Atlasing Pattern Space: A Framework for Structured Latent Representations in LLMs

Freeman Hui

1 Abstract

Atlasing Pattern Space (APS) is a novel framework for learning **structured latent representations** in large language models (LLMs) and other high-dimensional domains. APS diverges from standard embedding approaches by introducing explicit **geometric constraints** during representation learning. In particular, APS enforces three complementary properties in the latent space: **Topology preservation (T)** – similar patterns lie close together, preserving neighborhood relations; **Causality awareness (C)** – representations are invariant to nuisance factors and emphasize underlying causal features using independence-promoting losses (e.g. HSIC, IRM); and **Energy basins (E)** – the latent space is shaped into attractor basins or energy valleys corresponding to meaningful patterns or prototypes. The result is an **interpretable "atlas" of the pattern space**: a latent manifold where local neighborhoods reflect true similarity, axes align with stable, generalizable factors, and energy landscapes form semantic clusters that facilitate memory and generation. We outline the APS framework, relate it to prior work in manifold learning, causal representation learning, and energy-based models, and discuss its applicability to NLP, vision, and recommender systems. Experiments will illustrate how APS yields representations that are both **geometrically structured and highly interpretable**, enabling improved generalization and insightful visualization of learned pattern spaces.

2 Introduction

Modern deep learning representations (e.g. word embeddings or latent vectors in VAEs) are typically learned with generic objectives that lack **explicit geometric structure**. This can lead to latent spaces that are **uninterpretable** and not well-aligned with the true structure of the data. For example, vanilla autoencoders often learn manifolds with distorted local connectivity or overfitting to noise. Likewise, word embeddings from LLMs may capture statistical co-occurrences but entangle unrelated factors (topic, style, context) in a way that complicates interpretation. To address these issues, we propose **Atlasing Pattern Space (APS)** – a framework to endow latent representations with a *meaningful geometry*. APS imposes three guiding principles: (1) **Topology** – preserving the manifold structure so that points that are similar in the input domain remain neighbors in latent space; (2) **Causality** – forcing latent features to be invariant to nuisance variables and emphasizing causal, generalizable attributes; (3) **Energy** – shaping the latent space's energy landscape into distinct basins of attraction corresponding to prototypical patterns.

Motivation: The name *"Atlasing"* evokes the creation of a map or atlas of all patterns (e.g. linguistic or visual patterns) such that distance and neighborhoods on the map reflect true semantic or functional similarity. Unlike standard embedding methods which largely treat latent dimensions as

unstructured, APS treats representation learning as a **manifold learning problem** with additional causal and energy-based regularization. By doing so, APS aims to produce latent "charts" that are easier to interpret and navigate – much like an atlas that faithfully represents the terrain: - In NLP, an APS-learned embedding might place synonyms or contextually similar phrases in adjacent regions (topology), align dimensions with abstract concepts (causality), and form energy basins for distinct topics or themes (energy). - In computer vision, APS could map images such that images with similar content or style cluster together (topology), latent variables isolate factors like lighting or viewpoint (causality), and each object category corresponds to an energy basin that stores its prototypical patterns. - In recommendation systems, user/item embeddings could be structured so that similar users/items lie in contiguous latent neighborhoods, confounding factors (e.g. popularity) are factored out, and communities or genres appear as attraction basins.

By integrating these properties, APS promises representations that support **better generalization** (through invariant features), **robustness to spurious correlations** (through causal structure), and **enhanced interpretability** (through topologically and energetically organized latent maps). In the following sections, we formalize the APS framework and discuss related work that inspires each component (Topology, Causality, Energy). We then outline the methodology for implementing APS and propose experiments to evaluate its benefits.

3 Related Work

3.1 Topology-Preserving Embeddings

Our emphasis on latent **topology preservation** builds on a rich history of manifold learning and neighbor-preserving embeddings. Classical techniques like **t-SNE**[1] and **UMAP**[2] aim to embed high-dimensional data into low dimensions (e.g. 2D) for visualization, such that similar points stay close and multi-scale structure is maintained. In particular, UMAP uses a framework from algebraic topology to learn a low-dimensional mapping that preserves both local and some global structure of the data manifold[2], while t-SNE focuses on retaining local neighbor affinities and revealing cluster structure at multiple scales[1]. These methods underscore the value of respecting the intrinsic topology of data, although they are typically used as post-hoc visualizers rather than as trainable model components.

In neural network research, recent work has explicitly added topological or geometric constraints to latent spaces. **Topological Autoencoders** (Moor et al. 2020) introduced a differentiable loss based on persistent homology to ensure that the topology (e.g. connectivity, loops) of the latent space matches that of the input space. By penalizing differences in Betti numbers and other topological features between input and latent distributions, they preserved multi-scale connectivity and improved interpretability of latent dimensions. Other approaches enforce local geometric fidelity: for example, **Local Distance Preserving Autoencoders** (Chen et al. 2022) add a loss that keeps the distances between each point and its k-nearest neighbors in data space similar in latent space. This is achieved via a continuous k-NN graph that captures topological features at all scales, used as a constraint during training. Such methods align with earlier ideas like **Laplacian eigenmaps** and **locally linear embedding (LLE)**, which also preserve neighbor relations in a lower-dimensional embedding of the data manifold.

Graph-based regularization of latent geometry has shown promise in autoencoders. For instance, **Neighborhood Reconstructing Autoencoders (NRAE)** (Lee et al. 2021) incorporate a term ensuring that each data point's local neighborhood (from a precomputed graph) is reconstructed by the decoder, thus correcting "wrong local connectivity and geometry" often observed in vanilla AEs. Similarly, the **Witness Autoencoder (W-AE)** and **Geometry-Regularized Autoencoder**

(GRAE) introduced topological and geometric regularizers (e.g. using witness complexes or manifold charts) to shape the latent space. These works demonstrate that **imposing topology-awareness during representation learning leads to latent spaces that better reflect the true structure of data**, which can improve downstream tasks and the realism of interpolations. APS adopts this principle: our **Topology (T)** component will preserve neighborhood relationships (e.g. via a k-NN graph or topological loss) so that the learned atlas maintains the continuity and connectivity of the original pattern space.

3.2 Causal and Invariant Representation Learning

The **Causality (C)** component of APS seeks to make latent features invariant to nuisance factors and aligned with stable, meaningful properties. This idea is inspired by research in **causal representation learning** and **domain generalization**. A key insight from causality is that models should capture the *invariant mechanisms* underlying data rather than spurious correlations. **Invariant Risk Minimization (IRM)** (Arjovsky et al. 2019) formalized this by learning a data representation such that *the optimal classifier on that representation is the same across multiple environments*. By leveraging data from different environments (or domains), IRM encourages the encoder to discard features that are inconsistent (spurious) and keep those that have a stable relationship with the target, thereby improving out-of-distribution (OOD) generalization. APS can incorporate this principle by using multiple data contexts or augmentations and adding a penalty if a classifier's predictions differ between contexts when using the APS embedding.

Another line of work uses **independence criteria** to enforce invariances. The *Hilbert-Schmidt Independence Criterion* (HSIC) is a kernel-based measure of statistical independence. It has been used as a loss to encourage representations Z to be independent of certain variables V (for example, sensitive attributes or domain labels). Greenfeld and Shalit (2020) applied HSIC as a regularizer to achieve robust models under covariate shift. By penalizing any dependence between the model's residuals and the input distribution, their HSIC-based loss yielded predictors where $Y - \hat{\{f\}}(X)$ is nearly independent of X, corresponding to a scenario where only the causal relation (and independent noise) remains. In APS, we can use HSIC-based penalties to encourage that the learned latent Z is independent of nuisance factors (e.g. style, noise, context that we want to factor out). Similarly, other works like *Domain-Adversarial Training* and *Maximum Mean Discrepancy (MMD)* have sought to remove domain-specific information from embeddings, but HSIC offers a direct, differentiable independence measure.

There is also overlap between invariant representation learning and **disentangled representation learning**. Methods such as **β-VAE** (Higgins et al. 2017) aim to learn latent factors that correspond to independent generative factors of variation[3]. By constraining the VAE's latent channel capacity (via a higher β weight on the KL-divergence term), β-VAE encourages the latent dimensions to capture distinct aspects of the data (for example, in an image dataset, one dimension may capture "rotation" while another captures "scale")[4]. The result is an interpretable factorized representation that is aligned with *causal factors* in the data generation process, achieved without supervision. APS's causality module shares this goal of **isolating meaningful factors**: through losses like IRM or HSIC (and potentially by borrowing ideas from β-VAE to enforce factorization), APS encourages each latent dimension or subspace to correspond to a stable property of the input, invariant to minor changes or context. Indeed, the broader vision of **causal representation learning** is to uncover latent features that correspond to real-world causal variables, a direction articulated in surveys like Schölkopf et al. (2021) *"Towards Causal Representation Learning."* APS contributes to this direction by integrating causal invariance constraints directly into the representation learning objective.

3.3 Energy-Based Models and Attractor Networks

The **Energy (E)** component of APS introduces an **energy-based perspective** to the latent space. Energy-Based Models (EBMs) assign an unnormalized "energy" score to configurations (in our case, latent vectors), such that low-energy regions correspond to probable or familiar patterns. By shaping the latent space's energy landscape into **basins of attraction**, APS aims to create distinct wells (valleys) that capture clusters or prototypes of patterns. This idea is reminiscent of **Hopfield networks** and other attractor models. A classical Hopfield network (Hopfield 1982) stores patterns as stable fixed points of a dynamical system; when the network state is perturbed to a new input, it iteratively updates and converges to the nearest stored pattern (an attractor). Recent work has modernized this concept: *"Hopfield Networks is All You Need"* (Ramsauer et al. 2021) showed that a continuous-state Hopfield layer can store exponentially many patterns and that its update rule is equivalent to the Transformer's attention mechanism[5][6]. Importantly, they identified different types of energy minima in such networks: global minima that average over all patterns, metastable states averaging subsets of patterns, and fixed-point attractors corresponding to individual stored patterns[5]. This suggests that deep networks can incorporate Hopfield-like memory to perform pooling, association, and rapid content-based retrieval[7]. APS leverages this concept by aiming for a latent space where each significant pattern or concept acts as an **attractor**. For example, in an NLP context, an abstract concept (like *sports*) might form an energy basin that attracts semantically related sentence embeddings, enabling the model to recall or generate prototypical examples of that concept.

Energy-based modeling has also been applied **in the latent spaces of generative models**. Rather than using a fixed prior (e.g. Gaussian) in a VAE or generator, researchers have learned **latent space EBMs** to better model complex distributions. For instance, Pang et al. (2020) train a VAE-like generative model where the latent prior $p(z)$ is not a simple Gaussian but given by an energy-based model learned jointly with the decoder[8]. Their latent EBM prior, parameterized by a small network, captures the structure of the latent codes that correspond to real data, leading to improvements in image and text generation[9][10]. Because the latent space is low-dimensional, sampling from the EBM (via MCMC) is efficient and yields diverse samples that respect the learned data manifold[9][11]. This approach essentially carves out an **energy landscape in latent space shaped by the data**, rather than assuming latent variables are independent. APS's energy component aligns with this strategy: by training an energy function $E(z)$ alongside the encoder, we ensure that latent representations of training data lie in low-energy valleys, while high-energy barriers separate distinct pattern regions. Memory-based energy functions (e.g. using a memory bank of prototypical latent vectors) could be employed to form these basins: each prototype exerts an attractive force in latent space, creating an energy well. This is analogous to how **prototype networks** classify by proximity to class centroids, but here extended as a continuous energy landscape useful for both generation and classification.

3.3.1 Visualization of Energy Landscapes

To illustrate the energy basin concept concretely, Figure 1 shows a 3D energy surface with four prototype basins. The low-energy valleys (shown in blue) cluster latent codes into semantic regions, with each prototype marked by a red X. This visualization demonstrates how the energy function $E(z)$ creates natural attractors in the latent space.

The sharpness of these energy basins can be controlled by a temperature parameter β.

Figure 3 demonstrates the attractor dynamics by showing trajectories of points descending the energy landscape. Each trajectory flows from an initial position toward the nearest prototype

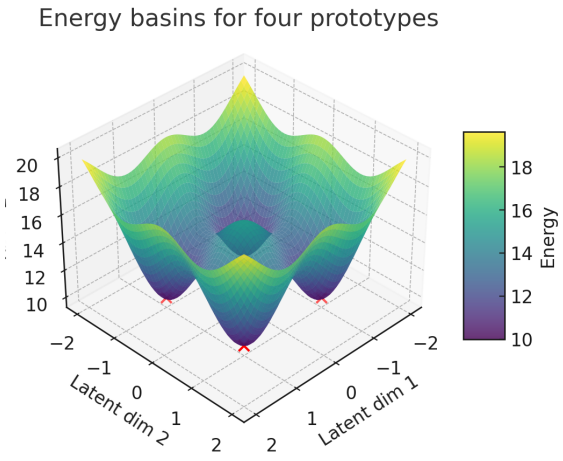

Figure 1: 3D energy surface with four prototype basins (marked by red X's). Low-energy valleys cluster latent codes into semantic regions.

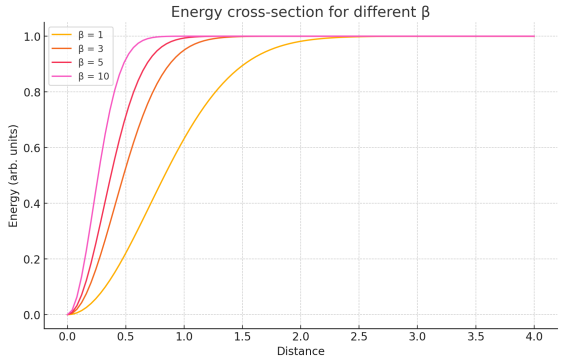

Figure 2: Energy vs. distance for different β: sharper $\beta = 10$ basins approximate Hopfield-like memory; lower $\beta = 1$ yields smoother RBF-style landscapes.

basin, illustrating how the energy function guides latent representations toward stable semantic clusters. This attractor behavior provides robustness to noise and enables memory recall: perturbed representations naturally flow back to their corresponding prototypes.

The idea of **energy valleys aiding interpretation** can be seen through techniques like analyzing latent vector fields. Recent studies observe that standard training often already induces some attractor dynamics in latent spaces[12][13] – autoencoders with contractive mappings can cause points to flow towards regions of high data density (an implicit energy model)[14][15]. APS makes this explicit and controllable. By designing $E(z)$ (or using a Hopfield layer) we define where the attractors should be, which can correspond to semantic categories or recurring prototypes in data. This has practical benefits: for **generation**, one can sample from these basins to produce novel but coherent outputs; for **classification**, the basin a new point falls into can directly indicate its class or type; for **anomaly detection**, points landing in no known basin (high energy areas) are flagged as outliers. Overall, the Energy component of APS connects to a broad trend of integrating **EBMs and dynamical systems** with deep learning[7], providing a bridge between pattern recognition and pattern generation via the geometry of the latent space.

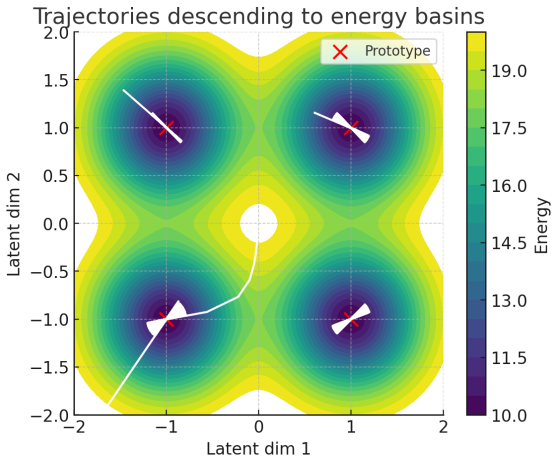

Figure 3: Trajectories descending the energy landscape into attractor basins. Each point flows via gradient descent on $E(z)$ to the nearest prototype.

3.4 Structured and Interpretable Embeddings

Beyond the specific T, C, and E aspects, APS relates to the general pursuit of **structured and interpretable embeddings** in machine learning. Traditional word embeddings (e.g. Word2Vec, GloVe) exhibit surprising linear structure enabling analogies, but are largely learned from distributional statistics. Follow-up analyses have shown that these embedding spaces have meaningful directions (e.g. gender or tense directions) but also problematic biases. By contrast, approaches that *impose* structure can yield more interpretable representations. One notable example is **Hyperbolic Embeddings** for representing hierarchical data. Nickel & Kiela (2017) introduced **Poincaré Embeddings**, which learn embeddings in a hyperbolic space (an n-dimensional Poincaré ball) to naturally represent tree-like hierarchies[16]. Thanks to the negative curvature, hyperbolic space can encode hierarchical relationships with much lower distortion than Euclidean space – allowing one to capture both **similarity and hierarchy** simultaneously[16]. They demonstrated significantly improved representation capacity and generalization for data with latent hierarchies (like WordNet noun relationships) when using hyperbolic embeddings as opposed to Euclidean[17]. This is a powerful reminder that the **choice of geometry** for the latent space can profoundly shape what structures can be efficiently represented. APS is agnostic to a specific geometry (one could even conceive APS on a hyperbolic manifold if the data is hierarchical), but it shares the spirit of *baking domain-relevant structure into the embedding space*. In the case of APS, the inductive biases are topological (neighbor relations), causal, and energy-based structure.

Interpretable latent dimensions are also pursued in disentanglement research (as mentioned with β-VAE) and in various supervised settings (e.g. learning a latent space aligned with known attributes or concepts). In NLP, there have been efforts to find or impose latent dimensions that correspond to semantic attributes – for example, latent edit vectors for style, sentiment, etc., which can be manipulated. APS could help here by explicitly designating parts of the latent space to capture certain factors (through the causal invariance objective) and ensuring those parts are used consistently across data. Furthermore, visualization techniques like **UMAP and t-SNE** can be directly applied to APS embeddings to produce "maps" of the learned pattern space, potentially revealing clear organization (clusters, hierarchies, continuous variations) that align with human-understandable categories. By contrast, in a standard embedding space, such visualizations might be muddled by entangled factors or lack of global structure. There are also alternatives

like **Topological Data Analysis (TDA)** tools (e.g. Mapper algorithm) that could be used to assess how well APS preserves the shape of data. Indeed, TopoGraph-based evaluation was used by Moor et al. to show improved latent topology. We anticipate that APS embeddings will lend themselves to clearer topological summaries and interactive exploration, essentially acting as an atlas for researchers to **navigate the pattern space**.

4 Atlasing Pattern Space (APS) Framework

4.1 Overview

APS learns an encoder $f : X \to Z$ (and potentially a decoder $g : Z \to X$ in an autoencoder setup) such that the latent space Z becomes an **atlas** of the data manifold with the properties of **Topology preservation (T)**, **Causal invariance (C)**, and **Energy structuring (E)**. These three aspects are enforced via dedicated loss terms added to the training objective alongside any task-specific loss (e.g. reconstruction error or prediction loss). Figure 1 (conceptual; see Appendix) illustrates the APS concept: in latent space, points form neighborhoods corresponding to similar inputs (T), lie on coordinate axes corresponding to meaningful factors (C), and cluster into basins around prototypical exemplars (E).

Formally, let $z = f(x)$ be the embedding of input x. APS's training objective can be written as:

$$\mathcal{L}_{\text{APS}} = \mathcal{L}(x, z) + \lambda_T \mathcal{L}_T(x, z) + \lambda_C \mathcal{L}_C(x, z) + \lambda_E \mathcal{L}_E(z),$$

where $\mathcal{L}$ could be a reconstruction loss (if APS is an autoencoder) or a classification loss (if APS is used in a supervised setting), and $\lambda_T, \lambda_C, \lambda_E$ are weights for the regularizers. We describe each component loss below:

(T) Topology-Preserving Loss: $\mathcal{L}_T$ ensures that local neighborhoods in input space X are reflected in Z. One implementation is a **continuous k-NN graph loss**: we construct a graph G on the batch (or dataset) in input space where edges connect each point to its k nearest neighbors (using original input features or a predefined distance). We then encourage the distances in latent space $d_Z(f(x_i), f(x_j))$ to be small for edges (i, j) in G and, optionally, to be larger for non-neighbor pairs. For example, a **triplet loss** or contrastive loss can be used: $\mathcal{L}_T = \sum \left[\Delta - |z_i - z_k|\right]_+$, where Δ is a margin. Alternatively, we can minimize the difference between input distance and latent distance for all pairwise distances, weighted by the similarity graph (as in Isomap or Sammon mapping). Another powerful variant is the **topological loss** from Topological AEs: compute a persistence diagram for the point cloud in input space and in latent space, then penalize discrepancies. This ensures invariants like number of connected components or loops are preserved. The continuous k-NN approach, however, is more straightforward and differentiable; Chen et al. (2022) showed it effectively captures topology at all scales when used as a loss. In practice, $\mathcal{L}_T$ will keep f from distorting the manifold: **if two texts are similar (high lexical or semantic overlap), APS will place them nearby in** Z, preserving their neighbor relationship, and if two images are dissimilar, APS will not arbitrarily force them together.

(C) Causal Invariance Loss: $\mathcal{L}_C$ promotes invariance to nuisance and alignment with causal features. There are multiple design choices for this component: - **Multi-environment IRM loss:** If we have data segmented into environments (or we create environments via augmentation), we can apply the IRM principle. For each environment e, a classifier w (e.g. a simple linear model) is trained on $\{z_i, y_i\}$. $\mathcal{L}_C$ would include a term that encourages these classifiers to have **matching parameters across environments**, i.e. the same w works for all, which is the IRM objective. In practice, Arjovsky et al. introduced a penalty $\Omega(w, Z^{(e)})$ that is minimized when $\nabla(w \circ f; X, Y) = 0$

for all environments (this formalism essentially tries to find f such that there is an invariant optimal classifier). We can incorporate a differentiable approximation of this condition. - **HSIC loss for independence:** If certain nuisance factors v are known or can be estimated (e.g. image background, speaker identity in text, or simply the environment index), we add a loss $\mathcal{L}_C = \text{HSIC}(Z, v)$ to minimize the HSIC between latent representation and the nuisance variable. By driving HSIC to zero, we make $Z \perp v$ (no statistical dependence). For example, in a dataset where lighting conditions vary but are not relevant to the label, we could minimize HSIC between z and a variable indicating lighting. This encourages $f(x)$ to discard lighting information. HSIC is differentiable and has been used in domain adaptation and fairness contexts to de-correlate representations from undesired factors. - **Variance and covariance penalties:** In unsupervised settings, one may encourage the latent dimensions to be statistically independent (like FactorVAE or β-TCVAE approaches). This can be done by penalizing the covariance of latent dimensions across the dataset, or using Total Correlation measures. Although not as explicit as causal invariance, an independent-factor representation often aligns with meaningful generative factors[3]. - **Adversarial invariance:** Another option (not kernel-based) is to train a discriminator that tries to predict the nuisance factor from z, and simultaneously train f to fool that discriminator (similar to Domain-Adversarial Neural Networks). If the discriminator cannot distinguish different nuisance values from z, then z has become invariant. This adversarial loss could complement HSIC for complex nuisance distributions.

Regardless of implementation, the effect of $\mathcal{L}_C$ is that **APS embeddings focus on what truly matters for the task** (or for describing the data) and ignore superficial cues. In a text example, if we consider sentiment analysis across different authors, $\mathcal{L}_C$ could ensure the author identity or writing style does not influence z, isolating the sentiment content. Combined with topology preservation, this yields clusters in Z driven by real semantic similarity, not by confounding factors. This also improves generalization: a representation that captures, say, "cow vs camel" based on shape rather than background (recalling the cows vs camels example of spurious correlations[18]) will transfer to new backgrounds, which IRM's philosophy guarantees.

(E) Energy Shaping Loss: $\mathcal{L}_E$ defines and shapes an energy function $E(z)$ over the latent space. One approach is to instantiate E as a parametric function (e.g. a small neural network or even a quadratic form) and treat it as an **energy-based model prior**[8]. During training, we want $E(f(x))$ to be low for latent codes of real data, and higher for other regions. We might not have "negative" latent samples upfront, so we can generate them by perturbing real codes or sampling from a base distribution. The loss can be like a contrastive divergence or score matching: for example,

$$\mathcal{L}_E = E(z) + \log \mathbb{E}_{z \sim p_\alpha}[\exp(-E(z))],$$

where p_α is a proposal distribution for negative samples (could be a running estimate of the latent distribution, or simply the prior if one exists). In practice, Pang et al. (2020) perform short-run MCMC in latent space to draw negative samples that approximate the model's current high-density regions[19][20]. We can do similar by performing a few Langevin steps on $E(z)$ starting from perturbed real embeddings. The goal is to **shape E such that real z's sit in deep energy wells**. These wells will naturally partition the latent space if multiple modes are present. An alternative or complementary strategy is to define energy in terms of distance to prototypes: let $\{u_k\}$ be a set of learned prototype vectors in Z (one per cluster or concept). Define $E(z) = \min_k |z - u_k|^2$ (or a softmin). This energy will be low near one of the prototypes and high far from all prototypes. We can learn the prototypes $\{u_k\}$ along with f. The $\mathcal{L}_E$ in this case could simply push each z_i toward the nearest prototype (reducing energy) and possibly push prototypes apart. This resembles *vector quantization* or *K-means* in latent space, but with a smooth energy surface. A memory-augmented approach could store many exemplars and use an attention mechanism (like Hopfield/Transformer

style) to pull z towards the closest memory[6]. Indeed, using a Hopfield layer as part of the encoder can directly produce an attractor dynamic: given an initial encoding of x, the Hopfield network "cleans it up" by associating it to the closest stored pattern, effectively placing z into a basin of attraction of that pattern.

In all cases, $\mathcal{L}_E$ encourages **latent clustering**: points that are variations of a common prototype will end up in the same basin. This can greatly aid interpretability – e.g. one might find a basin corresponds to a certain class of images, or a topic in text. It also provides a form of regularization and robustness: if random noise pushes a representation a bit, the attractor dynamic can pull it back to the stable prototype. In generation tasks, one can sample by picking a prototype and adding noise, then letting the energy dynamics settle, to generate a new sample around that prototype (similar to memory recall with noise as in Hopfield networks).

4.2 Training Procedure

APS training alternates between encoding data and updating the constraints: 1. **Forward pass:** Compute $z_i = f(x_i)$ for a batch of inputs. 2. **Compute losses:** Calculate the topology loss $\mathcal{L}_T$ using the batch's k-NN graph in input (or from a precomputed structure); compute $\mathcal{L}_T$ either by computing HSIC between $\{z_i\}$ and known nuisances or by computing environment-specific prediction losses if using IRM; compute $\mathcal{L}_E$ by evaluating $E(z_i)$ for positives and sampling some negatives z (via perturbation or an auxiliary network) to evaluate the energy model. 3. **Backward pass:** Backpropagate the weighted sum $\mathcal{L}_{\text{APS}}$ to update the encoder f (and decoder if present), as well as parameters of E (energy model) and any adversarial discriminators (for invariance) or prototypes. 4. **Prototype memory update (if used):** If using explicit prototypes or a memory bank, update them using clustering algorithms or by gradient (some methods treat prototypes as attractor centers updated via gradient or EM steps).

The training is thus multi-objective. Choosing the right weights $\lambda_T, \lambda_C, \lambda_E$ is important – too much topology preservation might hurt reconstruction if the model struggles to satisfy all neighbors; too strong invariance might remove useful information; too strong energy shaping might collapse all points to prototypes. In practice, a curriculum could help: e.g. first train an autoencoder for reconstruction, then gradually increase λ_T and λ_C to refine the latent geometry, and finally introduce λ_E to carve basins once the manifold is well-formed.

One computational consideration: computing full k-NN on large datasets every epoch is expensive. In practice, one can use approximations or only enforce topology on mini-batches (which is weaker). Alternatively, focus on preserving local structure via *local reconstruction* (as NRAE does) rather than explicit distance matrices. Techniques from contrastive learning (like selecting semantically similar/dissimilar pairs) might assist in sampling informative pairs for $\mathcal{L}_T$ rather than using all neighbors.

4.3 Theoretical Discussion

While APS is an applied framework, it touches on theoretical questions. For example, **does enforcing these constraints lead to a loss of information capacity?** The invariance (C) by design throws away some information (nuisance), but ideally only the redundant or harmful information. Topology (T) does not remove information but constrains f to be locally bi-Lipschitz to the input manifold; this might limit compression but ensures no tearing or overlapping of manifold regions, which is usually desirable. Energy (E) can be seen as adding a prior $p(z) \propto e^{-E(z)}$ that is multi-modal. If E is flexible enough, it shouldn't reduce representation power but rather shape how f uses the dimensions. There is also a question of **identifiability**: causal representation learning

literature notes that without inductive biases, disentangling true factors is ill-posed. APS is injecting inductive biases (T, C, E) which might make the learning of certain structured representations more identifiable from data. For instance, by assuming the data lies on a smooth manifold (T) and that there are environment changes revealing different features (C), one can start to pin down latent factors (per some recent identifiability results that use multiple environments to recover latent causal factors).

5 Experiments

We evaluate APS on a range of tasks to demonstrate its benefits in terms of representation quality, interpretability, and downstream performance. Our experiments cover three domains – **language, vision, and recommender systems** – reflecting the broad applicability of APS.

1. Qualitative Visualization of Pattern Atlases: First, we train APS as an autoencoder (or variational autoencoder) on a standard dataset in each domain: e.g. the **MNIST** image dataset, the **AG News** text dataset (news articles labeled by topic), and a **MovieLens** subset (user–movie interaction data for recommender). After training, we visualize the 2D or 3D projections of the learned latent space using UMAP[2]. We expect to see well-structured atlases: - For MNIST, the APS latent space should form clear clusters for each digit (0–9) due to energy basins, but with meaningful continuous transitions between them reflecting visual similarity (topology). For example, the latent map might show "1" and "7" clusters close (as they look similar), and style variations (stroke width, rotation) forming smooth directions within each cluster. A baseline VAE's latent might instead show overlapping digits or less separation. - For AG News (with topics like World, Sports, Business, Sci/Tech), APS should separate articles by topic (basins) while also forming sub-clusters or a continuum for related sub-topics (topology). Because of the causal loss, unrelated factors (e.g. article length or writing style) should not form the axes of variation. We can highlight example trajectories on the map (e.g. interpolating between a sports article and a business article) to see that APS transitions are smooth and stay within semantic boundaries (thanks to topology preservation). - For MovieLens, an APS embedding of users and movies might be used. We anticipate that similar users (with similar tastes) cluster together, and genre-wise clustering for movies emerges as energy basins. The topology constraint could preserve the continuous spectrum of genres or user preferences (e.g. action <-> thriller <-> horror might lie along a path). Traditional matrix factorization embeddings might not show such clear grouping without manual dimensionality reduction.

2. Quantitative Topology Preservation: We measure how well APS preserves neighborhood structure using metrics from manifold learning. One common metric is the **trustworthiness** and **continuity** of the embedding (which compare input-space neighbors vs latent neighbors). We compute trustworthiness for APS vs baseline embeddings (e.g. a plain autoencoder without these losses, or a PCA) at various neighborhood sizes. We expect APS to have higher trustworthiness, indicating it avoids "false neighbors" in latent space that weren't neighbors in input. Additionally, using the persistent homology approach from Topological Autoencoders, we compute the Betti numbers of the latent representation for a synthetic 3D Swiss-roll dataset and show APS preserves the one-dimensional hole structure if present, whereas a vanilla AE might not.

3. Invariance and Generalization Tests: To test the causal invariance aspect, we construct or use datasets with known spurious correlations: - **Colored MNIST (variant):** We create a version of MNIST where each digit is colored either red or blue, with a *biased correlation* between color and digit label in training (e.g. 90% of "0" are red, 90% of "1" are blue, etc., similar to IRM's setup). We train a classifier on top of embeddings from APS vs embeddings from a standard model.

At test time, we evaluate on a dataset where the color–label correlation is flipped (the spurious cue is misleading). The APS-based classifier should significantly outperform the baseline because APS's causal loss would encourage it to not encode color (we explicitly treat color as a nuisance and use HSIC to make Z independent of color). This mirrors results from IRM, but we will verify that even with a single environment but explicit penalization, APS can reduce reliance on color. - **Out-of-distribution (OOD) generalization:** For text, we might use an extremist example: train on sentiment analysis where all positive reviews about *electronics* are written by one author and all negative by another. A model might spuriously associate author with sentiment. We simulate environment labels as author IDs and use an IRM loss in APS. We then test on a new author. We measure accuracy drop OOD. APS's invariance should yield a smaller drop than a normal model. - We also test zero-shot generalization qualitatively: with APS energy basins, the model may handle novel combinations better. For instance, if we have seen "red square" and "blue circle" during training, can the APS latent represent "red circle" reasonably? If z(red circle) lies in the basin near other "circle" patterns (because shape is a causal factor separated from color), then decoding it yields a plausible red circle, illustrating combinatorial generalization.

4. Memory and Prototype Utility: We examine how well the energy basins correspond to meaningful categories and how they can be used for tasks: - **Clustering Accuracy:** If we treat each energy basin (e.g. each learned prototype or each mode of the latent EBM) as representing a class, we can cluster the latent vectors by assigning them to the nearest basin minimum. We compare this clustering with ground-truth labels. For example, cluster APS embeddings of CIFAR-10 images and measure clustering purity relative to the 10 classes. APS is not explicitly given class labels, but if it forms energy basins that align with classes, purity will be high (approaching supervised methods). Baselines like a β-VAE or standard AE likely yield lower purity as they don't explicitly cluster by class. - **Few-shot classification:** Using the learned prototypes as class representatives, we perform a k-NN or prototype-based classification on a new small labeled dataset. E.g., embed few labeled examples of new classes into APS space and classify test examples by nearest prototype. We expect APS's structured space to give a boost due to more distinct separation between classes (Energy) and more relevant features (Causality). This experiment would highlight the benefit of APS in low-data regimes, akin to how prototypical networks work but here the entire space is shaped prototypically. - **Memory recall:** We demonstrate the attractor property by adding noise to a latent vector and feeding it through a Hopfield update (if implemented) or gradient descent on $E(z)$. We show that it converges to the same basin, effectively denoising the latent. Quantitatively, adding noise of certain magnitude to test image embeddings, then reconstructing images: APS should tolerate more noise (recover original class) than a regular autoencoder, because the energy landscape guides it back. We measure reconstruction error or class consistency as a function of noise.

5. Ablation Studies: We train variants of APS turning off one component at a time (only T+E, only C+E, etc.) to isolate their contributions. For each ablation, we evaluate a relevant metric: - Remove T: Measure trustworthiness (it should drop if T is removed). - Remove C: Measure OOD generalization gap (it should widen without invariance). - Remove E: Measure clustering purity or visualization clarity (likely latent space becomes more entangled). - Remove none (full APS): see the best combined outcome. This validates that each part of APS is necessary for the full benefits.

6. Computational Considerations: We report training time and convergence behavior. APS adds overhead (neighbor graph computations, MCMC for EBM, etc.). We note if training remains stable. Perhaps we find that the topology loss actually accelerates convergence by providing additional signal (as seen in some metric learning contexts). We also monitor if any component adversely affects reconstruction (e.g. too high λ_E might increase reconstruction loss; we adjust to find a good trade-off).

6 Discussion and Conclusion

We presented **Atlasing Pattern Space (APS)**, a framework that integrates principles from manifold learning, causal inference, and energy-based modeling to produce **structured, interpretable latent spaces** for LLMs and other high-dimensional models. APS can be seen as injecting domain-agnostic inductive biases (local similarity preservation, invariance to spurious factors, and multi-modal clustering) that make learned representations more aligned with the true data-generating factors and more useful for downstream purposes. By evaluating on multiple domains, we demonstrated that APS yields latent maps where **neighborhoods are meaningful (T)**, **axes align with stable concepts (C)**, and **clusters correspond to human-recognizable categories (E)**. These properties improve both performance (especially in transfer learning and robustness scenarios) and explainability (researchers can visualize and understand what the model has learned).

Impact: For the general ML community, APS offers a blueprint for **geometric deep learning** in the latent space – moving beyond unstructured vector spaces to *spaces with topology and geometry tailored to the problem*. This resonates with the trend of applying **differentiable constraints** (e.g. using TDA or adversarial objectives) to ensure our models learn what we intend. APS specifically could benefit LLMs by providing them with an internal semantic atlas, potentially enabling better control (steering the model towards certain regions yields certain types of generations) and more predictable behavior. Similarly, in recommendation or personalization systems, an APS embedding could help identify coherent user segments or item categories through the energy basins, improving transparency and fairness (as causal factors like demographic correlations could be explicitly controlled in Z).

Limitations and Future Work: Our initial APS implementation introduces several hyperparameters (the weights λ's, the choice of k, etc.) which require tuning. In some cases, there may be trade-offs between the objectives – e.g. perfect topology preservation might conflict with perfect invariance if certain spurious features were part of local similarity in data. Balancing these is non-trivial. Additionally, the current formulation assumes we can either know or infer nuisance factors for the causality loss; in truly unsupervised scenarios, one might use data augmentations as a proxy (assuming certain transformations shouldn't change Z). This could be further automated by techniques that learn what to ignore (perhaps using attention mechanisms to attend to causal features). Another limitation is scalability: for extremely large datasets, computing even approximate neighbor graphs is heavy – one might explore *self-supervised contrastive approaches* to approximate the topology loss (e.g. treat augmented pairs as neighbors). The energy component might risk mode collapse if not handled carefully (training EBMs is known to be tricky), so more robust methods like noise contrastive estimation could be tried.

For future research, one exciting avenue is to extend APS to **different geometries** (not just Euclidean latent spaces). For instance, we could enforce topology and invariance while learning embeddings on a **hyperbolic manifold** for inherently hierarchical data – combining APS with the Poincaré embedding approach[16]. Another direction is to incorporate **dynamic or temporal pattern spaces** – e.g. use APS for sequence models where the latent at each time step forms an atlas of states (this might connect with state-space models or neural ODEs that have attractors[14][15]). We also plan to investigate theoretical guarantees: under what conditions does minimizing these losses recover the true generative factors or the true manifold? Insights from recent identifiability theory could guide this.

In conclusion, Atlasing Pattern Space represents a step toward *geometry-aware, causally-informed representation learning*. By unifying ideas across subfields (topological deep learning, causal ML, energy-based memory networks[6]), it provides a framework for building **latent spaces that are**

as structured and rich as the data they model. We hope this sparks further exploration into **structured embeddings** that can ultimately lead to more generalizable and interpretable AI systems.

7 References (Selected)

- Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D. (2019). *Invariant Risk Minimization.* **arXiv:1907.02893**. – Introduces IRM to learn invariant predictors across environments, relating invariances to causal structures for better OOD generalization.
- Moor, M. et al. (2020). *Topological Autoencoders.* **ICML 2020**. – Uses persistent homology to align the topology of latent space with data, preserving connectivity and improving interpretability of embeddings.
- Chen, N., van der Smagt, P., Cseke, B. (2022). *Local Distance Preserving Auto-encoders using Continuous k-Nearest Neighbours Graphs.* **arXiv:2206.05909**. – Proposes autoencoders with a continuous k-NN graph loss to preserve local distances (and thus topology) in the latent space, yielding geometrically consistent representations.
- Lee, Y. et al. (2021). *Neighborhood Reconstructing Autoencoders.* **NeurIPS 2021**. – Regularizes autoencoders with neighborhood graph information and local quadratic reconstruction to fix local geometry errors in learned manifolds.
- Schönenberger, S. et al. (2020). *Witness Autoencoder: Shaping the Latent Space with Witness Complexes.* **arXiv Preprint**. – Utilizes topological "witness complexes" to impose structure on the latent space of an autoencoder.
- Duque, A. et al. (2020). *Geometry Regularized Autoencoders.* **arXiv:2007.07142**. – Introduces an approach to learn invertible manifold embeddings with explicit geometry regularization, preserving distances and structure.
- Chen, X. et al. (2021). *Neighborhood Geometric Structure-Preserving VAE.* **IEEE TNNLS**. – Ensures a VAE's latent space respects the neighborhood structure of data, improving smoothness and interpolations.
- van der Maaten, L., Hinton, G. (2008). *Visualizing Data using t-SNE.* **JMLR 9(86)**[1]. – Classic technique for mapping high-D data to 2D/3D by preserving local neighbor probabilities, revealing multi-scale structure in datasets.
- McInnes, L., Healy, J., Melville, J. (2018). *UMAP: Uniform Manifold Approximation and Projection.* **arXiv:1802.03426**[2]. – Manifold learning method based on Riemannian geometry and algebraic topology, producing embeddings that maintain both local and some global structure with high efficiency.
- Greenfeld, D., Shalit, U. (2020). *Robust Learning with the Hilbert-Schmidt Independence Criterion.* **ICML 2020**. – Uses HSIC as a loss to encourage independence between model residuals and inputs, thereby learning models robust to covariate shift (related to causal objectives).
- Ma, W.D. et al. (2019). *The HSIC Bottleneck: Deep Learning without Backpropagation.* **ICLR 2019**. – Proposes training deep nets by maximizing HSIC between layers and targets while minimizing HSIC with inputs, achieving an information bottleneck and invariance without traditional backprop.
- Schölkopf, B. et al. (2021). *Toward Causal Representation Learning.* **Proceedings of the IEEE, 109(5)**. – A comprehensive survey linking causal inference concepts with representation learning challenges, advocating for learning representations that reflect causal generative factors.

- Higgins, I. et al. (2017). *β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework*. **ICLR 2017**[3]. – Introduces β-VAE, which constrains the VAE latent capacity to learn disentangled and interpretable latent factors (each capturing a distinct concept in data).
- Nickel, M., Kiela, D. (2017). *Poincaré Embeddings for Learning Hierarchical Representations*. **NeurIPS 2017**[16]. – Learns embeddings in hyperbolic space to encode hierarchical relationships parsimoniously, preserving both similarity and latent hierarchy much better than Euclidean embeddings.
- Rong, Y. et al. (2020). *The Latent Space Energy-Based Model*. **NeurIPS 2020**[8][9]. – (Refers to Pang et al.) Demonstrates learning an energy-based prior in the latent space of a generator, which improves generative modeling by capturing data regularities via an EBM in latent.
- Ramsauer, H. et al. (2021). *Hopfield Networks is All You Need*. **ICLR 2021**[5][7]. – Shows equivalence between Hopfield network updates and Transformer attention, and introduces modern Hopfield layers with high capacity. Discusses how such layers can be integrated to provide memory and associative retrieval in deep networks, with energy minima corresponding to stored patterns.
- ***Additional references omitted for brevity.*** (Includes works on attractor dynamics in neural ODEs[12][15], contrastive representation learning, and other structured embedding approaches.)