Applied Deep Learning

Unsupervised Learning

May 25th, 2020 http://adl.miulab.tw

國立臺灣大學
National Taiwan University

Introduction

- Big data ≠ Big annotated data
- Machine learning techniques include:
  - Supervised learning (if we have labelled data)
  - Reinforcement learning (if we have an environment for reward)
  - Unsupervised learning (if we do not have labelled data)

What can we do if there is no sufficient training data?

3

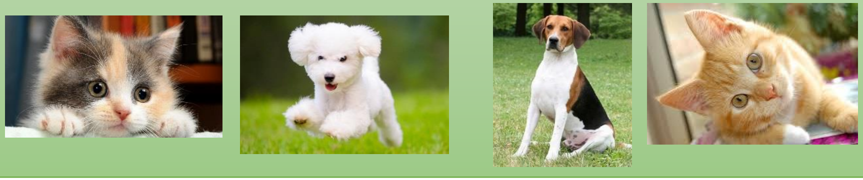
Semi-Supervised Learning

Labelled Data


cat


dog

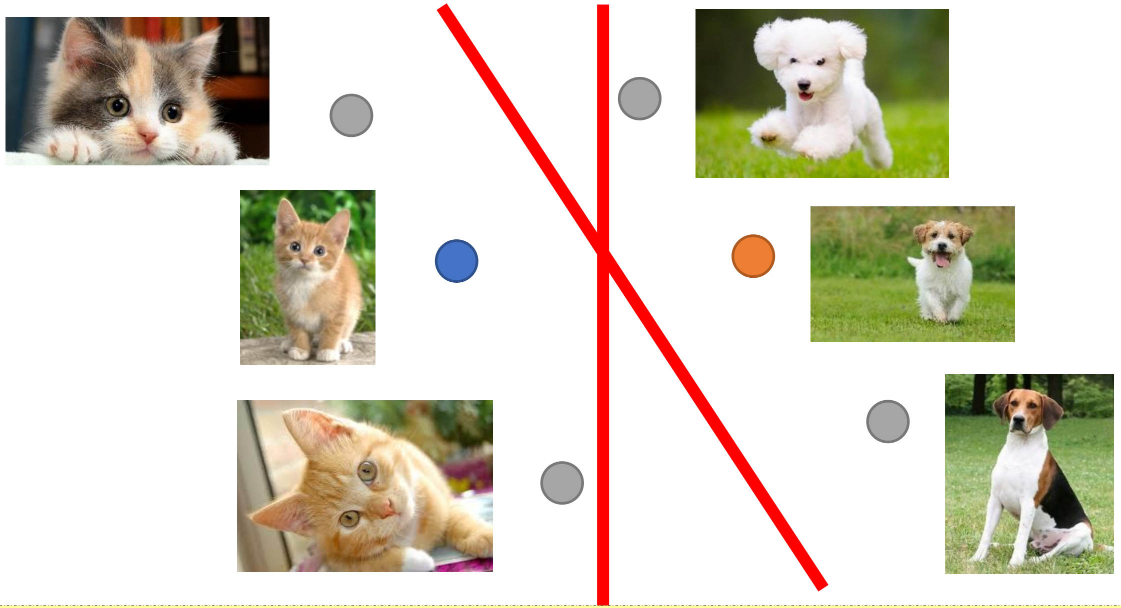
Unlabeled Data



(Image of cats and dogs without labeling)

4 Semi-Supervised Learning

- Why semi-supervised learning helps?



The distribution of the unlabeled data provides some cues

Transfer Learning

Source Data


cat


dog

Target Data


elephant


elephant

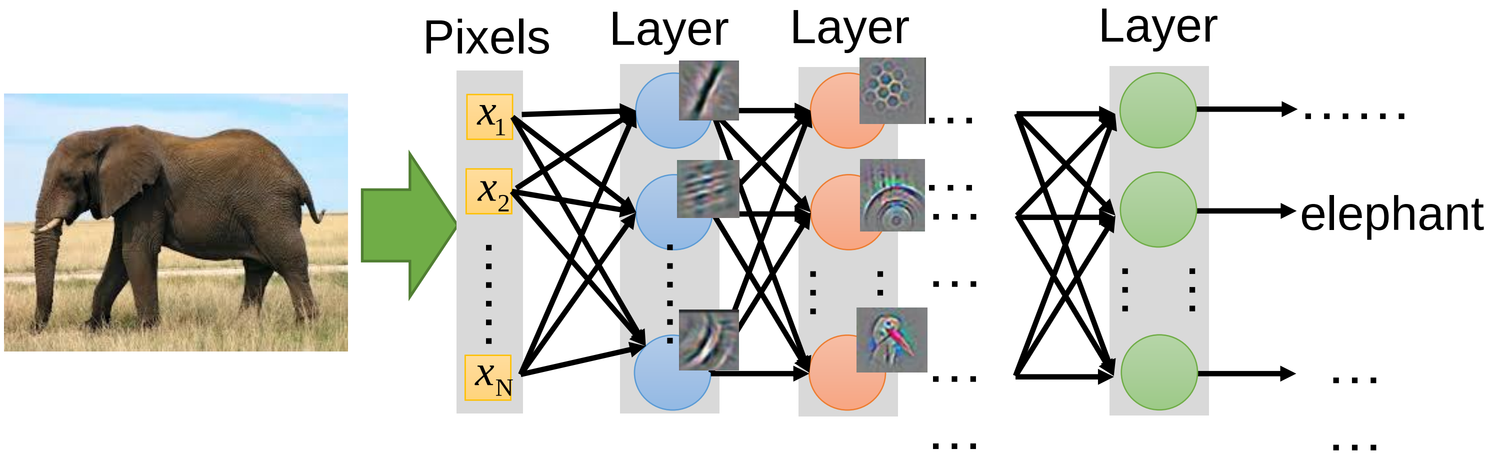

tiger


tiger

Not related to the task considered

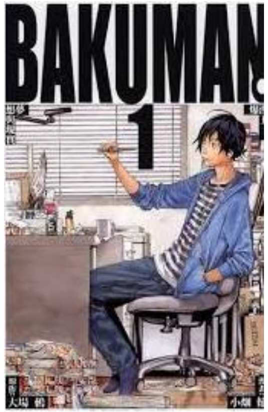
Transfer Learning

- Widely used on image processing
  - Using sufficient labeled data to learn a CNN
  - Using this CNN as feature extractor



Transfer Learning Example

研究生 online		***漫畫家 online***
研究生生存守則	研究生	漫畫家
	指導教授	責編
	跑實驗	畫分鏡
	投稿期刊	投稿 jump



爆漫王

Self-Taught Learning

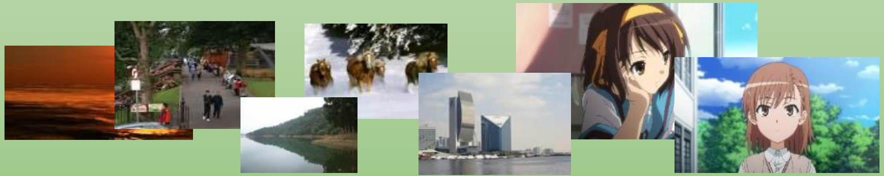
- The unlabeled data sometimes is not related to the task

Labelled Data

cat

dog

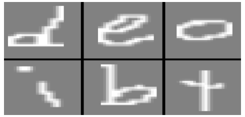
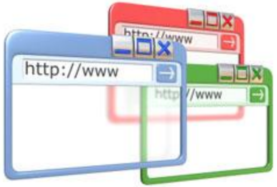
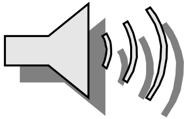
Unlabeled Data



(Just crawl millions of images from the Internet)

9 Self-Taught Learning

- The unlabeled data sometimes is not related to the task

	Labelled Data	Unlabeled Data
Digit Recognition	 Digits	 character
Document Classification	 News	 Webpages
Speech Recognition	 Taiwanese	YouTube English Chinese ……

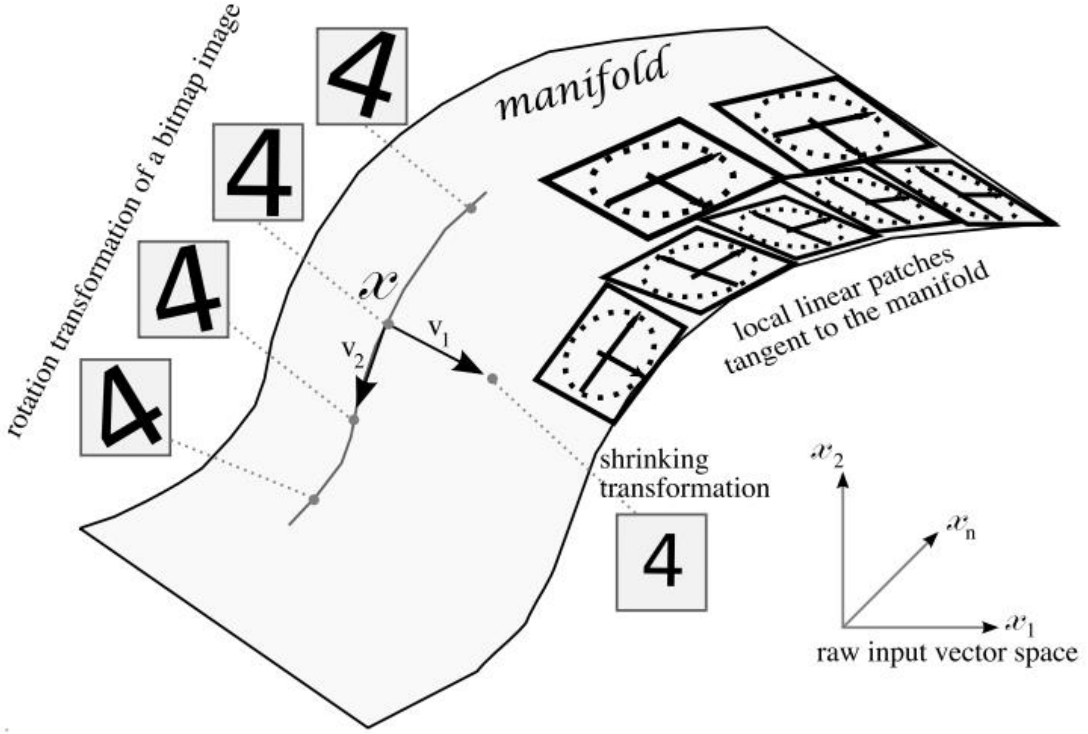
Why can we use unlabeled and unrelated data to help our tasks?

Self-Taught Learning

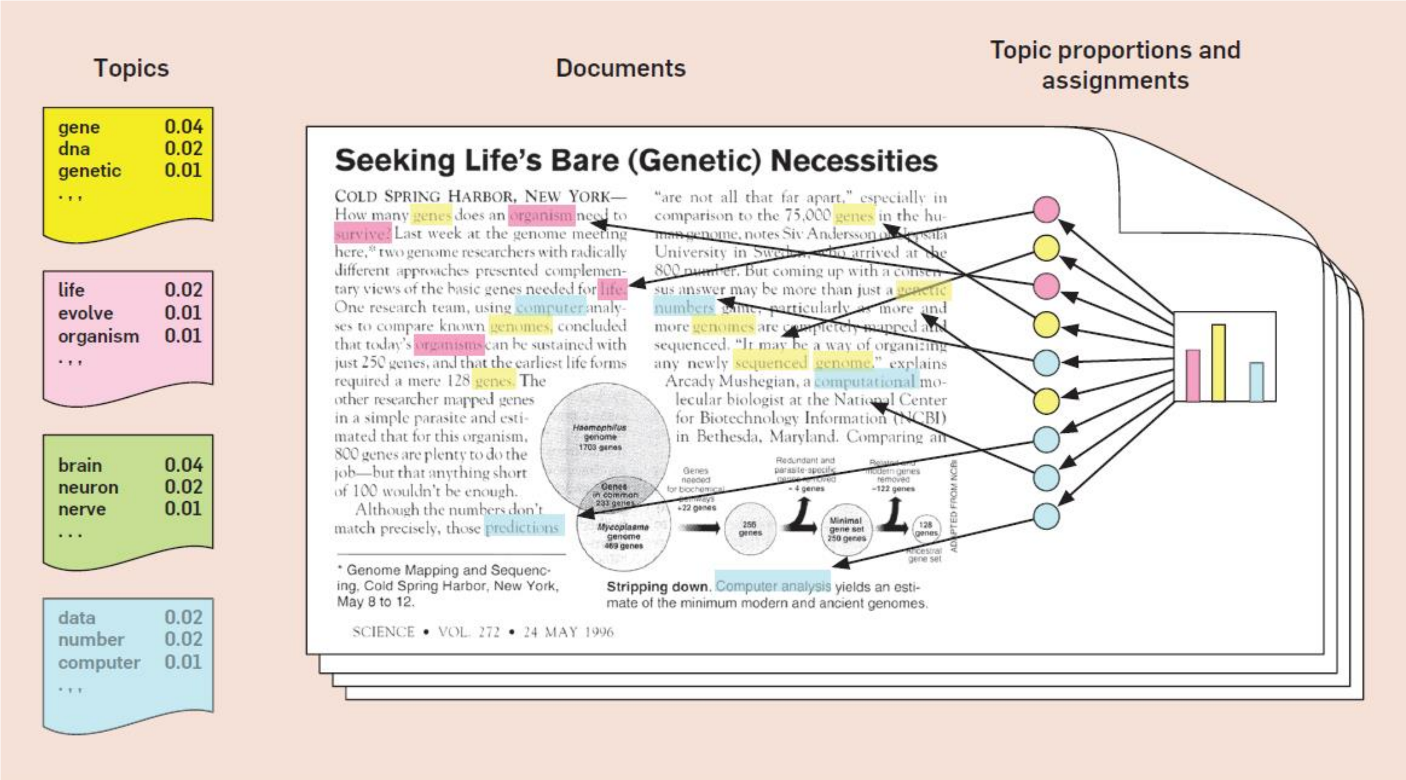
- How does self-taught learning work?
- Why does unlabeled and unrelated data help the tasks?

Finding latent factors that control the observations

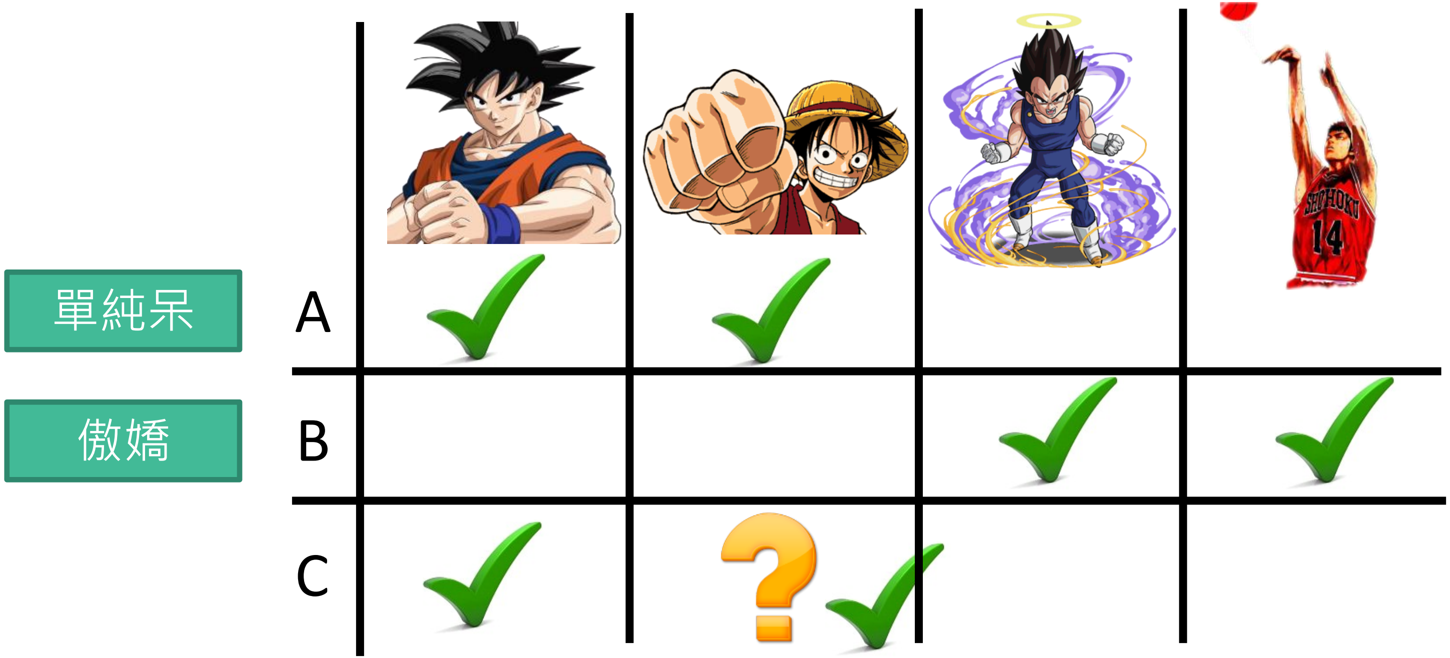
Latent Factors for Handwritten Digits



Latent Factors for Documents



Latent Factors for Recommendation System



Latent Factor Exploitation

- Handwritten digits

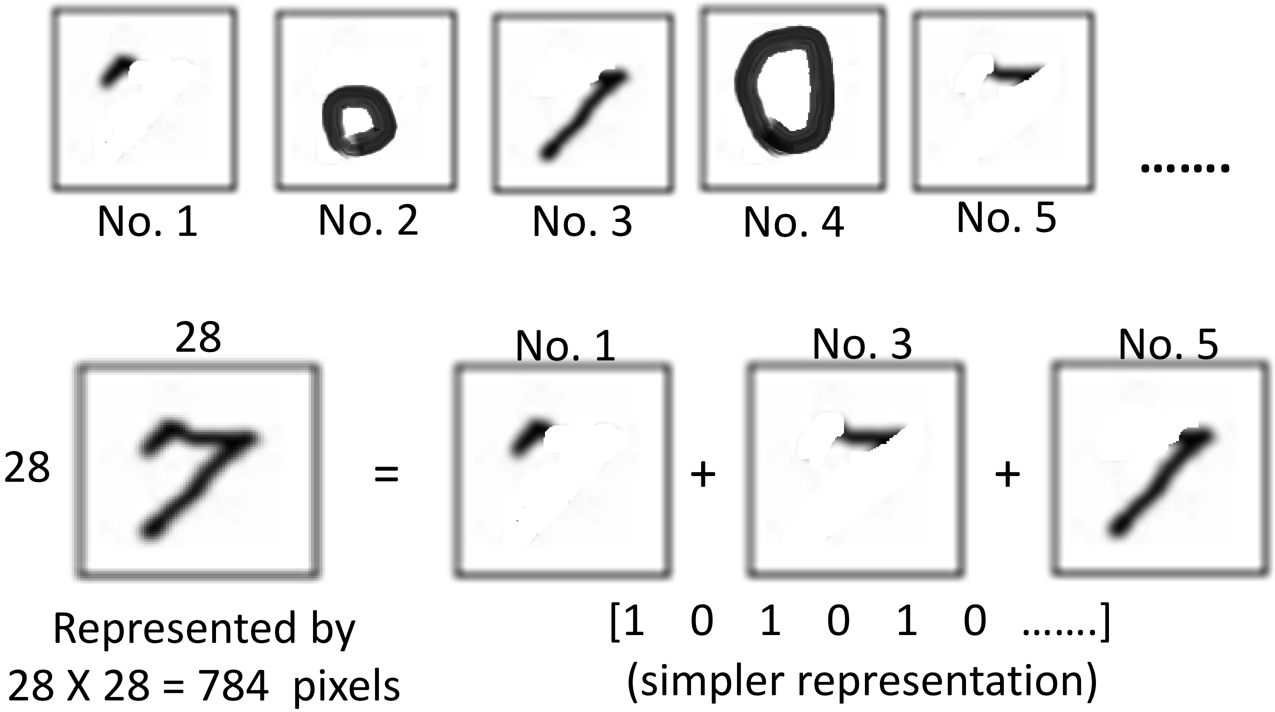
The handwritten images are composed of **strokes**

<u>Strokes (Latent Factors)</u>

No. 1 No. 2 No. 3 No. 4 No. 5 …….

Latent Factor Exploitation

Strokes (Latent Factors)



16

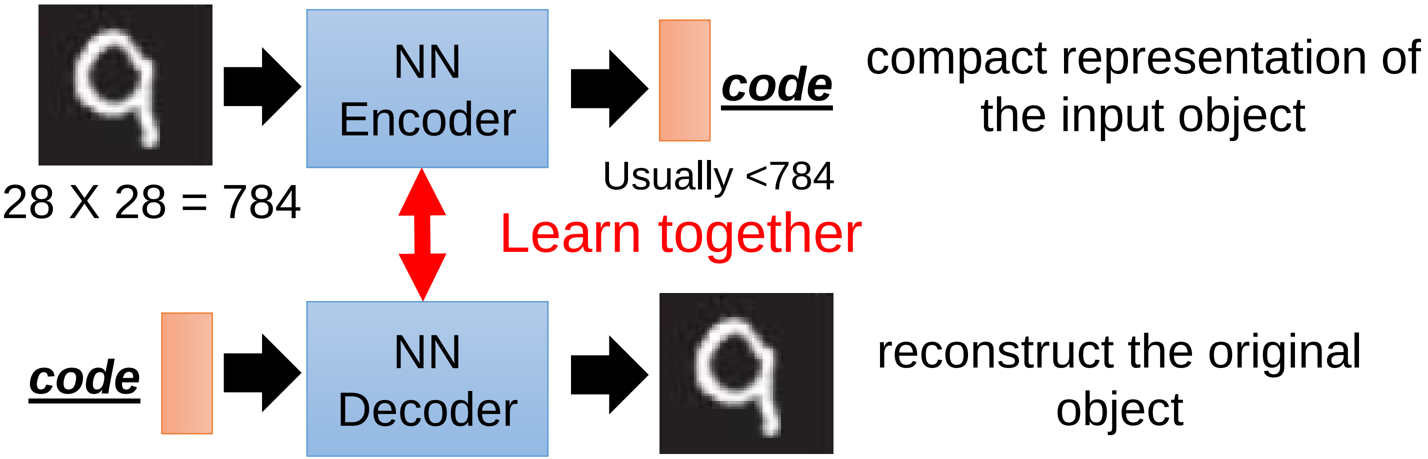
Autoencoder

Representation Learning

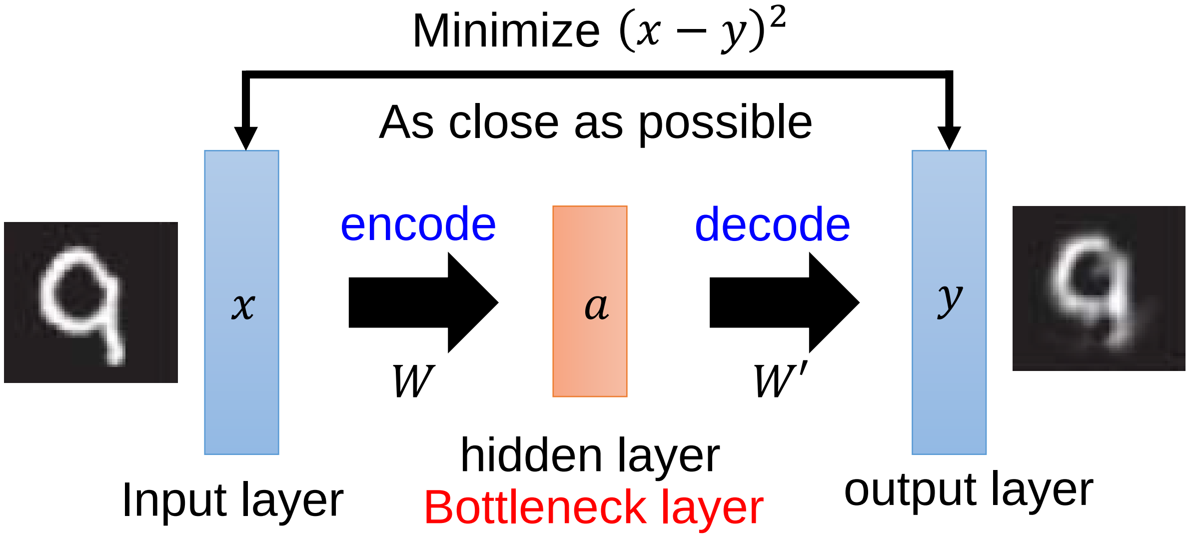
Autoencoder

- Represent a digit using 28 X 28 dimensions
- Not all 28 X 28 images are digits

Idea: represent the images of digits in a more compact way



Autoencoder

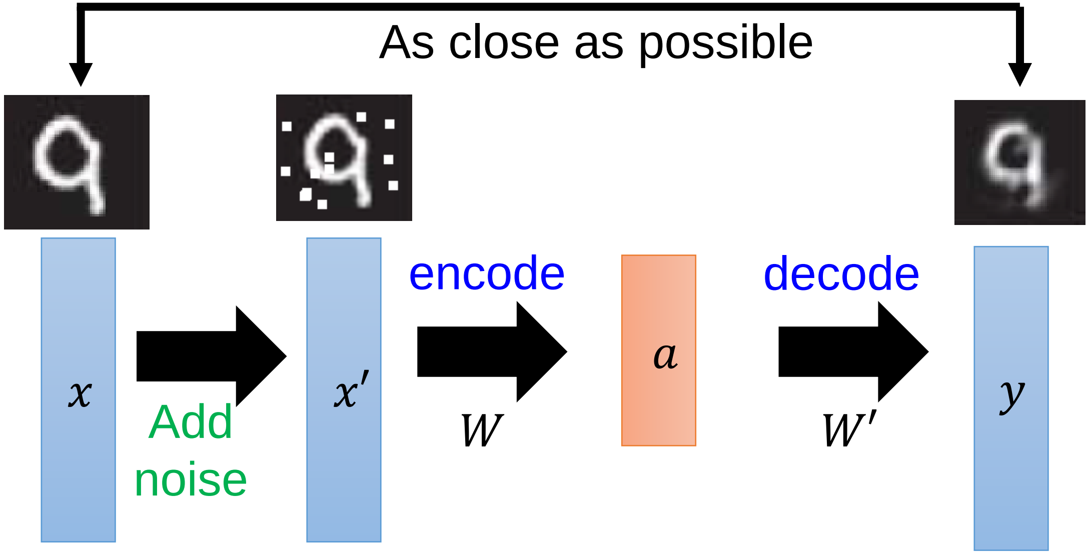


$a = \sigma(Wx + b)$ $y = \sigma(W'a + b')$

Output of the hidden layer is the code

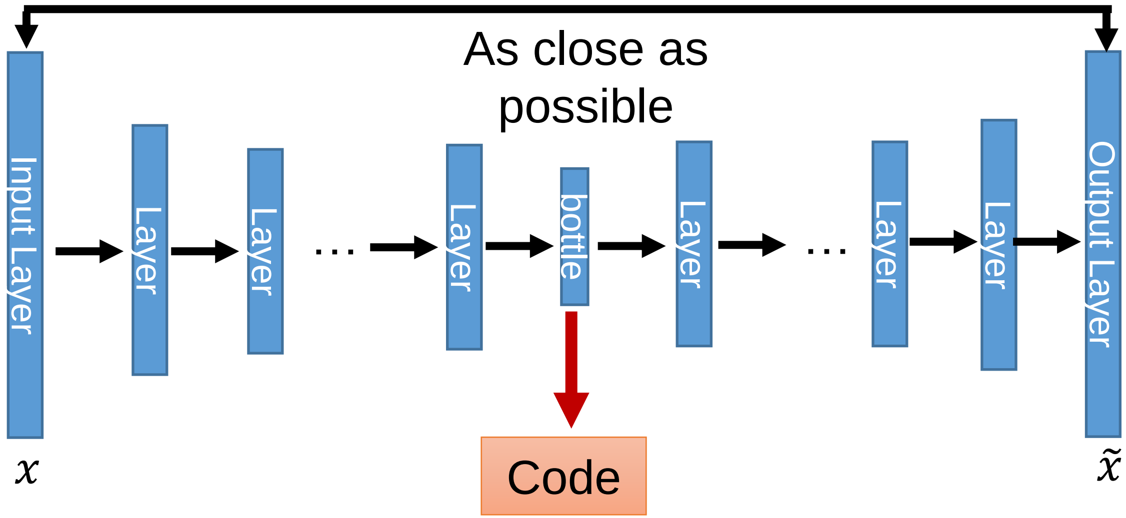
Autoencoder

- De-noising auto-encoder



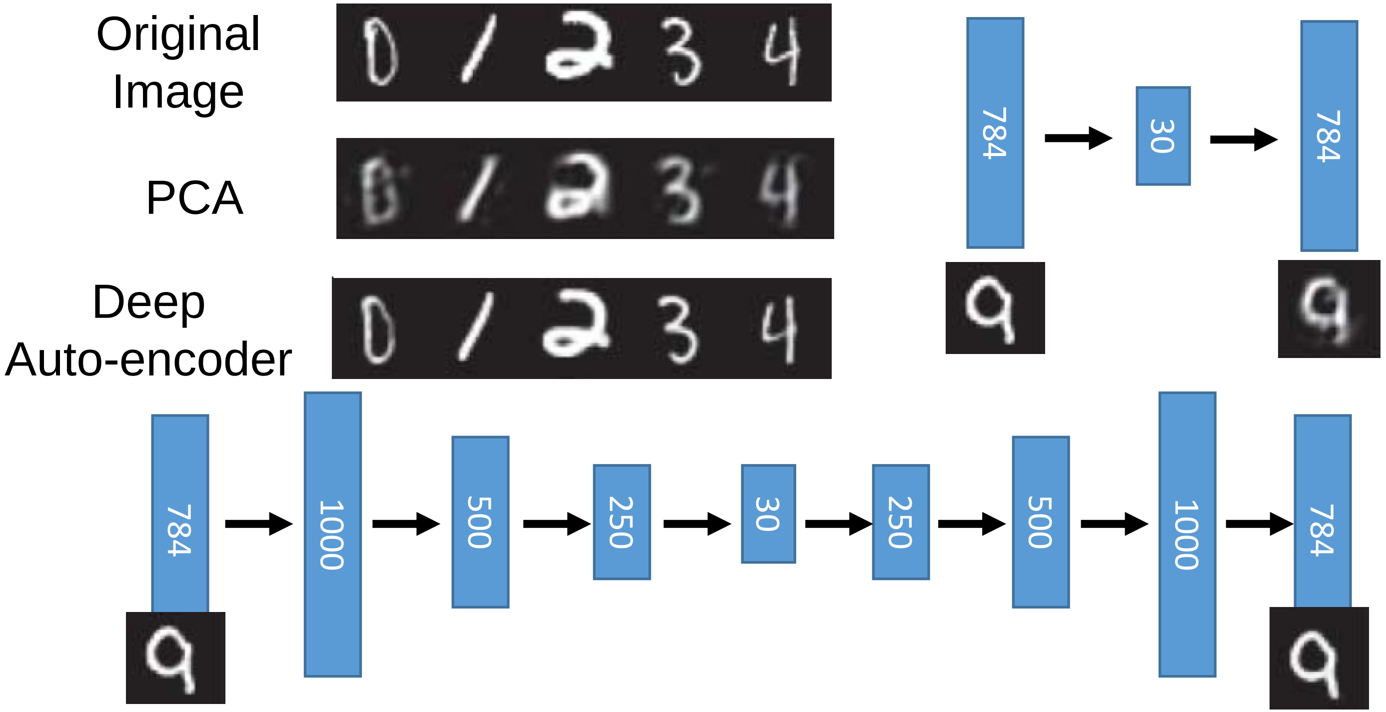
Rifai, et al. "Contractive auto-encoders: Explicit invariance during feature extraction,“ in *ICML*, 2011.

Deep Autoencoder

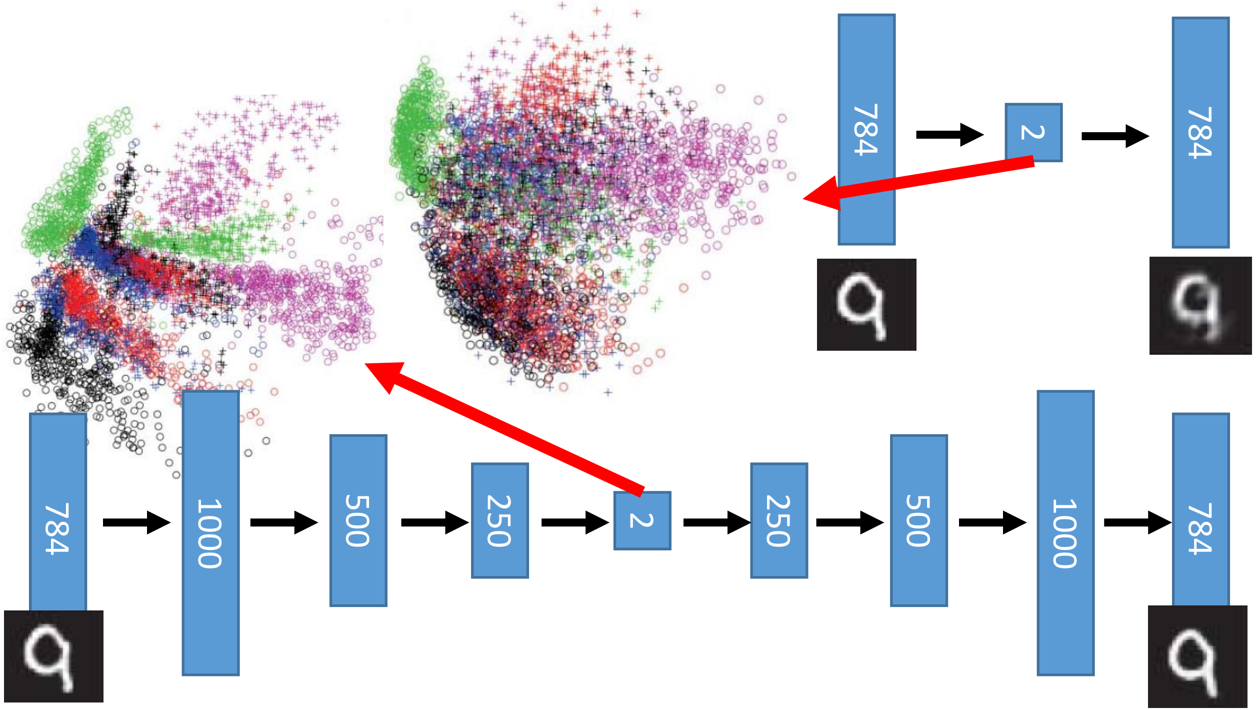


Hinton and Salakhutdinov. "Reducing the dimensionality of data with neural networks," *Science,* 2006.

Deep Autoencoder

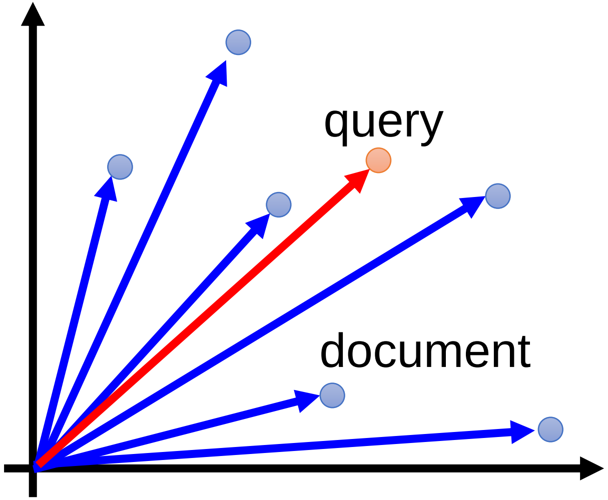


Feature Representation



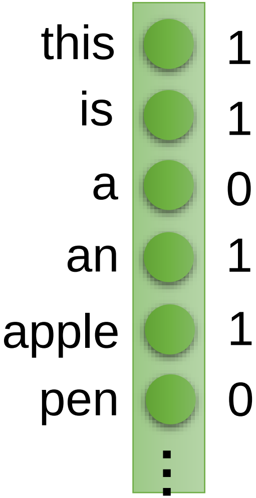
Auto-encoder – Text Retrieval

Vector Space Model



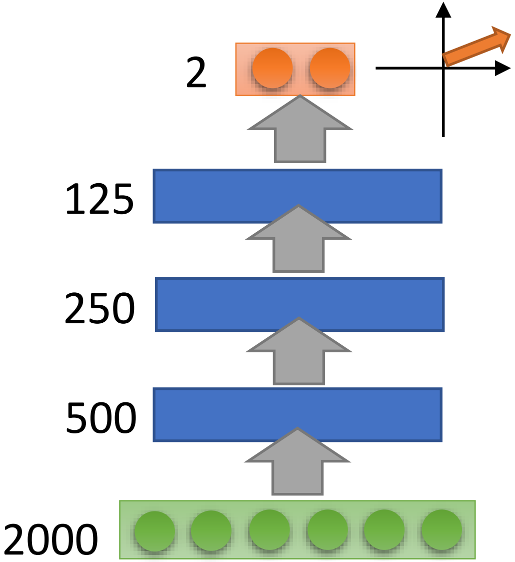
Bag-of-word

word string:
“This is an apple”

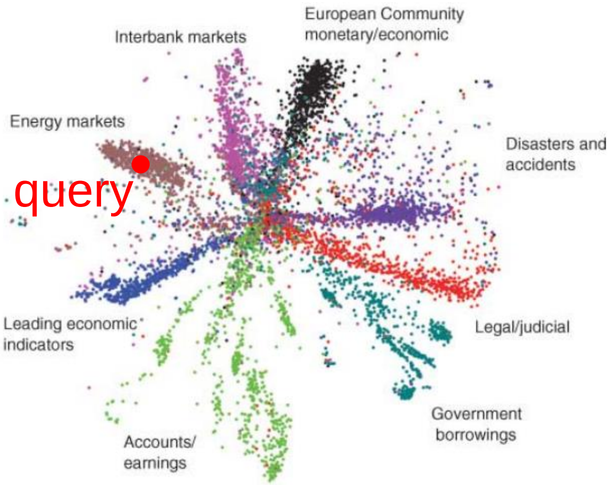


Semantics are not considered

Autoencoder – Text Retrieval



Bag-of-word (document or query)



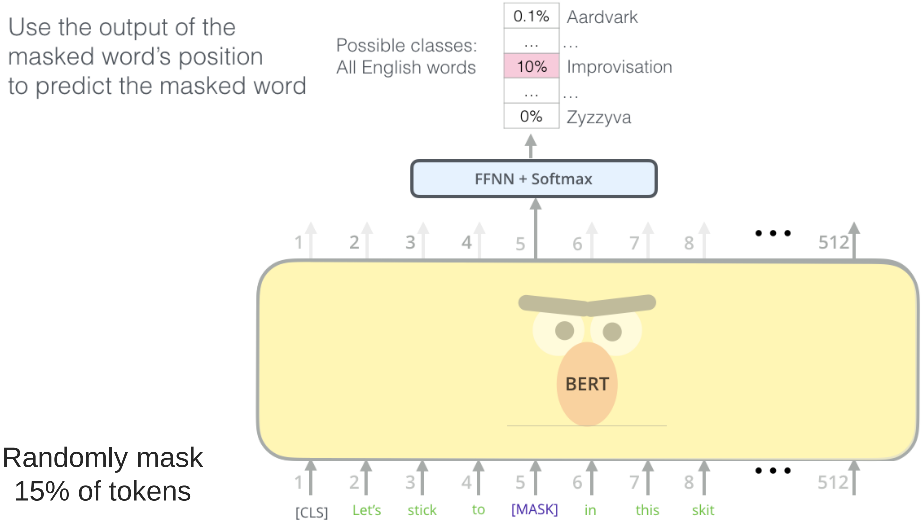
The documents talking about the same thing will have close code

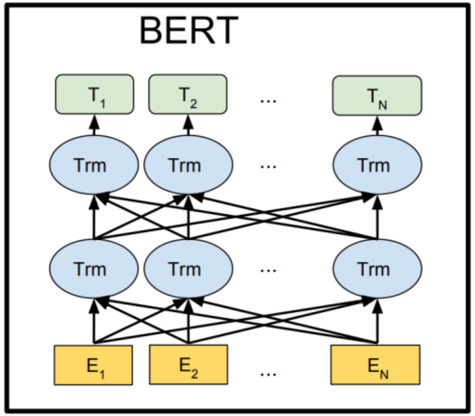
Auto-Encoding (AE)

- Objective: reconstructing $\bar{x}$ from $\hat{x}$

$$\max_{\theta} \quad \log p_{\theta}(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_{\theta}(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_{\theta}(\hat{\mathbf{x}})_t^{\top} e(x_t)\right)}{\sum_{x'} \exp\left(H_{\theta}(\hat{\mathbf{x}})_t^{\top} e(x')\right)}$$

  - dimension reduction or denoising (masked LM)



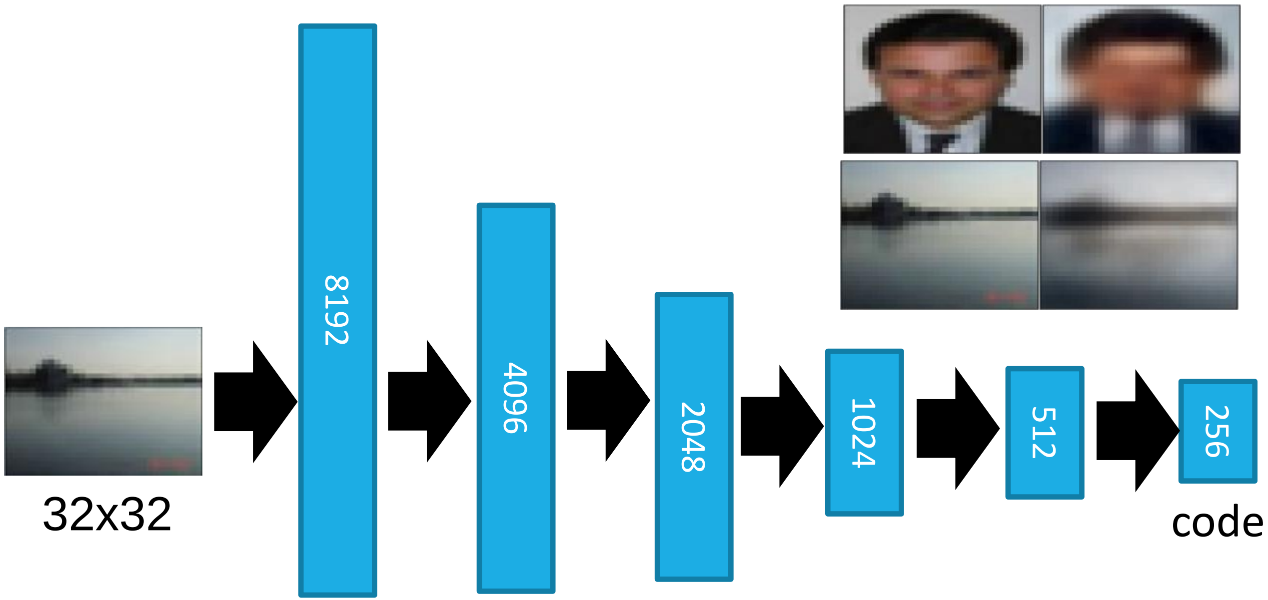


Autoencoder – Similar Image Retrieval

- Retrieved using Euclidean distance in pixel intensity space

Krizhevsky et al. "Using very deep autoencoders for content-based image retrieval," in *ESANN*, 2011.

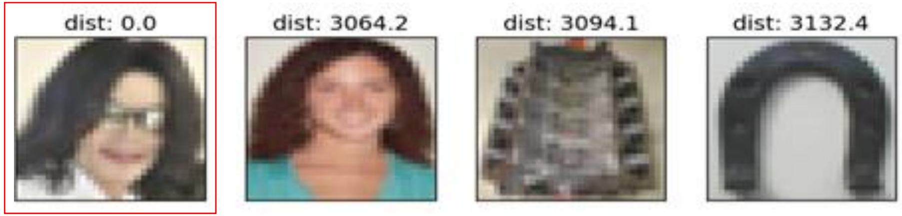
Autoencoder – Similar Image Retrieval



(crawl millions of images from the Internet)

Autoencoder – Similar Image Retrieval

- Images retrieved using Euclidean distance in pixel intensity space

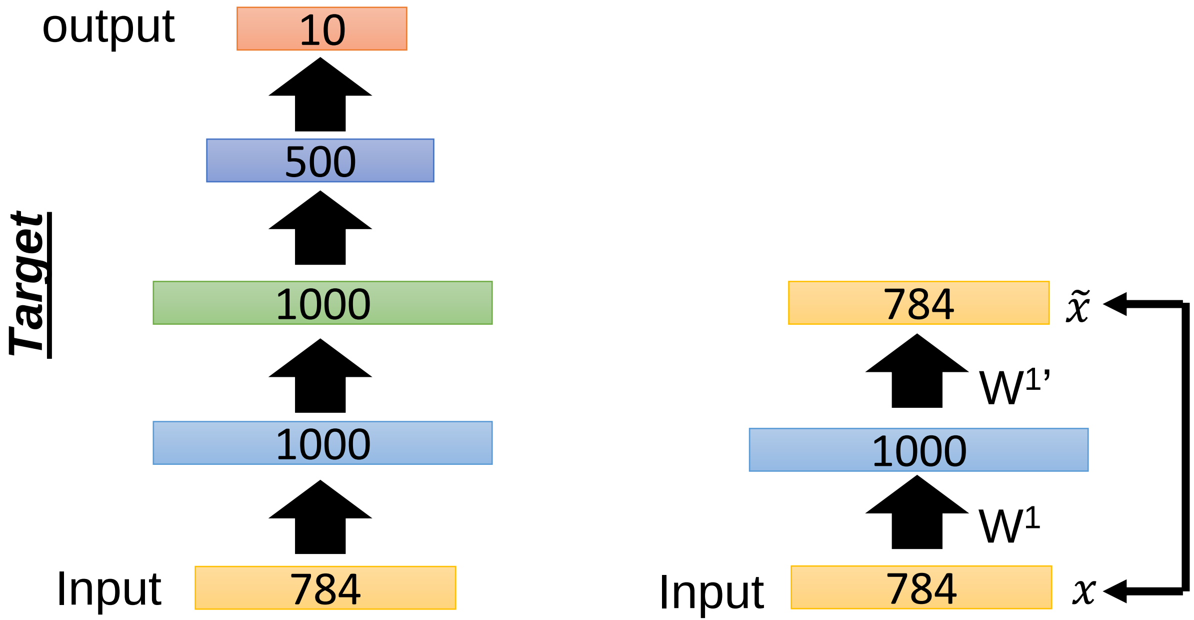


- Images retrieved using 256 codes

Learning the useful latent factors

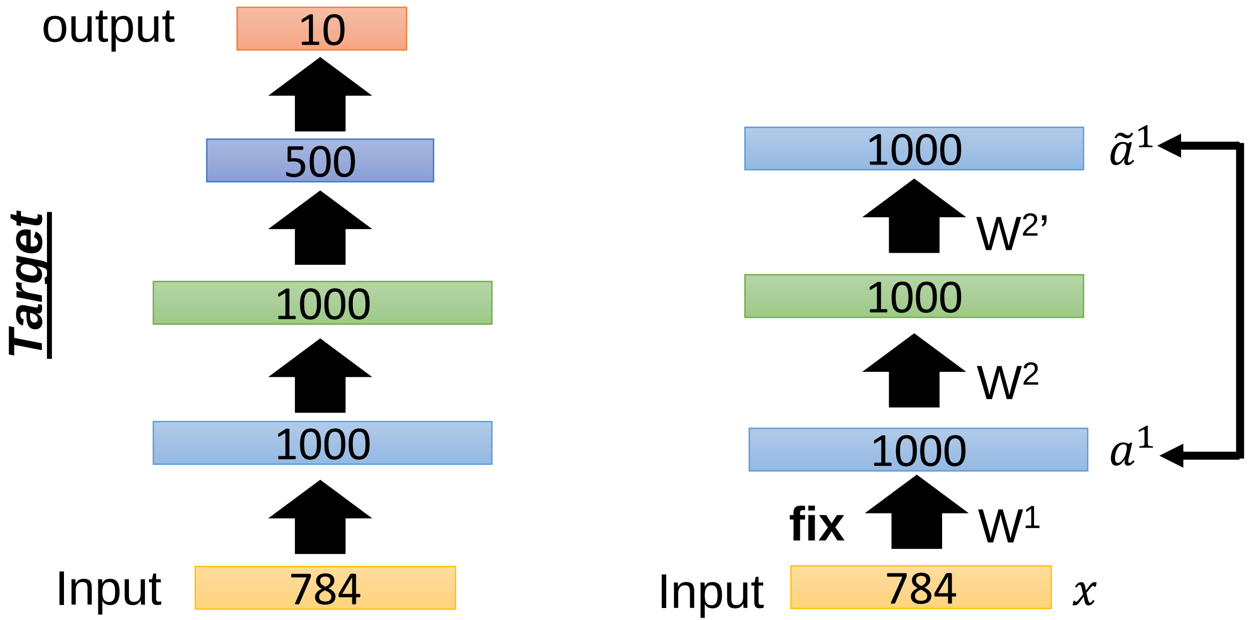
Autoencoder for DNN Pre-Training

- Greedy layer-wise pre-training *again*



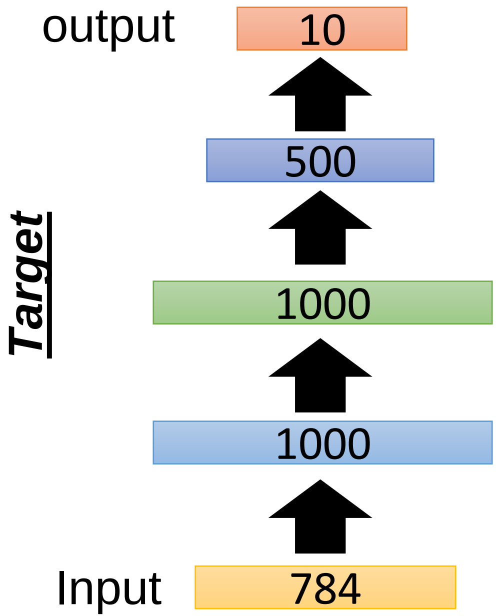
Autoencoder for DNN Pre-Training

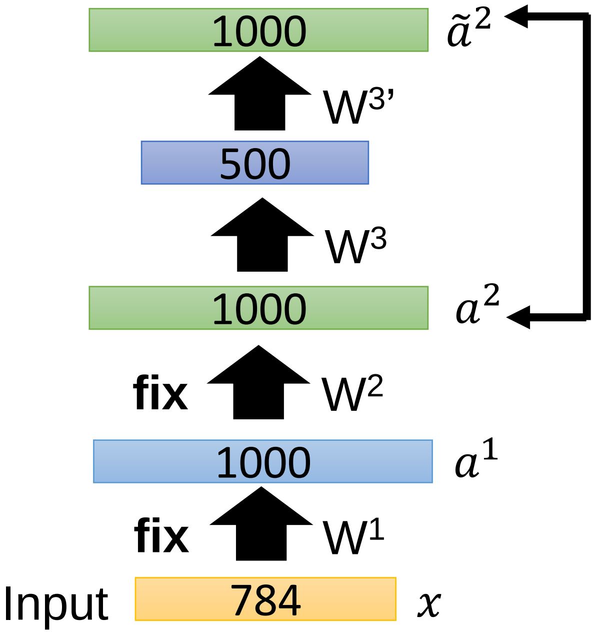
- Greedy layer-wise pre-training *again*



Autoencoder for DNN Pre-Training

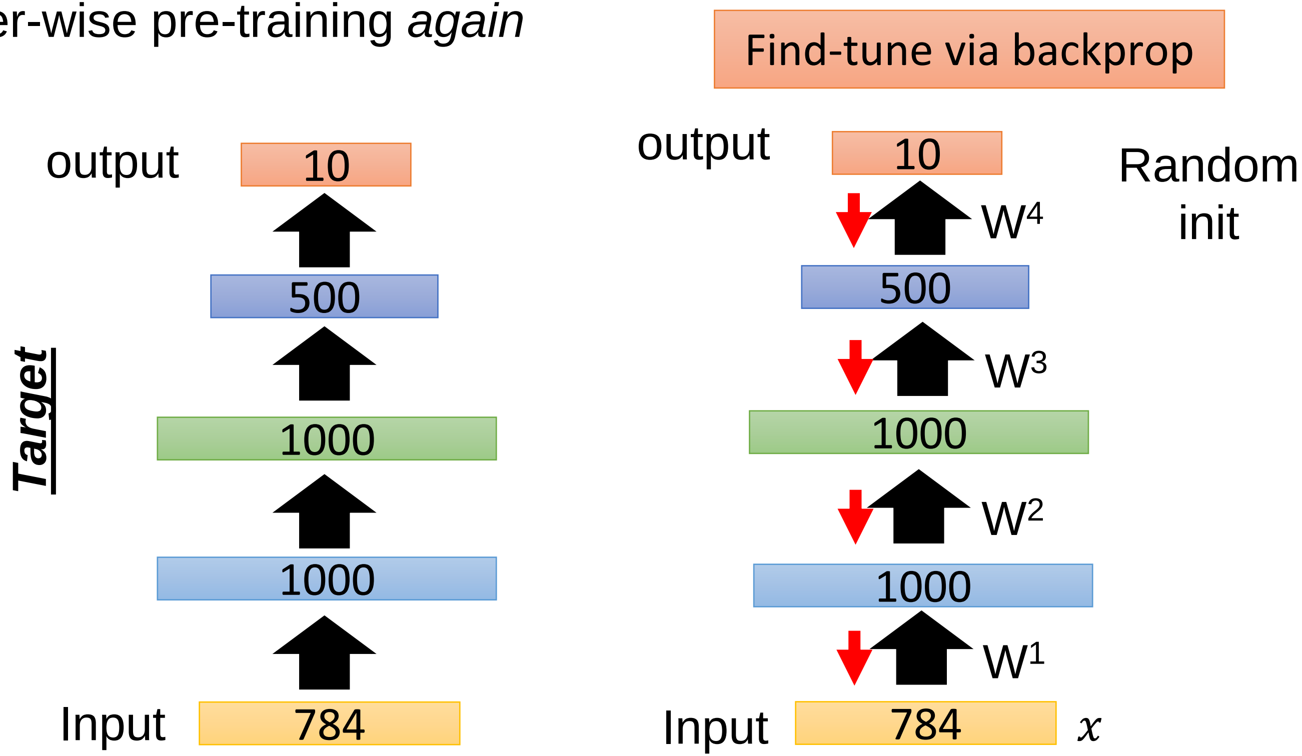
- Greedy layer-wise pre-training *again*





Autoencoder for DNN Pre-Training

- Greedy layer-wise pre-training *again*

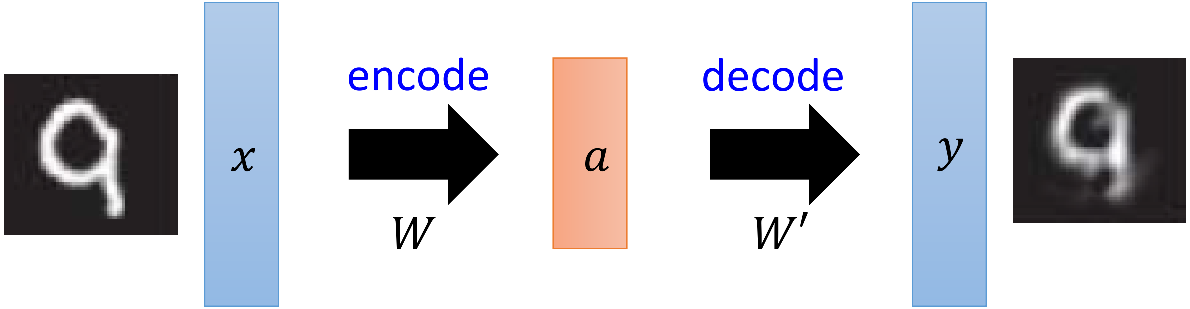


33

Variational Autoencoder

Representation Learning and Generation

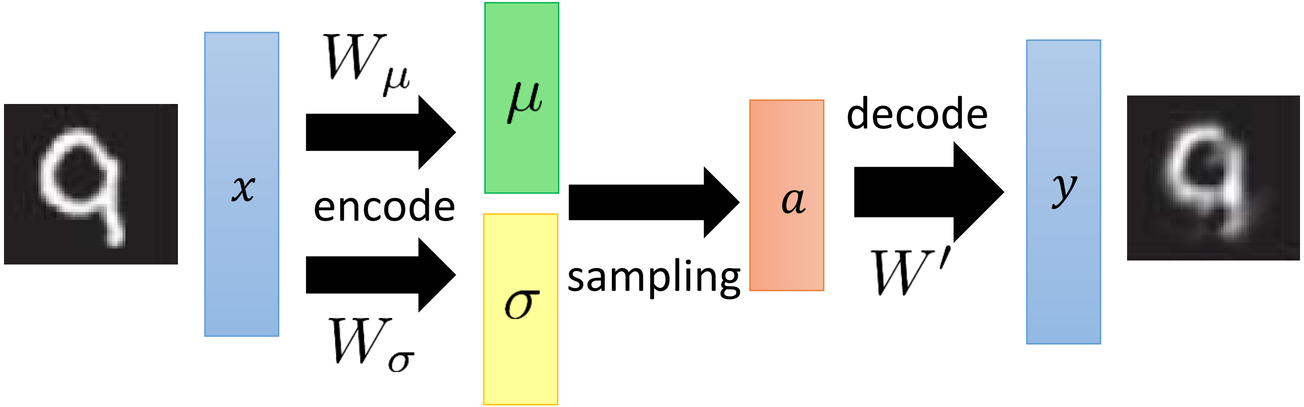
Generation from Latent Codes



How can we set a latent code for generation?

Latent Code Distribution Constraints

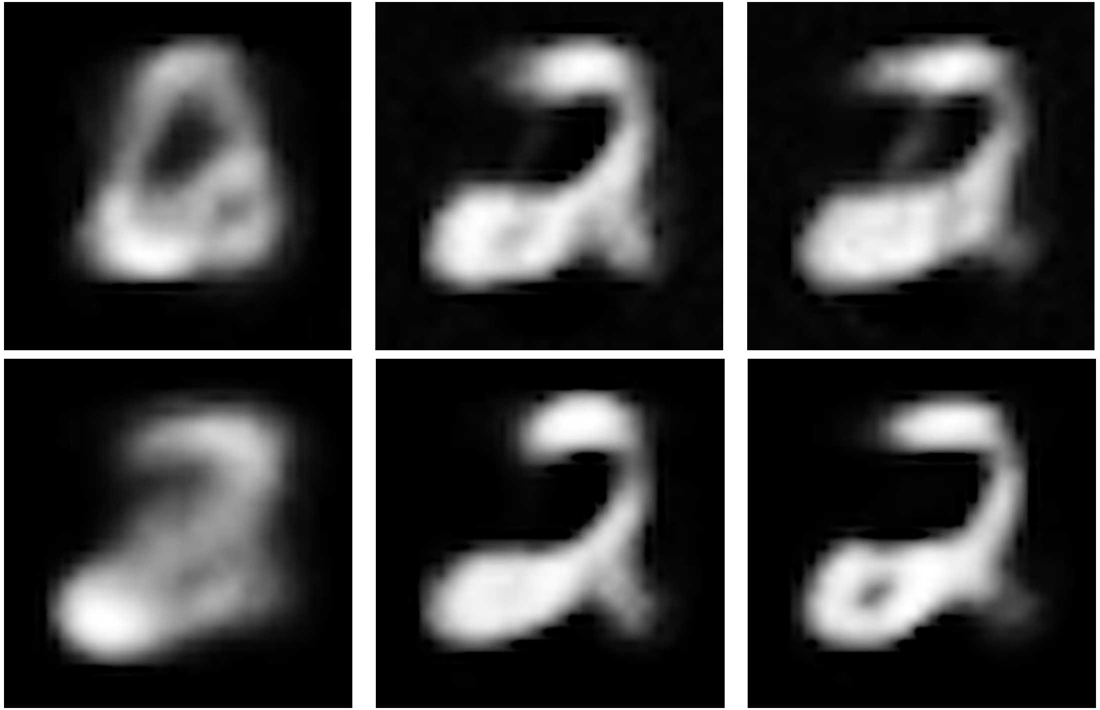
- Constrain the data distribution for learned latent codes
- Generate the latent code via a prior distribution



Reconstruction

AE

VAE

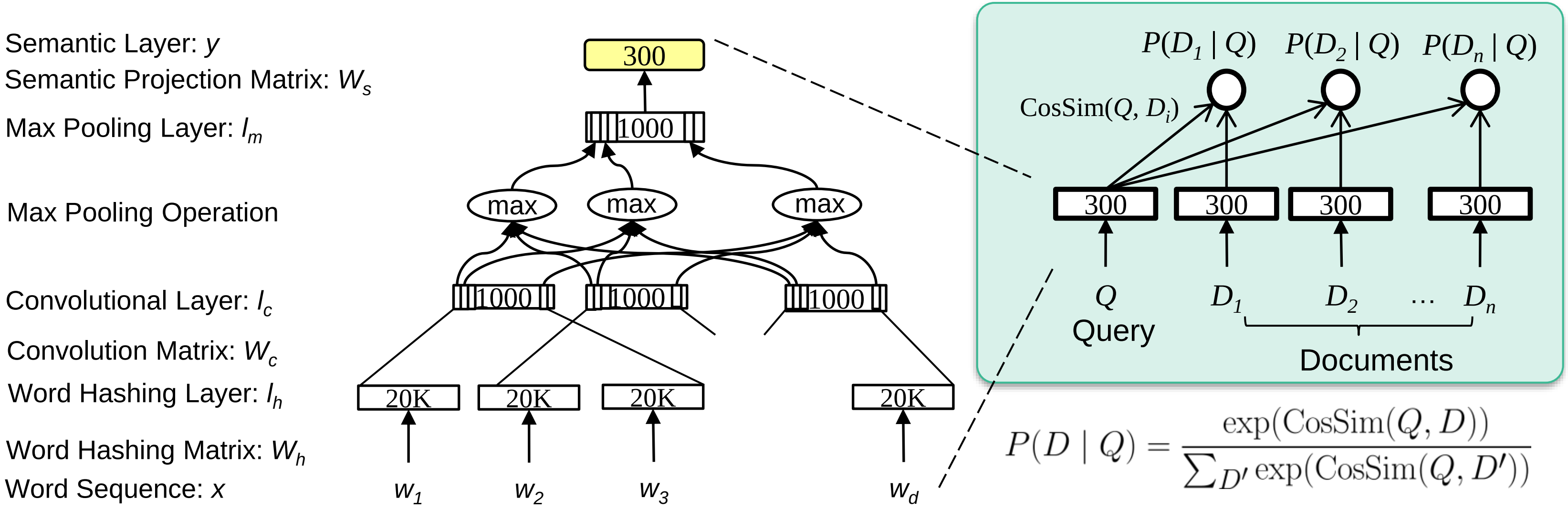


37

Distant Supervision

Representation Learning by Weak Labels

Convolutional Deep Structured Semantic Models (CDSSM/DSSM)



$$P(D \mid Q) = \frac{\exp(\mathrm{CosSim}(Q, D))}{\sum_{D'} \exp(\mathrm{CosSim}(Q, D'))}$$

$$\boxed{\Lambda(\theta) = \log \prod_{(Q,D^+)} P(D^+ \mid Q)}$$

Semantically related documents are close to the query in the encoded space

maximizes the likelihood of clicked documents given queries

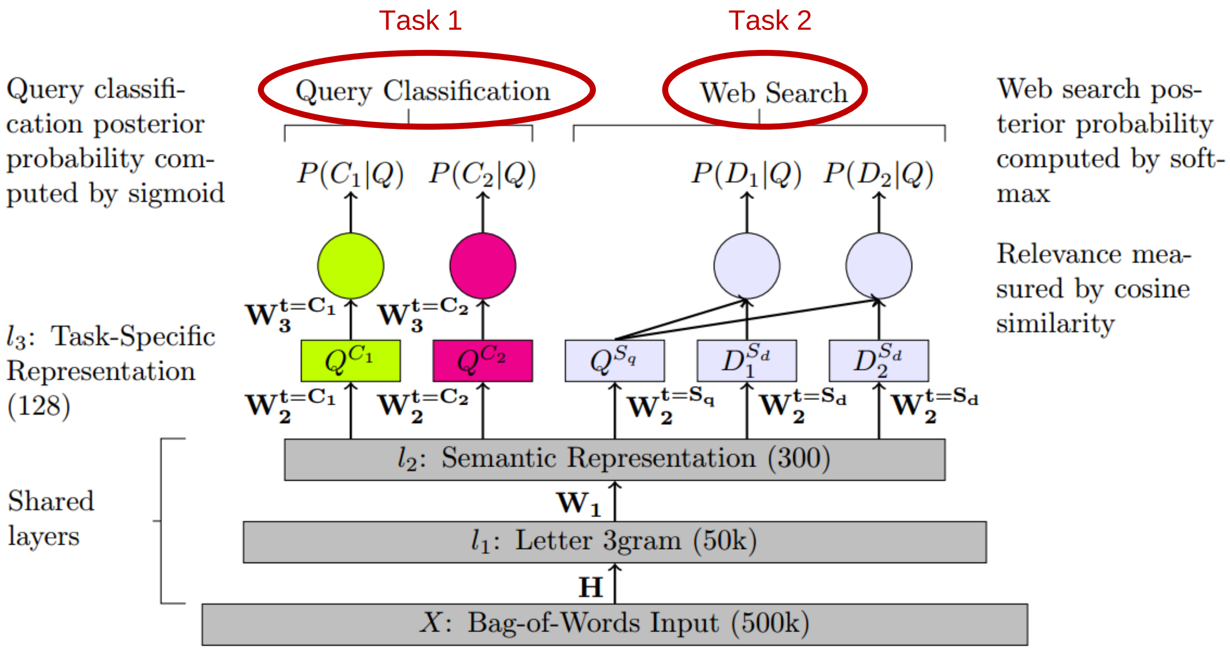
Huang et al., "Learning deep structured semantic models for web search using clickthrough data," in *Proc. of CIKM*, 2013.
Shen et al., “Learning semantic representations using ´ convolutional neural networks for web search," in *Proc. of WWW*, 2014.

39

Multi-Task Learning

Representation Learning by Different Tasks

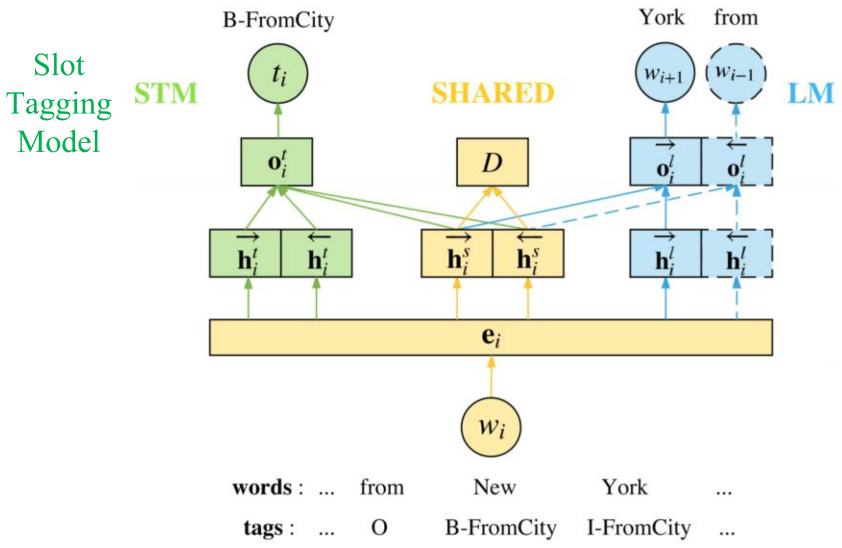
Task-Shared Representation



The latent factors can be learned by different tasks

Semi-Supervised Multi-Task SLU (Lan et al., 2018)

- Idea: language understanding objective can enhance other tasks



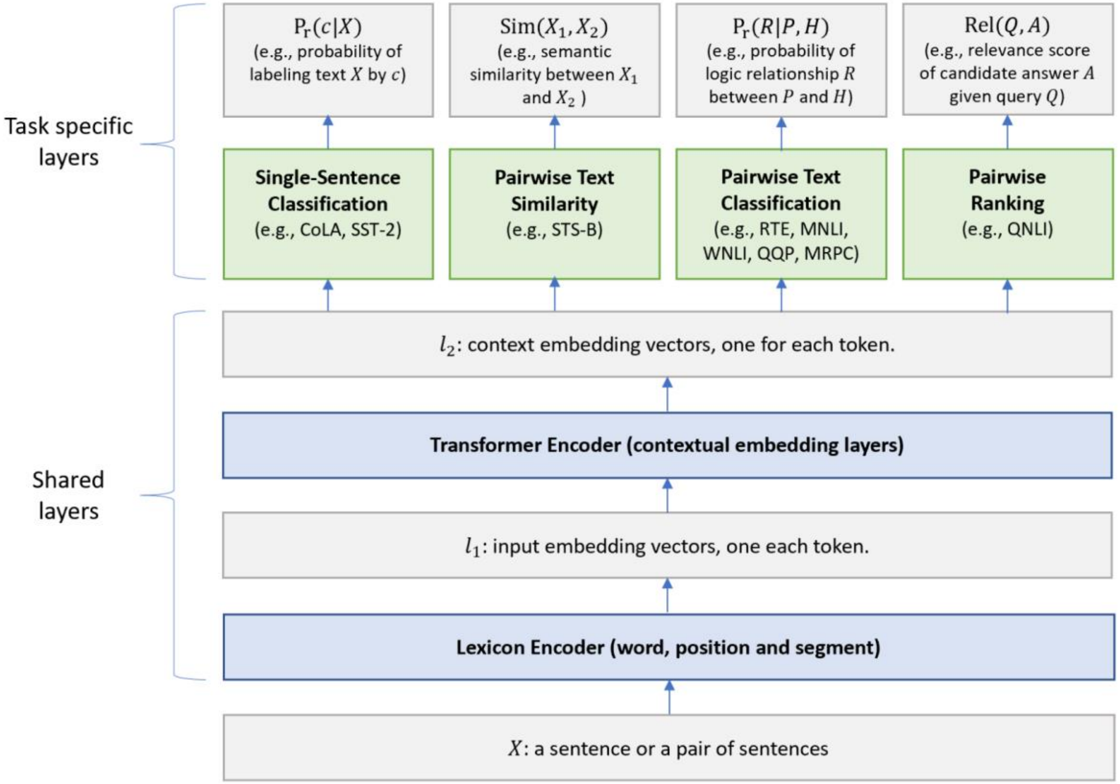
Algorithm 1: Adversarial Multi-task Learning for SLU

Input : Labeled training data $\{(\mathbf{w}^l, \mathbf{t}^l)\}$
Unlabeled data $\{\mathbf{w}^u\}$
Output: Adversarially enhanced slot tagging model

Initialize parameters $\{\theta^s, \theta^t, \theta^l, \theta^d\}$ randomly.
repeat
  `/* Sample from {(`$\mathbf{w}^l, \mathbf{t}^l$`)} */`
  Train the STM and shared model by Eq.(8).
  Train the task discriminator and the shared model by Eq.(6) or Eq.(7) as slot tagging task ($y = 1$).
  `/* Sample from {`$\mathbf{w}^l$`} and {`$\mathbf{w}^u$`} */`
  Train the LM and shared models by Eq.(9) (and Eq.(10) for BLM).
  Train the task discriminator and the shared model by Eq.(6) or Eq.(7) as LM task ($y = 0$).
until *convergence*;

BLM exploits the *unsupervised knowledge*, the *shared-private framework* and *adversarial training* make the slot tagging model more generalized

MT-DNN (Liu et al., 2019)



Algorithm 1: Training a MT-DNN model.

Initialize model parameters Θ randomly.
Pre-train the shared layers (i.e., the lexicon encoder and the transformer encoder).
Set the max number of epoch: $epoch_{max}$.
//Prepare the data for T tasks.
for t *in* $1, 2, ..., T$ **do**
  Pack the dataset t into mini-batch: D_t.
end
for $epoch$ *in* $1, 2, ..., epoch_{max}$ **do**
  1. Merge all the datasets:
  $D = D_1 \cup D_2 ... \cup D_T$
  2. Shuffle D
  for b_t *in* D **do**
    //b_t is a mini-batch of task t.
    3. Compute loss : $L(\Theta)$
      $L(\Theta)$ = Eq. 6 for classification
      $L(\Theta)$ = Eq. 7 for regression
      $L(\Theta)$ = Eq. 8 for ranking
    4. Compute gradient: $\nabla(\Theta)$
    5. Update model: $\Theta = \Theta - \epsilon\nabla(\Theta)$
  end
end

https://github.com/namisan/mt-dnn

Concluding Remarks

- Labeling data is expensive, but we have large unlabeled data
- Autoencoder
  - exploits unlabeled data to learn latent factors as representations
  - learned representations can be transfer to other tasks
- Distant Labels / Labels from Other Tasks
  - learn the representations that are useful for other tasks
  - learned representations may be also useful for the target task