Introduction to Transfer Learning

Thien Tran
tran0096@e.ntu.edu.sg

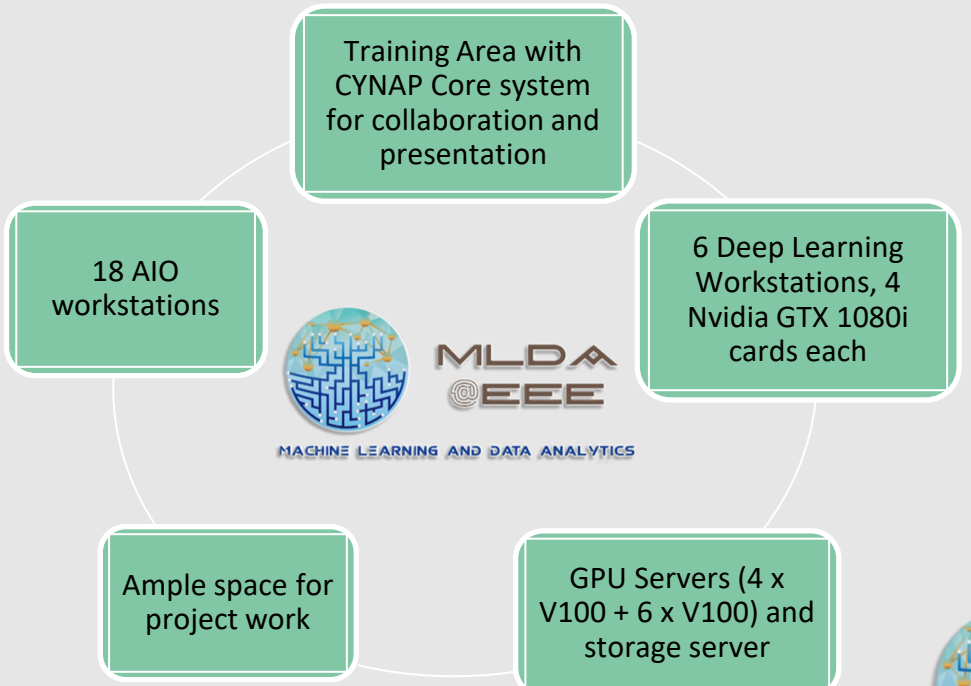
MLDA@EEE

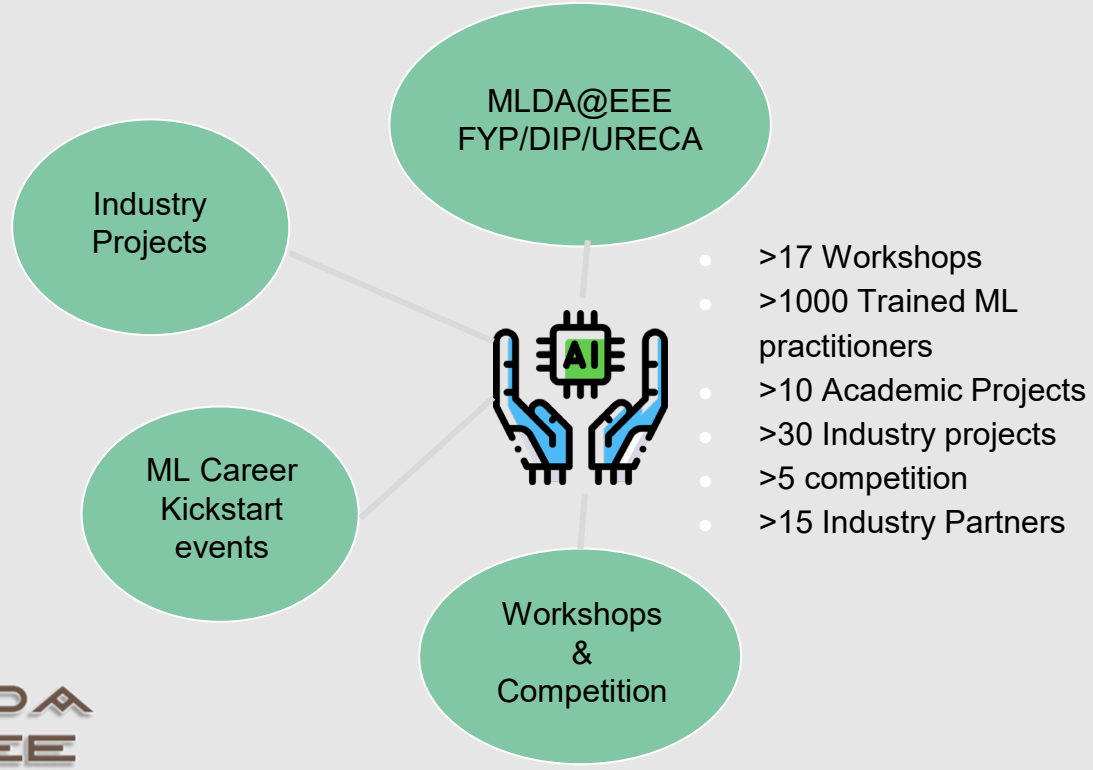
MLDA @EEE

MACHINE LEARNING AND DATA ANALYTICS

Our Mission

Provide an integrated platform for EEE/IEM students to learn and implement Machine Learning, Data Science & AI, as well as facilitate connections with the industry.



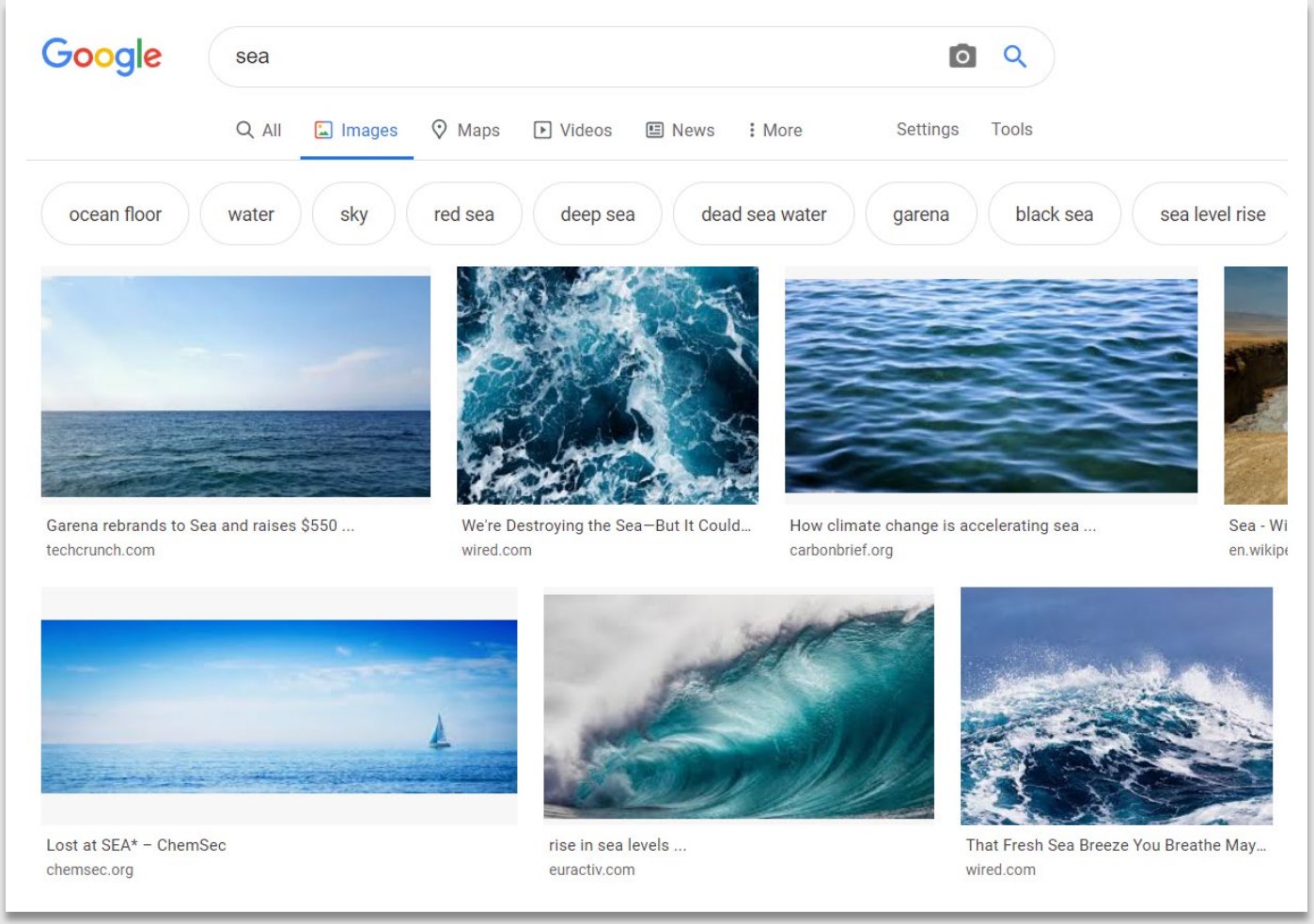


MLDA @EEE
MACHINE LEARNING AND DATA ANALYTICS

Agenda

- Theory of Transfer learning
- Transfer Learning workflow
- Hands-on: implement Transfer Learning with TensorFlow Keras

Image auto-tagging



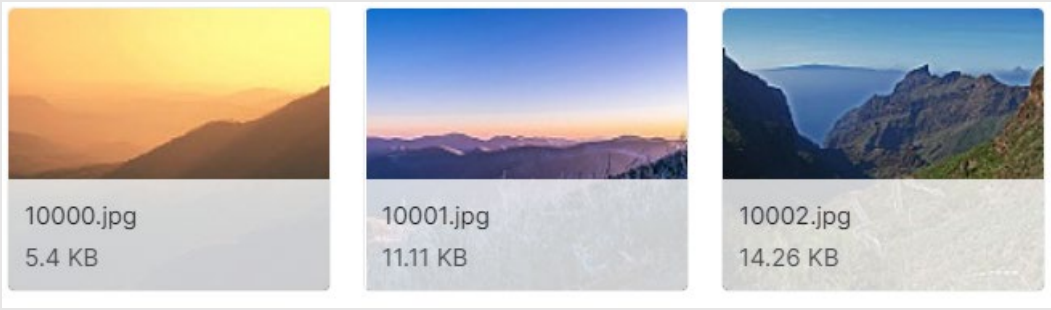
You have a collection of a lot of images

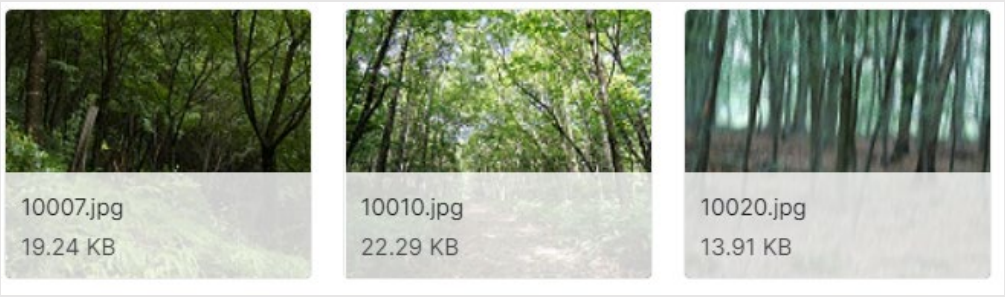
You want to view your **'sea'** images
And potentially all other types of landscape images also

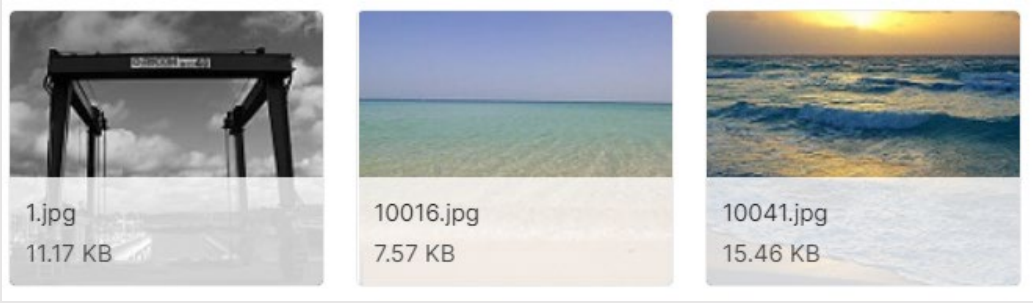
You can label some of them manually but cannot label the whole collection

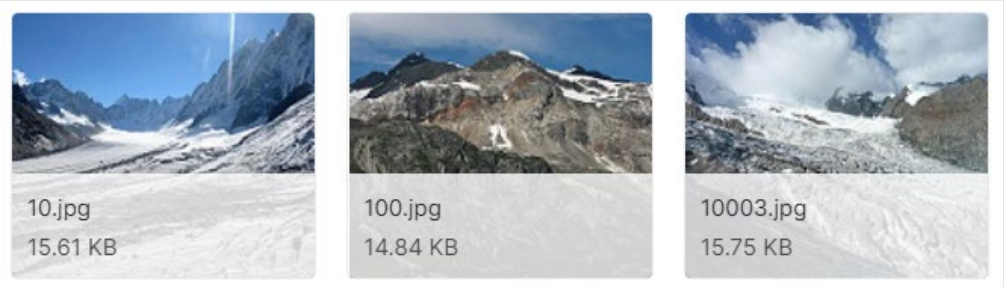
MACHINE LEARNING TASK	CONSTRAINTS
Image classification	Small, limited dataset

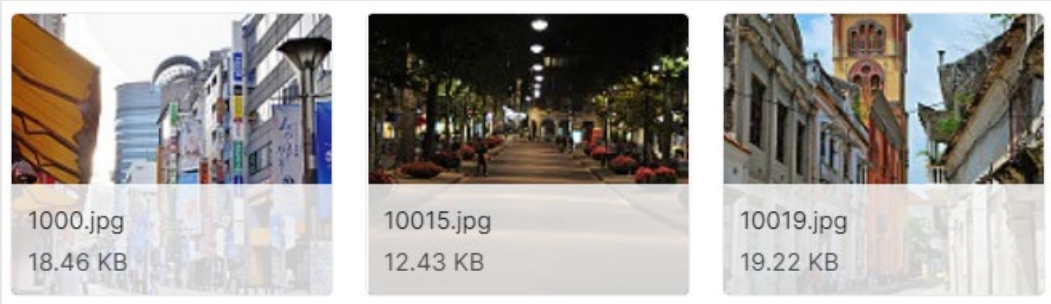
Classify landscape images











Intel Image Classification
https://www.kaggle.com/puneet6060/intel-image-classification

What is Transfer Learning

- Apply knowledge learned from **one task** to another **related task**

LAKE CLASSIFIER

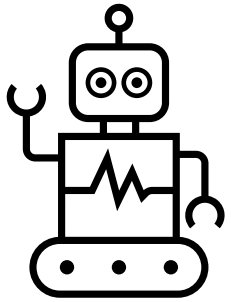
NTU Yunnan Garden
https://media.ntu.edu.sg/NewsReleases/Pages/newsdetail.aspx?news=a9f88bb3-c20b-4dea-a1e2-7eb65a4c8d9b

SEA CLASSIFIER

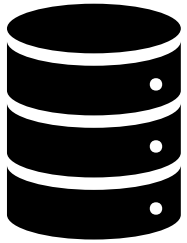
'sea' from Intel Image Classification
https://www.kaggle.com/puneet6060/intel-image-classification

What is Transfer Learning

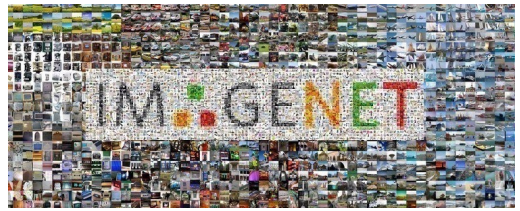
General model
Obtained from the Internet



Training

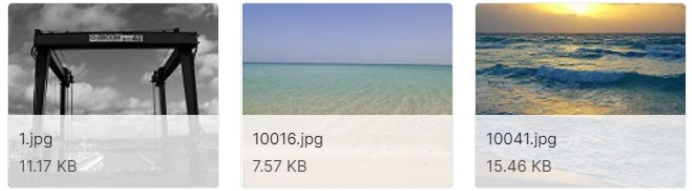


Large, general-purpose data

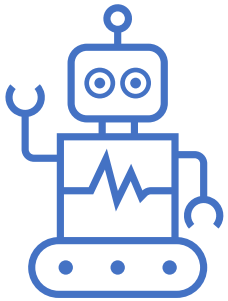


Fine-tuning

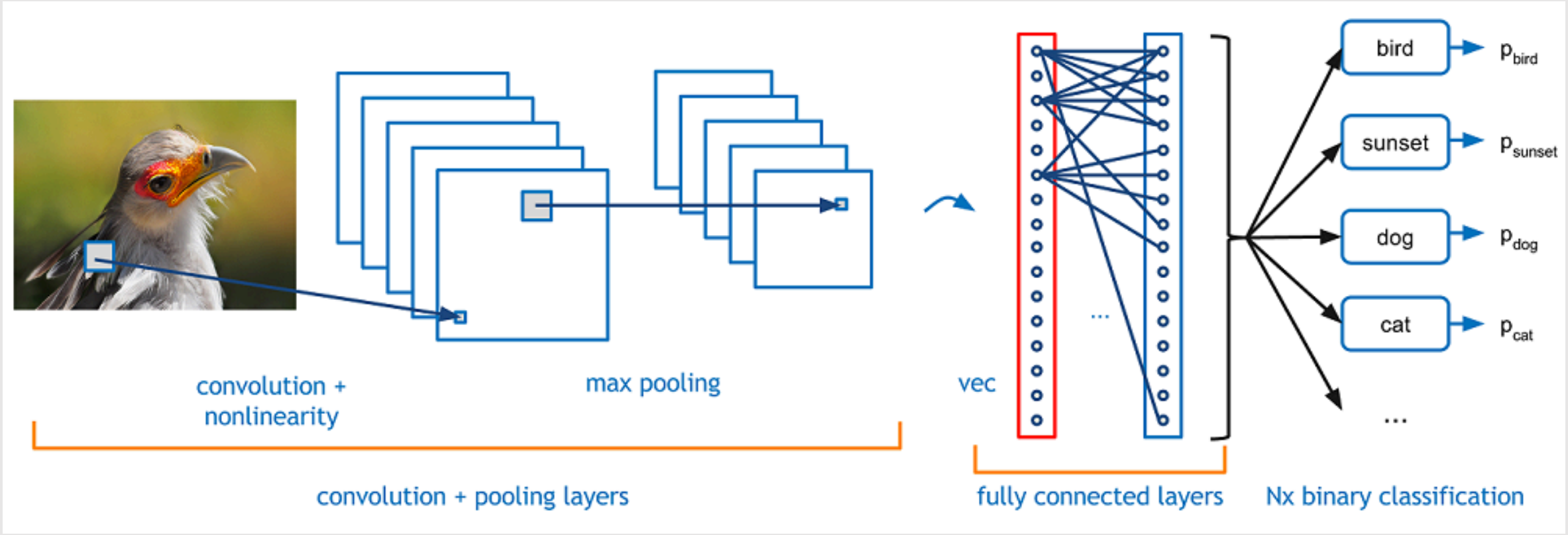
Small, application-specific data



Your model
Trained for your application



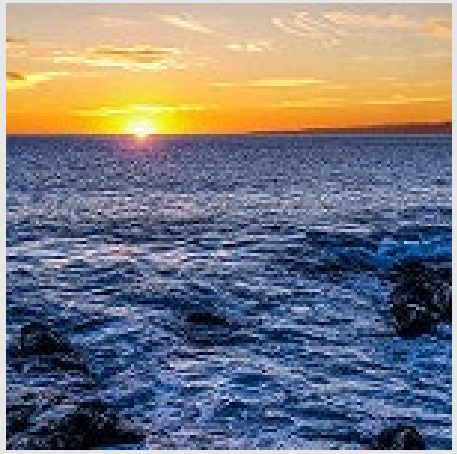
Transfer Learning – Learning the features



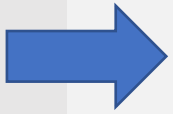
Extract Image features

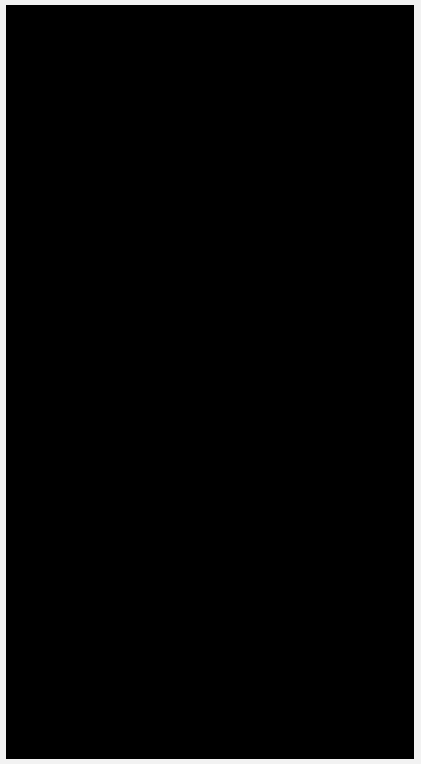
Features summary

Transfer Learning – Black box

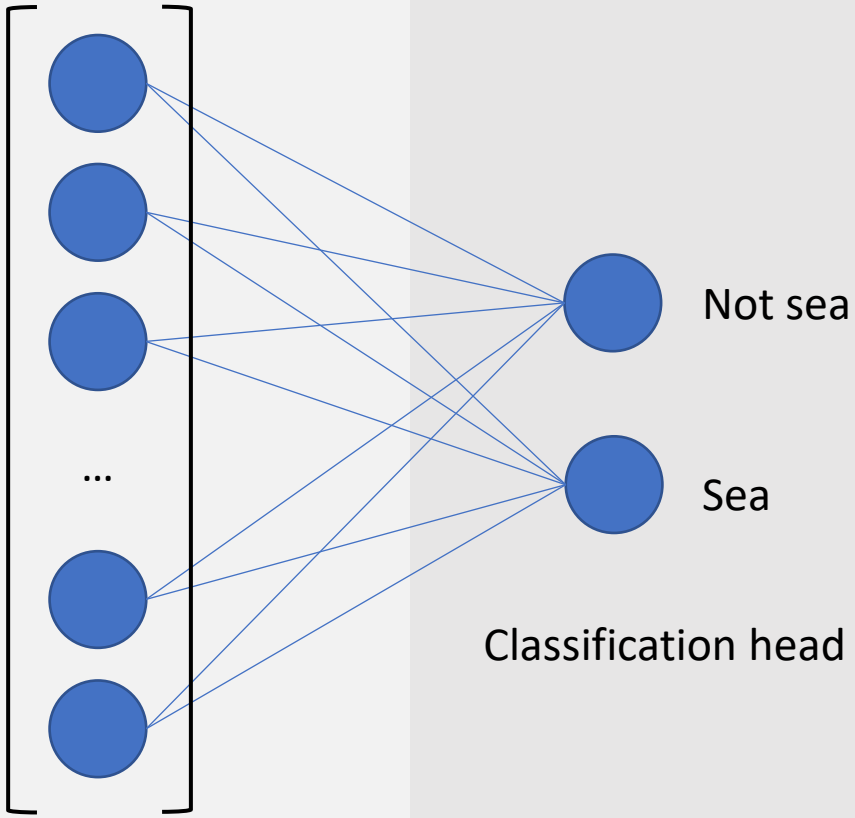


Input – Rank 3 Tensor



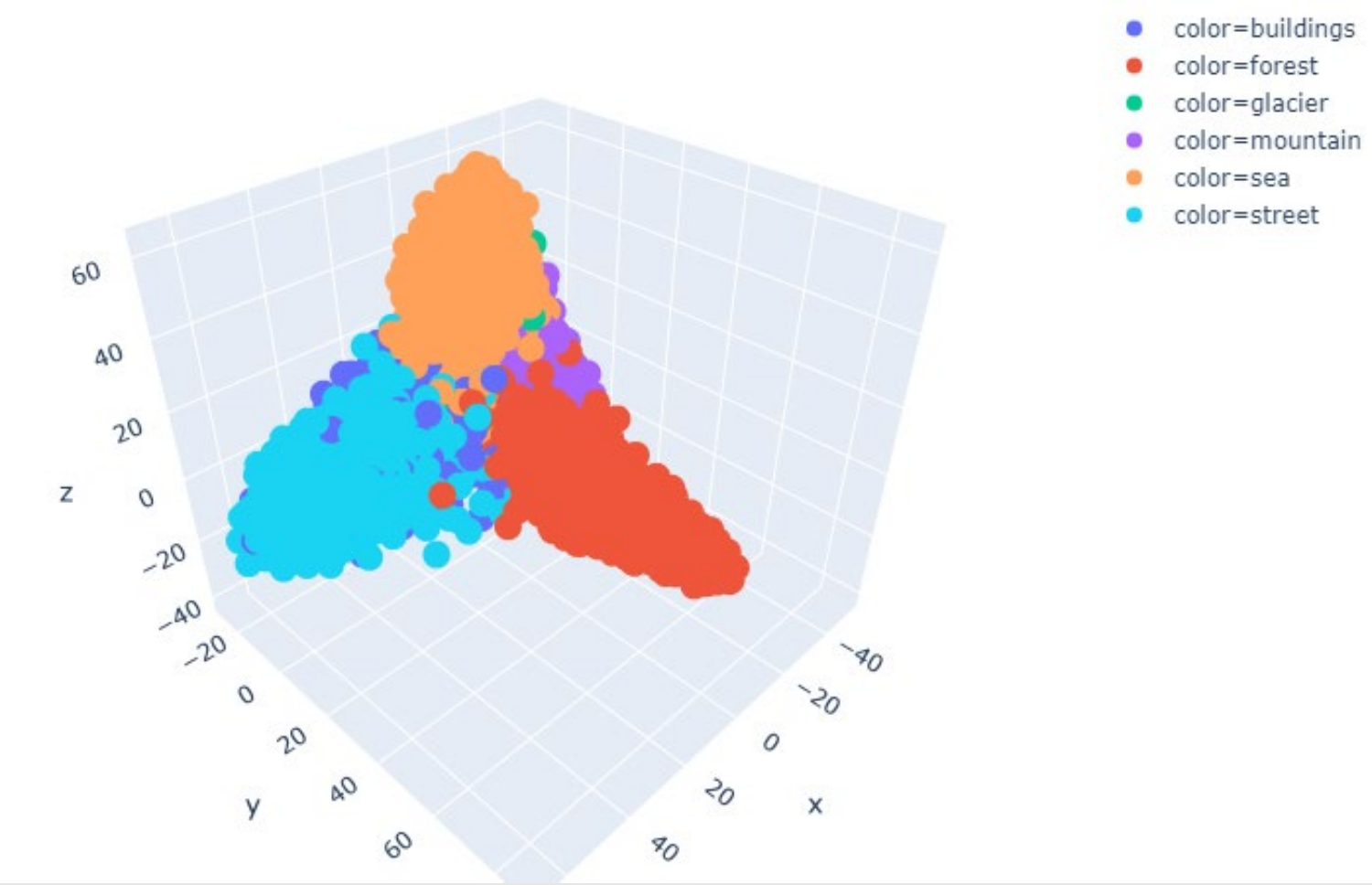


Black box – Pre-trained model



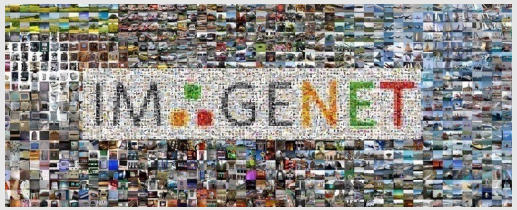
Output - Features vector

Transfer Learning – Black box



Features vectors of Intel Image Classification dataset in 3D space
(Generated from Google's BiT-M model)

Sample Transfer Learning Workflow



SOTA model → Pre-trained model

TensorFlow Hub

To be covered in this Workshop

Define the problem → Gather data → Choose an appropriate model → Train the model → Deploy the model

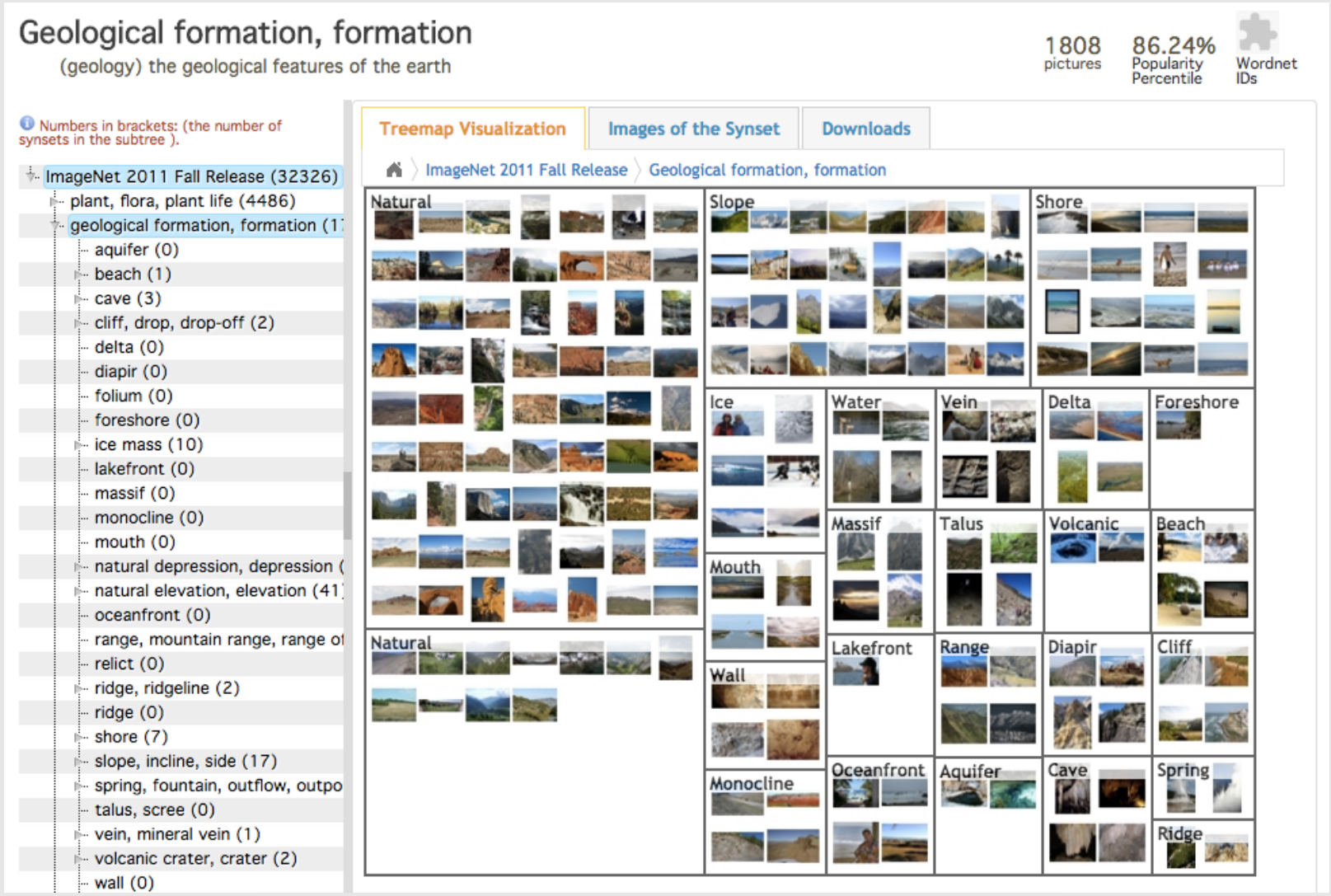
Image Classification

Intel Image Classification

Upstream Training - ImageNet

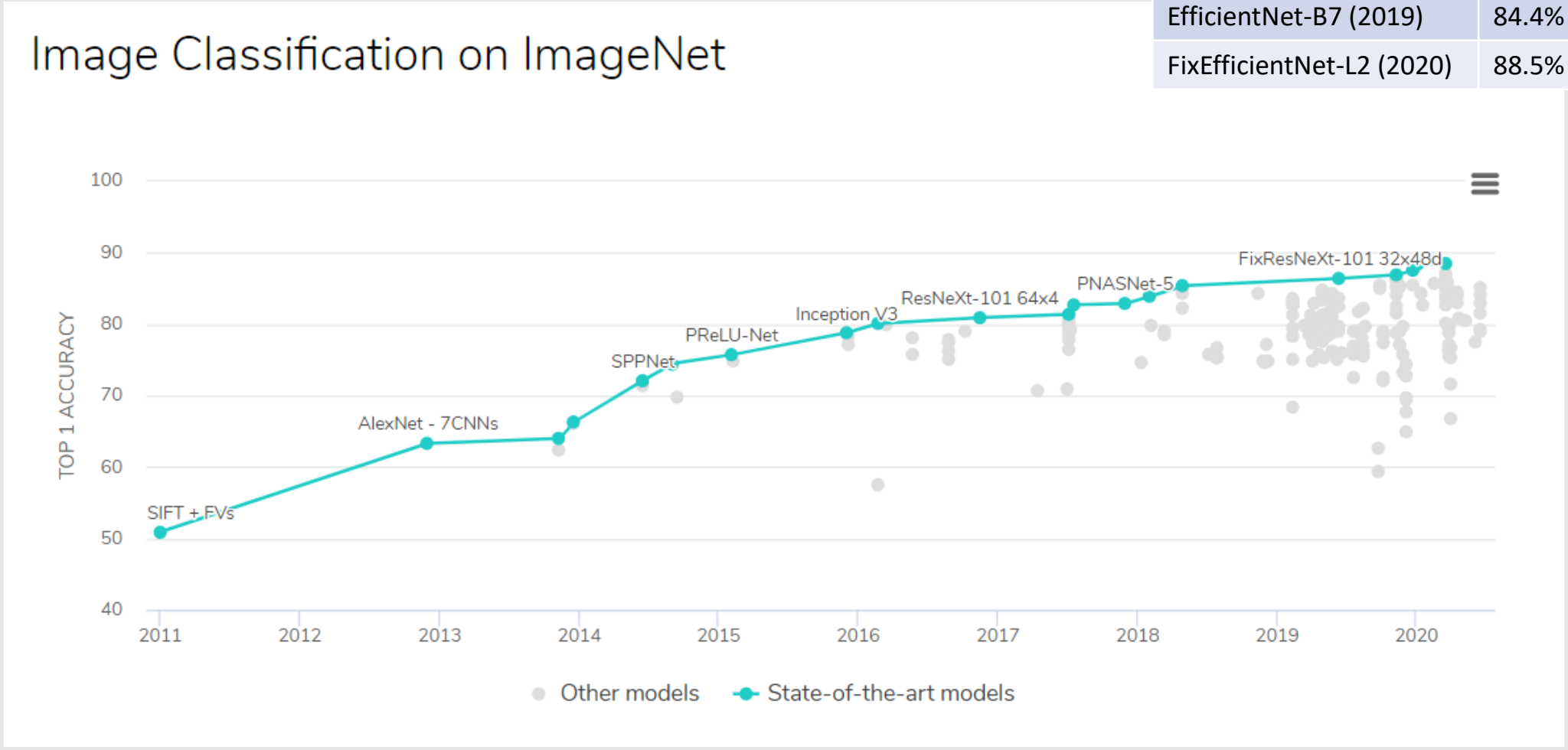
- ImageNet: a database of images for visual object recognition research
  - 14 million images, hand-annotated
  - 20,000 categories (classes)
  - The standard dataset for evaluating neural network architecture in research
  - http://image-net.org/explore

Upstream Training - ImageNet



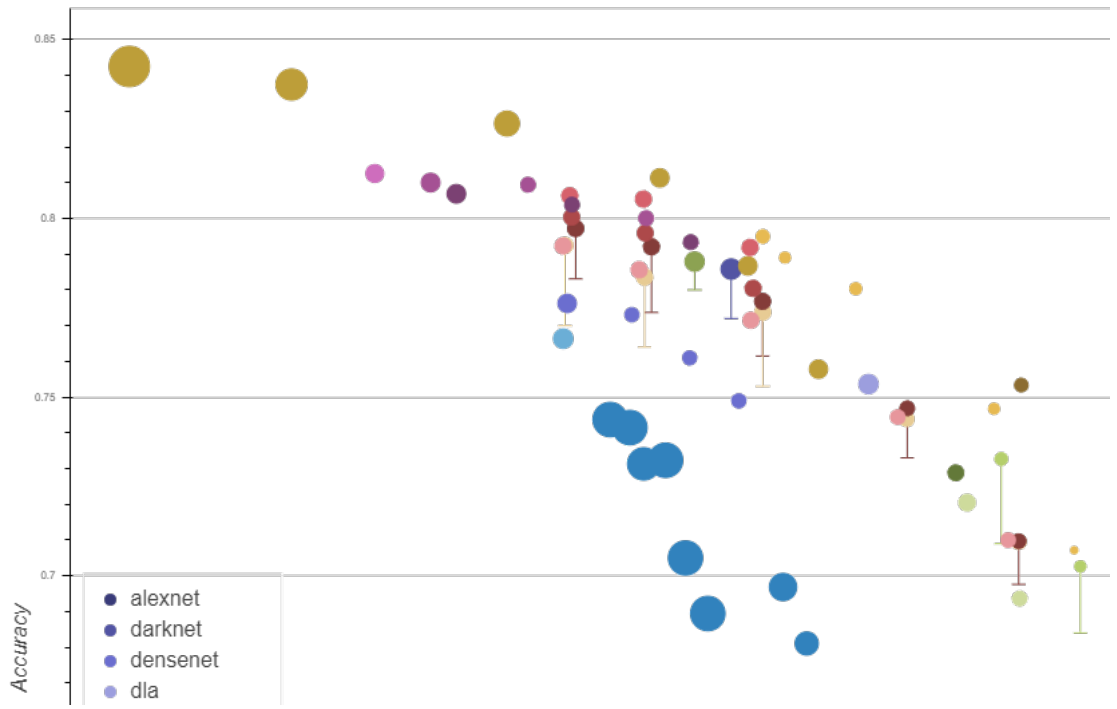
SOTA models

Model	Top-1 accuracy
AlexNet (2012)	63.3%
VGG-19 (2015)	74.5%
ResNet-152 (2016)	77.8%
ResNeXt-101 (2017)	80.9%
EfficientNet-B7 (2019)	84.4%
FixEfficientNet-L2 (2020)	88.5%

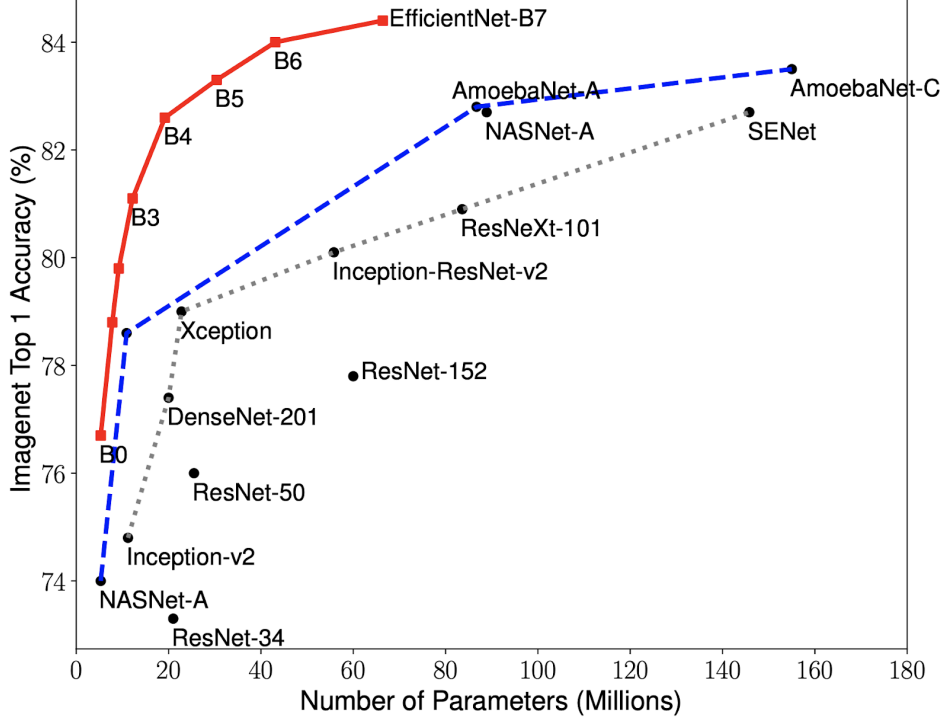


Source: https://paperswithcode.com/sota/image-classification-on-imagenet

Choose a model



Accuracy vs Throughput Trade-off

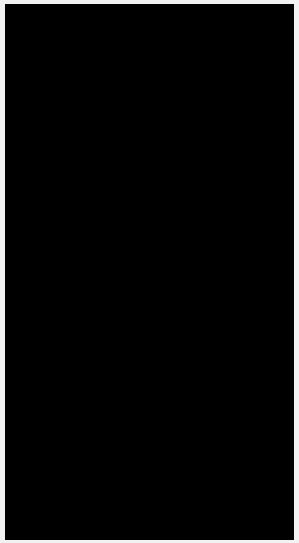


Model size

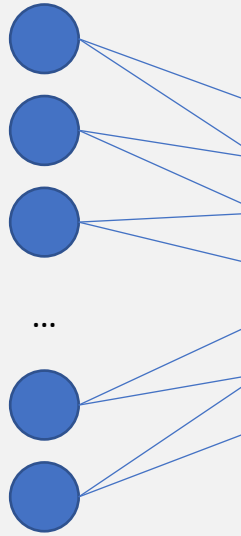
Define the problem → Gather data → Choose an appropriate model → Train the model → Deploy the model

Prepare the model

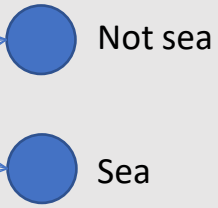
Prepare image input
according to how the pre-trained model was trained



Other layers
(Convolution…)



Features vector
(Dense layer)



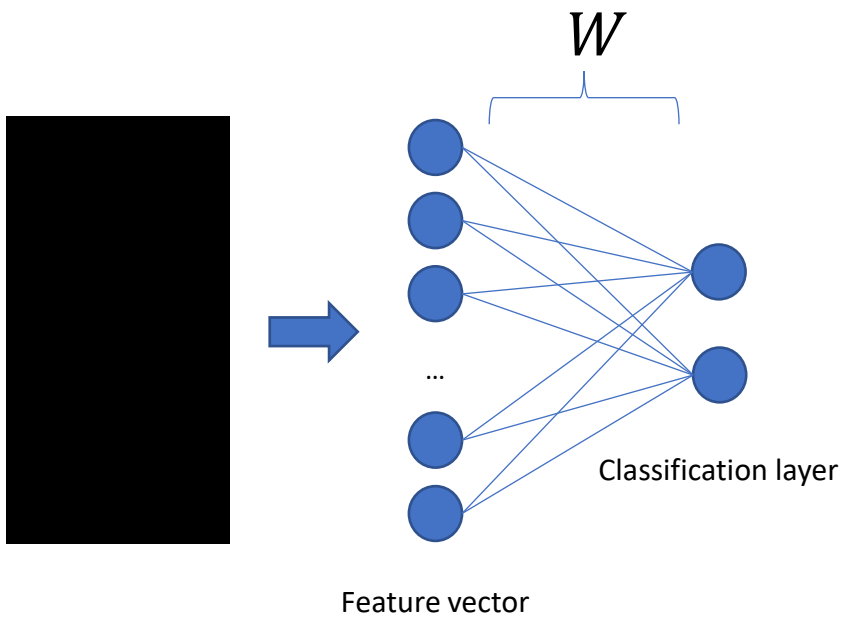
Add classification head
based on the number of output classes

Pre-trained model

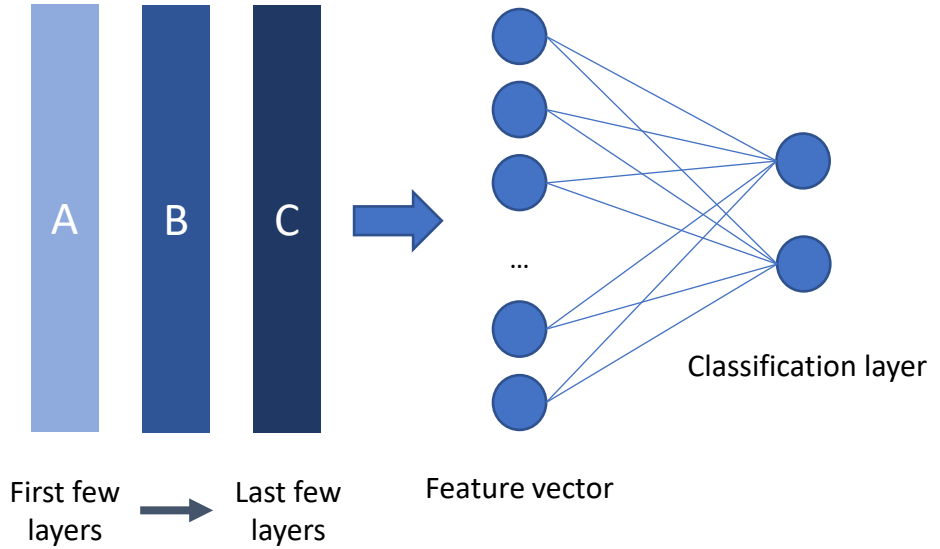
Define the problem → Gather data → Choose an appropriate model → Train the model → Deploy the model

Classification layer and Fine-tuning

Train the classification layer only



Train the whole network
Train selectively the last few layers



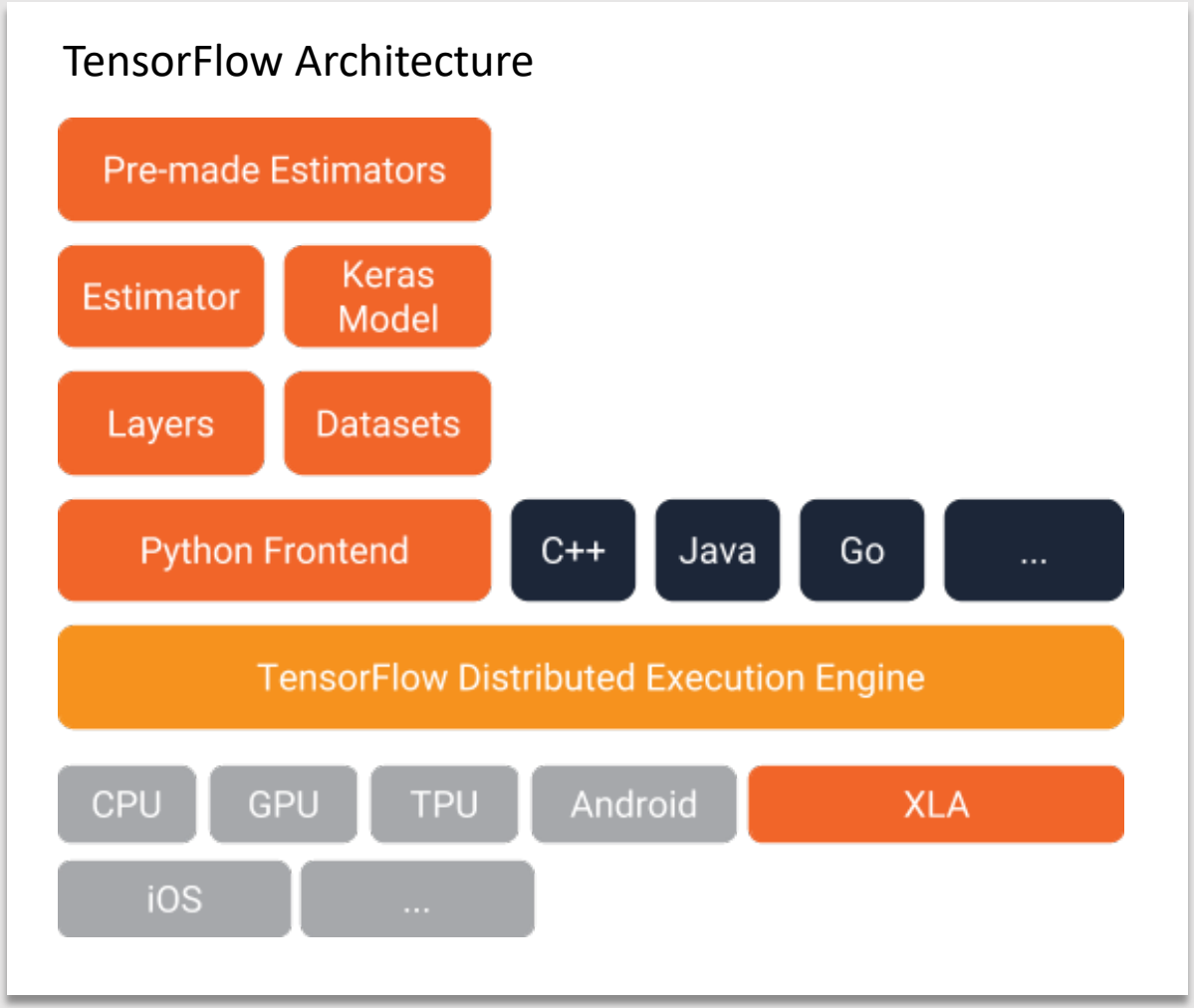
(Will not be covered during Hands-on)

Define the problem → Gather data → Choose an appropriate model → Train the model → Deploy the model

TensorFlow and Keras

TensorFlow's **high-level APIs** are based on the Keras API standard for defining and training neural networks.

Keras enables **fast** prototyping, state-of-the-art research, and production—all with **user-friendly APIs**.



TensorFlow and Keras

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

Run code now Try in Google's interactive notebook

```
class MyModel(tf.keras.Model):
  def __init__(self):
    super(MyModel, self).__init__()
    self.conv1 = Conv2D(32, 3, activation='relu')
    self.flatten = Flatten()
    self.d1 = Dense(128, activation='relu')
    self.d2 = Dense(10, activation='softmax')

  def call(self, x):
    x = self.conv1(x)
    x = self.flatten(x)
    x = self.d1(x)
    return self.d2(x)
model = MyModel()

with tf.GradientTape() as tape:
  logits = model(images)
  loss_value = loss(logits, labels)
grads = tape.gradient(loss_value, model.trainable_variable
optimizer.apply_gradients(zip(grads, model.trainable_varia
```

Run code now Try in Google's interactive notebook

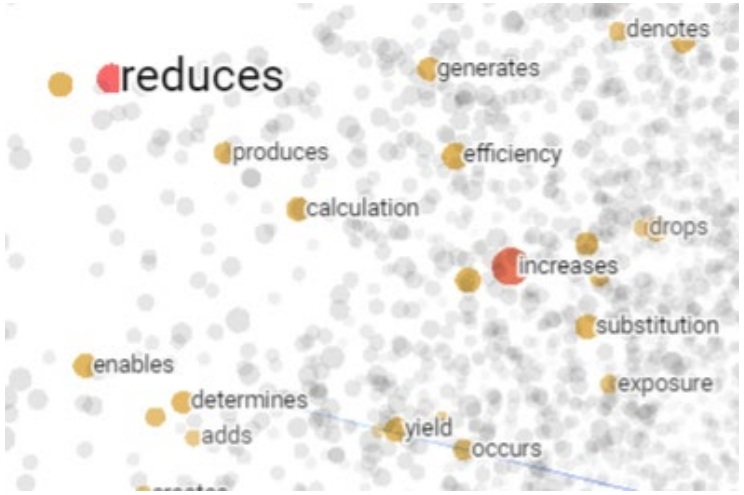
Hands-on session

https://github.com/MLDA-NTU/Transfer-Learning-DL2020

Transfer Learning for NLP tasks

Upstream training with Wikipedia data

Word embeddings



Word2Vec 10K
https://projector.tensorflow.org

Language models

GPT-3

BERT

Turing-NLG

RoBERTa

Can be obtained from

Hugging Face

Feedback

https://docs.google.com/forms/d/e/1FAIpQLSdxmE7yQsHjNOZVVY654AQufyxJHMF-G_ytjHg4ep7ZBiL5Mw/viewform

Thien Tran
tran0096@e.ntu.edu.sg

MLDA@EEE