EXPRESSIVE POWER OF TENSOR-TRAIN NETWORKS WITH EQUAL TT-CORES

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks show their extreme efficiency in solving a wide range of practical problems. Despite this, a theoretical explanation of this phenomenon is only beginning to emerge in scientific research. For some special kinds of deep neural networks, it has been shown that depth is the key to efficiency. In particular, it has recently been shown that Tensor-Train networks, i.e. recurrent neural networks, each layer of which implements a bilinear function, are exponentially more expressive than shallow networks. However, in practice, recurrent neural networks with identical layers are used, the analogue of which are Tensor-Train networks with equal TT-cores. For this class of networks, the analogous result was not proved, but formulated as a Hypothesis. We prove this Hypothesis and thus close the question of exponential expressivity of Tensor-Train networks with equal TT-cores. We also conduct a series of numerical experiments to confirm the theoretical result.

1 INTRODUCTION

Deep neural networks solve many practical tasks both in computer vision via Convolutional Neural Networks (CNNs) and in audio and text processing via Recurrent Neural Networks (RNNs). Although many practical problems are solved using deep neural networks (DNNs), the theoretical justification of their effectiveness has not been fully studied.

One approach for justifying the power of depth is to show that deep networks can efficiently express functions that would require shallow networks to have super-polynomial size. Early results related to this approach Hastad (1986), Hastad & Goldmann (1991), Delalleau & Bengio (2011), Martens & Medabalimi (2014) consider specific network architectures that are not commonly used in practice.

In 2016, Nadav Cohen, Or Sharir, and Amnon Shashua published a paper Cohen et al. (2016) in which they proposed a deep network architecture based on arithmetic circuits (also known as Sum-Product networks) that inherently uses three features of convolutional network architecture: locality, sharing and pooling. They also showed a connection between the proposed architecture and Hierarchical Tucker decomposition of the parameter tensor $\mathbf{A}^y$ in the following hypotheses space:

$$h_y(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_d) = \sum_{i_1,\ldots,i_d=1}^{m} \mathbf{W}_y^{i_1\ldots i_d} \cdot f_{\theta_{i_1}}(\boldsymbol{x}_1) \cdot \ldots \cdot f_{\theta_{i_d}}(\boldsymbol{x}_d),$$

where h_y is a score function for class label y; f_θ is a representation function, selected from a parametric family $\mathcal{F} = \{f_\theta : \mathbb{R}^n \to \mathbb{R}^m\}_{\theta\in\Theta}$; $\mathbf{W}_y^{i_1\ldots i_d}$ is a parameter tensor (see formal definition in Chapter 2). Using this connection the authors proved that deep CNNs are exponentially more expressive than shallow networks (see papers Cohen et al. (2016) and Cohen & Shashua (2016) for details).

In 2018, Valentin Khrulkov, Alexander Novikov and Ivan Oseledets showed in the paper Khrulkov et al. (2018) the connection between Tensor-Train tensor decomposition and deep recurrent-type neural networks, and used this connection to prove that deep recurrent-type neural networks in which all layers have their own parameters are exponentially more expressive than shallow neural networks. The authors also hypothesized that their results are also true for traditional RNNs, in which layers

share parameters. The research towards this hypothesis is particularly interesting because it refers to the architectures used to solve practical problems.

Relevance. Recurrent neural networks are widely used in audio and text processing. Since the theoretical result that deep recurrent-type neural networks in which all layers have their own parameters are exponentially more expressive than shallow neural networks is not applicable to traditional RNNs, the problem of theoretical estimation of the expressive power of traditional RNNs remains open and the solution of which is of immediate interest.

Scientific novelty. We formulate and prove the expressive power theorem for the Tensor-Train decomposition with equal TT-cores (see Theorem 2 in Chapter 2). We also confirm the obtained theoretical result with original numerical experiments (see Subsections 3.1, 3.2).

2 MAIN RESULT

In the following subsection we give a rigorous definition of hypotheses space considered in tensor analysis.

2.1 DEFINITION OF THE HYPOTHESES SPACE

We consider the task of classification of a collection of vectors $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d), \boldsymbol{x}_i \in \mathbb{R}^n$, into one of the categories $\mathbb{Y} := \{1, \ldots, Y\}$. Representing instances as a collection of vectors is natural in many applications. We assume that the components $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d$ of an instance X can be transformed by a function f_θ from the parametric family $\mathcal{F} = \{f_\theta : \mathbb{R}^n \to \mathbb{R}^m\}_{\theta \in \Theta}$ into vectors $f_\theta(\boldsymbol{x}_1), \ldots, f_\theta(\boldsymbol{x}_d)$, which we call *lower-dimensional representations* of $\{\boldsymbol{x}_k\}_{k=1}^d$. As usual, an instance X will be assigned to class y if and only if the value $h_y(X)$ of *score function* h_y for class label y is maximal among the values $h_1(X), \ldots, h_Y(X)$ of score functions $h_1, \ldots, h_Y$ for all class labels. We define our hypotheses space by the following formula for the score function h_y for a class label y:

$$h_y(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_d) = \sum_{i_1, \ldots, i_d = 1}^{m} \mathbf{W}_y^{i_1 \ldots i_d} \cdot f_{\theta_{i_1}}(\boldsymbol{x}_1) \cdot \ldots \cdot f_{\theta_{i_d}}(\boldsymbol{x}_d), \tag{1}$$

where $\mathbf{W}_y$ is a *coefficient tensor* of order d and dimension m in each mode (see the motivation for this definition of hypotheses space in Cohen et al. (2016)).

Denote $[n] := \{1, \ldots, n\}$ for any positive integer n. Let us recall some known tensor decompositions.

2.2 TENSOR DECOMPOSITIONS REMINDER

Canonical decomposition, also known as *CANDECOMP/PARAFAC* or *CP-decomposition* for short, of a tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ is defined as follows

$$\mathbf{X}^{i_1 i_2 \ldots i_d} = \sum_{\alpha=1}^{r} \boldsymbol{v}_{1,\alpha}^{i_1} \cdot \boldsymbol{v}_{2,\alpha}^{i_2} \cdot \ldots \cdot \boldsymbol{v}_{d,\alpha}^{i_d}, \qquad \boldsymbol{v}_{i,\alpha} \in \mathbb{R}^{n_i}. \tag{2}$$

Canonical rank, or *CP-rank* for short, of a tensor $\mathbf{X}$ is the minimal r such that canonical decomposition of $\mathbf{X}$ with r summands exists.

Tensor-Train decomposition, or *TT-decomposition* for short, of a tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ is defined as follows

$$\mathbf{X}^{i_1 i_2 \ldots i_d} = \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \ldots \sum_{\alpha_{d-1}=1}^{r_{d-1}} \mathbf{G}_1^{i_1 \alpha_1} \cdot \mathbf{G}_2^{\alpha_1 i_2 \alpha_2} \cdot \ldots \cdot \mathbf{G}_d^{\alpha_{d-1} i_d}, \tag{3}$$

where $\mathbf{G}_k \in \mathbb{R}^{r_{k-1} \times n_k \times r_k},\ k \in [d]$ $(r_0 := 1, r_d := 1)$, are tensors which we call *TT-cores*. Tensor-Train decomposition was introduced by Ivan Oseledets in Oseledets (2011) and was later studied in more depth in Oseledets & Tyrtyshnikov (2010).

Tensor-Train ranks, or *TT-ranks* for short, of a tensor $\mathbf{X}$ is the element-wise minimal ranks $\boldsymbol{r} = (r_1, r_2, \ldots, r_{d-1})$ such that Tensor-Train decomposition of $\mathbf{X}$ with such $r_1, r_2, \ldots, r_{d-1}$ exists.

2.3 Connection between Tensor-Train decomposition and RNNs

In paper Khrulkov et al. (2018), Khrulkov, V. et al. showed that a recurrent-type neural network with d layers, each implementing a bilinear function, lies in the hypotheses space defined by formula 1, with the parameters of each of the d layers being exactly equal to the TT-cores $\mathbf{G}_1, \ldots, \mathbf{G}_d$ of the corresponding Tensor-Train decomposition of the coefficient tensor $\mathbf{W}_y$. Moreover, the TT-ranks $r_1, r_2, \ldots, r_{d-1}$ of the corresponding Tensor-Train decomposition of $\mathbf{W}_y$ are equal to the dimensions of the hidden states after, respectively, the first layer, the second layer and so on up to the $(d-1)$-th layer. Further we call such networks *Tensor-Train networks*.

Let us denote $\boldsymbol{n} := (n_1, n_2 \ldots n_d)$. Set of all tensors $\mathbf{X}$ with mode sizes $\boldsymbol{n}$ representable in TT-format with

$$\operatorname{rank}_{TT} \mathbf{X} \leqslant \boldsymbol{r},$$

for some vector of positive integers $\boldsymbol{r}$ (inequality is understood entry-wise) forms an *irreducible algebraic variety* (see details in Shafarevich & Hirsch (1994)), which we denote by $\mathbb{M}_{\boldsymbol{n},\boldsymbol{r}}$.

One of the main results of paper Khrulkov et al. (2018) is the theorem about the lower bound on the CP-rank for almost all tensors from $\mathbb{M}_{\boldsymbol{n},\boldsymbol{r}}$.

Theorem 1 (Valentin Khrulkov, Alexander Novikov, Ivan Oseledets; 2018)**.** *Suppose that $d = 2k$ is even. Define the following set*

$$\mathbb{B} := \left\{ \mathbf{X} \in \mathbb{M}_{\boldsymbol{n},\boldsymbol{r}} : \operatorname{rank}_{CP} \mathbf{X} < q^{\frac{d}{2}} \right\},$$

where $q = \min\{n, r\}$.

Then

$$\mu(\mathbb{B}) = 0,$$

where μ is the standard Lebesgue measure on $\mathbb{M}_{\boldsymbol{n},\boldsymbol{r}}$.

2.4 Tensor-Train networks with equal TT-cores

A subset of $\mathbb{M}_{\boldsymbol{n},\boldsymbol{r}}$ consisting only of those tensors whose inner cores are equal, or in other words $\mathbf{G}_2 = \mathbf{G}_3 = \ldots = \mathbf{G}_{d-1}$, is denoted by $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$.

Remark 1. *The condition $\mathbf{G}_2 = \mathbf{G}_3 = \ldots = \mathbf{G}_{d-1}$ implies that $n_2 = n_3 = \ldots = n_{d-1}$ and $r_1 = r_2 = \ldots = r_{d-1}$, i.e. $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$ is defined only for such $\boldsymbol{n}, \boldsymbol{r}$ that $\boldsymbol{n} = (n_{first}, \underbrace{n_{inner}, \ldots, n_{inner}}_{d-2 \text{ times}}, n_{last})$ and $\boldsymbol{r} = (\underbrace{r, r, \ldots, r}_{d-1 \text{ times}})$.*

Let us denote $\mathcal{G}[n_{first}, n_{inner}, n_{last}, r] := \mathbb{R}^{n_{first} \times r} \times \mathbb{R}^{r \times n_{inner} \times r} \times \mathbb{R}^{r \times n_{last}}$. Then

$$\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}} = \left\{ \mathbf{G}_{first} \circledast \underbrace{\mathbf{G}_{inner} \circledast \ldots \circledast \mathbf{G}_{inner}}_{d-2 \text{ times}} \circledast \mathbf{G}_{last} : (\mathbf{G}_{first}, \mathbf{G}_{inner}, \mathbf{G}_{last}) \in \mathcal{G}[n_{first}, n_{inner}, n_{last}, r] \right\},$$

where $\circledast$ is a tensor dot product.

A similar to Theorem 1 result for $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$ is of particular interest because it is about Tensor-Train networks with equal inner layers, which is closest to traditional recurrent neural networks. We claim to have proved this result, which is formulated as the following theorem.

Theorem 2. *Suppose that $d = 2k$ is even. Define the following set*

$$\mathbb{B} := \left\{ \mathbf{X} \in \mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}} : \operatorname{rank}_{CP} \mathbf{X} < q^{\frac{d}{2}} \right\},$$

where $q = \min\{n, r-1\}$.

Then

$$\mu(\mathbb{B}) = 0,$$

where μ *is the standard Lebesgue measure on* $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$.

To prove this theorem we need to formulate Lemma 3, which is proved in Khrulkov et al. (2018).

Lemma 3 (rank of matricization)**.** *Let* $\mathbf{X}^{i_1 i_2 \ldots i_d}$ *and* $\operatorname{rank}_{CP} \mathbf{X} = r$. *Then for any matricization* $\mathbf{X}^{(s,t)}$ *we have* $\operatorname{rank} \mathbf{X}^{(s,t)} \leqslant r$, *where the ordinary matrix rank is assumed.*

Proof. Our proof is based on applying Lemma 3 to a particular matricization of $\mathbf{X}$. Namely, we would like to show that for $s = \{1, 3, \ldots, d-1\}$, $t = \{2, 4, \ldots, d\}$ the following set

$$\mathbb{B}^{(s,t)} := \left\{ \mathbf{X} \in \mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}} : \operatorname{rank} \mathbf{X}^{(s,t)} \leqslant q^{\frac{d}{2}} - 1 \right\}$$

has measure 0. Indeed, by Lemma 3 we have

$$\mathbb{B} \subset \mathbb{B}^{(s,t)},$$

so if $\mu(\mathbb{B}^{(s,t)}) = 0$ then $\mu(\mathbb{B}) = 0$ as well. Note that $\mathbb{B}^{(s,t)}$ is an algebraic subset of $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$ given by the conditions that the determinants of all $q^{\frac{d}{2}} \times q^{\frac{d}{2}}$ submatrices of $\mathbf{X}^{(s,t)}$ are equal to 0. Thus to show that $\mu(\mathbb{B}^{(s,t)}) = 0$ we need to find at least one $\mathbf{X}$ such that $\operatorname{rank} \mathbf{X}^{(s,t)} \geqslant q^{\frac{d}{2}}$. This follows from the fact that because $\mathbb{B}^{(s,t)}$ is an algebraic subset of the irreducible algebraic variety $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$, it is either equal to $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$ or has measure 0, as was explained before.

One way to construct such tensor is as follows. Let us define the following tensors:

$$\mathbf{G}_1^{i_1 \alpha_1} = [i_1 = \alpha_1], \quad \mathbf{G}_1 \in \mathbb{R}^{1 \times n \times r}$$

$$\mathbf{G}_k^{\alpha_{k-1} i_k \alpha_k} = \begin{cases} [\alpha_{k-1} = i_k] \cdot [\alpha_k = q+1], & \text{if } \alpha_{k-1} \leqslant q \\ [i_k = \alpha_k], & \text{if } \alpha_{k-1} = q+1 \\ 0, & \text{if } \alpha_{k-1} > q+1 \end{cases}, \quad \mathbf{G}_k \in \mathbb{R}^{r \times n \times r}, \quad k = 2, 3, \ldots, d-1$$

$$\mathbf{G}_d^{\alpha_{d-1} i_d} = [\alpha_{d-1} = i_d], \quad \mathbf{G}_d \in \mathbb{R}^{r \times n \times 1},$$

where $[\,\cdot\,]$ is the Iverson bracket notation.

The TT-ranks of the tensor $\mathbf{X}$ defined by the TT-cores are equal to $\operatorname{rank}_{TT} \mathbf{X} = (r, r, \ldots, r)$.

Lemma 4 (structure of non-zero summands)**.** *Consider* $(i_1, i_2, \ldots, i_d) \in [q]^d$ *and* $(\alpha_1, \alpha_2, \ldots, \alpha_{d-1}) \in [r]^{d-1}$ *such that*

$$\mathbf{G}_1^{i_1 \alpha_1} \cdot \ldots \cdot \mathbf{G}_d^{\alpha_{d-1} i_d} \neq 0.$$

Then

$$\alpha_k = \begin{cases} i_k, & \textit{if } k \textit{ is odd} \\ q+1, & \textit{if } k \textit{ is even} \end{cases} \qquad \textit{for any } k \in [d-1].$$

Proof. Let us prove the lemma by induction over k.

- *Base of the induction:* $k = 1$.

  Since $\mathbf{G}_1^{i_1 \alpha_1} = [i_1 = \alpha_1] \neq 0$, then $\alpha_1 = i_1$.

- *The induction step:* $k \to k+1$, $k \in [d-2]$.

  If k is odd, then $\alpha_k = i_k \in [q]$. Hence $\mathbf{G}_{k+1}^{\alpha_k i_{k+1} \alpha_{k+1}} = [\alpha_k = i_{k+1}] \cdot [\alpha_{k+1} = q+1]$. Since $\mathbf{G}_{k+1}^{\alpha_k i_{k+1} \alpha_{k+1}} \neq 0$, then $[\alpha_{k+1} = q+1] \neq 0$, so $\alpha_{k+1} = q+1$.

  If k is even, then $\alpha_k = q+1$. Hence $\mathbf{G}_{k+1}^{\alpha_k i_{k+1} \alpha_{k+1}} = [i_{k+1} = \alpha_{k+1}]$. Since $\mathbf{G}_{k+1}^{\alpha_k i_{k+1} \alpha_{k+1}} \neq 0$, then $[i_{k+1} = \alpha_{k+1}] \neq 0$, so $\alpha_{k+1} = i_{k+1}$.

□

Lets consider the following matricization of the tensor $\mathbf{X}$

$$\mathbf{X}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)}$$

The following identity holds true for any values of indices such that $i_k = 1,\ldots,q$, $k = 1,\ldots,d$.

$$\mathbf{X}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)} = \sum_{\alpha_1,\ldots,\alpha_{d-1}} \mathbf{G}_1^{i_1\alpha_1}\ldots\mathbf{G}_d^{\alpha_{d-1}i_d} =$$

$$\text{(by Lemma 4)} = \mathbf{G}_1^{i_1 i_1}\mathbf{G}_2^{i_1 i_2 (q+1)}\mathbf{G}_3^{(q+1) i_3 i_3}\ldots\mathbf{G}_d^{i_{d-1}i_d} =$$

$$\Big([i_1 = i_1]\Big)\cdot\Big([i_1 = i_2]\cdot[q+1 = q+1]\Big)\cdot\Big([i_3 = i_3]\Big)\cdot\ldots\cdot\Big([i_{d-1} = i_d]\Big) =$$

$$[i_1 = i_2]\cdot[i_3 = i_4]\cdot\ldots\cdot[i_{d-1} = i_d].$$

We obtain that

$$\mathbf{X}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)} = [i_1 = i_2]\cdot[i_3 = i_4]\cdot\ldots\cdot[i_{d-1} = i_d] = \boldsymbol{I}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)},$$

where $\boldsymbol{I}$ is the identity matrix of size $q^{\frac{d}{2}} \times q^{\frac{d}{2}}$.

To summarize, we found an example of a tensor $\mathbf{X}$ such that $\operatorname{rank}_{TT}\mathbf{X} \leqslant \boldsymbol{r}$ and the matricization $\mathbf{X}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)}$ has a submatrix being equal to the identity matrix of size $q^{\frac{d}{2}} \times q^{\frac{d}{2}}$, and hence $\operatorname{rank}\mathbf{X}^{(i_1,i_3,\ldots,i_{d-1}),(i_2,i_4,\ldots,i_d)} \geqslant q^{\frac{d}{2}}$. This means that the canonical $\operatorname{rank}_{CP}\mathbf{X} \geqslant q^{\frac{d}{2}}$ which concludes the proof. □

Theorem 2 gives a lower bound for almost all tensors of order d with modes $\boldsymbol{n}$ with Tensor-Train rank at most r. Using the connection between Tensor-Train decomposition and recurrent neural networks presented in Khrulkov et al. (2018), we have that traditional RNNs, which corresponds to tensors from $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$, are exponentially more expressive than shallow neural networks.

3 Numerical experiments

We distinguish two different purposes of numerical experiments:

- to confirm numerically the result of the Theorem 2, i.e., to generate a random tensor from $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$, evaluate numerically its CP-rank and compare it with the theoretical bound (see Subsection 3.1);
- to show that Tensor-Train network with equal inner TT-cores copes with classification tasks on real data.

All performed numerical experiments are available via the link to the Github repository.

3.1 Numerical verification of Theorem 2

Any experiment in this section consists of several steps:

1. fixing the parameters n_{first}, n_{inner}, n_{last}, r and the upper bound $d_{\max}$ for depth d;
2. selection of "typical" TT-cores $\mathbf{G}_{\text{first}}$, $\mathbf{G}_{\text{inner}}$, $\mathbf{G}_{\text{last}}$ from $\mathcal{G}[n_{\text{first}}, n_{\text{inner}}, n_{\text{last}}, r]$ (see Remark 1);
3. numerical evaluation of the CP-rank of the tensors from $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$ generated by the TT-cores $\mathbf{G}_{\text{first}}$, $\mathbf{G}_{\text{inner}}$, $\mathbf{G}_{\text{last}}$ for each d from $\{1,\ldots,d_{\max}\}$;
4. aggregation of the results and plotting of the graph.

Let us fix $n_{\text{first}} = n_{\text{inner}} = n_{\text{last}} = 2$, $r = 3$. Theorem 2 gives us the following lower bound on CP-rank of almost all tensors from $\mathbb{M}^{eq}_{\boldsymbol{n},\boldsymbol{r}}$: $q^{\frac{d}{2}} = \min\{n_{\text{first}}, n_{\text{inner}}, n_{\text{last}}, r-1\}^{\frac{d}{2}} = 2^{\frac{d}{2}}$.

Since the standard Lebesgue measure on $\mathcal{G}[n_{\text{first}}, n_{\text{inner}}, n_{\text{last}}, r]$ is not a probability, we will choose TT-cores $\mathbf{G}_{\text{first}}$, $\mathbf{G}_{\text{inner}}$, $\mathbf{G}_{\text{last}}$ using arbitrary distribution on $\mathcal{G}[n_{\text{first}}, n_{\text{inner}}, n_{\text{last}}, r]$, for example, standard normal.

A direct way to estimate the canonical rank of tensor $\mathbf{X}$ is to search for a low-rank approximation of tensor $\mathbf{X}$ by increasing the CP-rank of an approximation. If at some CP-rank the low-rank approximation differs from tensor $\mathbf{X}$ by a negligible error, then the number of summands in the low-rank approximation will be equal to the CP-rank of $\mathbf{X}$. The search for the low-rank approximation was performed both using gradient descent by minimising the Frobenius norm of the difference between $\mathbf{X}$ and the approximation, and using a function `cpd` (Canonical polyadic decomposition) from Tensorlab Vervliet et al. (2016) which is a MATLAB Inc. (2022) package.

The results of these experiments are shown in the following graph (see Figure 1).

The graph shows that the theoretical lower bound derived from Theorem 2 is lower than estimated CP-rank of chosen tensor for all considered values of depth d.

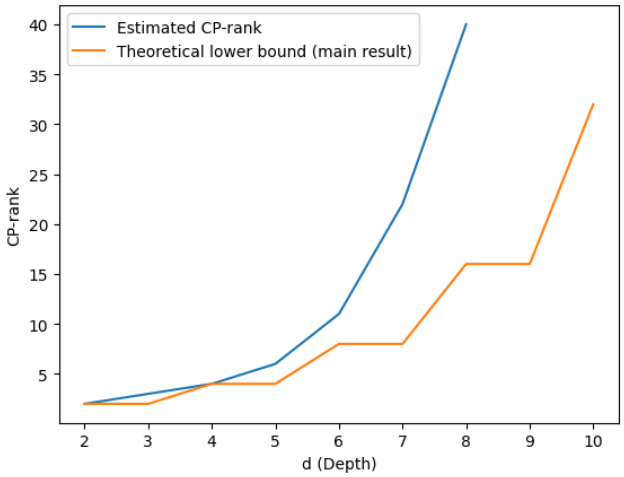

Figure 1: The graph compares the theoretical lower bound on CP-rank (orange line) and the numerical CP-rank estimate of the random tensor from $\mathbb{M}^{eq}_{2,3}$ (blue line).

3.2 EXPERIMENTS ON THE MNIST DATASET

In this section we consider standard computer vision datasets MNIST which is a collection of 70000 handwritten digits.

We have implemented Tensor-Train networks with both different and the same inner TT-cores using PyTorch Paszke et al. (2019). We use Adam optimizer with batch size 64 and learning rate $5\text{e}{-4}$. Each picture of size 28×28 pixels is split into 16 non-overlapping 7×7 patches, whose low-dimensional representations are alternately fed to the Tensor-Train network of depth 16.

The training process of these networks was as follows: in the initial stage, we added `BatchNorm1d` layer after each recurrent layer of the Tensor-Train network and trained such a network for 20 epochs; in the second stage, we removed the normalisation layers and further trained the Tensor-Train network for four epochs. This two-step approach avoided the problem of stopping training at the initial stage.

For MNIST, both Tensor-Train networks and Tensor-Train networks with equal cores show reasonable performance (see Table 1).

Table 1: The results of training Tensor-Train networks on MNIST dataset during 20+4 epochs.

	TT-net with different cores	TT-net with equal cores
Train accuracy	98.5	96.3
Test accuracy	97.3	95.7

3.3 EXPERIMENTS ON THE CIFAR-10 DATASET

In this section we consider computer vision datasets CIFAR-10 which is a collection of 60000 32×32 color images in 10 different classes.

We use Adam optimizer with batch size 64 and learning rate $5\text{e}{-4}$. Each picture of size 32×32 pixels is split into 16 non-overlapping 8×8 patches, whose low-dimensional representations are alternately fed to the Tensor-Train network of depth 16.

The training process of these networks was as follows: in the initial stage, we added `BatchNorm1d` layer after each recurrent layer of the Tensor-Train network and trained such a network for 40 epochs; in the second stage, we removed the normalisation layers and further trained the Tensor-Train network for ten epochs. This two-step approach avoided the problem of stopping training at the initial stage.

Despite the fact that we failed to train Tensor-Train networks with high accuracy to distinguish classes, a network with equal TT-cores is not worse in accuracy than a network with different TT-cores, but even better (see Table 2).

Table 2: The results of training Tensor-Train networks on CIFAR-10 dataset during 40+10 epochs.

	TT-net with different cores	TT-net with equal cores
Train accuracy	16.1	37.3
Test accuracy	15.9	37.6

ACKNOWLEDGMENTS

I would like to thank Ivan Oseledets, Sergey Matveev for helpful discussion.

REFERENCES

Nadav Cohen and Amnon Shashua. Convolutional rectifier networks as generalized tensor decompositions. *In International Conference on Machine Learning*, pp. 955–963, 2016.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. *In Conference on Learning Theory*, pp. 698–728, 2016.

Olivier Delalleau and Yoshua Bengio. Shallow vs. deep sum-product networks. *In Advances in Neural Information Processing Systems*, pp. 666–674, 2011.

Johan Hastad. Almost optimal lower bounds for small depth circuits. *In Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 6–20, 1986.

Johan Hastad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, pp. 1(2):113–129, 1991.

The MathWorks Inc. Matlab version: 9.14.0 (r2023a), 2022. URL `https://www.mathworks.com`.

Valentin Khrulkov, Alexander Novikov, and Ivan Oseledets. Expressive power of recurrent neural networks. *arXiv preprint arXiv:1711.00811*, 2018.

James Martens and Venkatesh Medabalimi. On the expressive efficiency of sum product networks. *arXiv preprint arXiv:1411.7717*, 2014.

I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011. doi: 10.1137/090752286. URL `https://doi.org/10.1137/090752286`.

Ivan Oseledets and Eugene Tyrtyshnikov. Tt-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*, 432(1):70–88, 2010. ISSN 0024-3795. doi: https://doi.org/10.1016/j.laa.2009.07.024. URL `https://www.sciencedirect.com/science/article/pii/S0024379509003747`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Igor Rostislavovich Shafarevich and Kurt Augustus Hirsch. *Basic algebraic geometry, vol. 2*. Springer, 1994.

N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer. Tensorlab 3.0, Mar. 2016. URL `https://www.tensorlab.net`. Available online.