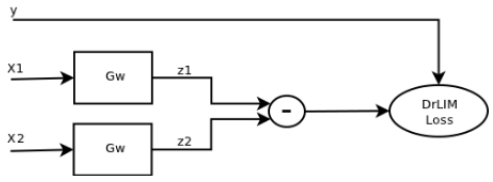
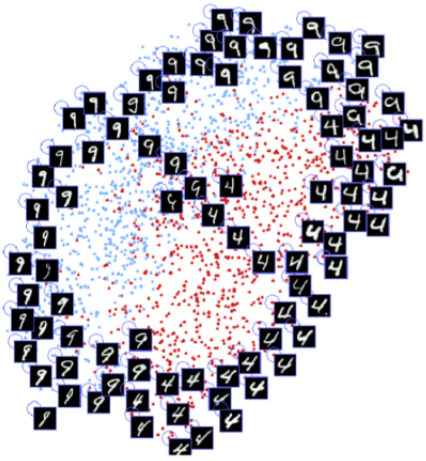
The Story
A metric learning-like framework is proposed for unsupervised training of neural networks. As demonstrated in algorithms such as DrLIM, ISOMAP and others, learning a metric can implicitly lead to a representation in which the underlying latent variables are disentangled or linearized. However the approach relies on an oracle to provide semantically meaningful similarity relationships. Temporal coherence can be exploited to provide similarity relationships needed to learn a semantically meaningful metric: temporally neighboring frames in a video are likely to be semantic neighbors. The objective thus becomes to extract slowly varying temporal features which closely resembles slow feature analysis (SFA). However the proposed algorithm differes from SFA in two important ways. Firstly, the slow features need not be linear functions of the input. Secondly, we explicitly enforce that the feature representation be information preserving by minimizing reconstruction error. The last requirement naturally leads to a Siamese auto-encoder like architecture which lead to the name Temporal Auto-Encoder (TAE).

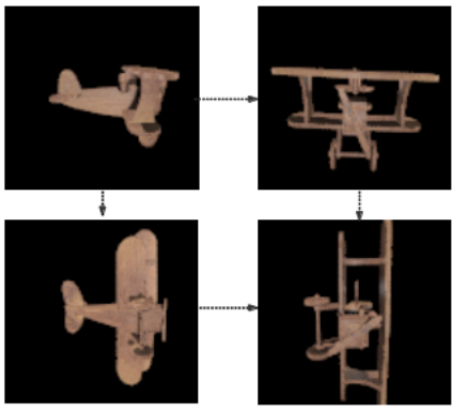
Motivation: Metric Learning as Unsupervised Feature Learning
Toy examples when DrLIM learns independent factors of variation:



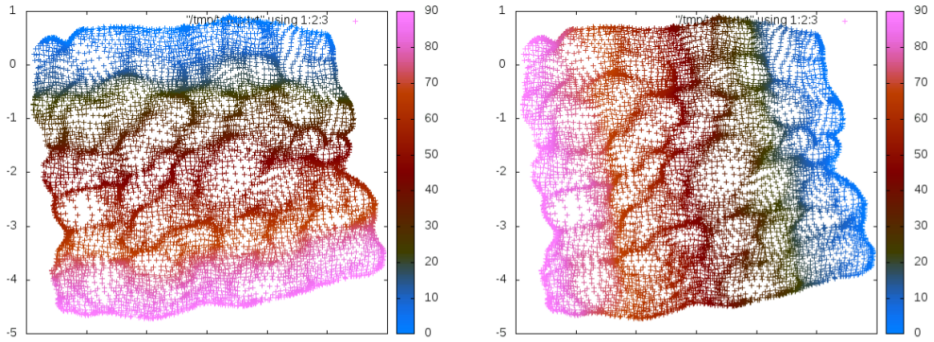


Temporal toy data: rotating airplane

- 2-dimensional manifold living in a $\approx 10,000$-dimensional space (96x96 images)
- Similarity relationships can be naturally assigned via adjacent frames in a video



Learns to implicitly extract angles of rotation, i.e. true latent variables.

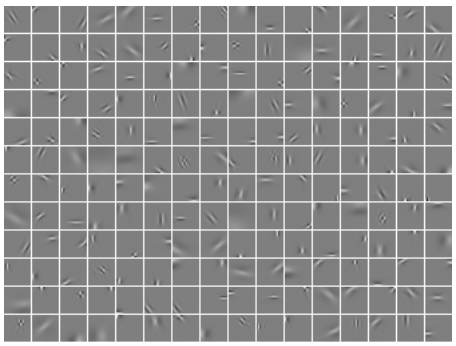


Summary of Results

Sparse-Transition Features

$$L = \sum_{i=1}^{2} \frac{1}{2}\|x_i - W_d\ z_i\|^2 + \alpha|z_1 - z_2| \text{ where } z_i = ReLU(W_e\ x_i)$$

The resulting basis found using slowness (or sparse-transitions since L_1 is used) alone is similar, but not identical to those obtained by a sparse auto-encoder.



Sparse/Slow Pooled Features

$$L = \sum_{i=1}^{2} \left(\frac{1}{2}\|x_i - W_d\ h_i)\|^2 + \alpha|h_i|\right) + \beta|z_1 - z_2|$$

Where, $h_i = ReLU(W_e x_i)$ and $z_i = pool(h)$. That is $pool()$ is a pooling operator which includes pooling across features as well as neighboring spatial locations in the convolutional setting. W_e and W_d are learned linear operators which may be convolutional. Thus the feed-forward mapping from the input x to the feature space z is given by $z = pool(ReLU(W_e\ x))$ which corresponds to a single stage in the most widely used convolutional networks if W_e is a convolutional operator.

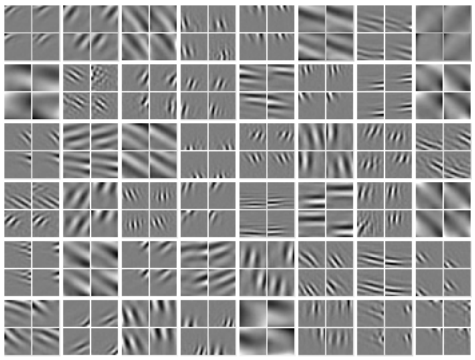


Figure 1: L_2 pooled fully connected sparse/slow features

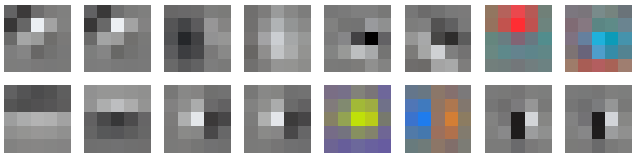


Figure 2: L_2 pooled 2-pooling

Sparse/Slow Pooled Convolutional Features