HIERARCHICAL TUCKER TENSOR REGRESSION: APPLICATION TO BRAIN IMAGING DATA ANALYSIS

Ming Hou, Brahim Chaib-draa

Laval University, Canada

ABSTRACT

We present a novel generalized linear tensor regression model, which takes tensor-variate inputs as covariates and finds low-rank (almost) best approximation of regression coefficient arrays using hierarchical Tucker decomposition. With limited sample size, our model is highly compact and extremely efficient as it requires only $\mathcal{O}(dr^3 + dpr)$ parameters for order d tensors of mode size p and rank r, which avoids the exponential growth in d, in contrast to $\mathcal{O}(r^d + dpr)$ parameters of Tucker regression modeling. Our model also maintains the flexibility like classical Tucker regression by allowing distinct ranks on different modes according to a dimension tree structure. We evaluate our new model on both synthetic data and real-life MRI images to show its effectiveness.

Index Terms— Tensor Regression, Hierarchical Tucker Decomposition, Brain Imaging, Magnetic Resonance Image.

1. INTRODUCTION

In recent years, tensor-variate regression approaches have attracted increasing interest in real-world applications such as computer vision [1, 2], signal processing [3, 4, 5] and medical imaging data analysis [6, 7]. These regression tasks involving higher order tensors pose great challenges in two aspects. First, the ultrahigh dimensionality of tensor input like 3D or 4D image results in gigantic number of parameters. Second, the underlying complex multiway structure of image covariate cannot be treated naively by turning image into a vector. To this end, several cutting-edge methods have been proposed based on the low-rank approximation of coefficient tensors. Among them, Zhou et al. [6] imposed CANDECOMP/PARAFAC (CP) decomposition [8, 9] on the input space; the resulting model is simple but not flexible. Li et al. [7] employed Tucker format [10] instead to overcome the inflexibility of CP regression, as it can admit different ranks on different modes. However, for larger order tensors, Tucker based model is ineffective and not scalable since the parameter size of core tensor grows exponentially with the order of tensor, which is known as the curse of dimensionality [11].

In this article, we present a highly compact, flexible and scalable tensor regression framework based on the hierarchical Tucker ($\mathcal{H}$-Tucker) decomposition [12, 13]. Specifically, we apply the $\mathcal{H}$-Tucker decomposition to approximate the coefficient tensor, leading to a sparse representation of fewer parameters. On one hand, our approach shares the advantage of CP based model in that its parameter complexity is free from exponential dependence on the order d. On the other hand, our approach preserves the flexibility like Tucker model by allowing distinct ranks according to a dimension tree structure. Then we develop a scalable block relaxation algorithm to conduct the parameter estimation. Finally, we impose $\ell 1$ penalty on the transfer matrix of the model to promote the sparsity. Empirical studies on both synthetic simulation and real-life task demonstrate the effectiveness of our approach.

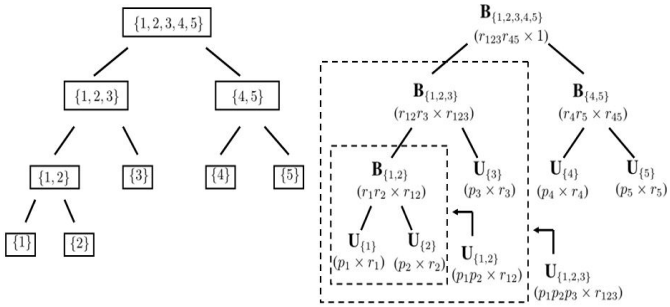

Fig. 1: A balanced dimension tree $\mathcal{T}$ (left) and $\mathcal{H}$-Tucker format (right).

2. HIERARCHICAL TUCKER DECOMPOSITION

The $\mathcal{H}$-Tucker is a novel structured format which efficiently represents a higher order tensor by means of subspace approximation in a multi-level fashion [12, 13]. In general, the idea is to recursively separate the modes of the tensor, which leads to a binary *dimension tree* $\mathcal{T}$ with a subset of modes $t \subset \{1, ..., d\}$ at each node. In this format, one can find the optimal subspace corresponding to each node t in $\mathcal{T}$. A balanced tree of $5th$ order tensor is shown in Fig.1. The set $\mathcal{L}(\mathcal{T})$ contains all the leaves, each with a singleton mode. $\mathcal{N}(\mathcal{T})$ contains the interior nodes each with a subset of all modes. For $t \in \mathcal{N}(\mathcal{T})$, it is a disjoint union of left and right children $t = t_l \cup t_r$. We employ the balanced $\mathcal{T}$ in this paper.

Unlike Tucker decomposition, $\mathcal{H}$-Tucker applies a hierarchy of matricizations $\mathbf{X}^{(t)}$ of a tensor $\mathcal{X}$ according to the

structure of $\mathcal{T}$. For each $t \in \mathcal{T}$, the $\mathbf{X}^{(t)}$ can be obtained by merging all the modes in subset t into the row indices and all the rest modes into the column indices. We then look for a set of *factor matrices* $\mathbf{U}_t$, columns of which span the column space of matrix $\mathbf{X}^{(t)}$, to represent the tensor $\mathcal{X}$. The dimension of $\mathbf{X}^{(t)}$, namely the rank $r_t = rank(\mathbf{X}^{(t)})$, is equal to the number of columns of $\mathbf{U}_t$. The collection of these ranks at all the nodes $(r_t)_{t\in\mathcal{T}}$ is defined as ht-rank of $\mathcal{X}$, i.e., the ht-rank in above example can be denoted as (1-$r_{123}r_{45}$-$r_{12}r_3r_4r_5$-r_1r_2). In contrast, the d-rank is defined as the column rank of mode-d matricization for Tucker. Let us now introduce the following nestedness property which is due to [14].

Proposition 1 Let $\mathcal{X} \in \mathbb{R}^{p_1\times\cdots\times p_d}$, $t = t_l \cup t_r$ and $t_l \cap t_r = \emptyset$, then $span(\mathbf{X}^{(t)}) \subset span(\mathbf{X}^{(t_r)} \otimes \mathbf{X}^{(t_l)})$.

This property suggests a relation of subspaces of matricizations between the parent and children, which enables us to establish the link between the corresponding factor matrices using a so-called *transfer matrix* $\mathbf{B}_t$ in

$$\mathbf{U}_t = (\mathbf{U}_{t_l} \otimes \mathbf{U}_{t_r})\mathbf{B}_t, \tag{1}$$

where $\mathbf{B}_t \in \mathbb{R}^{r_{t_l}r_{t_r}\times r_t}$ and r_{t_l}, r_{t_r}, r_t are the ht-rank at nodes t_l, t_l and t, respectively. Starting from the leaf singletons, the construction of $\mathcal{H}$-Tucker proceeds by applying equation (1) recursively until the root is reached. Note that r_t is fixed to 1 at the root node. As a result, this implies that the tensor $\mathcal{X}$ can be completely parameterized just by the transfer matrices $\{\mathbf{B}_t\}_{t\in\mathcal{N}(\mathcal{T})}$ and factor matrices $\{\mathbf{U}_t\}_{t\in\mathcal{L}(\mathcal{T})}$. Therefore, the overall complexity for storage is bounded by $\mathcal{O}(dr^3 + dpr)$, where $p := \max\{p_i\}_{i=1}^d$ and $r := \max\{r_t\}_{t\in\mathcal{T}}$.

3. $\mathcal{H}$-TUCKER TENSOR REGRESSION MODEL

Our modeling strategy for tensor regression is based on the framework of generalized linear model (GLM) [15] whose response variable is from an exponential family distribution. As previously stated, $\mathcal{H}$-Tucker is developed to keep all the merits of tensor subspace representation introduced via Tucker, while avoiding the exponential increase in the parameter size of the coefficient tensor. Inspired by these benefits, we naturally apply the $\mathcal{H}$-Tucker decomposition to the input space. That is, given a dimension tree $\mathcal{T}$ and ht-rank $\underline{r} = (r_t)_{t\in\mathcal{T}}$, the coefficient tensor $\mathcal{B}$ is assumed to follow a $\mathcal{H}$-Tucker($\underline{r}$) decomposition. The linear systematic part of generalized linear $\mathcal{H}$-Tucker tensor regression model can be obtained by

$$g(\mu) = \eta = \alpha^T\mathbf{z} + < \mathcal{B}, \mathcal{X} > = \alpha^T\mathbf{z} + < vec(\mathcal{B}), vec(\mathcal{X}) >, \tag{2}$$

where η is a linear combination of unknown parameters; μ is expected response and $g(\cdot)$ is the link function. α denotes the regular vector coefficient and $\mathbf{z}$ corresponds to the regular vector predictor. For the root node, we have $vec(\mathcal{B}) = \mathbf{U}_{root}$ $(r_{root} = 1)$ and by nestedness property (1)

$$\eta = \alpha^T\mathbf{z} + < (\mathbf{U}_{root_r} \otimes \mathbf{U}_{root_l})\mathbf{B}_{root},\ vec(\mathcal{X}) > . \tag{3}$$

For other inner nodes $t \in \mathcal{N}(\mathcal{T})$, we then recursively process the nestedness relation by replacing the $\mathbf{U}_t$ with its corresponding transfer matrix $\mathbf{B}_t$ and its children $\mathbf{U}_{t_l}$ and $\mathbf{U}_{t_r}$. This procedure stops when we reach all the leaf nodes $t \in \mathcal{L}(\mathcal{T})$. Finally, the resulting model becomes

$$g(\mu) = \eta = \alpha^T\mathbf{z} + < (\bigotimes_{t\in\mathcal{L}(\mathcal{T})^L} \mathbf{U}_t)(\bigotimes_{t\in\mathcal{L}(\mathcal{T})^{L-1}} \mathbf{U}_t \otimes \bigotimes_{t\in\mathcal{N}(\mathcal{T})^{L-1}} \mathbf{B}_t)\cdots(\bigotimes_{t\in\mathcal{T}^l} \mathbf{B}_t)\cdots(\mathbf{B}_{root}),\ vec(\mathcal{X}) >, \tag{4}$$

where the level $l\,(1 \le l \le L)$ of the tree $\mathcal{T}^l := \{t \in \mathcal{T}\}_{level(t)=l}$ denotes the set of all nodes having a distance of exactly l to the root, and we use $\mathcal{L}(\mathcal{T})^l$ (or $\mathcal{N}(\mathcal{T})^l$) to represent the set of leaf nodes (or interior nodes) in the level l. For a balanced binary tree, the leaf nodes may only appear in the highest or second highest levels, the number of leaf factor matrices to be estimated is d and the number of interior transfer matrices is $d-1$. Similar to the argument of the Tucker based model [7], the number of free parameters in $\mathcal{H}$-Tucker model can be obtained as $\sum_{t\in\mathcal{L}(\mathcal{T})} p_t r_t + \sum_{t\in\mathcal{N}(\mathcal{T})} r_{t_r} r_{t_l} r_t - \sum_{t\in\mathcal{T}\setminus root} r_t^2$, where the last term is used for the propose of non-singular transformation indeterminacy.

4. PARAMETER ESTIMATION

For the parameter estimation of our regression, we present in this section a maximum likelihood (ML) estimation. It is worth noticing that in (4), the linear systematic part is only linear in each $\mathbf{U}_t$ and each $\mathbf{B}_t$ separately. Hence, we can then alternately update one factor (or transfer) matrix $\mathbf{U}_t$ (or $\mathbf{B}_t$) at a time while keeping the rest of matrices fixed. This is referred by the block relaxation algorithm (BRA), and it enables us to break the simultaneous estimation of all parameters into a sequence of low dimensional parameter optimizations using classical GLM. Applying this strategy to our model, the estimation of factor matrices proceeds iteratively by sweeping all the nodes of $\mathcal{T}$ from bottom to top in a sequential way. By exploiting the mixed-product property of Kronecker product, the inner product part in (4) is equivalent to

$$< vec(\mathcal{B}), vec(\mathcal{X}) > = < (\bigotimes_{t\in\mathcal{L}(\mathcal{T})} \mathbf{U}_t)(\bigotimes_{t\in\mathcal{L}(\mathcal{T})^{L-1}} \mathbf{I}_t \otimes \bigotimes_{t\in\mathcal{N}(\mathcal{T})^{L-1}} \mathbf{B}_t)\cdots(\bigotimes_{t\in\mathcal{T}^l} \mathbf{B}_t)\cdots(\mathbf{B}_{root}),\ vec(\mathcal{X}) >, \tag{5}$$

where $\{\mathbf{I}_t \in \mathbb{R}^{r_t\times r_t}\}_{t\in\mathcal{L}(\mathcal{T})^{L-1}}$ is the identity matrix. We 'push' all the singleton factor matrices $\mathbf{U}_t$ to the highest level.

In order to perform the classical GLM, one needs to isolate the component of interest from the rest part. Particularly, for the leaf factor matrix $\mathbf{U}_t$, the inner product in (5) can be

rewritten as follows

$$< \mathbf{U}_t \mathcal{L}_L^{(t)} (\bigotimes_{t' \in \mathcal{L}(\mathcal{T}) \setminus t} \mathbf{U}_{t'})^T,\ \mathcal{X}^{(t)} > \\ =< \mathbf{U}_t,\ \mathcal{X}^{(t)} (\bigotimes_{t' \in \mathcal{L}(\mathcal{T}) \setminus t} \mathbf{U}_{t'}) (\mathcal{L}_L^{(t)})^T >, \quad (6)$$

where $\mathcal{L}_L := (\otimes_{t' \in \mathcal{L}(\mathcal{T})^{L-1}} \mathbf{I}_{t'} \otimes \otimes_{t' \in \mathcal{N}(\mathcal{T})^{L-1}} \mathbf{B}_{t'}) \cdots (\otimes_{t' \in \mathcal{T}^l} \mathbf{B}_{t'}) \cdots (\mathbf{B}_{root})$. As for the interior node $t \in \mathcal{T}^l$ in intermediate level, one can individually estimate $\mathbf{B}_t$ with only $r_{t_l} r_{t_r} r_t$ number of parameters by rewriting (5) as follows:

$$< \mathbf{B}_t \mathcal{L}_l^{(t)} (\bigotimes_{t' \in \mathcal{T}^l \setminus t} \mathbf{B}_{t'})^T,\ \mathcal{H}_l^{(t)} > \\ =< \mathbf{B}_t,\ \mathcal{H}_l^{(t)} (\bigotimes_{t' \in \mathcal{T}^l \setminus t} \mathbf{B}_{t'}) (\mathcal{L}_l^{(t)})^T > \quad (7)$$

here $\mathcal{H}_l := (\otimes_{t' \in \mathcal{T}^{l+1}} \mathbf{B}_{t'})^T \cdots (\otimes_{t' \in \mathcal{L}(\mathcal{T})} \mathbf{U}_{t'})^T vec(\mathcal{X})$ and $\mathcal{L}_l := (\otimes_{t' \in \mathcal{T}^{l-1}} \mathbf{B}_{t'})(\otimes_{t' \in \mathcal{T}^{l-2}} \mathbf{B}_{t'}) \cdots (\mathbf{B}_{root})$. The isolation of $\mathbf{U}_t$ by (6) (or $\mathbf{B}_t$ by (7)) is in fact the sequential downsizing of the original data into a smaller and more manageable size.

We iteratively run this block updating procedure from bottom to top and from left to right along each level of $\mathcal{T}$ until the log likelihood defined for classical GLM ceases to increase. The outline of the procedure is shown in Algorithm 1. Specifically, we first update the regular vector coefficient α using the previous factor and transfer matrices in line 3. Next, we estimate in lines 4-6 each factor matrix $\mathbf{U}_t$ from left to right in the leaf level by fixing the transfer matrices and other factor matrices of the tree $\mathcal{T}$. Subsequently, we update the transfer matrix $\mathbf{B}_t$ of each interior node again by keeping the factor matrices and other transfer matrices in the tree fixed (lines 8-10).

To summarize, it has been proven that BRA monotonically increases the likelihood, and the estimation is guaranteed to converge whenever likelihood function is bounded [16, 17]. Following the similar arguments in [6], Algorithm 1 reaches a stationary point where the likelihood stops increasing. However, in practice we can always expect Algorithm 1 to converge to at least a local maximum. Therefore, one should run this initialization multiple times in order to increase the chance of finding the global optimal estimate.

In tensor regression, regularization is essential when the number of parameters far exceeds the sample size, otherwise, the regression problem becomes ill-posed and the solution tends to be overfitting. Although our model is compact, we can further improve the learning performance by imposing sparsity penalty ($l1$-norm) on the transfer matrix $\{\mathbf{B}_t\}_{t \in \mathcal{N}(\mathcal{T})}$. Specifically, we instead maximize the penalized likelihood function ℓ when estimating the transfer matrix $\mathbf{B}_t$

$$\arg\max_{\mathbf{B}_t} \ell(\alpha^{[m+1])}, \{\mathbf{U}_{t'}\}_{t' \in \mathcal{L}(\mathcal{T})}^{[m+1]}, \{\mathbf{B}_{t'}\}_{t' < t, t' \in \mathcal{N}(\mathcal{T})}^{[m+1]}, \\ \mathbf{B}_t^{[m]}, \{\mathbf{B}_{t'}\}_{t' > t, t' \in \mathcal{N}(\mathcal{T})}^{[m]}) + \lambda \sum_{r_{t_l}, r_{t_r}, r_t} |b_{r_{t_l}, r_{t_r}, r_t}|, \quad (8)$$

Algorithm 1 BRA for the $\mathcal{H}$-Tucker tensor regression

1: **Initialize:** $\alpha^{[0]}$, $\{\mathbf{U}_t^{[0]}\}_{t \in \mathcal{L}(\mathcal{T})}$, $\{\mathbf{B}_t^{[0]}\}_{t \in \mathcal{N}(\mathcal{T})}$
2: **repeat**
3: $\quad \alpha^{[m+1]} = \arg\max_\alpha \ell(\alpha^{[m]}, \{\mathbf{U}_t\}_{t \in \mathcal{L}(\mathcal{T})}^{[m]}, \{\mathbf{B}_t\}_{t \in \mathcal{N}(\mathcal{T})}^{[m]})$
4: $\quad$ **for** each $t \in \mathcal{L}(\mathcal{T})$ **do**
5: $\quad\quad \mathbf{U}_t^{[m+1]} = \arg\max_{\mathbf{U}_t} \ell(\alpha^{[m+1]}, \{\mathbf{U}_{t'}\}_{t' < t, t' \in \mathcal{L}(\mathcal{T})}^{[m+1]}, \mathbf{U}_t^{[m]}, \{\mathbf{U}_{t'}\}_{t' > t, t' \in \mathcal{L}(\mathcal{T})}^{[m]}, \{\mathbf{B}_{t'}\}_{t' \in \mathcal{N}(\mathcal{T})}^{[m]})$
6: $\quad$ **end for**
7: $\quad$ **for** $l = L-1, ..., 1$ **do**
8: $\quad\quad$ **for** each $t \in \mathcal{N}(\mathcal{T})^l$ **do**
9: $\quad\quad\quad \mathbf{B}_t^{[m+1]} = \arg\max_{\mathbf{B}_t} \ell(\alpha^{[m+1]}, \{\mathbf{U}_{t'}\}_{t' \in \mathcal{L}(\mathcal{T})}^{[m+1]}, \{\mathbf{B}_{t'}\}_{t' < t, t' \in \mathcal{N}(\mathcal{T})}^{[m+1]}, \mathbf{B}_t^{[m]}, \{\mathbf{B}_{t'}\}_{t' > t, t' \in \mathcal{N}(\mathcal{T})}^{[m]})$
10: $\quad\quad$ **end for**
11: $\quad$ **end for**
12: **until** $\ell^{[m]} - \ell^{[m+1]} < \varepsilon$

where $b_{r_{t_l}, r_{t_r}, r_t}$ corresponds to each entry of $\mathbf{B}_t$, λ is the penalty tuning parameter. This form of $l1$-norm regularization causes many coefficients in $\mathbf{B}_t$ to be sufficiently small.

5. EXPERIMENTAL EVALUATION

5.1. Simulations on Synthetic Data

In this simulation, we were interested in how the sample size influences the estimation accuracy of the proposed model. In this respect, we first generate the true signal tensor $\mathcal{B}$ using Tucker format, where the core tensor $\mathcal{G}$ and factor matrices $\mathbf{U}$ are randomly drawn from standard normal distributions. Here, we test two setups of the dimension of $\mathcal{B}$, which is fixed at 16 for the $4th$ order tensor and 9 for the $5th$, respectively. The dimension of $\mathcal{G}$ is set as 4 for the $4th$ order tensor and 3 for the $5th$. We set the true vector signal α as $\mathbf{1}_5$ for both cases. Subsequently, the regular covariate vector $\mathbf{z} \in \mathbb{R}^5$ and tensor covariate $\mathcal{X} \in \mathbb{R}^{16 \times 16 \times 16 \times 16}$ (or $\mathcal{X} \in \mathbb{R}^{9 \times 9 \times 9 \times 9 \times 9}$ for the $5th$ case) are all generated from independent standard normals. Finally, the output samples are produced by $y = \alpha^T \mathbf{z} + < \mathcal{B}, \mathcal{X} > + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 1)$ is an additive noise. For comparison, we investigate the deviation between the learned $\hat{\mathcal{B}}$ and the true $\mathcal{B}$ for an increasing number of samples n in terms of root mean square error (RMSE). Note that the ht-ranks at leaf singletons are chosen to be the same as d-ranks of the corresponding modes. Moreover, the total number of free parameters of $\mathcal{H}$-Tucker model is less than or equal to Tucker's. E.g., in accord with d-ranks (4,4,4,4), ht-ranks can be (1-2,2-4,4,4,4) or (1-12,12-4,4,4,4) etc. We repeat the run 50 times. As shown in Fig.2, it is straightforward to see the RMSE of $\mathcal{H}$-Tucker model of all settings progressively decreases as the sample size increases from 200 to 1800, implying the stability of the proposed method. When the sample size is relatively small, the $\mathcal{H}$-Tucker model of lower ht-rank

Dimensions	$\mathcal{H}$-Tucker Regression				Tucker Regression			
	ht-Rank	Free Param	Misclassification Error		d-Rank	Free Param	Misclassification Error	
			Reg	Non-Reg			Reg	Non-Reg
$17 \times 20 \times 16$	(1-4,3-2,2)	117	**0.296**	**0.302**	(2,2,3)	117	0.308	0.331
	(1-3,3-3,3)	159	**0.302**	**0.308**	(3,3,3)	159	0.308	0.320
	(1-3,4-3,3)	170	**0.296**	**0.302**	(3,3,4)	177	0.302	0.320
$12 \times 14 \times 12$	(1-4,3-2,2)	83	**0.284**	**0.308**	(2,2,3)	83	0.290	0.314
	(1-4,3-3,3)	114	**0.278**	**0.290**	(3,3,3)	114	0.296	0.314
	(1-3,4-3,3)	122	**0.284**	**0.302**	(3,3,4)	128	0.290	0.314

Table 1: Performance comparison for the misclassification error of $\mathcal{H}$-Tucker and Tucker regression model.

obviously outperforms the Tucker model. For example, the Tucker model of rank (3,3,3,3,3) results in 333 free parameters. On the contrary, $\mathcal{H}$-Tucker model of ht-rank (1-2,2-2,3,3,3-3,3) has 130 free parameters. With only 800 training samples, the RMSE of learned $\mathcal{H}$ Tucker model is 10.3, which is evidently superior to that 21.9 of Tucker. This is the case especially for higher order data, since the parameter size of core tensor expands exponentially in d. This result indicates the $\mathcal{H}$-Tucker is more compact than Tucker, and scales better to higher order data. On the other hand, for larger n, we can boost the learning performance by increasing the ht-rank adjusted to the sample size. If n is big enough, we can perfectly recover the true signal $\mathcal{B}$ with the same number of free parameters as Tucker's. In this case, the ht-rank we select is (1-9,9-9,3,3,3-3,3), resulting in the exact 333 free parameters. Though the true signal is generated from the Tucker structure, we observe the result obtained by our method is better than Tucker model. When n is greater than 1400 for $\mathcal{H}$-Tucker (or 1600 for Tucker), the RMSE is almost zero. Compared with Tucker model, the $\mathcal{H}$-Tucker is more flexible to efficiently represent the core tensor $\mathcal{G}$ by allowing distinct ranks in the intermediate levels of the tree structure. It can thereby achieve better estimation accuracy while requiring a small number of free parameters given limited number of samples.

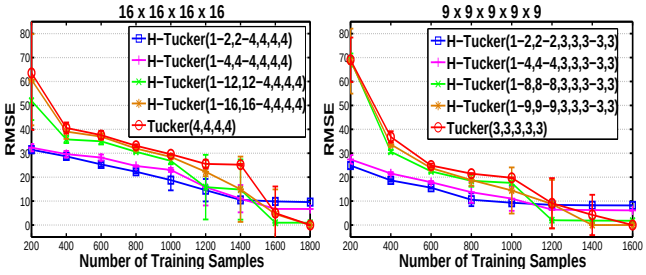

Fig. 2: Performance comparison vs. number of samples.

5.2. ADHD Brain Imaging Dataset

We also validated our model on the real-life attention deficit hyperactivity disorder (ADHD) brain imaging data. The ADHD data is freely available from the Neuro Bureau and is preprocessed according to the standard Burner pipeline, resulting in T1 images of size $121 \times 145 \times 121$ for each subject [18]. For our experiment, we obtain 761 training subjects and 169 testing subjects after removing those with missing observations. We treat each MRI image represented by a $3rd$ order tensor $\mathcal{X}$ as the image predictor, and the corresponding response y is the binary diagnosis outcome. Additionally, the regular vector predictor $\mathbf{z}$, namely individual's gender, age and handiness, is also included. Notice that the whole dataset remains non-normalized in this experiment. We compare our model with the classical Tucker regression model which is regarded as the state of the art and has been proved to be superior to the CP based modeling in [7]. Toward this purpose, we downsize the original image into different sizes using the Daubechies D4 wavelet transformation. The reduced dimensions we consider are $12 \times 14 \times 12$ and $17 \times 20 \times 16$. We also experiment with distinct ranks to see how the performance can be affected by the free parameter size. Note carefully that we follow the same guideline for choosing the ranks as previous section, and the sample size is approximately 4.5-9 times as large as number of free parameters. In the case of the regularization, we tune the penalty parameter λ based on the Bayesian information criterion (BIC) [19]. Table 1 shows the results of misclassification errors on the testing set w.r.t. different setups of dimensions and ranks. As expected, the $\mathcal{H}$-Tucker model outperforms the Tucker model in all cases with smaller or equal number of free parameters. We also observe that the non-regularized $\mathcal{H}$-Tucker model is already better or comparable to the regularized version of Tucker model. It is worth noticing that the ADHD-200 is a really hard task, the error rate around 0.28 achieved by the proposed method is fairly attractive especially when the original data is unbalanced.

6. CONCLUSION

In this paper we have introduced a novel generalized linear tensor regression model based on the $\mathcal{H}$-Tucker decomposition. The experiment results show that $\mathcal{H}$-Tucker is a highly compact, flexible and scalable model, which has better low-rank approximation using only a small number of parameters. This key advantage makes it the perfect model for the applications, such as MRI, fMRI analysis, where data is extremely high dimensional but with quite restricted sample size.

7. REFERENCES

[1] Q. Zhao, G. Zhang, and A. Cichocki, "Tensor-variate gaussian processes regression and its application to video surveillance.," *In Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference*, pp. 1265–1269, 2014.

[2] W. Guo, I. Kotsia, and I. Patras, "Tensor learning for regression," *IEEE Transactions on 21(2)*, pp. 816–827, 2012.

[3] Q. Zhao, C.F. Caiafa, D.P. Mandic, Z.C. Nagasaka, Y. Fujii, and A. Cichocki, "Higher order partial least squares (hopls): A generalized multilinear regression method," *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 35(7), pp. 1660–1673, 2013.

[4] Q. Zhao, G. Adali, T. Zhang, and A. Cichocki, "Kernelization of tensor-based models for multiway data analysis: processing of multidimensional structured data," *Signal Processing Magazine, IEEE Transactions*, vol. 30(4), pp. 137–148, 2013.

[5] A. Eliseyev and T. Aksenova, "Recursive n-way partial least squares for brain-computer interface," *PloS one*, 2013.

[6] H. Zhou, L. Li, and H. Zhu, "Tensor regression with applications in neuroimaging data analysis," *Journal of the American Statistical Association*, vol. 108(502), pp. 540–552, 2013.

[7] X. Li, H. Zhou, and L. Li, "Tucker tensor regression and neuroimaging analysis," *arXiv preprint arXiv:1304.5637*, 2013.

[8] J.D. Carroll and J.J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition," *Psychometrika arXiv:1203.3209*, pp. 283–319, 1970.

[9] R.A. Harshman, "Foundations of the parafac procedure: Models and conditions for an explanatory multi-modal factor analysis," *UCLA Working Papers in Phonetics*, pp. 1–84, 1970.

[10] L.R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," *Problems in Measuring Change, University of Wisconsin Press*, pp. 122–137, 1963.

[11] T. Oommen, D. Misra, N.K.C. Twarakavi, A. Prakash, B. Sahoo, and S. Bandopadhyay, "An objective analysis of support vector machine based classification for remote sensing," *Mathematical Geosciences*, 2008.

[12] W. Hackbusch, S. Kuhn, and L. Li, "A new scheme for the tensor representation," *J. Fourier Anal. Appl.*, pp. 706–722, 2009.

[13] L. Grasedyck, "Hierarchical singular value decomposition of tensors," *SIAM J. Matrix Anal. Appl.*, p. 20292054, 2010.

[14] L. Grasedyck, "Hierarchical low rank approximation of tensors and multivariate functions," *Lecture notes of Zurich summer school on Sparse Tensor Discretizations of High Dimensional Problems*, 2010.

[15] P. McCullagh and J.A. Nelder, "Generalized linear models," *Monographs on Statistics and Applied Probability*, 1983.

[16] J. de Leeuw, "In information systems and data analysis," *Monographs on Statistics and Applied Probability*, pp. 308–325, 1994.

[17] K. Lange, "Numerical analysis for statisticians," *Statistics and Computing*, 2010.

[18] ADHD (2014), "The burner data of adhd-200 consortium," *http://neurobureau.projects.nitrc.org/ADHD200*.

[19] G. E. Schwarz, "Estimating the dimension of a model," *Annals of Statistics 6 (2)*, pp. 461–464, 1978.