

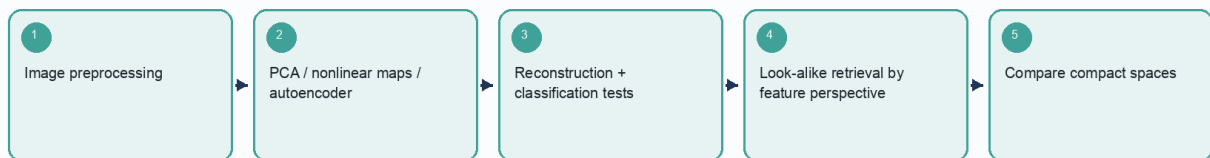
Machine Learning

Practical Project 2 - Dimensionality Reduction

Dataset	Fashion-MNIST: 60,000 training and 10,000 test grayscale fashion images.
Main question	Which compact image representations preserve class information and support useful look-alike fashion search?
Core methods	PCA, nonlinear visualization (t-SNE, UMAP,...), autoencoder latent code, nearest-neighbour retrieval, and optional color/texture descriptors.
Extra angle	Similarity must be studied from different perspectives: shape, texture/edges, color when a color dataset is used, and learned embeddings.

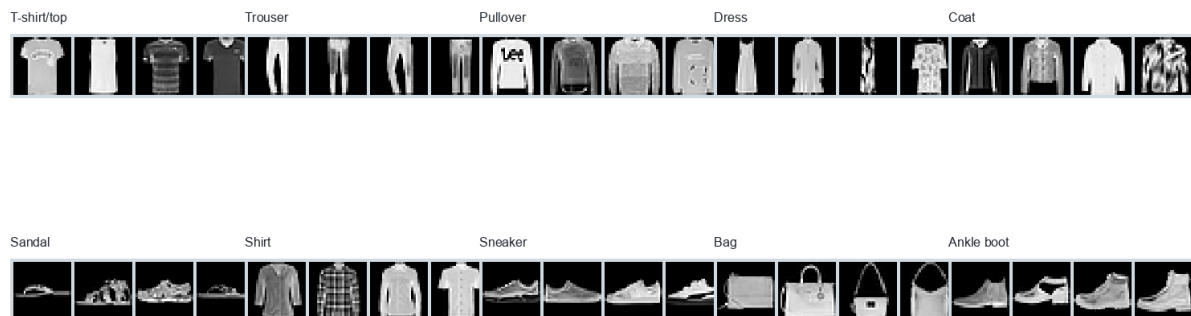
Project 2 scheme

Dimensionality reduction and fashion item search



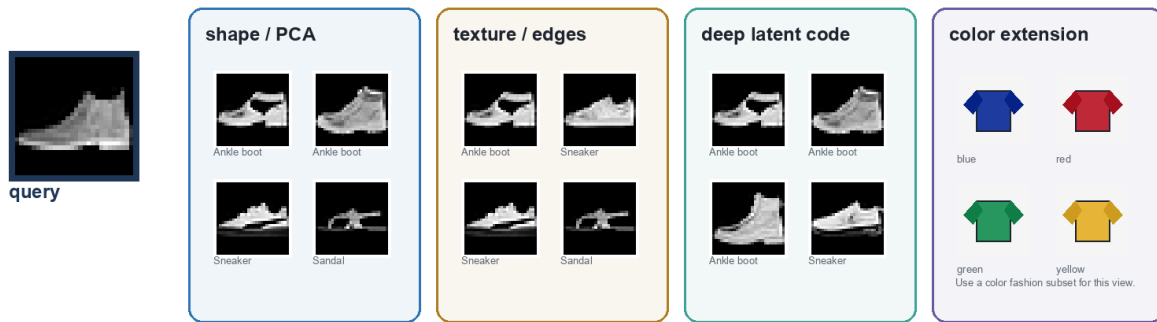
Central idea: compare what the model learns from data alone with what changes when representation, objective, or physics constraints are changed.

Fashion-MNIST examples: 28 x 28 grayscale images, 10 classes



Students will study whether compact representations preserve class structure and support look-alike search.

Look-alike retrieval: the same query can be similar in different spaces



Major project idea

You will learn compact representations for fashion images and then use those representations for two tasks: classification and look-alike retrieval. A good solution must show that the best representation for classification is not necessarily the best representation for visual similarity search.

Major goals

- Reduce the dimensionality of fashion images while preserving relevant visual information.
- Compare PCA-based, nonlinear, and neural latent representations using quantitative and visual evidence.
- Build a look-alike search module that retrieves visually similar items under different feature perspectives.
- Explain the limitations of grayscale Fashion-MNIST and extend the analysis to color/texture when a color subset is available.

Minor goals / checkpoints

- Normalize image intensities and create reproducible train/validation/test partitions.
- Implement PCA and study explained variance, reconstruction error, and the number of retained components.
- Create 2D visualizations with t-SNE, UMAP, or another nonlinear map for diagnostic purposes.
- Train an autoencoder and compare its latent representation with PCA components.
- Train a simple classifier in raw pixel space and in each compact space.
- Implement k-nearest-neighbour retrieval and compute precision@k or class-consistency@k.
- For color data, compare color histograms, texture descriptors, and deep embeddings separately.

Implementation requirements

1. Create a representation.py module with PCA projection/reconstruction and autoencoder encoding/decoding functions.
2. Create a retrieval.py module where an input image returns the top-k nearest items according to at least three distance spaces.
3. Use Fashion-MNIST as the mandatory benchmark. For the color/texture perspective, use a small subset of a color fashion dataset or an instructor-provided equivalent.
4. Compare compact dimensions such as 2, 8, 16, 32, 64, and 128 when applicable.
5. Keep labels hidden during retrieval ranking; labels are used only for evaluation and interpretation.

Experiments to report

- Reconstruction panels: original image, PCA reconstruction, and autoencoder reconstruction for several classes.

- Classification table: raw pixels vs PCA vs autoencoder latent code.
- Retrieval panels: same query item retrieved by shape, texture/edges, color extension, and learned embedding.
- Dimensionality analysis: how many components are needed before accuracy/retrieval quality saturates?
- Failure cases: show at least five cases where two items look similar but have different labels, or same label but visually different style.

What to submit

- Python scripts/notebook implementing dimensionality reduction, reconstruction, classification, and retrieval.
- A report with explained variance, reconstruction, classifier, and retrieval results.
- A visual appendix containing representative look-alike retrieval panels.
- A README describing whether the color extension dataset was used and how it was prepared.

Suggested data sources

Fashion-MNIST: <https://github.com/zalandoresearch/fashion-mnist>

Suggested 4-week workflow

- Load Fashion-MNIST, prepare image splits, and create raw/PCA baselines.
- Train autoencoder and compare reconstruction/classification behavior across latent dimensions.
- Build look-alike retrieval with shape, texture/edges, latent-code, and optional color perspectives.
- Create visual retrieval panels, failure-case analysis, final report, and README.