LinkedIn

Deep Transfer Learning for Search and Recommendation

WWW 2020, TAIPEI

Yang Yang

Sen Zhou

Jian Qiao

Bo Long

Agenda

1 **Introduction**

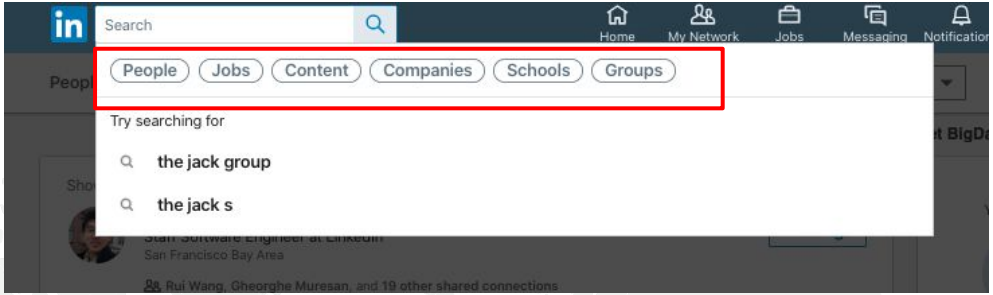
2 Deep Transfer Learning

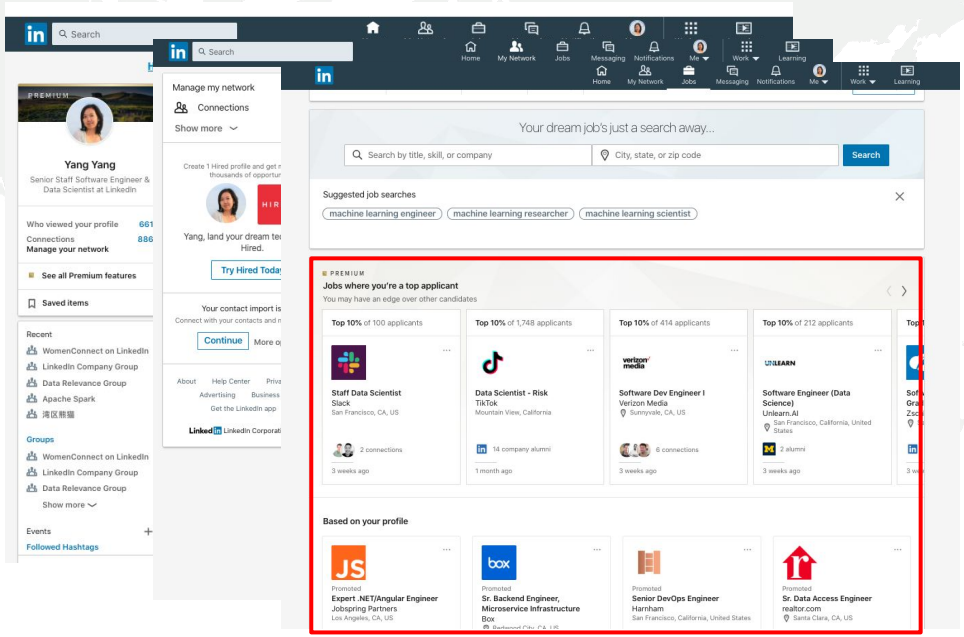
3 Deep Transfer Learning for Search and Recommendation

4 Applications at LinkedIn

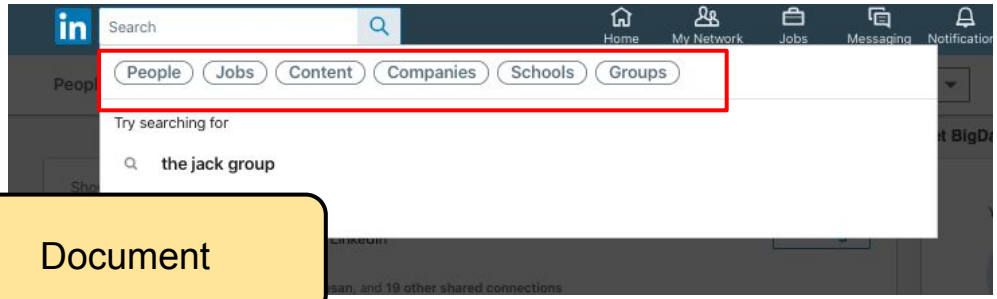
Search and Recommendation

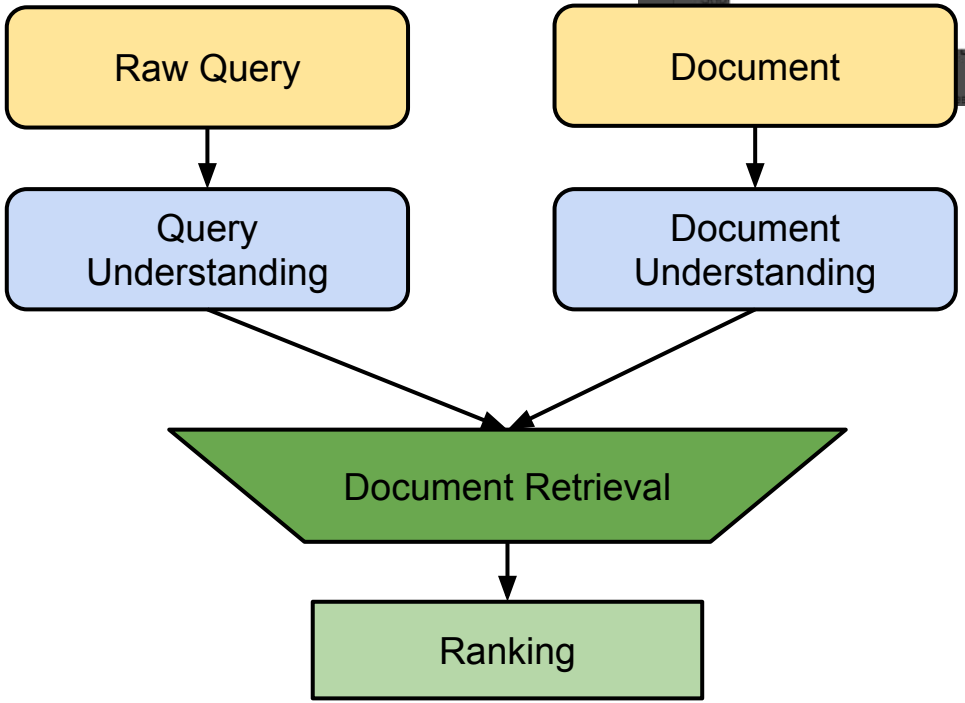
Are Everywhere



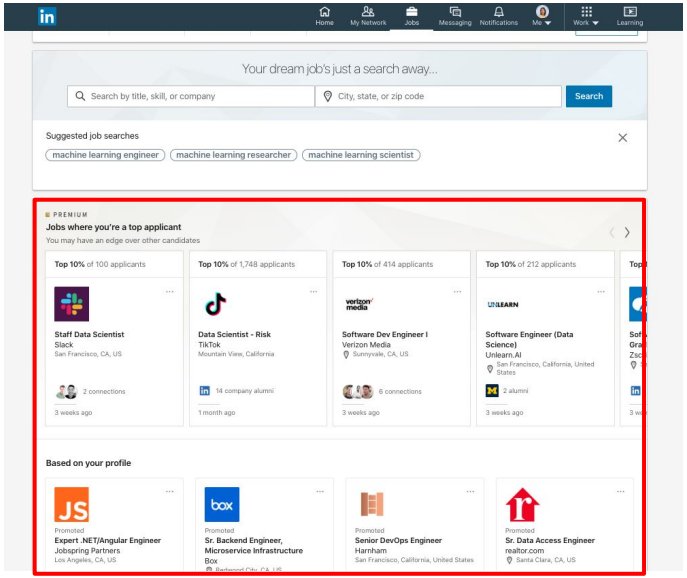


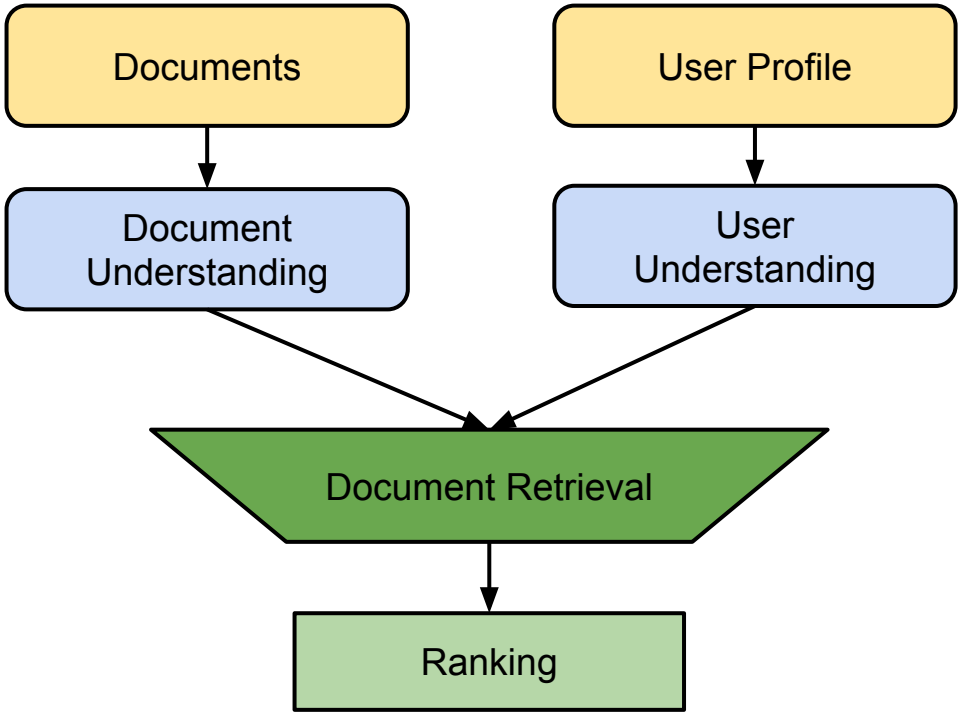
Search


Search
Home
My Network
Jobs
Messaging
Notifications
People
Jobs
Content
Companies
Schools
Groups
Try searching for
the jack group

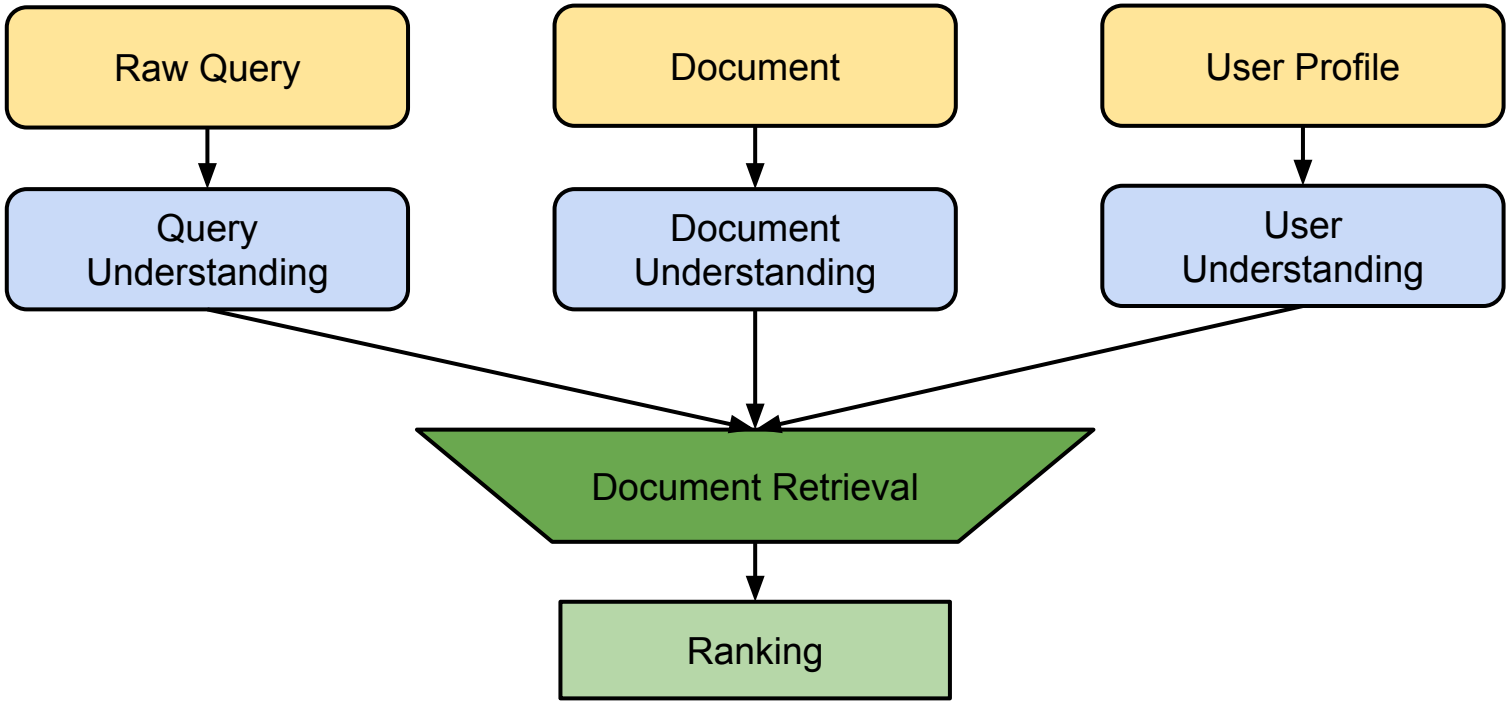

Raw Query
Document
Query Understanding
Document Understanding
Document Retrieval
Ranking

Recommendation





Overview of Search and Recommendation Systems



Why Do We Need Transfer Learning for Search and Recommendation Systems?

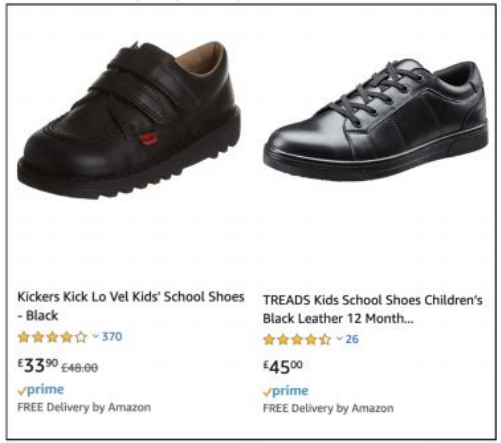
Improve learning performance for systems with sparse training data

Italian Products With Limited Search Data

Query: *scarpe scuola ragazzi*
Platform language: *Italian*

US Products With Extensive Search Data

Query: *school shoes for boys*
Platform language: *English*

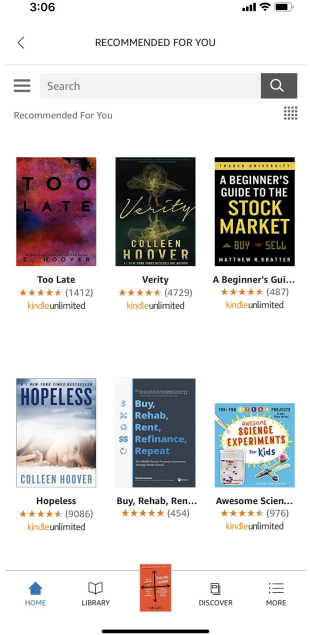


Source: [Ahuja et al., 2020]

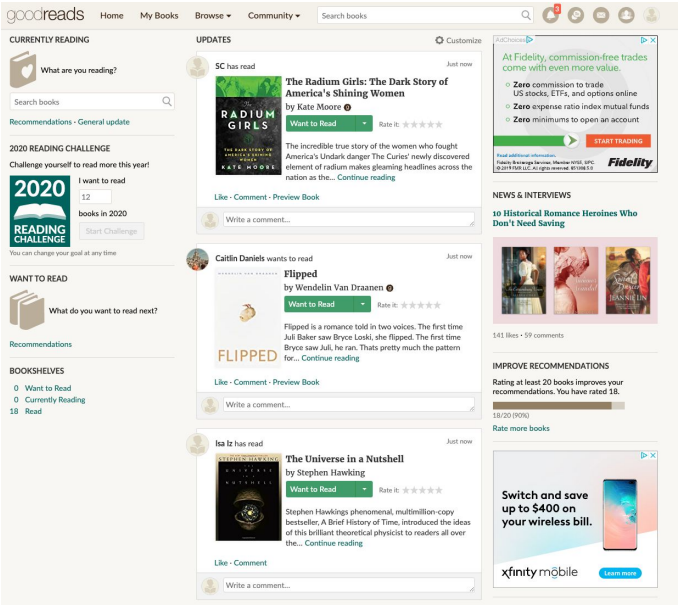
Why Do We Need Transfer Learning for Search and Recommendation Systems?

Bridge the generalization gap from the related systems to a new one

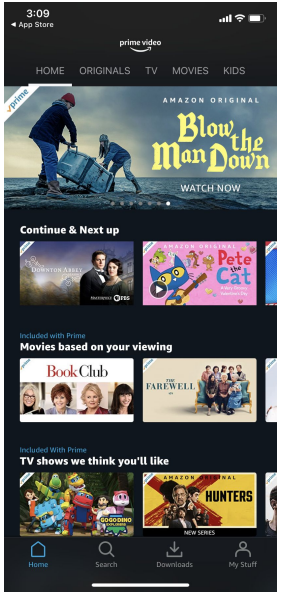
Book Recommendation



Existing Book Review Data From Another System



Existing Movie Review Data From Another System



How Do Human Transfer Knowledge?

Python for JAVA Developers: Basics V 1.2

Referring python 3.7.1

1 Basic Syntax

1.1 End of Statements

Unlike the Java, to end a statement in Python, we don't have to type in a semicolon, you simply press Enter. But semicolons can be used to delimit statements if you wish to put multiple statements on the same line.

```
message1 = 'Hello World!'
message2 = "Python gives no missing semicolon error!"

# Instead of System.out.print, we use print
print (message1) # print 'Hello World!' on the console output
print ("Hello"); print ("Python!"); # usage of the semicolon
```

1.2 Code Blocks and Indentation

One of the most distinctive features of Python is its use of indentation to mark blocks of code. Consider the following if-statement from our non-zero number checking program:

JAVA

```
if (0 == value) {
   System.out.print("Number is Zero");
} else {
   System.out.print("Number is non-Zero.");
}

System.out.print("All done!");
```

Python

```
if 0 == value:
   print('Number is Zero')
else:
   print('Number is non-Zero.')

print('All done!')
```

To indicate a block of code in Python, you must indent each line of the block by the same amount. The two blocks of code in

You declare multiple variables by separating each variable name with a comma.

```
a, b = True, False
```

2.2 Assigning Values

```
a = 300
```

The same value can be assigned to multiple variables at the same time:

```
a = b = c = 1
```

And multiple variables can be assigned different values on a single line:

```
a, b, c = 1, 2, "john"
```

This is the same as:

```
a = 1
b = 2
c = "john"
```

3 Data Types

Python sets the variable type based on the value that is assigned to it. Unlike JAVA, Python will change the variable type if the variable value is set to another value.

```
var = 123 # This will create a number integer assignment
var = 'john' # the 'var' variable is now a string type.
```

3.1 Numbers

Most of the time using the standard Python number type is fine. Python will automatically convert a number from one type to another whenever required. We don't require to use the type casting

Learning another programming language is easier for human as we draw analogy from our past experience.

- Extract common concepts
- Acknowledge the difference

Source: https://medium.com/nestedif/cheatsheet-python-for-java-developers-98f75c94a1a

Transfer Learning v.s. Traditional Machine Learning

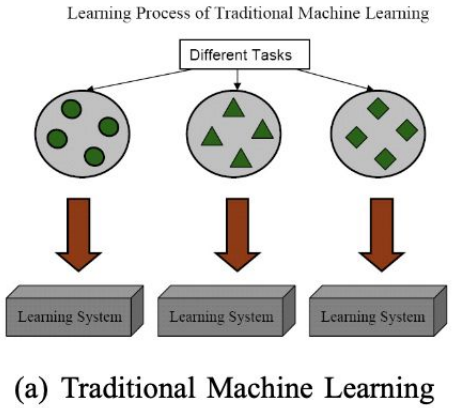
[Pan et al., 2009]

Traditional Learning

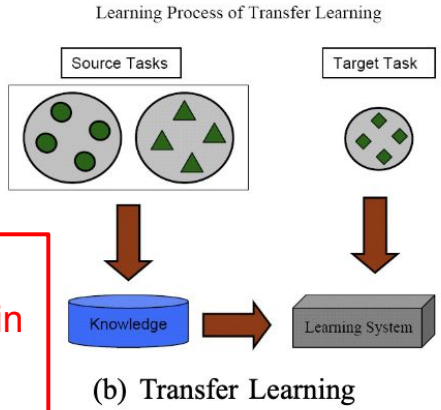
- Assume the learning systems are built upon the data collected for the problem

Transfer Learning

- Transfer the knowledge from the **source** domains (relevant problems) to the **target** domain (problem of interest)



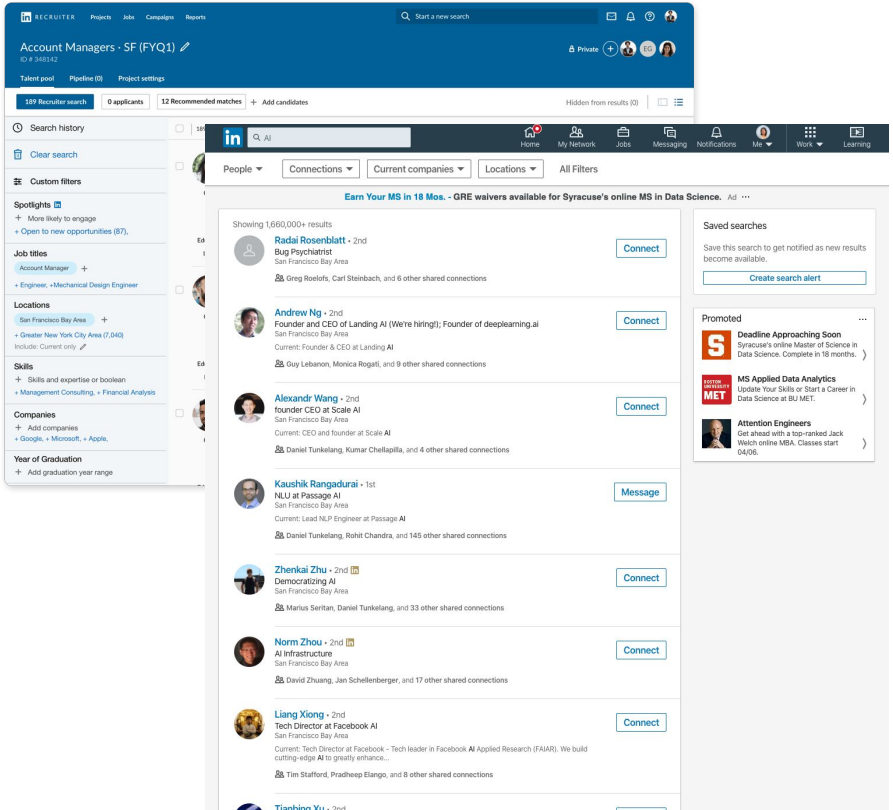
(a) Traditional Machine Learning



(b) Transfer Learning

Between source and target, the data domain and/or the task could be very different

Transfer Learning: An Example



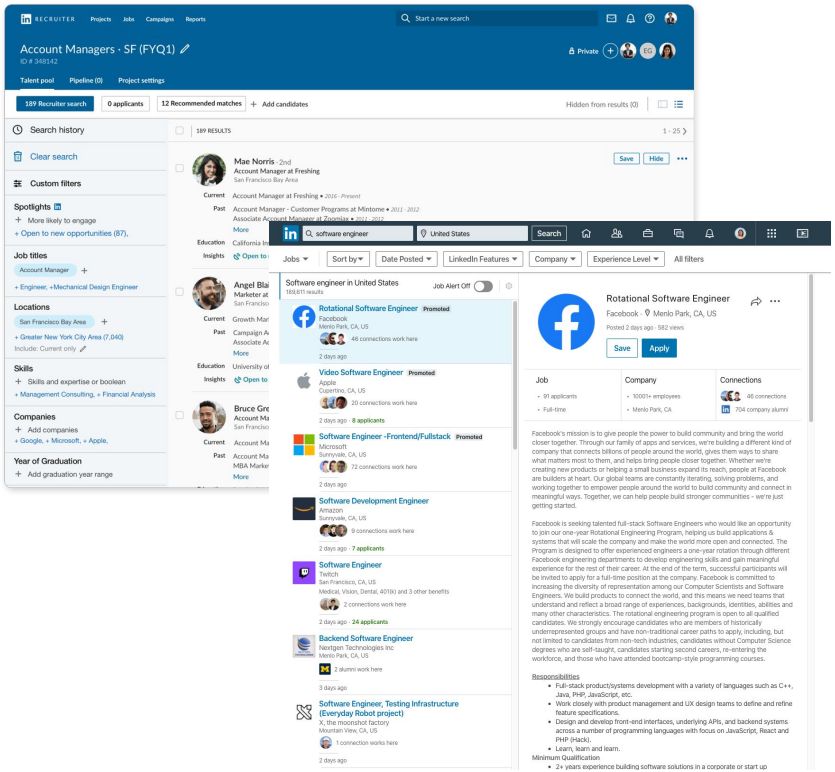
Target: Recruiter Search (a recruiter submits queries to search for job candidates)

- Domain D
  - X - features describing queries, members
  - P(X) - feature distribution
- Task T
  - Y - click/no click actions on the members in search result
  - P(Y|X) - Click probability

Source: People Search (a LinkedIn member submits queries to search for members).

- Domain D_s
  - X_s : features describing queries, members
  - $P(X_s)$: feature distribution
- Task T_s
  - Y_s - click/no click actions on the members in search result
  - $P(Y_s|X_s)$ - Click probability

Transfer Learning: An Example



Target: Recruiter Search (a recruiter submits queries to search for job candidates)

- Domain D
  - X - features describing queries, members
  - P(X) - feature distribution
- Task T
  - Y - click/no click actions on the members in search result
  - P(Y|X) - Click probability

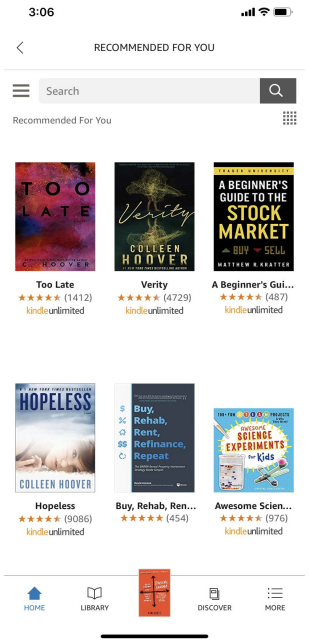
Source: Job Search (a LinkedIn member submits queries to search for jobs.)

- Domain D_s
  - X_s : features describing queries,jobs, members
  - $P(X_s)$: feature distribution
- Task T_s
  - Y_s - click/no click actions on the jobs in search result
  - $P(Y_s|X_s)$ - Click probability

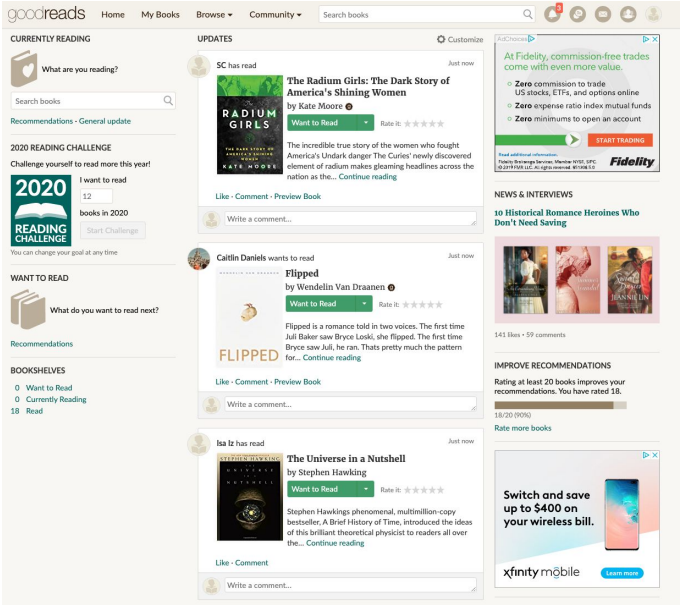
Transfer Learning Approaches

- Model Transfer:
  - Adapt learnt model parameters/structure

Book Recommendation



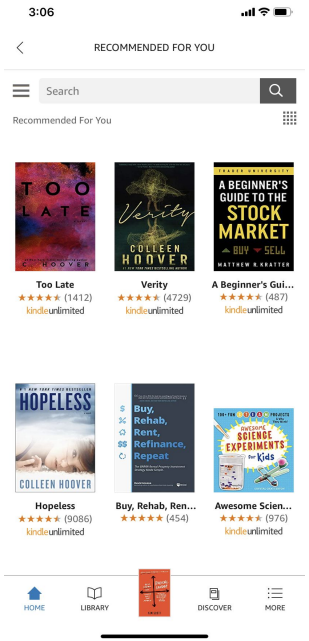
Existing Book Review Data From Another System



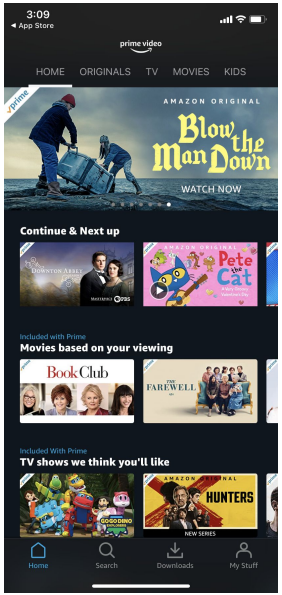
Transfer Learning Approaches

- Model Transfer:
  - Adapt learnt model parameters/structures
- Feature Representation Transfer:
  - Learning shareable features cross two domains.

Book Recommendation



Existing Movie Review Data From Another System

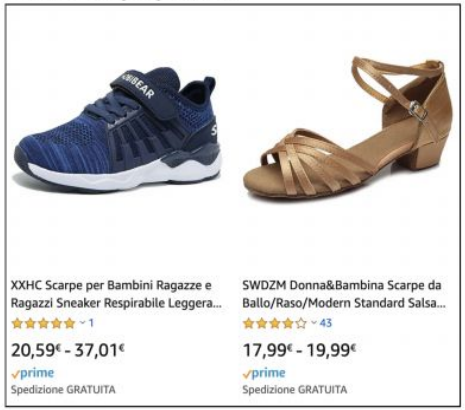


Transfer Learning Approaches

- Model Transfer:
  - Adapt learnt model parameters/structures
- Feature Representation Transfer:
  - Learning shareable features cross domains.
- Instance Transfer:
  - Borrow data instances for training

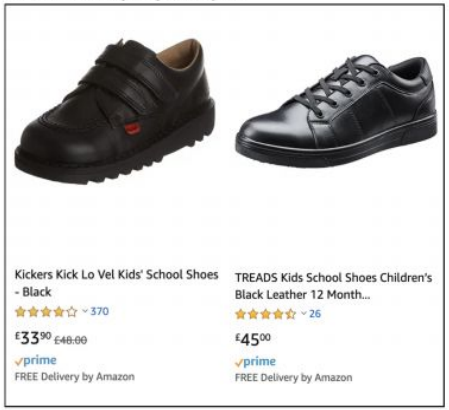
Italian Products With Limited Search Data

Query: *scarpe scuola ragazzi*
Platform language: *Italian*



US Products With Extensive Search Data

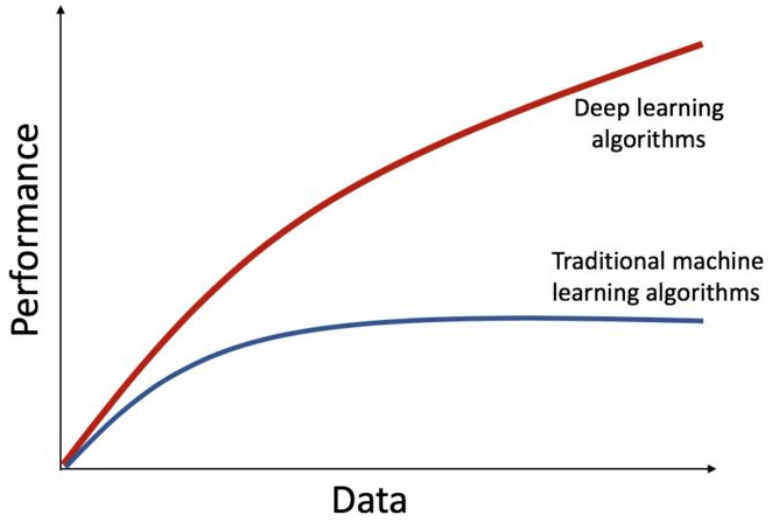
Query: *school shoes for boys*
Platform language: *English*



Why Deep Transfer Learning (DTL) ?

- With flexible and complicated network structure, DTL learns hidden transferable knowledge more effectively

- DTL can be better integrated with deep models that are widely used in search and recommendation applications.



Source: https://bluehexagon.ai/blog/what-is-deep-learning-and-how-is-it-different-from-machine-learning/

Agenda

1 Introduction

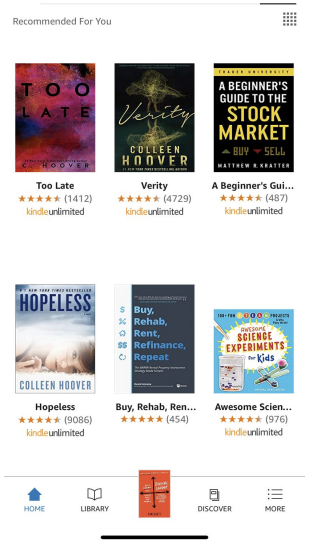
2 **Deep Transfer Learning**

3 Deep Transfer Learning for Search and Recommendation

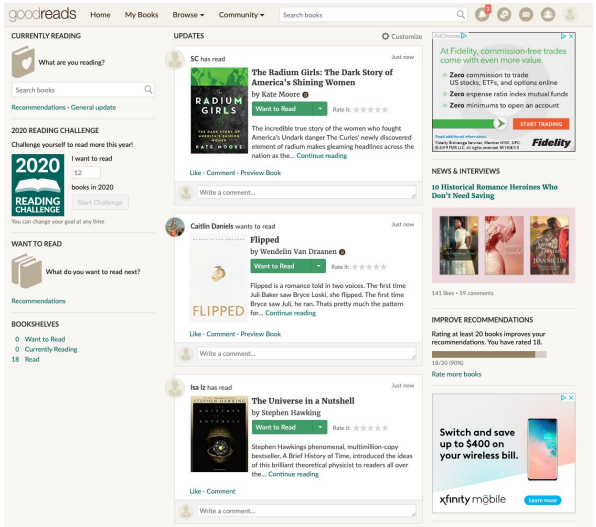
4 Applications at LinkedIn

Deep Transfer Learning Techniques

Target



Source



How do **deep learning techniques** bridge the generalization errors between target and source?

- Model parameters: share model parameters or mutual regularization
- Feature representation: minimize domain divergence in a shared feature space
- Instance: select or re-weight

The three types of transfer learning techniques are more inter-related instead of being well-categorized.

Deep Transfer Learning

Model Transfer

Feature Transfer

Instance Transfer

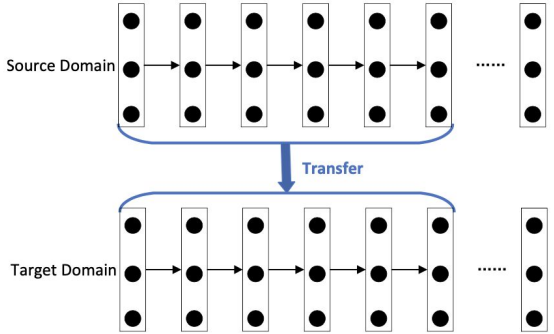
Model Transfer

Share the neural network structure and parameters between source and target via

- Sequential training by pretraining a model on source (a much richer and larger dataset) and fine-tuning on target (a smaller dataset)

Or

- Joint training target and source (related auxiliary tasks) together



[Tan et al., 2018]

Model Transfer: Sequential Training

Two Stages

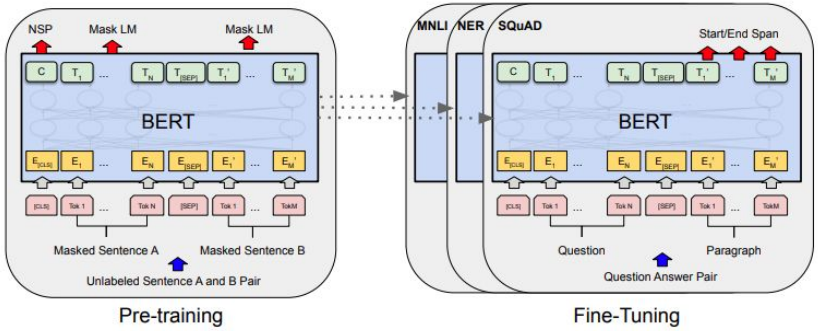
- Pre-training on source data
  - A source training that can benefit multiple target tasks
  - Computation is costly
- Adaptation on target data
  - Target data usually have limited label data
  - Fine-Tune: source model weights are starting points, trainable by target data

Commonly Used Pre-trained Models

BERT [Devlin et al., 2018]

Pre-trained Transformers for NLP Target Tasks such as

- Sentiment analysis
- Question Answering
- Entity Recognition
- …



ResNet [He et al., 2015]

Residual Learning for Computer Vision Tasks

- Image classification
- Object detection
- …

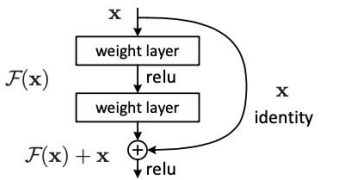
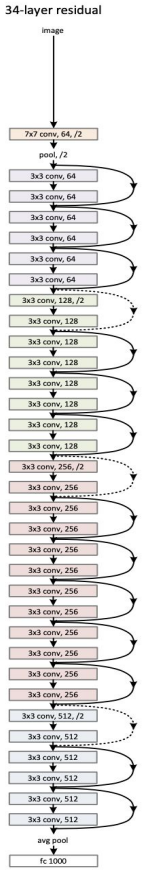


Figure 2. Residual learning: a building block.



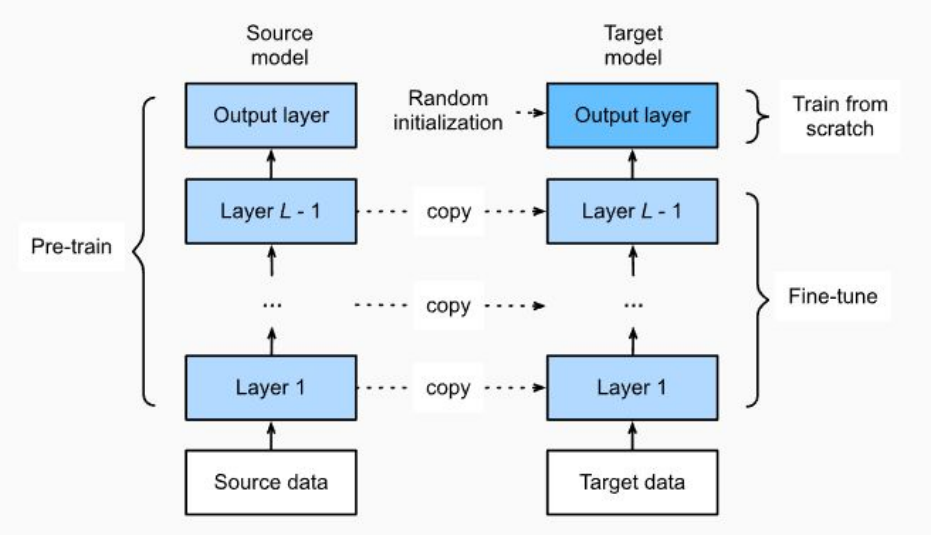
How to Fine-Tune?

Model Layers

- Top layers are task-specific
- Lower layers capture general representations
-

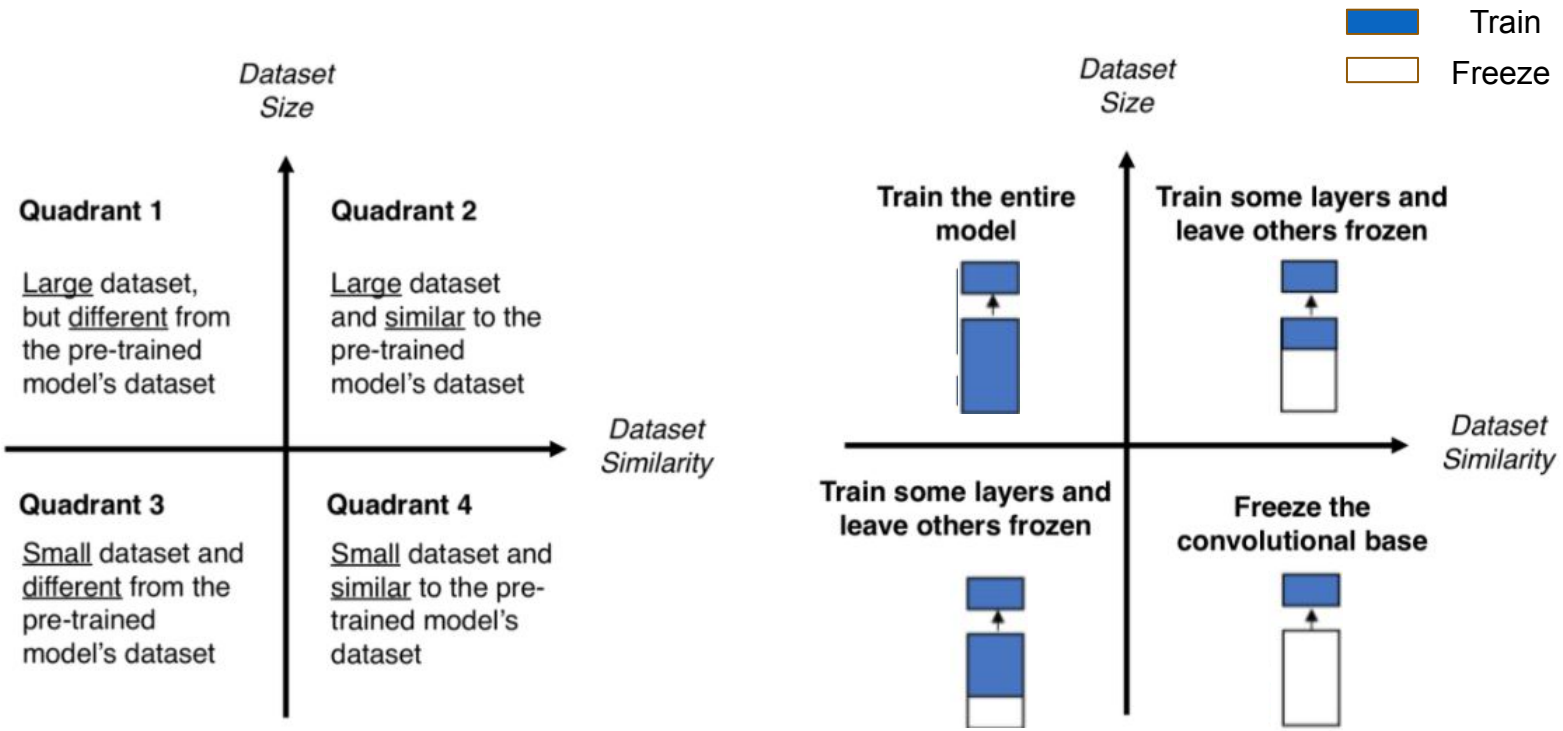
Fine-tuning Strategy

- Freeze n bottom layers learnt from source domain;
- Re-train the top layers with target domain data



Source: https://d2l.ai/chapter_computer-vision/fine-tuning.html

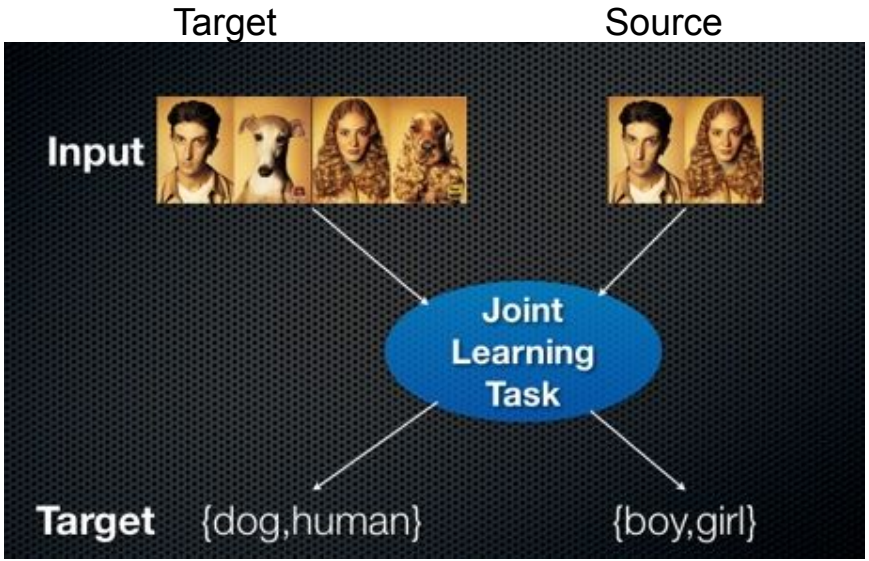
Fine-Tune Based on Target Datasets



Source: https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751

Model Transfer: Joint Training

- Also called Multi-Task Learning: whenever we try to optimize more than one loss function we are practically doing MTL.
- Layman view: learning from related tasks like what human does
- Machine learning view: improve generalization by adding auxiliary tasks (source)

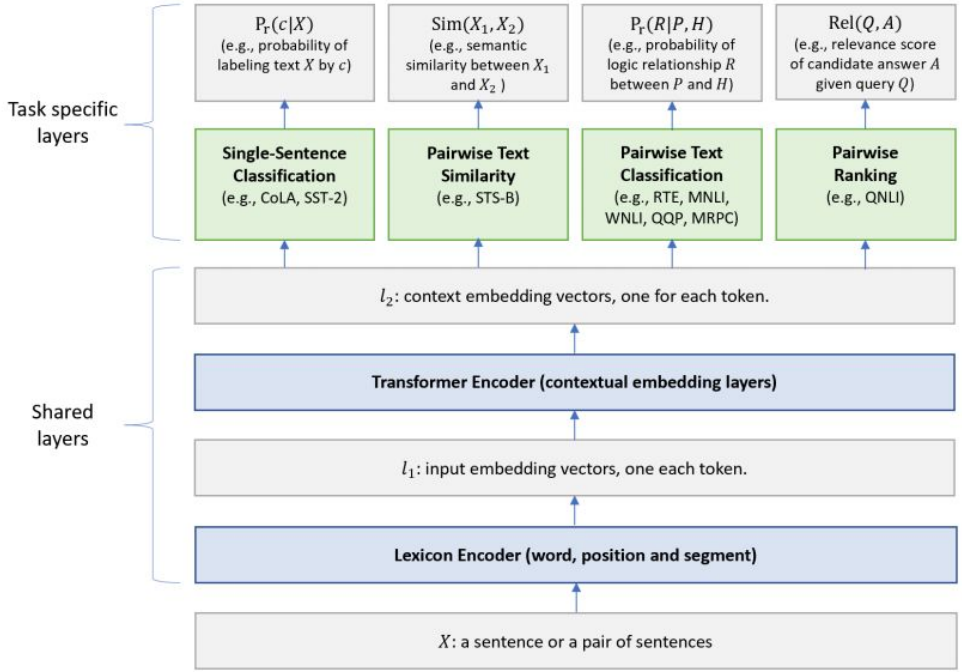


Source: http://www.cs.cornell.edu/~kilian/research/multitasklearning/multitasklearning.html

How to Joint Training Target with Source?

Hard Parameter Sharing

- Sharing the hidden layers directly while keeping the task-specific layers



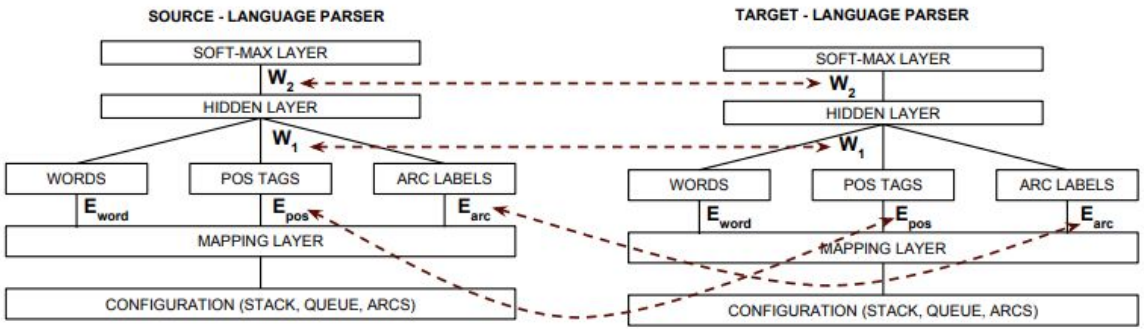
[Liu et al., 2019]

How to Joint Training Target with Source?

Soft Parameter Sharing

- No directly sharing of the parameters
- Use regularization terms to encourage parameters to be similar

$$\mathcal{L} = \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}) - \frac{\lambda_1}{2}\Big[\|W_1^{pos} - W_1^{en:pos}\|_F^2 + \|W_1^{arc} - W_1^{en:arc}\|_F^2 + \|W_2 - W_2^{en}\|_F^2\Big] - \frac{\lambda_2}{2}\Big[\|E_{pos} - E_{pos}^{en}\|_F^2 + \|E_{arc} - E_{arc}^{en}\|_F^2\Big]$$



[Duong et al., 2015]

How to Pick Auxiliary Tasks?

Assume that the auxiliary task should be related to the main task in some way and should be helpful for predicting the target task

If picking the wrong auxiliary task, can cause negative transfer, which hurts performance of target tasks.

Examples of Related Tasks

Original task	Auxiliary task
Speech recognition	Phonetic recognition, phoneme duration and frequency profile
Machine translation	POS tagging and NER
Video captioning	Next-frame and entailment prediction
Question answering	Sentence selection and answer generation model
Information retrieval	Domain classification and web search ranking
Chunking	POS tagging

Source: https://www.topbots.com/transfer-learning-in-nlp/

Model Transfer - Multi-task Learning

[Ma et al., 2018]

Multi-gate Mixture of Experts:

Can better handle the scenario where tasks are less related

- Expert networks are shared across tasks
- Gating network is trained on each individual task
- Gating network learns sophisticated task relationships

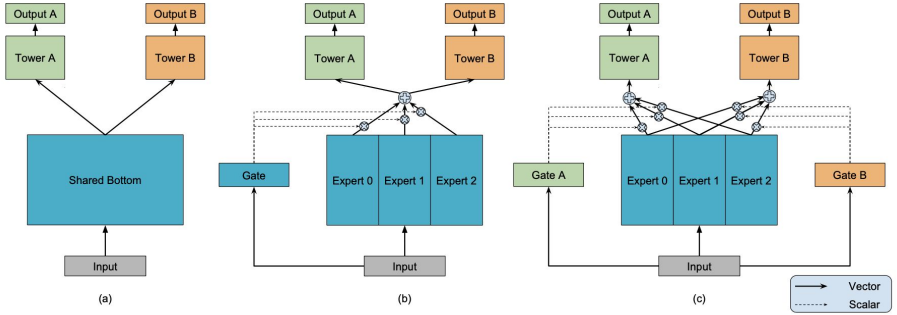
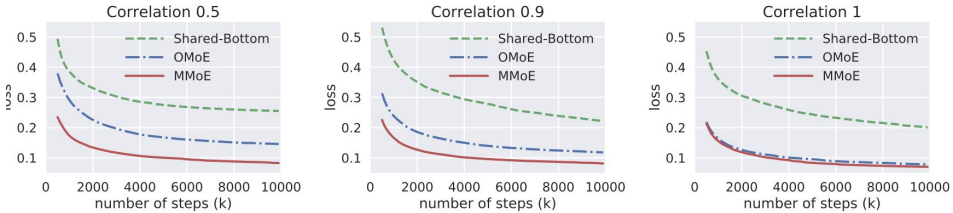


Figure 1: (a) Shared-Bottom model. (b) One-gate MoE model. (c) Multi-gate MoE model.



(a) Performance with correlation 0.5 (b) Performance with correlation 0.9 (c) Performance with two identical tasks

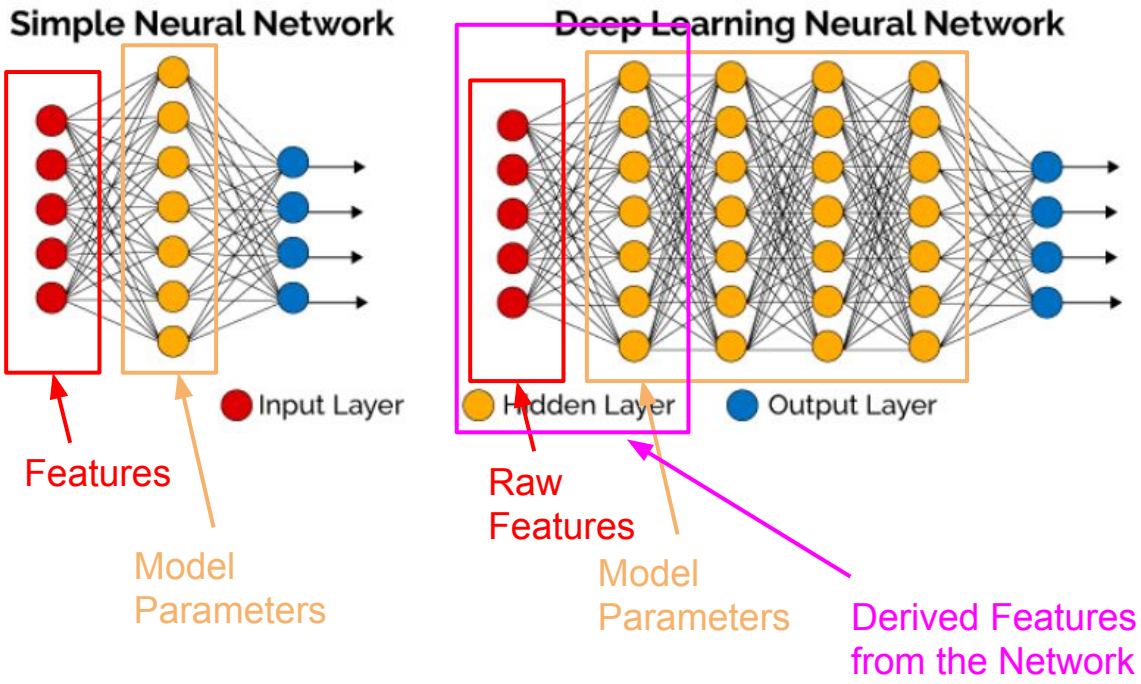
Deep Transfer Learning

Model Transfer

Feature Transfer

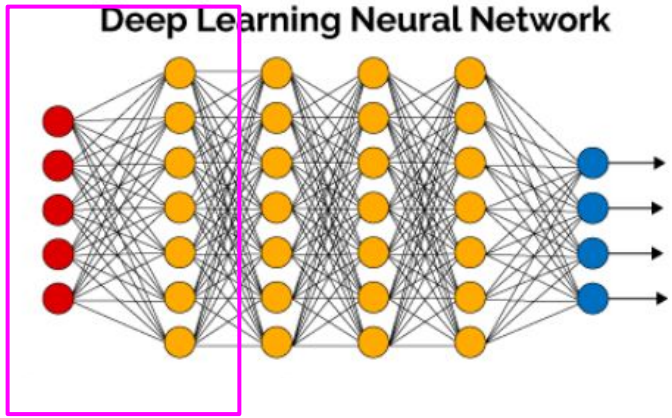
Instance Transfer

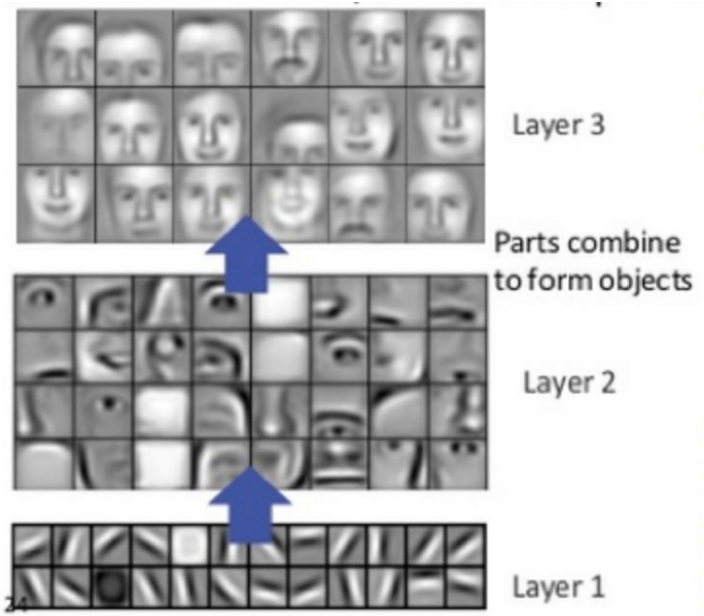
Feature Representation and Model Parameters in Deep Neural Networks



Source: https://www.kdnuggets.com/2017/12/deep-learning-made-easy-deep-cognition.html

Feature Hierarchy in Deep Neural Networks

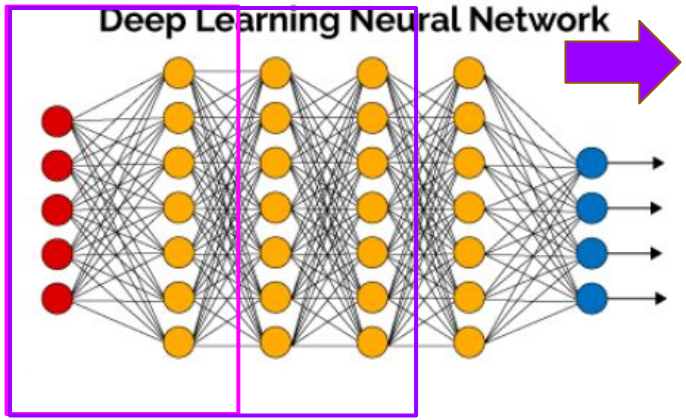




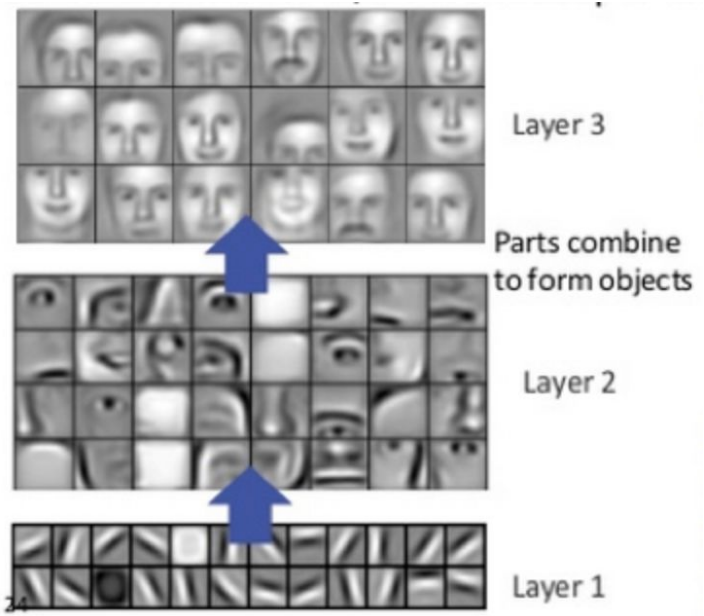
Source: https://pathmind.com/wiki/neural-network

Feature Hierarchy in Deep Neural Networks

Complex and Abstract



Concrete and Simple

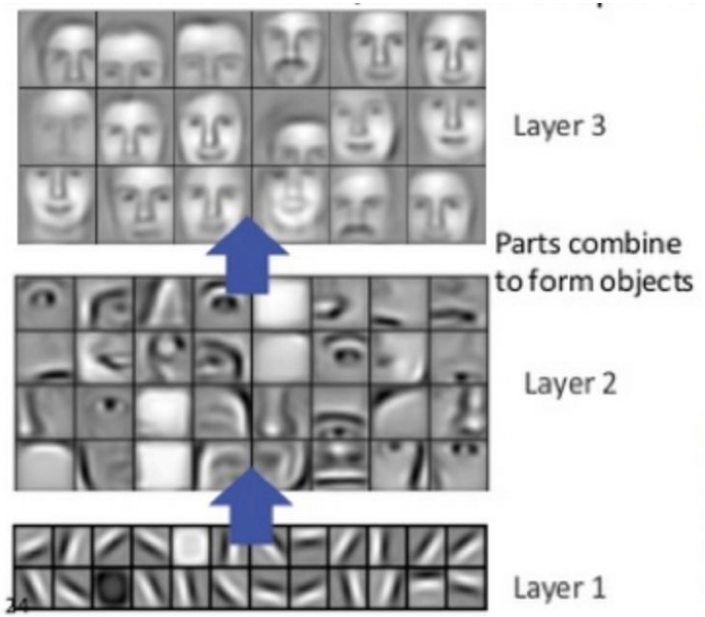


Source: https://pathmind.com/wiki/neural-network

Feature Representation vs. Model Parameters

In deep neural networks, learning the feature representation is equivalent to learning model parameters.

- Upper layers learn more task specific representation
- Lower layers learn more general representation, which can be shared across



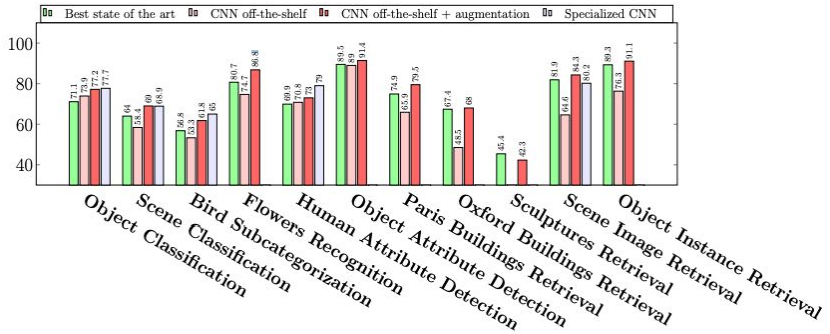
Source: https://pathmind.com/wiki/neural-network

Feature Representation Transfer

Share the feature representation between source and target

1. No adaptation to target: effective if target is closely related to source
   - Extract the lower representation layers from pre-trained source model directly and apply on target input

[Razavian et al., 2014]:
features extracted from CNN + SVM (red) outform hand crafted features + SVM (green)



Feature Representation Transfer

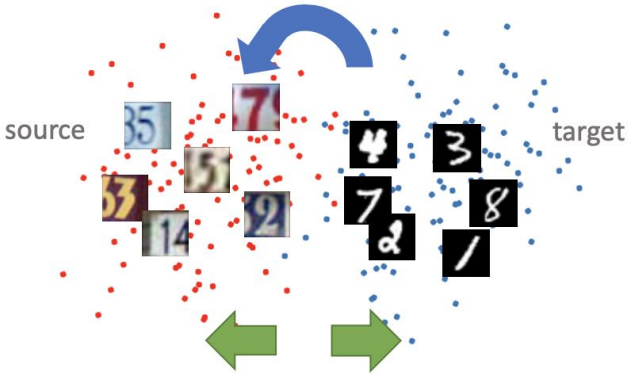
Share the feature representation between source and target

1. No adaptation to target: effective if target is closely related to source
   - Extract the lower representation layers from pre-trained source model directly and apply on target input
2. With adaptation to target: more performant when there is a gap between source and target
   - Enough labels in target: adapt the feature representation through sequential or joint training with task labels ← Similar to Model Transfer
   - When no or limited label in target: adapt the feature representation through **domain adaptation**

Feature Representation Transfer: Domain Adaptation

- Domain Adaptation Assumption:
  - The same task between domains, $Y_s = Y_t$
  - Convert source X_s and target X_t into a new space Z, being Z_s, Z_t respectively
  - $P(Y_s|Z_s) = P(Y_t|Z_t)$

- min $L_{task} + L_{\text{domain invariant measures}}$
  - L_{task} learns how to predict labels from Z
  - $L_{\text{domain invariant measures}}$ learns to align X_s and X_t into Z



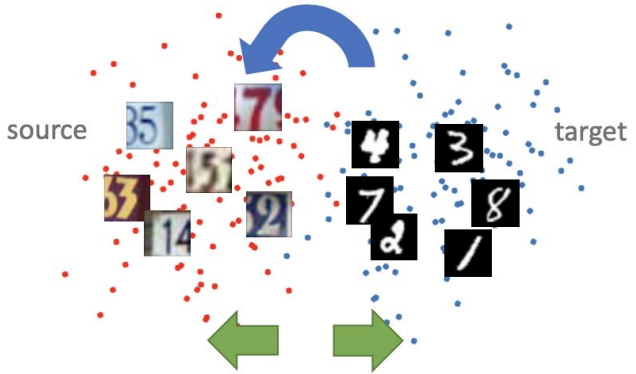
Source: [Tzeng et al., 2017]

Feature Representation Transfer: Domain Adaptation

Domain invariant measures

Categorization	Estimate Representation Invariance Between Source And Target
Discrepancy-based	Using divergence based distance
Adversarial-based	Using domain discriminators to encourage domain confusion through an adversarial objective
Reconstruction-based	Using the data reconstruction as an auxiliary task to ensure feature invariance

[Wang et al., 2018]



Source: [Tzeng et al., 2017]

Discrepancy-Based Domain Adaptation

[Tzeng et al., 2014]

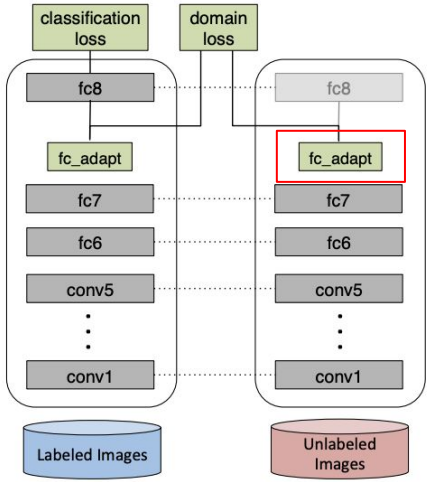
- Labeled source to predict unlabeled target

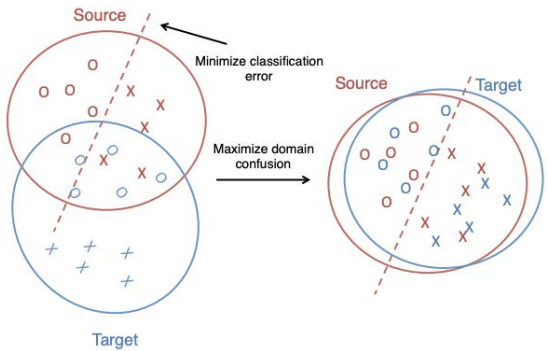
Loss function

$$\mathcal{L} = \mathcal{L}_C(X_L, y) + \lambda \mathrm{MMD}^2(X_S, X_T)$$

Classification loss on source

Domain Invariant Measure: Maximum Mean Discrepancy (MMD)

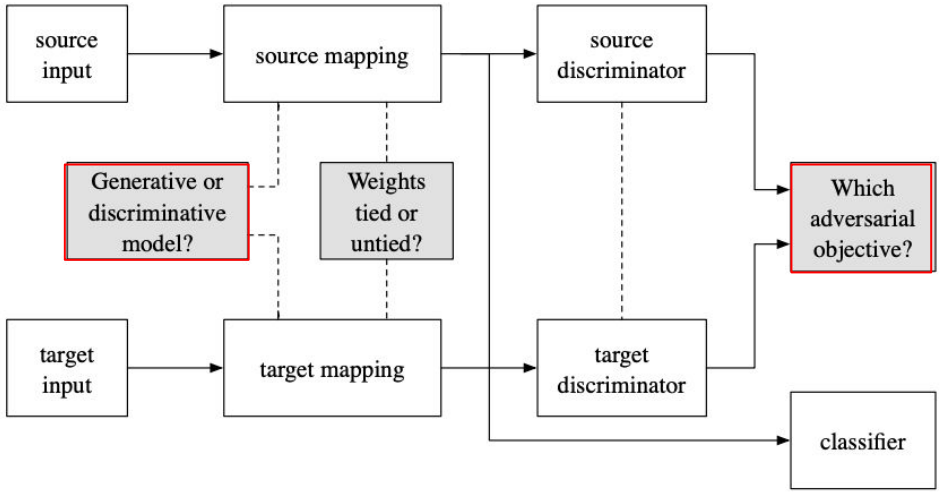




$$\mathrm{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\|$$

Adversarial-Based Domain Adaptation

- Use adversarial method to map source and target domains
  - Non-generative
    To transform data into a common representation
  - Generative
    To generate target domain synthetic data
- Tied weights
  - Enforce same transformation
- Adversarial objective
  - GAN-like
  - minimax
  - confusion

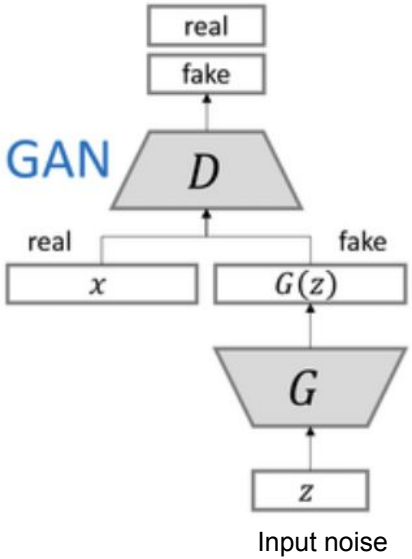


Source: [Tzeng et al., 2017]

Adversarial-Based Domain Adaptation

- Idea from success of Generative Adversarial Networks (GAN) [Goodfellow et.al, 2014]
  - Generator (G): generate fake data from noise
  - Discriminator (D): distinguish real data and generated data
- D and G play a two-player minimax game:
  - Train D to maximize correct assignment of (real, fake) labels
  - Train G to minimize the differences of real and generated data to fool the discriminator
- Iteratively optimize max_D and min_G

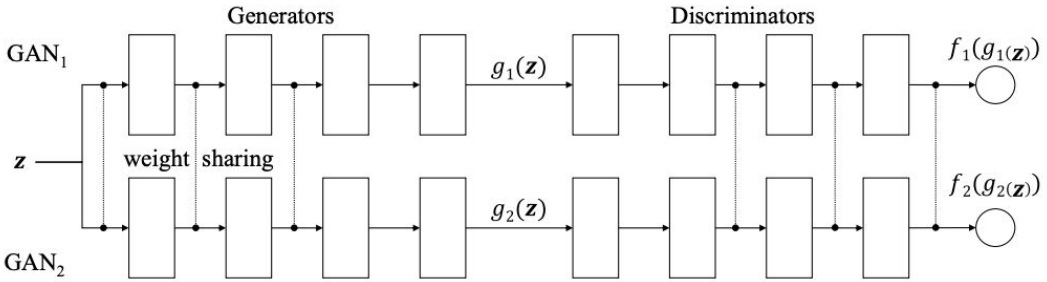
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$



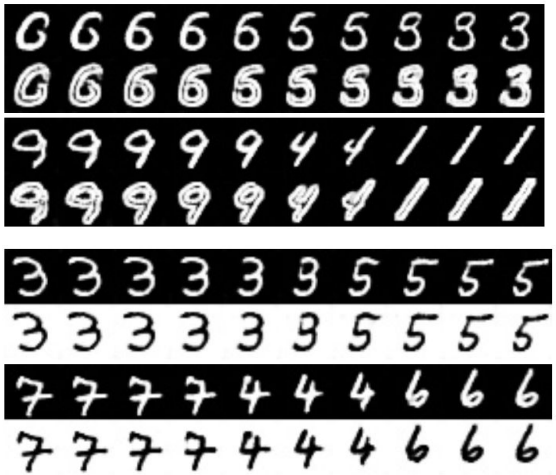
Source: https://www.researchgate.net/figure/GAN-conditional-GAN-CGAN-and-auxiliary-classifier-GAN-ACGAN-architectures-where-x_fig1_328494719

Adversarial-Based Domain Adaptation (Generative + GAN loss)

- Coupled GAN(CoGAN) [Liu et.al, 2016]
- Use the same noise to generate synthetic target images with same label as source
- Weight sharing on high level representation layers



$$V(f_1, f_2, g_1, g_2) = E_{\mathbf{x}_1 \sim p_{\mathbf{X}_1}}[-\log f_1(\mathbf{x}_1)] + E_{\mathbf{z} \sim p_{\mathbf{Z}}}[-\log(1 - f_1(g_1(\mathbf{z})))]$$
$$+ E_{\mathbf{x}_2 \sim p_{\mathbf{X}_2}}[-\log f_2(\mathbf{x}_2)] + E_{\mathbf{z} \sim p_{\mathbf{Z}}}[-\log(1 - f_2(g_2(\mathbf{z})))].$$

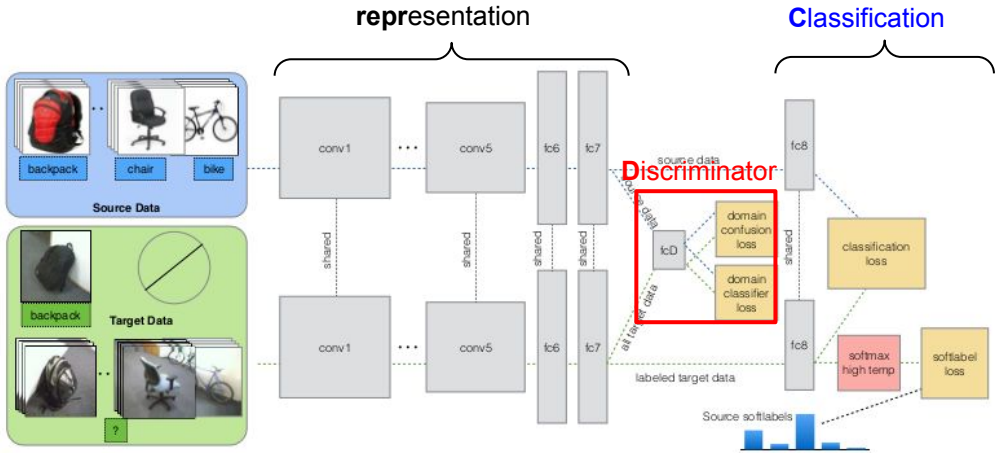


digit vs. edged image

digit vs. negative image

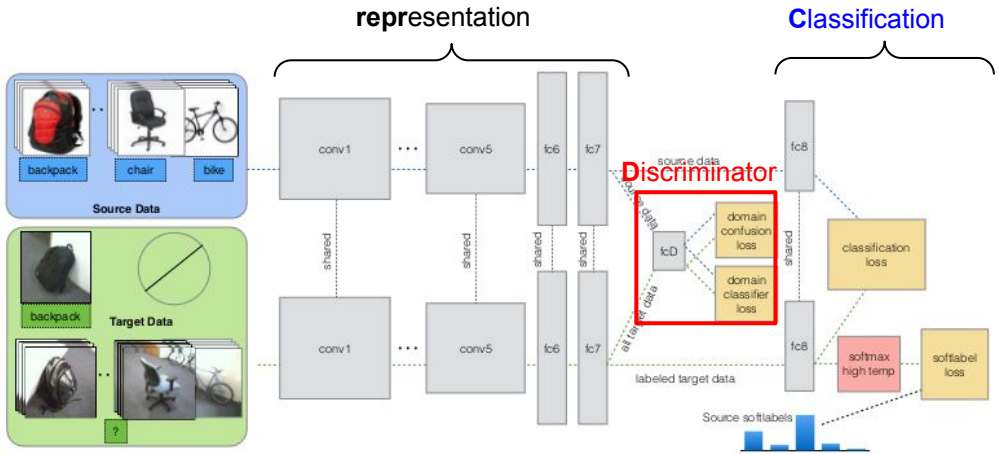
Adversarial-Based Domain Adaptation (Non-Generative + Confusion Loss)

- [Tzeng et al., 2015]
- Labeled source to predict partially labeled target



Adversarial-Based Domain Adaptation (Non-Generative + Confusion Loss)

- [Tzeng et al., 2015]
- Labeled source to predict partially labeled target



$$\mathcal{L}(x_S, y_S, x_T, y_T, \theta_D; \theta_{\text{repr}}, \theta_C) =$$

1. Classification Loss $\mathcal{L}_C(x_S, y_S, x_T, y_T; \theta_{\text{repr}}, \theta_C)$
2. Domain Confusion Loss $+ \lambda \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}})$
3. Soft Label Loss $+ \nu \mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C).$

Domain Discrimination Loss

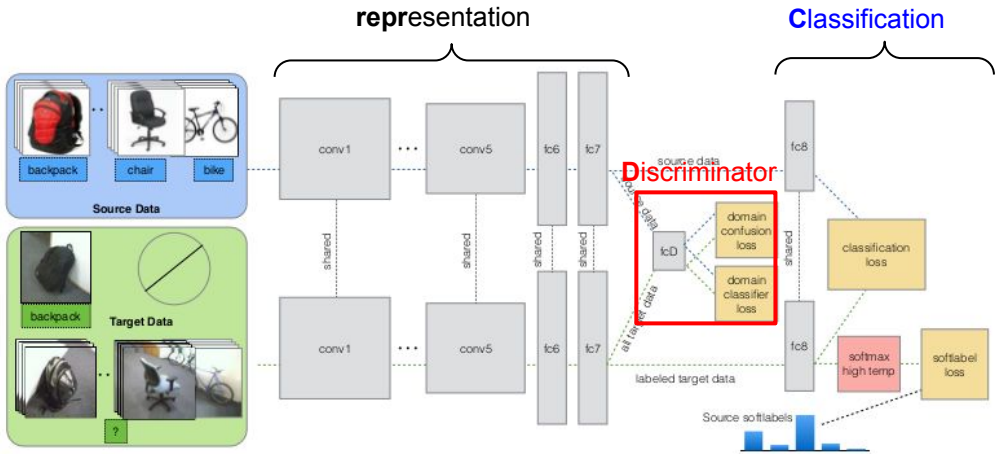
$$\min_{\theta_D} \mathcal{L}_D(x_S, x_T, \theta_{\text{repr}}; \theta_D) = -\sum \mathbb{1}[y_D = d] \log q_d$$

$$\min_{\theta_{\text{repr}}} \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}) = -\sum_d \frac{1}{D} \log q_d$$

Domain Confusion Loss

Adversarial-Based Domain Adaptation (Non-Generative + Confusion Loss)

- [Tzeng et al., 2015]
- Labeled source to predict partially labeled target
- Domain confusion loss
  - Learning auxiliary discriminator: minimize loss L_D to learn differences between domains
  - Learning representation: minimize loss L_{conf} to fool the discriminator
  - Iteratively Trained



$$\mathcal{L}(x_S, y_S, x_T, y_T, \theta_D; \theta_{\text{repr}}, \theta_C) =$$

1. Classification Loss $\mathcal{L}_C(x_S, y_S, x_T, y_T; \theta_{\text{repr}}, \theta_C)$
2. Domain Confusion Loss $+ \lambda \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}})$
3. Soft Label Loss $+ \nu \mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C).$

Domain classification loss

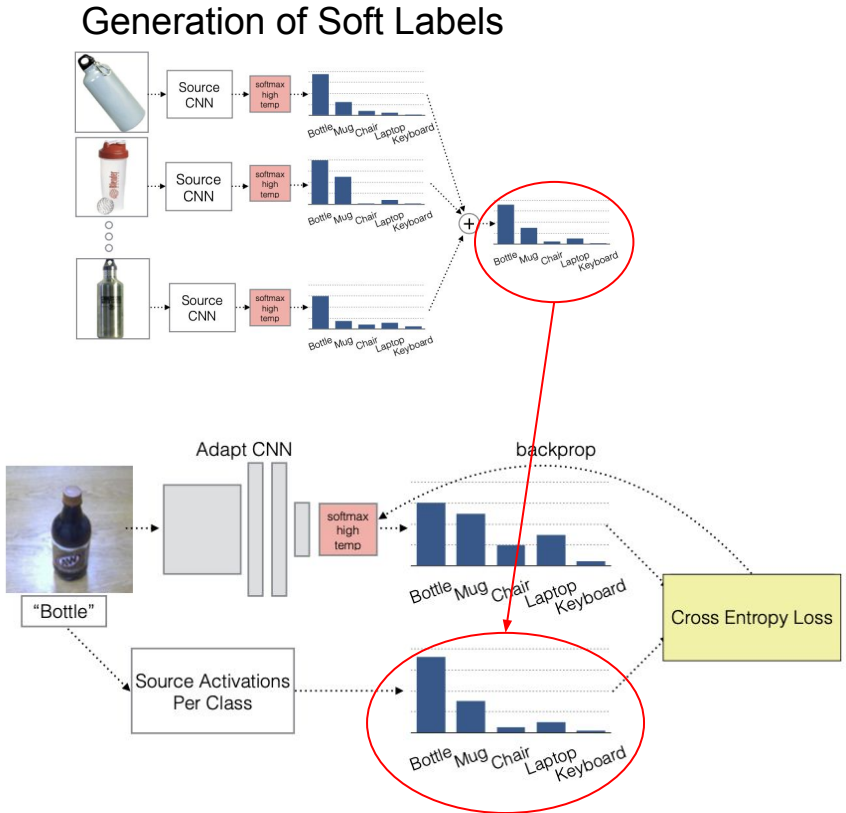
$$\min_{\theta_D} \mathcal{L}_D(x_S, x_T, \theta_{\text{repr}}; \theta_D) = -\sum \mathbb{1}[y_D = d] \log q_d$$

$$\min_{\theta_{\text{repr}}} \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}) = -\sum_d \frac{1}{D} \log q_d$$

cross entropy between predicted domain label and uniform distribution

Adversarial-Based Domain Adaptation (Non-Generative + Confusion Loss)

- [Tzeng et al., 2015]
- Labeled source to predict partially labeled target
- Soft Label Loss
  - Pre-generated from source
  - Used to align labels between the two domains by learn relationships between label classes

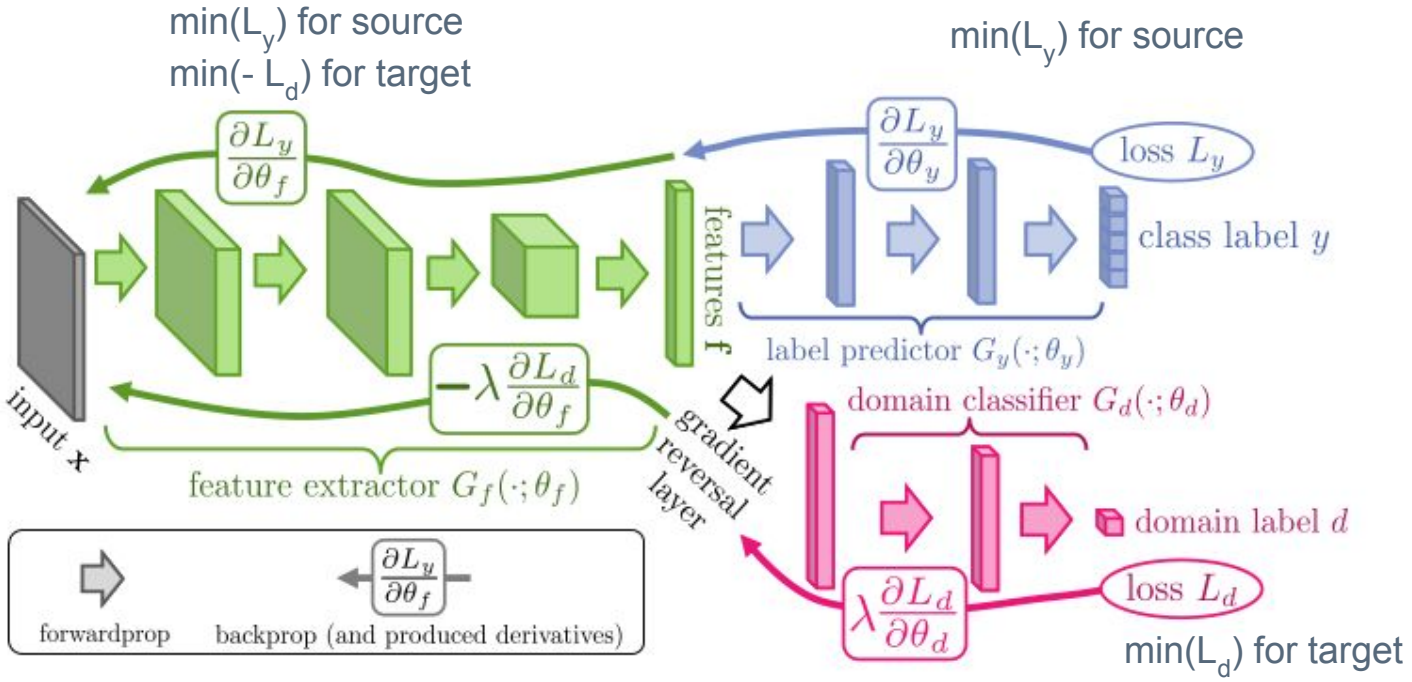


$$\mathcal{L}(x_S, y_S, x_T, y_T, \theta_D; \theta_{\text{repr}}, \theta_C) = \mathcal{L}_C(x_S, y_S, x_T, y_T; \theta_{\text{repr}}, \theta_C) + \lambda \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}) + \nu \mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C).$$

3. Soft Label Loss

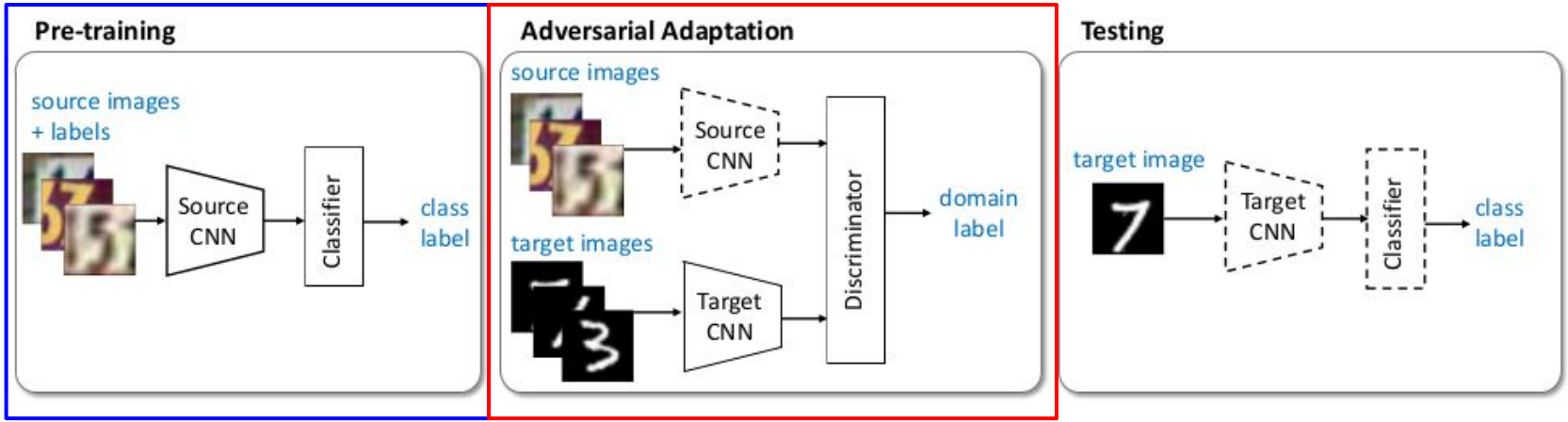
Adversarial-Based Domain Adaptation (Non-Generative + Minimax Loss)

- Domain-Adversarial Neural Networks (DANN) [Ganin et al., 2016]
- Labeled source to predict unlabeled target



Adversarial-Based Domain Adaptation (Non-Generative + GAN loss)

- Adversarial Discriminative Domain Adaptation (ADDA) [Tzeng et al., 2017]
- Labeled source to predict unlabeled target
- Optimize the classification and discriminative objectives sequentially



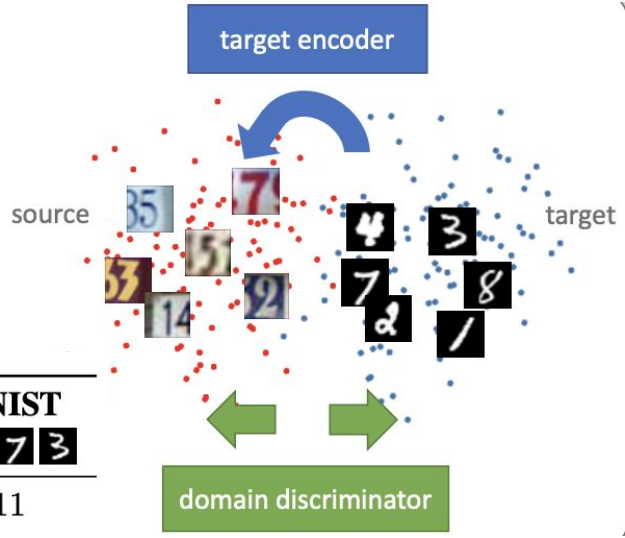
Dashed shapes mean frozen coefficients

Adversarial-Based Domain Adaptation

- Comparison [Tzeng et al., 2017]

Partially shared

Method	Base model	Weight sharing	Adversarial loss
Gradient reversal	discriminative	shared	minimax
Domain confusion	discriminative	shared	confusion
CoGAN	generative	unshared	GAN
ADDA	discriminative	unshared	GAN



Method	MNIST → USPS	USPS → MNIST	SVHN → MNIST
Source only	0.752 ± 0.016	0.571 ± 0.017	0.601 ± 0.011
Gradient reversal	0.771 ± 0.018	0.730 ± 0.020	0.739
Domain confusion	0.791 ± 0.005	0.665 ± 0.033	0.681 ± 0.003
CoGAN	0.912 ± 0.008	0.891 ± 0.008	did not converge
ADDA	0.894 ± 0.002	0.901 ± 0.008	0.760 ± 0.018

Non-converge due to distant domains

Table 2: Experimental results on unsupervised adaptation among MNIST, USPS, and SVHN.

Reconstruction-Based Domain Adaptation

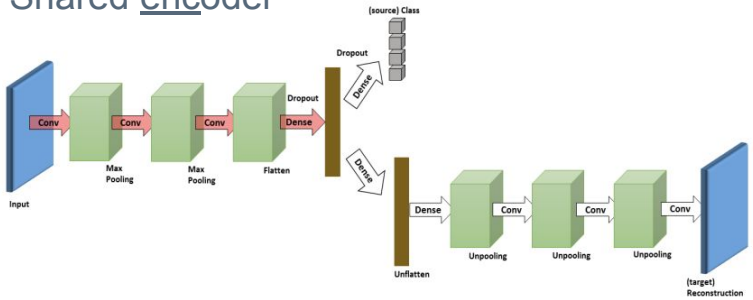
- Deep Reconstruction Classification Networks (DRCN) [Ghifary et al., 2016]
- Labeled source to predict unlabeled target
- Make sure representation contains necessary information by comparing reconstructed image to the input one

$$\min \lambda \mathcal{L}_c^{n_s}(\{\Theta_{\text{enc}}, \Theta_{\text{lab}}\}) + (1-\lambda)\mathcal{L}_r^{n_t}(\{\Theta_{\text{enc}}, \Theta_{\text{dec}}\}),$$

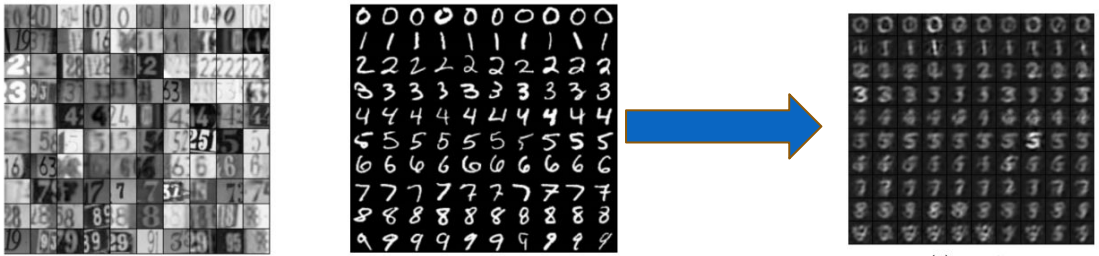
Source classification — target reconstruction

Source
label classification

Shared encoder



Target decoder



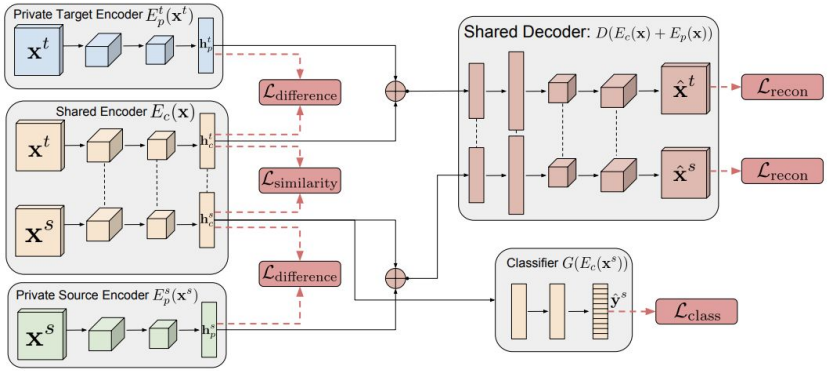
(a) Source (SVHN) (b) Target (MNIST) (c) DRCN

Hybrid Model with Multiple Loss Included

- Domain Separation Networks (DSN) [Bousmalis et al., 2016]
- Mix reconstruction, discrepancy and adversarial losses

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\,\mathcal{L}_{\text{recon}} + \beta\,\mathcal{L}_{\text{difference}} + \gamma\,\mathcal{L}_{\text{similarity}}$$

- Shared Encoder to capture domain shared knowledge
- Private Encoder to capture domain specific knowledge
- Task classifier based on the shared encoder



(a) MNIST (source) (b) MNIST-M (target)

Columns left to right:
1. original image, 2. reconstructed, 3. shared recon, 4. private recon

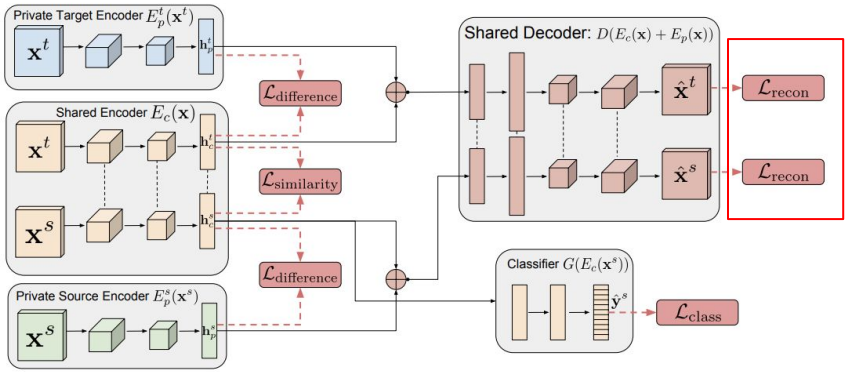
Hybrid Model with Multiple Loss Included

- Domain Separation Networks [Bousmalis et al., 2016]
- Mix reconstruction, discrepancy and adversarial losses

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\,\mathcal{L}_{\text{recon}} + \beta\,\mathcal{L}_{\text{difference}} + \gamma\,\mathcal{L}_{\text{similarity}}$$

- Reconstruction loss uses MSE to make sure learnt representation contain necessary information

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{N_s} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^s, \hat{\mathbf{x}}_i^s) + \sum_{i=1}^{N_t} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^t, \hat{\mathbf{x}}_i^t)$$



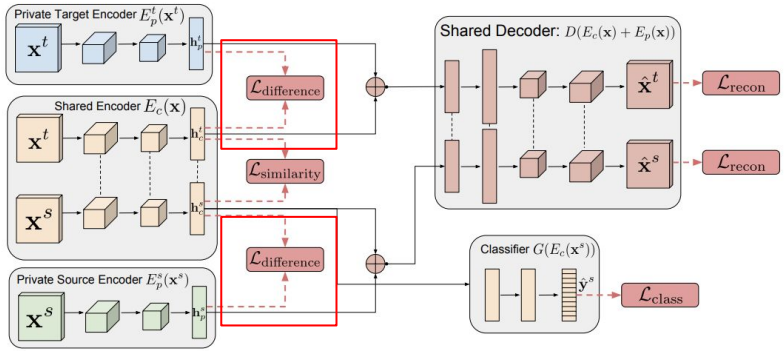
Hybrid Model with Multiple Loss Included

- Domain Separation Networks [Bousmalis et al., 2016]
- Mix reconstruction, discrepancy and adversarial losses

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\, \mathcal{L}_{\text{recon}} + \beta\, \mathcal{L}_{\text{difference}} + \gamma\, \mathcal{L}_{\text{similarity}}$$

- Discrepancy loss to make sure private and shared encoder remain orthogonal

$$L_{\text{difference}} = \left\| \mathbf{H}_c^{s\top} \mathbf{H}_p^s \right\|_F^2 + \left\| \mathbf{H}_c^{t\top} \mathbf{H}_p^t \right\|_F^2$$

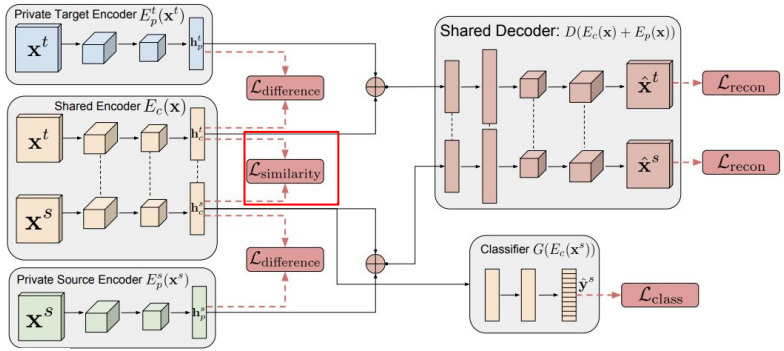


Hybrid Model with Multiple Loss Included

- Domain Separation Networks [Bousmalis et al., 2016]
- Mix reconstruction, discrepancy and adversarial losses

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\, \mathcal{L}_{\text{recon}} + \beta\, \mathcal{L}_{\text{difference}} + \gamma\, \mathcal{L}_{\text{similarity}}$$

- Similarity loss to make sure shared encoder cannot be distinguished
  - Both discrepancy and adversarial tried



$$\mathcal{L}_{\text{similarity}}^{\text{MMD}} = \frac{1}{(N^s)^2}\sum_{i,j=0}^{N^s}\kappa(\mathbf{h}_{ci}^s, \mathbf{h}_{cj}^s) - \frac{2}{N^s N^t}\sum_{i,j=0}^{N^s,N^t}\kappa(\mathbf{h}_{ci}^s, \mathbf{h}_{cj}^t) + \frac{1}{(N^t)^2}\sum_{i,j=0}^{N^t}\kappa(\mathbf{h}_{ci}^t, \mathbf{h}_{cj}^t),$$

$$\mathcal{L}_{\text{similarity}}^{\text{DANN}} = \sum_{i=0}^{N_s+N_t}\left\{d_i \log \hat{d}_i + (1-d_i)\log(1-\hat{d}_i)\right\}$$

Hybrid Model Accuracy Comparison

- Source-only and target-only are the lower/upper bounds of domain adaptation

Model	MNIST to MNIST-M	Synth Digits to SVHN	SVHN to MNIST	Synth Signs to GTSRB
Source-only	56.6 (52.2)	86.7 (86.7)	59.2 (54.9)	85.1 (79.0)
CORAL	57.7	85.2	63.1	86.9
MMD	76.9	88.0	71.1	91.1
DANN	77.4 (76.6)	90.3 (91.0)	70.7 (73.8)	92.9 (88.6)
DSN w/ MMD	80.5	88.5	72.2	92.6
DSN w/ DANN	**83.2**	**91.2**	**82.7**	**93.1**
Target-only	98.7	92.4	99.5	99.8

Deep Transfer Learning

Model Transfer

Feature Transfer

Instance Transfer

Instance Transfer

Because there is a domain gap between source and target, blindly add source data into target could deteriorate the target model performance.

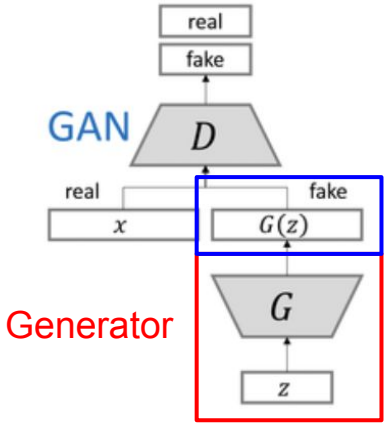
Example: Paraphrase Identification problem

Domain	Sentence 1	Sentence 2
Source (Open Domain)	Which answers does Quora show first for each question? What ***order*** should the Matrix movies be watched in How can I **order** a cake from Walmart online?	How does Quora decide the ***order*** of the answers to a question? Is there any particular ***order*** in which I should watch the Madea movies How do I **order** a cake from Walmart?
Target (E-commerce Domain)	How long is my **order** arriving? Is it over? Will I have the refund? How can i get an **order** receipt or invoice? I need to understand why my **orders** have been cancelled	I have escalated an **order** and have not been updated in over a week. How do I get an invoice to pay? Why my **order** have been closed?

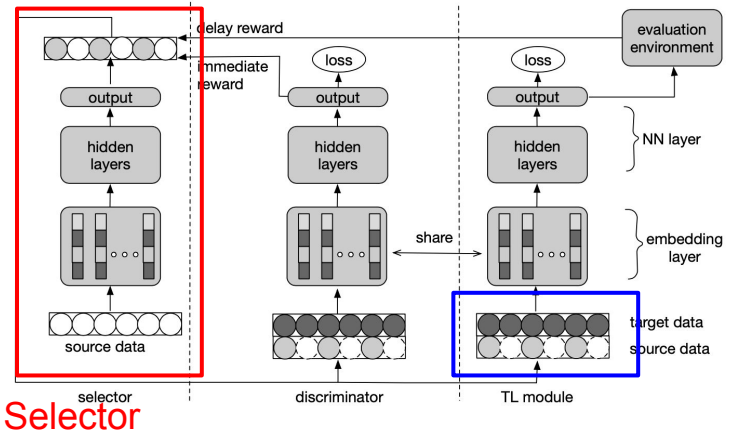
[Qu et al., 2018]

Instance Transfer vs. Generative Data Augmentation

Instance transfer selects source data to augment target data, while generative data augmentation method like GAN create synthetic data instead.



Source: Mino et al., 2018



Source: Wang et al., 2019

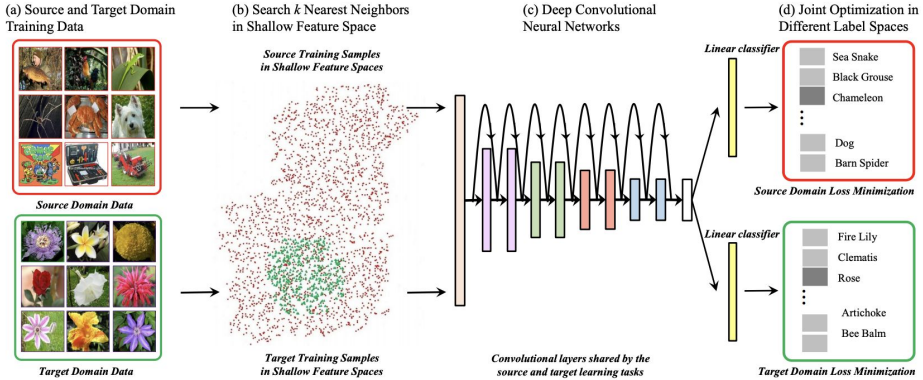
Instance Transfer

Select data instances from source for target training and avoid negative transfer

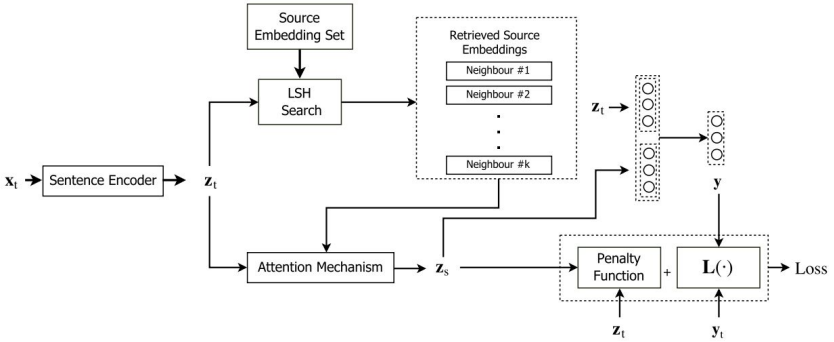
Assumption: Homogeneous feature space, where $X_s = X_t$, $P(X_s) \neq P(X_t)$

- Traditional transfer learning methods use reweighting or instance selection
- In the deep learning world,
  - selection is the predominantly used approach
  - tightly coupled with shared feature representation learning between source and target

Instance Transfer with Feature Representation Transfer



[Ge et al., 2017]



[Chowdhury et al., 2018]

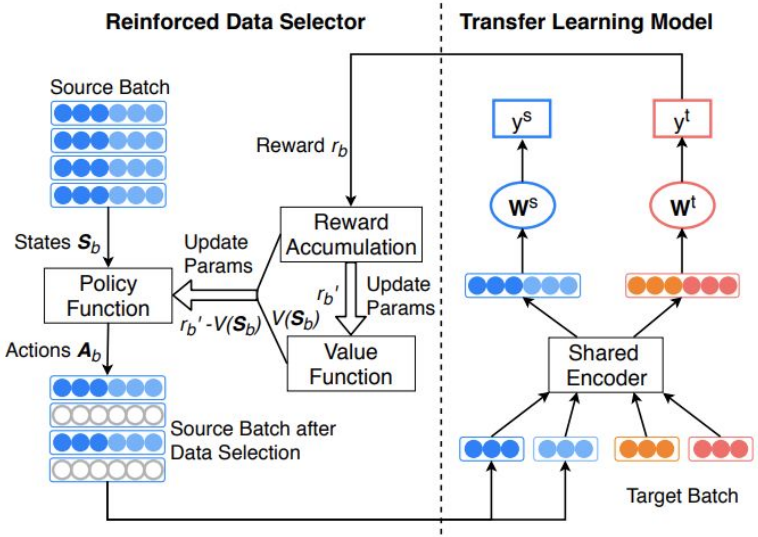
1. Find similar source samples in a shared representation space
2. Train selected source and target for task
3. Sometimes couple task training with shared representation learning

Instance Transfer with Feature Representation Transfer

[Qu et al., 2018]

Deep Text Matching for NLP tasks

- A reinforced selector collecting reward in terms of target accuracy improvement
- Selector model is jointly trained with the target task
- Selected source and target shares representation layers



Methods	PI		NLI	
	Acc	AUC	Acc	AUC
Source Only — Base Model [22]	0.8393	0.8548	0.7300	0.7663
Source+Target — Transfer Learning Model	0.8488	0.8706	0.7453	0.8044
Ruder and Plank [27]	0.8458	0.8680	0.7521	0.8062
Selected Source +Target — RTL	**0.8616**‡	**0.8829**	**0.7672**‡	**0.8163**

Key Takeaways

In Deep Transfer Learning,

- Use sequential training or joint training to achieve model transfer from source to target
- Feature representation transfer is to share the lower layers of the neural network from source model to target. They can be directly used to extract features or adapted using model transfer techniques
- Domain adaptation is used when the target label is sparse and the feature representation space needs to be aligned between source and target
- Instance transfer is integrated with feature representation learning between source and target

Key Takeaways (cont'd)

Relevancy between source and target determines the knowledge transfer upper bound. How to find the optimal settings to reach maximum transfer efficiency?

- Model Transfer: tune the number of layers shared, learning rates in fine-tuning, weights of tasks in joint training
- Feature Representation Transfer: Find the right level of feature space alignment of source and target
- Instance Transfer: integrate selector build with representation learning and task training

Agenda

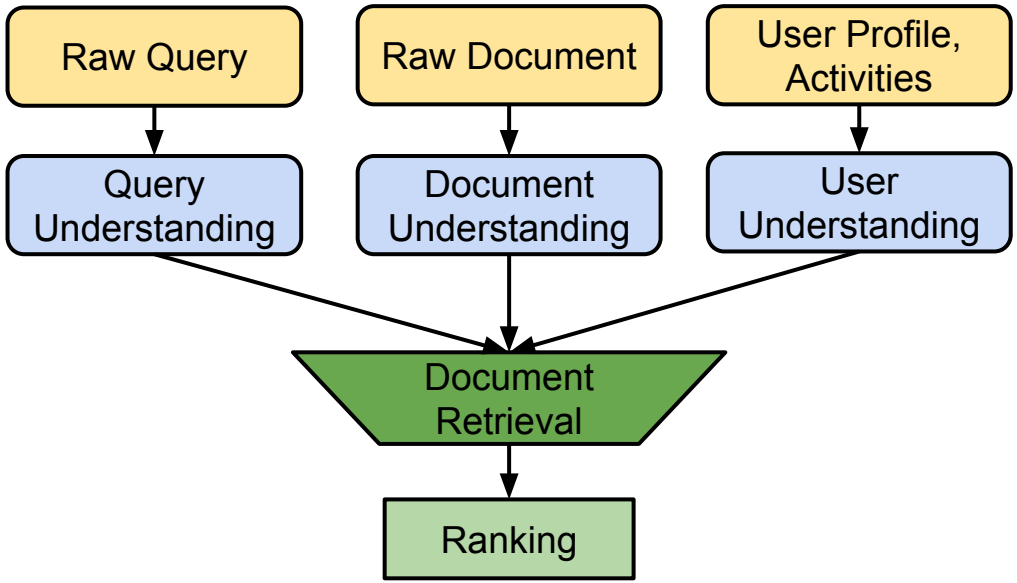
1 Introduction

2 Deep Transfer Learning

3 **Deep Transfer Learning for Search and Recommendation**

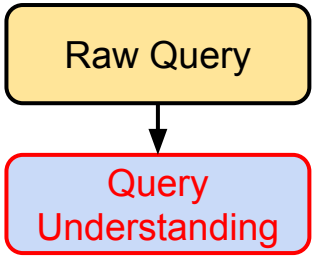
4 Applications at LinkedIn

Overview of Search and Recommendation Systems

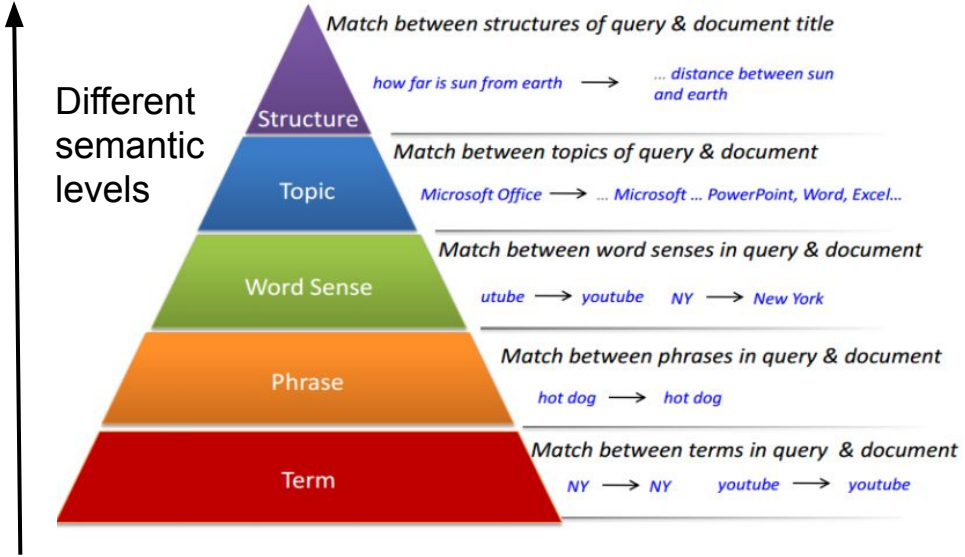


- Deep transfer learning is well integrated with deep models that are widely used now in search and recommendation applications.
- 3 phases in search and recommendation systems
  - Understanding
  - Retrieval
  - Ranking

Deep Neural Network for Query Understanding



Understand Query Semantics

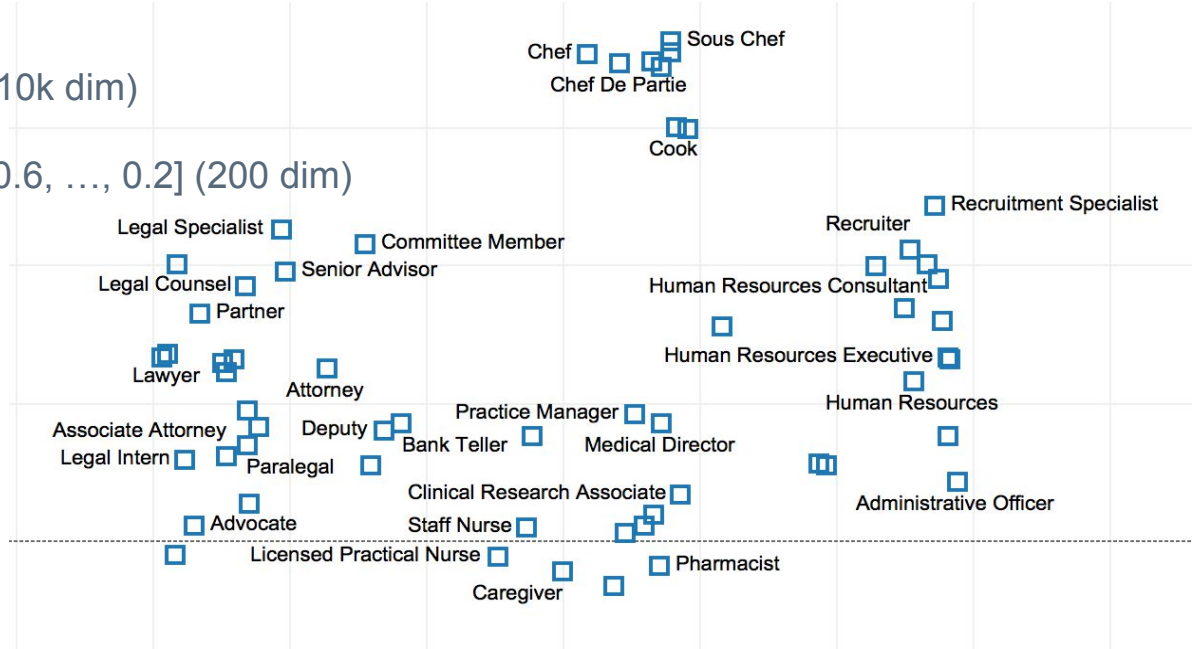


Source: Prakash, 2015

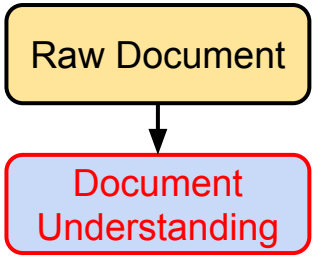
Embeddings

Low dimensional, learnt continuous vector representations of categorical variable, obtained by extracting the lower layers of trained neural network model.

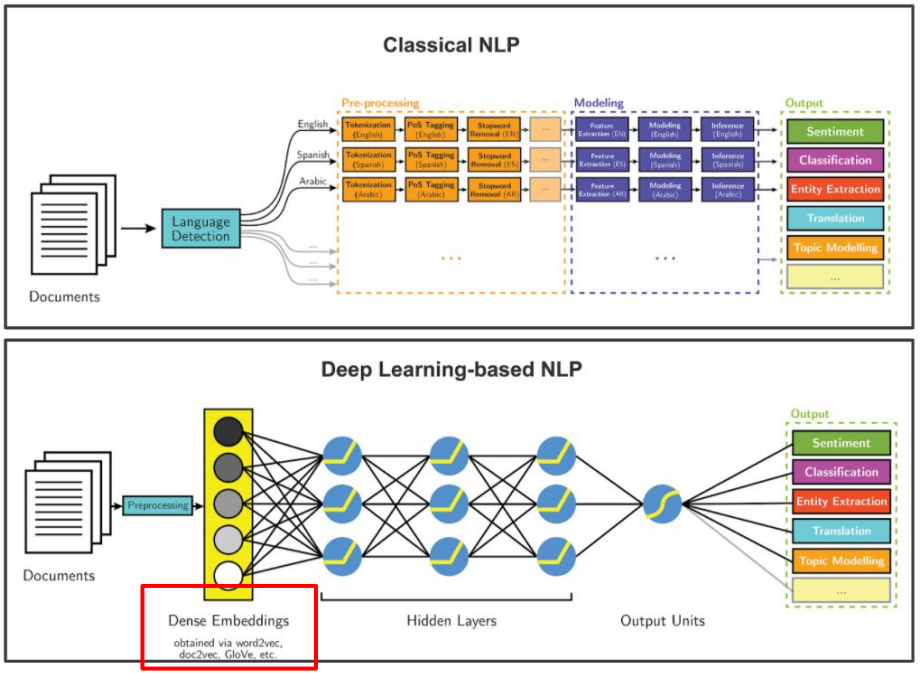
Recruiter:
1-hot sparse: [0, …, 1, …, 0] (>10k dim)
→
Dense embeddings: [0.1, 0.9, -0.6, …, 0.2] (200 dim)



Deep Neural Network for Representation Learning

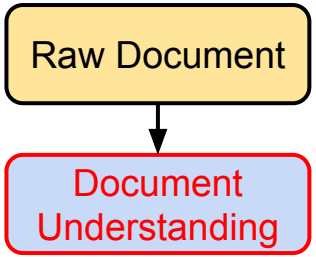


Learning Document Text Representation



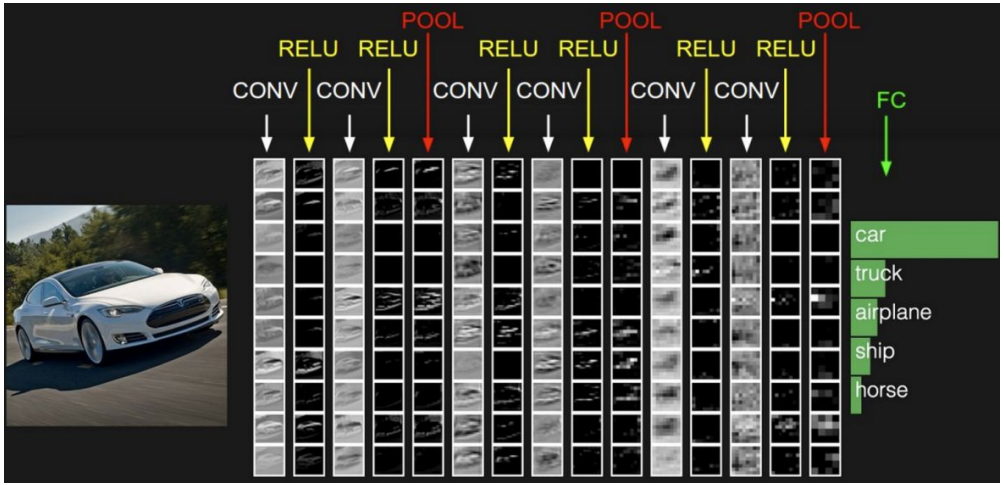
Source: https://s3.amazonaws.com/aylien-main/misc/blog/images/nlp-language-dependence-small.png

Deep Neural Network for Representation Learning



Learning Document Image Representation

Embeddings



Source: https://cs231n.github.io/convolutional-networks/

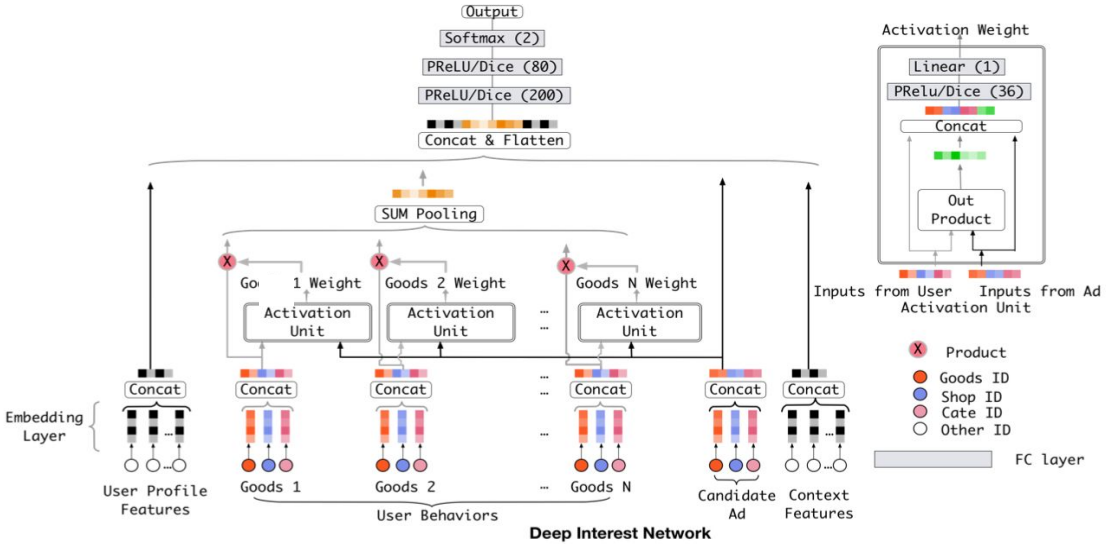
Deep Neural Network for Representation Learning

User Profile, Activities

↓

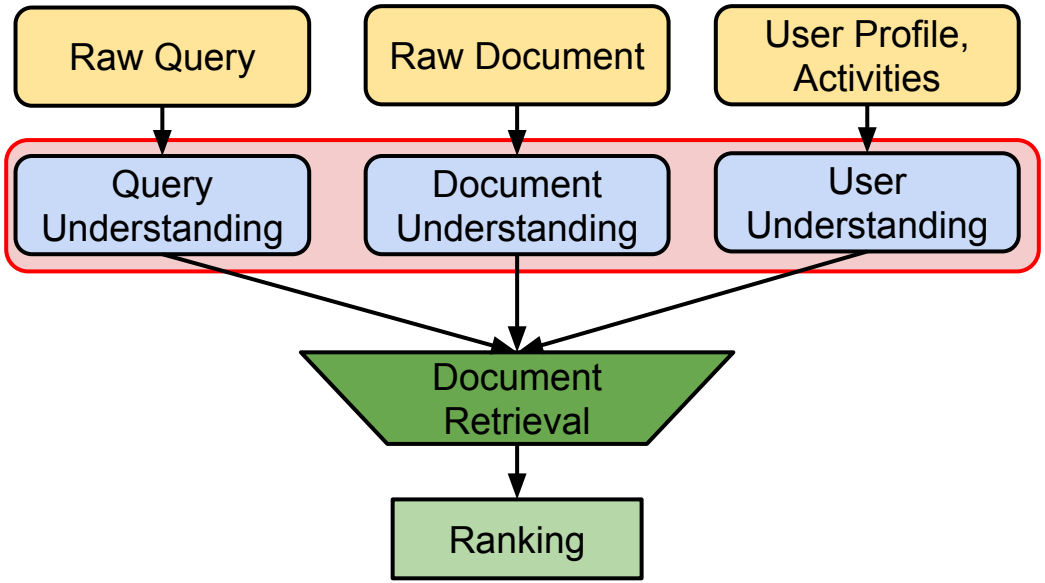
User Understanding

Learn User Interest and Intent



Source: [Zhou et al., 2018]

Deep Neural Networks for Understanding

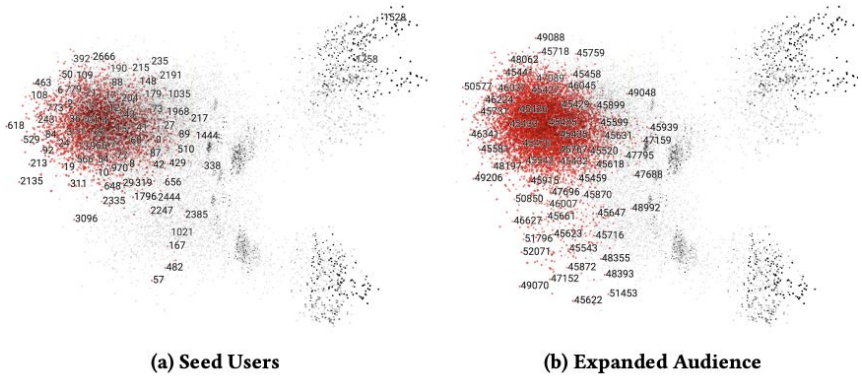


Understanding:

- Deep models can enable deeper query semantics and representation learning
- Use feature transfer techniques to extract embedding from pre-trained models

Expand Retrieval Set in the Embedding Space

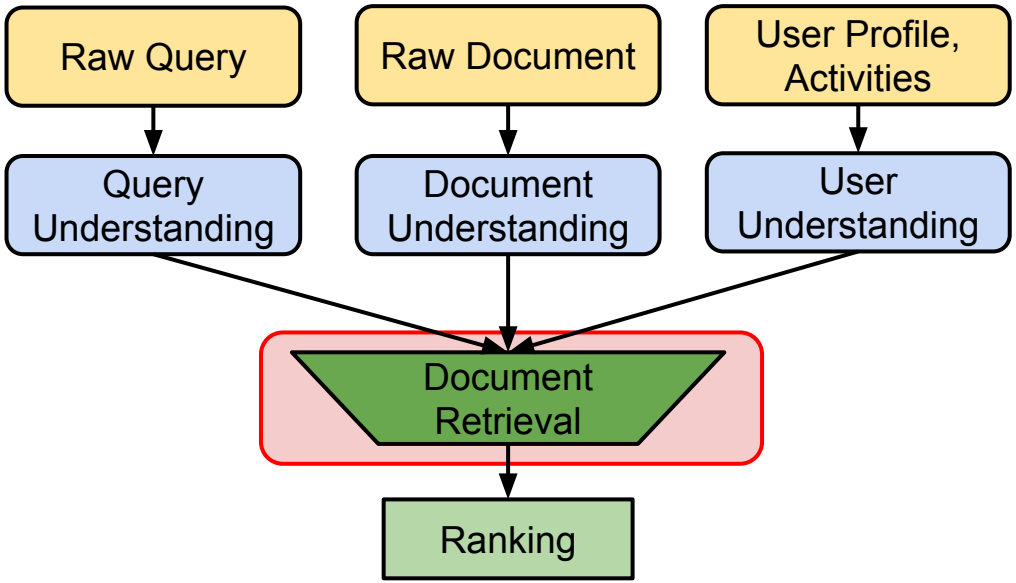
Capture candidates that can't be reached by rule or term-based matching



Source: https://labs.pinterest.com/audience_expansion

Source: https://cloud.google.com/solutions/machine-learning/images.png

Deep Neural Networks for Retrieval



Retrieval/Candidate Selection:

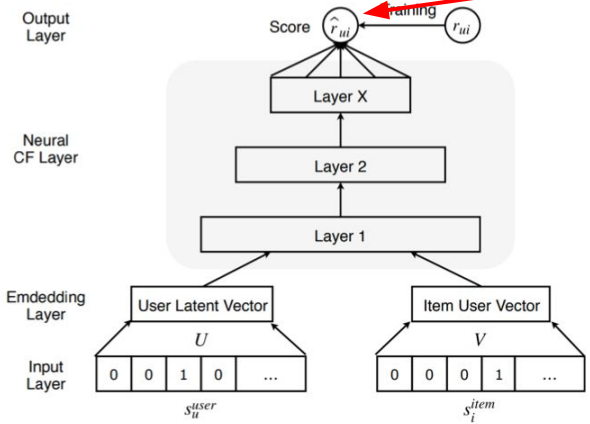
- Expand the candidate set in the embedding space to capture more eligible candidates
- This embedding space is obtained beforehand from a pre-trained model

Deep Neural Network Structure for Ranking

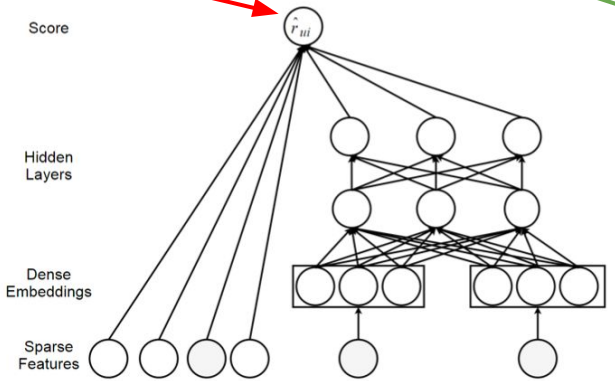
[Zhang et al., 2019]

Ranking Objective: user rating or action on the document

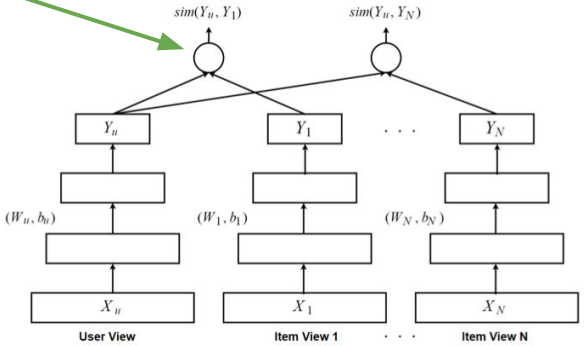
+ user rating or action on related applications



Neural Collaborative Filtering

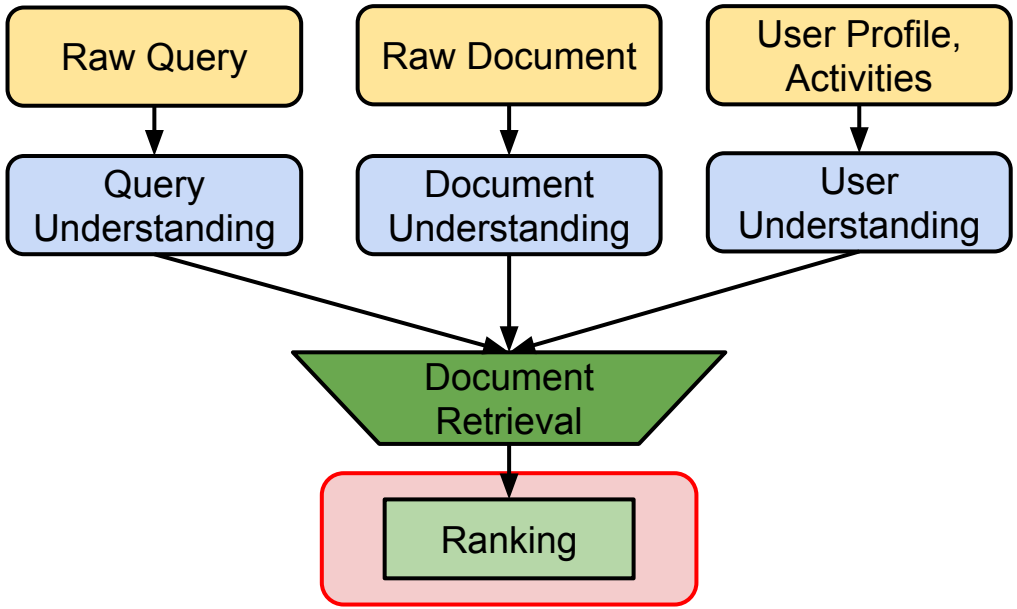


Wide & Deep Learning



Multi-View Deep Neural Network

Deep Neural Networks for Ranking



Ranking:

- Deep models can outperform shallow models with their flexible model structure
- Leverage user data from related applications, use model transfer and instance transfer can further improve the learning performance

How Can Deep Transfer Learning Help?

Deep models are widely adopted now and deep transfer learning is tightly coupled with them to improve the learning performance

- Understanding and Document Retrieval:
  - Use feature representation transfer to learn better query understanding and document and user representation
- Ranking:
  - Use model transfer, sometimes coupled with instance selection to leverage matching signals and knowledge across applications

Feature Transfer for Understanding and Representation

Deep Transfer Learning for Understanding and Representation

Feature Representation Transfer

Use pre-trained models to extract features and better learn

- User intent and interest from
  - Search queries (Search)
  - User profile and past activities (Recommendation)
- Document representation from
  - Content in the document: Text, image, video etc.

Deep Query Understanding Leveraging Pre-Trained Models

Move embeddings from context-free to context-aware

food

due to its tartness, it is often combined with sweeter juices, such as **apple** or grape

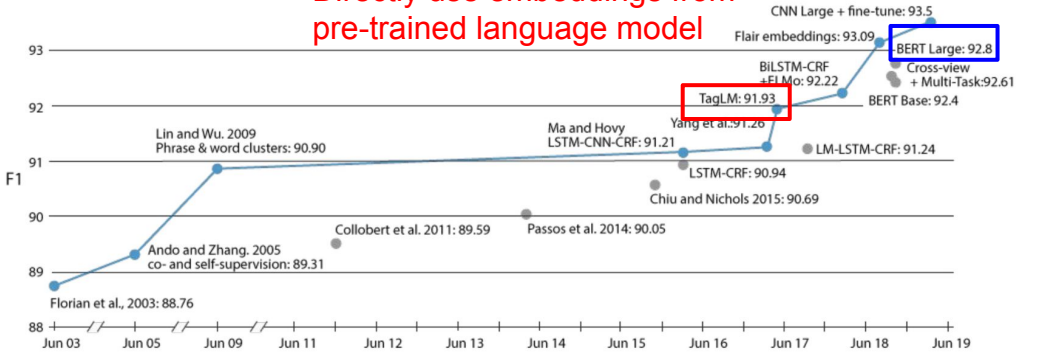
organization

apple is rumored to be working on a smartwatch, which maybe be called an "iwatch."

Source: context-of-embeddings

Directly use embeddings from pre-trained language model

Pre-train with Fine-tuning on Task



Performance on Named Entity Recognition (NER) on CoNLL-2003 (English) over time.

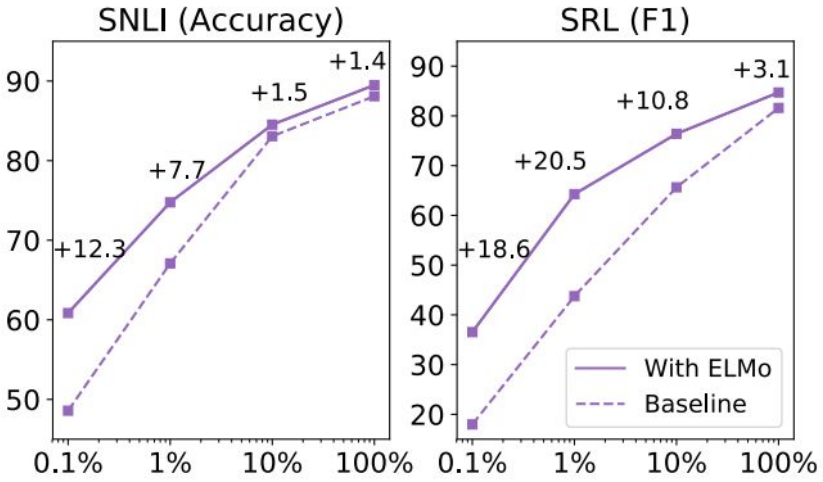
Source: [Ruber et al., 2019]

Deep Query Understanding Leveraging Pre-Trained Models

Reduce the labeling needs on target and get better performance faster

Semantic role labeling (SRL) - answering “who did what to whom”

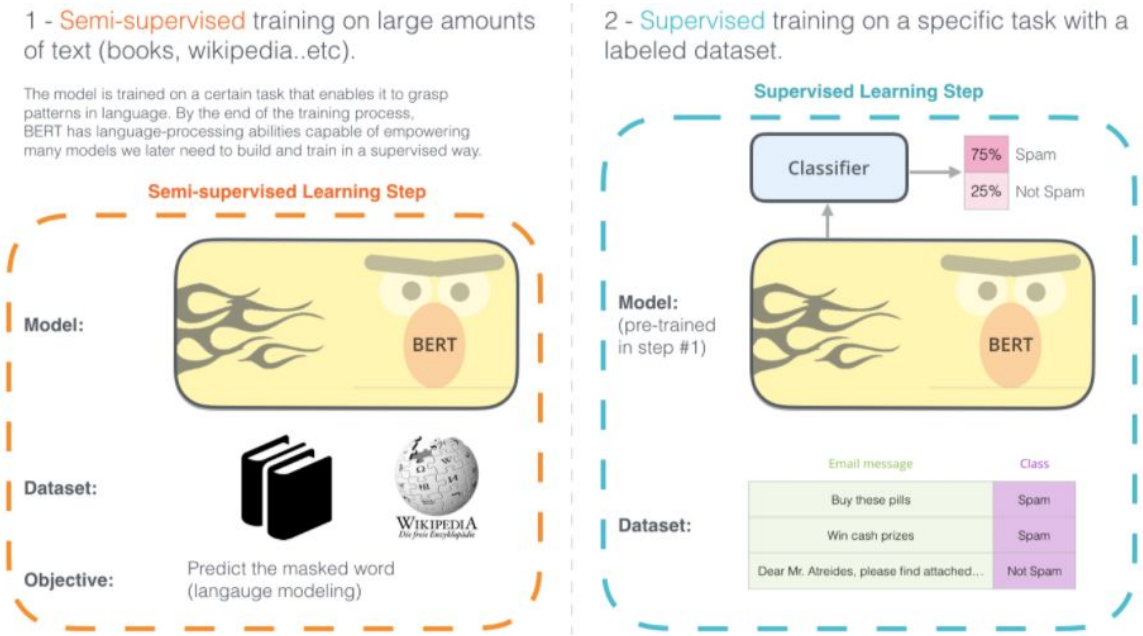
- Without transfer, peak performance at 486 epochs; with ELMo transfer, exceed at 10 epochs (-98% updates)
- With ELMo transfer, 1% data = 10% without



[Peters et al., 2018]

BERT: Query Understanding and Representation Learning

Bidirectional Encoder Representations from Transformers by Google



1. Pre-trained the model from a huge source set
2. Fine tune for the target task of interest

Source: http://jalammar.github.io/illustrated-bert/

BERT: Query Understanding and Representation Learning

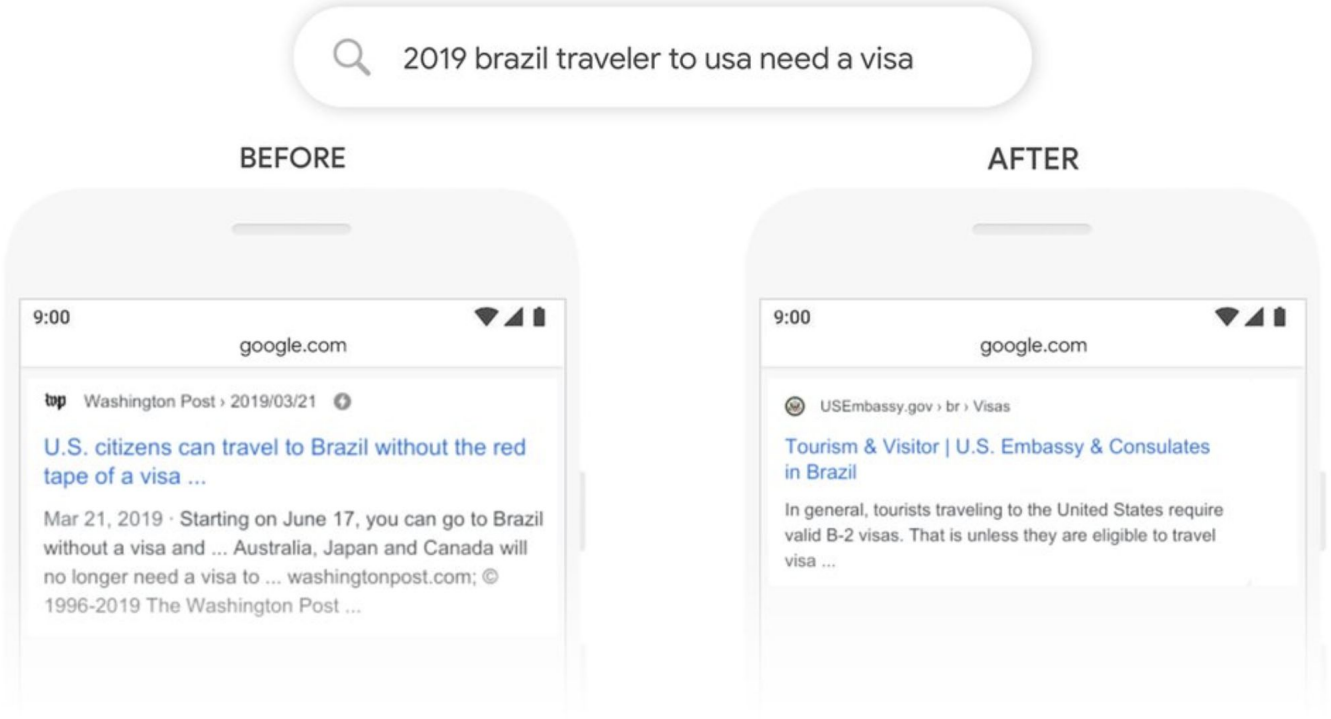
CoNLL-2003 Named Entity Recognition Test Results

BERT is effective for both fine-tuning and feature-based approaches.

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

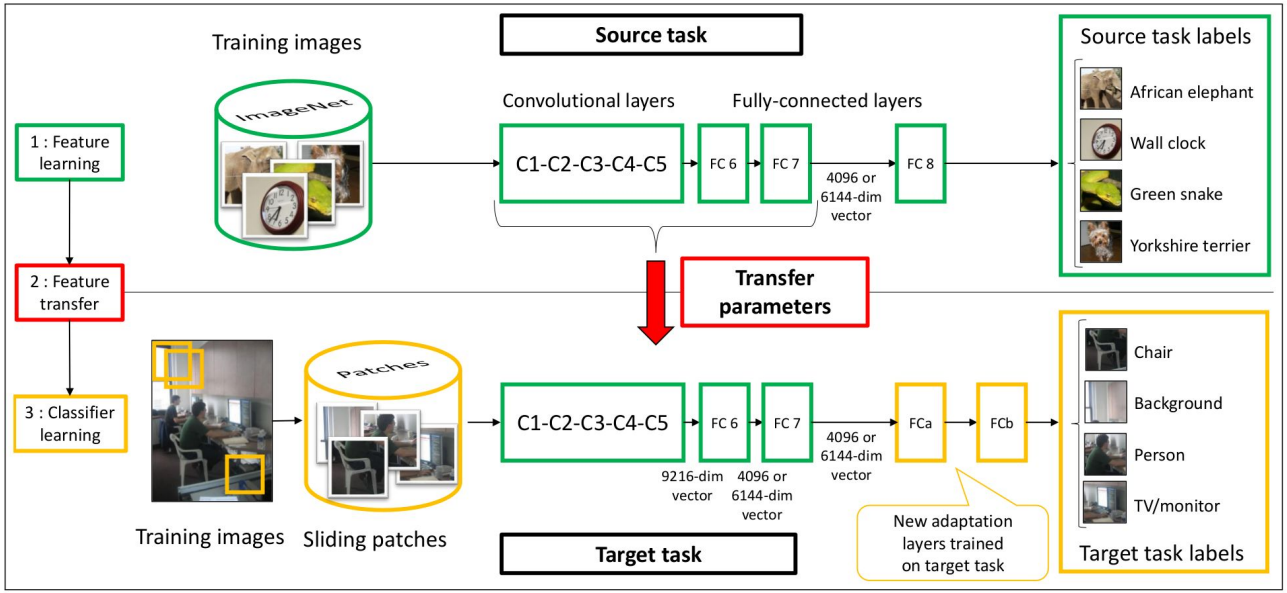
[Devlin et al., 2019]

Google Search Before and After BERT



Source: https://www.blog.google/products/search/search-language-understanding-bert/

Pre-trained Model for Image Representation

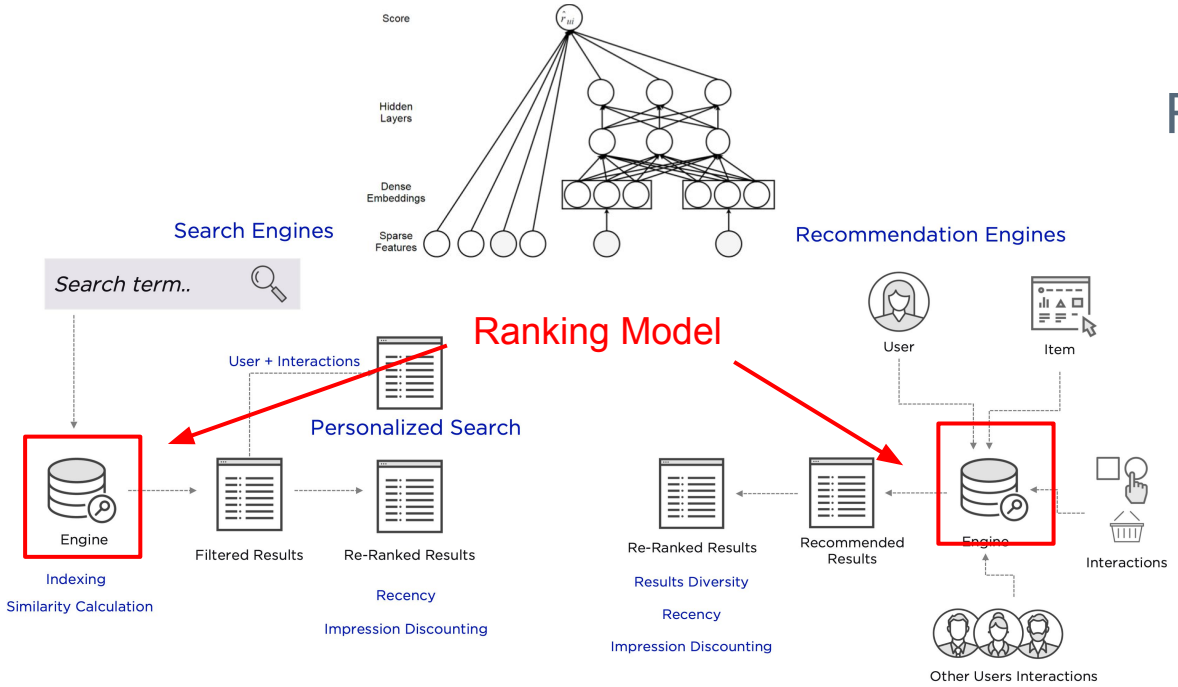


[Oquab et al., 2014]

Transfer model parameters from AlexNet to create mid-level image representations

Model and Instance Transfer for Ranking

Deep Neural Network Model for Ranking



Ranking Model

- Order relevant documents for the query/user
- Output the documents with relevancy score
- Real time ranking is usually required for production use

Source: https://conferences.oreilly.com/strata/strata-ca-2018/public/schedule/detail/63818

Deep Transfer Learning for Building Ranking Model

- Pre-trained models can be extracted to construct the features as model input; however, fine-tuning the large pre-trained model for ranking is very costly for production system
  - E.g., BERT large has 24 layers, 16 attention heads and 340m parameters, to slow for real-time ranking engine

- **Model transfer through multi-task training** with relevant ranking problems is common in practice as now users interact with multiple products on the site
  - Multimodal
  - Cross domain
  - Cross lingual

Multimodal

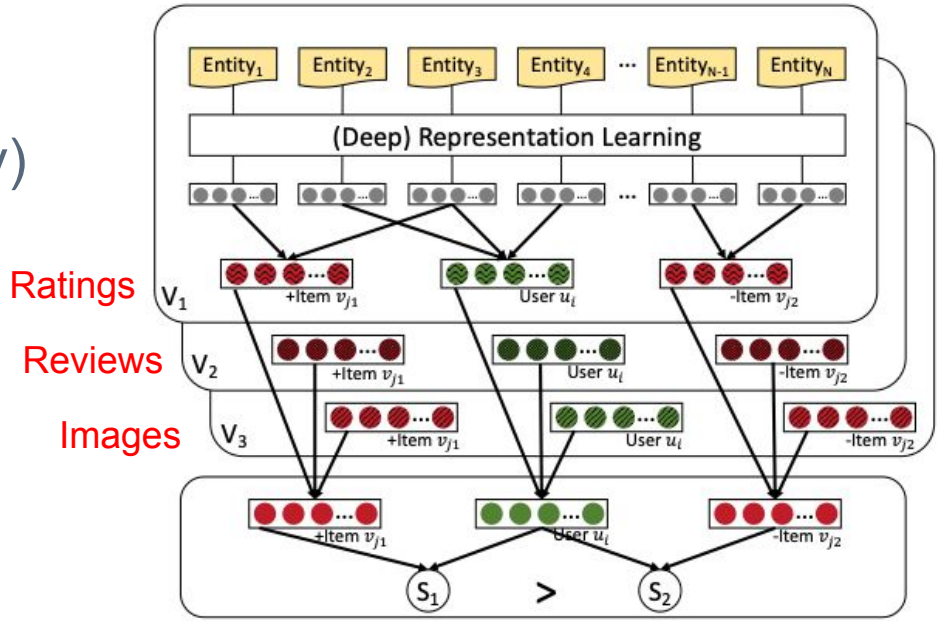
[Zhang et al., 2017]

Integrate various information between the users (u) and items (v) into a unified representation space to build product purchase model.

$$\mathbf{u} = f(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3),\ \mathbf{v} = f(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$$

f: concatenation or average

u_i, v_i: representation learnt from each view



Multimodal

Joint training of multiple information about users and items [text review, rating and images]

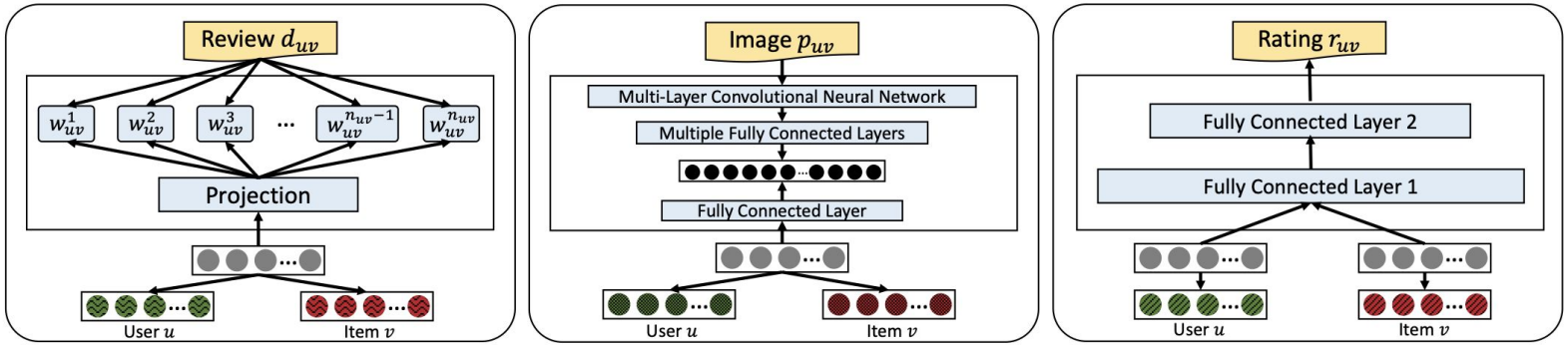
Pairwise ranking

$$\underset{\mathcal{W},\Theta}{\text{maximize}}\ \mathcal{L} = \sum_{(u,v)\in\mathcal{R}} g(\mathbf{u}, \mathbf{v}^+, \mathbf{v}^-) + \lambda_1\mathcal{L}_1 - \lambda_2\mathcal{L}_2 - \lambda_3\mathcal{L}_3$$

Negative PV-DBOW loss

Image Reconstruction loss

Rating Reconstruction loss



(a) Representation Learning of Reviews (b) Representation Learning of Images (c) Representation Learning of Ratings

Performance Summary of Joint vs. Single

The joint trained model outperform any of the single task by a large margin.

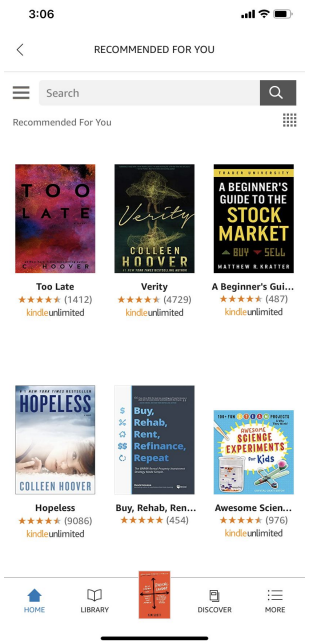
Dataset	Movies				CDs				Clothing				Cell Phones				Beauty			
Measures(%)	NDCG	Recall	HT	Prec	NDCG	Recall	HT	Prec	NDCG	Recall	HT	Prec	NDCG	Recall	HT	Prec	NDCG	Recall	HT	Prec
BPR	1.267	1.988	4.421	0.528	2.009	2.679	8.554	1.085	0.601	1.046	1.767	0.185	1.998	3.258	5.273	0.595	2.753	4.241	8.241	*1.143
BPR-HFT	*2.092	*3.255	*6.378	*0.776	*2.661	*3.570	*9.926	*1.268	*1.067	*1.819	*2.872	*0.297	*3.151	*5.307	*8.125	*0.860	*2.934	*4.459	*8.268	1.132
VBPR	0.849	1.534	2.976	0.324	0.631	0.845	2.930	0.328	0.560	0.968	1.557	0.166	1.797	3.489	5.002	0.507	1.901	2.786	5.961	0.902
DeepCoNN	3.800	4.671	10.522	0.886	4.218	6.001	13.857	1.681	1.310	2.332	3.286	0.229	3.636	6.353	9.913	0.999	3.359	5.429	9.807	1.200
CKE	4.091	5.466	11.053	1.319	4.620	6.483	14.541	1.779	1.502	2.509	4.275	0.388	3.995	7.005	10.809	1.070	3.717	5.938	11.043	1.371
Review	4.222	6.145	12.958	1.465	5.286	7.454	16.592	2.079	1.270	2.211	3.527	0.336	4.184	7.275	10.632	1.062	4.216	6.766	12.422	1.467
Image	2.648	4.035	9.489	1.048	3.191	4.564	11.547	1.379	1.393	2.481	3.773	0.354	3.777	6.439	9.444	0.932	3.310	5.288	10.280	1.211
Rating	0.432	0.700	2.242	0.234	0.528	0.747	2.394	0.248	0.377	0.732	1.219	0.112	1.506	2.706	3.845	0.369	0.876	1.442	3.322	0.313
eJRL	**4.405**	6.289	**13.292**	**1.521**	5.023	6.973	16.081	2.002	1.523	2.679	4.182	0.396	4.185	7.130	10.531	1.054	3.896	6.010	11.090	1.355
JRL	4.334	**6.334**	13.245	1.492	**5.378**	**7.545**	**16.774**	**2.085**	**1.735**	**2.989**	**4.634**	**0.442**	**4.364**	**7.510**	**10.940**	**1.096**	**4.396**	**6.949**	**12.776**	**1.546**
Review-Impr	3.20	12.43	17.24	11.09	14.43	14.99	14.11	16.86	-15.46	-11.90	-17.51	-13.36	4.72	3.86	-1.64	-0.73	13.42	13.94	12.49	6.98
Image-Impr	-35.28	-26.19	-14.15	-20.58	-30.92	-29.59	-20.59	-22.50	-7.24	-1.12	-11.75	-8.84	-5.47	-8.08	-12.63	-12.89	-10.96	-10.95	-6.91	-11.67
eJRL-Impr	7.67	15.07	20.26	15.33	8.72	7.57	10.59	12.54	1.41	6.75	-2.19	1.92	4.74	1.79	-2.57	-1.52	4.81	1.21	0.42	-1.14
JRL-Impr	5.92	15.89	19.84	13.08	16.40	16.39	15.36	17.21	15.52	19.12	8.38	13.87	9.23	7.21	1.21	2.41	18.27	17.03	15.69	12.76

Single

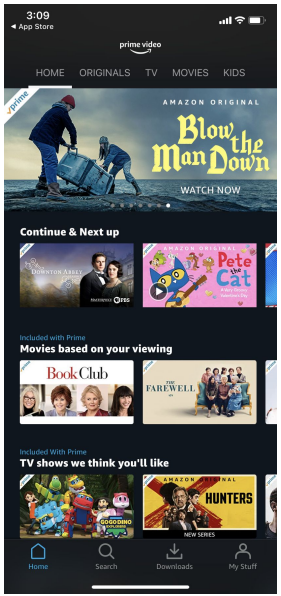
Joint

Cross Domain

Book Recommendation



Existing Movie Reviews From Another System



[Cantador et al. 2015]

- Addressing the cold-start problem
  - recommending to new users
  - cross-selling of products
- Improving accuracy
  - reducing data sparsity
  - learning more robust features
- Enhancing user models
  - discovering new user preferences
- Offering added value to recommendations
  - diversity, novelty, serendipity

Cross Domain: Multi-View

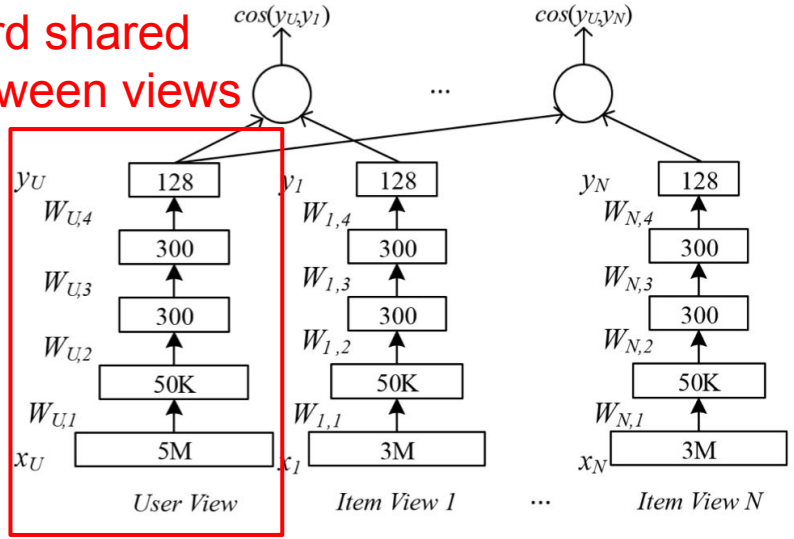
[Elkahky et al., 2015] Also called Multi-View Models

Objective:

$$\max_{W_u, W_1, .. W_v} \sum_{j=1}^{N} \frac{e^{\alpha_a \cos(Y_u, Y_{a,j})}}{\sum_{X' \in R^{d_a}} e^{\alpha \cos(Y_u, f_a(X', W_a))}}$$

Type	DataSet	UserCnt	Feature Size	Joint Users
User View	Search	20M	3.5M	/
Item View	News	5M	100K	1.5M
	Apps	1M	50K	210K
	Movie/TV	60K	50K	16K

Hard shared between views



Features

- User: user search queries and their clicked urls
- News: title, named entities in the article.
- App: title, category
- Movie/TV: title, description, genre

Cross Domain: Multi-View (cont'd)

Target: App Installation

Shallow Baselines

Single View

Multi-View

	Algorithm	All Users		New Users	
		MRR	P@1	MRR	P@1
I	Most Frequent	0.298	0.103	0.303	0.119
	CF	0.337	0.142	/	/
	CCA (TopK) [29]	0.295	0.105	0.295	0.104
	CTR [32]	0.448	0.277	0.319	0.142
II	SV- Kmeans	0.359	0.159	0.336	0.154
	SV-LSH	0.372	0.169	0.339	0.158
	SV-TopK	**0.497**	**0.315**	**0.436**	**0.268**
III	MV-Kmeans	0.362	0.16	0.339	0.156
	MV-TopK	0.517	0.335	0.466	0.297
	MV-TopK w/ Xbox	**0.527**	**0.347**	**0.473**	**0.306**

Target: News Recommendation

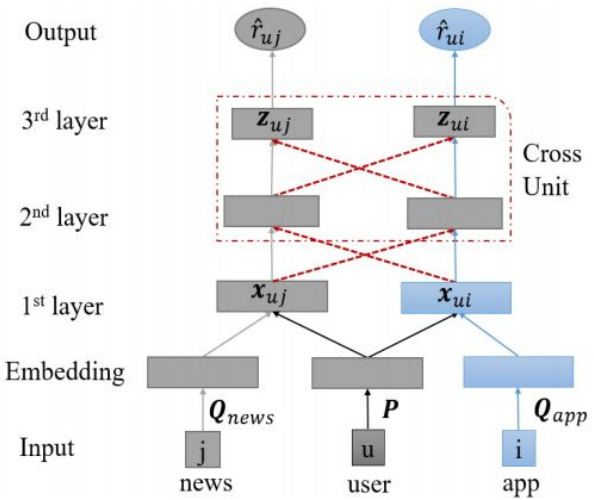
	Algorithm	All Users		New Users	
		MRR	P@1	MRR	P@1
I	Most Frequent	0.301	0.111	0.305	0.111
	CTR [32]	0.427	0.215	0.276	0.123
II	SV-Kmeans	0.386	0.192	0.294	0.143
	SV-LSH	0.45	0.247	0.34	0.186
	SV-TopK	**0.486**	**0.286**	**0.358**	**0.208**
III	MV-Kmeans	0.391	0.194	0.296	0.145
	MV-TopK	0.494	0.303	0.368	0.222
	MV-TopK w/ Xbox	**0.503**	**0.321**	**0.398**	**0.245**

- Multi-View improves upon Single-View
- Even more for new users

Kmeans, LSH and TopK different feature dimension reduction techniques used with the model

Cross Domain: Partial Model Sharing

[Hu et al., 2018] Multi-task training by using cross connection unit instead of directly hard parameter sharing



Scalar

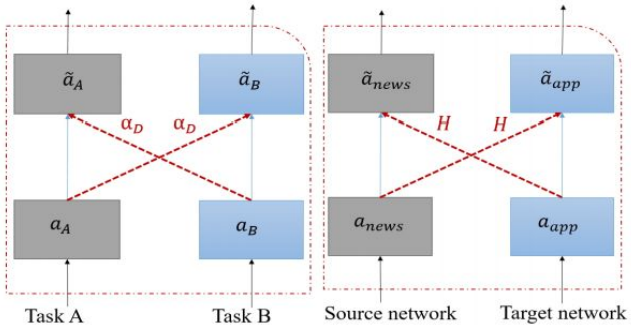
$$\tilde{a}_A^{ij} = \alpha_S a_A^{ij} + \alpha_D a_B^{ij}$$

$$\tilde{a}_B^{ij} = \alpha_S a_B^{ij} + \alpha_D a_A^{ij}$$

Matrix

$$\tilde{\boldsymbol{a}}_{app} = \boldsymbol{W}_{app}\boldsymbol{a}_{app} + \boldsymbol{H}\boldsymbol{a}_{news}$$

$$\tilde{\boldsymbol{a}}_{news} = \boldsymbol{W}_{news}\boldsymbol{a}_{news} + \boldsymbol{H}\boldsymbol{a}_{app}$$



(a) Cross-stitch unit (b) Cross connection unit

Cross Domain: Partial Model Sharing (cont'd)

MLP++: user embedding shared between but no sharing in task adaptation layers

CoNet: cross connection unit to share

SCoNet: CoNet + H sparsity penalty

Dataset	Metric	BPRMF	CMF	CDCF	MLP	MLP++	CSN	CoNet	SCoNet	improve	paired t-test
Mobile	HR	.6175	.7879	.7812	.8405	.8445	.8458*	.8480	**.8583**	1.47%	$p = 0.20$
	NDCG	.4891	.5740	.5875	.6615	.6683	.6733*	.6754	**.6887**	2.29%	$p = 0.25$
	MRR	.4489	.5067	.5265	.6210	.6268	.6366*	.6373	**.6475**	1.71%	$p = 0.34$
Amazon	HR	.4723	.3712	.3685	.5014	.5050*	.4962	.5167	**.5338**	5.70%	$p = 0.02$
	NDCG	.3016	.2378	.2307	.3143	.3175*	.3068	.3261	**.3424**	7.84%	$p = 0.03$
	MRR	.2971	.1966	.1884	.3113*	.3053	.2964	.3163	**.3351**	7.65%	$p = 0.05$

- With partial sharing the task adaptation layers, CoNet outperforms MLP++
- The structure of H matrix is also critical, the sparse version performs the best

Cross Domain: Partial Representation Layers Sharing

Recall Domain Separation Networks (DSN) for Domain Adaptation

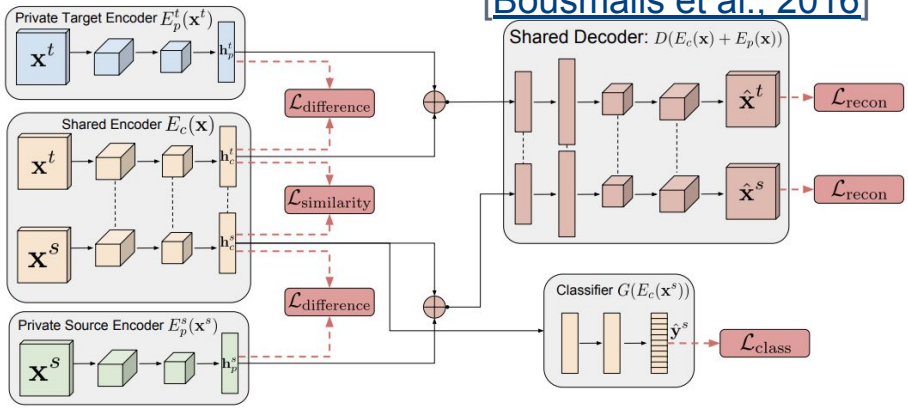
$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\,\mathcal{L}_{\text{recon}} + \beta\,\mathcal{L}_{\text{difference}} + \gamma\,\mathcal{L}_{\text{similarity}}$$

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{N_s} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^s, \hat{\mathbf{x}}_i^s) + \sum_{i=1}^{N_t} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^t, \hat{\mathbf{x}}_i^t)$$

$$L_{\text{difference}} = \left\| \mathbf{H}_c^{s\top} \mathbf{H}_p^s \right\|_F^2 + \left\| \mathbf{H}_c^{t\top} \mathbf{H}_p^t \right\|_F^2$$

$$\mathcal{L}_{\text{similarity}}^{\text{DANN}} = \sum_{i=0}^{N_s+N_t} \left\{ d_i \log \hat{d}_i + (1-d_i)\log(1-\hat{d}_i) \right\}$$

[Bousmalis et al., 2016]

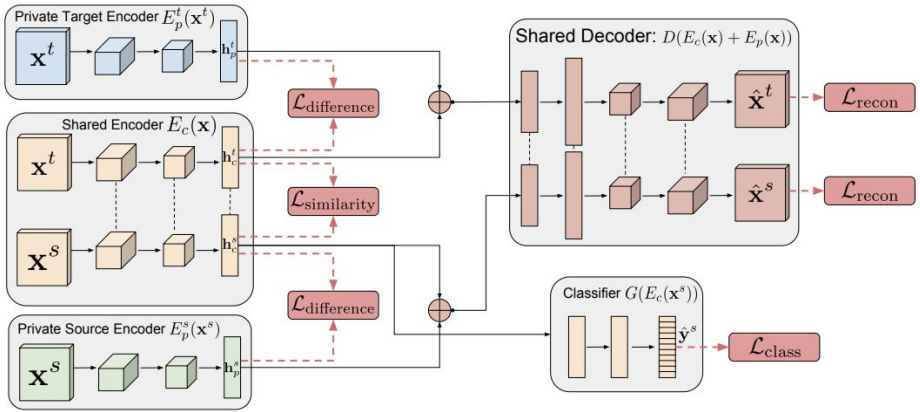


[Kanagawa et al., 2018] Adapt DSN to do cross-domain recommendation for News (target) with Video (source)

Cross Domain: Partial Representation Sharing (cont'd)

NN: without partial domain adaptation (no L_{diff}, $L_{similar}$)

- Partial domain adaptation improves the ranking performance



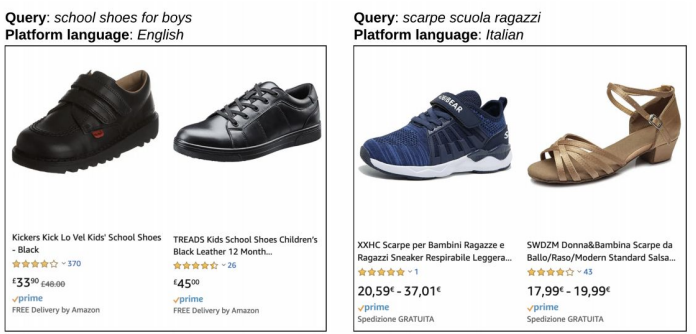
Neural network models are selected according to the value of nDCG@100

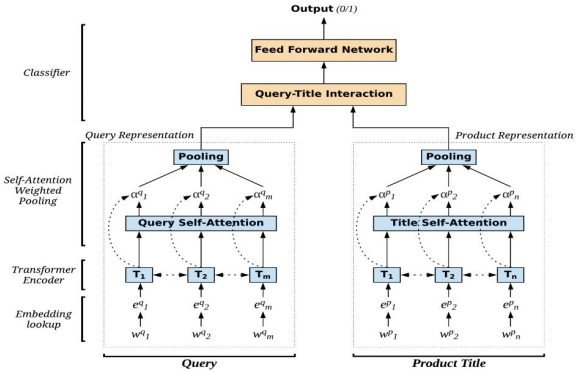
Method	nDCG@1	nDCG@10	nDCG@50	nDCG100
I-DSN	0.0440 ± 0.0256	0.1881 ± 0.0282	0.2693 ± 0.0211	0.2785 ± 0.0192
DSN	**0.0618 ± 0.0212**	**0.2133 ± 0.0154**	**0.2873 ± 0.0151**	**0.2945 ± 0.0153**
NN	0.0415 ± 0.0211	0.1938 ± 0.0131	0.2735 ± 0.0102	0.2797 ± 0.0107
POP	0.0398 ± 0.0007	0.2099 ± 0.0012	0.2790 ± 0.0016	0.2871 ± 0.0010

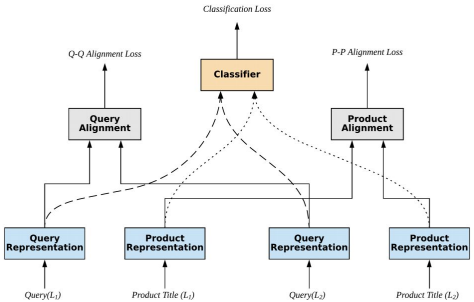
Cross-Lingual for Product Search

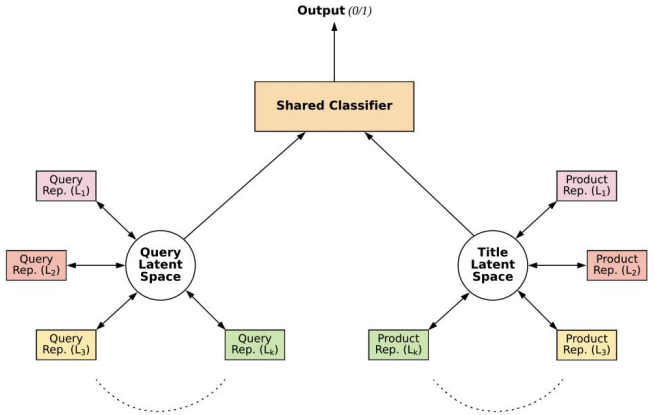
[Ahuja et al., 2020]

- multi-tasking product searches in different languages









Mono-lingual

Bi-lingual

Multi-lingual

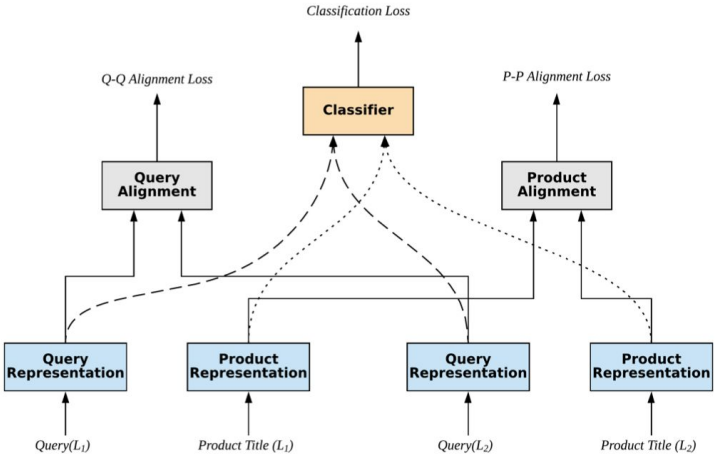
Query-Product Pairing from Multiple Languages

1. Hard shared classifier among query-product matching in different languages
2. Soft sharing on query-query, product-product via alignment losses

$$\mathcal{L}_{l_i}^{cls} = \sum_{(q_j^{l_i}, p_j^{l_i}), y_j^{l_i} \in d_{l_i}^{cls}} y_j^{l_i} \log f^{cls}(q_j^{l_i}, p_j^{l_i}) + \alpha * (1 - y_j^{l_i})(1 - \log f^{cls}(q_j^{l_i}, p_j^{l_i}))$$

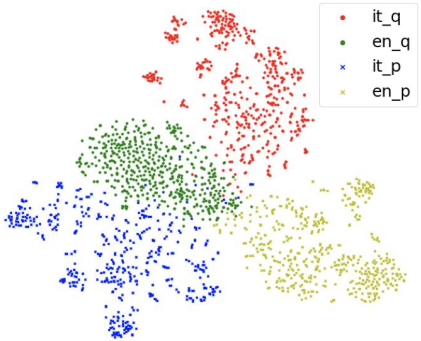
$$\mathcal{L}_{l_i,l_j}^{QQ} = \sum_{(q^{l_i}, q^{l_j}) \in d_{l_i,l_j}^{qq}} ||v_q^{l_i} - v_q^{l_j}||_2^2,$$

$$\mathcal{L}_{l_i,l_j}^{PP} = \sum_{(p^{l_i}, p^{l_j}) \in d_{l_i,l_j}^{pp}} ||v_p^{l_i} - v_p^{l_j}||_2^2$$

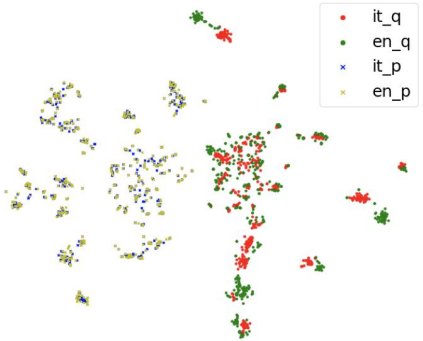


Resolve Data Sparsity and Better Alignment

Model	Dataset														
	FR			ES			IT			EN			DE		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Mono-lingual	40.93	44.15	42.48	41.54	44.72	43.07	49.93	49.86	49.89	38.98	38.51	38.74	34.30	37.61	35.88
Translate (EN)	35.89	49.93	41.76	31.08	56.01	39.97	34.44	49.75	40.70	38.98	38.51	38.74	22.95	34.85	27.68
Translate (IT)	24.68	**60.89**	35.12	23.86	54.05	33.10	49.93	49.86	49.89	12.34	**49.09**	19.72	12.56	48.09	19.92
MV-LSTM [32]	30.59	53.09	38.82	30.19	43.12	35.51	30.08	52.87	38.34	30.97	31.93	31.44	29.03	**51.07**	37.02
MatchPyramid [23]	19.02	53.71	28.09	17.63	**63.61**	27.61	22.09	**55.07**	31.53	9.29	40.93	15.15	13.01	46.40	20.33
MAML [11]	42.97	42.26	42.61	43.88	51.76	47.50	49.04	52.10	50.52	32.06	41.60	36.21	36.85	37.85	37.34
LAPS-2(+ FR)	NA	NA	NA	47.19	45.13	46.14	50.42	51.08	50.75	38.38	36.13	37.22	37.47	41.12	39.21
LAPS-2(+ ES)	41.25	47.72	44.25	NA	NA	NA	52.34	48.41	50.30	37.07	39.01	38.01	39.71	40.24	39.97
LAPS-2(+ IT)	49.53	42.89	45.97	40.94	54.46	46.74	NA	NA	NA	36.94	41.95	39.29	43.18	40.68	41.89
LAPS-2(+ EN)	50.75	46.15	48.34	52.17	52.89	52.33	49.87	54.27	51.98	NA	NA	NA	43.51	43.18	43.35
LAPS-2(+ DE)	45.90	48.81	47.31	51.51	54.08	52.76	50.71	51.05	50.88	37.74	40.09	38.88	NA	NA	NA
LAPS-3(ES+DE+UK)	NA	NA	NA	51.74	53.05	52.39	NA	NA	NA	39.52	40.58	40.04	44.50	42.09	43.26
LAPS-3(IT+DE+UK)	NA	NA	NA	NA	NA	NA	55.50	49.68	52.43	38.81	40.49	39.63	**45.61**	41.12	43.25
LAPS-5	**53.47**	52.28	**52.87**	**56.83**	54.63	**55.71**	**57.93**	52.23	**54.93**	**39.54**	41.63	**40.56**	44.80	43.77	**44.28**



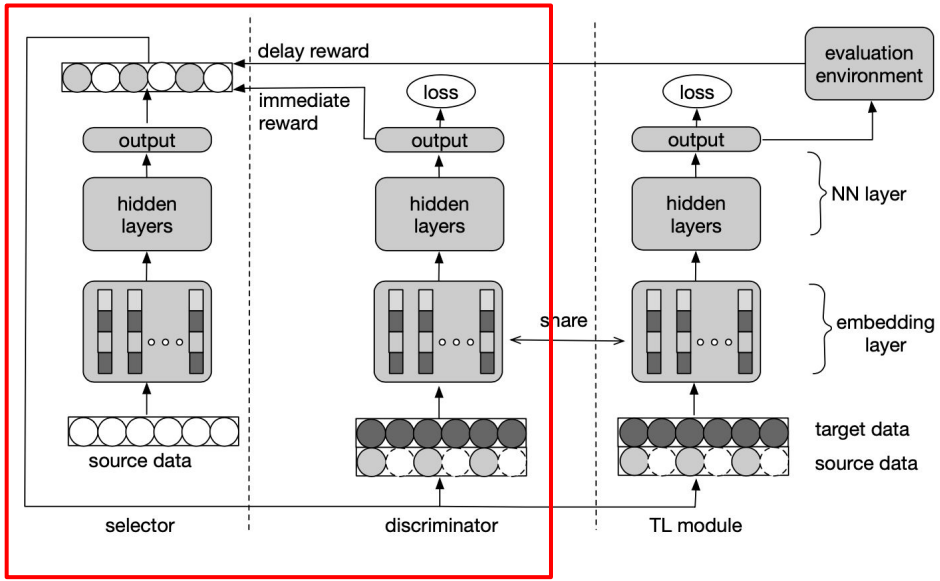
Mono-lingual



Multi-lingual

- Comparing to Mono-lingual, the LAPs-5 (Multi-lingual) improves the most for languages with fewer training data, least for EN.
- Multi-lingual aligns queries and items across multiple languages much better than Mono-lingual while maintaining query-item separability.

Selective Instance Transfer



Similar to GAN but with a Selector for Source

[Wang et al., 2019]

Data selector updated with

- immediate reward (discrimination)
- delay reward (target improvement)

$$\theta^* = \arg\min_\theta \{[\mathbb{E}_{d_i \sim p_{tgt}(d)} \log \mathcal{D}(d_i) + \mathbb{E}_{d_i \sim p_\theta(d)} \log(1 - \mathcal{D}(d_i))] + \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} [L'(y_i, \mathcal{M}_{p_\theta}(d_i)) - L'(y_i, \mathcal{M}(d_i))]\}$$

$$= \arg\max_\theta \{\underbrace{\mathbb{E}_{d_i \sim p_\theta(d)} - \log(1 - \mathcal{D}(d_i)))}_{\text{Immediate Reward}} + \underbrace{\frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} [L'(y_i, \mathcal{M}(d_i)) - L'(y_i, \mathcal{M}_{p_\theta}(d_i)]\}}_{\text{Delayed Reward}}.$$

Performance Improvement by Selective Taking Source

Src-only: source data used for model training

Tgt-only: target data used for model training

TL Method: source and target joint learnt the model with representation layers shared

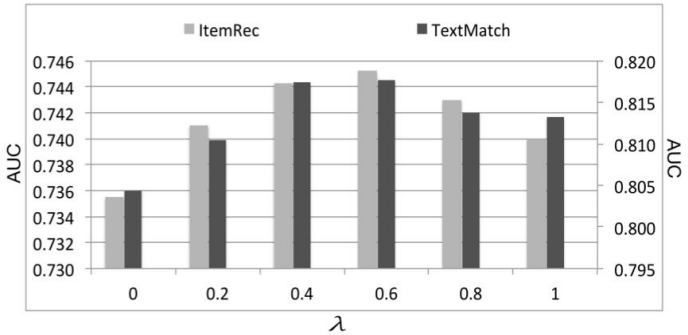
Methods	Item Recommendation		Text Matching	
	ACC	AUC	ACC	AUC
Src-only	0.8905	0.6913	0.7112	0.7087
Tgt-only	0.9013	0.7031	0.7300	0.7663
TL Method	0.9097	0.7212	0.7453	0.8044
Ruder and Plank	0.9120	0.7328	0.7521	0.8062
RTL	0.9125	0.7331	0.7672	0.8163
MGTL	**0.9153**†	**0.7452**	**0.7782**†	**0.8247**

- TL Method improves upon Tgt-only
- With selected samples from source, the performance can be further improved

Performance Improvement by Selective Taking Source

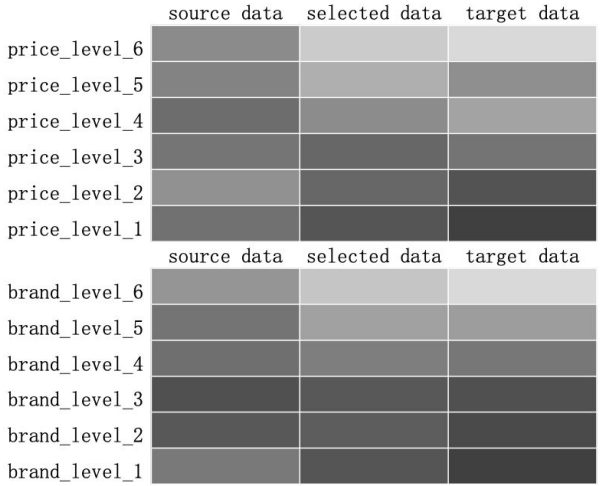
λ is to weight the importance of delay reward and immediate reward: only delayed reward with λ = 0.

- In general, it is beneficial to consider both delay and immediate rewards.



Feature distribution

- Selected samples are more alike the target than not selected



Key Takeaways

- Pre-trained models are great source models to extract feature representations for text and image for query and document understanding
- Ranking model can be improved through model transfer, doing multi-task training with related source data
  - Multimodal to leverage different information between user and item
  - Cross-domain recommendations
  - Cross-lingual models to amend for data sparsity
- Several ways to share model: hard sharing, soft sharing
- Selective instance transfer help improve transfer learning efficiency by focusing on relevant knowledge for transfer

Agenda

1 Introduction

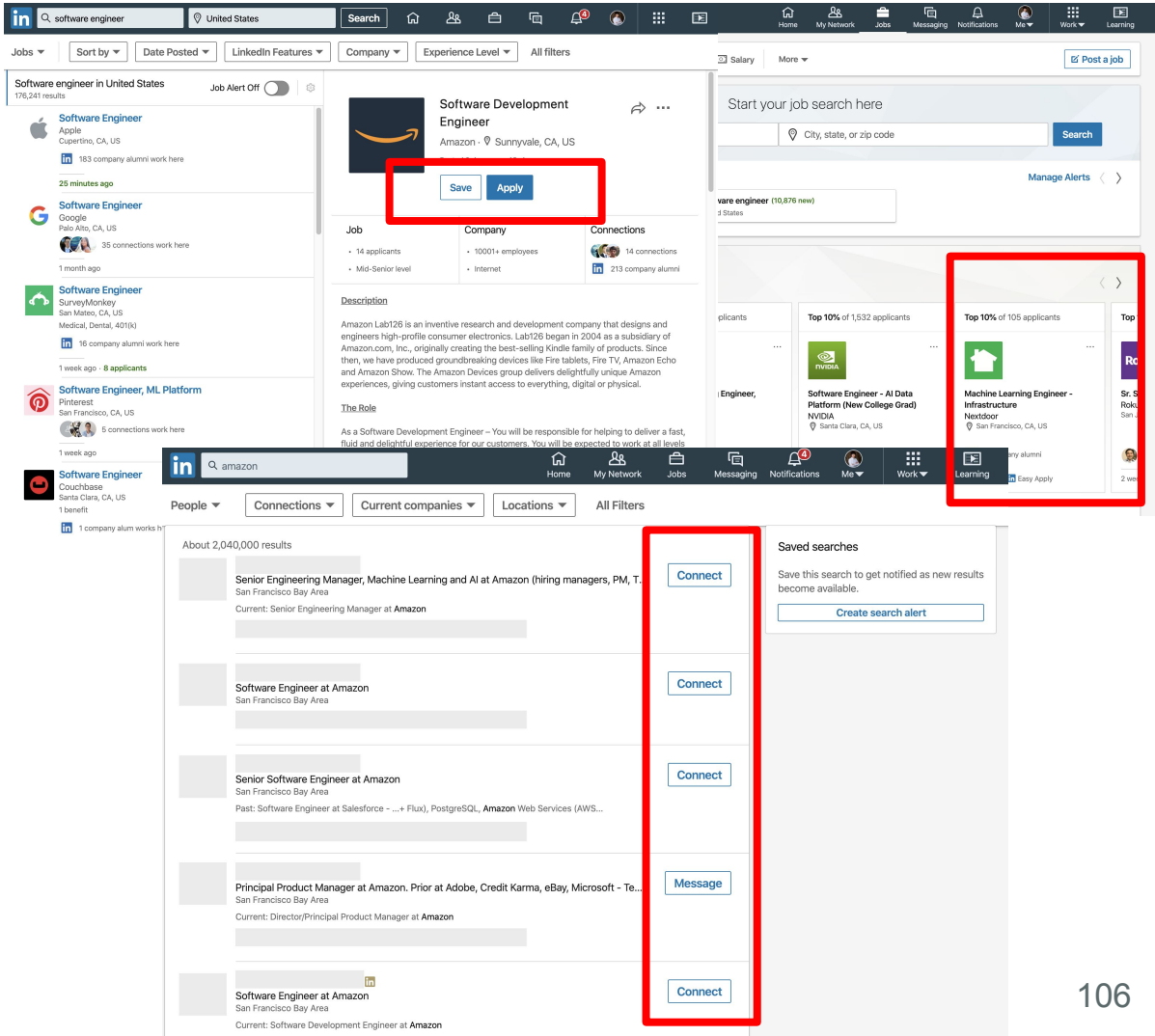
2 Deep Transfer Learning

3 Deep Transfer Learning for Search and Recommendation

4 **Applications at LinkedIn**

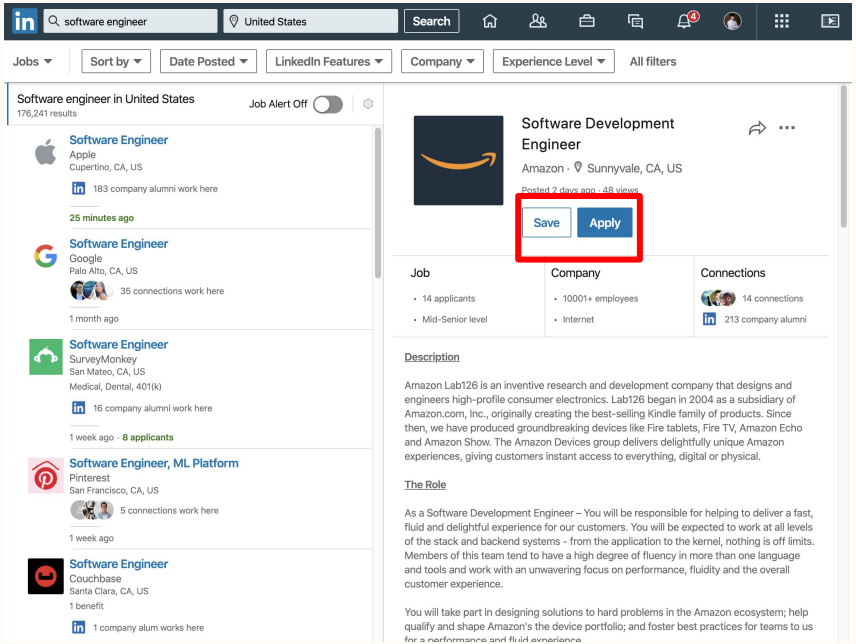
Multiple Products Capture User Behaviors

Same intent, various actions

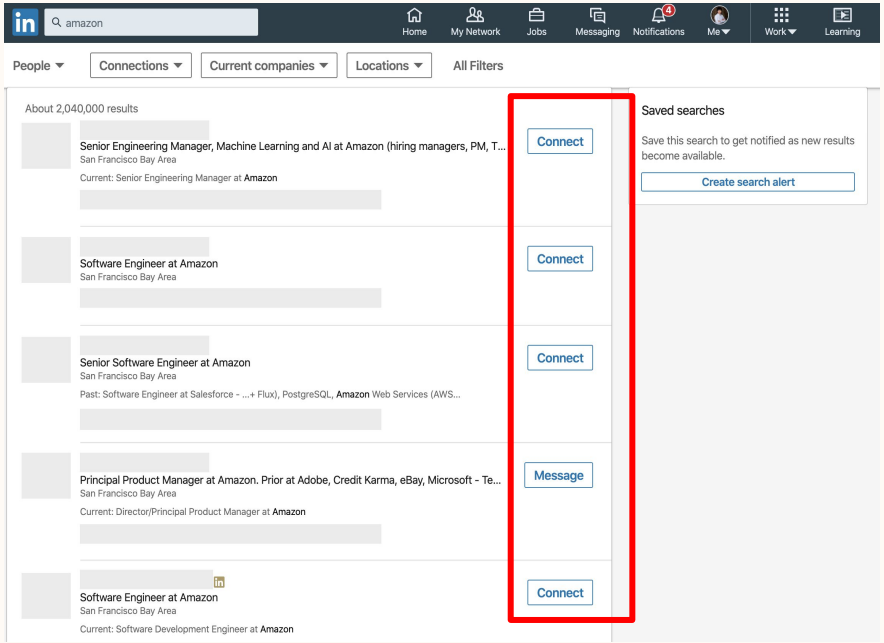


Each product traditionally captures its own signals

Job Search



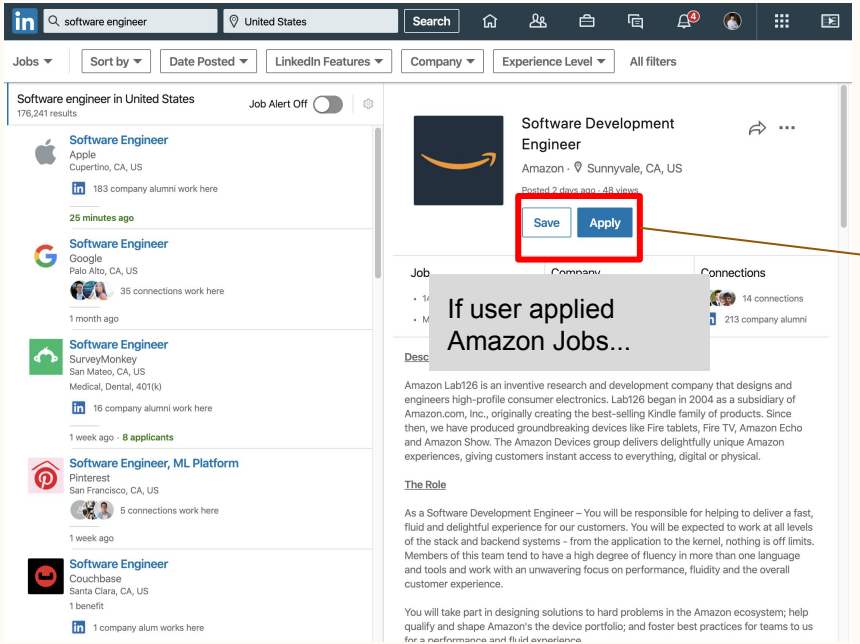
People Search

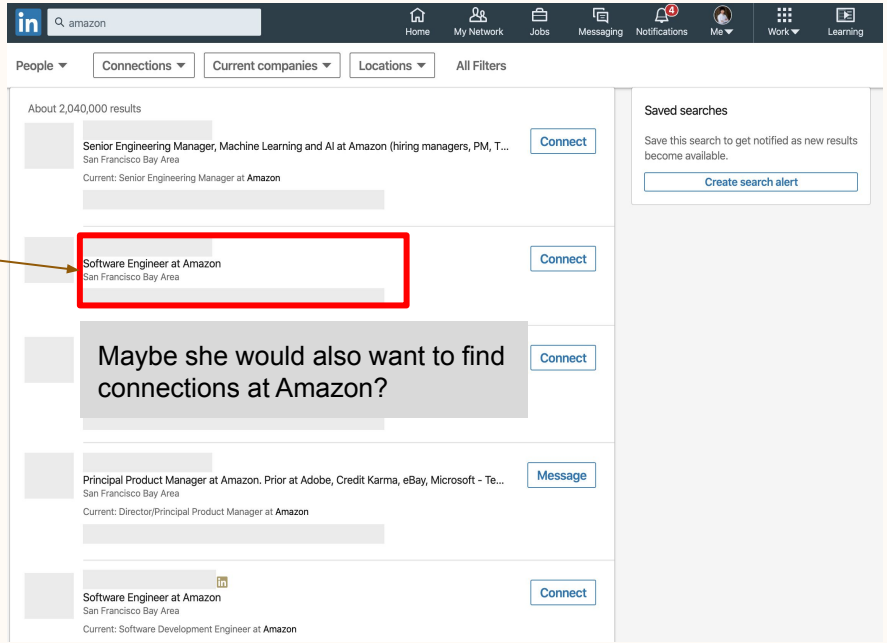


Could signal from one product help the other?

Job Search

People Search





Transfer Learning by Feature Sharing

- **User Activity Features** from source domain can help with improvements in target domain
  - E.g. Member's historical job apply activities on company → company interest
  - Member's historical people search → member's probability to reply to company sales inmail

Target Domain	Source Domain	Features Added into Production
People Search	Jobs Recommendation, Contents	Inferred job seeker, hiring manager, content contributor, content consumer
Flagship Search	Jobs Recommendation, Contents	Inferred job seeker, hiring manager, content contributor, content consumer
Flagship Search	Job Recommendation, Recruiter Search	Job apply, recruiter inmail accept/reject
Notification	Feed, Message	Overall desktop/mobile session, share, comment, like, message, inmail etc.

Examples of Cross-domain User Activity Features

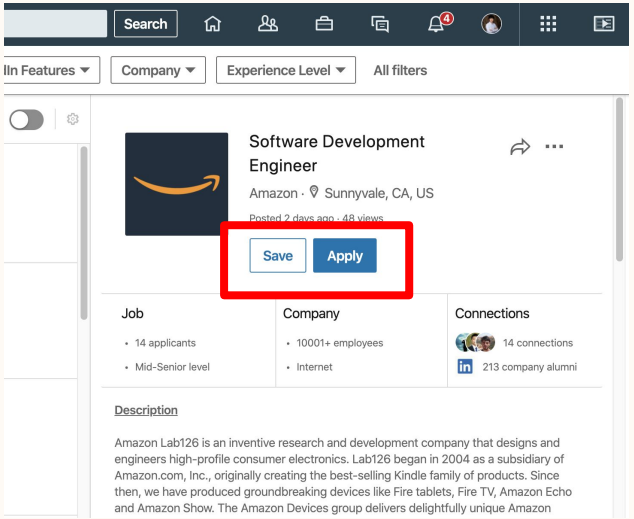
Transfer Learning by Feature Sharing

Raw activity feature sharing has its disadvantages:

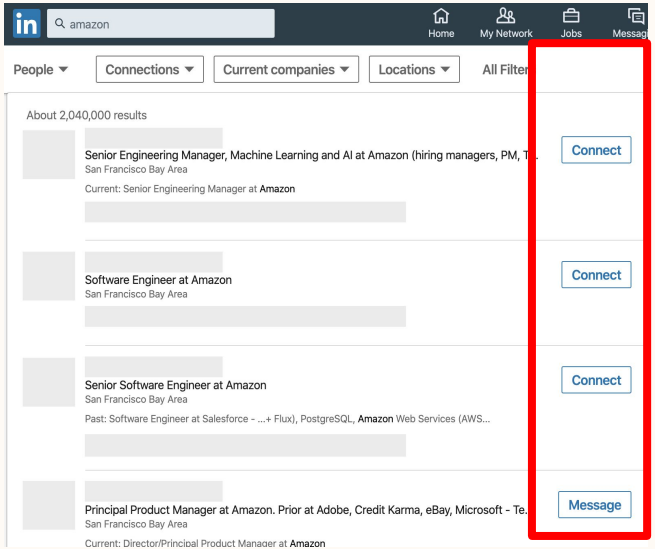
- Member level aggregations
  - limited signals for document side
- (Member, Document) level aggregations
  - Too sparse because the dimension of (Member, Document) is huge
- (Member, Document Attribute) level aggregations (i.e., Title, Company, Skill)
  - Too many combinations
  - knowledge about overall member preferences are still scattered

What if we can “summarize” user’s intention based on user activity signals...

Job Search



People Search



User Behavior

“I’m a job seeker looking for opportunity in tech industry
And I want to get connected with fold that can help me with it

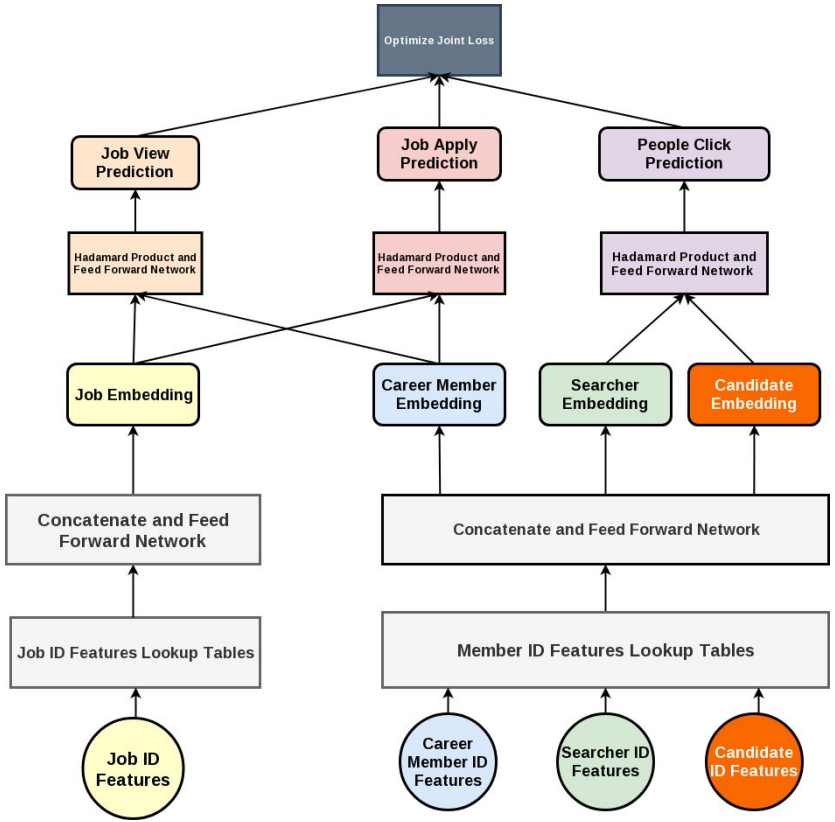
Drives Search and Recommendation Results

Feature Representation Transfer via Unified Embedding Learning

Deep Transfer Learning to generate **entity latent embeddings supervised by multiple user activities**

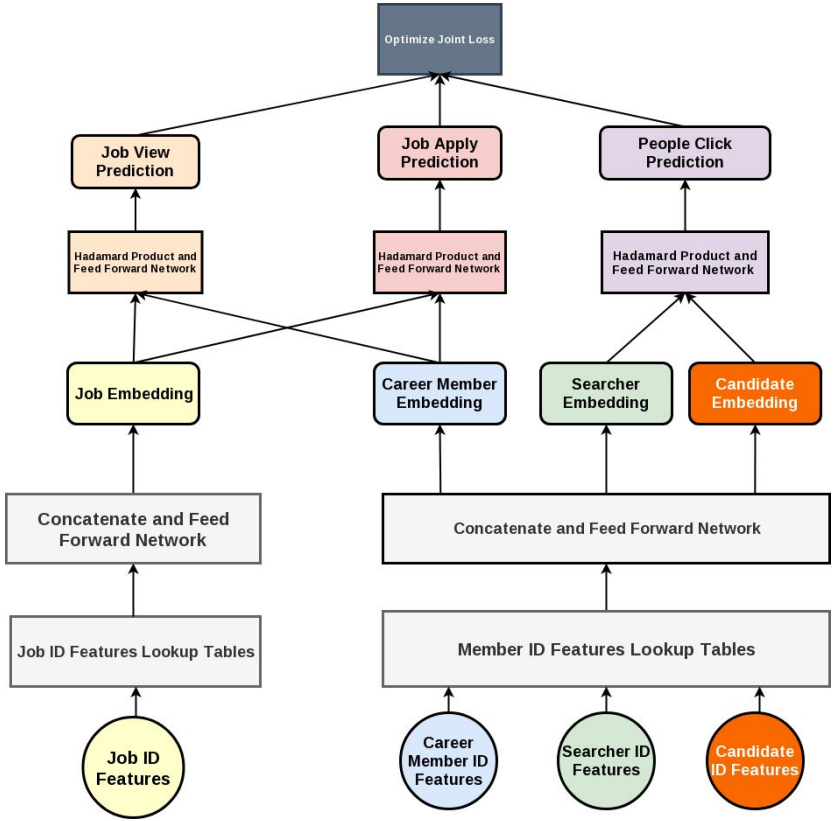
- **Use Model Transfer + Feature Representation Transfer**
- **Model Transfer**: generate entity latent embeddings by joint training
  - Member
  - Job
  - Company
  - ...
- **Feature Representation Transfer**: share the embedding as horizontal features

Feature Representation Transfer via Unified Embedding Learning



- Idea: **Multi-task Learning**
- Learn general embeddings on shared entities (Member, Items) between different recommendation tasks
- Learned embedding is generalized over all tasks and good as horizontal feature

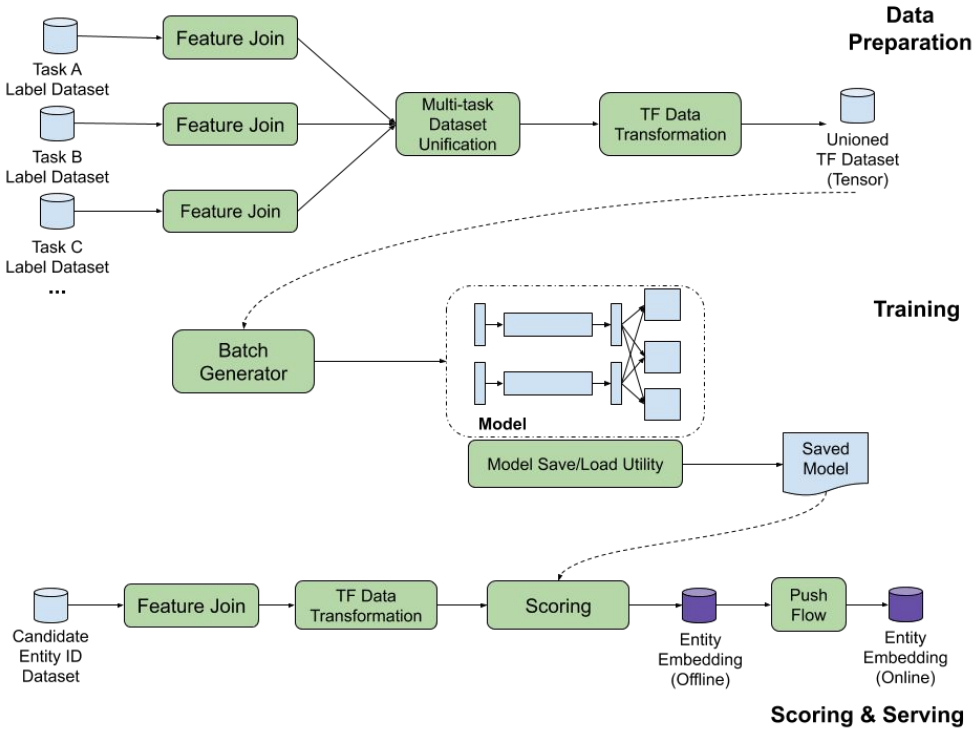
Feature Representation Transfer via Unified Embedding Learning



- Objective: User actions from multiple member-item objectives
  - E.g. People Search Click, Job Apply, Job View
- Features
  - Member and Item Profile e.g. title, skill, company etc.,
  - Member activity history features
- Model <- Model Transfer
  - Multi-task learning network with **hard parameter sharing** for each shared entity (member, job)
- Output: <- Feature Representation Transfer
  - Member and Item embeddings, usable as features in downstream tasks (e.g. People Search Ranking)

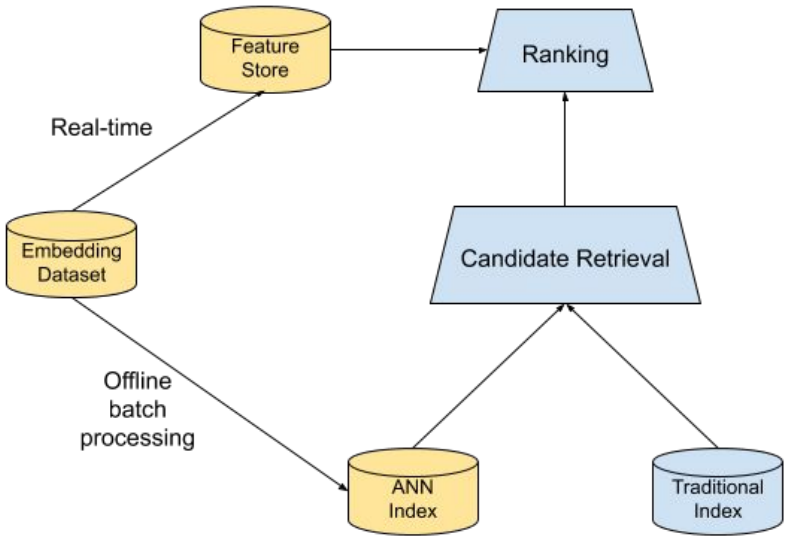
Infrastructure - Training

- Built Multi-task learning Framework to support fast setup of model transfer for verticals
  - Utility for assembling multi-task training dataset from multiple task datasets
  - Model skeleton to simplify task alternating setup and save partial model graphs



Infrastructure - Serving

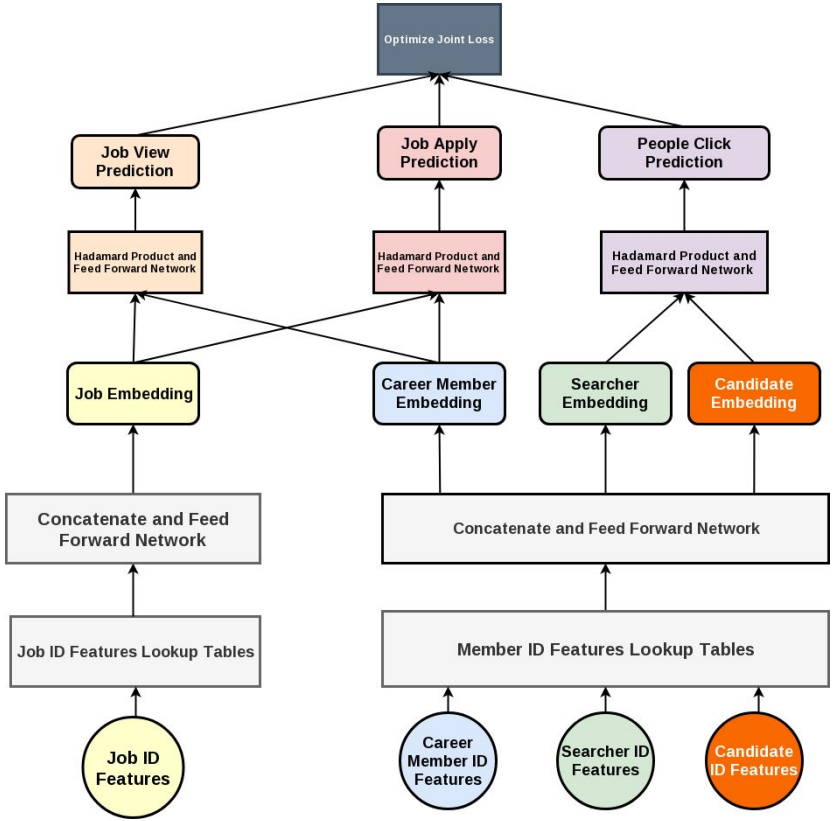
- Make embedding serving efficient
  - Ranking stage
    - Serve in online feature store
    - Store-side pre-computation to reduce latency (e.g. cosine similarity)
  - Retrieval stage
    - Approximate kNN index with embedding
    - Merge the kNN search results with traditional method to be compatible with existing search infra



Evaluation Results

Case Study 1: People Search + Careers Job Recommendation

- Source: Job Recommendation
- Target: People Search
- Joint training of People Search click, Job Apply, Job View signals
- Transfer the insights by using the learned member and job representation as features in People Search Ranking model (Feature Representation Transfer)



Evaluation Results

Case Study 1: People Search + Careers Job Recommendation

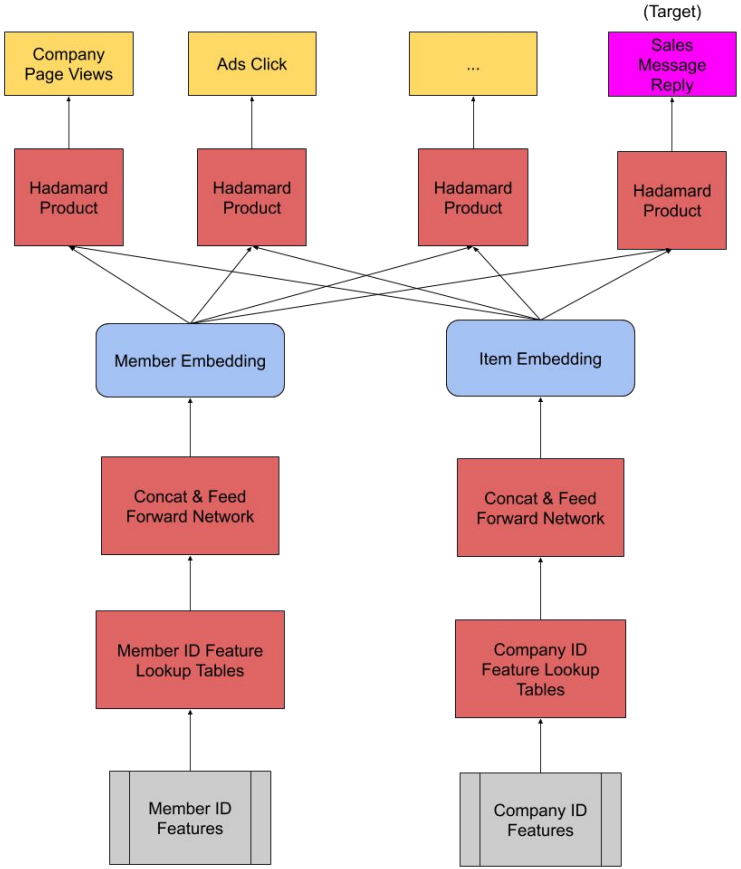
- Source: Job Recommendation
- Target: People Search
- Joint training of People Search click, Job Apply, Job View signals
- Transfer the insights by using the learned member and job representation as features in People Search Ranking model (Feature Representation Transfer)

Add embeddings trained with...	AUC Lift	NDCG@25 Lift
Tree Model Baseline (no embedding added)	0.00%	0.00%
+ People Search Embedding (Single-task)	1.25%	0.85%
+ Multi-task Embedding	**1.82%**	**1.50%**

Evaluation Results

Case Study 2: Member-Company Recommendation Tasks

- Source: Ads, General Inmail, Recruiter Inmail
- Target: Linkedin Sales Solution
- 6 Tasks related to Member-Company Entity
- Multi-modal approach to conduct model transfer from source to target



Evaluation Results

Case Study 2: Member-Company Recommendation Tasks

- Source: Ads, General Inmail, Recruiter Inmail
- Target: Linkedin Sales Solution
- 6 Tasks related to Member-Company Entity
- Multi-modal approach to conduct model transfer from source to target

Task Setup	Validation AUC Lift
Baseline (LSS single task)	0.00%
LSS + 5 verticals	2.58%

Conclusion

Why Transfer Learning?

- Improve learning performance for systems with sparse training data or even no labels
- Bridge the generalization gap from the related systems to a new one
- 3 Categories:
  - Model Transfer:
    - adapt learnt model parameters
  - Feature Representation Transfer:
    - learning shareable features to represent query, document, user etc.
  - Instance Transfer:
    - borrow data instances for training

Deep Transfer Learning - Motivation

- With flexible and complicated network structure, DTL can more effectively learn hidden transferable knowledge

- DTL can be better integrated with deep models widely used in search and recommendation applications.

Deep Transfer Learning - Key Takeaways

Model Transfer

- Use sequential or joint training to achieve model transfer from source to target

Feature Representation Transfer

- Share the lower layers of the network from source to target. They can be directly used or adapted through model transfer techniques
- Use domain adaptation by adding loss terms on source/target distance when the target label is sparse and the feature representation space needs to be aligned between source and target

Instance Transfer

- Selected the data instance from source to augment target data when feature space is homogeneous

Deep Transfer Learning - Key Takeaways

How to achieve the most effective transfer?

Principle:

- Relevancy between source and target domain determines the transfer upper bound
- Find the optimal setting to reach maximum effectiveness
  - Model Transfer
    - Tune number of parameters/layers shared, learning rates in fine-tuning, regularization
  - Feature Representation Transfer
    - Find the right distance measurement between source and target feature space
    - Control the alignment of 2 feature spaces
  - Instance Transfer
    - Find balance between long/short-term rewards in selector

Deep Transfer Learning In Search and Recommendation

Bring knowledge transfer to improve the performance of deep neural networks for search and recommendation systems

- Understanding
  - Use pre-trained model to extract user intent and document representation from text, images, profiles, etc. in query and document side
- Ranking
  - Model transfer (sometimes coupled with instance selection) to leverage matching signals and knowledge across applications to improve ranking performance
  - Fine-tuning pre-trained model for ranking tasks could be expensive
  - Use joint training to build cross-domain recommendation models to link relevant ranking tasks

Our own experience...

- Use model transfer to learn user behavior signals representation across related search and recommendation products
- Use feature representation transfer to improve vertical product models performance with the generalized user behavior insights
- Dedicated infrastructure support could make model iteration and feature representation serving more efficient and streamlined

Thank You!

More Info:
https://bit.ly/www2020-dtl-tutorial

LinkedIn

References

Introduction

- Ahuja, Aman, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K. Reddy. "Language-Agnostic Representation Learning for Product Search on E-Commerce Platforms." In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 7-15. 2020.
- Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." *IEEE Transactions on knowledge and data engineering* 22, no. 10 (2009): 1345-1359.
- Lawrence, Neil D., and John C. Platt. "Learning to learn with the informative vector machine." In *Proceedings of the twenty-first international conference on Machine learning*, p. 65. 2004.
- S. J. Pan, I. W. Tsang, J. T. Kwok and Q. Yang, "Domain Adaptation via Transfer Component Analysis," in *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199-210, Feb. 2011.
- Dai, Wenyuan, Qiang Yang, Gui-Rong Xue, and Yong Yu. "Boosting for transfer learning." In *Proceedings of the 24th international conference on Machine learning*, pp. 193-200. 2007.
- Zhang, Shuai, et al. "Deep learning based recommender system: A survey and new perspectives." ACM Computing Surveys (CSUR) 52.1 (2019): 1-38.
- Tan, Chuanqi, et al. "A survey on deep transfer learning." International conference on artificial neural networks. Springer, Cham, 2018.
- Ruder, Sebastian. "An overview of multi-task learning in deep neural networks." arXiv preprint arXiv:1706.05098 (2017).
- Wang, Mei, and Weihong Deng. "Deep visual domain adaptation: A survey." *Neurocomputing* 312 (2018): 135-153.

Deep Transfer Learning

Model Transfer

- Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. "A survey on deep transfer learning." In *International conference on artificial neural networks*, pp. 270-279. Springer, Cham, 2018.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.
- Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. "How transferable are features in deep neural networks?." In *Advances in neural information processing systems*, pp. 3320-3328. 2014.
- Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009.
- Liu, Xiaodong, Pengcheng He, Weizhu Chen, and Jianfeng Gao. "Multi-task deep neural networks for natural language understanding." *arXiv preprint arXiv:1901.11504* (2019).
- Duong, Long, Trevor Cohn, Steven Bird, and Paul Cook. "Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser." In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 845-850. 2015.

Deep Transfer Learning (cont'd)

- Ma, Jiaqi, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts." In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1930-1939. 2018.

Feature Representation Transfer

- Sharif Razavian, Ali, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. "CNN features off-the-shelf: an astounding baseline for recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 806-813. 2014.
- Tzeng, Eric, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. "Deep domain confusion: Maximizing for domain invariance." *arXiv preprint arXiv:1412.3474* (2014).
- Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.
- Tzeng, Eric, et al. "Adversarial discriminative domain adaptation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.
- Liu, Ming-Yu, and Oncel Tuzel. "Coupled generative adversarial networks." In *Advances in neural information processing systems*, pp. 469-477. 2016.
- Tzeng, Eric, et al. "Simultaneous deep transfer across domains and tasks." *Proceedings of the IEEE International Conference on Computer Vision*. 2015.

Deep Transfer Learning (cont'd)

- Ganin, Yaroslav, et al. "Domain-adversarial training of neural networks." *The Journal of Machine Learning Research* 17.1 (2016): 2096-2030.
- Tzeng, Eric, et al. "Adversarial discriminative domain adaptation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.
- Ghifary, Muhammad, et al. "Deep reconstruction-classification networks for unsupervised domain adaptation." *European Conference on Computer Vision*. Springer, Cham, 2016.
- Bousmalis, Konstantinos, et al. "Domain separation networks." *Advances in neural information processing systems*. 2016.
- Sun, Baochen, Jiashi Feng, and Kate Saenko. "Return of frustratingly easy domain adaptation." Thirtieth AAAI Conference on Artificial Intelligence. 2016.
- Long, Mingsheng, et al. "Learning Transferable Features with Deep Adaptation Networks." International Conference on Machine Learning. 2015.
- Bousmalis, Konstantinos, et al. "Domain separation networks." Advances in neural information processing systems. 2016.

Instance Transfer

- Qu, Chen, et al. "Learning to selectively transfer: Reinforced transfer learning for deep text matching." *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 2019.

Deep Transfer Learning (cont'd)

- Mino, Ajkel, and Gerasimos Spanakis. "LoGAN: Generating logos with a generative adversarial neural network conditioned on color." *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018.
- Bo Wang, et al. 2019. A Minimax Game for Instance based Selective Transfer Learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19). Association for Computing Machinery, New York, NY, USA, 34–43.
- Ge, Weifeng, and Yizhou Yu. "Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1086-1095. 2017.
- Chowdhury, Somnath Basu Roy, K. M. Annervaz, and Ambedkar Dukkipati. "Instance-based Inductive Deep Transfer Learning by Cross-Dataset Querying with Locality Sensitive Hashing." *arXiv preprint arXiv:1802.05934* (2018).
- Qu, Chen, et al. "Learning to selectively transfer: Reinforced transfer learning for deep text matching." *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 2019.

Deep Transfer Learning For Search and Recommendation

- Prakash, Abhay, and Dhaval Patel. "Techniques for Deep Query Understanding." *arXiv preprint arXiv:1505.05187* (2015).
- Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1059–1068.
- deWet, Stephanie, and Jiafan Ou. "Finding Users Who Act Alike: Transfer Learning for Expanding Advertiser Audiences." *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019.
- Zhang, Shuai, et al. "Deep learning based recommender system: A survey and new perspectives." *ACM Computing Surveys (CSUR)* 52.1 (2019): 1-38.
- Peters, Matthew E., et al. "Deep contextualized word representations." *arXiv preprint arXiv:1802.05365* (2018).
- Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
- Oquab, Maxime, et al. "Learning and transferring mid-level image representations using convolutional neural networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
- Misra, Ishan, et al. "Cross-stitch networks for multi-task learning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

Deep Transfer Learning For Search and Recommendation (cont'd)

- Bansal, Trapit, David Belanger, and Andrew McCallum. "Ask the gru: Multi-task learning for deep text recommendations." Proceedings of the 10th ACM Conference on Recommender Systems. 2016.
- Zhang, Yongfeng, et al. "Joint representation learning for top-n recommendation with heterogeneous information sources." *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017.
- Cantador, Iván, et al. "Cross-domain recommender systems." *Recommender systems handbook*. Springer, Boston, MA, 2015. 919-959.
- Elkahky, Ali Mamdouh, Yang Song, and Xiaodong He. "A multi-view deep learning approach for cross domain user modeling in recommendation systems." *Proceedings of the 24th International Conference on World Wide Web*. 2015.
- Hu, Guangneng, Yu Zhang, and Qiang Yang. "Conet: Collaborative cross networks for cross-domain recommendation." *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 2018.
- Kanagawa, Heishiro, et al. "Cross-domain recommendation via deep domain adaptation." *European Conference on Information Retrieval*. Springer, Cham, 2019.
- Ahuja, Aman, et al. "Language-Agnostic Representation Learning for Product Search on E-Commerce Platforms." *Proceedings of the 13th International Conference on Web Search and Data Mining*. 2020.
- Bo Wang, Minghui Qiu, Xisen Wang, Yaliang Li, Yu Gong, Xiaoyi Zeng, Jun Huang, Bo Zheng, Deng Cai, and Jingren Zhou. A Minimax Game for Instance based Selective Transfer Learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. 2019.