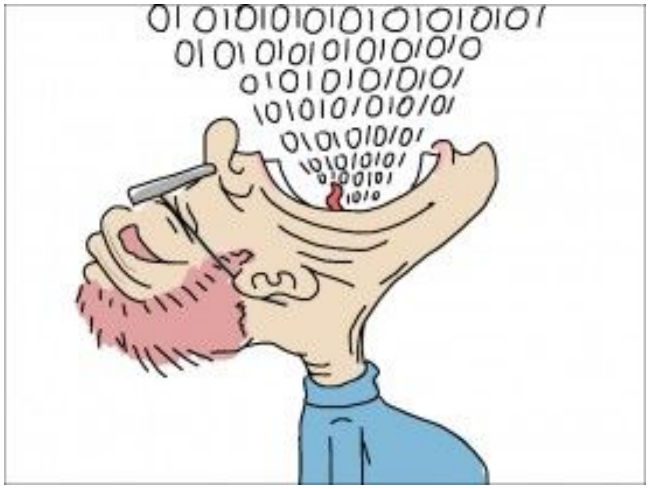
UNIVERSITÄT OSNABRÜCK

Transfer Learning

Leila Malihi

Why Transfer learning?

- Data hungry



- Training is very expensive

Why Transfer learning?

- BERT, developed by Google, has been trained on 16 Cloud TPUs (64 TPU chips total) for 4 days.

Why Transfer learning?

- The biggest problem is that models like BERT can only be done in a single job
- Future work requires a new set of data points

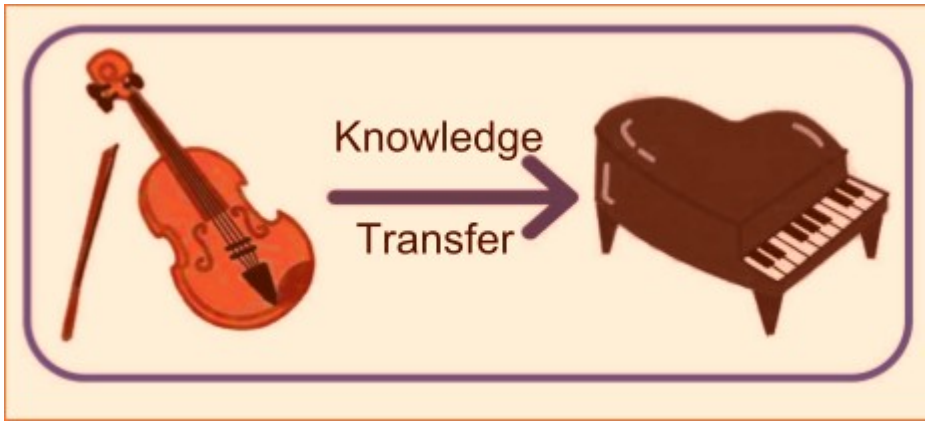
Why Transfer learning?

- If you know how to ride a motorbike, then you can learn how to drive a car
- If you know math and statistics, then you can learn machine learning
- If you know how to play classical piano, then you can learn how to play jazz piano

Transfer learning Or Traditional ML?

- Deep learning algorithms are designed to work in isolation

- Transfer learning is utilizing knowledge acquired for one task to solve related ones

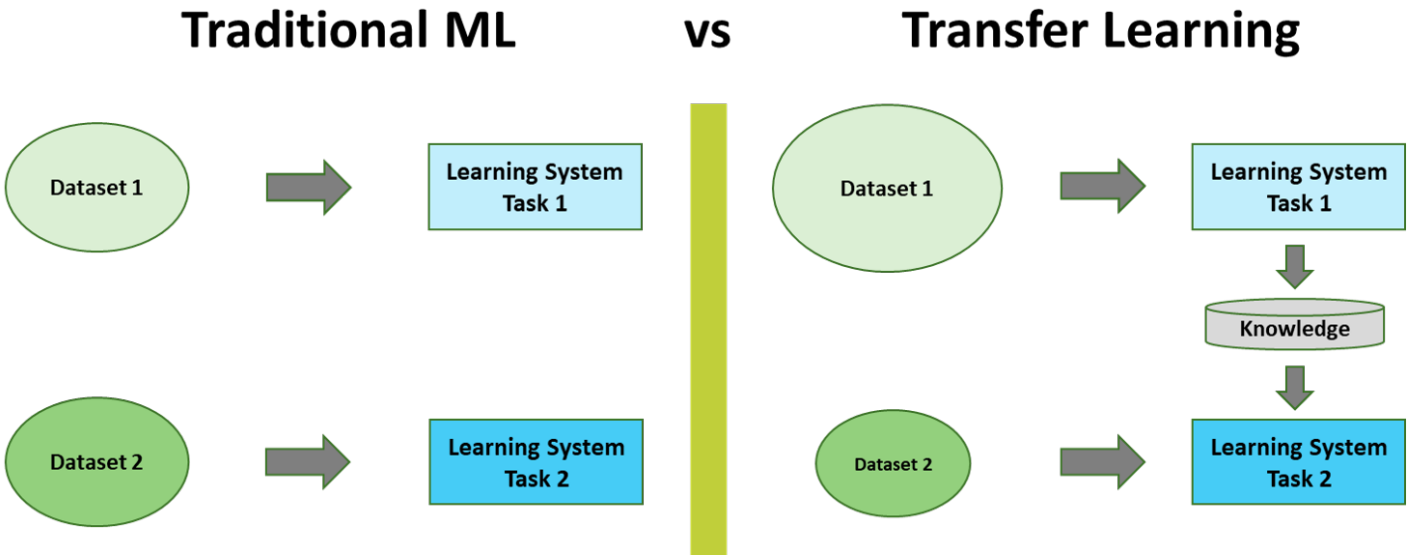


Fuzhen Zhuang, et al. A Comprehensive Survey on Transfer Learning

Negative Transfer learning

- If there is little in common between domains, knowledge transfer could be unsuccessful
- The similarities between domains do not always facilitate learning, because sometimes the similarities may be misleading
- For example, although Spanish and French have a close relationship with each other, but people who learn Spanish may experience difficulties in learning French.
- Previous experience has a negative effect on learning new tasks is called negative transfer

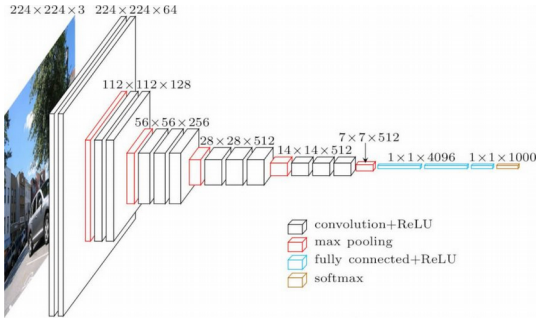
Transfer learning Or Traditional ML?



Examples of transfer learning

- Transfer learning in Computer-vision (image data)

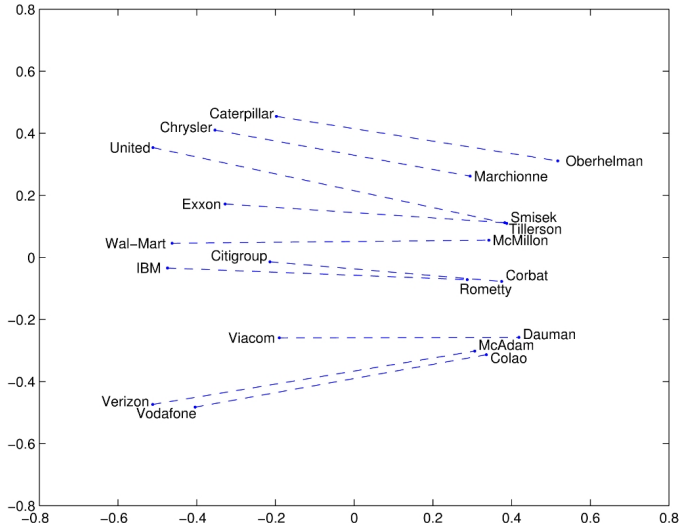
- Some of the models are:
  - Oxford VGG Model
  - Google Inception Model
  - Microsoft ResNet Model



Examples of transfer learning

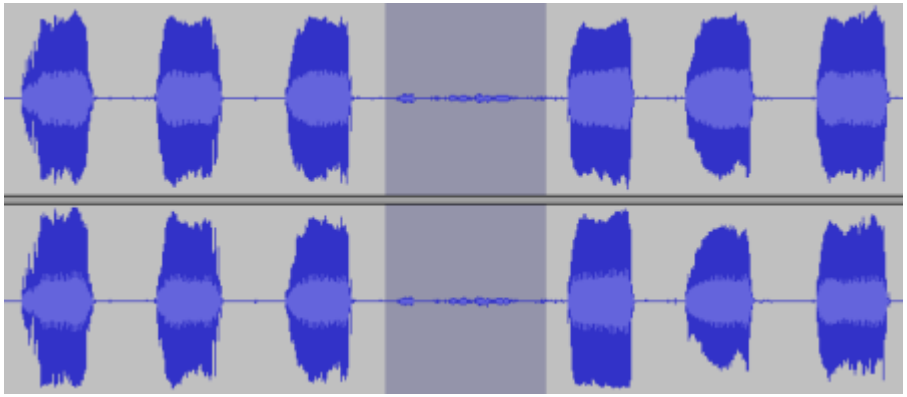
- Transfer learning in NLP (text data)

- Some of the pre-trained models are:
- Google's word2vec Model
- Stanford's GloVe Model



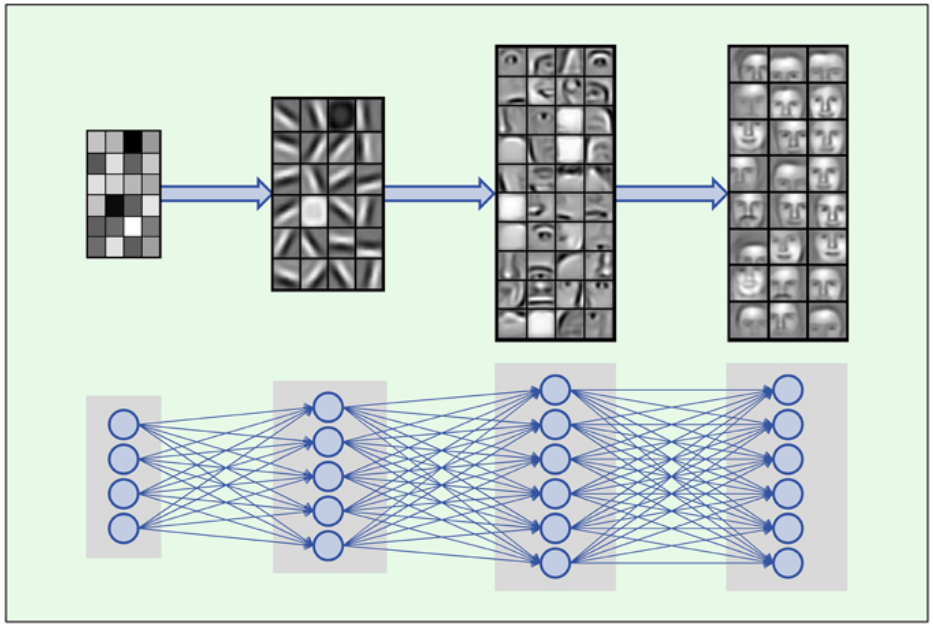
Examples of transfer learning

- Transfer learning in Audio/Speech



Why is transfer learning a better choice?

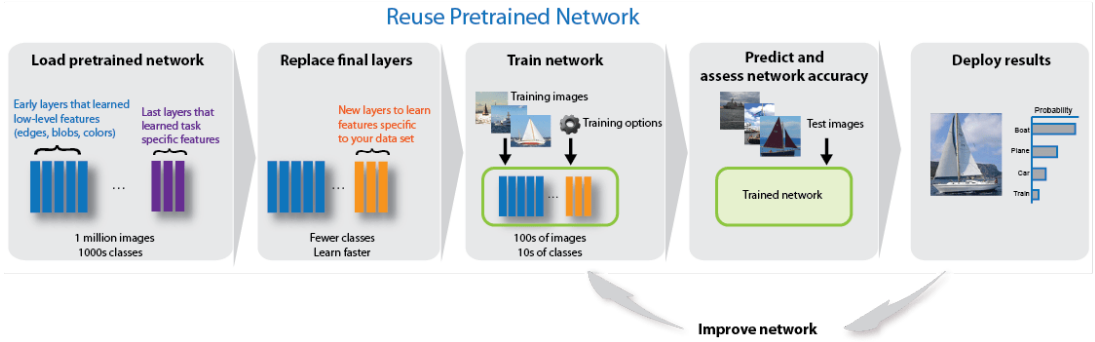
- To learn complex features and train complex model with poor dataset



Andres Mayer, et al. DeepTox: Toxicity Prediction Using Deep Learning

How to solve the problem

- Use some pre-trained networks
- Some of the pre-trained models are:
  - Alexnet
  - VGG19
  - VGG16
  - MobileNet
  - ResNet
  - Word2vec
  - Glove and many more…

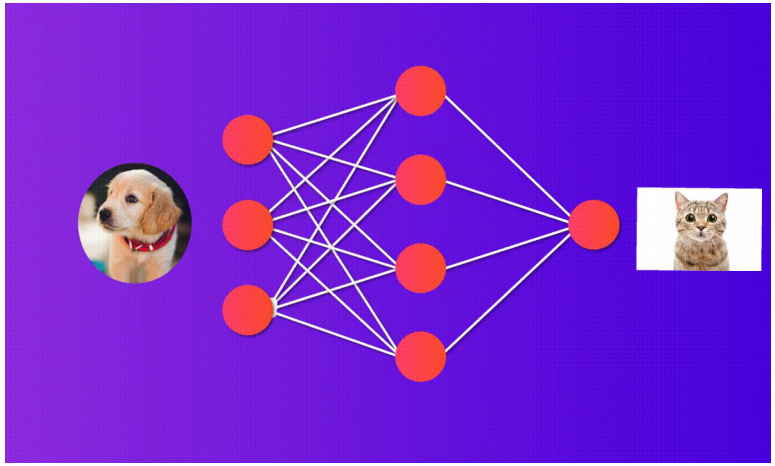


When to use transfer learning?

- **Lack of data**

- The tasks can be different but their domains should be the same

- We are unable to do transfer learning between speech recognition and image classification tasks since the input datasets are of different types



When to use transfer learning?

- Speed

- Social Good

Deep Transfer Learning Strategies

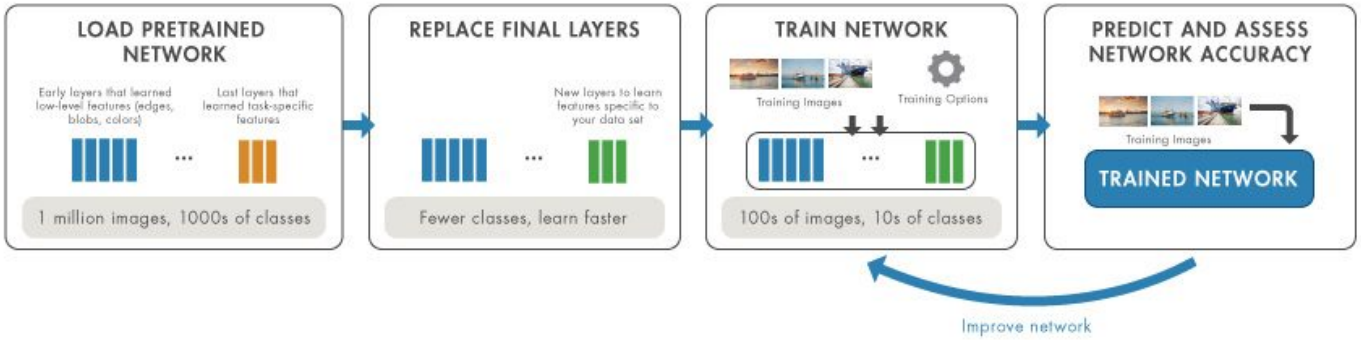
- Direct use of pre-trained models

- Some pre-trained models used directly :
  - BERT
  - YOLO (You Only Look Once),
  - GloVe, UnsupervisedMT

Deep Transfer Learning Strategies

➢Leveraging feature extraction from pre-trained models

- Treat the pre-trained neural network as a feature extractor by discarding the last fully-connected layer

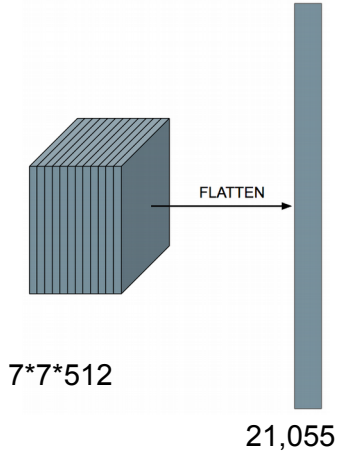


Deep Transfer Learning Strategies

➢**Leveraging feature extraction from pre-trained models**

It brings 2 main advantages:

- Allows for specifying the dimensions of the last fully-connected layer

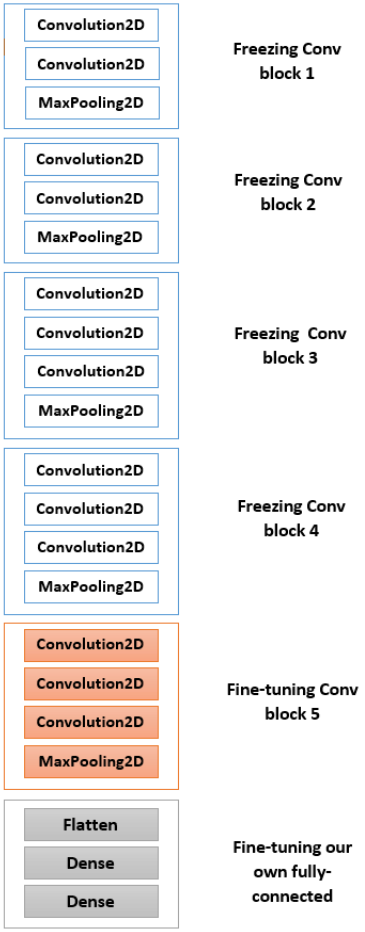


- Allows for using a lightweight linear model (e.g. Linear SVM, Logistic Regression).

Deep Transfer Learning Strategies

➢ **Fine-tuning last layers of pre-trained models**

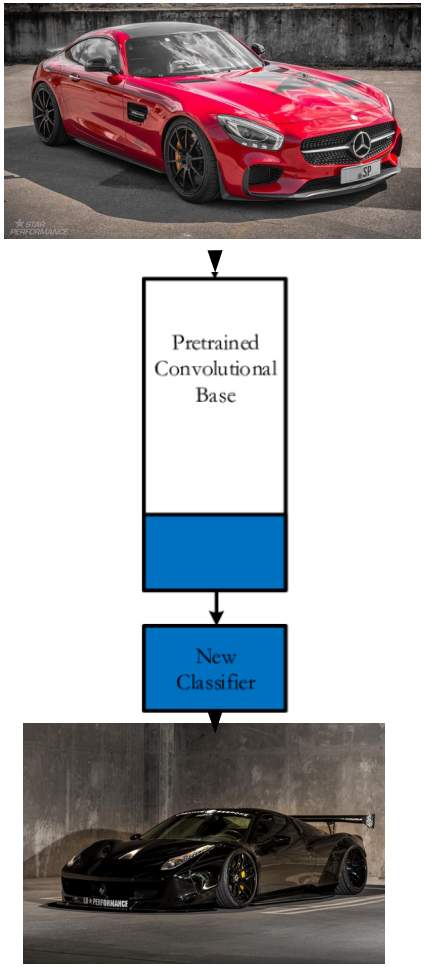
- Not only training the output classifier but also fine-tune weights in some layers of the pre-trained model



Deep Transfer Learning Strategies

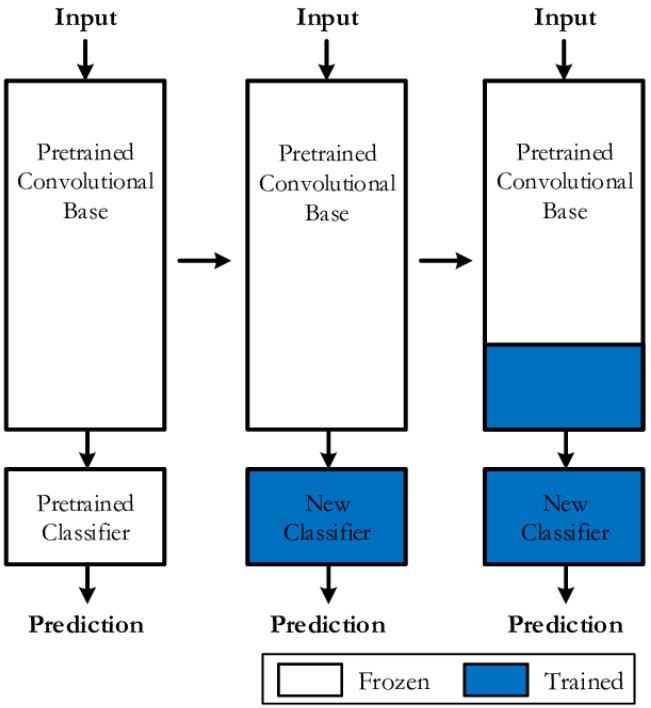
➢ **Fine-tuning last layers of pre-trained models**

- Example:
- Detecting Ferrari Car from Mercedes Car



Deep Transfer Learning Strategies

- **Comparing three strategies**



Confusing between related task

- Transfer learning
- Domain adaptation
- Multi-task learning
- One-shot learning
- Zero-shot learning

Confusing between related task

➢**Transfer learning**

- Target domain's feature space is different from the source feature space

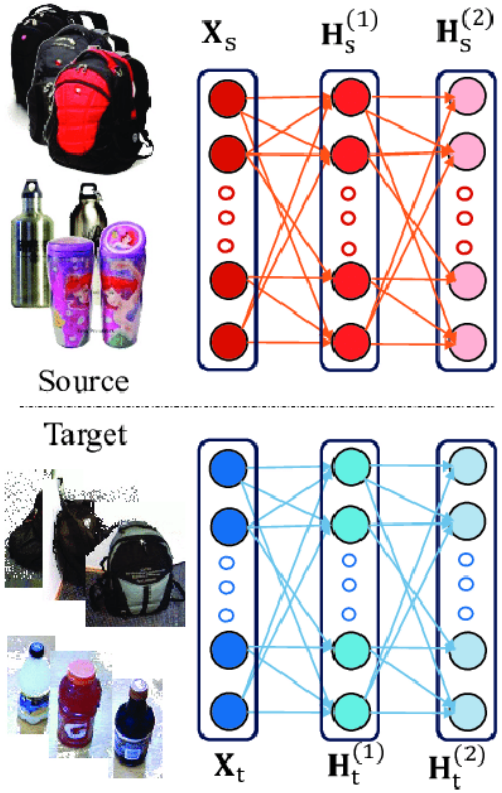
➢**Domain adaptation**

- The source and target domain fearture sapce are same but different distribution

Confusing between related task

➢Domain adaptation

- Example:Both target and source have same feature space but with different distribution

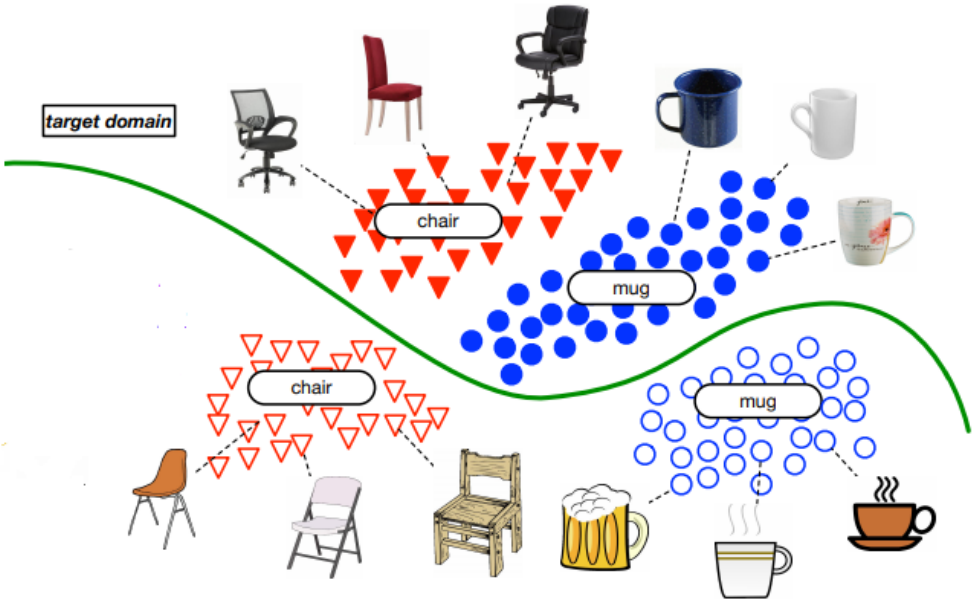


Zhengming Ding, et al., Semi-supervised Deep Domain Adaptation via Coupled Neural Networks, 2018.

Confusing between related task

➢Domain adaptation

- Example:Both target and source have same feature space but with different distribution



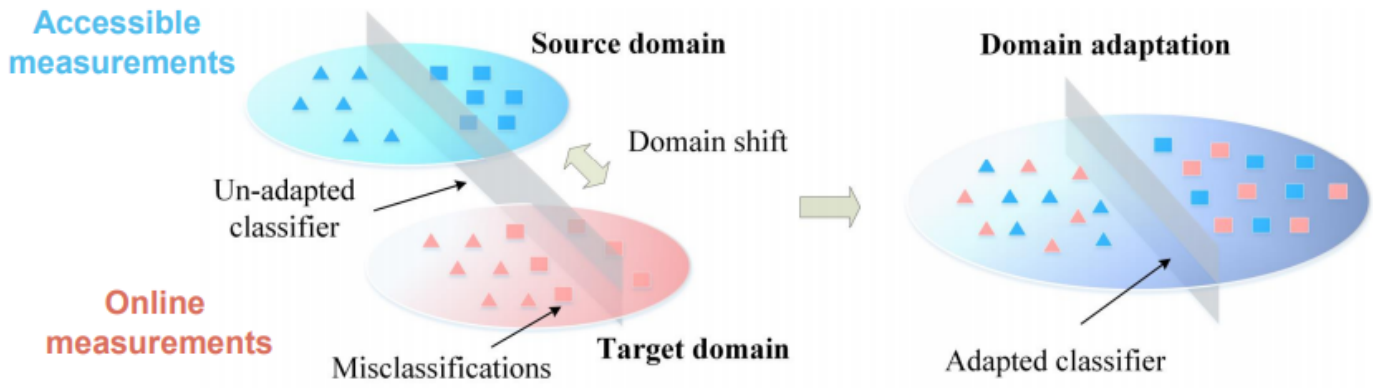
Zhangjie Cao, et al., Partial Adversarial Domain Adaptation, 2018.

Confusing between related task

➢Domain adaptation

- Two domains D^s and D^t are with different distribution if $P^s(X^s)$!= $P^t(X^t)$ (χ is the feature space, P(X) is the marginal probability distribution)

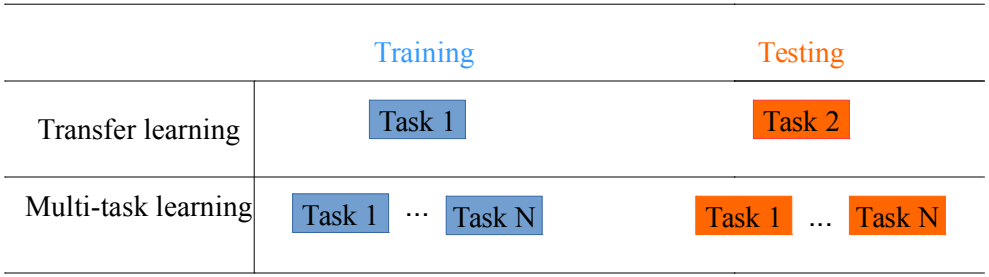
- We have domain adaptation when $\chi^s = \chi^t$ and $P^s(X^s)$!= $P^t(X^t)$
- Transfer learning: χ^s != χ^t



Confusing between related task

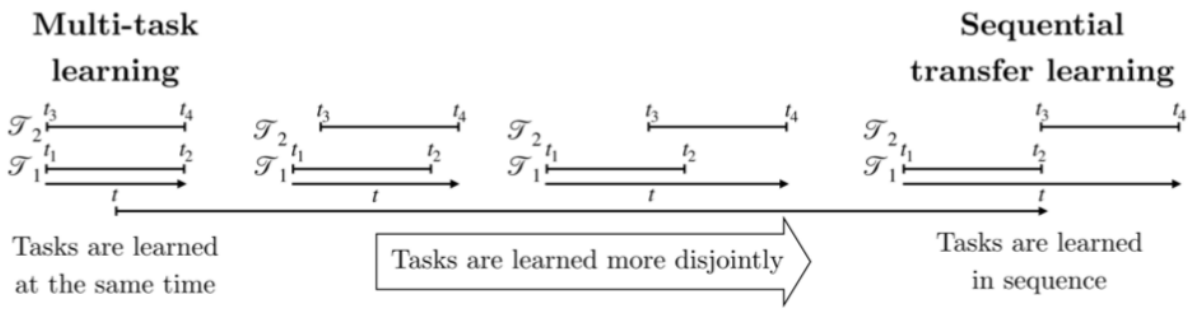
➢ Multi-task learning

•Transfer Learning only aims at achieving high performance in the target task by transferring knowledge from the source task, while Multi-task Learning tries to learn the target and the source task simultaneously.



Confusing between related task

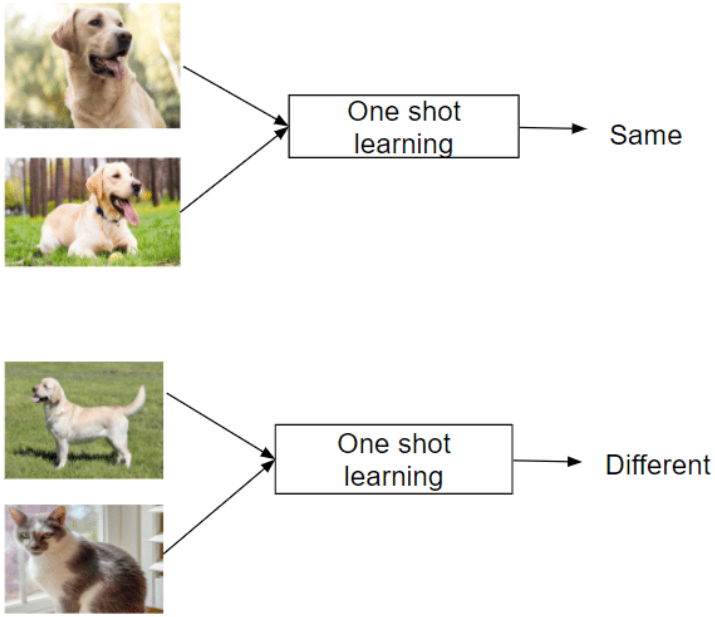
- **Multi-task learning**



Confusing between related task

➢One-shot learning

- one-shot learning aims to learn information about object categories from one, or only a few, training samples/images



Shivaank Agarwal, et al., Application of Computer Vision Techniques for Segregation of PlasticWaste based on Resin Identification Code.

Confusing between related task

➢Zero-shot learning

- Zero-shot learning is another extreme variant of transfer learning, which relies on no labeled examples to learn a task.
- ***Can you classify an object without ever seeing it?***

Shivaank Agarwal, et al., Application of Computer Vision Techniques for Segregation of PlasticWaste based on Resin Identification Code.

Question?

Implement a simple transfer learning algorithm

- **Choose the best pre-tranied model**
- **First see the dataset:**
- Image-Net dataset

High level category	# synset (subcategories)	Avg # images per synset	Total # images
amphibian	94	591	56K
animal	3822	732	2799K
appliance	51	1164	59K
bird	856	949	812K
covering	946	819	774K
device	2385	675	1610K
fabric	262	690	181K
fish	566	494	280K
flower	462	735	339K
food	1495	670	1001K
fruit	309	607	188K
fungus	303	453	137K
furniture	187	1043	195K
geological formation	151	838	127K
invertebrate	728	573	417K
mammal	1138	821	934K
musical instrument	157	891	140K
plant	1666	600	999K
reptile	268	707	190K
sport	166	1207	200K
structure	1239	763	946K
tool	316	551	174K
tree	993	568	564K
utensil	86	912	78K
vegetable	176	764	135K
vehicle	481	778	374K
person	2035	468	952K

- VGGFace2 dataset:
- Number Of Images= 3.3 million
- Number Of subjects=9,131
- VGGFace dataset:

Number Of Images= 2.6 million

Number Of subjects=2,622

Implement a simple transfer learning algorithm

- **Choose the best pre-tranied model**
  - **Find the model:**
  - https://keras.io/api/applications/

Available models

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-
EfficientNetB0	29 MB	-	-	5,330,571	-
EfficientNetB1	31 MB	-	-	7,856,239	-
EfficientNetB2	36 MB	-	-	9,177,569	-
EfficientNetB3	48 MB	-	-	12,320,535	-
EfficientNetB4	75 MB	-	-	19,466,823	-
EfficientNetB5	118 MB	-	-	30,562,527	-
EfficientNetB6	166 MB	-	-	43,265,143	-
EfficientNetB7	256 MB	-	-	66,658,687	-

Implement a simple transfer learning algorithm

➢Face recognition using transfer learning

- **Select VGGFace or VGGFace 2 model**

- The VGGFace model, named later, was described by Omkar Parkhi in the 2015 paper titled “Deep Face Recognition.”

- The VGGFace2 model,was described by Qiong Cao,in the 2017 paper “VGGFace2: A dataset for recognizing faces across pose and age.”

Implement a simple transfer learning algorithm

➢**Face recognition using transfer learning**

•**VGGFace Models:**

✔vggface = VGGFace(model='vgg16')

✔vggface = VGGFace(model='resnet50')

✔vggface = VGGFace(model='senet50')

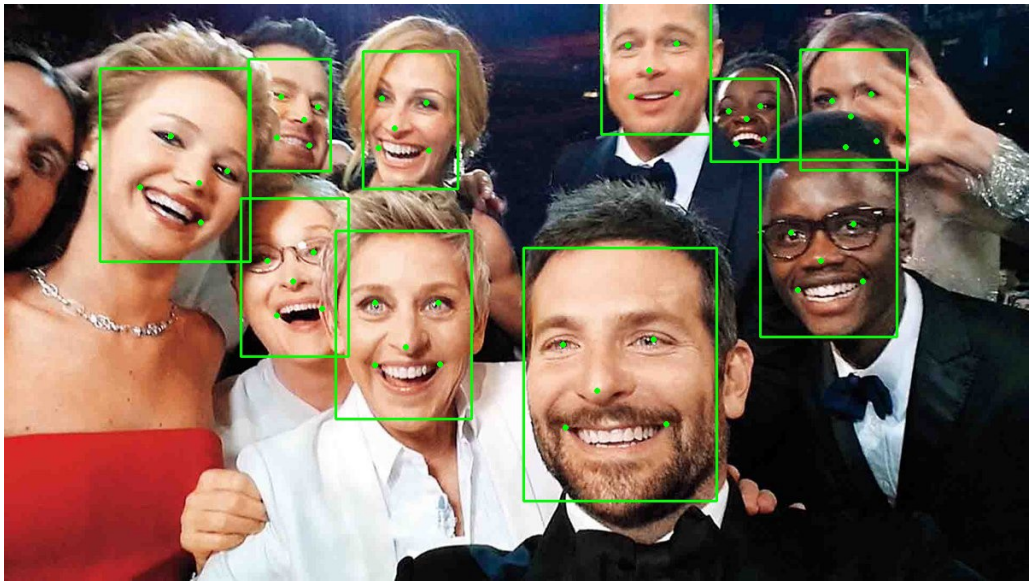
•Perhaps the best-of-breed third-party library for using the VGGFace2 (and VGGFace) models in Keras is the keras-vggface project and library by Refik Can Malli

Implement a simple transfer learning algorithm

➢How to Detect Faces for Face Recognition?

- Use the Multi-Task Cascaded Convolutional Neural Network, or MTCNN, for face detection

- This is a state-of-the-art deep learning model for face detection, described in the 2016 paper titled "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks."



Implement a simple transfer learning algorithm

➢ **Transfer-Learning**

- Allow the CNN network parameters to be transferred from a large datasets to small one

- Starting point to learn a new task

- Transfer learned features to a new task using a smaller number of training images

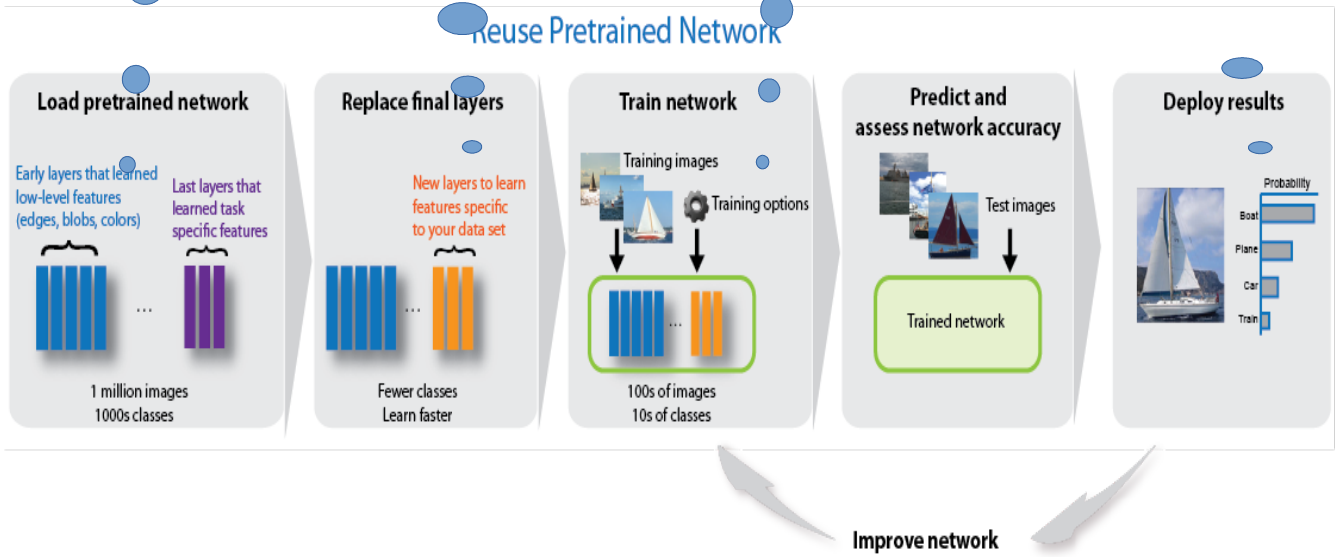
Implement a simple transfer learning algorithm

Closed to our dataset

1- Number of neurons
2- Number of fully-connected layers
3- loss function

Tuning the learning rate

Number of last neurons= number of classes



How to determine number of Layers in fully-connected layer?

- why we would want to have multiple layers?

•A single-layer neural network can only be used to represent **linearly** separable functions.

•If your problem is relatively simple, perhaps a single layer network would be sufficient.

•A Multilayer Perceptron can be used to represent convex regions.

•They can learn to draw shapes around examples in some high-dimensional space

• that can separate and classify them.

How do we choose a learning rate?

➢A naive approach is to try a few different values and see which one gives you the best loss without sacrificing speed of training.

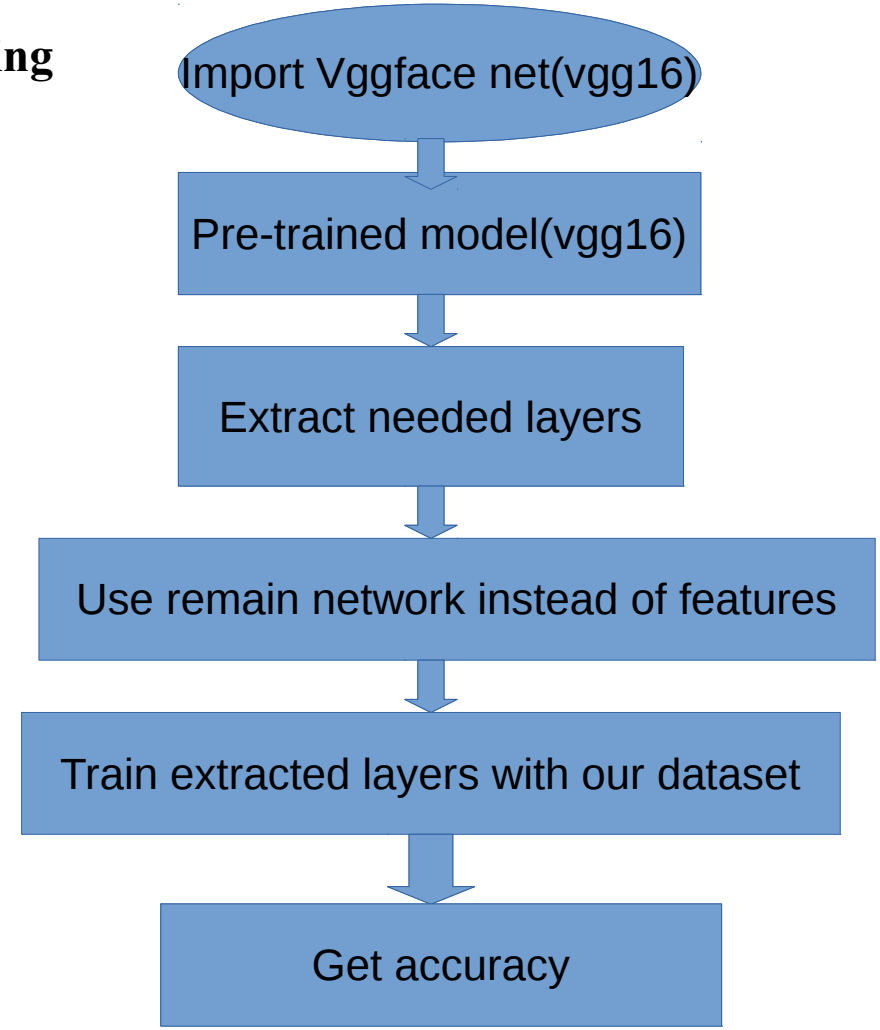
•We might start with a large value like 0.1, then try exponentially lower values: 0.01, 0.001, etc.

•**What happens if the learning rate is too high?**

A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution,whereas a learning rate that is too small can cause the process to get stuck.

Implement a simple transfer learning algorithm

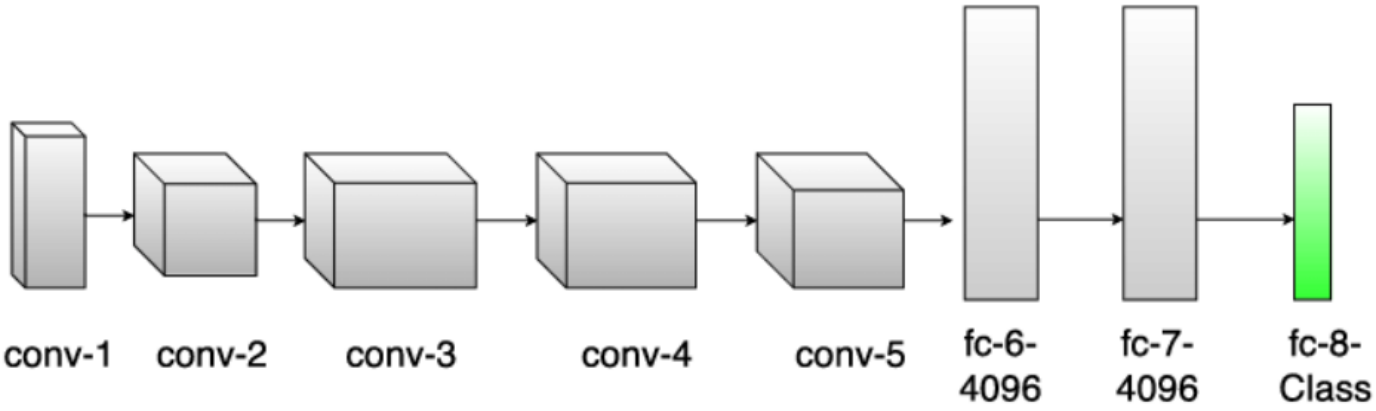
- **Face recognition using transfer learning**



Implement a simple transfer learning algorithm

➢ **Face recognition using transfer learning**

- Retrain the last fully-connected layers



Question?

A New article

➢DT-LET: Deep Transfer Learning by Exploring where to Transfer

- How to transfer knowledge ?

✔the number of source and target domain should be same

- The problem appears when the data from the two domains are heterogeneous with different resolutions

- Solution:“where to transfer” proposed

A New article

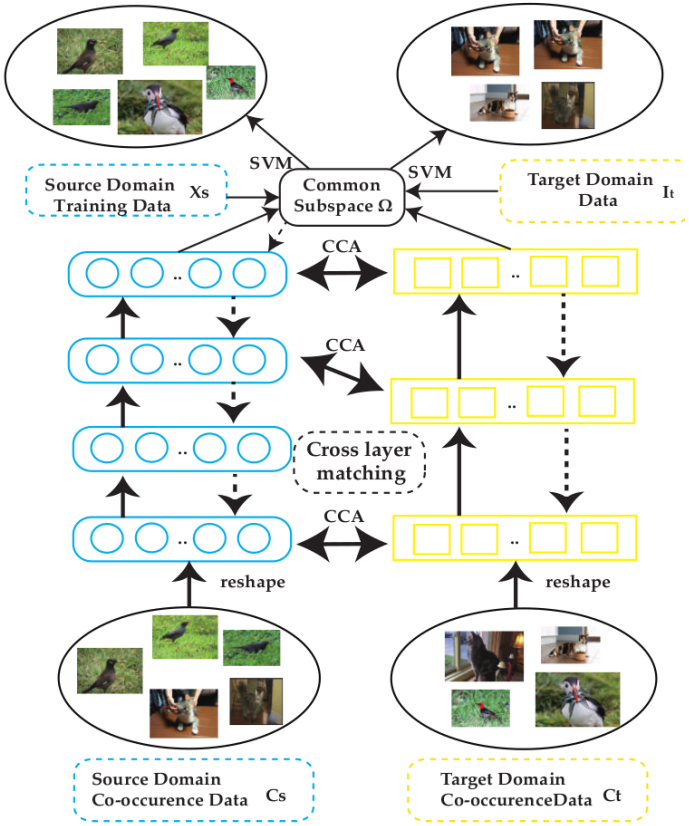
➢DT-LET: Deep Transfer Learning by Exploring where to Transfer

- The number of layers for two domains does not need to be the same

- Optimal matching of layers will be found

A New article

- DT-LET: Deep Transfer Learning by Exploring where to Transfer




Thank
you!!!