How unlabeled data helps us to learn a good representation.

Greedy Layer-wise Unsupervised Pretraining

- Basic idea: for each layer, training them independently, and when focus on one layer, set other layers unchanged to perform "greedy" algorithm.
- Modern methods often do supervised training and unsupervised training simultaneously, to ensure the quality of unsupervise pretraining.
- Unsupervised pretraining initializes the weights in a stable start point which is near the optimal, or invariant to big changes.
- NLP uses unsupervised pretraining mostly nowadays, other tasks such as CV do not use these, maybe for CV, the vectors / tensors of pixels naturally provide metrics of similiarity.

Transfer Learning and domain adaption

Key of talking on this topic is it provides nice cases for representation learning.

- Transfer learning (general idea): Explanation of task 1 can help learning task 2, especially when task 2 has few data or poor training condition.
- Domain adaption: Tasks on different data domain use some shared models or weights to save learning effort, e.g. when analyzing sentiment, we can use model trained on book to develope model trained on web comments.
- One/ zero shot learning:

  Idea behind one-shot learning: Learning an appropriate representation in first settings, then for a new come one-shot example, it will clearly map it into representive domain.

  Idea behind zero-shot learning: Learning a feasible relationship of two information, (taking advantage of huge base data domain), and then even we do

not have data in observed domain, we get all the connections in latent or representive spaces.

- Transfer learning can be summarized on learning three tasks: learning "x", learning "y", learning (x, y).

Semi-supervised Disentangling of Causal factors

Basic idea: Semi-supervised learning on representation helps when it learns the underneath causal factors of data, namely how we get the observed data from true distributive generation.

- Motivation of semi-supervised learning: In general, we get underlying representations explaining attributes.
- Consider formulative explanation:

  Suppose we have x, and label y, then if we modeling (representing) distribution P(x) as $\sum_h P(x|h)P(h)$, then as P(h) is closely related to P(y), the representation will be more feasible.

- Always, in deep learning scenario specially, the causal factor is uncertain and large. Examples demonstrated by auto encoder and GAN, where GAN is more powerful of capturing trivial patterns.
- Causal factor remain invariant among tasks, which means that if we successfully model P(x|h), no matter what pattern of y, we can easily develop based on P(x|h). (Four legs → dog, Four legs → cat)

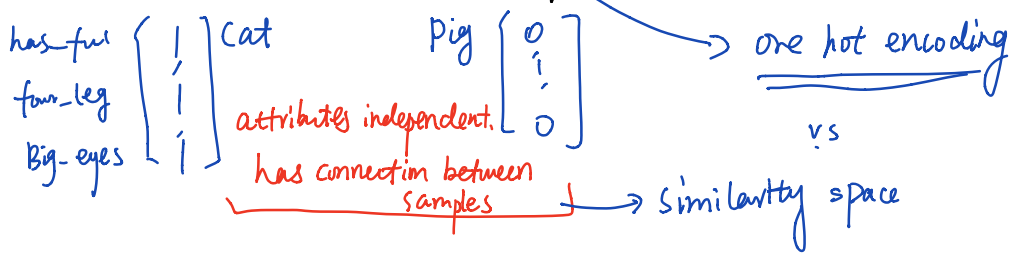
Distributed Representation

Distributed vs Non-distributed:

↓ More tight parameters, less data required, more powerful representation capacity.

→ Need samples for each class, more weights, less powerful representation capacity

Distributed / Non-distributed examples:



Deep learning aims to learn distributed representation, also distributed representation gives a prior belief of functions we learned is underlying causal factors. May be not all the possible combinations. Also no need all samples cover all the situations.

Exponetial Gain from depth

Learning such a representation job benefits a lot of designing neural networks with deeper layers.

Providing clues to discovering underlying causes

Smooth, linearity, multiple explanatory factors, causal factors, hierachy, shared factors across tasks, manifolds, natural clustering, temporal and space coherence, sparsity, simplicity for factor dependencies.