Physalia MLOmics Barcelona 2025 - Day 3 Notes

Deep Learning for Data Integration - Final Day

Course Participant

December 17, 2024

Contents

1 Day 3 Notes **2**
1.1 Course Start . . . 2
1.2 Key Observations . . . 2
1.2.1 RNA-seq Data Patterns . . . 2
1.2.2 Data Preprocessing . . . 3
1.2.3 Framework Comparison . . . 3
1.3 Deep Learning Practical Insights . . . 3
1.3.1 Hyperparameter Optimization . . . 3
1.3.2 Regularization Strategies . . . 3
1.4 Dimensionality Reduction and UMAP for Omics Integration . . . 3
1.4.1 t-SNE vs PCA Comparison . . . 3
1.4.2 Purpose Beyond Visualization . . . 3
1.4.3 Linear vs Nonlinear Methods . . . 3
1.4.4 Mathematical Foundations . . . 3
1.5 PCA Implementation Steps . . . 4
1.5.1 Step-by-Step PCA Algorithm . . . 4
1.5.2 Mathematical Notation: . . . 4
1.5.3 Estimating Optimal Number of PCs . . . 4
1.5.4 PCA Limitations in Single Cell Analysis . . . 4
1.5.5 Complexity and Method Performance . . . 4
1.5.6 UMAP Technical Details . . . 4
1.5.7 UMAP for Data Integration . . . 4
1.5.8 Why to Do Dimensionality Reduction? . . . 4
1.6 Curse of Dimensionality: Three Examples . . . 5
1.6.1 Example 1: Low Dimensional (n=20, p=2) . . . 5
1.6.2 Example 2: High Dimensional (n=20, p=10) . . . 5
1.6.3 Example 3: Extreme Case (n=20, p=20) . . . 5
1.6.4 Analysis Summary . . . 5
1.7 Practical Example: PCA vs MDS on MNIST Data . . . 5
1.7.1 MNIST Dataset Overview . . . 5
1.7.2 PCA and MDS Implementation Approach . . . 5
1.8 t-distributed Stochastic Neighbor Embedding (t-SNE) . . . 6
1.8.1 When Linear Methods Are Not Enough . . . 6
1.8.2 Non-linear Dimensionality Reduction Techniques . . . 6
1.8.3 How t-SNE Works . . . 6
1.8.4 Mathematical Foundation . . . 6
1.8.5 When to Use t-SNE vs Linear Methods . . . 6
1.8.6 Important Cautions About t-SNE . . . 6

1.8.7 Key Differences: Linear vs Non-linear Methods . . . 6
1.9 UMAP vs t-SNE: Key Advantages . . . 7
1.9.1 Why UMAP is Often Preferred . . . 7
1.9.2 Comparison: UMAP vs t-SNE . . . 7
1.9.3 UMAP Technical Advantages . . . 7
1.9.4 When to Choose Which Method . . . 7

2 Principles of Batch Correction (ComBat-like) 8
2.1 Understanding Batch Effects . . . 8
2.1.1 What are Batch Effects? . . . 8
2.1.2 ComBat Method Overview . . . 8
2.2 Mutual Nearest Neighbors (MNN) . . . 8
2.2.1 Core Concept . . . 8
2.2.2 MNN Algorithm Steps . . . 8
2.2.3 Key Advantages of MNN . . . 8
2.2.4 When to Use MNN vs ComBat . . . 8
2.2.5 MNN Implementation Notes . . . 8
2.3 Integration Strategy Summary . . . 8
2.3.1 Choosing the Right Approach . . . 8
2.4 Advanced Single-Cell Integration Methods . . . 9
2.4.1 Seurat CCA + Dynamic Time Warping Integration . . . 9
2.4.2 Harmony Integration Method . . . 9
2.4.3 Scanorama Integration Method . . . 9
2.4.4 Method Comparison Summary . . . 9
2.4.5 Evidence-Based Method Selection . . . 10
2.5 Vertical Integration: Cross-Omics Methods . . . 10
2.5.1 Seurat Transfer Learning Approach . . . 10
2.5.2 Workflow for Cross-Omics Integration: . . . 10
2.6 Course Summary: Key Takeaways . . . 10
2.6.1 Day 3 Learning Objectives Achieved . . . 10
2.6.2 Future Directions . . . 11

1 Day 3 Notes

1.1 Course Start

Started the last day with a quick recap of Lab about autoencoder.

1.2 Key Observations

1.2.1 RNA-seq Data Patterns

One important note: when looking RNA-seq data, it turns out there is less difference in same tissue between two individuals than different tissues within the individual.

Supporting References: - GTEx Consortium (2013). The Genotype-Tissue Expression (GTEx) project. *Nature Genetics*, 45(6), 580-585. - GTEx Consortium (2015). Human genomics. The Genotype-Tissue Expression (GTEx) pilot analysis: multitissue gene regulation in humans. *Science*, 348(6235), 648-660. - GTEx Consortium (2017). Genetic effects on gene expression across human tissues. *Nature*, 550(7675), 204-213. - Mele, M., et al. (2015). Human genomics. The human transcriptome across tissues and individuals. *Science*, 348(6235), 660-665.

Note: This observation aligns with similar patterns observed in our previous analyses

1.2.2 Data Preprocessing

Simple log transformation is enough, and no normalization required for single cell data.

1.2.3 Framework Comparison

Keras better than tensorflow for neural network. tensorflow is more complicated?

1.3 Deep Learning Practical Insights

1.3.1 Hyperparameter Optimization

Mastering hyperparameter tuning takes time and experience.

1.3.2 Regularization Strategies

Less sample size requires dropout regularization to prevent overfitting.

1.4 Dimensionality Reduction and UMAP for Omics Integration

1.4.1 t-SNE vs PCA Comparison

t-SNE vs PCA: t-SNE provides more clear grouping and separation of clusters.

1.4.2 Purpose Beyond Visualization

Dimensionality reduction is not just a visualization tool but an important data preprocessing step.

1.4.3 Linear vs Nonlinear Methods

Linear Methods: NMF, MMDS, NMDS, SVD, SPCA, LDA, PCA, ICA, FA

Brief Descriptions: - **PCA** (Principal Component Analysis): Finds orthogonal components that maximize variance in the data - **ICA** (Independent Component Analysis): Separates data into statistically independent components - **LDA** (Linear Discriminant Analysis): Supervised method that finds components maximizing class separation - **NMF** (Non-negative Matrix Factorization): Decomposes data into non-negative factors, useful for parts-based representation - **SVD** (Singular Value Decomposition): Mathematical foundation for PCA, decomposes matrix into three matrices - **SPCA** (Sparse PCA): PCA variant that enforces sparsity in components for better interpretability - **FA** (Factor Analysis): Models observed variables as linear combinations of latent factors - **MMDS** (Metric Multidimensional Scaling): Preserves pairwise distances when reducing dimensions - **NMDS** (Non-metric Multidimensional Scaling): Preserves rank order of distances rather than exact distances

1.4.4 Mathematical Foundations

Linear Dimensionality Reduction: Based on matrix factorization approaches

Nonlinear Dimensionality Reduction: Based on neighborhood graph construction

1.4.4.1 Nonlinear Method Algorithm:

1. Construct high-dimensional graph with probabilities **p__ij**
2. Construct low-dimensional graph with probabilities **q__ij**
3. Minimize difference between graphs using **Kullback-Leibler divergence**

This approach (used in t-SNE) preserves local neighborhood structure while reducing dimensions

1.5 PCA Implementation Steps

1.5.1 Step-by-Step PCA Algorithm

Algorithm Overview: 1. **Center the data**: Subtract mean from each feature 2. **Calculate covariance matrix**: Compute relationships between features
3. **Eigendecomposition**: Find principal components and their importance 4. **Visualize components**: Plot first two principal components 5. **Plot explained variance**: Show how much variance each component captures

1.5.2 Mathematical Notation:

- **M = X -** : Centered data matrix (where is the mean of feature j)
- **A = (1/N) × M^T × M**: Covariance matrix
- **A × u = × u**: Eigenvalue equation (= eigenvalues, u = eigenvectors)

1.5.3 Estimating Optimal Number of PCs

Seurat Method: Jackstraw procedure - Statistical test to determine significant principal components - Compares observed eigenvalues to those from random data - Helps identify meaningful components vs. noise

1.5.4 PCA Limitations in Single Cell Analysis

Key Limitation: PCA alone is often insufficient for single cell data analysis due to: - High sparsity and noise in single cell data - Complex non-linear biological relationships - Need for specialized normalization and scaling

1.5.5 Complexity and Method Performance

Simple Cases (3 classes): - Both PCA and t-SNE tell similar stories - Linear relationships can be captured adequately

Complex Cases (10+ classes): - t-SNE becomes more informative than PCA - Non-linear structure requires advanced methods - UMAP often superior to both

1.5.6 UMAP Technical Details

Key Differences from t-SNE: - Uses **local connectivity** for high-dimensional probabilities - **Does not normalize probabilities** (computational speedup) - Can deliver **multiple components** for clustering downstream analysis - Uses **Laplacian eigenmap** for initialization (vs. random in t-SNE) - Uses **cross-entropy** (not Kullback-Leibler divergence) as cost function

1.5.7 UMAP for Data Integration

Graph Intersection Method: - Constructs separate neighborhood graphs for each omics layer - Finds intersections between graphs to identify shared structure - Enables integration while preserving layer-specific information

1.5.8 Why to Do Dimensionality Reduction?

Dimensionality Reduction concept is really not just about visualization like many of use might think. This is a necessity in Data Science in order to overcome the Curse of Dimensionality, also known as Rao's paradox. What is it about? When we work with data we have n observations (samples) for p variables (features). Very often (almost always unless you are lucky) we have p»n, i.e. we have a highly dimensional space. It turns out that the classical Frequentist statistics blows up in a highly-dimensional space, i.e. the conclusions of the models are not valid (robust) any more.

1.6 Curse of Dimensionality: Three Examples

1.6.1 Example 1: Low Dimensional (n=20, p=2)

- **Normal behavior**: No spurious relationships
- **R-squared**: 0.022 (low, appropriate)
- **Result**: Statistics work correctly

1.6.2 Example 2: High Dimensional (n=20, p=10)

- **Overfitting begins**: Some "significant" false discoveries
- **R-squared**: 0.659 (inflated due to overfitting)
- **Result**: Model appears better but fits noise

1.6.3 Example 3: Extreme Case (n=20, p=20)

- **Complete breakdown**: n = p (no degrees of freedom)
- **R-squared**: 1.0 (perfect fit to noise!)
- **Residuals**: ALL residuals = 0
- **Standard errors**: All NaN (undefined)
- **Result**: **Perfect interpolation of random noise**

Key Insight: When p >= n, you can fit any data perfectly, but the model is completely meaningless. This is why dimensionality reduction is essential in omics where p »> n.

1.6.4 Analysis Summary

- **Scenario 1**: Correct behavior - no spurious relationships found
- **Scenario 2**: False discoveries appear due to overfitting
- **Scenario 3**: Complete mathematical breakdown - perfect fit to pure noise!

1.7 Practical Example: PCA vs MDS on MNIST Data

1.7.1 MNIST Dataset Overview

The MNIST database (Modified National Institute of Standards and Technology) contains handwritten digits (0-9) with: - **784 features** (28x28 pixel values) - **10,000+ samples** (handwritten digit images) - Each pixel intensity ranges from 0-255

1.7.2 PCA and MDS Implementation Approach

Key Steps for Analysis: 1. Load MNIST data and separate features from labels 2. Take subset for computational efficiency (e.g., 1000 samples) 3. Standardize the pixel data 4. Compute PCA using principal component analysis 5. Compute MDS using classical multidimensional scaling 6. Compare the 2D representations and explained variance

Expected Results and Interpretation:

Key Observation: MDS gives quite similar 2D representation to PCA, and this is not surprising when considering the relationship between Euclidean distance and variance-covariance matrix.

Why PCA ~ MDS for this data: 1. **Mathematical relationship**: Classical MDS with Euclidean distances is equivalent to PCA 2. **Distance preservation**: Both methods aim to preserve the most important relationships 3. **Linear transformation**: Both use linear dimensionality reduction

Differences you might observe: - **Orientation**: MDS plots may be rotated/flipped compared to PCA - **Scale**: Axes may have different scales - **Clustering**: Similar digit separation in both methods

Biological relevance: This demonstrates why both PCA and MDS are effective for high-dimensional omics data where Euclidean relationships are meaningful.

1.8 t-distributed Stochastic Neighbor Embedding (t-SNE)

1.8.1 When Linear Methods Are Not Enough

PCA or MDS make sense when we suspect **linear relations** between the variables in X. However, sometimes correlation between two variables can be zero - does this mean that the two variables are not related?

No, it does not! The relationship can be **non-linear**: - Quadratic relationships - Logarithmic relationships - Sinusoidal relationships - Other complex non-linear patterns

1.8.2 Non-linear Dimensionality Reduction Techniques

To discover non-linear relationships between observations, we use non-linear dimensionality reduction techniques: - **t-SNE** (t-distributed Stochastic Neighbor Embedding) - **Isomaps** - **LLE** (Locally Linear Embedding) - **Self-Organizing Maps**

Among these, **t-SNE is especially popular** in many Data Science areas due to its interesting visualization properties.

1.8.3 How t-SNE Works

Core Principle: t-SNE projects high-dimensional data into low-dimensional space such that points that are **close/far in high-dimensional space** are also **close/far in low-dimensional space**.

Key Innovation: t-SNE has its special way to measure similarity in both high- and low-dimensional spaces, using the **Gaussian law** for probability distributions.

1.8.4 Mathematical Foundation

High-dimensional similarities: Uses Gaussian distribution to compute probabilities **p__ij**

Low-dimensional similarities: Uses t-distribution to compute probabilities **q__ij**

Optimization: Minimizes Kullback-Leibler divergence between p__ij and q__ij distributions

This approach preserves local neighborhood structure while allowing for non-linear relationships

1.8.5 When to Use t-SNE vs Linear Methods

Thus t-SNE is handy when it concerns **non-linear relations** between data points which **cannot be captured by PCA or MDS**.

1.8.6 Important Cautions About t-SNE

Distance Interpretation Warning: Due to its highly non-linear nature, the **visual distances on the t-SNE plot do not necessarily reflect the true distances** in the high-dimensional space. In other words, it is hard to say with certainty how far or how close two clusters on the t-SNE plot are since **t-SNE distances do not have a trivial meaning**.

Feature Interpretability Limitation: Another consequence of the non-linear transformation is that the **features that drive the clustering** on the t-SNE plot are **not easy to extract** since we are not doing any linear matrix decomposition as with PCA.

1.8.7 Key Differences: Linear vs Non-linear Methods

Method	Distance Meaning	Feature Extraction	Use Case
PCA/MDS	Distances preserved	Easy (loadings)	Linear relationships

Method	Distance Meaning	Feature Extraction	Use Case
t-SNE	Distances distorted	Difficult	Non-linear relationships

Practical Implication: Use t-SNE for **visualization and cluster discovery**, but use PCA/MDS for **quantitative distance analysis** and **feature interpretation**.

1.9 UMAP vs t-SNE: Key Advantages

1.9.1 Why UMAP is Often Preferred

UMAP provides more condensed embeddings and the **distances between clusters are more meaningful** compared to t-SNE.

1.9.2 Comparison: UMAP vs t-SNE

Aspect	t-SNE	UMAP
Embedding Quality	Good local structure	More condensed, preserves both local and global structure
Distance Interpretation	Distances distorted/meaningless	Distances between clusters more meaningful
Speed	Slower	Faster (doesn't normalize probabilities)
Components	Typically 2D	Can deliver multiple components for clustering
Initialization	Random	Uses Laplacian eigenmap
Cost Function	Kullback-Leibler divergence	Cross-entropy
Global Structure	Poor preservation	Better preservation

1.9.3 UMAP Technical Advantages

From earlier notes on UMAP characteristics: - Uses **local connectivity** for high-dimensional probabilities - **Does not normalize probabilities** (computational speedup) - Can deliver **multiple components** for downstream clustering - Uses **Laplacian eigenmap** for initialization - Uses **cross-entropy** (not KL divergence) as cost function

1.9.4 When to Choose Which Method

- **Use UMAP** when you need:
  - Meaningful distance interpretation between clusters
  - Faster computation
  - Multiple components for further analysis
  - Better global structure preservation
- **Use t-SNE** when you need:
  - Focus purely on local neighborhood visualization
  - Well-established, widely-used method
  - Maximum separation of distinct clusters (may sacrifice distance meaning)

MDS = machine learning, PCA = linear algebra

2 Principles of Batch Correction (ComBat-like)

2.1 Understanding Batch Effects

2.1.1 What are Batch Effects?

- **Technical variation**: Non-biological differences between experimental batches
- **Common sources**: Different processing dates, laboratory conditions, reagent lots
- **Impact**: Can confound biological signal and lead to false discoveries

2.1.2 ComBat Method Overview

- **Statistical approach**: Empirical Bayes method for batch correction
- **Assumptions**: Batch effects follow parametric distributions
- **Application**: Widely used for microarray and RNA-seq data

2.2 Mutual Nearest Neighbors (MNN)

2.2.1 Core Concept

Mutual Nearest Neighbors (MNN) is a method for batch correction and data integration that identifies cells that are each other's nearest neighbors across different batches.

2.2.2 MNN Algorithm Steps

1. **Identify MNN pairs**: Find cells that are mutual nearest neighbors between batches
2. **Calculate correction vectors**: Compute differences between MNN pairs
3. **Smooth correction vectors**: Apply local smoothing to avoid overcorrection
4. **Apply corrections**: Transform batch coordinates to align datasets

2.2.3 Key Advantages of MNN

- **Preserves biological variation**: Only corrects technical batch effects
- **Robust to population differences**: Works when batches have different cell type compositions
- **Single-cell friendly**: Designed specifically for sparse single-cell data
- **Reference-free**: Does not require a reference batch

2.2.4 When to Use MNN vs ComBat

Method	Best Use Case	Data Type	Assumptions
ComBat	Bulk omics data	Dense matrices	Parametric batch effects
MNN	Single-cell data	Sparse matrices	Non-parametric, preserves cell diversity

2.2.5 MNN Implementation Notes

- **Distance metric**: Typically uses cosine distance for high-dimensional data
- **Neighborhood size**: Critical parameter - affects correction strength
- **Batch order**: Can process multiple batches sequentially
- **Quality control**: Check that biological structure is preserved after correction

2.3 Integration Strategy Summary

2.3.1 Choosing the Right Approach

For Dimensionality Reduction: - Use **PCA** for linear relationships and interpretability - Use **t-SNE** for visualization of complex clusters - Use **UMAP** for both visualization and downstream analysis

For Batch Correction: - Use **ComBat** for bulk omics data with known batch structure - Use **MNN** for single-cell data with unknown population differences

For Multi-omics Integration: - Combine dimensionality reduction with batch correction - Consider UMAP's graph intersection method for multiple omics layers - Validate biological interpretation after technical correction

2.4 Advanced Single-Cell Integration Methods

2.4.1 Seurat CCA + Dynamic Time Warping Integration

Reference: Butler et al. (2018), "Integrating single-cell transcriptomic data across different conditions, technologies, and species", *Nature Biotechnology*

Key Features: - **Canonical Correlation Analysis (CCA)**: Identifies shared correlation structures between datasets - **Dynamic Time Warping (DTW)**: Aligns datasets by finding optimal matching between cells - **Multi-modal integration**: Can handle different data types and experimental conditions

Applications: - Cross-species comparisons - Technology platform integration - Temporal data alignment

2.4.2 Harmony Integration Method

Reference: Korsunsky et al. (2019), "Fast, sensitive and accurate integration of single-cell data with Harmony", *Nature Methods*

Core Principle: - **Iterative clustering approach**: All batches contribute to cluster formation where possible - **Soft clustering**: Cells can belong to multiple clusters with different probabilities - **Batch-aware correction**: Removes batch effects while preserving biological variation

Key Advantages: - Fast computational performance - Handles large datasets efficiently - Preserves global structure - Works with multiple batch variables simultaneously

Algorithm Steps: 1. Initial clustering of all cells 2. Compute batch-specific centroids 3. Apply correction vectors to remove batch effects 4. Repeat until convergence

2.4.3 Scanorama Integration Method

Reference: Hie et al. (2019), "Efficient integration of heterogeneous single-cell transcriptomes using Scanorama", *Nature Biotechnology*

Innovation: - **All-to-all dataset matching**: Generalization of MNN to many datasets - **No shared cell type requirement**: Does not require at least one cell type shared across datasets - **Panoramic stitching approach**: Inspired by image stitching algorithms

Key Features: - **Mutual nearest neighbor generalization**: Extends MNN concept beyond pairwise comparisons - **Robust to dataset heterogeneity**: Handles datasets with completely different cell type compositions - **Scalable integration**: Can integrate dozens of datasets simultaneously

Workflow: 1. Identify correspondences between all dataset pairs 2. Build integration graph connecting datasets 3. Apply batch correction using identified correspondences 4. Merge datasets into unified embedding

2.4.4 Method Comparison Summary

Method	Best For	Key Strength	Limitations
Seurat CCA	Multi-condition, cross-species	Dynamic time warping alignment	Requires shared biology
Harmony	Large-scale integration	Speed and soft clustering	May over-correct rare populations

Method	Best For	Key Strength	Limitations
Scanorama	Heterogeneous datasets	No shared cell type requirement	Computational complexity

Integration Strategy Recommendations: - Use **Seurat CCA** for temporal or developmental datasets - Use **Harmony** for large-scale, well-characterized datasets - Use **Scanorama** for highly heterogeneous collections of datasets

2.4.5 Evidence-Based Method Selection

Complex Datasets: Based on benchmarking studies, **Harmony** and **Scanorama** are preferred for: - Large-scale multi-batch datasets - High cellular complexity - Multiple experimental conditions - Heterogeneous populations

Simple Datasets: **ComBat** remains effective for: - Low complexity data - Few distinct populations - Well-controlled batch structure - Bulk omics data

2.5 Vertical Integration: Cross-Omics Methods

2.5.1 Seurat Transfer Learning Approach

Concept: **Transfer anchors across omics** - enables integration of different data modalities (RNA, ATAC, protein, etc.)

Key Features: - **Cross-modal anchors**: Identifies shared cellular states across omics layers - **Reference-query framework**: Uses well-characterized reference to annotate query data - **Multi-modal integration**: Combines transcriptomics, epigenomics, and proteomics

Applications: - Single-cell multiome analysis (RNA + ATAC) - CITE-seq integration (RNA + protein) - Cross-platform data transfer - Cell type annotation transfer

2.5.2 Workflow for Cross-Omics Integration:

1. **Identify integration anchors** between omics modalities
2. **Transfer information** from reference to query datasets
3. **Validate integration** through known biological relationships
4. **Downstream analysis** on integrated multi-omics space

2.6 Course Summary: Key Takeaways

2.6.1 Day 3 Learning Objectives Achieved

1. **Deep Learning Fundamentals**:
   - Autoencoder applications for dimensionality reduction
   - Hyperparameter optimization strategies
   - Regularization techniques for omics data
2. **Advanced Integration Methods**:
   - Comprehensive understanding of PCA, t-SNE, and UMAP
   - Batch correction approaches (ComBat, MNN, Harmony, Scanorama)
   - Multi-omics integration strategies
3. **Practical Implementation**:
   - Method selection guidelines based on data complexity
   - Evidence-based recommendations from benchmarking studies
   - Cross-omics integration workflows

2.6.2 Future Directions

The field of multi-omics integration continues to evolve with: - **Graph-based methods** for complex biological networks - **Foundation models** trained on large-scale omics datasets
- **Causal inference** approaches for understanding biological mechanisms - **Real-time integration** for clinical applications

Final Note: The choice of integration method should always be guided by the biological question, data characteristics, and computational resources available.