ORTHOGONAL AUTO-SELECTIVE ENCODING TUCKER FACTORIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Tucker factorization provides a promising framework for learning representations of high-order tensor data by modeling multiplicative interactions among mode-specific embeddings. The outer-product construction from mode-specific embeddings defines a latent interaction space whose coordinates are not explicitly organized according to their relevance to downstream tasks. A central challenge is to adapt this space so that task-relevant interaction directions can be identified and retained. We propose Orthogonal Auto-Selective Encoding Tucker Factorization (OASENT), which combines an orthogonal coordinate transformation with a task-driven selection module. A Householder-parameterized layer learns a norm-preserving coordinate transformation. A Gumbel-Softmax selector then forms a compact K-channel task representation using continuous masks during training and deterministic coordinate selections at inference. The nonlinear decoder then maps this compact representation to the original interaction dimension for prediction. Our analysis shows that a full-dimensional orthogonal transformation alone leaves the represented decoder function class unchanged. Subsequent selection forms a bottleneck with effective width $J_{\text{eff}} \leq K$ at inference. Across seven real-world tensor datasets, OASENT significantly outperforms all evaluated baselines. All findings indicate that adapting the interaction representation space before selective encoding and decoding can improve higher-order tensor representation learning.

1 INTRODUCTION

Learning representations of multiway relationships is central to modeling tensor-structured data in recommendation, spatiotemporal monitoring, and cloud services (Shang et al., 2019; Luo et al., 2020; Wang et al., 2019). Tensor factorization represents entities along each mode through low-dimensional latent embeddings and combines these embeddings to model cross-mode interactions. Canonical formulations generally impose a multilinear structure on this combination, such as CP and Tucker decomposition (Kolda & Bader, 2009). From a representation learning perspective, the design involves two related choices: how to parameterize mode-specific latent spaces and how to construct an interaction representation from them. With fixed mode embeddings, the interaction structure determines what relationships can be expressed and made available to application tasks.

Neural factorization methods hence explore different ways to represent interactions among latent factors. For instance, neural collaborative filtering replaces an inner-product interaction with a learned neural function (He et al., 2017). As for higher-order tensors, approaches like convolution over factor vectors (Liu et al., 2019; 2025), self-attention over mode embeddings (Dai et al., 2024), per-component additive networks (Ahn et al., 2024), and CNN-attention hybrids (Li et al., 2025) are proposed. Neural Tucker Factorization (Tang & Luo, 2025) takes a more explicit route. It forms the outer product of mode-wise embeddings and applies a linear prediction layer to the vectorized result. This gives each multiplicative interaction a coordinate in a tensor-product feature space.

However, the interaction space grows quickly with the mode ranks: for embedding dimensions $R_1, \ldots, R_N$, it has dimension $D = \prod_{n=1}^{N} R_n$. This raises an essential question: what set of interaction directions should be retained in a compact representation for a specific task? Although the mode embeddings are learned for the task, their outer product does not separately learn a basis

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

for organizing the interaction vector before selection. This matters because the information retained by coordinate selection depends on the basis in which selection takes place. Existing approaches address parts of this problem. Orthogonal weight regularization encourages orthogonality through an auxiliary penalty (Bansal et al., 2018), while auto-encoding bottlenecks compress interaction representations before prediction (Tang et al., 2025). These approaches leave open how an explicit orthogonal coordinate transformation of the interaction vector can be coupled with task-driven coordinate selection.

Hence, we propose Orthogonal Auto-Selective Encoding Tucker Factorization, a neural tensor factorization framework that jointly learns an orthogonal interaction basis and an auto-selective latent encoding. A Householder-parameterized transformation maps the interaction vector into a learned orthogonal basis. An auto-selective encoder then produces a compact K-channel latent representation, using continuous Gumbel-Softmax masks during training and deterministic coordinate selection at inference. A decoder maps the latent representation to a D-dimensional representation for prediction. Our analysis separates the isometric transformation from the selection bottleneck and characterizes the bottleneck's effective width when channels select the same coordinate.

Our contributions are summarized as follows:

1. We introduce an OASENT framework which learns a task-adapted orthogonal basis for tensor-product interactions and selects coordinates from that basis to form a compact latent representation. The basis and selection mechanism are optimized jointly.
2. We establish the exact orthogonality and isometry of the Householder parameterization and derive the representational effect of coupling it with task-driven feature selection. Our analysis characterizes how the selection bottleneck forms a compact task representation from the learned orthogonal basis.
3. We evaluate OASENT on seven real-world tensor datasets. Comparisons with baselines establish its effectiveness on tensor completion, while component ablations show that orthogonal basis learning is most useful when coupled with selection. The flexibility of its design is verified through alternative module replacements.

2 Related Work

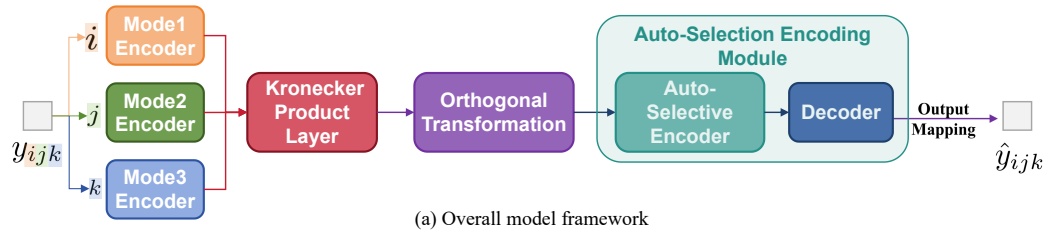
2.1 Tensor Representation Learning

Classical CP and Tucker decompositions model tensor data through multilinear interactions (Kolda & Bader, 2009). HOSVD yields a Tucker decomposition with orthonormal factor matrices and an all-orthogonal core (De Lathauwer et al., 2000), but does not learn task-specific coordinates for selecting cross-mode interactions. Neural factorization methods learn more flexible interaction functions, including convolutions over factor vectors (Liu et al., 2019; 2025), self-attention over mode embeddings (Dai et al., 2024), additive networks for individual components (Ahn et al., 2024), and combinations of convolution and attention (Li et al., 2025). Neural Tucker factorization instead retains explicit multiplicative interactions by vectorizing the outer product of mode embeddings and applies a neural network to the resulting interaction vector (Tang & Luo, 2025; Tang et al., 2025).

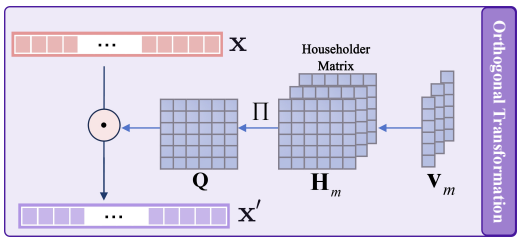
A separate line of work considers how to constrain interaction representations. Orthogonal weight regularization discourages correlations among learned parameters (Bansal et al., 2018), while information bottleneck objectives constrain the information retained in latent representations (Kawaguchi et al., 2023). ANTucF instead adopts an autoencoder structure to enforce a low-dimensional bottleneck for information compression (Tang et al., 2025). However, these methods still leave open *how to learn task-oriented coordinates for the interaction representation and select among them*.

2.2 Orthogonal Parameterization in Neural Networks

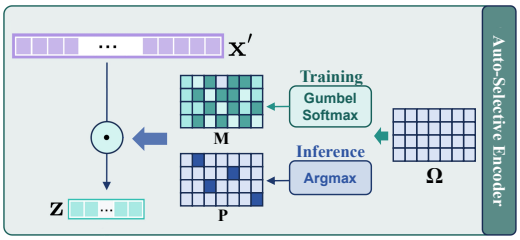
Orthogonal transformations preserve vector norms and inner products, providing a geometry-controlled operation for neural representation learning. Householder reflections (Mhammedi et al., 2017) provide exact orthogonality by algebraic construction without constrained optimization or matrix inversion. The Cayley transform (Helfrich et al., 2018) uses $Q = (I - A)(I + A)^{-1}$ with A



(a) Overall model framework



(b) Orthogonal transformation layer



(c) Auto-selective encoder layer

Figure 1: Overview of OASENT. (a) Overall architecture. (b) Householder-parameterized orthogonal transformation. (c) Auto-selective encoder with the continuous training mask $\mathbf{M}$ and deterministic inference mask $\mathbf{P}$.

skew-symmetric, requiring a matrix inversion at each forward pass. Gram-Schmidt orthogonalization (Björck, 1994) sequentially orthogonalizes vectors through projection and normalization, with known numerical instability in finite precision.

2.3 Differentiable Feature Selection

The Gumbel-Softmax (Jang et al., 2017) and Concrete distribution (Maddison et al., 2017) provide a differentiable relaxation of discrete selection via temperature-controlled softmax. Concrete autoencoders (Balın et al., 2019) apply this mechanism to global feature selection with a reconstruction objective. Indirectly parameterized variants (Nilsson et al., 2024) generate Gumbel-Softmax logits from learned embeddings through a shared mapping, improving training stability and reducing duplicate selections. SwiGLU gating (Shazeer, 2020) offers an alternative through component-wise multiplication.

3 Methodology

OASENT constructs and adapts an interaction representation for tensor completion (Figure 1(a)). For each observation, the three mode encoders retrieve mode-specific embeddings, which the Kronecker product layer combines into an interaction vector. The orthogonal transformation layer reorganizes this vector in a learned orthogonal coordinate system, as detailed in panel (b). The auto-selective encoder then constructs a compact task representation $\mathbf{z}$ with K channels through task-driven feature selection, as detailed in panel (c). The nonlinear MLP decoder maps $\mathbf{z}$ to a D-dimensional representation $\mathbf{h}$, and the output mapping produces the scalar prediction $\hat{y}_{ijk}$. The auto-selective encoder and decoder together form the selection module shown in panel (a).

3.1 Problem Formulation

Let $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be a three-way HDI tensor with observed values y_{ijk} indexed by Λ. For each $(i, j, k) \in \Lambda$, we retrieve the mode-specific embeddings $\mathbf{a} = \mathbf{e}_i^{(1)} \in \mathbb{R}^{R_1}$, $\mathbf{b} = \mathbf{e}_j^{(2)} \in \mathbb{R}^{R_2}$, and $\mathbf{c} = \mathbf{e}_k^{(3)} \in \mathbb{R}^{R_3}$. Following Neural Tucker Factorization (Tang & Luo, 2025), their Kronecker product forms the interaction representation

$$\mathbf{x} = \mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c} \in \mathbb{R}^{1 \times D}, \tag{1}$$

where $D = R_1 R_2 R_3$. The interaction vector's dimension is the product of the mode ranks, and all coordinates are generated from the same mode-specific embeddings. Their shared multiplica-

tive factors give the interaction representation its algebraically coupled structure, which is formally characterized in Appendix A.1.

For each observed tuple, the prediction task is to estimate y_{ijk} from $\mathbf{x}$. OASENT first expresses the interaction representation in a learned orthogonal coordinate system as $\mathbf{x}' = \mathbf{x}\mathbf{Q} \in \mathbb{R}^{1\times D}$. The auto-selective encoder then forms a compact task representation $\mathbf{z} \in \mathbb{R}^{1\times K}$ with $K \ll D$, and a nonlinear decoder maps $\mathbf{z}$ to $\mathbf{h} \in \mathbb{R}^{1\times D}$ for the final prediction layer. The mode encoders, orthogonal transformation, auto-selective encoder, decoder, and final prediction layer are optimized jointly using the prediction loss.

3.2 Orthogonal Transformation Layer

The orthogonal layer maps the interaction representation to $\mathbf{x}' = \mathbf{x}\mathbf{Q}$, where $\mathbf{Q} \in \mathbb{R}^{D\times D}$ satisfies $\mathbf{Q}^\top\mathbf{Q} = \mathbf{I}_D$. This transformation expresses $\mathbf{x}$ in a learned orthogonal coordinate system while retaining its full dimension. We parameterize $\mathbf{Q}$ with Householder reflections (Mhammedi et al., 2017). For a nonzero learnable vector $\mathbf{v}_m \in \mathbb{R}^D$, the m-th reflector is

$$\mathbf{H}_m = \mathbf{I}_D - 2\frac{\mathbf{v}_m\mathbf{v}_m^\top}{\|\mathbf{v}_m\|_2^2}. \tag{2}$$

The coordinate transformation is the product $\mathbf{Q} = \prod_{m=1}^{D}\mathbf{H}_m$ of D reflectors (Golub & Van Loan, 2013). Each reflector satisfies $\mathbf{H}_m^\top\mathbf{H}_m = \mathbf{I}_D$, and their product remains orthogonal. The resulting coordinate transformation maintains exact orthogonality throughout training without constrained optimization, matrix inversion, or a penalty term.

3.3 Gumbel-Softmax Auto-Selective Encoder

The auto-selective encoder maps $\mathbf{x}'$ to a compact task representation $\mathbf{z} \in \mathbb{R}^{1\times K}$ with $K \ll D$. During training, the Gumbel-Softmax relaxation provides a differentiable approximation to coordinate selection (Jang et al., 2017; Maddison et al., 2017). Following the indirect parameterization in Nilsson et al. (2024), its logits are generated as $\mathbf{\Omega} = \mathbf{\Psi}\mathbf{W}^\top + \mathbf{b}$, where $\mathbf{\Psi} \in \mathbb{R}^{K\times D}$, $\mathbf{W} \in \mathbb{R}^{D\times D}$, and $\mathbf{b} \in \mathbb{R}^D$. These parameters are independent of $\mathbf{x}'$, giving observation-independent coordinate scores. Training constructs a continuous mask $\mathbf{M} \in (0,1)^{K\times D}$ as

$$\mathbf{M}_{u,d} = \frac{\exp((\mathbf{\Omega}_{u,d} + g_{u,d})/\tau)}{\sum_{j=1}^{D}\exp((\mathbf{\Omega}_{u,j} + g_{u,j})/\tau)}, \tag{3}$$

where $g_{u,d} \sim \text{Gumbel}(0,1)$ and $\tau > 0$ controls the sharpness of each row. Let $\Lambda_{\text{tr}} \subset \Lambda$ denote the training index set. The main experiments anneal the temperature from $\tau_0 = 15$ to $\tau_{\min} = 0.1$ with the multiplicative schedule

$$\alpha = \left(\frac{\tau_{\min}}{\tau_0}\right)^{1/T_{\text{target}}}, \qquad T_{\text{target}} = E_{\text{target}} \cdot \left\lceil \frac{|\Lambda_{\text{tr}}|}{s} \right\rceil, \qquad \tau \leftarrow \max(\tau_{\min},\ \alpha\cdot\tau), \tag{4}$$

where E_{target} is the target annealing epoch, s is the batch size, and $|\Lambda_{\text{tr}}|$ is the number of training observations. The temperature is updated after each training batch and reaches $\tau_{\min}$ at approximately epoch E_{target}.

The positive minimum temperature keeps the training mask continuous, while lower temperatures make each sampled mask increasingly concentrated toward a one-hot selection (Jang et al., 2017; Maddison et al., 2017; Nilsson et al., 2024). This annealing reduces the difference in representation form between differentiable training and the deterministic hard selection used at inference. Specifically, deterministic row-wise argmax gives a one-hot selection matrix with entries $\mathbf{P}_{u,d} = \mathbb{I}\big[d = \arg\max_j \mathbf{\Omega}_{u,j}\big]$. Each of the K rows defines a separate Gumbel-Softmax selector. Because no exclusivity constraint is imposed across rows, multiple rows may select the same coordinate. Training and inference use

$$\mathbf{z} = \begin{cases} \mathbf{x}'\mathbf{M}^\top, & \text{during training,} \\ \mathbf{x}'\mathbf{P}^\top = \mathbf{x}\mathbf{Q}\mathbf{P}^\top, & \text{at inference.} \end{cases} \tag{5}$$

The continuous training path passes prediction gradients to the selection parameters. At inference, defining $\mathbf{B} = \mathbf{Q}\mathbf{P}^\top \in \mathbb{R}^{D\times K}$ gives $\mathbf{z} = \mathbf{x}\mathbf{B}$.

3.4 Decoding and Prediction

The compact task representation $\mathbf{z}$ is fed into a nonlinear decoder implemented as an MLP with ReLU activations, giving $\mathbf{h} = \mathcal{N}(\mathbf{z}) \in \mathbb{R}^{1\times D}$. The final prediction layer gives $\hat{y}_{ijk} = \sigma(\mathbf{w}_{\text{out}}^{\top}\mathbf{h}^{\top})$, where $\mathbf{w}_{\text{out}} \in \mathbb{R}^{D}$ and $\sigma(\cdot)$ is the sigmoid function. The framework is trained end-to-end by minimizing the mean squared error over the training observations:

$$\mathcal{L} = \frac{1}{|\Lambda_{\text{tr}}|} \sum_{(i,j,k)\in\Lambda_{\text{tr}}} (y_{ijk} - \hat{y}_{ijk})^2. \tag{6}$$

The computational complexity and complete training–inference procedure are provided in Appendix B.2.

3.5 Geometry of the Learned Representation

We characterize the representation map implemented by the orthogonal transformation and selection operation. Proofs of Propositions 1 and 2, together with supporting results, are provided in Appendix A.2.

Isometric Coordinate Representation

Proposition 1 (Isometric Coordinate Transformation)**.** *For every $\mathbf{Q} \in \mathbb{O}(D)$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{1\times D}$,*

$$\|\mathbf{x}_1\mathbf{Q} - \mathbf{x}_2\mathbf{Q}\|_2 = \|\mathbf{x}_1 - \mathbf{x}_2\|_2, \qquad \langle \mathbf{x}_1\mathbf{Q}, \mathbf{x}_2\mathbf{Q}\rangle = \langle \mathbf{x}_1, \mathbf{x}_2\rangle. \tag{7}$$

The orthogonal layer preserves the geometry of the interaction representation. It learns a coordinate system for the subsequent selection stage without compressing the representation.

Geometry of the Selected Representation

Proposition 2 (Rank of Hard Selection)**.** *Let $\mathbf{B} = \mathbf{Q}\mathbf{P}^{\top}$, where each row of $\mathbf{P}$ is one-hot, and let n_d denote the number of rows that select coordinate d. Then $\mathbf{B}^{\top}\mathbf{B} = \mathbf{P}\mathbf{P}^{\top}$, and*

$$J_{\text{eff}} = \operatorname{rank}(\mathbf{B}) = \operatorname{rank}(\mathbf{P}) = |\{d : n_d > 0\}|. \tag{8}$$

Consequently, $J_{\text{eff}} \le K$. Distinct selected indices give $J_{\text{eff}} = K$ and $\mathbf{B}^{\top}\mathbf{B} = \mathbf{I}_K$, while repeated selections give $J_{\text{eff}} < K$.

The selection budget K specifies the number of selection channels. At inference, J_{eff} is the effective width of the decoder input $\mathbf{z}$, defined as the number of unique orthogonal directions represented by those channels. The selection parameters are optimized jointly with the prediction objective, making both the selected coordinates and the effective width learned properties of the tensor completion model.

A full-dimensional orthogonal transformation alone can be absorbed into an unconstrained first decoder layer and therefore does not change the represented decoder function class. The subsequent selection operation changes this situation: by exposing only selected directions of the learned coordinate system to the decoder, it turns coordinate learning into a task-driven representation bottleneck. At inference, the selected representation $\mathbf{z}$ has an effective width $J_{\text{eff}} \le K$. During training, the continuous mask provides a differentiable approximation to this bottleneck, with annealing making its rows increasingly concentrated toward the one-hot form used at inference. Lemma 1 further shows that, at hard-selection inference, repeated channels can be merged by the decoder's first linear layer without changing its output.

3.6 Configurability

The OASENT framework admits alternative instantiations of the orthogonal layer and selection module. The orthogonal layer can be parameterized via the Cayley transform (Helfrich et al., 2018) or Gram-Schmidt orthogonalization (Björck, 1994). Within the selection module, Gumbel-Softmax selection can be replaced by SwiGLU gating (Shazeer, 2020), a nonlinear MLP encoder, or a joint selection-plus-gating scheme. These variants retain an associated decoder and differ in how they construct the K-channel task representation. We compare these instantiations in Section 4.6.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate OASENT on seven real-world HDI tensor datasets spanning four application domains. D1 (HZMetro) captures metro passenger flow (Chen et al., 2020), and D2 (NYCTaxi) records taxi trips (Chen & Sun, 2022). D3 (PCTemp) contains sea surface temperature observations (Hølleland & Karlsen, 2020), while D4 (NAPrec) contains gridded precipitation measurements (Thornton et al., 2021). D5 (QoS-RT) and D6 (QoS-TP) measure Web-service response time and throughput (Zhang et al., 2011), and D7 (Alibaba) contains time-indexed response-time observations between upstream and downstream microservices (Luo et al., 2022). For each dataset, we partition the observed index set Λ into disjoint training, validation, and test subsets Λ_{tr}, Λ_{va}, and Λ_{te}, respectively. We apply a $\log(1+y)$ transformation to the observed values, followed by min-max normalization using the minimum and maximum of the transformed training values for all three subsets. Table 1 summarizes the tensor sizes, entries, densities, and data splits. Further dataset details are provided in Appendix B.1.

Table 1: Summary of the seven HDI tensor datasets. Entries and density refer to Λ_{tr}. Split ratio lists the proportions assigned to Λ_{tr}, Λ_{va}, and Λ_{te}, respectively.

No.	Dataset	Tensor Size	Split Ratio	Entries	Density (%)
D1	HZMetro	$80 \times 25 \times 108$	10:20:70	20,976	9.7111
D2	NYCTaxi	$30 \times 30 \times 1464$	10:20:70	97,445	7.3956
D3	PCTemp	$30 \times 84 \times 399$	10:20:70	100,548	10.0000
D4	NAPrec	$100 \times 340 \times 372$	10:20:70	132,772	1.0497
D5	QoS-RT	$142 \times 4500 \times 64$	5:15:80	1,358,905	3.3228
D6	QoS-TP	$142 \times 4500 \times 64$	5:15:80	1,282,616	3.1363
D7	Alibaba	$18885 \times 27922 \times 120$	70:20:10	226,765	3.5837×10^{-4}

Baselines. We compare OASENT against seven models spanning the development from classical tensor factorization to neural tensor completion. CPD (Kolda & Bader, 2009) is the canonical polyadic decomposition with L2 regularization. CoSTCo (Liu et al., 2019) applies convolutional layers to factor vectors for feature extraction, aggregated via an MLP. HOCTC (Dai et al., 2024) employs self-attention with query expansion and multi-level contrastive learning. CoATF (Li et al., 2025) combines a two-layer CNN with attention-based feature weighting. NEAT (Ahn et al., 2024) learns independent neural networks per latent component. ConvNTC (Liu et al., 2025) uses a convolutional encoder with multilinear constraints and a KAN-based predictor. ANTucF (Tang et al., 2025) applies an auto-encoding bottleneck to the NeuTucF interaction vector. We tune each baseline's hyperparameters for our experiments.

Training and Evaluation. All models use embedding rank $R = 5$. For OASENT, the interaction dimension is $D = 125$. We use Householder reflections and Gumbel-Softmax selection with a selection-channel budget of $K = 15$. Training uses the Adam optimizer with learning rate 10^{-3}, batch size 1024, and a maximum of 300 epochs. OASENT's temperature is annealed from $\tau_0 = 15$ to $\tau_{\min} = 0.1$ via the multiplicative schedule in Eq. 4. Sensitivity analyses of K and the target annealing epoch E_{target} are provided in Appendix B.3. Every model is evaluated over 10 independent runs with random initialization. We report mean absolute error (MAE), mean relative error (MRE), root mean squared error (RMSE), best-result counts, and average ranks across the 21 dataset-metric combinations.

4.2 MAIN RESULTS

Table 2 reports performance across seven datasets under three metrics. OASENT attains the lowest mean error in all 21 dataset–metric combinations and an average rank of 1.00. Across datasets, two-sided paired Wilcoxon signed-rank tests against every baseline yield $p = 0.0156 < 0.05$ on each metric.

The largest relative gains occur on D3 (PCTemp), where OASENT reduces MAE, MRE, and RMSE by 19.1%, 19.6%, and 22.8%, respectively, relative to ANTucF, the strongest baseline. On D6 (QoS-TP), the corresponding reductions are 7.6%, 1.9%, and 9.1%, illustrating that the improvement can vary across metrics within the same dataset.

Table 2: Performance evaluation on seven HDI tensor datasets. Best results are **bold**, second-best are underlined. All values are averaged over 10 runs. Sample standard deviations appear in Appendix D.

		CPD	CoSTCo	HOCTC	CoATF	NEAT	ConvNTC	ANTucF	OASENT
D1	MAE	52.7699	29.9612	34.2456	25.1594	28.1055	24.9536	26.3048	**23.3296**
	MRE	0.6140	1.3110	0.2850	0.2372	0.2456	0.2308	0.2529	**0.2123**
	RMSE	116.3830	58.9369	71.5450	53.1192	61.0536	57.4976	53.7448	**48.3073**
D2	MAE	8.5408	8.4526	4.4776	3.4214	4.1451	3.4495	3.5285	**3.3736**
	MRE	2.6037	2.3833	0.6551	0.4911	0.5795	0.5057	0.4931	**0.4511**
	RMSE	12.0059	12.2616	7.7251	5.6893	7.1572	5.8060	5.9277	**5.6824**
D3	MAE	2.4938	0.6554	0.7582	0.6402	0.9918	0.7222	0.5083	**0.4110**
	MRE	0.1000	0.0272	0.0309	0.0257	0.0402	0.0291	0.0204	**0.0164**
	RMSE	2.9231	0.9079	1.0419	0.8446	1.3127	0.9506	0.6861	**0.5300**
D4	MAE	9.9003	4.7653	5.0317	3.9584	3.9619	3.9461	3.2587	**3.1816**
	MRE	1.2792	1.2714	1.1339	0.9302	0.8636	0.9110	0.5549	**0.5116**
	RMSE	18.6089	9.5898	10.1165	7.5577	7.8303	8.2672	6.4257	**6.1434**
D5	MAE	1.2040	0.7317	0.7750	0.7121	0.6729	0.7172	0.6390	**0.6168**
	MRE	1.9760	1.7567	1.7813	1.6707	1.2901	1.4240	1.2361	**1.0572**
	RMSE	2.9117	1.9418	1.9684	1.7639	1.8256	1.9126	1.7782	**1.7380**
D6	MAE	10.9362	6.3075	6.5655	5.2686	4.5173	6.4138	4.3773	**4.0462**
	MRE	1.3815	1.5937	1.0777	1.0091	0.9317	1.2019	0.8261	**0.8103**
	RMSE	55.2629	42.5711	41.7357	36.4626	34.1956	42.9911	32.3394	**29.3827**
D7	MAE	0.3051	0.1211	0.1054	0.1011	0.0916	0.1176	0.0949	**0.0862**
	MRE	1.1308	0.3960	0.2931	0.2784	0.2580	0.3181	0.2673	**0.2376**
	RMSE	0.4149	0.1769	0.1610	0.1515	0.1480	0.1728	0.1515	**0.1445**
Best/Total		0/21	0/21	0/21	0/21	0/21	0/21	0/21	**21/21**
Avg. Rank		7.86	6.19	6.14	3.31	4.19	4.57	2.74	**1.00**

D2 (NYCTaxi) provides the closest comparison in MAE and RMSE. OASENT lowers these two errors by 1.4% and 0.12% relative to CoATF, while reducing MRE by 8.1%. On D7 (Alibaba), whose training density is $3.5837 \times 10^{-4}\%$, OASENT reduces MAE, MRE, and RMSE by 5.9%, 7.9%, and 2.4% relative to NEAT. The improvement margins vary across datasets and metrics, ranging from the narrow comparison on D2 to the larger reductions observed on D3.

4.3 Ablation Study

To determine how the orthogonal transformation layer and selection module affect prediction separately and together, we conduct an ablation study on datasets from three application domains: urban mobility (D2, NYCTaxi), environmental monitoring (D3, PCTemp), and web-service performance (D5, QoS-RT). All variants retain the same mode-encoder architecture and Kronecker product layer. Taking OASENT (ν_4) as the reference, we implement three ablated variants: (a) ν_3 without the orthogonal transformation layer but with the selection module, (b) ν_2 without the selection module but with the orthogonal transformation layer, and (c) ν_1 without either component. Figure 2 reports mean RMSE for OASENT and the three ablated variants.

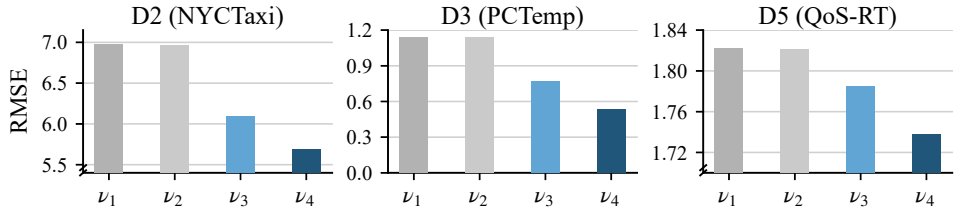


Figure 2: RMSE component ablation across three application domains.

OASENT (ν_4) achieves the lowest mean RMSE on all three datasets. Relative to ν_1, which lacks both components, it reduces RMSE by 18.5% on D2, 53.5% on D3, and 4.6% on D5. The two single ablations separate the contributions of the modules. Without the selection module, ν_2 remains within 0.2% of ν_1 in RMSE on each dataset. Without the orthogonal transformation layer,

ν_3 still outperforms ν_1, while OASENT reduces RMSE relative to ν_3 by 6.8%, 31.3%, and 2.7%, respectively.

These results are consistent with the representation analysis in Section 3.5. The near-identical RMSE of ν_2 and ν_1 is in line with the full-dimensional orthogonal layer leaving the downstream function class unchanged without selection. The improvement of ν_3 over ν_1 shows that the selection module is useful even without the orthogonal layer. The further reduction from ν_3 to ν_4 supports the complementary contribution of orthogonal coordinate learning when combined with selection.

4.4 Correlation Structure of Representations

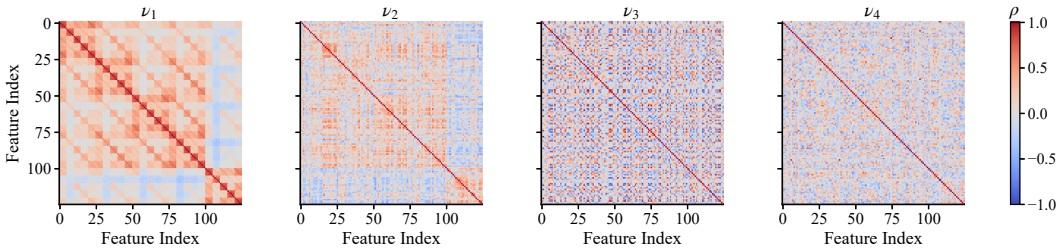


Figure 3: Pearson correlation heatmaps of the representations before prediction for the four component variants on D5.

Figure 3 shows the heatmaps of the Pearson correlation matrices for the D-dimensional representations immediately before prediction on D5: the interaction vector $\mathbf{x}$ in ν_1, the orthogonally transformed vector $\mathbf{x}'$ in ν_2, and the decoder output $\mathbf{h}$ in ν_3 and ν_4. All heatmaps use the same color scale, data split, and random seed.

In ν_1, the interaction representation shows blocks of correlated coordinates, consistent with the multiplicative construction of the Tucker interaction. The orthogonal layer in ν_2 redistributes these coordinate-wise correlation patterns and attenuates the block structure. Although it visibly changes the correlation structure, it produces only a small RMSE change on its own. With selection and decoding, ν_3 shows a much more dispersed pattern than ν_1. The off-diagonal correlations in ν_4 appear weaker and less structured than in ν_3, alongside the lower RMSE observed in the ablation.

4.5 The Selection Behavior

We next investigate how the trained auto-selective encoder uses its K output channels. Because multiple channels may select the same coordinate, the effective width J_{eff} is defined by the number of distinct selected coordinates and can be smaller than K. With $K = 15$, we report the mean and sample standard deviation of J_{eff} over 10 independent runs and compare default OASENT with a matched forced-distinct variant. Unlike the default independent selectors, this control constructs continuous masks sequentially during training and selects coordinates without replacement at inference. By Proposition 2, its inference map has $\text{rank}(\mathbf{B}) = K = 15$, whereas default OASENT permits $\text{rank}(\mathbf{B}) = J_{\text{eff}} \leq K$. Figure 4 presents both aspects of the selection behavior.

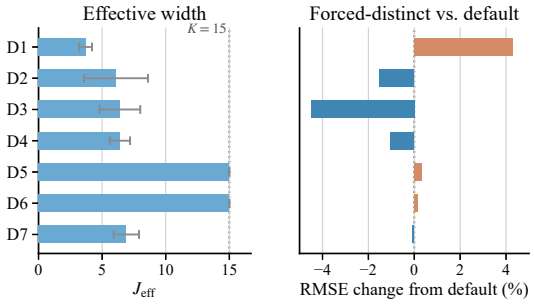


Figure 4: Effective width (left) and RMSE change under forced-distinct selection relative to default OASENT (right).

The left panel shows that the effective widths vary substantially across datasets. D1 has the narrowest representation with $J_{\text{eff}} = 3.7 \pm 0.5$. D2, D3, D4, and D7 retain 6.1–6.9 distinct coordinates on average, while D5 and D6 use all 15 coordinates in every run. Repeated selections therefore reduce the effective width below K on five datasets, whereas the other two use the full selection-channel

budget. This variation suggests that the encoder adapts the number of distinct selected coordinates to each dataset, with K setting an upper bound rather than a fixed width.

The right panel examines whether forced-distinct selection consistently improves prediction. Under matched settings, it reduces mean RMSE on D2–D4 and D7 but increases it on D1, D5, and D6. The changes on D5–D7 are small, while D1 and D3 show the largest changes in opposite directions. The forced-distinct mechanism therefore does not consistently lower mean RMSE. Appendix B.4 gives the full control procedure and reports results under all three error metrics.

4.6 Configurability of Orthogonal Transformation and Selection

We test alternative implementations of the orthogonal transformation and selection module while holding the rest of OASENT fixed. For the orthogonal transformation, we compare Householder reflections with Gram-Schmidt orthogonalization and the Cayley transform while using the default selection module. For the selection module, we fix Householder reflections and compare Gumbel-Softmax selection, SwiGLU gating, their joint scheme, and an auto-encoding bottleneck. Figure 5 reports MAE for both comparisons. Appendix Tables 5 and 6 provide the complete results under all three metrics.

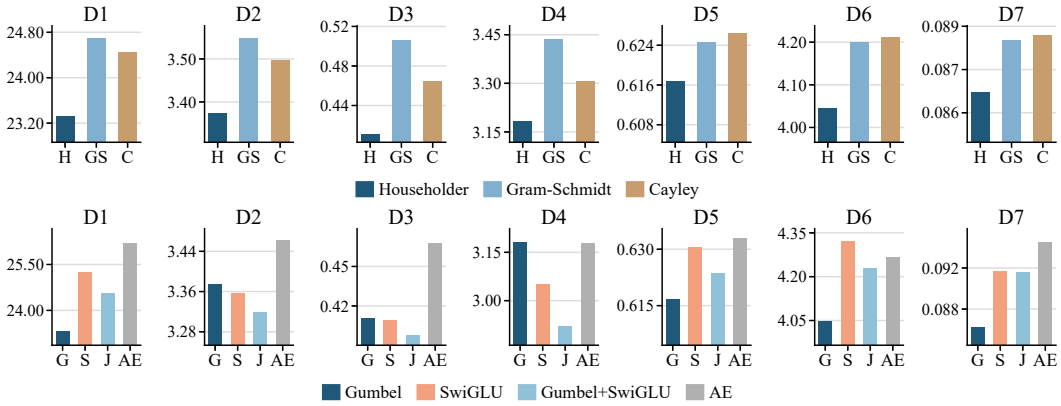


Figure 5: MAE for alternative orthogonal parameterizations (upper) and selection modules (lower), with the other stage fixed to the default Gumbel-Softmax or Householder implementation, respectively. Vertical scales are dataset-specific.

Among the orthogonal parameterizations, Householder achieves the lowest mean MAE on all seven datasets. Among the selection modules, Gumbel+SwiGLU yields the lowest mean MAE on D2, D3, and D4, while the default Gumbel-Softmax selection module is best on D1, D5, D6, and D7. Together, these results support Householder reflections as the default orthogonal parameterization and show that the preferred selection module varies across datasets. Gumbel-Softmax nevertheless remains the default, achieving the lowest mean MAE on four of seven datasets.

5 Conclusion

In this work, we introduced OASENT, a neural tensor factorization framework for learning compact task representations from high-order multiplicative interactions. By placing task-driven feature selection after an isometric orthogonal coordinate transformation, our method provides a structured way to compress interaction representations while preserving their geometry during coordinate learning. Across seven real-world datasets, OASENT significantly outperforms all evaluated baselines. Our component ablations clarify the complementary roles of the two stages, while the selection-behavior experiments reveal how a fixed selection-channel budget yields different effective widths across datasets. We further instantiate the framework with alternative orthogonal parameterizations and selection mechanisms, providing a configurable design for compact representation learning in high-order interaction models.

AI USE STATEMENT

In this work, we only used ChatGPT to assist with translation and assist in the writing of proof. We have not used any generative AI tools for all other tasks with or without required disclosure. We carefully reviewed all AI-assited work. Especially, mathematical proofs were verified for correctness by two authors. We take full responsibility for the final content of this work, including all text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The model formulation, training objective, and training–inference behavior of the auto-selective encoder are specified in Section 3. Section 4.1 reports the datasets, data splits, preprocessing, hyperparameters, and repeated-run evaluation protocol. Appendix B provides the algorithmic complexity and hyperparameter sensitivity results, while Appendix A contains complete proofs of the theoretical results. The forced-distinct control and the alternative module instantiations are documented in Appendix B.4 and Appendix C, respectively.

REFERENCES

Dawon Ahn, Uday Singh Saini, Evangelos E Papalexakis, and Ali Payani. Neural additive tensor decomposition for sparse tensors. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 14–23, 2024.

Muhammed Fatih Balın, Abubakar Abid, and James Zou. Concrete autoencoders: Differentiable feature selection and reconstruction. In *Proc. 36th Int. Conf. Mach. Learn.*, volume 97, pp. 444–453, 2019.

Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? *Advances in Neural Information Processing Systems*, 31, 2018.

Åke Björck. Numerics of Gram-Schmidt orthogonalization. *Linear Algebra and Its Applications*, 197–198:297–316, 1994.

Xinyu Chen and Lijun Sun. Bayesian temporal factorization for multidimensional time series prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4659–4673, 2022.

Xinyu Chen, Jinming Yang, and Lijun Sun. A nonconvex low-rank tensor completion model for spatiotemporal traffic data imputation. *Transportation Research Part C: Emerging Technologies*, 117:102673, 2020.

Yu Dai, Junchen Shen, Zijie Zhai, Danlin Liu, Jingyang Chen, Yu Sun, Ping Li, Jie Zhang, and Kai Zhang. High-order contrastive learning with fine-grained comparative levels for sparse ordinal tensor completion. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 9856–9871, 2024.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4th edition, 2013.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pp. 173–182, 2017.

Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled Cayley transform. In *International Conference on Machine Learning*, pp. 1969–1978. PMLR, 2018.

Sondre Hølleland and Hans Arnfinn Karlsen. A stationary spatio-temporal GARCH model. *Journal of Time Series Analysis*, 41(2):177–209, 2020.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017.

Kenji Kawaguchi, Zhun Deng, Xu Ji, and Jiaoyang Huang. How does information bottleneck help deep learning? In *International conference on machine learning*, pp. 16049–16096. PMLR, 2023.

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

Hao Li, Jianli Zhao, Qingqian Guan, Lutong Yao, Jianjian Chen, and Guojun Sheng. CoATF: Convolution and attention based tensor factorization model for context-aware recommendation. *IEEE Transactions on Network Science and Engineering*, 12(5):3682–3693, 2025.

Hanpeng Liu, Yaguang Li, Michael Tsang, and Yan Liu. CoSTCo: A neural tensor completion model for sparse tensors. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 324–334, 2019.

Pei Liu, Xiao Liang, Yue Li, and Jiawei Luo. ConvNTC: convolutional neural tensor completion for detecting "A–A–B" type biological triplets. *Briefings in Bioinformatics*, 26(4):bbaf372, 2025.

Shutian Luo, Huanle Xu, Kejiang Ye, Guoyao Xu, Liping Zhang, Guodong Yang, and Chengzhong Xu. The power of prediction: Microservice auto scaling via workload learning. In *Proceedings of the 13th Symposium on Cloud Computing*, pp. 355–369. Association for Computing Machinery, 2022.

Xin Luo, Hao Wu, Huaqiang Yuan, and MengChu Zhou. Temporal pattern-aware QoS prediction via biased non-negative latent factorization of tensors. *IEEE Trans. Cybern.*, 50(5):1798–1809, 2020.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Zakaria Mhammedi, Andrew Hellicar, Ashfaqur Rahman, and James Bailey. Efficient orthogonal parametrisation of recurrent neural networks using Householder reflections. In *International Conference on Machine Learning*, pp. 2401–2409. PMLR, 2017.

Alfred Nilsson, Klas Wijk, Sai Bharath Chandra Gutha, Erik Englesson, Alexandra Hotti, Carlo Saccardi, Oskar Kviman, Jens Lagergren, Ricardo Vinuesa Motilva, and Hossein Azizpour. Indirectly parameterized concrete autoencoders. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 38237–38252, 2024.

Mingsheng Shang, Xin Luo, Zhigang Liu, Jia Chen, Ye Yuan, and MengChu Zhou. Randomized latent factor model for high-dimensional and sparse matrices from industrial applications. *IEEE/CAA J. Autom. Sinica*, 6(1):131–141, 2019.

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Peng Tang and Xin Luo. Neural Tucker factorization. *IEEE/CAA Journal of Automatica Sinica*, 12 (2):475–477, 2025.

Peng Tang, Xin Luo, and Jim Woodcock. Auto-encoding neural Tucker factorization. *IEEE Transactions on Knowledge and Data Engineering*, 37(10):5795–5807, 2025.

Peter E. Thornton, Rupesh Shrestha, Michele Thornton, Shih-Chieh Kao, Yaxing Wei, and Bruce E. Wilson. Gridded daily weather data for north america with comprehensive uncertainty quantification. *Scientific Data*, 8:190, 2021.

Shangguang Wang, You Ma, Bo Cheng, Fangchun Yang, and Rong N Chang. Multi-dimensional QoS prediction for service recommendations. *IEEE Trans. Serv. Comput.*, 12(01):47–57, 2019.

Yilei Zhang, Zibin Zheng, and Michael R. Lyu. WSPred: A time-aware personalized QoS prediction framework for web services. In *2011 IEEE 22nd International Symposium on Software Reliability Engineering*, pp. 210–219, 2011.

A PROOFS

A.1 KRONECKER STRUCTURE OF THE INTERACTION VECTOR

Proposition 3. *Let $\mathbf{x} = \mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c} \in \mathbb{R}^D$, where $\mathbf{a} \in \mathbb{R}^{R_1}$, $\mathbf{b} \in \mathbb{R}^{R_2}$, and $\mathbf{c} \in \mathbb{R}^{R_3}$ are nonzero, and $D = R_1 R_2 R_3$. Then $\mathbf{x}$ is the vectorized form of a rank-one tensor, and its coordinates satisfy the corresponding rank-one polynomial constraints. The family of nonzero interaction vectors has dimension $R_1 + R_2 + R_3 - 2$ in $\mathbb{R}^D$.*

Proof. For nonzero $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$, the Kronecker product $\mathbf{x} = \mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c} \in \mathbb{R}^D$ is the vectorized form of the rank-one tensor $\boldsymbol{\mathcal{T}} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$. The mode-1 unfolding is $\mathbf{T}_{(1)} \in \mathbb{R}^{R_1 \times R_2 R_3}$, defined entrywise by $(\mathbf{T}_{(1)})_{p,(q,r)} = a_p b_q c_r$. In matrix form,

$$\mathbf{T}_{(1)} = \mathbf{a}(\mathbf{b} \otimes \mathbf{c})^\top. \tag{9}$$

Both vectors in this outer product are nonzero, so $\operatorname{rank}(\mathbf{T}_{(1)}) = 1$. All 2×2 minors of $\mathbf{T}_{(1)}$ vanish:

$$a_p b_q c_r \cdot a_{p'} b_{q'} c_{r'} - a_{p'} b_q c_r \cdot a_p b_{q'} c_{r'} = 0, \tag{10}$$

with analogous relations for the mode-2 and mode-3 unfoldings. These polynomial identities express the algebraic coupling among the interaction coordinates. The parameterization contains $R_1 + R_2 + R_3$ scalars. The column spaces of the three unfoldings determine the directions of $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$. Any other nonzero factorization of the same $\mathbf{x}$ can be written as $\lambda\mathbf{a}$, $\mu\mathbf{b}$, and $(\lambda\mu)^{-1}\mathbf{c}$ for nonzero λ and μ. These two independent rescalings leave $\mathbf{x}$ unchanged, giving the nonzero interaction family dimension $R_1 + R_2 + R_3 - 2$. If any mode embedding is zero, then $\mathbf{x} = \mathbf{0}$ and the polynomial identities hold trivially. Each component $x_{pqr} = a_p b_q c_r$ is the product of three scalars from a shared pool of $R_1 + R_2 + R_3$ parameters. □

A.2 REPRESENTATION GEOMETRY

Proof of Proposition 1. For each applied reflector, let $\mathbf{u}_m = \mathbf{v}_m / \|\mathbf{v}_m\|_2$. Then

$$\mathbf{H}_m^\top \mathbf{H}_m = \left(\mathbf{I}_D - 2\mathbf{u}_m \mathbf{u}_m^\top\right)^2 = \mathbf{I}_D. \tag{11}$$

If $\mathbf{v}_m^\top \mathbf{v}_m < 10^{-12}$, the numerical safeguard replaces the reflector by $\mathbf{I}_D$, which is also orthogonal. The product $\mathbf{Q}$ satisfies $\mathbf{Q}^\top \mathbf{Q} = \mathbf{Q}\mathbf{Q}^\top = \mathbf{I}_D$. For any $\mathbf{x}_1$ and $\mathbf{x}_2$,

$$\begin{aligned} \|\left(\mathbf{x}_1 - \mathbf{x}_2\right)\mathbf{Q}\|_2^2 &= \left(\mathbf{x}_1 - \mathbf{x}_2\right)\mathbf{Q}\mathbf{Q}^\top\left(\mathbf{x}_1 - \mathbf{x}_2\right)^\top \\ &= \|\mathbf{x}_1 - \mathbf{x}_2\|_2^2, \end{aligned} \tag{12}$$

and

$$\left(\mathbf{x}_1\mathbf{Q}\right)\left(\mathbf{x}_2\mathbf{Q}\right)^\top = \mathbf{x}_1\mathbf{Q}\mathbf{Q}^\top\mathbf{x}_2^\top = \mathbf{x}_1\mathbf{x}_2^\top. \tag{13}$$

□

Proof of Proposition 2. Using $\mathbf{Q}^\top\mathbf{Q} = \mathbf{I}_D$,

$$\mathbf{B}^\top\mathbf{B} = \mathbf{P}\mathbf{Q}^\top\mathbf{Q}\mathbf{P}^\top = \mathbf{P}\mathbf{P}^\top. \tag{14}$$

Because $\mathbf{Q}$ is invertible, $\operatorname{rank}(\mathbf{B}) = \operatorname{rank}(\mathbf{P})$. For a one-hot matrix $\mathbf{P}$,

$$\mathbf{P}^\top\mathbf{P} = \operatorname{diag}(n_1, \ldots, n_D), \tag{15}$$

whose rank equals the number of positive n_d. Hence

$$J_{\text{eff}} = \operatorname{rank}(\mathbf{B}) = \operatorname{rank}(\mathbf{P}) = \left|\{d : n_d > 0\}\right|. \tag{16}$$

The number of positive counts is at most K. When all selected indices are distinct, the rows of $\mathbf{P}$ are orthonormal, so $\mathbf{P}\mathbf{P}^\top = \mathbf{I}_K$ and $J_{\text{eff}} = K$. A repeated index reduces the number of positive counts below K. □

Lemma 1 (Repeated-Channel Reduction)**.** *At hard-selection inference, partition the K output channels into J_{eff} groups $G_1, \ldots, G_{J_{\text{eff}}}$ according to their selected coordinates. If the decoder begins with an unconstrained linear layer, the repeated channels can be collapsed into J_{eff} distinct inputs without changing the decoder output.*

Proof. Let $\widetilde{z}_j$ denote the common value of all channels in G_j, collect these values in $\widetilde{\mathbf{z}}$, and let $\mathbf{C}_{k,:}$ be the weights applied to channel k by the first decoder layer. Then

$$\mathbf{z}\mathbf{C} = \sum_{k=1}^{K} z_k \mathbf{C}_{k,:} = \sum_{j=1}^{J_{\text{eff}}} \widetilde{z}_j \left(\sum_{k \in G_j} \mathbf{C}_{k,:} \right). \tag{17}$$

Defining $\widetilde{\mathbf{C}}_{j,:} = \sum_{k \in G_j} \mathbf{C}_{k,:}$ gives $\mathbf{z}\mathbf{C} = \widetilde{\mathbf{z}}\widetilde{\mathbf{C}}$. The first-layer response, and hence every subsequent decoder output, is unchanged. Therefore, repeated hard-selection channels contribute no additional distinct decoder inputs. □

Representational consequence. For an MLP decoder with an unconstrained first linear layer $\mathbf{C}$, the full-dimensional orthogonal transformation can be absorbed because $(\mathbf{x}\mathbf{Q})\mathbf{C} = \mathbf{x}(\mathbf{Q}\mathbf{C})$. Since $\mathbf{Q}$ is invertible, the map $\mathbf{C} \mapsto \mathbf{Q}\mathbf{C}$ is bijective, so this coordinate transformation does not change the represented decoder function class. Selection changes the dependency on $\mathbf{x}$: for each continuous training mask $\mathbf{M}$, the map $\mathbf{Q}\mathbf{M}^\top$ has rank at most K, while Proposition 2 gives $\operatorname{rank}(\mathbf{B}) = J_{\text{eff}} \leq K$ at inference. Thus, the orthogonal layer changes coordinates without reducing their dimension, whereas the selection operation restricts the representation supplied to all subsequent decoder layers.

B Additional Method and Experimental Details

B.1 Dataset Details

D1 (HZMetro) records passenger mobility across 80 stations of the Hangzhou metro system over 25 days in January 2019. D2 (NYCTaxi) captures taxi trip records in New York City with pick-up/drop-off locations and trip start time. D3 (PCTemp) provides monthly sea surface temperature observations on the Pacific Ocean over 399 consecutive months from January 1970 to March 2003, partitioned on a $2^\circ \times 2^\circ$ grid. D4 (NAPrec) contains monthly precipitation records over North America from 1989 to 2019, covering 372 consecutive months on a $10^\circ \times 10^\circ$ grid. D5 (QoS-RT) and D6 (QoS-TP) are derived from the WSDream dataset, capturing interactions between PlanetLab users and Web services across 15-minute time slices. D5 records response time in seconds, while D6 records throughput in kbps. D7 (Alibaba) is derived from the Alibaba Microservices Trace v2022. Its three modes index 18,885 upstream microservices, 27,922 downstream microservices, and 120 time intervals retained after preprocessing. Each observed entry records the processed response time, in milliseconds, of a call from an upstream microservice to a downstream microservice within the corresponding interval.

B.2 Complexity and Algorithm

For the equal-rank setting $R_1 = R_2 = R_3 = R$ used in our experiments, each interaction vector has dimension R^3. The orthogonal transformation is applied without materializing the dense matrix $\mathbf{Q}$. Given a mini-batch $\mathbf{X}_0 = \mathbf{X} \in \mathbb{R}^{s \times R^3}$, the m-th Householder reflector gives

$$\mathbf{X}_m = \mathbf{X}_{m-1} - 2\frac{\mathbf{X}_{m-1}\mathbf{v}_m}{\mathbf{v}_m^\top \mathbf{v}_m}\mathbf{v}_m^\top, \qquad m = 1, \ldots, R^3. \tag{18}$$

When $\mathbf{v}_m^\top \mathbf{v}_m < 10^{-12}$, the corresponding step is treated as the identity transformation.

For a mini-batch of size s, applying all R^3 reflectors costs $O(sR^6)$, while computing the selection logits costs $O(KR^6)$. Since $K \ll R^3$, the remaining interaction, mask, and output operations are lower-order R^3 terms. Let $C_{\mathcal{N}}$ denote the per-sample decoder cost. The resulting training cost per mini-batch is

$$O\big((s+K)R^6 + sC_{\mathcal{N}}\big). \tag{19}$$

Backpropagation has the same asymptotic order. For $N = |\Lambda_{\text{tr}}|$ entries and E epochs, the total cost is $O\big(E\lceil N/s \rceil((s+K)R^6 + sC_{\mathcal{N}})\big)$. Constructing the deterministic mask after training costs $O(KR^6)$ once, followed by $O(sR^6 + sC_{\mathcal{N}})$ per inference mini-batch. Algorithms 1 and 2 summarize the training and inference paths, respectively.

Algorithm 1: OASENT Training.

Operation	Cost
Input: $\Lambda_{\text{tr}}, E, s, \tau_0, \tau_{\min}, \alpha$	
Optimization	
Initialize Θ and set $\tau \leftarrow \tau_0$	
for $e = 1, \ldots, E$ **do**	
for each $\Lambda_s \subset \Lambda_{\text{tr}}$ **do**	
Form interaction batch $\mathbf{X}$	
$\mathbf{X}' \leftarrow \text{Householder}(\mathbf{X})$	$O(sR^6)$
$\mathbf{\Omega} \leftarrow \mathbf{\Psi}\mathbf{W}^\top + \mathbf{b}$	$O(KR^6)$
Construct $\mathbf{M}$ using Eq. 3	$O(KR^3)$
$\mathbf{Z} \leftarrow \mathbf{X}'\mathbf{M}^\top$	$O(sKR^3)$
$\mathbf{H} \leftarrow \mathcal{N}(\mathbf{Z})$	$O(sC_{\mathcal{N}})$
Predict $\widehat{\mathbf{y}}$ and evaluate $\mathcal{L}$	
Update Θ by backpropagation	
$\tau \leftarrow \max(\tau_{\min}, \alpha\tau)$	
end for	
end for	

Algorithm 2: OASENT Inference.

Operation	Cost
Input: $\Theta^*, \Lambda_{\text{eval}} \in \{\Lambda_{\text{va}}, \Lambda_{\text{te}}\}, s$	
Mask construction	
Fix the learned parameters Θ^*	
$\mathbf{\Omega} \leftarrow \mathbf{\Psi}\mathbf{W}^\top + \mathbf{b}$	$O(KR^6)$
$\mathbf{P}_{u,d} \leftarrow \mathbb{I}[d = \arg\max_j \mathbf{\Omega}_{u,j}]$	
Prediction	
for each $\Lambda_s \subset \Lambda_{\text{eval}}$ **do**	
Form interaction batch $\mathbf{X}$	
$\mathbf{X}' \leftarrow \text{Householder}(\mathbf{X})$	$O(sR^6)$
$\mathbf{Z} \leftarrow \mathbf{X}'\mathbf{P}^\top$	$O(sKR^3)$
$\mathbf{H} \leftarrow \mathcal{N}(\mathbf{Z})$	$O(sC_{\mathcal{N}})$
$\widehat{\mathbf{y}} \leftarrow \sigma(\mathbf{H}\mathbf{w}_{\text{out}})$	
Store $\widehat{\mathbf{y}}$	
end for	
Output: predictions $\widehat{\mathbf{y}}$	

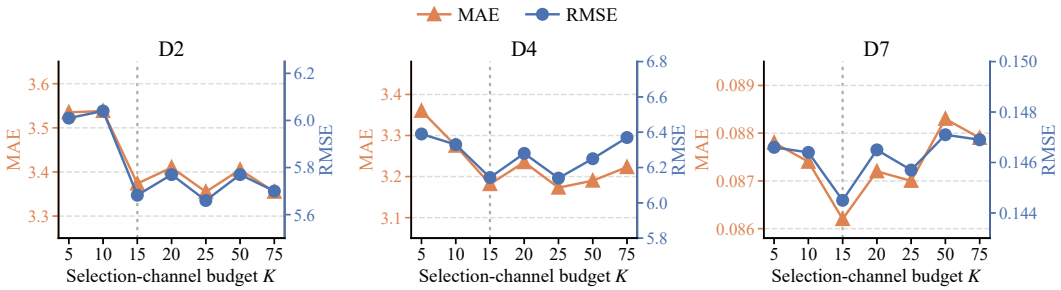


Figure 6: MAE and RMSE across different selection-channel budgets K on D2 (NYCTaxi), D4 (NAPrec), and D7 (Alibaba). The vertical dashed line marks $K = 15$, the value used in all main experiments.

B.3 Hyperparameter Sensitivity

Selection-channel budget. Figure 6 reports MAE and RMSE across selection-channel budgets $K \in \{5, 10, 15, 20, 25, 50, 75\}$ on three datasets with different training densities. Both errors remain stable across the tested range. On D2, MAE improves from 3.54 at $K = 5$ to 3.37 at $K = 15$. Increasing K to 75 yields under 2% additional improvement. On D4, the lowest tested MAE occurs at $K = 25$ and offers a marginal gain over $K = 15$. On D7, the result at $K = 15$ is lower than those at neighboring settings, while the absolute variation remains small across the tested range. These results indicate that the learned compact task representation is stable across the tested selection-channel budgets. We use $K = 15$ in the main experiments.

Target annealing epoch. The schedule in Eq. 4 uses E_{target} to specify when the temperature reaches $\tau_{\min}$. Table 3 reports validation-set screening results for the candidate epochs on each dataset. Hyperparameters are selected on the validation split and then held fixed for the repeated test evaluation. For E_{target}, we jointly consider the three validation metrics. Across the tested values, the preferred annealing speed varies by dataset, motivating the dataset-specific settings used in the main experiments.

B.4 Forced-Distinct Selection Control

The default selector constructs its K row-wise Gumbel-Softmax masks independently, allowing multiple rows to select the same coordinate. The forced-distinct control retains the same logits and temperature schedule but constructs the masks sequentially without replacement. During training, the selector rows are randomly permuted at each forward pass. Each row computes a continuous Gumbel-Softmax mask over the coordinates not selected by preceding rows, after which its

Table 3: Sensitivity to the target annealing epoch E_{target}. Bold denotes the setting used in the main experiments.

D1						D2					
E_{target}	50	70	130	160	**190**	E_{target}	50	70	145	175	**205**
MAE	25.0966	25.8624	25.0935	23.9322	23.1735	MAE	3.4147	3.5686	3.5425	3.5080	3.4198
MRE	0.2228	0.2244	0.2282	0.2196	0.2114	MRE	0.4662	0.4770	0.4793	0.4661	0.4579
RMSE	51.3614	53.9646	50.1799	45.8864	47.6326	RMSE	5.8440	6.1468	6.0914	6.0249	5.8479
D3						D4					
E_{target}	50	70	140	170	**200**	E_{target}	50	70	**100**	130	160
MAE	0.6548	0.4400	0.4333	0.4413	0.4180	MAE	3.3632	3.2936	3.1774	3.1903	3.2387
MRE	0.0261	0.0175	0.0173	0.0176	0.0166	MRE	0.5383	0.5183	0.5131	0.5482	0.5536
RMSE	0.8608	0.5733	0.5652	0.5743	0.5416	RMSE	6.5575	6.4929	6.1458	6.1129	6.3408
D5						D6					
E_{target}	20	75	**100**	120		E_{target}	20	**75**	100	120	
MAE	0.6269	0.6213	0.6196	0.6177		MAE	4.1058	4.0751	4.1872	4.1454	
MRE	1.3180	1.3288	1.2649	1.2652		MRE	0.8296	0.8427	0.7962	0.8129	
RMSE	1.7383	1.7460	1.7371	1.7414		RMSE	31.6705	29.4678	31.7355	31.9945	

D7					
E_{target}	50	70	145	**175**	205
MAE	0.0895	0.0884	0.0886	0.0881	0.0883
MRE	0.2457	0.2432	0.2479	0.2421	0.2522
RMSE	0.1482	0.1485	0.1486	0.1475	0.1479

maximum-probability coordinate is masked from the remaining rows. The masks are then restored to their original row order. Each row therefore remains continuous over its available coordinates, while the maximum-probability selections are distinct across rows.

At inference, the selector rows are processed sequentially without Gumbel noise. Each row selects the highest-scoring coordinate not selected by an earlier row, producing a deterministic one-hot mask $\mathbf{P} \in \{0,1\}^{K\times D}$ that satisfies $\mathbf{P}\mathbf{1}_D = \mathbf{1}_K$ and $\mathbf{P}^\top \mathbf{1}_K \leq \mathbf{1}_D$. The forced-distinct variant therefore has $J_{\text{eff}} = K$ by Proposition 2, whereas the default selector permits $J_{\text{eff}} \leq K$. For both variants, we measure J_{eff} from the final deterministic inference mask as the number of distinct selected coordinates.

For a matched comparison, both variants retain the Householder orthogonal layer and the same backbone and decoder. We fix $K = 15$ and use the same data splits, preprocessing, optimization settings, dataset-specific temperature schedules, and 10 random seeds, changing only whether distinctness is enforced across the output channels. Table 4 reports MAE, MRE, RMSE, and J_{eff}.

Table 4: Default and forced-distinct selection under matched experimental settings. Values are mean $\pm$ sample standard deviation over 10 runs. Lower mean errors within each dataset are shown in **bold**.

Dataset	**Constraint**	**MAE**	**MRE**	**RMSE**	$\boldsymbol{J}_{\text{eff}}$
D1	Default	**23.3296**$\pm$0.4017	**0.2123**$\pm$0.0028	**48.3073**$\pm$1.6474	3.7$\pm$0.5
	Forced distinct	23.9862$\pm$0.4366	0.2135$\pm$0.0028	50.3870$\pm$1.0640	15.0$\pm$0.0
D2	Default	3.3736$\pm$0.0285	0.4511$\pm$0.0071	5.6824$\pm$0.0523	6.1$\pm$2.5
	Forced distinct	**3.3367**$\pm$0.0337	**0.4503**$\pm$0.0042	**5.5961**$\pm$0.0652	15.0$\pm$0.0
D3	Default	0.4110$\pm$0.0077	0.0164$\pm$0.0003	0.5300$\pm$0.0097	6.4$\pm$1.6
	Forced distinct	**0.3889**$\pm$0.0061	**0.0155**$\pm$0.0003	**0.5061**$\pm$0.0081	15.0$\pm$0.0
D4	Default	3.1816$\pm$0.0661	**0.5116**$\pm$0.0261	6.1434$\pm$0.1725	6.4$\pm$0.8
	Forced distinct	**3.1628**$\pm$0.0638	0.5120$\pm$0.0342	**6.0798**$\pm$0.1576	15.0$\pm$0.0
D5	Default	**0.6168**$\pm$0.0083	1.0572$\pm$0.0216	**1.7380**$\pm$0.0201	15.0$\pm$0.0
	Forced distinct	0.6210$\pm$0.0048	**1.0444**$\pm$0.0262	1.7434$\pm$0.0074	15.0$\pm$0.0
D6	Default	4.0462$\pm$0.0378	0.8103$\pm$0.0089	**29.3827**$\pm$0.3435	15.0$\pm$0.0
	Forced distinct	**4.0342**$\pm$0.0599	**0.8048**$\pm$0.0132	29.4328$\pm$0.5925	15.0$\pm$0.0
D7	Default	0.0862$\pm$0.0008	**0.2376**$\pm$0.0033	0.1445$\pm$0.0009	6.9$\pm$1.0
	Forced distinct	**0.0858**$\pm$0.0007	0.2378$\pm$0.0040	**0.1444**$\pm$0.0008	15.0$\pm$0.0

The forced-distinct variant yields lower mean MAE, MRE, and RMSE on D2 and D3, while default OASENT has lower means under all three metrics on D1. The remaining datasets show mixed results across the three metrics.

C ALTERNATIVE INSTANTIATIONS

The alternative instantiations follow the same representation pipeline: the orthogonal layer maps $\mathbf{x}$ to $\mathbf{x}' = \mathbf{x}\mathbf{Q}$, and the selection module forms a K-channel representation $\mathbf{z}$ and decodes it into the D-dimensional representation $\mathbf{h} = \mathcal{N}(\mathbf{z})$ for scalar prediction. We vary the orthogonal parameterization or the encoding mechanism while retaining the prediction objective in Eq. 6.

C.1 ORTHOGONAL PARAMETERIZATIONS

We compare differentiable constructions of the orthogonal coordinate transformation: the default Householder parameterization, the Cayley transform, and Gram-Schmidt orthogonalization. Each construction retains the D-dimensional interaction space, with the default Gumbel-Softmax encoding, decoder, and remaining model components held fixed.

Cayley transform. The Cayley map parameterizes orthogonal matrices via a skew-symmetric matrix $\mathbf{A} = -\mathbf{A}^\top$ (Helfrich et al., 2018):

$$\mathbf{Q} = (\mathbf{I}_D - \mathbf{A})(\mathbf{I}_D + \mathbf{A})^{-1}. \tag{20}$$

For skew-symmetric $\mathbf{A}$, the resulting $\mathbf{Q}$ is orthogonal. This parameterization requires solving a linear system involving $\mathbf{I}_D + \mathbf{A}$ at each forward pass.

Gram-Schmidt orthogonalization. Given an unconstrained full-rank matrix $\mathbf{V} \in \mathbb{R}^{D\times D}$, Gram-Schmidt sequentially projects and normalizes its columns to construct $\mathbf{Q}$ (Björck, 1994). The result is orthogonal in exact arithmetic, while finite-precision accuracy depends on the numerical conditioning of the columns.

Table 5 reports the complete matched comparison of the three parameterizations on all seven datasets.

Householder achieves the lowest mean error on all 21 metric-dataset combinations. Gram-Schmidt or Cayley provides the second-lowest mean depending on the dataset and metric, but neither alternative improves upon Householder in this comparison. This consistent result supports Householder as the default orthogonal parameterization in OASENT.

C.2 SELECTION MODULES

With the Householder layer fixed, we compare four mechanisms that map $\mathbf{x}' \in \mathbb{R}^{1\times D}$ to $\mathbf{z} \in \mathbb{R}^{1\times K}$: coordinate selection, input-dependent gating, their weighted combination, and nonlinear MLP encoding. Each mechanism is paired with its associated decoder to form a selection module.

Gumbel-Softmax selection (default). The Gumbel pathway uses $\mathbf{z}_{\mathrm{G}} = \mathbf{x}'\mathbf{M}^\top$ with the continuous mask $\mathbf{M}$ during training and $\mathbf{z}_{\mathrm{G}} = \mathbf{x}'\mathbf{P}^\top$ with the deterministic one-hot mask $\mathbf{P}$ during inference. Proposition 2 characterizes the orthonormality of distinct selected directions and the effective width J_{eff} when selections repeat.

SwiGLU gating. An input-dependent SiLU gate modulates a second feature projection through component-wise multiplication (Shazeer, 2020):

$$\mathbf{z}_{\mathrm{S}} = \mathrm{Down}\big(\mathrm{SiLU}(\mathbf{x}'\mathbf{W}_{\text{gate}}) \odot (\mathbf{x}'\mathbf{W}_{\text{up}})\big), \tag{21}$$

where $\mathbf{W}_{\text{gate}}, \mathbf{W}_{\text{up}} \in \mathbb{R}^{D\times D}$ and $\mathrm{Down} : \mathbb{R}^D \to \mathbb{R}^K$ linearly projects the gated features to K channels.

Gumbel + SwiGLU joint scheme. A single learned scalar $\beta \in (0, 1)$, shared across observations and channels, combines the K-channel outputs of the two parallel pathways before the associated decoder:

$$\mathbf{z}_{\mathrm{joint}} = \beta\mathbf{z}_{\mathrm{G}} + (1-\beta)\mathbf{z}_{\mathrm{S}}. \tag{22}$$

The Gumbel pathway uses the continuous mask $\mathbf{M}$ during training and the deterministic mask $\mathbf{P}$ during inference, while the SwiGLU pathway remains input-dependent in both phases.

Auto-encoding bottleneck. This alternative uses a nonlinear MLP encoder whose layer-width structure is symmetric to that of the associated decoder. It maps the D-dimensional input to the compact representation

$$\mathbf{z}_{\mathrm{AE}} = \mathrm{Encoder}(\mathbf{x}'). \tag{23}$$

Table 5: Performance of the three orthogonal parameterizations on seven HDI tensor datasets. The Householder column reports the OASENT results from Table 2, and the selection module is fixed to its default Gumbel-Softmax instantiation. Values are mean ± sample standard deviation over 10 runs. Best means are **bold** and second-best means are underlined.

		Householder	Gram-Schmidt	Cayley
D1	MAE	**23.3296**±0.4017	24.6972±0.4553	24.4473±0.4312
	MRE	**0.2123**±0.0028	0.2214±0.0052	0.2194±0.0067
	RMSE	**48.3073**±1.6474	51.9432±1.1272	51.9052±1.5377
D2	MAE	**3.3736**±0.0285	3.5492±0.0918	3.4969±0.0440
	MRE	**0.4511**±0.0071	0.4828±0.0201	0.4767±0.0177
	RMSE	**5.6824**±0.0523	6.0729±0.1763	5.9759±0.1243
D3	MAE	**0.4110**±0.0077	0.5056±0.0366	0.4643±0.0189
	MRE	**0.0164**±0.0003	0.0202±0.0015	0.0185±0.0008
	RMSE	**0.5300**±0.0097	0.6623±0.0501	0.6059±0.0248
D4	MAE	**3.1816**±0.0661	3.4370±0.0496	3.3048±0.0448
	MRE	**0.5116**±0.0261	0.6052±0.0606	0.5486±0.0483
	RMSE	**6.1434**±0.1725	6.6475±0.1099	6.3913±0.1213
D5	MAE	**0.6168**±0.0083	0.6245±0.0057	0.6264±0.0032
	MRE	**1.0572**±0.0216	1.2173±0.0446	1.1386±0.0774
	RMSE	**1.7380**±0.0201	1.7509±0.0176	1.7799±0.0181
D6	MAE	**4.0462**±0.0378	4.1990±0.0629	4.2120±0.0947
	MRE	**0.8103**±0.0089	0.8218±0.0137	0.8203±0.0208
	RMSE	**29.3827**±0.3435	31.6005±1.0315	31.4033±0.9982
D7	MAE	**0.0862**±0.0008	0.0880±0.0008	0.0882±0.0006
	MRE	**0.2376**±0.0033	0.2458±0.0046	0.2473±0.0068
	RMSE	**0.1445**±0.0009	0.1471±0.0007	0.1474±0.0011

The encoder and decoder form a D-to-K-to-D bottleneck optimized only through the prediction loss, without an auxiliary reconstruction loss.

Table 6 reports the complete matched comparison of the four mechanisms on all seven datasets.

Gumbel-Softmax achieves the lowest mean error on 14 of the 21 metric-dataset combinations, the Gumbel+SwiGLU joint scheme on six, and SwiGLU gating on one. The auto-encoding bottleneck does not attain the lowest mean in this comparison. At the dataset level, Gumbel-Softmax leads on all three metrics for D1, D5, D6, and D7. The joint scheme leads on MAE and RMSE for D2 and D4, and on MAE and MRE for D3. Gumbel-Softmax obtains the lowest MRE on D2 and the lowest RMSE on D3, while SwiGLU gating obtains the lowest MRE on D4.

All four modules combine a K-channel bottleneck with a nonlinear decoder. Their distinct encoding mechanisms provide alternative ways to form task representations within the same orthogonal-coordinate pipeline.

Table 6: Performance of four selection modules on seven HDI tensor datasets. The Gumbel-Softmax column reports the OASENT results from Table 2, and the orthogonal layer is fixed to Householder reflections. Values are mean ± sample standard deviation over 10 runs. Best means are **bold** and second-best means are underlined.

		Gumbel-Softmax	SwiGLU gate	Gumbel+SwiGLU	AE bottleneck
D1	MAE	**23.3296**±0.4017	25.2516±0.6964	24.5783±0.2798	26.2089±0.7507
	MRE	**0.2123**±0.0028	0.2375±0.0092	0.2315±0.0047	0.2553±0.0095
	RMSE	**48.3073**±1.6474	51.8737±1.7139	50.7657±1.0050	53.0037±0.9281
D2	MAE	3.3736±0.0285	3.3557±0.0360	**3.3196**±0.0271	3.4623±0.0782
	MRE	**0.4511**±0.0071	0.4686±0.0106	0.4672±0.0078	0.4778±0.0163
	RMSE	5.6824±0.0523	5.6582±0.0789	**5.5421**±0.0572	5.8213±0.1827
D3	MAE	0.4110±0.0077	0.4093±0.0121	**0.3981**±0.0118	0.4676±0.0164
	MRE	0.0164±0.0003	0.0164±0.0005	**0.0160**±0.0005	0.0187±0.0007
	RMSE	**0.5300**±0.0097	0.5507±0.0170	0.5320±0.0157	0.6254±0.0215
D4	MAE	3.1816±0.0661	3.0489±0.0825	**2.9200**±0.0277	3.1779±0.0855
	MRE	0.5116±0.0261	**0.4832**±0.0479	0.4935±0.0392	0.5285±0.0711
	RMSE	6.1434±0.1725	5.8706±0.2049	**5.7103**±0.1306	6.2396±0.1689
D5	MAE	**0.6168**±0.0083	0.6306±0.0043	0.6235±0.0024	0.6329±0.0085
	MRE	**1.0572**±0.0216	1.3138±0.0462	1.2014±0.0226	1.1459±0.0820
	RMSE	**1.7380**±0.0201	1.7782±0.0144	1.7592±0.0069	1.7807±0.0456
D6	MAE	**4.0462**±0.0378	4.3205±0.0376	4.2277±0.0437	4.2660±0.1019
	MRE	**0.8103**±0.0089	0.8423±0.0297	0.8219±0.0178	0.8420±0.0227
	RMSE	**29.3827**±0.3435	32.9571±0.3247	31.9860±0.8027	31.5348±1.1219
D7	MAE	**0.0862**±0.0008	0.0917±0.0005	0.0916±0.0005	0.0945±0.0006
	MRE	**0.2376**±0.0033	0.2570±0.0047	0.2562±0.0038	0.2617±0.0042
	RMSE	**0.1445**±0.0009	0.1501±0.0014	0.1498±0.0008	0.1509±0.0012

D MAIN RESULTS WITH STANDARD DEVIATIONS

Table 7: Main results reported as mean ± sample standard deviation over 10 runs. Best means are **bold**, and second-best means are underlined across all eight models.

		CPD	CoSTCo	HOCTC	CoATF
D1	MAE	52.7699±4.7478	29.9612±0.5188	34.2456±1.1738	25.1594±0.6358
	MRE	0.6140±0.1756	1.3110±0.0069	0.2850±0.0123	0.2372±0.0095
	RMSE	116.3830±6.1703	58.9369±1.4934	71.5450±4.3547	53.1192±1.5801
D2	MAE	8.5408±0.0183	8.4526±0.0000	4.4776±0.1448	3.4214±0.0065
	MRE	2.6037±0.0110	2.3833±0.0000	0.6551±0.0053	0.4911±0.0173
	RMSE	12.0059±0.0201	12.2616±0.0000	7.7251±0.2978	5.6893±0.0687
D3	MAE	2.4938±0.0012	0.6554±0.0145	0.7582±0.0407	0.6402±0.0307
	MRE	0.1000±0.0001	0.0272±0.0006	0.0309±0.0017	0.0257±0.0012
	RMSE	2.9231±0.0012	0.9079±0.0173	1.0419±0.0523	0.8446±0.0420
D4	MAE	9.9003±0.0046	4.7653±0.2103	5.0317±0.1924	3.9584±0.1060
	MRE	1.2792±0.0012	1.2714±0.0833	1.1339±0.1307	0.9302±0.0469
	RMSE	18.6089±0.0056	9.5898±0.4489	10.1165±1.0687	7.5577±0.2775
D5	MAE	1.2040±0.0000	0.7317±0.0074	0.7750±0.0163	0.7121±0.0239
	MRE	1.9760±0.0002	1.7567±0.1228	1.7813±0.2877	1.6707±0.1365
	RMSE	2.9117±0.0000	1.9418±0.0161	1.9684±0.0585	1.7639±0.0088
D6	MAE	10.9362±0.0002	6.3075±0.1635	6.5655±0.3062	5.2686±0.0845
	MRE	1.3815±0.0010	1.5937±0.0553	1.0777±0.1632	1.0091±0.0751
	RMSE	55.2629±0.0001	42.5711±0.3927	41.7357±0.4196	36.4626±0.5092
D7	MAE	0.3051±0.0009	0.1211±0.0039	0.1054±0.0027	0.1011±0.0012
	MRE	1.1308±0.1177	0.3960±0.0251	0.2931±0.0081	0.2784±0.0068
	RMSE	0.4149±0.0008	0.1769±0.0039	0.1610±0.0022	0.1515±0.0009

		NEAT	ConvNTC	ANTucF	OASENT
D1	MAE	28.1055±0.8505	24.9536±0.5511	26.3048±1.4396	**23.3296**±0.4017
	MRE	0.2456±0.0067	0.2308±0.0075	0.2529±0.0116	**0.2123**±0.0028
	RMSE	61.0536±2.5235	57.4976±1.4095	53.7448±3.5626	**48.3073**±1.6474
D2	MAE	4.1451±0.0668	3.4495±0.0330	3.5285±0.0740	**3.3736**±0.0285
	MRE	0.5795±0.0125	0.5057±0.0095	0.4931±0.0332	**0.4511**±0.0071
	RMSE	7.1572±0.1160	5.8060±0.0607	5.9277±0.1150	**5.6824**±0.0523
D3	MAE	0.9918±0.0286	0.7222±0.0277	0.5083±0.0417	**0.4110**±0.0077
	MRE	0.0402±0.0011	0.0291±0.0011	0.0204±0.0017	**0.0164**±0.0003
	RMSE	1.3127±0.0369	0.9506±0.0349	0.6861±0.0568	**0.5300**±0.0097
D4	MAE	3.9619±0.0318	3.9461±0.1505	3.2587±0.0948	**3.1816**±0.0661
	MRE	0.8636±0.0631	0.9110±0.0517	0.5549±0.0554	**0.5116**±0.0261
	RMSE	7.8303±0.0768	8.2672±0.2886	6.4257±0.2004	**6.1434**±0.1725
D5	MAE	0.6729±0.0052	0.7172±0.0077	0.6390±0.0126	**0.6168**±0.0083
	MRE	1.2901±0.0263	1.4240±0.0682	1.2361±0.1038	**1.0572**±0.0216
	RMSE	1.8256±0.0094	1.9126±0.0103	1.7782±0.0169	**1.7380**±0.0201
D6	MAE	4.5173±0.0294	6.4138±0.1587	4.3773±0.0748	**4.0462**±0.0378
	MRE	0.9317±0.0177	1.2019±0.0547	0.8261±0.0116	**0.8103**±0.0089
	RMSE	34.1956±0.2658	42.9911±0.3669	32.3394±0.6113	**29.3827**±0.3435
D7	MAE	0.0916±0.0007	0.1176±0.0039	0.0949±0.0009	**0.0862**±0.0008
	MRE	0.2580±0.0036	0.3181±0.0211	0.2673±0.0082	**0.2376**±0.0033
	RMSE	0.1480±0.0008	0.1728±0.0042	0.1515±0.0013	**0.1445**±0.0009