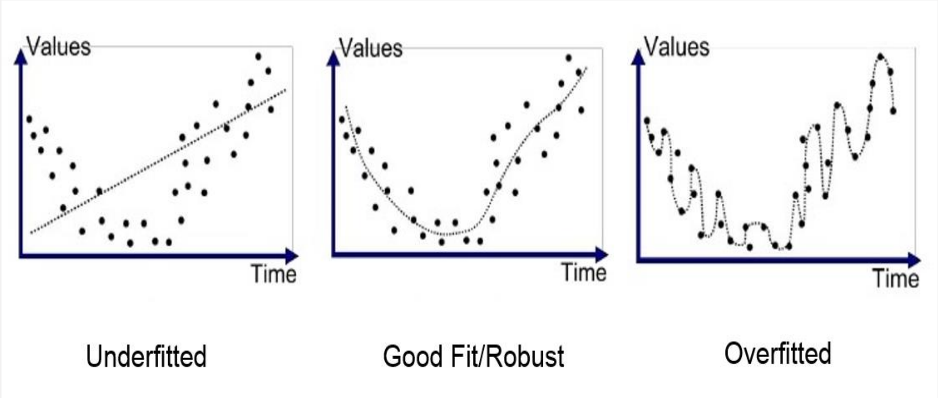
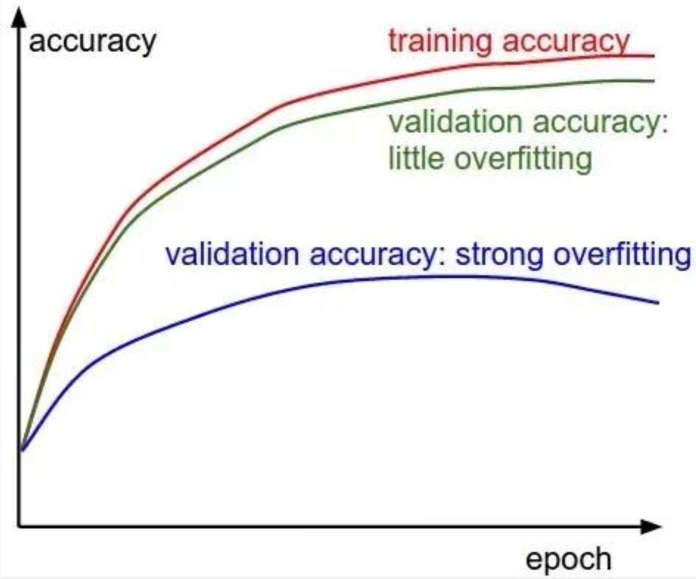
TENSOR DECOMPOSITION METHOD COMPARISON

By: Leonel Ramirez, Alan Urteaga, Miguel Gutierrez, Het Patel



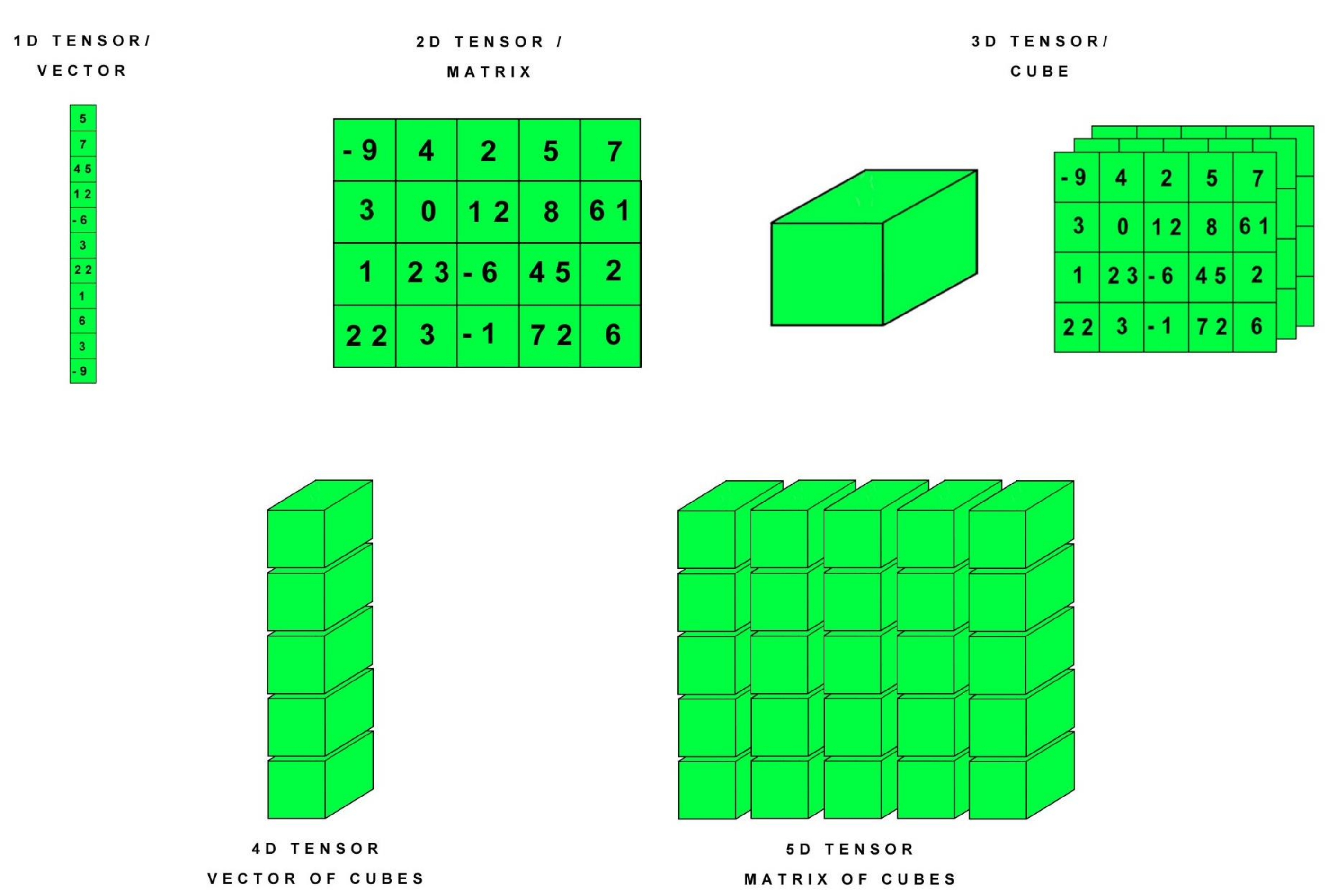


Problems with CNNs

- CNNs are suited for learning these intricate & complex relationships. However, this comes with added risk of overfitting and lack robustness.
- Model generalizability is needed.
- DNNs leverage the spatial structure of input data via convolutions, pooling, point-wise non-linearities (like ReLU), etc. However, this structure typically wasted by a flattening layer followed by one or several fully connected layers.
- Tensor dropout provides a hand in this.

What is a Tensor?

- **Tensor** - can be described as an *nth*-dimensional array
- **Mode** - is the dimension of the tensor or tensor shape.
  Ex: a tensor of mode 3 will be have a 3D shape and a mode 2 tenor will have a 2D shape
- **Rank** - is the number of underlining components that the tensor can be broken down into when computing tensor decomposition.



Tensor Dropout

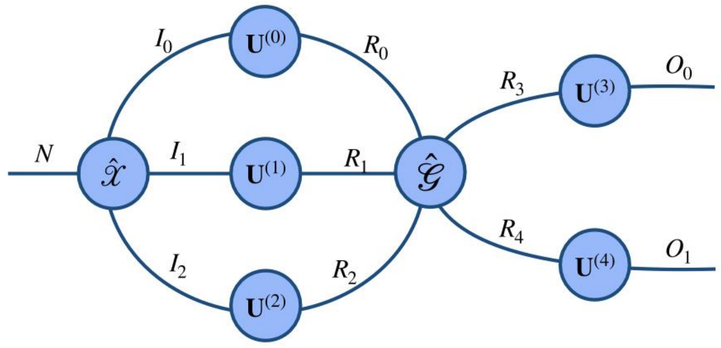
- **Stochastic**
- **Decomposed Forms**
- **Applied to Tucker and CP**

Theorem 1: Tensor Dropout on Tucker decomposition and deterministic regularized loss (proof in Appendix[1]).

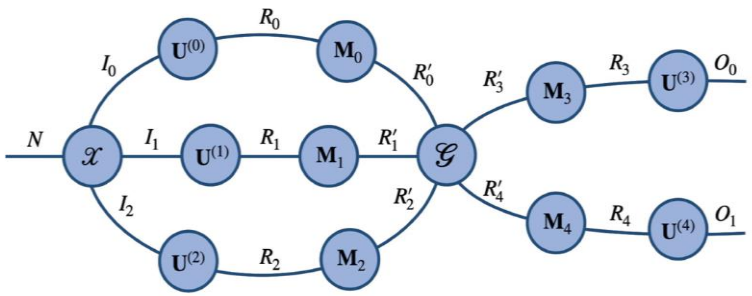
$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\lambda}}\left[\frac{\mathbf{1}}{\boldsymbol{S-1}}\sum_{\boldsymbol{k=0}}^{\boldsymbol{S-1}}\left(\boldsymbol{y}^{(\boldsymbol{k})}-\langle\tilde{\boldsymbol{\mathcal{W}}},\boldsymbol{\mathcal{X}}^{(\boldsymbol{k})}\rangle\right)^{\mathbf{2}}\right]\\
&=\frac{1}{S-1}\sum_{k=0}^{S-1}\left(\boldsymbol{y}^{(k)}-\theta^{N}\langle\mathcal{W},\mathcal{X}^{(k)}\rangle\right)^{2}\\
&\quad+\frac{\theta^{N}(1-\theta^{N})}{S-1}\sum_{k=0}^{S-1}\langle\mathcal{G}^{\star 2}\times_{0}(\mathbf{U}^{(0)})^{\star 2}\cdots\times_{N}(\mathbf{U}^{(N)})^{\star 2},\\
&\quad(\mathcal{X}^{(k)})^{\star 2}\rangle. \qquad (7)
\end{aligned}$$

Theorem 2: Tensor Dropout on CP decomposition is equivalent to a deterministic regularized loss. (Proof in Appendix[1])

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\lambda}}\left[\frac{1}{S-1}\sum_{k=0}^{S-1}\left(\boldsymbol{y}^{(k)}-\langle[\![\boldsymbol{\lambda};\mathbf{U}^{(0)},\ldots,\mathbf{U}^{(N)}]\!],\mathcal{X}^{(k)}\rangle\right)^{2}\right]\\
&=\frac{1}{S-1}\sum_{k=0}^{S-1}\left(\boldsymbol{y}^{(k)}-\theta\langle[\![\mathbf{U}^{(0)},\ldots,\mathbf{U}^{(N)}]\!],\mathcal{X}^{(k)}\rangle\right)^{2} \qquad (11)\\
&\quad+\frac{\theta(1-\theta)}{S-1}\sum_{k=0}^{S-1}\langle[\![(\mathbf{U}^{(0)})^{\star 2},\ldots,(\mathbf{U}^{(N)})^{\star 2}]\!],(\mathcal{X}^{(k)})^{\star 2}\rangle.
\end{aligned}$$



(a) Tensor diagram of a TRL



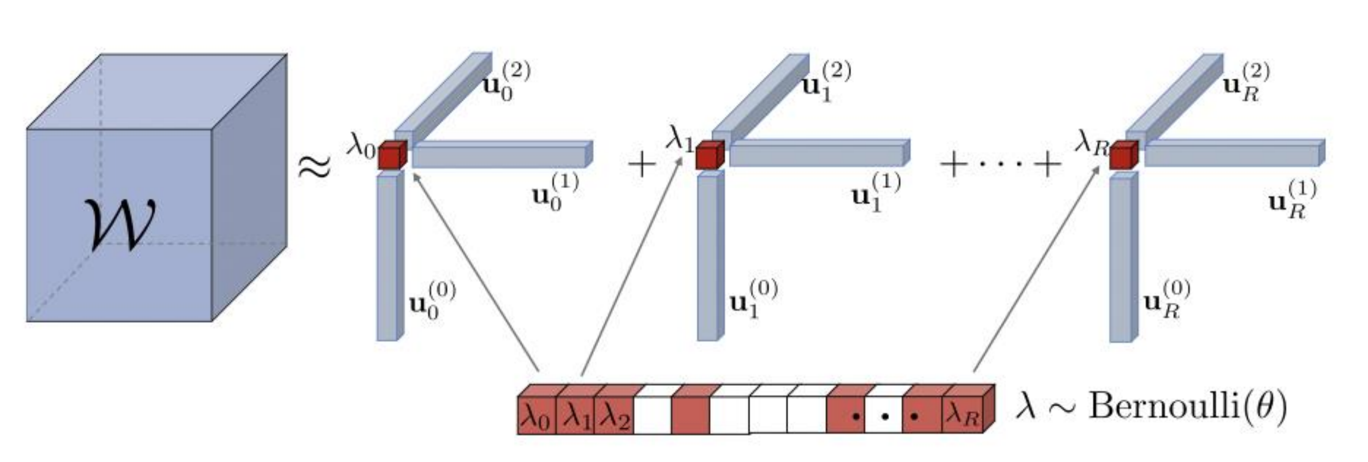
(b) Tensor diagram of a R-TRL

Tensor Proposal

- **Tensor Regression Networks propose to replace fully connected layers entirely with a tensor regression layer (TRL)**
- **This preserves the structure of the multidimensional data (input) by expressing an output tensor as the result of a tensor contraction between the input tensor and some low rank regression weight tensor**
- **TRL estimate the regression weight tensor $\mathcal{W} = \mathbb{R}^{I_0 x I_1 x \dots x I_N}$, expressed under some low rank tensor decomposition**

Canonical-Polyadic (CP) Dropout Decomposition

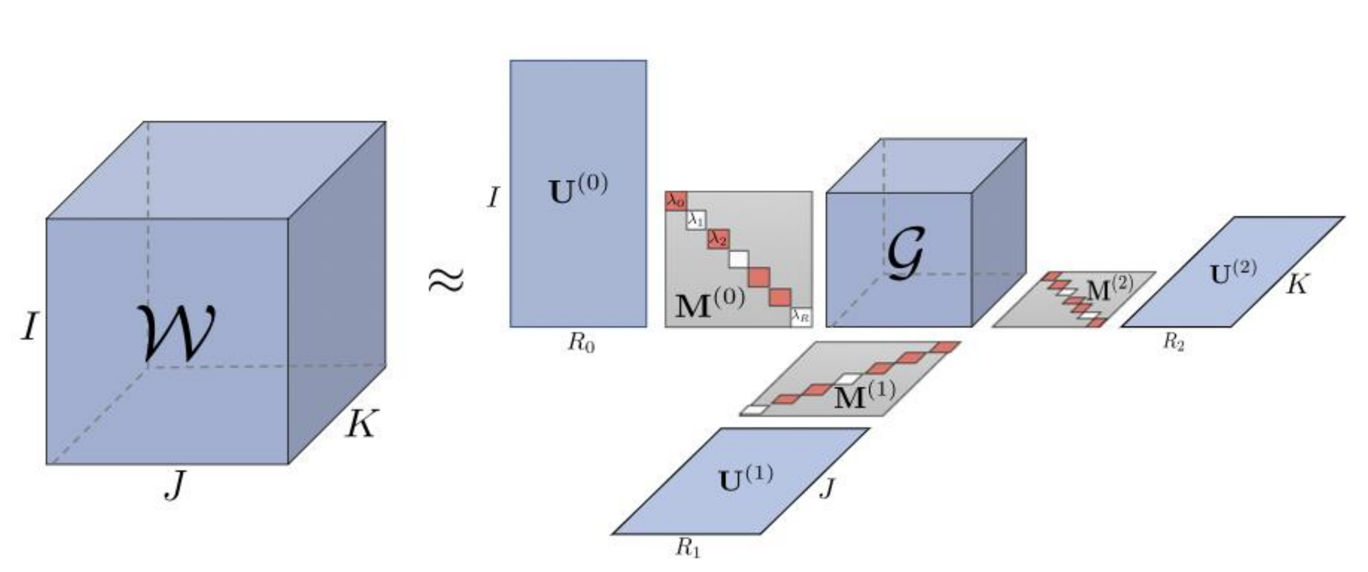
- Approximates the input tensor(weight) by summing the different vectors generated when decomposing the tensor (A, B, C)
- For the Drop out version of CP each component has a Bernoulli variable that determines if the component is dropped or not when decomposing the tensor. (“Acts like inducing stochasticity on the rank of decomposition.”)[2]
- Vector sizes are of one of the mode of *X* by rank. Ex: *X* = R#^I x J x K is decomposed into A, B, C respectively. A.size = I x r



$$\mathcal{X} = \sum_{k=0}^{R-1} \underbrace{\lambda_k \boldsymbol{u}_k^{(0)} \circ \boldsymbol{u}_k^{(1)} \circ \cdots \circ \boldsymbol{u}_k^{(N)}}_{\text{rank-1 components}},$$

Tucker Dropout Decomposition

- This method involves decomposing a tensor into n-modes, depending on its dimension.
- The result is a breakdown of a tensor into a low rank core G (the dimensions are determined by the tensors rank), which projects the core along each mode with diagonal matrices (which are subject to *Bernoulli* tensor dropout) and set of factor matrices multiplied along the specified mode.



$$\mathcal{X} = \mathcal{G} \times_0 \mathbf{U}^{(0)} \times_1 \mathbf{U}^{(1)} \times \cdots \times_N \mathbf{U}^{(N)}$$

$$\mathbf{M}^{(k)} = \mathrm{diag}(\boldsymbol{\lambda}^{(k)})$$

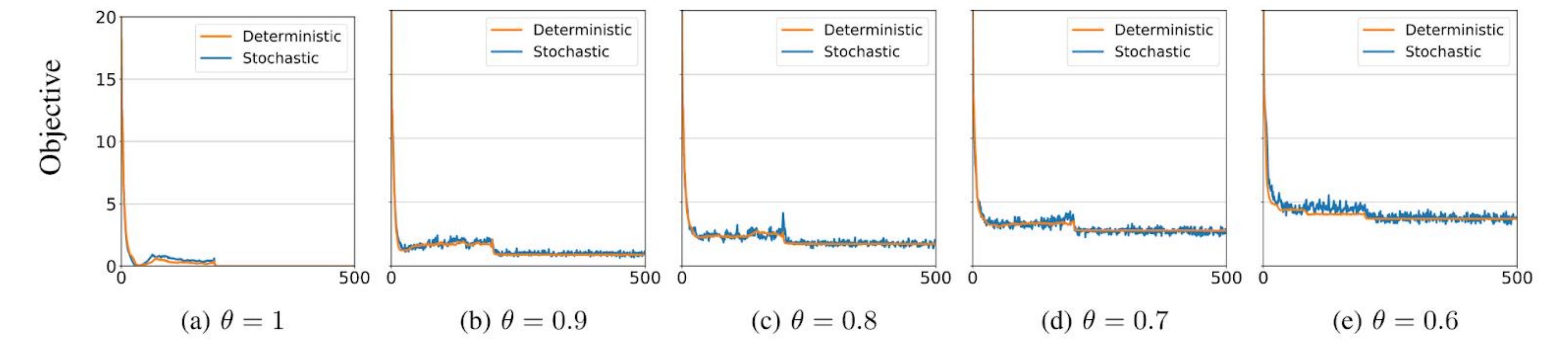


Fig. 4. **Experiment on synthetic data:** loss of the CP R-TRL as a function of the number of epochs for the stochastic version (orange) and the deterministic one based on the regularized objective function (blue). As expected, both formulations are empirically the same.

1. Synthetic Experiments

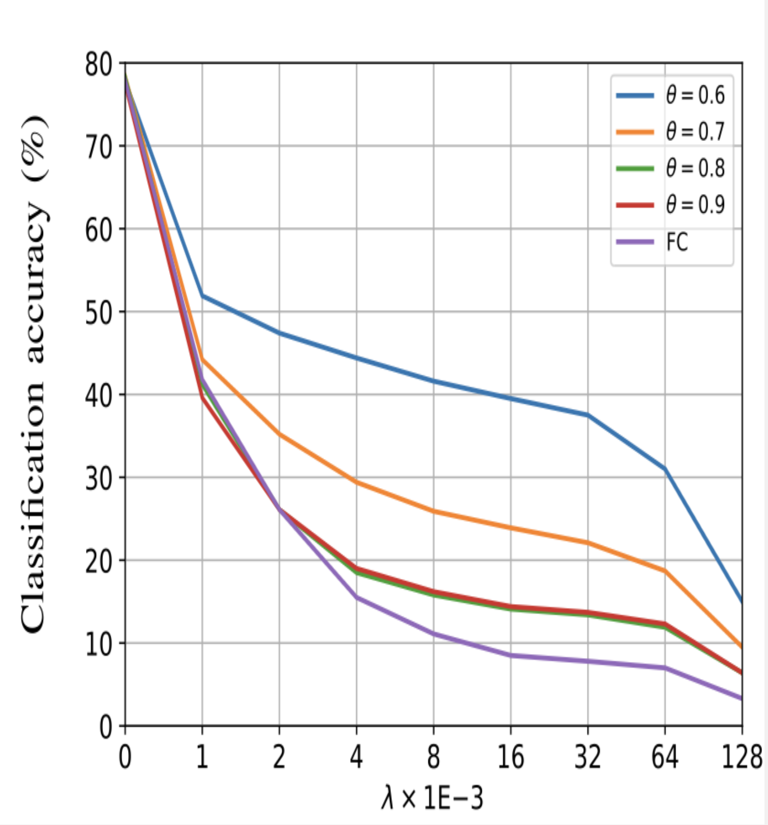
- 3rd Order Random Regression Weight Tensors
- Rank 10

2. ImageNet

- **Robustness: In addition to reducing overfitting, it is demonstrated that the proposed tensor dropout makes the model more robust to perturbations in input, for both random noise & adversarial attacks.**
- **The R-TRL architecture with tensor dropout is much more robust to adversarial attacks, depending on the tensor dropout rate (θ).**

TABLE II
REAL-VALUED NETWORK PERFORMANCE ON IMAGENET FOR FGSM, BIM AND PGD ATTACKS WITH $\lambda \in \{2, 8, 16\}$. WE REPORT CLASSIFICATION ACCURACY IN ALL CASES

Attack Type	λ	Method: Baseline	Ours: $\theta = 0.8$	Ours: $\theta = 0.7$	Ours: $\theta = 0.6$
Clean *(no attack)*		77.1	77.7	77.4	**78.0**
FGSM	**2**	26.1	26.0	35.3	**47.4**
	8	11.1	15.4	26.0	**41.6**
	16	8.5	14.1	24.0	**39.5**
BIM	**2**	26.1	26.0	35.3	**47.3**
	8	1.0	4.0	9.9	**26.1**
	16	0.1	1.0	2.8	**13.2**
PGD	**2**	26.0	26.0	35.6	**47.1**
	8	0.8	4.5	11.3	**27.9**
	16	0	1.4	3.8	**13.5**

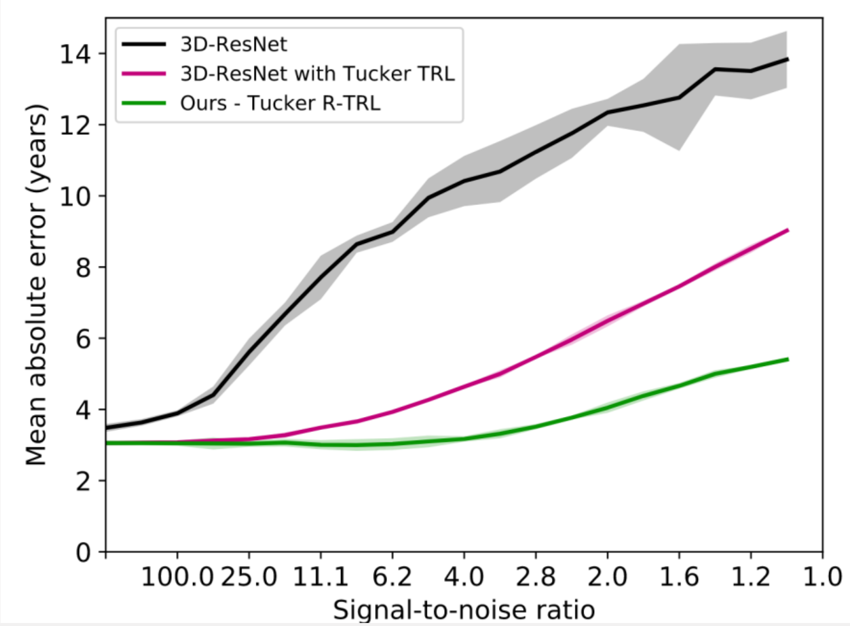


3. Phenotypic Trait MRI DATA

- **Brain Age Difference**
- **Robustness Study: resistance to added gaussian white noise**

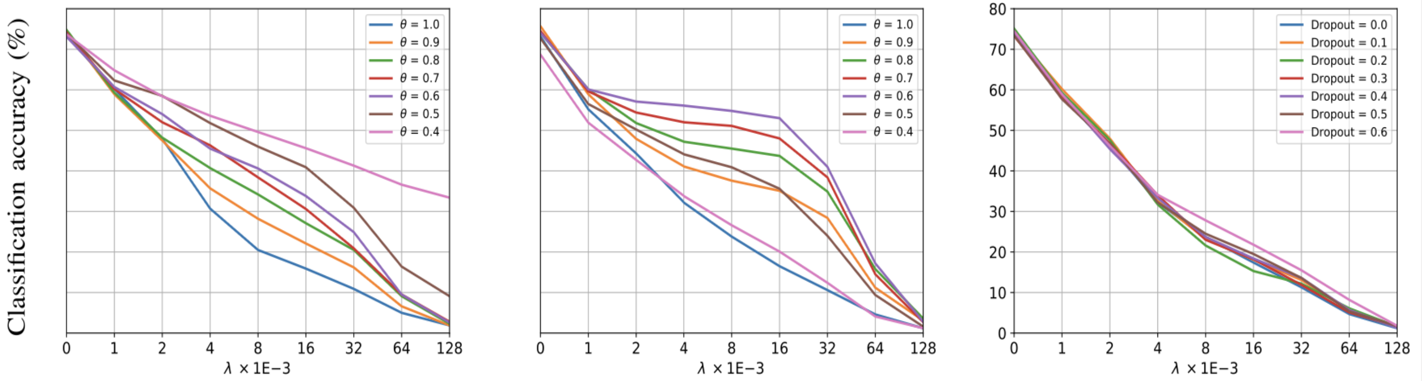
TABLE III
CLASSIFICATION ACCURACY FOR U.K. BIOBANK MRI. THE RESNET MODELS WITH R-TRL SIGNIFICANTLY OUTPERFORMS THE VERSION WITH A FULLY-CONNECTED (FC) LAYER

Architecture	Regression	MAE
3D-ResNet	FC	2.96 years
3D-ResNet	Tucker	2.70 years
Ours	Randomized Tucker	**2.65 years**
Ours	Randomized CP	**2.58 years**



4. CIFAR-100

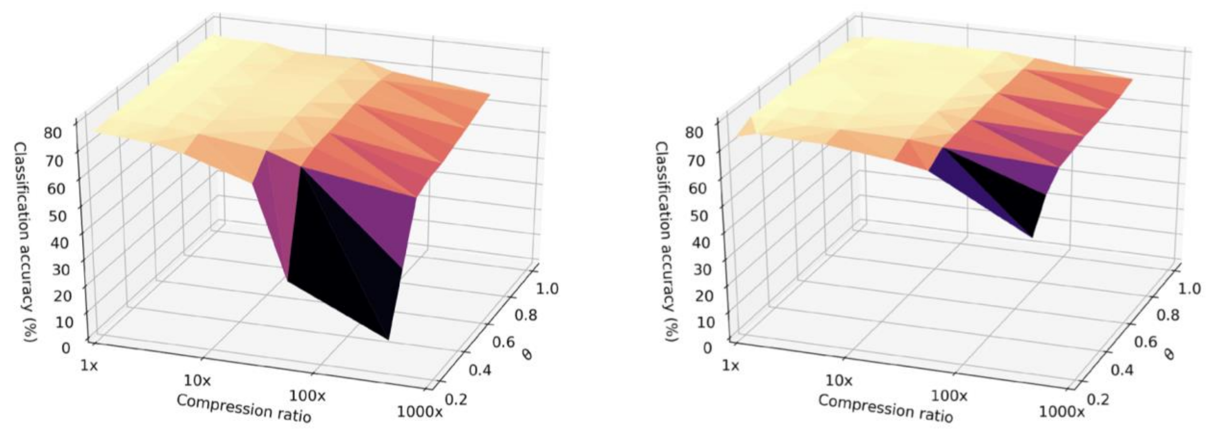
- Comparison of traditional tensor regression, and regular dropout, and asses the robustness of each method in the face of adversarial noise
- Tucker TRL + Tensor Dropout & CP TRL + Tensor Dropout prove to be more robust than Tucker TRL + Regular Dropout
- Results demonstrate that the performance is not overly sensitive to the choice of rank & tensor dropout probability



(a) **Tucker TRL + Tensor Dropout**. FGS attack on a Tucker TRL with Bernoulli tensor dropout and different drop rates.

(b) **CP TRL + Tensor Dropout**. FGS attack on CP TRL with Bernoulli tensor dropout and different drop rates.

(c) **Tucker TRL + Regular Dropout**. FGS attack on Tucker TRL where *regular* dropout is applied to the regression weights, with different dropout rates.



(a) Tucker TRL + Tensor Dropout

(b) Tucker TRL + Tensor Dropout with replacement

What is NORMO?

- Is an algorithm that approximates the number of components in a tensor decomposition
- Main purpose to reduce the number of components/rank (by determining the redundancy of certain components) to reduce computational cost
- Tested on artificial/real world tensor datasets with known number of components/rank using CP decomposition
- Preformed the best on both high rank and low rank for both synthetic and real world datasets when compared to the different algorithms

Table 4
Estimation results in the real-world datasets with known model order (correct estimates in bold).

	True model Order	DIF FIT	CORC ONDIA	ARD ridge	ARD sparse	Convex Hull	N-D MDL	NOR MO
amino	4	3	**4**	⋄	⋄	[3 3 3]	11	**4**
dorrit	4	⋆	**4**	[13 6 13]	[10 5 12]	–	–	**4**
wbnmr	4	⋆	3	⋄	⋄	[24 25 25]	–	**4**
sugar	4	6	**4**	⋄	⋄	[1 4 4]	550	5
tongue	3	**3**	**3**	⋄	⋄	[2 2 1]	6	1

Table 5
Estimation results in the real-world datasets with unknown model order.

	DIF FIT	CORC ONDIA	ARD ridge	ARD sparse	Convex Hull	N-D MDL	NOR MO
enron	⋆	12	⋄	⋄	[2 2 2]	183	11
dblp	⋆	\	⋄	⋄	[25 25 7]	2722	†
challenge	⋆	7	⋄	⋄	\	123	5
friends	⋆	†	[25 25 17]	[25 25 12]	[25 25 16]	128	†
reality	⋆	11	⋄	⋄	[2,2,2]	92	11

Table from “A new method for estimating the number of components in CP tensor decomposition”,[1]

Conclusions

- Paper [2] (Tensor Dropout) Tensor dropout helps constrain the learned representations to be more general, which leads to reduced overfitting & better out of sample generalization
- Spatial information is traditionally discarded during the flattening process, which is avoided using a tensor regression layer (TRL) and (R−TRL)
- Paper [1] (NORMO) proposed a different algorithm to determine the rank of a tensor decomposition along with accounting for the redundancy of some of the ranks
- Paper [3] (Tensor Methods) gives a detailed background on tensors and how they are used in computer vision and deep learning

Next Steps weeks 3 - 5

- Fortify our knowledge weak points
- Become more familiar with the implementations of R-TRL from paper [2] (Tensor Dropout)
- Search for existing implementations for R-TRL from paper [2] (Tensor Dropout)
- Try to Replicate the results from paper [2] (Tensor Dropout)

Thank you for your time!

Any Questions?

Citations:

1) Sofia Fernandes, Hadi Fanaee-T, João Gama,NORMO: A new method for estimating the number of components in CP tensor decomposition,Engineering Applications of Artificial Intelligence,Volume 96,2020,103926,ISSN 0952-1976, https://www.sciencedirect.com/science/article/pii/S0952197620302530
2) A. Kolbeinsson et al., "Tensor Dropout for Robust Learning," in IEEE Journal of Selected Topics in Signal Processing, vol. 15, no. 3, pp. 630-640, April 2021, doi: 10.1109/JSTSP.2021.3064182, https://ieeexplore.ieee.org/document/9381098
3) Y. Panagakis et al., "Tensor Methods in Computer Vision and Deep Learning," in Proceedings of the IEEE, vol. 109, no. 5, pp. 863-890, May 2021, doi: 10.1109/JPROC.2021.3074329, https://ieeexplore.ieee.org/abstract/document/9420085
4) “Tamara G. Kolda: ‘Tensor Decomposition.’” YouTube. YouTube, April 6, 2018. https://www.youtube.com/watch?v=L8uT6hgMt00
5) “Applied Linear Algebra: Tensor Decompositions.” YouTube. YouTube, December 2, 2020. https://www.youtube.com/watch?v=tm5am60CId4
6) “Applied Linear Algebra: Implementing Tensor Decompositions.” YouTube. YouTube, December 4, 2020. https://www.youtube.com/watch?v=1UbAJeix3To
7) dfleisch. “What's a Tensor?” YouTube. YouTube, November 20, 2011. https://www.youtube.com/watch?v=f5liqUk0ZTw.
8) Krishan. “Understanding Compression of Convolutional Neural Nets: Part 3.” From Data to Decisions, June 9, 2020. https://iksinc.online/2020/06/09/understanding-compression-of-convolutional-neural-nets-part-3/.
9) Krishan. “Understanding Tensors and Tensor Decompositions: Part 1.” From Data to Decisions, June 15, 2021. https://iksinc.online/2018/02/12/understanding-tensors-and-tensor-decompositions-part-1/
10) “Investigating Tensors with Pytorch.” DataCamp. Accessed June 9, 2022. https://www.datacamp.com/tutorial/investigating-tensors-pytorch