Chapter 1

Background

1.1. Weight Space Learning

Weight space learning is a field within machine learning that focuses on understanding and leveraging the structure of the neural network weight space. The central aim is to model how network parameters are shaped by data, architecture, and training dynamics, and to capture these relationships within a learnable representation.

At its core, weight space learning seeks to construct *meta-models*—models that learn from other models. Unlike standard machine learning models that capture patterns in data, meta-models capture patterns in the *weights* of networks trained on that data. In this way, the goal shifts from learning a direct input–output mapping to learning the structure that governs how such mappings are formed.

Due to the enormous scale and dimensionality of modern neural networks [], it is typically infeasible to operate directly on raw model weights. Furthermore, redundancy is well known to exist in neural networks; smaller architectures can often achieve comparable performance to larger ones []. These challenges motivate one of the central subproblems of weight space learning: the discovery of low-dimensional representations of weight space.

A *latent representation* of weight space provides a compact and structured encoding of a model's parameters. The transformations—linear or non-linear—that map weights into this latent space are learned to preserve the essential information required to reconstruct or analyse the original weights. Among the many dimensionality reduction techniques available, a useful distinction can be made between *reversible* and *non-reversible* methods.

Reversibility is of particular importance in weight space learning. While encoding weights into a latent representation (real $\rightarrow$ latent) is informative, the ability to reconstruct the original weights (real $\rightarrow$ latent $\rightarrow$ real) is far more valuable. This reversibility enables the synthesis of entirely new weight configurations, supporting generative applications such as zero-shot model creation and performance-guided model generation. Consequently, reversible latent representation methods are the most prevalent within weight space learning, especially for

encoding and decoding neural network weights.

This chapter proceeds as follows. Section 1.2 introduces Principal Component Analysis (PCA), a linear and probabilistic approach that provides a simple yet effective reversible dimensionality reduction method. Section 1.3 discusses reversible, non-linear approaches, focusing on autoregressive encoder architectures and presenting the Sequential Autoencoder for Neural Embeddings (SANE) []. Finally, Section 1.4 explores a non-reversible, modality-heterogeneous technique, contrastive learning, including a description of the NT-Xent loss and its implementation in CLIP [].

1.2. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a linear dimensionality reduction technique used to represent high-dimensional data in a lower-dimensional form while retaining as much variance as possible. It provides a compact latent representation that captures the most informative directions of variation in the data. This section outlines the motivation behind PCA, its mathematical formulation, and its relevance and limitations as a latent representation method.

Problem Setup and Intuition. Consider a dataset $X \in \mathbb{R}^{n \times d}$ with n samples and d features, centred such that each feature has zero mean. The goal of PCA is to find a new coordinate system whose axes are linear transformations of the original dimensions, ordered by the amount of variance they capture. Orthogonality between these axes ensures that each captures unique, non-redundant information about the data.

Intuitively, PCA identifies the directions that best describe the "shape" or spread of the data cloud in feature space. Projecting the data onto the top k directions provides a compressed representation that preserves most of the information while discarding redundancy.

Mathematical Formulation. PCA seeks a linear projection matrix $W \in \mathbb{R}^{d \times k}$ that maps the data to a lower-dimensional space:

$$Z = XW,$$

where $Z \in \mathbb{R}^{n \times k}$ represents the latent representation. The sample covariance matrix of X is

$$\Sigma = \frac{1}{n} X^\top X,$$

and the total variance captured by the projection is

$$\text{Var}(Z) = \text{Tr}(W^\top \Sigma W),$$

where the trace operator $\text{Tr}(\cdot)$ sums the variances along all projected directions.

PCA thus maximises the variance of the projected data:

$$\max_{W} \text{Tr}(W^\top \Sigma W) \quad \text{subject to} \quad W^\top W = I_k.$$

The constraint enforces orthonormality among the new axes so that each captures distinct variance. Solving this optimisation leads to the eigenvalue problem:

$$\Sigma W = W\Lambda,$$

where the columns of W are the eigenvectors of Σ, and the diagonal entries of Λ are the corresponding eigenvalues that quantify the variance explained by each principal component. The top k eigenvectors define the optimal projection directions.

Latent Representation and Limitations. The resulting embedding,

$$Z = XW_k,$$

is the *latent representation* of the data, capturing the dominant linear structure of the dataset. PCA provides a reversible mapping that probabilistically preserves the greatest possible amount of variance under a linear projection.

However, its linear nature limits its ability to capture nonlinear relationships when the data lie on curved manifolds. Furthermore, PCA does not adapt to unseen data that deviate from the original subspace—it yields a fixed, non-learnable transformation.

Incremental PCA. In practice, the covariance matrix Σ can be prohibitively large for high-dimensional data, such as neural network weights. Incremental PCA (IPCA) addresses this by processing data in smaller batches and updating the principal components iteratively. Rather than recomputing the full covariance, it approximates it using partial updates and truncated singular value decompositions (SVD). This approach makes IPCA suitable for large-scale or streaming datasets while maintaining results comparable to standard PCA. However, it remains an approximation and inherits PCA's linear and non-generalisable nature.

1.3. Autoencoder

1.3.1. Sequential Autoencoder for Neural Embeddings

1.4. Contrastive Learning

1.4.1. NT-Xent Loss

1.4.2. CLIP

Bibliography