality should be actually implemented. Your response format
<code>
Your code
<code>
Also, you don't need to add any other words.

Algorithm 1: . . .

"Pilots" for the final prompt construction

Numer...

Natural Image Compression Task: Use TN-SS to find compact TNs to represent natural images.

Objective:

$$F(G, \mathbf{r}) = \underbrace{\frac{1}{\epsilon(G, \mathbf{r})}}_{\text{compression ratio (CR)}} + \lambda \cdot \underbrace{\min_{\mathcal{Z} \in TNS(G, \mathbf{r})} \|\mathcal{X} - \mathcal{Z}\|^2 / \|\mathcal{X}\|^2,}_{\text{relative square error (RSE)}}$$

Settings of tnGPS: *four images for training, ten images for testing.*

Gaussian process Compression: an Out-of-Domain Experiment

Ablation Experiments:

	Initialization	Baseline	TNGA	TNLS	GREEDY	TnALE	Ho-1	Ho-2	Ho-3
CCPP	random	2.64	2.36 [1900]	2.50 [1900]	2.60 [850]	2.36 [588]	2.60 [1900]	**2.24** [1600]	2.74 [1300]
MG		3.36	12.69 [8400]	17.25 [7600]	-	-	6.81 [9200]	**3.01** [10000]	27.74 [8400]
CCPP	TT	2.64	2.24 [500]	3.01 [500]	2.24 [21]	2.24 [18]	2.24 [500]	2.24 [500]	2.24 [300]
MG		3.36	3.36 [100]	3.01 [500]	3.01 [64]	3.01 [42]	3.01 [500]	3.01 [1200]	3.01 [2400]

	baseline	tnGPS	KR	II	DI
Objective	0.1558	**0.1102**	0.1308	0.1273	0.1239

The results highlight the importance of the *'KR', 'II',* and *'DI'* components

	baseline	GPT-4	GPT-3.5	Claude-1	Claude-2	Incomplete descriptions
	0.1847	**0.1813**	0.1842	0.1840	0.1834	0.1819

More powerful LLMs like GPT-4 enhance tnGPS performance

Insights gained from the generated algorithms:

tnGPS, can leverage insights gained from the existing algorithms and the embedded knowledge in LLMs for novel algorithm generation
- Non-Markovian searching dynamic
- Inverse annealing for mutation
- Gaussian perturbation mutation
- Best structure crossover

The algorithm discovered by tnGPS implicitly reflects the hidden structure of the data
- Boundary mutation

Ho-1, Ho-2 and Ho-3 are algorithms discovered by tnGPS

Concluding Remarks

- **tnGPS: a LLM-driven framework for discovering new TN-SS algorithms.**

- **tnGPS is designed by prompting LLMs to mimic human experts.**

- **LLMs provide us new ideas of solving more broad tensor problems.**