feature learning

Authors

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

Abstract

Feature learning is the task of learning, from a dataset, a feature transformation that turns the raw features of a data point into new features that make a subsequent learning task easier. A feature learning method is specified by the two feature spaces, by a hypothesis space of candidate feature transformations, and by a quantitative measure of how useful a feature transformation is, computed from a dataset that need not carry labels. Principal component analysis (PCA) learns a linear feature transformation to fewer features, judged by the linear reconstruction error, and an autoencoder obtains a nonlinear one by the same criterion. The hidden layers of a deep net form a chain of feature transformations learned jointly with the output layer, so that each layer builds its features from those of the previous one. Pretraining an encoder on unlabeled data points learns features that are reused across learning tasks. In weather forecasting, forty measurements from the five days before are turned into two learned features from which linear regression predicts the maximum temperature of the next day.

Definition

Consider the task of forecasting the weather of tomorrow at a weather station from the measurements of the previous five days. Each day is a data point, its label is the maximum temperature of that day, and its raw features are the eight daily measurements (lowest, highest and mean air temperature, precipitation, sunshine duration, humidity, air pressure and wind speed) of each of the five previous days: a list of forty numbers. A point with forty coordinates cannot be drawn in a scatterplot. A feature transformation that delivers two new features from the forty raw ones places every day in a scatterplot, and a linear model of the two new features may already predict the label. Feature learning is the task of learning such a feature transformation from a dataset instead of designing it by hand.

Feature learning is the task of learning a feature transformation

$$\boldsymbol{\Phi} : \mathcal{X} \rightarrow \mathcal{X}', \qquad \mathbf{x} \mapsto \mathbf{z} := \boldsymbol{\Phi}(\mathbf{x}).$$

The transformation reads in the raw feature vector $\mathbf{x} \in \mathcal{X}$ of a data point and delivers a new feature vector $\mathbf{z}$ from a new feature space $\mathcal{X}'$. A feature learning method is specified by three design choices: the two feature spaces $\mathcal{X}$ and $\mathcal{X}'$, a hypothesis space $\mathcal{H}$ of candidate feature transformations, and a quantitative

measure of how useful a specific $\mathbf{\Phi} \in \mathcal{H}$ is. The measure is computed from a dataset, which need not carry labels, and it need not be the dataset of the learning task at hand. A useful feature transformation is one that makes a subsequent learning task easier (Goodfellow et al., 2016, Ch. 15): the data points become separable by a linear model, fewer features suffice, or the features learned once serve several learning tasks. Alternative constructions of a feature transformation do not learn $\mathbf{\Phi}$ (Goodfellow et al., 2016, Ch. 6). A kernel method fixes $\mathbf{\Phi}$ through the choice of a kernel, with $\mathcal{X}'$ the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ and $\mathbf{z} = k\big(\mathbf{x}, \cdot\big)$ never evaluated explicitly, and learns only a linear hypothesis on $\mathcal{X}'$. A hand-designed $\mathbf{\Phi}$ encodes domain knowledge instead of a dataset.

Principal component analysis (PCA) is feature learning with $\mathcal{X} := \mathbb{R}^d$, $\mathcal{X}' := \mathbb{R}^{d'}$ for some $d' < d$, and the hypothesis space of linear maps,

$$\mathcal{H} := \Big\{ \mathbf{\Phi} : \mathbb{R}^d \to \mathbb{R}^{d'} : \mathbf{x} \mapsto \mathbf{W}\mathbf{x} \text{ with some } \mathbf{W} \in \mathbb{R}^{d' \times d} \Big\}.$$

The usefulness of a specific $\mathbf{\Phi}(\mathbf{x}) = \mathbf{W}\mathbf{x}$ is measured by the minimum linear reconstruction error on the dataset $\mathcal{D} = \big\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\big\}$,

$$\min_{\mathbf{R} \in \mathbb{R}^{d \times d'}} \sum_{r=1}^{m} \Big\| \mathbf{x}^{(r)} - \mathbf{R}\mathbf{W}\mathbf{x}^{(r)} \Big\|_2^2,$$

and the rows of the best $\mathbf{W}$ are the d' eigenvectors of the sample covariance matrix with the largest eigenvalues (Hastie et al., 2009, Sect. 14.5.1). An autoencoder measures usefulness by the same reconstruction error but replaces the linear maps $\mathbf{W}$ and $\mathbf{R}$ by artificial neural networks (ANNs), so that the new features are obtained by a nonlinear feature transformation.

Applied to the weather example, PCA delivers the two learned features of Fig. 1. The forty raw features of the 361 days of 2024 at Krems, each scaled to zero sample mean and unit sample variance because temperatures, precipitation and pressure carry different units, are turned into two learned features. Warm and cold days separate along the first learned feature: 89 percent of the warm days have a positive z_1 and 92 percent of the cold days a nonpositive one. Linear regression on the two learned features, after training on the days from January to August, predicts the maximum temperature of the days from September to December with a validation error of 13.5 (squared degrees Celsius), against 10.2 for linear regression on all forty raw features and 110.9 for predicting the sample mean of the training labels.

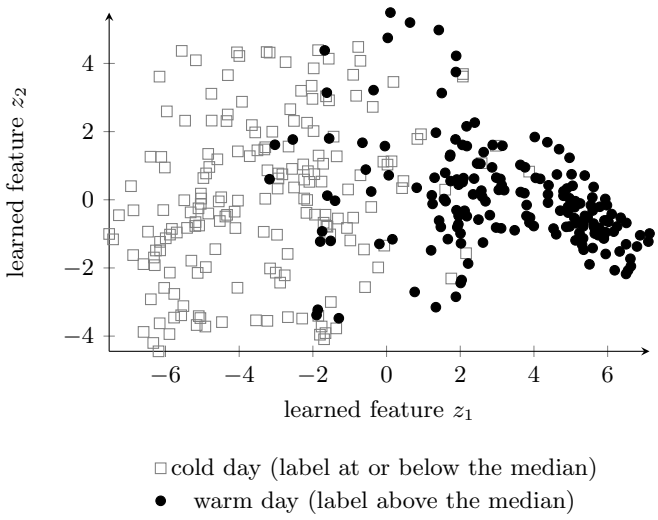

Figure 1: The 361 days of 2024 at Krems in the two features learned by PCA from forty raw features, the eight daily measurements of the five previous days. Warm and cold days, split at the median of the label, separate along the first learned feature z_1. Data generated by `pythondemos/featlearn.py`

A deep net learns a chain of feature transformations. Each hidden layer $\ell = 1, \ldots, L$ is a feature transformation $\mathbf{\Phi}^{(\ell)}$ that reads in the activations $\mathbf{z}^{(\ell-1)}$ of the previous layer, with $\mathbf{z}^{(0)} := \mathbf{x}$. The features delivered to the output layer are the composition $\mathbf{z}^{(L)} = \mathbf{\Phi}^{(L)} \circ \cdots \circ \mathbf{\Phi}^{(1)}(\mathbf{x})$ (see Fig. 2). The feature transformations are learned jointly with the output layer by empirical risk minimization (ERM) with the loss of the final prediction, so the usefulness of the learned features is measured by how well the output layer, typically a linear model, predicts the label from $\mathbf{z}^{(L)}$ (Goodfellow et al., 2016, Ch. 15). The chain is useful because each layer builds its features from those of the previous one, so the feature transformations $\mathbf{\Phi}^{(\ell)} \circ \cdots \circ \mathbf{\Phi}^{(1)}$ grow deeper with ℓ. In image classification, early layers deliver edges, later ones corners and object parts, and the features at the last hidden layer separate the classes by a linear surface (Bishop and Bishop, 2024, Sect. 6.3.3; Goodfellow et al., 2016, Ch. 1).

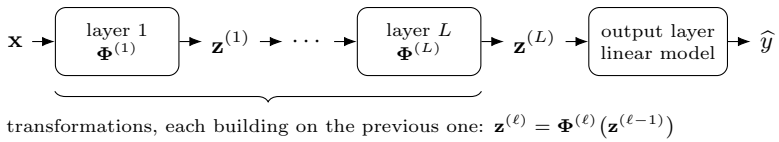

Figure 2: A deep net as a chain of feature transformations: hidden layer ℓ turns the activations $\mathbf{z}^{(\ell-1)}$ of the previous layer into $\mathbf{z}^{(\ell)}$, with $\mathbf{z}^{(0)} := \mathbf{x}$, and the output layer applies a linear model to $\mathbf{z}^{(L)}$. All layers are learned jointly with the linear model by ERM with the loss of the final prediction

Pretraining an encoder on a large dataset of unlabeled data points, with a loss constructed from the data points themselves (see self-supervised learning), learns features that are reused across learning tasks (Goodfellow et al., 2016, Sect. 15.1). A new learning task appends a task-specific hypothesis with few model parameters to the learned $\boldsymbol{\Phi}$ and adapts it by fine-tuning on its own dataset. The dataset used to learn $\boldsymbol{\Phi}$ can then be much larger than any labeled dataset, because unlabeled data points are cheap to collect (Bishop and Bishop, 2024, Sect. 6.3.3).

Cross References

feature transformation, feature, feature space, hypothesis space, PCA, autoencoder, dimensionality reduction, deep net, layer, pretraining

References

1. Bishop and Bishop (2024). Deep Learning: Foundations and Concepts. Springer Nature, Cham, Switzerland.

2. Goodfellow et al. (2016). Deep Learning. MIT Press, Cambridge, MA, USA.

3. Hastie et al. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer Science+Business Media, New York, NY, USA.