Transfer Learning

Dr Haider Raza
Lecturer in AI for Decision-Making,
School of Computer Science and Electronics Engineering,
University of Essex, UK.

About

- We will be discussing what is transfer learning
  - why we need transfer learning?
- Different types of transfer learning
  - Which type of learning model to use when?
- We will see transfer learning using keras and tensorflow
  - For example we will use pre-trained model VGG-16 and use it in our dataset

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or $P(Y \cap X)$ (i.e. Probability of Y and X)
4. $(X \rightarrow Y)$: Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. $(Y \rightarrow X)$: Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - $P(Y|X)P(X)$ in $X \rightarrow Y$ problems
   - $P(X|Y)P(Y)$ in $Y \rightarrow X$ problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or $P(Y \cap X)$ (i.e. Probability of Y and X)
4. ($X \rightarrow Y$): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. ($Y \rightarrow X$): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - $P(Y|X)P(X)$ in $X \rightarrow Y$ problems
   - $P(X|Y)P(Y)$ in $Y \rightarrow X$ problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y $\cap$ X) (i.e. Probability of Y and X)
4. (X $\rightarrow$ Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y $\rightarrow$ X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X $\rightarrow$ Y problems
   - P(X|Y)P(Y) in Y $\rightarrow$ X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y ∩ X) (i.e. Probability of Y and X)
4. (X → Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y → X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X → Y problems
   - P(X|Y)P(Y) in Y → X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y $\cap$ X) (i.e. Probability of Y and X)
4. (X $\rightarrow$ Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y $\rightarrow$ X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X $\rightarrow$ Y problems
   - P(X|Y)P(Y) in Y $\rightarrow$ X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y $\cap$ X) (i.e. Probability of Y and X)
4. (X $\rightarrow$ Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y $\rightarrow$ X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X $\rightarrow$ Y problems
   - P(X|Y)P(Y) in Y $\rightarrow$ X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y $\cap$ X) (i.e. Probability of Y and X)
4. (X $\rightarrow$ Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y $\rightarrow$ X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X $\rightarrow$ Y problems
   - P(X|Y)P(Y) in Y $\rightarrow$ X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

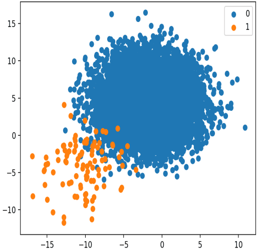
Notation

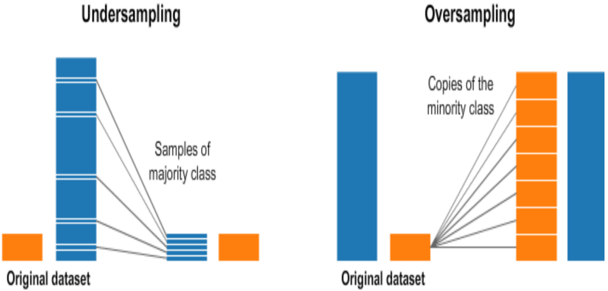
1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or P(Y $\cap$ X) (i.e. Probability of Y and X)
4. (X $\rightarrow$ Y): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. (Y $\rightarrow$ X): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - P(Y|X)P(X) in X $\rightarrow$ Y problems
   - P(X|Y)P(Y) in Y $\rightarrow$ X problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Notation

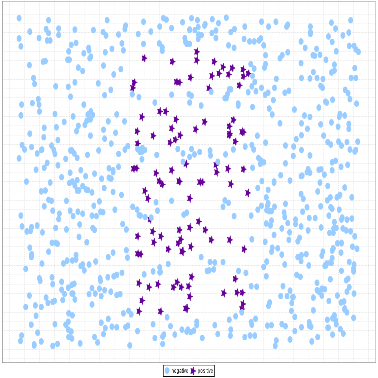
1. A set of features or covariates X
2. A set of target or class variables Y
3. A joint distribution P(Y,X) or $P(Y \cap X)$ (i.e. Probability of Y and X)
4. ($X \rightarrow Y$): Y is determined by values of X (e.g. credit card fraud detection) Predictive models (e.g. Logistic Regression, SVM, and Neural Networks)
5. ($Y \rightarrow X$): Y determines the values of X (e.g. medical diagnosis) Generative models (e.g. GMM, HMM, and Naive Bayes)
6. The joint distribution P(Y,X) can be written as:
   - $P(Y|X)P(X)$ in $X \rightarrow Y$ problems
   - $P(X|Y)P(Y)$ in $Y \rightarrow X$ problems
7. P_{tr}: Data distribution in training
8. P_{ts}: Data distribution in testing

Why difficult to learn DATA?: Imbalanced

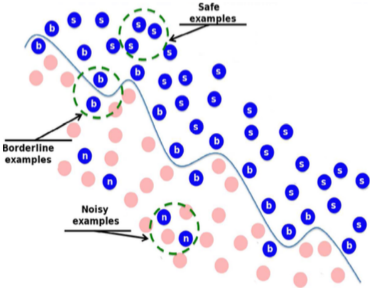




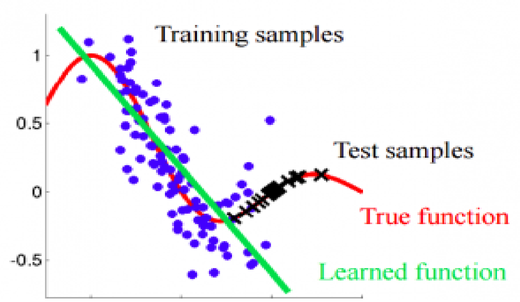
Why difficult to learn DATA?: Overlapping



Why difficult to learn DATA?: Noise



Why difficult to learn DATA?: Dataset Shift



Motivation

- In learning theory independent and identically distributed (i.i.d) assumption (i.e. each random variable has the same probability distribution as the others and all are mutually independent)
- In practice train and test inputs have different distributions
- The difference in distribution arises from operating in non-stationary environments in real-world application such as finance, healthcare, brain signals, much more… }
- Learning in such non-stationary environment is difficult and we need an think before operating

Motivation

- In learning theory independent and identically distributed (i.i.d) assumption (i.e. each random variable has the same probability distribution as the others and all are mutually independent)
- In practice train and test inputs have different distributions
- The difference in distribution arises from operating in non-stationary environments in real-world application such as finance, healthcare, brain signals, much more... }
- Learning in such non-stationary environment is difficult and we need an think before operating

Motivation

- In learning theory independent and identically distributed (i.i.d) assumption (i.e. each random variable has the same probability distribution as the others and all are mutually independent)
- In practice train and test inputs have different distributions
- The difference in distribution arises from operating in non-stationary environments in real-world application such as finance, healthcare, brain signals, much more... }
- Learning in such non-stationary environment is difficult and we need an think before operating

Motivation

- In learning theory independent and identically distributed (i.i.d) assumption (i.e. each random variable has the same probability distribution as the others and all are mutually independent)
- In practice train and test inputs have different distributions
- The difference in distribution arises from operating in non-stationary environments in real-world application such as finance, healthcare, brain signals, much more. . . }
- Learning in such non-stationary environment is difficult and we need an think before operating

Dataset Shift

Cases where the joint distribution of inputs and outputs differs between training and test stage [1]

- concept shift/drift G. Widmer et al., 1996, 1998
- changes of classification K. Wang et al., 2003
- changing environments R. Alaiz-Rodrıguez et al., 2008
- fracture point N.V. Chawla et al., 2009
- fractures between data J.G. Moreno-Torres et al., 2010

[1] A. Storkey, Dataset Shift in Machine Learning, 2009

Dataset Shift

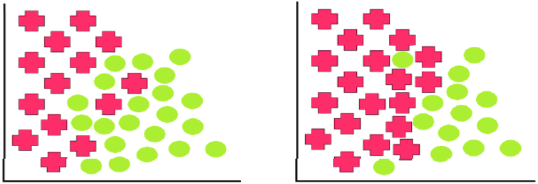
Cases where the joint distribution of inputs and outputs differs between training and test stage [1]

- concept shift/drift G. Widmer et al., 1996, 1998
- changes of classification K. Wang et al., 2003
- changing environments R. Alaiz-Rodrıguez et al., 2008
- fracture point N.V. Chawla et al., 2009
- fractures between data J.G. Moreno-Torres et al., 2010

[1] A. Storkey, Dataset Shift in Machine Learning, 2009

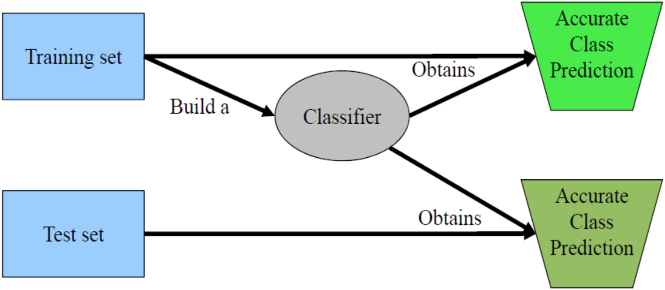
Dataset Shift...

Dataset shift appears when training and test joint distributions are different. That is, when $P_{tr}(X,Y) \neq P_{ts}(X,Y)$



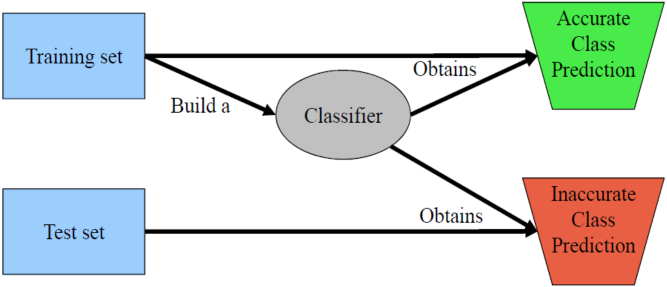
Dataset Shift...

Basic assumption for classification in operating under stationary environment



Dataset Shift...

But sometimes...



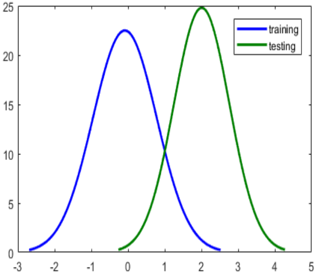
Types of dataset shift

1. Covariate shift
2. Prior probability shift
3. Concept Shift

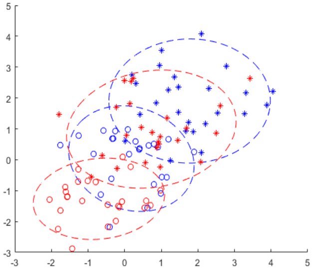
Covariate Shift

Covariate shift appears only in X → Y problems, and is defined as the case where, $P_{tr}(Y \mid X) = P_{ts}(Y \mid X)$ and $P_{tr}(X) \neq P_{ts}(X)$

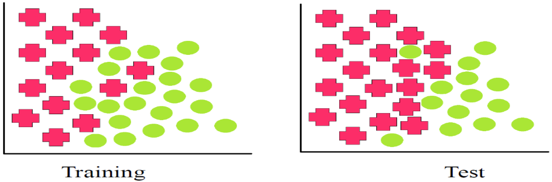
Univariate



Bivariate



Prior probability shift

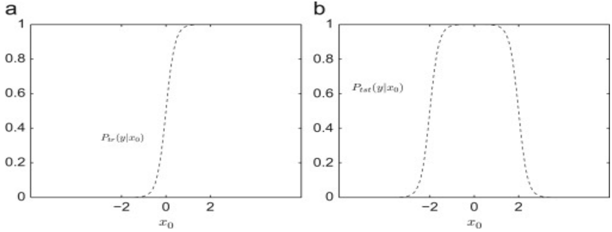


Prior probability shift appears only in Y → X problems, and is defined as the case where, $P_{tr}(Y \mid X) = P_{ts}(Y \mid X)$ & $P_{tr}(Y) \neq P_{ts}(Y)$

Concept Shift

X → Y problems: $P_{tr}(Y \mid X) \neq P_{ts}(Y \mid X)$ and $P_{tr}(X) = P_{ts}(X)$

Y → X problems: $P_{tr}(X \mid Y) \neq P_{ts}(X \mid Y)$ and $P_{tr}(Y) = P_{ts}(Y)$



Causes of Dataset Shift. . .

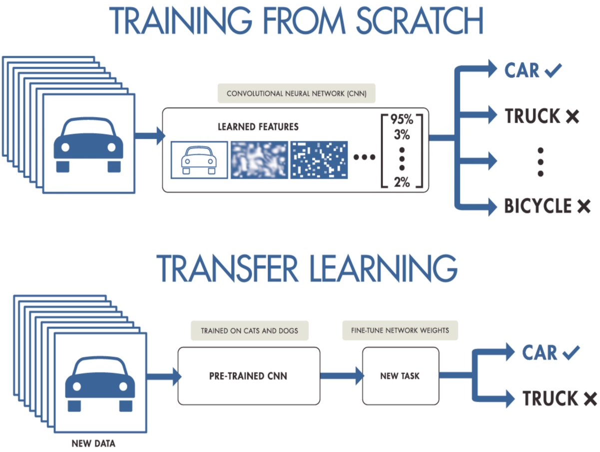
1. Sample selection bias: the discrepancy in distribution is due to the fact that the training examples have been obtained through a biased method, and thus do not represent reliably the operating environment where the classiffier is to be deployed (In ML terms, would constitute the test set)
2. Non-stationary environments: It appears when the training environment is different from the test one, whether it is due to a temporal or a spatial change

Learning in Non-stationary environments

Mind Map of NSE [2]

[2] Learning in Nonstationary Environments : A Survey. *IEEE Computational Intelligence Magazine*, 10(4), 12–25

Transfer Learning



Transfer Learning

Transfer learning is a kind of learning method, where a model developed for a task is reused as the starting point for a model on a second but related task

- Example: Knowledge gained while learning to recognize cars could apply when trying to recognize buses

Why Transfer Learning?

- In practice, people train a CNN from scratch (random initialisation) because it is rare to get enough dataset
- Very Deep Networks are expensive to train (take weeks to train using hundreds of machines equipped with expensive GPUs)

Transfer Learning...

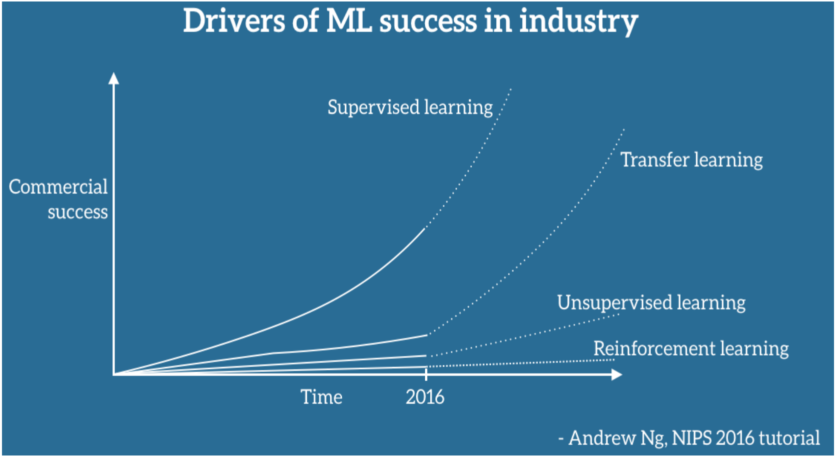
- "Transfer learning and domain adaptation refer to the situation where what has been learned in one setting is exploited to improve generalization in another setting", I. Goodfellow., Y. Bengio., A. Courville., and F. Bach., Deep Learning, 2016
- "Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned" Chapter 11: Transfer Learning, Handbook of Research on Machine Learning Applications, 2009

Transfer Learning in ML industry

Andrew Ng, chief scientist at Baidu and professor at Stanford, said during his widely popular NIPS 2016 – after supervised learning – Transfer Learning is the next driver of ML commercial success

– Andrew Ng 

Driver of ML Success in Industry

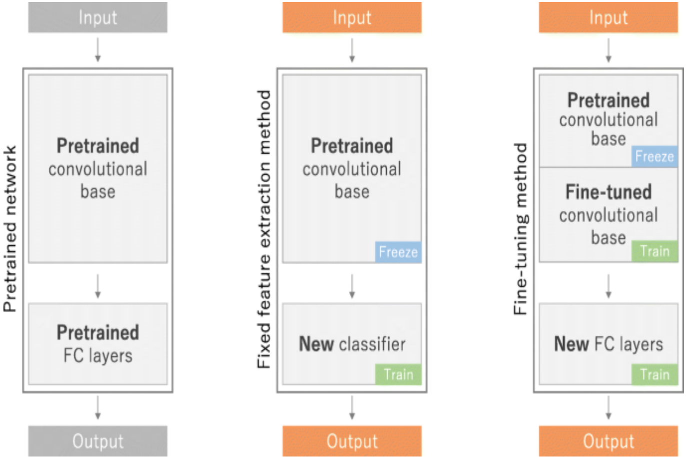


Ways of using Transfer Learning

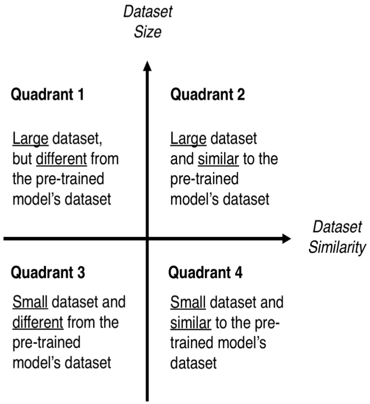
Transfer learning can be used in different scenarios

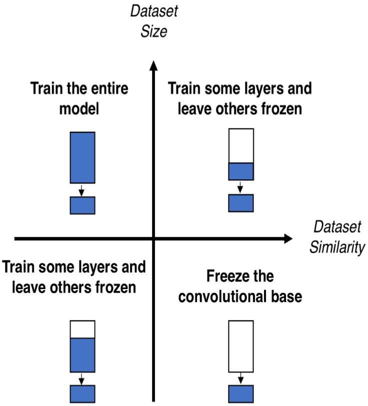
- ConvNet as fixed feature extracture
- Fine-tuning ConvNet
- Pretrained models

Ways of using Transfer Learning



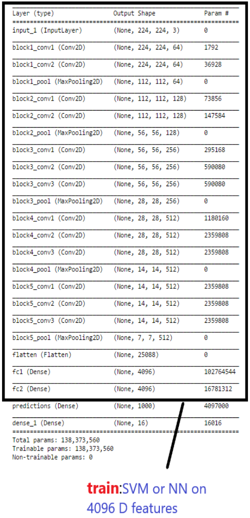
Decide your Transfer Learning Case:





ConvNet as fixed feature extracture

- Take a ConvNet pretrained on ImageNet, remove the last fully-connected layer (this layer's outputs are the 1000 class scores for a different task like ImageNet), then treat the rest of the ConvNet as a fixed feature extractor for the new dataset
- Once you extract the 4096-D codes for all images, train a linear classifier (e.g. Linear SVM or Softmax classifier) for the new dataset
- Better to use this with small datasets

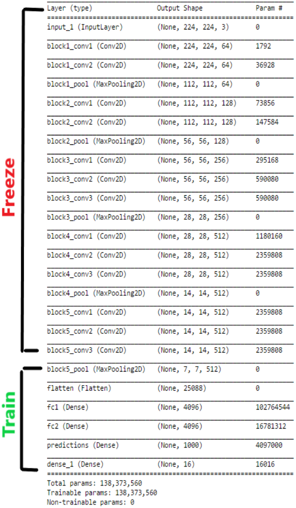


Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000
dense_1 (Dense)	(None, 16)	16016

Total params: 138,373,560
Trainable params: 138,373,560
Non-trainable params: 0

Fine-tuning ConvNet

- Not only replace and retrain the classifier of ConvNet on the new dataset, but also fine-tune the weights of the pretrained network.
- Fine-tune all the layers of the ConvNet, or keep some of the earlier layers fixed (overfitting concerns) and only fine-tune some higher-level portion of the network.
- Earlier features of a ConvNet contain more generic features (e.g. edge detectors) and useful to many tasks, but later layers of the ConvNet becomes progressively more specific to the details of the classes contained in the original dataset
- Better to use this with medium or large sized datasets



Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000
dense_1 (Dense)	(None, 16)	16016

Total params: 138,373,560
Trainable params: 138,373,560
Non-trainable params: 0

Pretrained models

- Since modern ConvNets take 2-3 weeks to train across multiple GPUs on ImageNet, it is common to see people release their final ConvNet checkpoints for the benefit of others who can use the networks for fine-tuning. For example, the Caffe library has a Model Zoo where people share their network weights

Conclusion

- We have seen dataset shift and types of dataset shift
- Never, start learning without understanding your data and its properties
- Transfer Learning is the next driver of ML commercial success