Causal Capsules and Tensor Factor Analysis

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper seeks to connect tensor factor analysis with deep neural network learning. Forward causal inference is addressed with an architecture of causal capsules that implements (i) the M-mode SVD / multilinear PCA, and (ii) a generalized architecture that that employs kernels and implements the doubly non-linear kernel-MPCA/MICA. Each architecture computes the constituent causal factor representations with a recurrent causal capsule chain. For distributed computation, we break the links and compute the causal representation separately shuttling causal information between capsules. The causal information for the object whole may be computed efficiently by parts. These causal capsule architectures can be "glommed" together to create a part-based hierarchy of causal capsules. The architecture parallels the Incremental M-mode Block SVD, a block tensor decomposition that employs a hierarchy of data columns and incrementally computes levels of abstractions bottom-up. Inverse causal inference is performed with a multilinear projection architecture that inverts the estimated forward causal model and employs a set of observations that constrain the solution set and render the problem well-posed.

1 INTRODUCTION

A key component to data understanding is modeling the mechanism of data formation in terms of causal factors and addressing the difficult problem of disentangling the causal contributions.

Forward causal inference uses data and domain knowledge to model the mechanism of data formation, and predict the effects of interventions [Pearl (2000); Imbens & Rubin (2015); Spirtes et al. (2000); Bollen & Pearl (2013); Vasilescu & Terzopoulos (2002a; 2005); Vasilescu et al. (2021)]. Inverse causal inference estimates the causes of effects given an estimated forward causal model and a set of observations that constrain the solution set [Vasilescu & Terzopoulos (2007); Vasilescu (2011; 2009); Gelman & Imbens (2013)].

There are two mathematical frameworks for causal inference: structural equations and potential-outcome. Donald Rubin and his collaborators have advocated the potential outcome approach by framing the causal inference as a missing data problem. Judea Pearl has been advocating the directed acyclic graph approach as a mathematical language that he has unified with

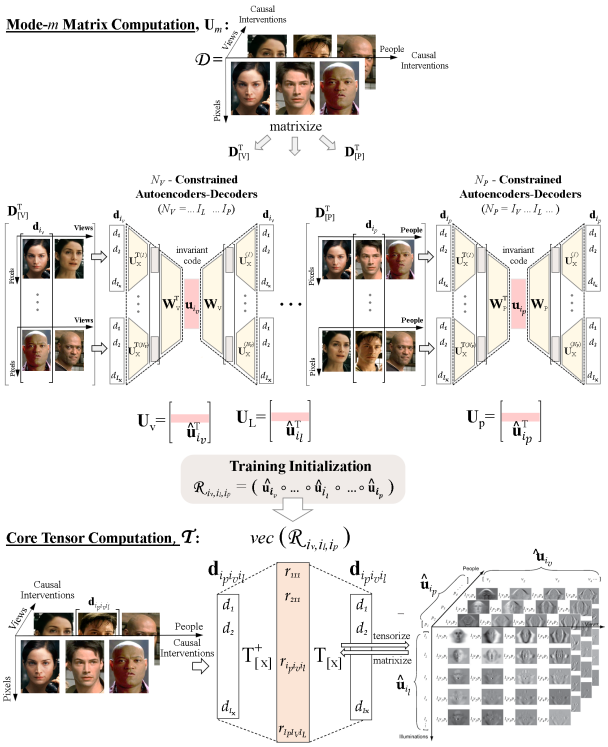

Figure 1: Neural network architecture that implements the M-mode SVD (Alg. 1), Fig. 2(a). The architecture computes each factor or mode matrix, $\mathbf{U}_{\mathrm{m}}$, with either a single autoencoder-decoder (dashed lines), or a set of constrained "cluster"-based autoencoders whose PCA basis vectors are rotated to compute invariant representations irrespective of "cluster", *i.e.*, irrespective of other causal factors. The extended core tensor, $\mathcal{T}$, governs the interaction between causal factor.

structural equations and counterfactuals. Pearl's causation ladder provides a way of thinking about causal discovery, causal reasoning, and decision making Pearl (2000).

The multilinear (tensor) algebra have been employed to develop causal models, regression models, and reduce neural network parameters, Fig. 2. The multilinear (tensor) structural equation approach has been employed for causal factor analysis in psychometrics [Tucker (1966); Harshman (1970); Carroll & Chang (1970); Bentler & Lee (1979); Kroonenberg & de Leeuw (1980)], econometrics [Kapteyn et al. (1986); Magnus & Neudecker (1988)], chemometrics [Bro (1997); Acar et al. (2014)], signal processing [de Lathauwer (1997; 2008)], computer vision [Vasilescu & Terzopoulos (2002b); Wang & Ahuja (2003)], computer graphics [Vasilescu (2002); Davis & Gao (2003); Vasilescu & Terzopoulos (2004); Vlasic et al. (2005); Hsu et al. (2005)], and machine learning [Vasilescu (2009); **?**]. Tensor structural equations have also been used to develop regressive data models [Chu & Ghahramani (2009); Tang et al. (2013); Wang et al. (2017); Kossaifi et al. (2020)]. Tensor methods have also been employed to reparameterize neural networks. Neural network weights are organized into "data tensors" and tensor methods are dimensionally reduced for computational efficiency. [Oseledets (2011); Lebedev et al. (2014); Novikov et al. (2015); Kim et al. (2015); Kossaifi et al. (2017); Khrulkov (2020)].

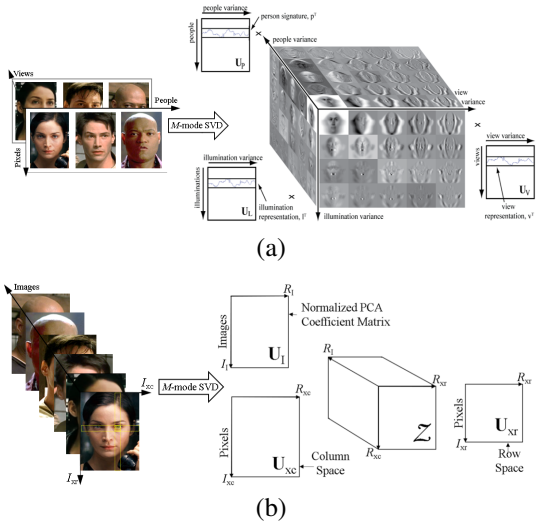

Figure 2: For objective causal inference, experimental design of data acquisition and problem setup trump data analysis algorithm. (a) Causal model derived from a collection of vectorized images rendered in Maya from the University of Freiburg 3D scans. (b) Computing the column and row space from a collection of images where each image is treated as a collection of column/row observations rather than a single vectorized observation.

This paper seeks to connect tensor causal factor analysis to deep learning methods by proposing a causal capsule hierarchy of parts and wholes that is consistent the capsule theory proposed by Geoffrey Hinton and his collaborators [Hinton (2021); Sabour et al. (2017); Hinton et al. (2011)].

This paper addresses forward causal inference with a causal capsules architecture that implements the M-mode SVD and the alternating least square M-mode SVD (multilinear principal component analysis) for subspace learning, Fig. 1 and Fig 5(e). We will show that the M-mode SVD rotates the basis vectors of various "clusters" such that the causal representation is the same regardless of data cluster, *i.e.*, invariant to all the other causal factors. These M-Mode SVD algorithms and the corresponding architectures serve as a basis for a nonlinear causal capsule that employs kernel activations and implements kernel MPCA/MICA. A causal capture architectures may be "glommed" together to create a hierarchy of part-based causal capsules that are composed of a hierarchy of columns of data [Vasilescu et al. (2021); Vasilescu & Kim (2019); de Lathauwer & Nion (2008)]. This hierarchical architecture implements the Incremental M-mode Block SVD algorithm. The Incremental M-mode Block SVD is a generalized hierarchical part-based decomposition that computes an exact global Tucker decomposition and is suitable for streaming data, as opposed to the Hierarchical Tucker and Tensor train which that employs a hierarchical re-balancing of the modes trick to avoid computing SVDs of skinny matrices, but does not take into consideration the data variance which may result in t approximate subspace [**?**,Grasedyck (2010); Perros et al. (2015),**?**],Inverse causal inference may be addressed with a set of linear projections Vasilescu & Terzopoulos (2002b) or a single multilinear projection architecture [Vasilescu & Terzopoulos (2007); Vasilescu (2009)], that computes the inverse of the estimated forward model. The multilinear projection architecture geometrically is the upside down M-mode SVD architecture, Fig. 7.

After reviewing the mathematical foundations of our work in the next section, we discuss forward causal models and depict their neural network arcitectures in Section 3 and discuss inverse causal inference and depict the multilinear projection neural network architectures in Section 4. Section 5 concludes the paper.

2 MATHEMATICAL BACKGROUND

We will use standard textbook notation, denoting scalars by lower case italic letters $(a, b, \ldots)$, vectors by bold lower case letters $(\mathbf{a}, \mathbf{b}, \ldots)$, matrices by bold uppercase letters $(\mathbf{A}, \mathbf{B}, \ldots)$, and higher-order tensors by bold uppercase calligraphic letters $(\mathcal{A}, \mathcal{B}, \ldots)$. Index upper bounds are denoted by italic uppercase (*i.e.*, $1 \leq i \leq I$). The zero matrix is denoted by $\mathbf{0}$, and the identity matrix is denoted by $\mathbf{I}$. Kolda & Bader (2009)] provide a quick tutorial, but Vasilescu (2009); de Lathauwer (1997); Bro (1997) provide an in depth treatment of the field.

2.1 LINEAR AUTOENCODER AND LINEAR PCA

An autoencoder-decoder that minimizes the reconstruction loss function,

$$l = \sum_{\mathrm{i}=1}^{N} \|\mathbf{d}_{\mathrm{i}} - \mathbf{B}\mathbf{c}_{\mathrm{i}}\| + \lambda \|\mathbf{B}^{\mathrm{T}}\mathbf{B} - \mathbf{I}\| = \|\mathbf{D} - \mathbf{B}\mathbf{C}\| + \lambda \|\mathbf{B}^{\mathrm{T}}\mathbf{B} - \mathbf{I}\|, \tag{1}$$

and has a linear decoder learns a set of weights that are identical to PCA basis vectors when the weights of each neuron are computed sequentially. Thus, an autoencoder is implemented with a cascade of Hebb neurons [Hebb (1949)], Fig 3. The contribution of each neuron, $c_1, \ldots, c_{\mathrm{r}}$, is subtracted from a centered training data set, and the difference is driven through the next Hebb neuron, $c_{\mathrm{r+1}}$ [Oja (1982); Ackley et al. (1985); Rumelhart et al. (1986); Sanger (1989)]The weights of a Hebb neuron, c_{r}, are updated by

$$\Delta \mathbf{b}_{\mathrm{r}}(t+1) = \eta \left(\mathbf{d} - \sum_{i_{\mathrm{r}}=1}^{r} \mathbf{b}_{i_{\mathrm{r}}}(t) c_{i_{\mathrm{r}}}(t) \right) c_{\mathrm{r}}(t) \tag{2}$$

$$= \eta \left(\mathbf{d} - \sum_{i_{\mathrm{r}}=1}^{r} \mathbf{b}_{i_{\mathrm{r}}}(t) \mathbf{b}_{i_{\mathrm{r}}}^{\mathrm{T}}(t) \mathbf{d} \right) \mathbf{d}^{\mathrm{T}} \mathbf{b}_{r}(t),$$

$$\mathbf{b}_{\mathrm{r}}(t+1) = \frac{(\mathbf{b}_{\mathrm{r}}(t) + \Delta \mathbf{b}_{\mathrm{r}}(t+1))}{\|\mathbf{b}_{\mathrm{r}}(t) + \Delta \mathbf{b}_{\mathrm{r}}(t+1)\|} \tag{3}$$

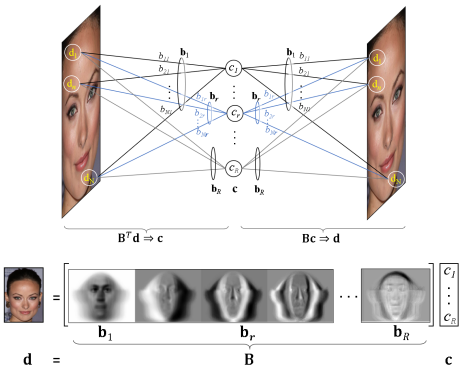

Figure 3: Autoencoder-decoder architecture and its standard mathematical counterpart, Principal Component Analysis. (All images have been vectorized, but they are displayed as a grid of numbers. The eigenvector $\mathbf{b}_1$ is the mean and activation c_1 is set to 1.)

where $\mathbf{d} \in \mathbb{C}^{I_0}$ is a vectorized centered observation with I_0 measurements, η is the learning rate, $\mathbf{b}_{\mathrm{r}}$ are the autoencoder weights of the r neuron, c_{r} is the activation, and t is the time iteration. Back-propagation [LeCun et al. (1988; 2012)] updates autoencoder weights by performing PCA gradient descent[Jolliffe (1986)].

2.2 RELEVANT TENSOR ALGEBRA

Briefly, the natural generalization of matrices (i.e., linear operators defined over a vector space), tensors define multilinear operators over a *set* of vector spaces. A *"data tensor"* denotes an M-way data array.

Definition 1 (Tensor) *Tensors are multilinear mappings over a set of vector spaces,* $\mathbb{R}^{I_m}$, $1 \leq m \leq M$, *to a range vector space* $\mathbb{R}^{I_0}$*:*

$$\mathcal{A} : \left\{ \mathbb{R}^{I_1} \times \mathbb{R}^{I_2} \times \cdots \times \mathbb{R}^{I_M} \right\} \mapsto \mathbb{R}^{I_0}.$$

The order of tensor $\mathcal{A} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_M}$ *is* $M+1$. *An element of* $\mathcal{A}$ *is denoted as* $\mathcal{A}_{i_0 i_1 \ldots i_m \ldots i_M}$ *or* $a_{i_0 i_1 \ldots i_m \ldots i_M}$, *where* $1 \leq i_m \leq I_m$.

The mode-m vectors of an M-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ are the I_m-dimensional vectors obtained from $\mathcal{A}$ by varying index i_m

Algorithm 1 M-mode SVD algorithm.

Input the data tensor $\mathcal{D} \in \mathbb{R}^{I_0 \times \cdots \times I_M}$.

1. For $m := 0, \ldots, M$,
   Let $\mathbf{U}_m$ be the left orthonormal matrix of $[\mathbf{U}_{\mathrm{m}} \mathbf{S}_{\mathrm{m}} \mathbf{V}_{\mathrm{m}}^{\mathrm{T}}] := \mathrm{svd}(\mathbf{D}_{[m]})$[a]
2. Set $\mathcal{Z} := \mathcal{D} \times_0 \mathbf{U}_0{}^{\mathrm{T}} \times_1 \mathbf{U}_1{}^{\mathrm{T}} \cdots \times_m \mathbf{U}_m{}^{\mathrm{T}} \cdots \times_M \mathbf{U}_M{}^{\mathrm{T}}$.

Output mode matrices $\mathbf{U}_0, \mathbf{U}_1 \ldots, \mathbf{U}_M$ and the core tensor $\mathcal{Z}$.

[a] When $\mathbf{D}_{[m]}$ is a nonsquare matrix, the computation of $\mathbf{U}_m$ in the SVD $\mathbf{D}_{[m]} = \mathbf{U}_m \mathbf{\Sigma} \mathbf{V}_m$ can be performed efficiently, depending on which dimension of $\mathbf{D}_{[m]}$ is smaller, by decomposing either $\mathbf{D}_{[m]}\mathbf{D}_{[m]} = \mathbf{U}_m \mathbf{\Sigma}^2 \mathbf{U}_m$ (note that $\mathbf{V}_m = \mathbf{\Sigma}^{+} \mathbf{U}_m \mathbf{D}_{[m]}$) or by decomposing $\mathbf{D}_{[m]}\mathbf{D}_{[m]} = \mathbf{V}_m \mathbf{\Sigma}^2 \mathbf{V}_m$ and then computing $\mathbf{U}_m = \mathbf{D}_{[m]} \mathbf{V}_m \mathbf{\Sigma}^{+}$.

while keeping the other indices fixed. In tensor terminology, column vectors are the mode-1 vectors and row vectors as mode-2 vectors. The mode-m vectors of a tensor are also known as *fibers*. The mode-m vectors are the column vectors of matrix $\mathbf{A}_{[m]}$ that results from *matrixizing* (a.k.a. *flattening*) the tensor $\mathcal{A}$.

Definition 2 (Mode-m Matrixizing) *The mode-m matrixizing of tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_M}$ is defined as the matrix $\mathbf{A}_{[m]} \in \mathbb{R}^{I_m \times (I_1 \ldots I_{m-1} I_{m+1} \ldots I_M)}$. As the parenthetical ordering indicates, the mode-m column vectors are arranged by sweeping all the other mode indices through their ranges, with smaller mode indexes varying more rapidly than larger ones; thus,*

$$[\mathbf{A}_{[m]}]_{jk} = a_{i_1 \ldots i_m \ldots i_M}, \quad \textit{where} \qquad j = i_m \quad \textit{and} \quad k = 1 + \sum_{\substack{n=0 \\ n \neq m}}^{M} (i_n - 1) \prod_{\substack{l=0 \\ l \neq m}}^{n-1} I_l. \quad (4)$$

A generalization of the product of two matrices is the product of a tensor and a matrix [de Lathauwer (1997); Carroll et al. (1980)].

Definition 3 (Mode-m Product, $\times_{\mathbf{m}}$) *The mode-m product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_m \times \cdots \times I_M}$ and a matrix $\mathbf{B} \in \mathbb{R}^{J_m \times I_m}$, denoted by $\mathcal{A} \times_m \mathbf{B}$, is a tensor of dimensionality $\mathbb{R}^{I_1 \times \cdots \times I_{m-1} \times J_m \times I_{m+1} \times \cdots \times I_M}$ whose entries are computed by*

$$[\mathcal{A} \times_m \mathbf{B}]_{i_1 \ldots i_{m-1} j_m i_{m+1} \ldots i_M} = \sum_{i_m} a_{i_1 \ldots i_{m-1} i_m i_{m+1} \ldots i_M} b_{j_m i_m},$$

$$\mathcal{C} = \mathcal{A} \times_m \mathbf{B}. \quad \underset{\textit{tensorize}}{\overset{\textit{matrixize}}{\rightleftarrows}} \quad \mathbf{C}_{[m]} = \mathbf{B}\mathbf{A}_{[m]}.$$

The M-mode SVD (aka. the Tucker decomposition), Alg. 1, is a "generalization" of the conventional matrix (i.e., 2-mode) SVD which may be written in tensor notation as

$$\mathbf{D} = \mathbf{S} \times_0 \mathbf{U}_0 \times_1 \mathbf{U}_1 \quad \Leftrightarrow \quad \mathbf{D} = \mathbf{U}_0 \mathbf{S} \mathbf{U}_1{}^{\mathrm{T}}$$

The M-mode SVD orthogonalizes the M spaces and decomposes a tensor as the *mode-m product*, denoted $\times_m$, of M-orthonormal mode matrices, and a core tensor $\mathcal{Z}$

$$\begin{aligned} \mathcal{D} &= \mathcal{Z} \times_0 \mathbf{U}_0 \cdots \times_m \mathbf{U}_m \cdots \times_M \mathbf{U}_M. && (5) \\ \mathbf{D}_{[m]} &= \mathbf{U}_{\mathrm{m}} \mathbf{Z}_{[m]} \left(\mathbf{U}_M \cdots \otimes \mathbf{U}_{\mathrm{m+1}} \otimes_{\mathrm{m-1}} \mathbf{U} \cdots \times_0 \mathbf{U}_0 \right)^{\mathrm{T}}, && (6) \\ vec(\mathcal{D}) &= \left(\mathbf{U}_M \cdots \otimes \mathbf{U}_{\mathrm{m+1}} \otimes \mathbf{U}_{\mathrm{m-1}} \cdots \otimes \mathbf{U}_0 \right) vec(\mathcal{Z}). && (7) \end{aligned}$$

The latter two equations express the decomposition in matrix form and in terms of the vec operator.

3 Forward Causal Inference with Causal Capsules

Forward causal inference is the estimation of effects of causes. Forward causal inference questions are framed in terms of interventions and they address the "what if" question. What if c were changed by one unit, how much would $\mathbf{d}$ be expected to change?

The causal relationship may be modeled by analyzing an ensemble of vectorized training observations organized in a M-way array of vectorized observations, aka. higher order "data tensor", $\mathcal{D}$. These observations are acquired by systematically varying each causal factor while holding the rest of them fixed. The best evidence comes from randomized controlled studies. Now, let us assume that $m = 0$ is the measurement mode and that the remaining modes $m = 1, \ldots, M$ are the causal factor modes; i.e., those modes associated with the relevant causal factors that underlie the generation of observations. For example, the pixel values (the measurements) in an image are the result of several the

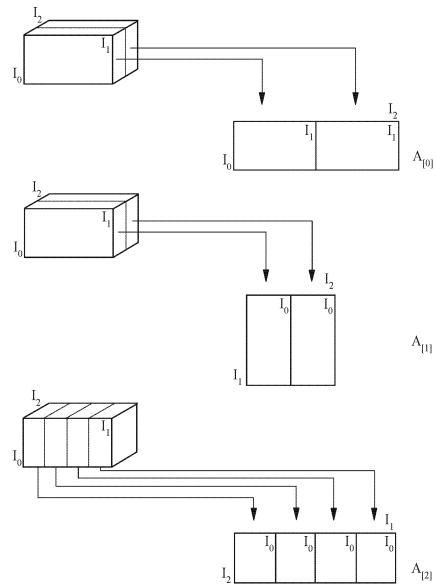

Figure 4: Matrixizing a (3rd-order) tensor. The tensor can be matrixized in 3 ways.

Algorithm 2 Kernel Multilinear PCA/ICA (K-MPCA/MICA) algorithm.

Input the data tensor $\mathcal{D} \in \mathbb{R}^{I_0 \times \cdots \times I_M}$, where mode $m = 0$ is the measurement mode, and the desired ranks $\tilde{R}_1, \ldots, \tilde{R}_M$.

1. For $m := 1, \ldots, M$,
   Compute the elements of the mode-m covariance matrix, for $j, k := 1, \ldots, I_m$, as follows:

$$[\mathbf{D}_{[m]}\mathbf{D}_{[m]}{}^{\mathrm{T}}]_{\mathrm{jk}} := \sum_{i_1=1}^{I_1} \cdots \sum_{i_{m-1}=1}^{I_{m-1}} \sum_{i_{m+1}=1}^{I_{m+1}} \cdots \sum_{i_M=1}^{I_M} K(\mathbf{d}_{i_1 \ldots i_{m-1}\, j\, i_{m+1} \ldots i_M}, \mathbf{d}_{i_1 \ldots i_{m-1}\, k\, i_{m+1} \ldots i_M}).$$

   $$\begin{cases} \textit{For K-MPCA:} & \text{Set } \mathbf{Y}_m \text{ to the left matrix of the SVD of } \mathbf{D}_{[m]}\mathbf{D}_{[m]}{}^{\mathrm{T}} = \mathbf{Y}_m \mathbf{\Sigma}^2 \mathbf{Y}_m{}^{\mathrm{T}} \\ & \text{Truncate to } \tilde{R}_m \text{ columns } \mathbf{Y}_m \in \mathbb{R}^{I_m \times \tilde{R}_m}. \\ \textit{For K-MICA:} & \text{Compute } \mathbf{Y}_m := \mathbf{C}_m \in \mathbb{R}^{I_m \times \tilde{R}_m} \text{ using Vasilescu \& Terzopoulos (2005)} \\ & \text{and truncating the initial SVD to dimensionality } \tilde{R}_m. \end{cases}$$

2. Set $\mathcal{T} := \mathcal{D} \times_1 \mathbf{Y}_1^+ \cdots \times_m \mathbf{Y}_m^+ \cdots \times_M \mathbf{Y}_M^+$.
3. *Local optimization via alternating least squares:*
   Iterate for $n := 1, \ldots, N$
   For $m := 1, \ldots, M$,
   Set $\mathcal{X}_{\mathrm{m}} := \mathcal{D} \times_1 \mathbf{Y}_1^+ \cdots \times_{m-1} \mathbf{Y}_{m-1}^+ \times_{m+1} \mathbf{Y}_{m+1}^+ \cdots \times_M \mathbf{Y}_M^+$.
   Set $\mathbf{Y}_m$ to the $\tilde{R}_m$ leading left-singular vectors of the SVD of $\mathbf{X}_{\mathrm{m,[m]}}$.
   Set $\mathcal{T} := \mathcal{X} \times_M \mathbf{Y}_M^+$.
   until convergence.

Output the converged extended core tensor $\mathcal{T} \in \mathbb{R}^{I_0 \times \tilde{R}_1 \times \cdots \times \tilde{R}_M}$ and causal factor mode matrices $\mathbf{Y}_1, \ldots, \mathbf{Y}_M$.

causal factors associated with the scene structure, viewing conditions (camera location, orientation, lens type, etc), illumination conditions (light location, orientation and type) *e*tc., Fig 2(a).

Therefore, let a "data-tensor", $\mathcal{D} \in \mathbb{C}^{I_0 \times I_1 \cdots \times I_{\mathrm{m}} \cdots \times I_{\mathrm{M}}}$, contains a collection of vectorized and centered observations, $\mathbf{d}_{i_1 \ldots i_{\mathrm{m}} \ldots i_{\mathrm{M}}} \in \mathbb{C}^{I_0}$, that are the result of M causal factors. The m causal factor ($1 \leq m \leq M$) takes one of I_m values that are indexed by i_m, $1 \leq i_m \leq I_m$. An observation that is result of the confluence M causal factors is modeled by a multilinear structural equation with multimode latent variables, $\mathbf{r}_{\mathrm{m}}$, that represent the causal factors

$$\mathbf{d}_{i_1,\ldots,i_{\mathrm{m}},\ldots,i_{\mathrm{M}}} = \mathcal{T} \times_1 (\mathbf{r}_{i_1}^{\mathrm{T}} + \epsilon_{i_1}^{\mathrm{T}}) \cdots \times_{\mathrm{m}} (\mathbf{r}_{i_{\mathrm{m}}}^{\mathrm{T}} + \epsilon_{i_{\mathrm{m}}}^{\mathrm{T}}) \cdots \times_{\mathrm{M}} (\mathbf{r}_{i_{\mathrm{M}}}^{\mathrm{T}} + \epsilon_{i_{\mathrm{M}}}^{\mathrm{T}}) + \epsilon i_1, \ldots, i_{\mathrm{m}}, \ldots, i_{\mathrm{M}} \quad (8)$$

where $\mathcal{T} = \mathcal{Z} \times_0 \mathbf{U}_0$ is the extended core which modulates the interaction between the latent variables, and $\boldsymbol{\epsilon}_{i_{\mathrm{c}}} \in \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ is an Gaussian noise, Fig. 2(a0.

The M-mode SVD [Vasilescu & Terzopoulos (2002a)], Alg 1, the generalized MPCA that employs kernel nonlinearities [Vasilescu (2009)], Alg. 2, and their neural network architecture counterparts, Fig. 1, Fig. 5 may be employed to represent data in terms of their causal factors by minimizing the reconstruction error e,

$$e = \|\mathcal{D} - \mathcal{T} \times_1 \mathbf{U}_1 \cdots \times_{\mathrm{m}} \mathbf{U}_{\mathrm{m}} \cdots \times_{\mathrm{M}} \mathbf{U}_{\mathrm{M}}\| + \sum_{m=1}^{M} \lambda_{\mathrm{m}} \|\mathbf{U}_{\mathrm{m}} \mathbf{U}_{\mathrm{c}}^{\mathrm{T}} - \mathbf{I}\|, \quad (9)$$

where $\mathcal{T}$ is the extended core and the mode matrices, $\mathbf{U}_{\mathrm{m}}$, spans the m causal factor representation given the correct experimental design and problem setup.

Each mode matrix, $\mathbf{U}_{\mathrm{m}}$ was computed using alternating least squares

$$\mathcal{X}_{\mathrm{m}} := \mathcal{D} \times_1 \cdots \times_{\mathrm{m-1}} \mathbf{U}_{\mathrm{m-1}}^{\mathrm{T}} \times_{\mathrm{m+1}} \mathbf{U}_{\mathrm{m+1}}^{\mathrm{T}} \cdots \times_{\mathrm{M}} \mathbf{U}_{\mathrm{M}}^{\mathrm{T}} \quad (10)$$

and setting $\mathbf{U}_{\mathrm{m}}$ to the subspace of the matrixized $\mathcal{X}_{\mathrm{m}}$, $\mathbf{U}_{\mathrm{m}} \mathbf{S}_{\mathrm{m}} \mathbf{V}_{\mathrm{m}}^{\mathrm{T}} := \mathrm{svd}(\mathbf{X}_{\mathrm{m}[m]})$. Figure 5(d) displays the mode matris computations for one time itteration by unrolling the for-loop from step 3, Alg 2.

Linear kernel:	$K(\mathbf{u},\mathbf{v}) = \mathbf{u}^{\mathrm{T}}\mathbf{v} = \mathbf{u}\cdot\mathbf{v}$
Polynomial kernel of degree d:	$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}^{\mathrm{T}}\mathbf{v})^d$
Polynomial kernel up to degree d:	$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}^{\mathrm{T}}\mathbf{v}+1)^d$
Sigmoidal kernel:	$K(\mathbf{u},\mathbf{v}) = \tanh(\alpha\,\mathbf{u}^{\mathrm{T}}\mathbf{v}+\beta)$
Gaussian (radial basis function (RBF)) kernel:	$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{\lVert\mathbf{u}-\mathbf{v}\rVert^2}{2\sigma^2}\right)$

Table 1: Common kernel functions. Kernel functions are symmetric, positive semi-definite functions (corresponding to symmetric, positive semi-definite Gram matrices). The linear kernel does not modify or warp the feature space.

3.1 Causal Capsules and M-mode SVD

Computing the mode matrices, $\mathbf{U}_c$, may be viewed as equivalent to computing a set of mutually constrained, cluster-based PCAs. When dealing with data that can be separated into clusters, the standard machine learning approach is to compute a separate PCA. When data from different clusters are generated by the same underlying process (e.g., facial images of the same people under different viewing conditions), the underlying data can be concatenated in the measurement mode and the common causal factor can be modeled by one PCA.[1]

Thus, we define a *constrained, cluster-based PCA* as the computation of a set of PCA basis vectors that are rotated such that the latent representation is constrained to be the invariant of the cluster. MPCA performs M constrained, cluster-based PCAs, since the computation of the mode-m matrix $\mathbf{U}_m$, which involves a mode-c data tensor flattening and subsequent SVD, is equivalent to performing a constrained, cluster-based PCA; i.e., data cluster concatenation followed by an SVD. This is self evident when employing our modified datum-centric flattening operator, Fig. 4. In the context of our multifactor data analysis, we define a cluster as a set of observations for which all factors are fixed except one, the c factor. Note that there are $N_m = I_1 I_2 \ldots I_{m-1} I_{m+1} \ldots I_M$ possible clusters and the data in each cluster varies with the same causal mode.[2] Thus, the data across different clusters share one of the underlying causal factors. The constrained, cluster-based PCA concatenates the clusters in the measurement mode and analyzes the data with a linear model, such as PCA.

To see this, let $\mathcal{D}_{i_1\ldots i_{m-1} i_{m+1}\ldots i_M} \in \mathbb{R}^{I_0\times 1\times 1\cdots\times m\times I_m\times 1\cdots\times 1}$ denote a subtensor of $\mathcal{D}$ that is obtained by fixing all modes except causal factor mode m and mode (the measurement mode). Matrixizing this subtensor in the measurement mode 0, we obtain $\mathbf{D}_{i_1\ldots i_{m-1} i_{m+1}\ldots i_M\,[0]} \in \mathbb{R}^{I_0\times I_m}$. This data matrix comprises a cluster of data obtained by varying the m causal factor, to which one can traditionally apply PCA. Since there are $N_m = I_1 I_2 \ldots I_{m-1} I_{m+1} \ldots I_M$ possible clusters that share the same underlying space associated with the $cmth$ factor, the data can be concatenated and PCA performed in order to extract the same representation for the m factor regardless of the cluster. Now, consider the MPCA computation of mode matrix $\mathbf{U}_m$, Fig. 5(a), which can be written in terms of matrixized subtensors as

$$\mathbf{D}_{[\mathrm{m}]}^{\mathrm{T}} = \begin{bmatrix} \mathbf{D}_{1\ldots 11\ldots 1[m]}{}^{\mathrm{T}} \\ \vdots \\ \mathbf{D}_{I_1\ldots 11\ldots 1\,[m]}{}^{\mathrm{T}} \\ \vdots \\ \mathbf{D}_{I_q\ldots I_{m-1} I_{m+1}\ldots I_M\,[m]}{}^{\mathrm{T}} \end{bmatrix} = \mathbf{V}_m \mathbf{\Sigma}_m \mathbf{U}_m{}^{\mathrm{T}}. \tag{11}$$

Clearly, this is equivalent to computing a set of $N_m = I_1 I_2 \ldots I_{m-1} I_{m+1} \ldots I_M$ cluster-based PCAs concurrently by combining them into a single statistical model and representing the underlying causal factor m common to the clusters. Thus, rather than computing a separate linear PCA model

[1] The active appearance model [Cootes et al. (2001); Si et al. (2013)] concatenated two different measurements, facial feature locations and texture, to compute a person representation for a particular viewpoint invariant of measurement. More c generally, one can concatenate the measurements of a person from different viewpoint clusters to compute a person representation that is invariant of the measurement and the viewpoint.

[2] Observations in the cluster may not be in Euclidean proximity in the measurement space. Consequently, a cluster may not be easily identified through a standard EM algorithm.

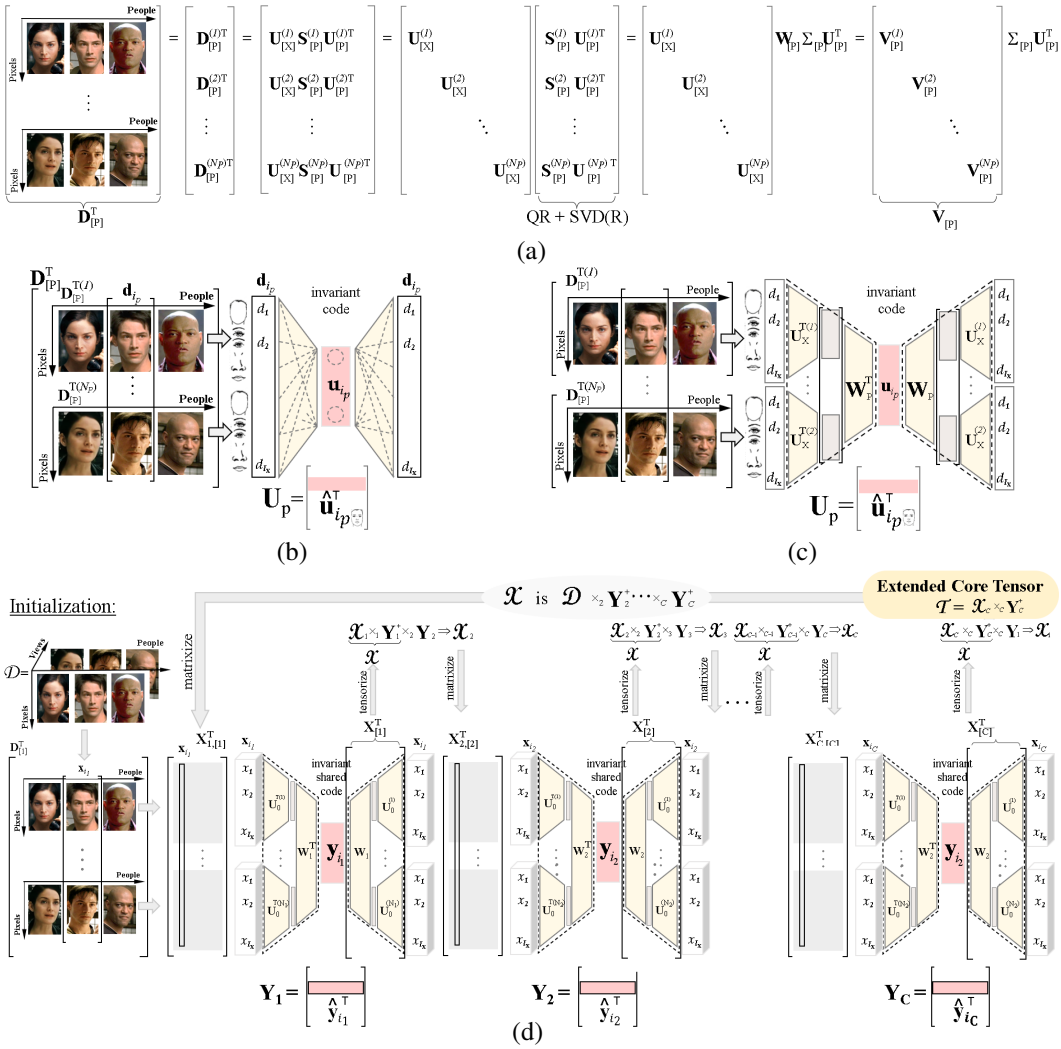

Figure 5: Face recognition example (a) Mode matrix computation, $\mathbf{U}_{\mathrm{P}}$, that spans the person representation is written in terms of constrained cluster-based PCA. This is depicted as a neural network architecture in (c). (b) Mode matrix computation using a single autoencoder-decoder. (c) Mode matrix computation using a constrained cluster-based autoencoder-decoder based on the derivation in part (a). (e) The neural network architecture consists of a cascade of constrained autoencoders-decoders where the weights of one constrained autoencoder-decoder are the inputs of the next one. This constrained recurrent causal chain is the unrolled for-loop that computes the mode matrices by employing alternating least squares. When the autoencoders employ kernels then the architecture implements M-MPCA/ K-MICA, Alg. 2.

for each cluster, MPCA concatenates the clusters into a single statistical model and computes a representation (coefficient vector) for mode m that is invariant relative to the other causal factor modes $1, \ldots, (m-1), (m+1), \ldots, M$. Therefore, MPCA is a multilinear, constrained, cluster-based PCA.

To clarify the relationship, let us number each of the matrices

$$\mathbf{D}_{i_1 \ldots i_{m-1} i_{m+1} \ldots i_{M\,[m]}} = \mathbf{D}_m^{(n)} \tag{12}$$

with a parenthetical superscript $1 \leq n = 1 + \sum_{k=1, k\neq c}^{M} (i_n - 1) \prod_{l=1, l\neq c}^{k-1} I_l \leq N_c$. Let each of the linear SVDs be

$$\mathbf{D}_m^{(n)} = \mathbf{U}_c^{(n)} \boldsymbol{\Sigma}_m^{(n)} \mathbf{V}_m^{(n)\mathrm{T}} \tag{13}$$

$$\mathbf{D}_{[m]} = \left[\mathbf{U}_m^{(1)} \boldsymbol{\Sigma}_m^{(1)} \ldots \mathbf{U}_m^{(N_M)} \boldsymbol{\Sigma}_m^{(N_m)}\right] \mathrm{diag}([\mathbf{V}_m^{(1)} \ldots \mathbf{V}_m^{(N_m)}])^{\mathrm{T}}, \tag{14}$$

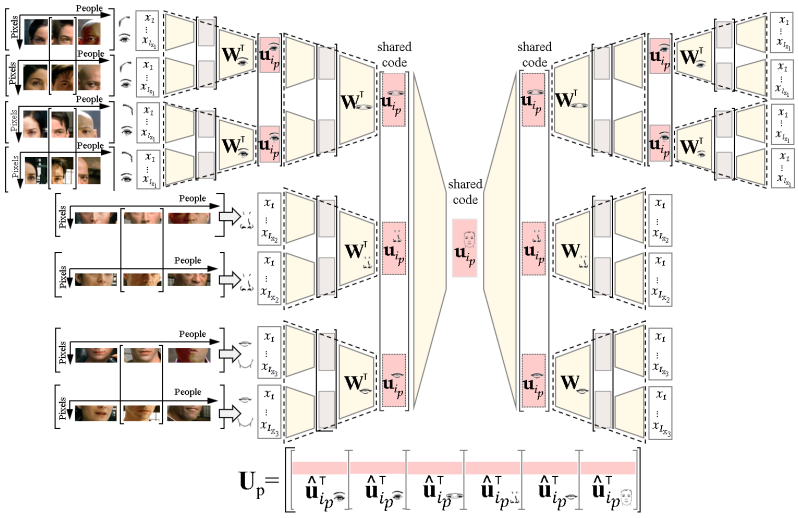

Figure 6: Learning levels of abstractions bottom-up. This is a neural network implementation of the hierarchical part-based block tensor decomposition, the Incremental M-mode Block SVD. Each mode matrix computations in Fig. 1 and Fig. 5 with the hierarchical part based block tensor decomposition depicted above.

where $\mathrm{diag}(\cdot)$ denotes a diagonal matrix whose elements are each of the elements of its vector argument. The mode matrix $\mathbf{V}_m^{(n)}$ is the measurement matrix $\mathbf{U}_0^{(n)}$ that contains the eigenvectors that span the observed data in cluster n. The SVD of the first matrix is

$$\begin{aligned} [\mathbf{U}_m^{(1)}\mathbf{\Sigma}_m^{(1)} \ldots \mathbf{U}_m^{(N_m)}\mathbf{\Sigma}_m^{(N_m)}] &= \mathbf{U}_m \mathbf{\Sigma}_m \mathbf{W}_m{}^{\mathrm{T}} \\ \mathbf{D}_{[m]} &= \mathbf{U}_m \mathbf{\Sigma}_m \mathbf{V}_m{}^{\mathrm{T}}, \text{ where } \mathbf{V}_m{}^{\mathrm{T}} = \mathbf{W}_m{}^{\mathrm{T}} \, \mathrm{diag}([\mathbf{V}_m^{(1)} \ldots \mathbf{V}_m^{(N_m)}])^{\mathrm{T}} \end{aligned} \tag{15}$$

Matrix $\mathbf{W}_m$ rotates each of the measurement mode matrices $\mathbf{V}_m^{(n)}$ for each cluster computed by linear PCA such that the representation $\mathbf{U}_m$ is the same across the clusters, Fig. **??**(a).

The architecture of the M-mode SVD is depicted in Fig. 1. The architecture for M-mode SVD for subspace analysis 2 is depicted in Fig. 5(d). Performing one iteration of the M-mode SVD and returning the extended core tensors $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_{\mathrm{C}}$ is equivalent to performing a tensor train subspace decomposition.

Data "clusters" can be formed from parts and not just a complete observation, Fig **??**.

3.2 Nonlinear Causal Capsules and Kernel MPCA/Kernel MICA

An autoencoder with a non-linear activation function represents an observation with

$$\mathbf{d}_{\mathrm{i}} = f_{\mathrm{d}}(\mathbf{B}_{\mathrm{d}} \underbrace{f_{\mathrm{e}}(\mathbf{B}_{\mathrm{e}} d_{\mathrm{i}} + \mathbf{a}_{\mathrm{e}})}_{\mathbf{c}_{\mathrm{i}}} + \mathbf{a}_{\mathrm{d}}), \tag{16}$$

where f_{e}, f_{d} are the encoder, decoder activation functions, and $\mathbf{B}_{\mathrm{e}}, \mathbf{B}_{\mathrm{d}}$ are the encoder, decoder weights respectively. Kernel PCA (KPCA) is often given as an example of a "true" nonlinear model. KPCA first applies a nonlinear transformation to the data and then it performs a linear decomposition. Thus, KPCA derives its nonlinearity from its preprocessing step. Other nonlinear methods include nonlinear PCA (NLPCA) Kramer (1991), as well as kernel PCA (KPCA) Schölkoph et al. (1998) and kernel LDA (KLDA) Yang (2002) methods in which kernel functions that satisfy Mercer's theorem correspond to inner products in infinite-dimensional space. An alternative approach is to apply linear models to nonlinear problems through the "kernel trick", specifically the kernel PCA Schölkoph et al. (1998) and kernel ICA Yang et al. (2005) techniques.[3] Kernel PCA/ICA are nonlinear versions of their conventional linear counterparts, but of course they are not multimodal factor models.

[3]The so-called "kernel trick" maps the original non-linear measurements into a higher-dimensional space, where a linear classifier is subsequently used; this makes a linear classification in the new space equivalent to non-linear classification in the original space. This is done using Mercer's theorem, which states that any continuous, symmetric, positive semi-definite kernel $K(\mathbf{u}, \mathbf{v})$ can be expressed as an inner product in a high-dimensional space. Wherever an inner product between two vectors is used, it is replaced with a kernel of the vectors. Thus, a linear algorithm is easily transformed into a nonlinear algorithm. This trick has been applied to numerous algorithms in machine learning and statistics.

The kernel trick can also be applied to our multilinear, multifactor PCA/ICA models to further nonlinearize them, thus enabling them to deal with arbitrarily nonlinear data, Alg. 2.

To accomplish this, recall that the computation of the mode-m covariance matrix $\mathbf{D}_{[m]}\mathbf{D}_{[m]}{}^{\mathrm{T}}$ involves inner products $\mathbf{d}^{\mathrm{T}}_{i_1\ldots i_{\mathrm{m-1}}\, j\, i_{\mathrm{m+1}}\ldots i_{\mathrm{M}}}\mathbf{d}_{i_2\ldots i_{\mathrm{m-1}}\, k\, i_{\mathrm{m+1}}\ldots i_{\mathrm{M}}}$ between pairs of images in the image data tensor $\mathcal{D}$ associated with causal factor mode m, for $m = 2, \ldots, M$. We replace the inner products with a generalized distance measure between images, $K(\mathbf{d}_{i_2\ldots i_{m-1}\, j\, i_{m+1}\ldots i_M}, \mathbf{d}_{i_2\ldots i_{m-1}\, k\, i_{m+1}\ldots i_M})$, where $K(\cdot,\cdot)$ is a suitable kernel function (Table 1), which corresponds to an inner product in some expanded feature space. This generalization naturally leads us to a *Kernel Multilinear PCA (K-MPCA) Algorithm*, where the covariance step computation in in Algorithm 1 is replaced by

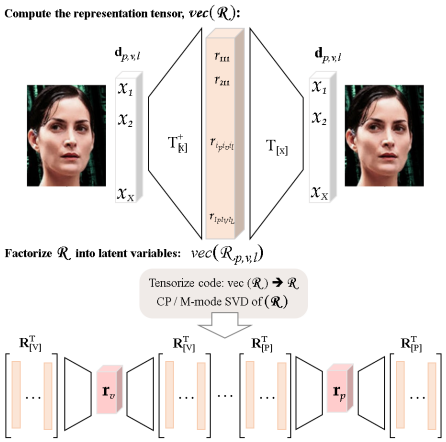

Figure 7: Neural network architecture of the multilinear projection [Vasilescu & Terzopoulos (2007)] given an estimated forward causal model, $\mathcal{T}$ (*i.e.*, $\mathbf{T}_{[\mathrm{x}]}$). Mathematically performing a pseudo-inverse of a multilinear forward model results is equivalent to inverting the architecture of the multilinear (tensor) forward model, Fig **??**

$$[\mathbf{D}_{[m]}\mathbf{D}_{[m]}{}^{\mathrm{T}}]_{jk} := \sum_{i_2=1}^{I_2}\cdots\sum_{i_{m-1}=1}^{I_{m-1}}\sum_{i_{m+1}=1}^{I_{m+1}}\cdots\sum_{i_M=1}^{I_M} K(\mathbf{d}_{i_2\ldots i_{m-1}\, j\, i_{m+1}\ldots i_M}, \mathbf{d}_{i_2\ldots i_{m-1}\, k\, i_{m+1}\ldots i_M}). \tag{17}$$

Similarly, a *Kernel Multilinear ICA (K-MICA) Algorithm* results from making the same generalization in the MICA algorithm [Vasilescu & Terzopoulos (2005)]. Algorithm 2 specifies both K-MPCA and K-MICA. Figure 5(d) unrolls the for-loop in step 3 of Alg. 2.

3.3 Inverse Causal Inference with Multilinear Projection

Inverse causal inference addresses the "why" question, to which Donald Rubin referred to as the "cocktail party chatter" [[Gelman & Imbens (2013)]. This is not surprising since inverse problems often violate one of the conditions of a well-posed problem[4]. Often there are multiple combinations of same causal factors that can produce the same outcome. In imaging these types of outcomes are known as visual illusions. Therefore, we define inverse causal inference as the estimation of causes of effects given an estimated forward causal model and a set of unlabeled observations that constrain the solution set and render the problem well-posed [Vasilescu (2011); Vasilescu & Terzopoulos (2007); Vasilescu et al. (2021)].

Multilinear projection [Vasilescu (2011); Vasilescu & Terzopoulos (2007)] simultaneously projects one or more unlabeled test images that are not part of the training data set into multiple constituent causal factor spaces associated with data formation, in order to infer the mode labels:

$$\mathcal{R} = \mathcal{T}^{+_{\mathrm{x}}} \times_{\mathrm{x}}^{\mathrm{T}} \mathbf{d}_{\mathrm{new}} \qquad M\text{-mode SVD}(\mathcal{R}) \approx \mathbf{r}_1 \cdots \circ \mathbf{r}_{\mathrm{m}} \cdots \circ \mathbf{r}_{\mathrm{M}} \circ \mathbf{r}_{\mathrm{E}}. \tag{18}$$

The multilinear projection architecture geometrically is the upside down M-mode SVD architecture, Fig. 7.

4 Conclusion

This paper provides a bridge between multilinear structural equations and deep neural network architectures. We introduce a causal capsule hierarchy architecture that implements tensor methods for forward and inverse causal inference.

References

Evrim Acar, Evangelos E Papalexakis, Gözde Gürdeniz, Morten A Rasmussen, Anders J Lawaetz, Mathias Nilsson, and Rasmus Bro. Structure-revealing data fusion. *BMC bioinformatics*, 15(1):239, 2014. 2

[4]For a problem to be well-posed a solution must exit, it must be unique and the solution's behaviour ought to change continuously with the initial conditions[[Hadamard (1952)].

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. ISSN 0364-0213. doi: https://doi.org/10.1016/S0364-0213(85)80012-4. URL https://www.sciencedirect.com/science/article/pii/S0364021385800124. 3

Peter M. Bentler and Sik-Yum Lee. A statistical development of three-mode factor analysis. *British J. of Math. and Stat. Psych.*, 32(1):87–104, 1979. doi: 10.1111/j.2044-8317.1979.tb00754.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2044-8317.1979.tb00754.x. 2

Kenneth A. Bollen and Judea Pearl. *Eight Myths About Causality and Structural Equation Models*, pp. 301–328. Springer Netherlands, Dordrecht, 2013. ISBN 978-94-007-6094-3. doi: 10.1007/978-94-007-6094-3_15. URL https://doi.org/10.1007/978-94-007-6094-3_15. 1

Rasmus Bro. Parafac: Tutorial and applications. In *Chemom. Intell. Lab Syst., Special Issue 2nd Internet Cont. in Chemometrics (INCINC'96)*, volume 38, pp. 149–171, 1997. 2, 3

J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of 'Eckart-Young' decomposition. *Psychometrika*, 35:283–319, 1970. 2

J. D. Carroll, S. Pruzansky, and J. B. Kruskal. CANDELINC: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. *Psychometrika*, 45:3–24, 1980. 4

Wei Chu and Zoubin Ghahramani. Probabilistic models for incomplete multi-dimensional arrays. volume 5 of *Proceedings of Machine Learning Research*, pp. 89–96, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR. URL http://proceedings.mlr.press/v5/chu09a.html. 2

T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001. 6

J. Davis and H. Gao. Recognizing human action efforts: An adaptive three-mode PCA framework. In *Proc. IEEE Inter. Conf. on Computer Vision, (ICCV)*, pp. 1463–69, Nice, France, Oct 13-16 2003. 2

L. de Lathauwer. Decompositions of a higher-order tensor in block terms—part ii: Definitions and uniqueness. *SIAM J. on Matrix Analysis and Applications*, 30(3):1033–1066, 2008. doi: 10.1137/070690729. URL https://doi.org/10.1137/070690729. 2

L. de Lathauwer and D. Nion. Decompositions of a higher-order tensor in block terms–part iii: Alternating least squares algorithms. *SIAM J. on Matrix Analysis and Applications*, pp. 1067–1083, 2008. 2

Lieven de Lathauwer. *Signal Processing Based on Multilinear Algebra*. PhD dissertation, Katholieke Univ. Leuven, Belgium, 1997. 2, 3, 4

Andrew Gelman and Guido Imbens. Why ask why? forward causal inference and reverse causal questions. Tech.report, Nat.Bureau of Econ Research, 2013. 1, 9

Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM J. on Matrix Analysis and Applications*, 31(4):2019–54, 2010. 2

J. Hadamard. *Lectures on Cauchy's Problem in Linear Partial Differential Equations*. Dover Publisher, 1952. 9

R. Harshman. Foundations of the PARAFAC procedure: Model and conditions for an explanatory factor analysis. Tech. Report Working Papers in Phonetics 16, UCLA, CA, Dec 1970. 2

D. O. Hebb. *The organization of behavior: A neuropsychological theory*. John Wiley And Sons, Inc., New York, 1949. 3

Geoffrey Hinton. How to represent part-whole hierarchies in a neural network, 2021. 2

Geoffrey E. Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders, 2011. 2

Eugene Hsu, Kari Pulli, and Jovan Popovic. Style translation for human motion. *ACM Transactions on Graphics*, 24(3):1082–89, 2005. URL http://people.csail.mit.edu/jovan/assets/papers/hsu-2005-stf.pdf. 2

Guido W. Imbens and Donald B. Rubin. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge Univ. Press, 2015. ISBN 0521885884. 1

I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986. 3

A. Kapteyn, H. Neudecker, and T. Wansbeek. An approach to n-mode component analysis. *Psychometrika*, 51 (2):269–275, Jun 1986. 2

Valentin Khrulkov. *Geometrical Methods in Machine Learning and Tensor Analysis*. PhD dissertation, Skolkovo Institute of Science and Technology, 2020. 2

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *CoRR*, abs/1511.06530, 2015. URL http://arxiv.org/abs/1511.06530. 2

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 3

Jean Kossaifi, Aran Khanna, Zachary C. Lipton, Tomasso Furlanello, and Anima Anandkumar. Tensor contraction layers for parsimonious deep nets. In *Computer Vision and Pattern Recognition (CVPR), Tensor Methods in Computer Vision Workshop*, pp. 1940–46, Jul 2017. 2

Jean Kossaifi, Zachary C. Lipton, Arinbjorn Kolbeinsson, Aran Khanna, Tommaso Furlanello, and Anima Anandkumar. Tensor regression networks. *Journal of Machine Learning Research*, 21(123):1–21, 2020. URL http://jmlr.org/papers/v21/18-503.html. 2

M. A. Kramer. Nonlinear principal components analysis using autoassociative neural networks. *AIChE J.*, 32(2): 233–243, Feb 1991. 8

P. M. Kroonenberg and J. de Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45:69–97, 1980. 2

Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan V. Oseledets, and Victor S. Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *CoRR*, abs/1412.6553, 2014. URL http://arxiv.org/abs/1412.6553. 2

Yann LeCun, D Touresky, G Hinton, and T Sejnowski. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pp. 21–28, 1988. 3

Yann A. LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. *Efficient BackProp*, pp. 9–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi: 10.1007/978-3-642-35289-8_3. URL https://doi.org/10.1007/978-3-642-35289-8_3. 3

J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley & Sons, 1988. 2

Alexander Novikov, Dmitrii Podoprikhin, Anton Osokin, and Dmitry P Vetrov. Tensorizing neural networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 28*, pp. 442–450. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5787-tensorizing-neural-networks.pdf. 2

Erkki Oja. A simplified neuron model as a principal component analyzer. 15:267–2735, 1982. 3

I. V. Oseledets. Tensor-train decomposition. *SIAM J. on Scientific Computing*, 33(5):2295–2317, 2011. doi: 10.1137/090752286. URL https://doi.org/10.1137/090752286. 2

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge Univ. Press, 2000. ISBN 9780521773621. URL https://books.google.com/books?id=wnGU_TsW3BQC. 1, 2

I. Perros, R. Chen, R. Vuduc, and J. Sun. Sparse hierarchical tucker factorization and its application to healthcare. In *Proc. IEEE Inter. Conf. on Data Mining*, pp. 943–948, Nov 2015. doi: 10.1109/ICDM.2015.29. 2

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. *Learning internal representations by error propagation*. 1986. 3

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/2cad8fa47bbef282badbb8de5374b894-Paper.pdf. 2

Terry Sanger. Optimal unsupervised learnig in a single layer linear feedforward neural network. 12:459–473, 1989. 3

B. Schölkoph, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998. 8

Weiguang Si, Kota Yamaguchi, and M. Alex O. Vasilescu. Face tracking with multilinear (tensor) active appearance models. http://pdfs.semanticscholar.org/6c64/59d7cadaa210e3310f3167dc181824fb1bff.pdf, Jun 2013. URL https://pdfs.semanticscholar.org/6c64/59d7cadaa210e3310f3167dc181824fb1bff.pdf. 6

Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000. 1

Yichuan Tang, Ruslan Salakhutdinov, and Geoffrey Hinton. Tensor analyzers. volume 28 of *Proceedings of Machine Learning Research*, pp. 163–171, Atlanta, Georgia, USA, 17–19 Jun 2013. URL http://proceedings.mlr.press/v28/tang13.html. 2

L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966. 2

M. A. O. Vasilescu. Human motion signatures: Analysis, synthesis, recognition. In *Proc. Int. Conf. on Pattern Recognition*, volume 3, pp. 456–460, Quebec City, Aug 2002. 2

M. A. O. Vasilescu. Multilinear projection for face recognition via canonical decomposition. In *Proc. IEEE Inter. Conf. on Automatic Face Gesture Recognition (FG 2011)*, pp. 476–483, Mar 2011. doi: 10.1109/FG.2011.5771445. 1, 9

M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: TensorFaces. In *Proc. European Conf. on Computer Vision (ECCV 2002)*, pp. 447–460, Copenhagen, Denmark, May 2002a. 1, 5

M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis for facial image recognition. In *Proc. Int. Conf. on Pattern Recognition*, volume 2, pp. 511–514, Quebec City, Aug 2002b. 2

M. A. O. Vasilescu and D. Terzopoulos. TensorTextures: Multilinear image-based rendering. *ACM Transactions on Graphics*, 23(3):336–342, Aug 2004. Proc. ACM SIGGRAPH 2004 Conf., Los Angeles, CA. 2

M. A. O. Vasilescu and D. Terzopoulos. Multilinear independent components analysis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume I, pp. 547–553, San Diego, CA, 2005. 1, 5, 9

M. A. O. Vasilescu and D. Terzopoulos. Multilinear projection for appearance-based recognition in the tensor framework. In *Proc. 11th IEEE Inter. Conf. on Computer Vision (ICCV'07)*, pp. 1–8, 2007. 1, 2, 9

M. Alex O. Vasilescu. *A Multilinear (Tensor) Algebraic Framework for Computer Graphics, Computer Vision, and Machine Learning*. PhD dissertation, University of Toronto, 2009. 1, 2, 3, 5

M. Alex O. Vasilescu and Eric Kim. Compositional hierarchical tensor factorization: Representing hierarchical intrinsic and extrinsic causal factors. In *The 25th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD'19): Tensor Methods for Emerging Data Science Challenges Workshop*, Aug. 5 2019. 2

M. Alex O. Vasilescu, Eric Kim, and Xiao S. Zeng. Causalx: Causal explanations and block multilinear factor analysis. In *2020 25th International Conference of Pattern Recognition (ICPR 2020)*, pp. 10736–10743, Jan 2021. 1, 2, 9

D. Vlasic, M. Brand, H. Pfister, and J. Popovic. Face transfer with multilinear models. *ACM Transactions on Graphics (TOG)*, 24(3):426–433, Jul 2005. URL http://people.csail.mit.edu/jovan/assets/papers/vlasic-2005-ftm.pdf. 2

Hongcheng Wang and Narendra Ahuja. Facial expression decomposition. In *Proc, 9th IEEE Inter. Conf. on Computer Vision (ICCV)*, pp. 958–65,v.2, 2003. doi: 10.1109/ICCV.2003.1238452. 2

M. Wang, Y. Panagakis, P. Snape, and S. Zafeiriou. Learning the multilinear structure of visual data. In *2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 6053–6061, Jul 2017. doi: 10.1109/CVPR.2017.641. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.641. 2

J. Yang, X. Gao, D. Zhang, and J. Yang. Kernel ICA: An alternative formulation and its application to face recognition. *Pattern Recognition*, 38(10):1784–87, 2005. 8

M. Yang. Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods. In *Proc. 5th IEEE Inter. Conf. on Automatic Face Gesture Recognition*, pp. 215–220, Washington, DC, May 2002. doi: 10.1109/AFGR.2002.4527207. 8