Journal of Computational Mathematics
Vol.42, No.6, 2024, 1714–1742.
http://www.global-sci.org/jcm
doi:10.4208/jcm.2307-m2022-0233

TENSOR NEURAL NETWORK AND ITS NUMERICAL INTEGRATION*

Yifan Wang and Hehu Xie[1)]
LSEC, NCMIS, Institute of Computational Mathematics, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
Emails: wangyifan@lsec.cc.ac.cn, hhxie@lsec.cc.ac.cn
Pengzhan Jin
School of Mathematical Sciences, Peking University, Beijing 100871, China
Email: jpz@pku.edu.cn

Abstract

In this paper, we introduce a type of tensor neural network. For the first time, we propose its numerical integration scheme and prove the computational complexity to be the polynomial scale of the dimension. Based on the tensor product structure, we develop an efficient numerical integration method by using fixed quadrature points for the functions of the tensor neural network. The corresponding machine learning method is also introduced for solving high-dimensional problems. Some numerical examples are also provided to validate the theoretical results and the numerical algorithm.

Mathematics subject classification: 65N30, 65N25, 65L15, 65B99.
Key words: Tensor neural network, Numerical integration, Fixed quadrature points, Machine learning, High-dimensional eigenvalue problem.

1. Introduction

Partial differential equations (PDEs) appear in many scientific and industrial applications since they can describe physical and engineering phenomena or processes. So far, many types of numerical methods have been developed such as the finite difference method, finite element method, and spectral method for solving PDEs in three spatial dimensions plus the temporal dimension. But there exist many high-dimensional PDEs such as many-body Schrödinger, Boltzmann equations, Fokker-Planck equations, and stochastic PDEs (SPDEs), which are almost impossible to be solved using traditional numerical methods. Recently, many numerical methods have been proposed based on machine learning to solve the high-dimensional PDEs [2, 6, 7, 14, 25, 28, 31, 37, 38]. Among these machine learning methods, neural network-based methods attract more and more attention. Neural networks can be used to build approximations of the exact solutions of PDEs by machine learning methods. The reason is that neural networks can approximate any function given enough parameters. This type of method provides a possible way to solve many useful high-dimensional PDEs from physics, chemistry, biology, engineering, and so on.

Due to its universal approximation property, the fully-connected neural network (FNN) is the most widely used architecture to build the functions for solving high-dimensional PDEs.

* Received October 19, 2022 / Revised version received January 17, 2023 / Accepted July 26, 2023 / Published online December 4, 2023 /
[1)] Corresponding author

There are several types of FNN-based methods such as well-known the deep Ritz [7], deep Galerkin method [37], PINN [31], and weak adversarial networks [38] for solving high-dimensional PDEs by designing different loss functions. Among these methods, the loss functions always include computing high-dimensional integration for the functions defined by FNN. For example, the loss functions of the deep Ritz method require computing the integrations on the high-dimensional domain for the functions which is constructed by FNN. Direct numerical integration for the high-dimensional functions also meets the "curse of dimensionality". Always, the Monte-Carlo method is adopted to do the high-dimensional integration with some types of sampling methods [7, 15]. Due to the low convergence rate of the Monte-Carlo method, the solutions obtained by the FNN-based numerical methods are difficult to obtain high accuracy and stable convergence process. In other words, the Monte-Carlo method decreases computational work in each forward propagation by decreasing the simulation efficiency and stability of the FNN-based numerical methods for solving high-dimensional PDEs.

The CANDECOMP/PARAFAC (CP) tensor decomposition builds a low-rank approximation method and is a widely used way to cope with the curse of dimensionality. The CP method decomposes a tensor as a sum of rank-one tensors which can be considered as the higher-order extensions of the singular value decomposition (SVD) for the matrices. This means the SVD idea can be generalized to the decomposition of the high-dimensional Hilbert space into the tensor product of several Hilbert spaces. The tensor product decomposition has been used to establish low-rank approximations of operators and functions [5, 13, 19, 32]. If we use the low-rank approximation to do the numerical integration, the computational complexity can avoid the exponential dependence on the dimension in some cases [4, 28]. Inspired by CP decomposition, this paper focuses on a special low-rank neural networks structure and its numerical integration. It is worth mentioning that although CP decomposition should be useful to obtain a low-rank approximation, there is no known general result to give the relationship between the rank (hyperparameter p in this paper) and error bounds. For more details, please refer to [17, 23] and numerical investigations [5].

This paper aims to propose a type of tensor neural network (TNN) to build the trial functions for solving high-dimensional PDEs. The TNN is a function being designed by the tensor product operations on the neural networks or by low-rank approximations of FNNs. An important advantage is that we do not need to use Monte-Carlo method to do the integration for the functions which is constructed by TNN. This is the main motivation to design the TNN for high-dimensional PDEs in this paper. We will show, the computational work for the integration of the functions by TNN is only a polynomial scale of the dimension, which means the TNN overcomes the "curse of dimensionality" in some sense for solving high-dimensional PDEs.

An outline of the paper goes as follows. In Section 2, we introduce the way to build TNN. The numerical integration method for the functions constructed by TNN is designed in Section 3. Section 4 is devoted to proposing the TNN-based machine learning method for solving the high-dimensional eigenvalue problem with the numerical integration method. Some numerical examples are provided in Section 5 to show the validity and efficiency of the proposed numerical methods in this paper. Some concluding remarks are given in the last section.

2. Tensor Neural Network

2.1. Architecture of tensor neural network

In this section, we introduce the TNN and its approximation property. Without loss of