Nonlinear Dimensionality Reduction using Neural Networks

Ruslan Salakhutdinov

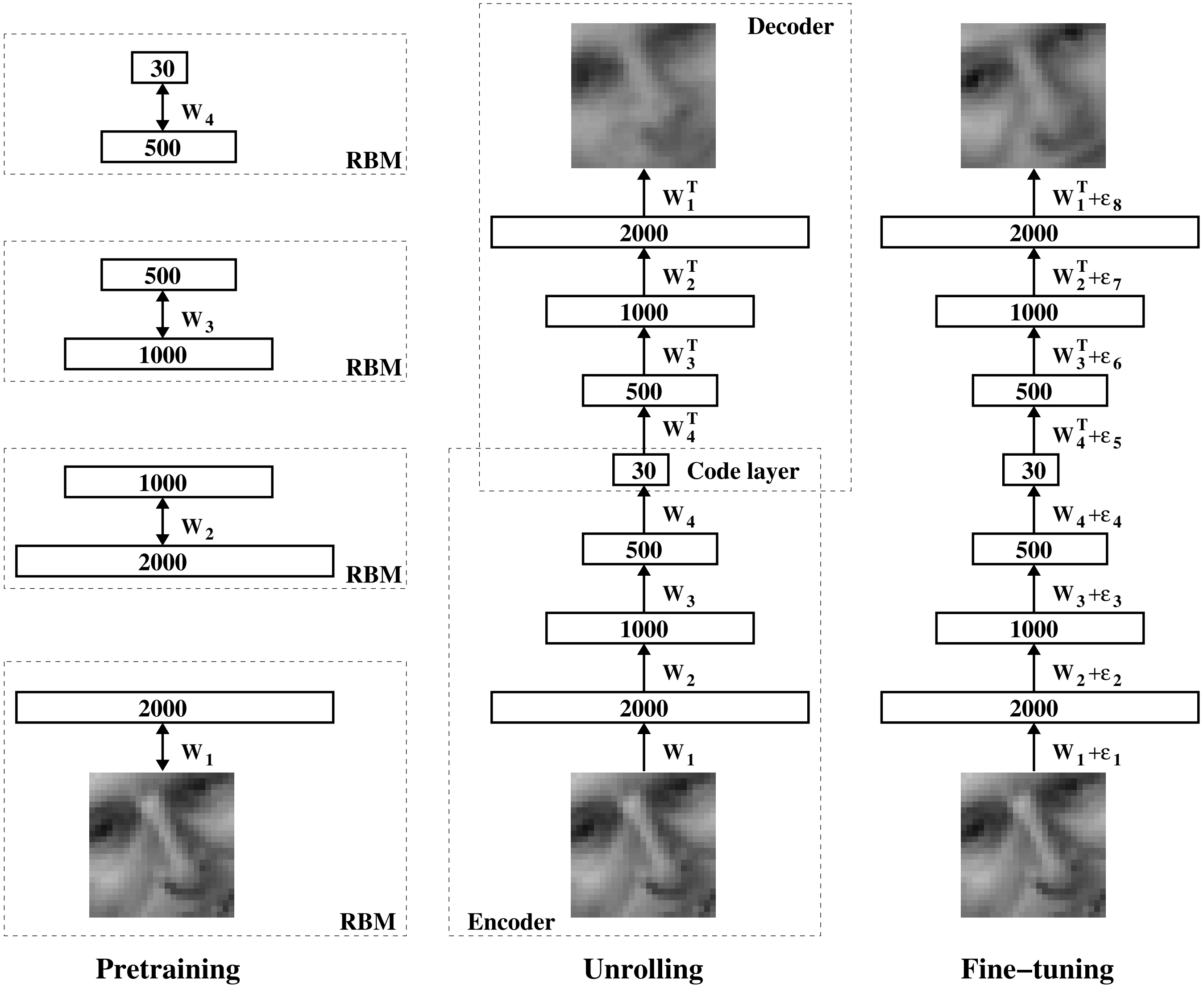
University of Toronto, Machine Learning Group

Joint work with Geoff Hinton

Drawbacks of Existing Methods

- Linear Methods:
  - If the data lie on an embedded low-dimensional nonlinear manifold then linear methods cannot recover this structure.
- Proximity based methods are more powerful, BUT
  - computational cost scales quadratically with the number of observations.
  - cannot be applied to very large high-dimensional data sets.
- Nonlinear mapping algorithms, such as autoencoders:
  - painfully slow to train.
  - prone to getting stuck in local minima.

Pretraining and Fine-Tuning Deep Autoencoders



- First learn good generative model of data.
- Then fine-tuned using backpropagation of error derivatives.

Training an Autoencoder

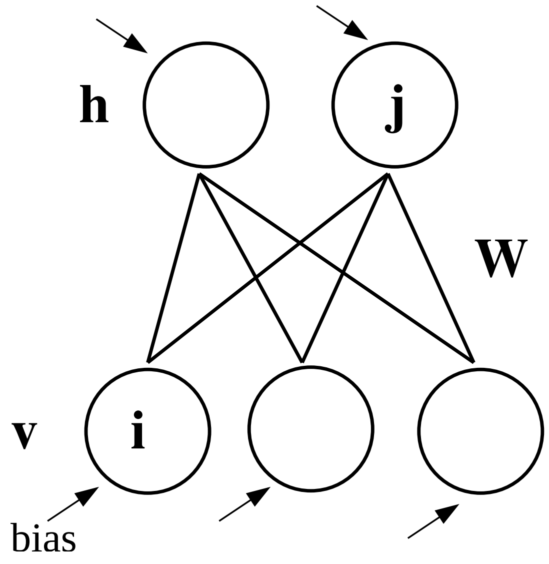
- The standard way to train autoencoders is to use a back-propagation algorithm to reduce reconstruction error.

- Autoencoders with multiple hidden layers seldom work well:
  - initial random weights are large $\rightarrow$ backprop finds poor local minima.
  - initial random weights are small $\rightarrow$ optimization takes very long time.

  - initial weights are close to a solution $\rightarrow$ backprop works well.

- How can we learn these initial weights?

Restricted Boltzmann Machines



- We can model an ensemble of binary images using Restricted Boltzmann Machine (RBM).

- RBM is a two-layer network in which visible, binary stochastic pixels $\mathbf{v}$ are connected to hidden binary stochastic feature detectors $\mathbf{h}$.

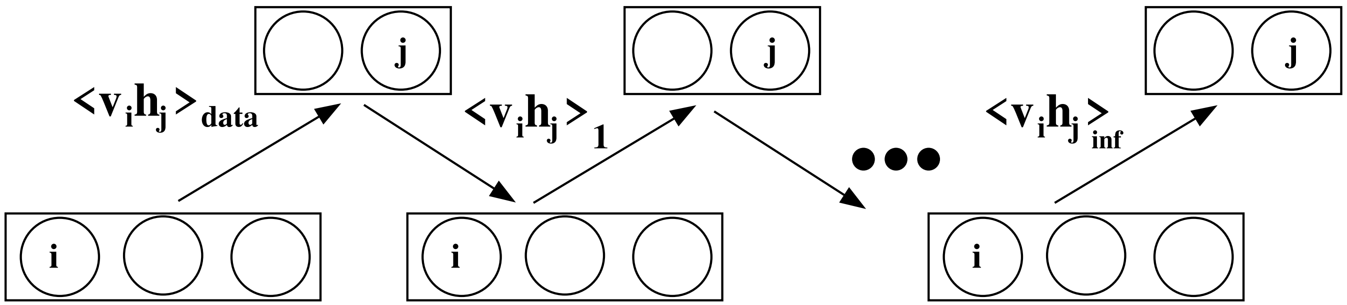
- A joint configuration $(\mathbf{v}, \mathbf{h})$ has an energy:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in \text{pixels}} b_i v_i - \sum_{j \in \text{features}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

- The probability that the model assigns to $\mathbf{v}$ is

$$p(\mathbf{v}) = \sum_{\mathbf{h} \in \mathcal{H}} p(\mathbf{v}, \mathbf{h}) = \frac{\sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}))}{\sum_{\mathbf{u},\mathbf{g}} \exp(-E(\mathbf{u}, \mathbf{g}))}$$

Inference and Learning



- Conditional distributions over hidden and visible units are given by logistic function:

$$p(h_j = 1|\mathbf{v}) = \frac{1}{1 + \exp(-b_j - \sum_i v_i w_{ij})}$$
$$p(v_i = 1|\mathbf{h}) = \frac{1}{1 + \exp(-b_i - \sum_j h_j w_{ji})}$$

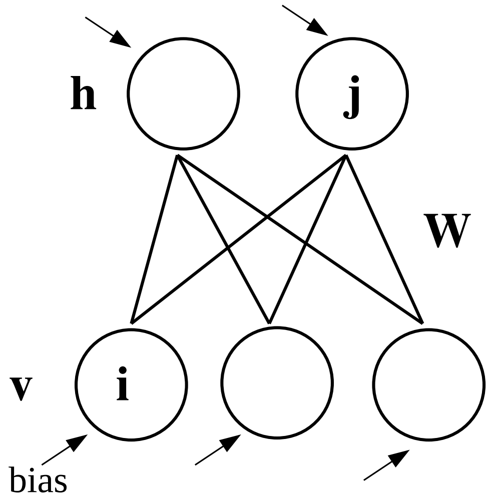
- Maximum Likelihood learning:

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{data} - < v_i h_j >_{\infty})$$

- Contrastive Divergence (1-step) learning:

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{data} - < v_i h_j >_{1})$$

RBM for continuous data



- Hidden units remain binary.

- The visible units are replaced by linear stochastic units that have Gaussian noise.

- The energy becomes:

$$E(\mathbf{v}, \mathbf{h}) = \sum_{i \in \text{pixels}} \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{j \in \text{features}} b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij}$$

- Conditional distributions over hidden and visible units are:

$$p(h_j = 1 | \mathbf{v}) = \frac{1}{1 + \exp(-b_j - \sum_i w_{ij} v_i / \sigma_i)}$$

$$v_i | \mathbf{h} \sim \mathcal{N}(b_i + \sum_j \sigma_i h_j w_{ij}, \sigma_i^2)$$

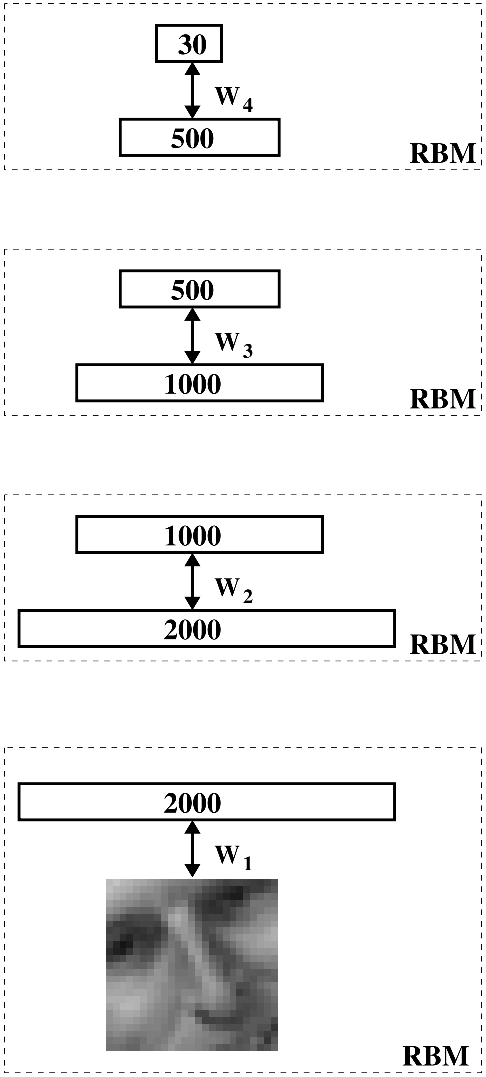
Learning Multiple Layers - Pretraining

- A single layer of binary features generally cannot perfectly model the structure in the data.

- Perform greedy, layer-by-layer learning:
  - Learn and Freeze W_1.
  - Treat the existing feature detectors, driven by training data, $W_1^T V$ as if they were data.
  - Learn and Freeze W_2.
  - Proceed recursive greedy learning as many times as desired.

- Under certain conditions adding an extra layer always improves a lower bound on the log probability of data. (In our case, these conditions are violated)

- Each layer of features captures strong high-order correlations between the activities of units in the layer belows.



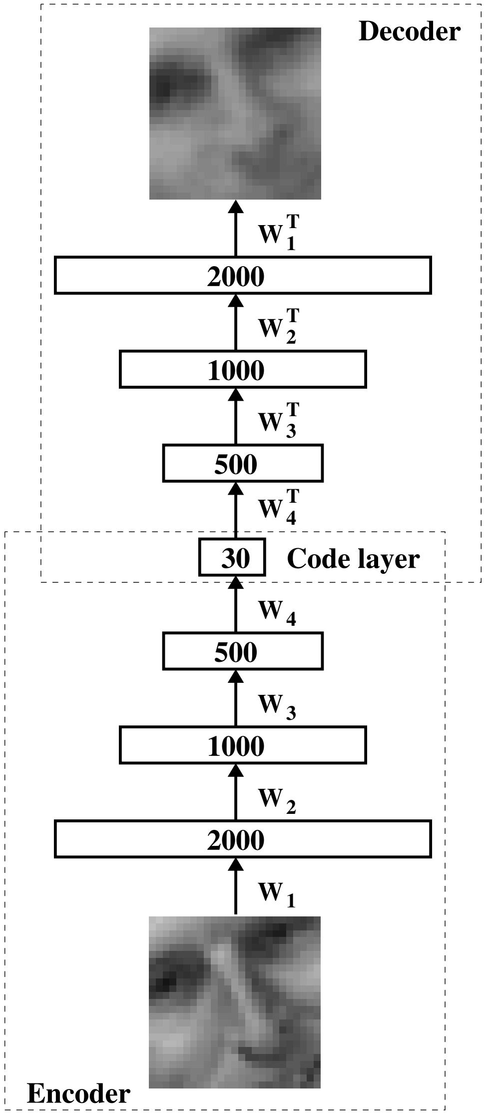
Unrolling and Fine-tuning

- After pretraining multiple layers, the model is unrolled.

- Initially encoder and decoder networks use the same weights.

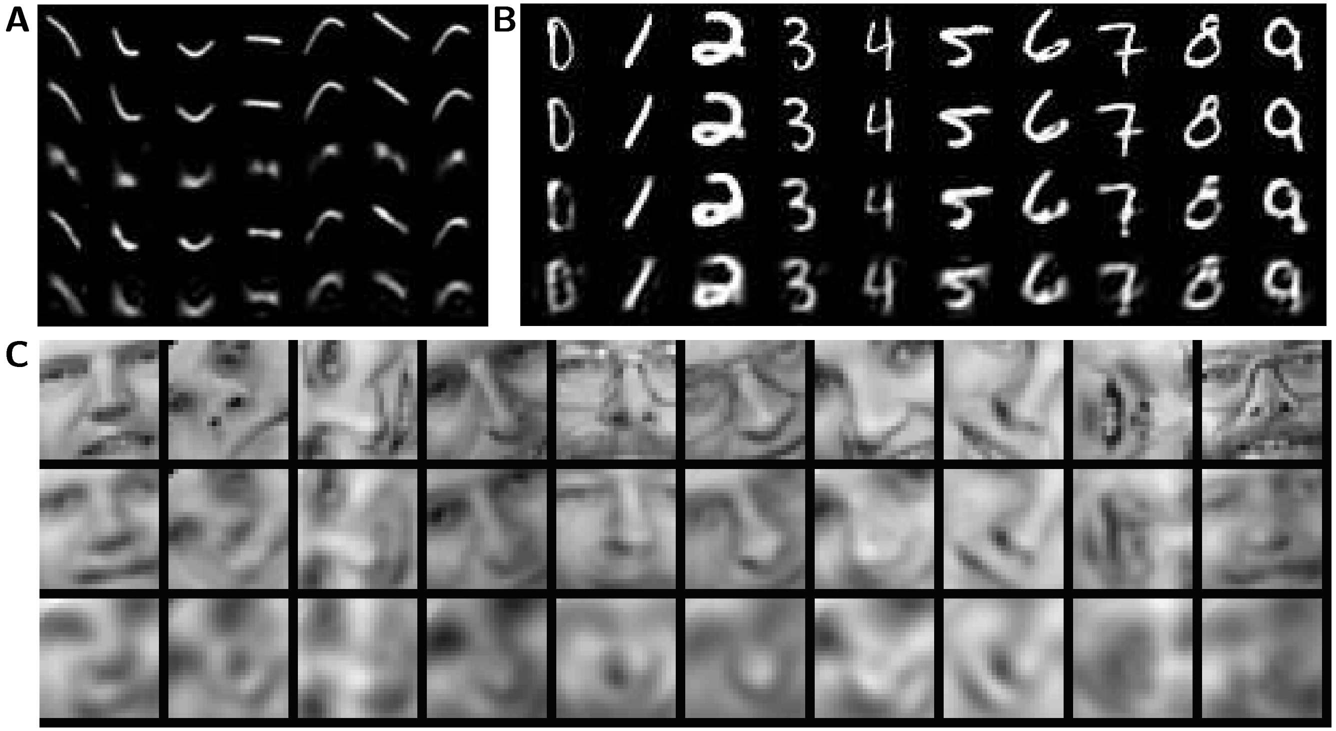
- The global fine-tuning uses backpropagation through the whole autoencoder to fine-tune the weights for optimal reconstruction.

- Backpropagation only has to do local search.



Unrolling

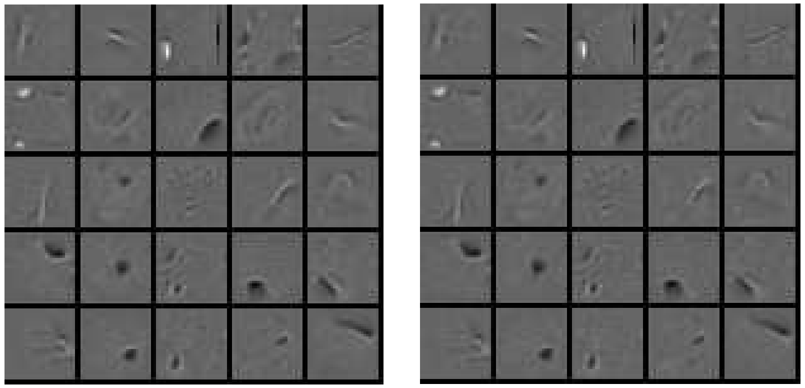
Results



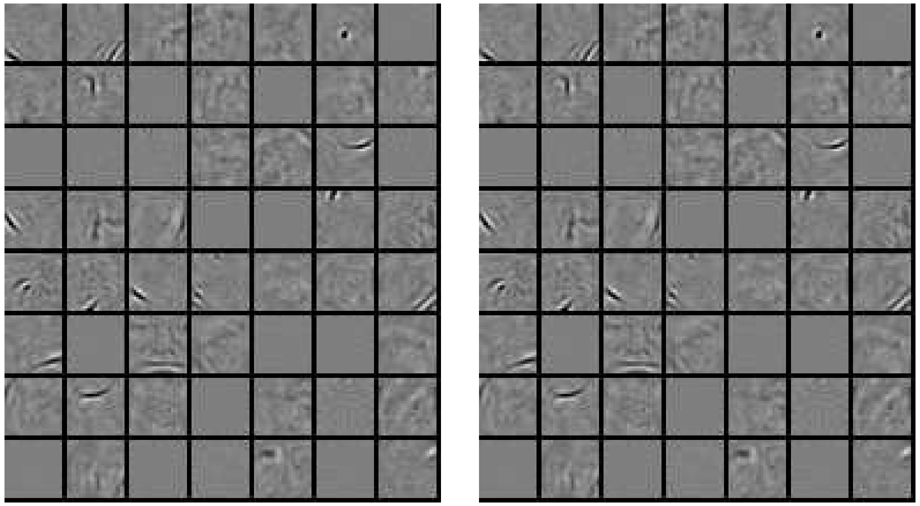
- **A** Top left panel (by row): Random samples of curves from the test dataset; reconstructions produced by the 6-dimensional deep autoencoder (784-400-200-100-50-25-6); reconstructions by "logistic PCA" using 6 components; reconstructions by logistic and standard PCA using 18 components. The average squared error per image for the last four rows is 1.44, 7.64, 2.45, 5.90.
- **B** Top right panel (by row): A random test image from each class; reconstructions by the 30-dimensional autoencoder (784-1000-500-250-30); reconstructions by 30-dimensional logistic PCA and standard PCA. The average squared errors for the last three rows are 3.00, 8.01, and 13.87.
- **C** Bottom panel (by row): Random samples from the test dataset; reconstructions by the 30-dimensional autoencoder (625-2000-1000-500-30); reconstructions by 30-dimensional PCA. The average squared errors are 126 and 135.

Results

- Random sample of recognition receptive fields before and after fine-tuning for MNIST digits.



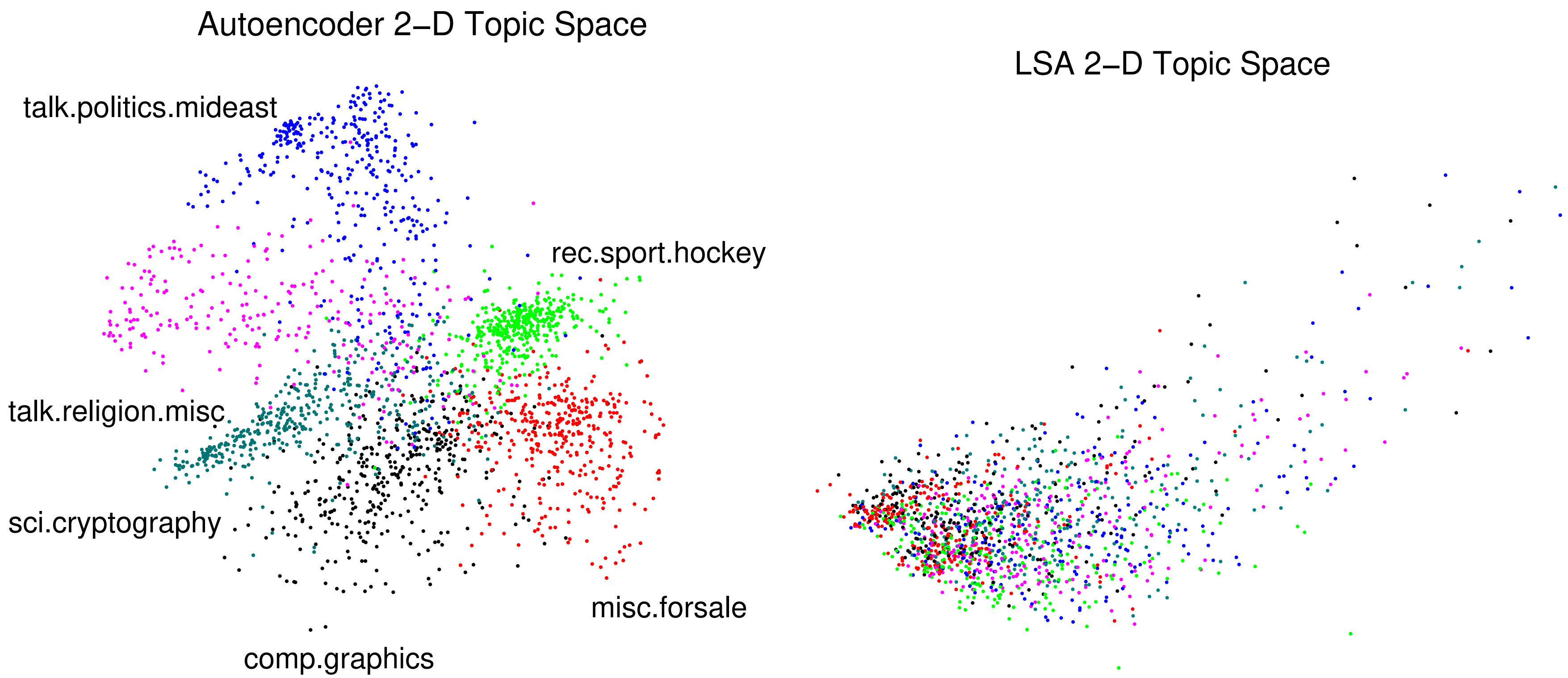
- Random sample of recognition and generative receptive fields for unaligned Olivetti faces after fine-tuning:



Document Retrieval

- We use a 2000-500-250-125-10 autoencoder to convert a document into a low-dimensional code.
- The 20 newsgroup corpus contains 18,845 postings (11,314 training and 7,531 test) taken from the Usenet newsgroup collection.
- The Reuters Corpus Volume II contains 804,414 newswire stories. The data was randomly split into **402,207** training and **402,207** test articles.
- We used a simple “bag-of-words” representation in which each posting is represented as a vector containing most frequent 2000 word counts in the training dataset.

20 newsgroup corpus: Learning 2-D topic space



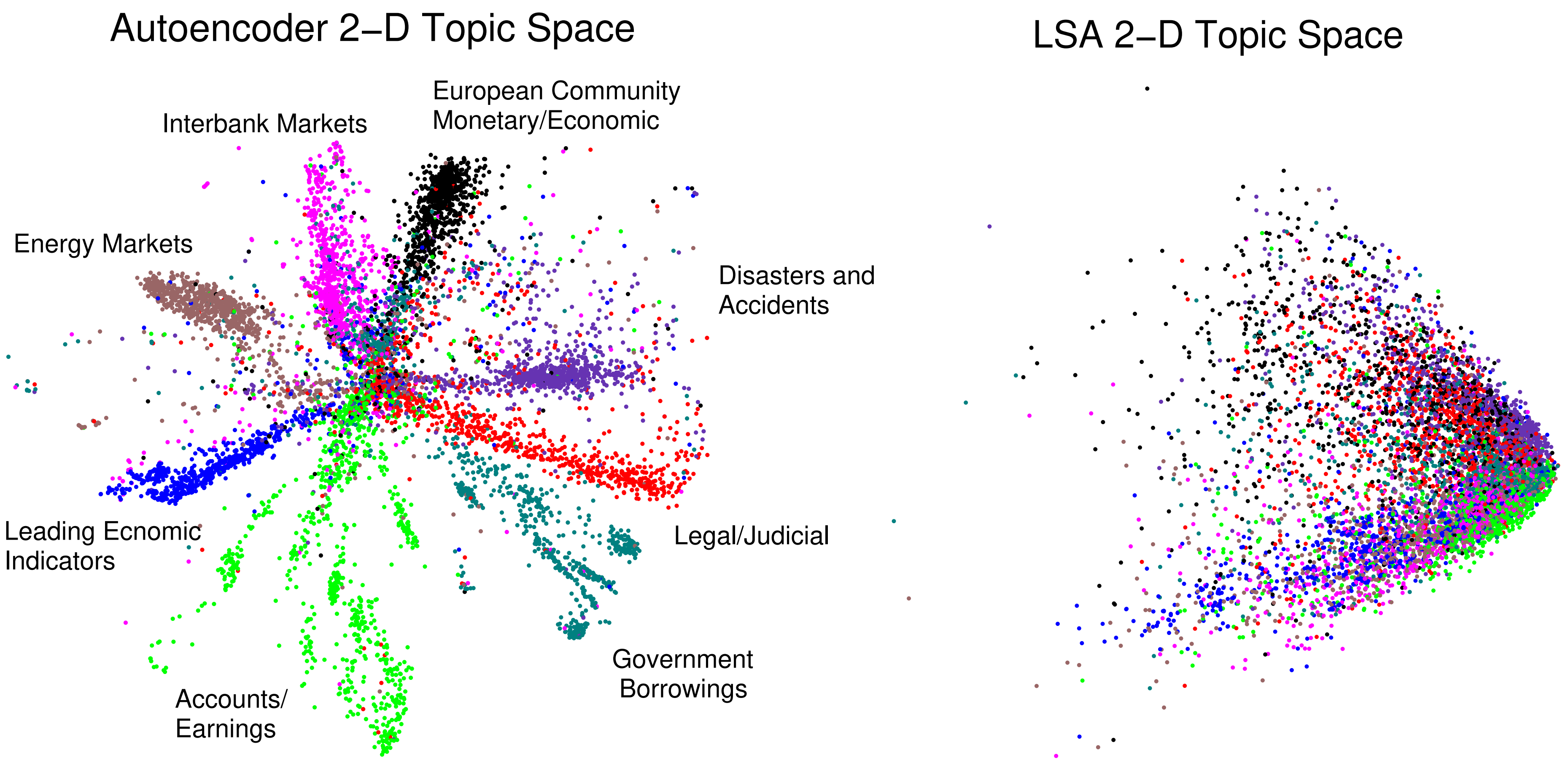
- Latent Semantics Analysis (LSA) uses SVD to get a low-rank approximation of the log of term-frequency matrix:

$$\log(1 + M(doc, w)) \sim USV$$

$U = |doc| \times d$, $S = d \times d$, $V = d \times |w|$.

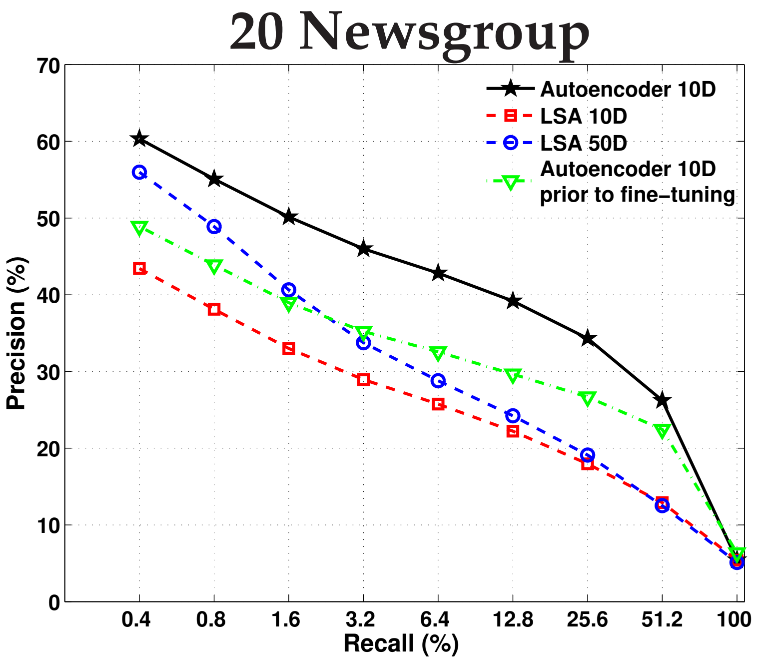
- A test query q is represented as d-dim vector $S^{-1}V \log(1 + q)$.

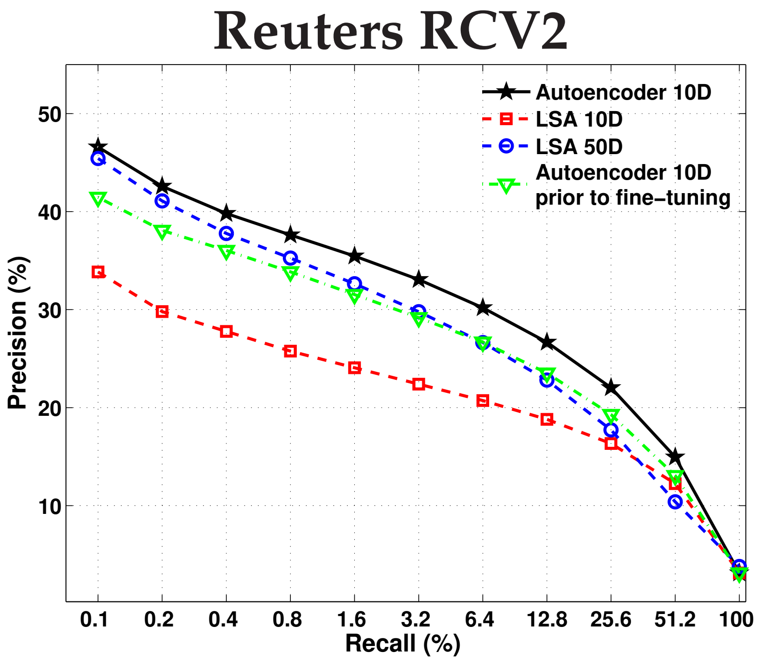
Reuters Corpus: Learning 2-D topic space



Precision-Recall Curves: 10-D topic space

- We use the cosine of the angle between two codes as a measure of similarity.





- Precision-recall curves when a 10-D query document from the test set is used to retrieve other test set documents, averaged over 7,531 (20 Newsgroup) and 402,207 (RCV2) possible queries.

Conclusion

- Autoencoders are very effective for non-linear dimensionality reduction.
- They give mappings in both directions between the data space and the code space
- Both pretraining and fine-tuning scale linearly in time and space with the number of training vectors.
- So we can apply autoencoders to large datasets.

Details of the pretraining

- All datasets were subdivided into mini-batches, each containing 100 data vectors.

- Each hidden layer was greedily pretrained for 50 passes through the entire training set.

- The weights were updated using a learning rate of 0.1.

- Weights were initialized with small random values sampled from $\mathcal{N}(0, 0.01)$

Details of fine-tuning

- For the fine-tuning, we used conjugate gradients ”minimize” on larger minibatches containing 1000 data vectors.

- To check for overfitting, we fine-tuned each autoencoder on a fraction of the training data and tested its performance on the remainder validation set.

- For the hand-written digits, we used 200 epochs of fine-tuning and no overfitting was observed.

- For the faces we used 20 epochs and there was slight overfitting.

- For the documents we used 50 epochs (both for 20 Newsgroups and Reuters RCV2). Slight overfitting was observed for the 20 newsgroup data.