×
-
(http://play.google.com/store/apps/details?id=com.analyticsvidhya.android)

LOGIN / REGISTER (HTTPS://ID.ANALYTICSVIDHYA.COM/ACCOUNTS/LOGIN/?NEXT=HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2018/07/TOP-10-PRETRAINED-MODELS-GET-STARTED-DEEP-LEARNING-PART-1-COMPUTER-VISION/)

Top 10 Pretrained Models to get you Started with Deep Learning (Part 1 – Computer Vision)

PRANAV DAR (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/AUTHOR/DATASCIENCE22/), JULY 27, 2018 LOGIN TO BOO...

Introduction

Pretrained models are a wonderful source of help for people looking to learn an algorithm or try out an existing framework. Due to time restrictions or computational restraints, it's not always possible to build a model from scratch which is why pretrained models exist! You can use a pretrained model as a benchmark to either improve the existing model, or test your own model against it. The potential and possibilities are vast.

Source: Facebook AI

In this article, we will look at various pretrained models in Keras that have applications in computer vision. Why Keras? First, because I believe it's a great library for people starting out with neural networks. Second, I wanted to stick with one framework throughout the article. This will help you in moving from one model to the next without having to worry about frameworks.

I encourage you to try each model out on your own machine, understand how it works, and how you can improve or tweak the internal parameters.

We have segmented this topic into a series of articles. Part II will focus on Natural Language Processing (NLP) and Part III will look at Audio and Speech models. The aim is to get you up and running in these fields with existing solutions which will fast track your learning process.

Table of Contents

Object Detection

Object detection is one of the most common applications in the field of computer vision. It has applications in all walks of life, from self-driving cars to counting the number of people in a crowd. This section deals with pretrained models that can be used for detecting objects. You can also check out the below articles to get familiar with this topic:

- Understanding and Building an Object Detection Model from Scratch in Python (https://www.analyticsvidhya.com/blog/2018/06/understanding-building-object-detection-model-python/)

Mask R-CNN (https://github.com/matterport/Mask_RCNN)

Mask R-CNN is a flexible framework developed for the purpose of object instance segmentation. This pretrained model is an implementation of this Mask R-CNN technique on Python and Keras. It generates bounding boxes and segmentation masks for each instance of an object in a given image (like the one shown above).

This GitHub repository features a plethora of resources to get you started. It includes the source code of Mask R-CNN, the training code and pretrained weights for MS COCO, Jupyter notebooks to visualize each step of the detection pipeline, among other things.

YOLOv2 (https://github.com/experiencor/keras-yolo2)

YOLO is an ultra popular object detection framework for deep learning applications. This repository contains implementations of YOLOv2 in Keras. While the developers have tested the framework on all sorts of object images – like kangaroo detection, self-driving car, red blood cell detection, etc., they have released the pretrained model for raccoon detection.

You can download the **raccoon dataset here (https://github.com/experiencor/raccoon_dataset)** and get started with this pretrained model now! The dataset consists of 200 images (160-training, 40-validation). You can download the pretrained weights for the entire model **here (https://1drv.ms/f/s!ApLdDEW3ut5feoZAEUwmSMYdPIY)**. According to the developers, these weights can be used for an object detector for one class.

MobileNet (https://keras.io/applications/#mobilenet)

As the name suggests, MobileNet is an architecture designed for mobile devices. It has been built by none other than Google. This particular model, which we have linked above, comes with pretrained weights on the popular ImageNet database (it's a database containing millions of images belonging to more than 20,000 classes).

As you can see above, the applications of MobileNet are not just limited to object detection but span a variety of computer vision tasks – like facial attributes, landmark recognition, finegrain classification, etc.

Ripe/Unripe Tomato Classification (https://github.com/fyrestorm-sdb/tomatoes)

If you were given a few hundred images of tomatoes, how would you classify them – say defective/non-defective, or ripe/unripe? When it comes to deep learning, the go-to technique for this problem is image processing. In this classification problem, we have to identify whether the tomato in the given image is grown or unripe using a pretrained Keras VGG16 model.

The model was trained on 390 images of grown and unripe tomatoes from the ImageNet dataset and was tested on 18 different validation images of tomatoes. The overall result on these validation images is given below:

Recall	0.8888889
Precision	0.9411765
F1 Score	0.9142857

Car Classification (https://github.com/michalgdak/car-recognition)

There are numerous ways of classifying a vehicle – by it's body style, number of doors, open or closed roof, number of seats, etc. In this particular problem, we have to classify the images of cars into various classes. These classes include make, model, year, e.g. 2012 Tesla Model S. To develop this model, the **car dataset from Stanford (http://ai.stanford.edu/~jkrause/cars/car_dataset.html)** was used which contains 16,185 images of 196 classes of cars.

The model was trained using pretrained VGG16, VGG19 and InceptionV3 models. The VGG network is characterized by its simplicity, using only *3×3* convolutional layers stacked on top of each other in increasing depth. The 16 and 19 stand for the number of weight layers in the network.

As the dataset is small, the simplest model, i.e. VGG16, was the most accurate. Training the VGG16 network gave an accuracy of 66.11% on the cross validation dataset. More complex models like InceptionV3 were less accurate due to bias/variance issues.

Facial Recognition and Regeneration

Facial recognition is all the rage in the deep learning community. More and more techniques and models are being developed at a remarkable pace to design facial recognition technology. Its applications span a wide range of tasks – phone unlocking, crowd detection, sentiment analysis by analyzing the face, among other things.

Face regeneration on the other hand, is the generation of a 3D modelled face from a closeup image of a face. The creation of 3D structured objects from mere two dimensional information is another thought out problem in the industry. The applications of face regeneration are vast in the film and gaming industry. Various CGI models can be automated thus saving tons of time and money in the process.

This section of our article deals with pretrained models for these two domains.

VGG-Face Model (https://gist.github.com/EncodeTS/6bbe8cb8bebad7a672f0d872561782d9)

Creating a facial recognition model from scratch is a daunting task. You need to find, collect and then annotate a ton of images to have any hope of building a decent model. Hence using a pretrained model in this domain makes a lot of sense.

VGG-Face (http://www.robots.ox.ac.uk/~vgg/data/vgg_face/) is a dataset that contains 2,622 unique identities with more than two million faces. This pretrained model has been designed through the following method:

- **vgg-face-keras:** Directly convert the vgg-face model to a keras model
- **vgg-face-keras-fc:** First convert the vgg-face Caffe model to a mxnet model, and then convert it to a keras model

3D Face Reconstruction from a Single Image (https://github.com/dezmoanded/vrn-torch-to-keras)

This is a really cool implementation of deep learning. You can infer from the above image how this model works in order to reconstruct the facial features into a 3 dimensional space.

This pretrained model was **originally developed using Torch (https://github.com/AaronJackson/vrn)** and then transferred to Keras.

Segmentation

Semantic Image Segmentation – Deeplabv3+ (https://github.com/bonlime/keras-deeplab-v3-plus)

Semantic image segmentation is the task of assigning a semantic label to every single pixel in an image. These labels can be “sky”, “car”, “road”, “giraffe”, etc. What this technique does is it finds the outlines of objects and thus places restrictions on the accuracy requirements (this is what separates it from image level classification which has a much looser accuracy requirement).

Deeplabv3 is Google’s latest semantic image segmentation model. It was originally created using TensorFlow and has now been implemented using Keras. This GitHub repository also has code for how to get labels, how to use this pretrained model with custom number of classes, and of course how to trail your own model.

Robot Surgery Segmentation (https://github.com/ternaus/robot-surgery-segmentation)

This model attempts to address the problem of image segmentation of surgical instruments in a robot-assisted surgery scenario. The problem is further divided into two parts, which are as follows:

1. **Binary segmentation:** Every pixel in an image is labelled as an instrument or background
2. **Multi-class segmentation:** Different instruments or different parts of an instrument are distinguished from the background

This pretrained model is based on the U-Net network architecture and is further improved by using state-of-the-art semantic segmentation neural networks known as LinkNet and TernausNet. The model was trained on 8 × 225-frame sequences of high resolution stereo camera images.

Miscellaneous

Image Captioning (https://github.com/boluoyu/ImageCaption)

Remember those games where you were given images and had to come up with captions? That's basically what image captioning is. It uses a combination of NLP and Computer Vision to produce the captions. This task has been a challenging one for a long time as it requires huge datasets with unbiased images and scenarios. Given all these constraints, the algorithm must be generalized for any given image.

A lot of businesses are leveraging this technique nowadays but how can you go about using it? The solution lies in converting a given input image into a short and meaningful description. The encoder-decoder framework is widely used for this task. The image encoder is a convolutional neural network (CNN).

This is a VGG 16 pretrained model on the **MS COCO dataset (http://cocodataset.org/#download)** where the decoder is a long short-term memory (LSTM) network predicting the captions for the given image. For detailed explanation and walk through it's recommended that you follow up with our article on **Automated Image Captioning** (https://www.analyticsvidhya.com/blog/2018/04/solving-an-image-captioning-task-using-deep-learning/).

End Notes

Deep learning is a tricky field to get acclimated with, that's why we see researchers releasing so many pretrained models. Having personally used them to understand and expand my knowledge of object detection tasks, I highly recommend picking a domain from the above and using the given model to get your own journey started.

In the next article, we will dive into Natural Language Processing. If you have any feedback or suggestions for this article and the series as a whole, use the comments section below to let me know.

(https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/?share=linkedin&nb=1&nb=1)

(https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/?share=facebook&nb=1&nb=1)

(https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/?share=twitter&nb=1&nb=1)

(https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/?share=pocket&nb=1&nb=1)

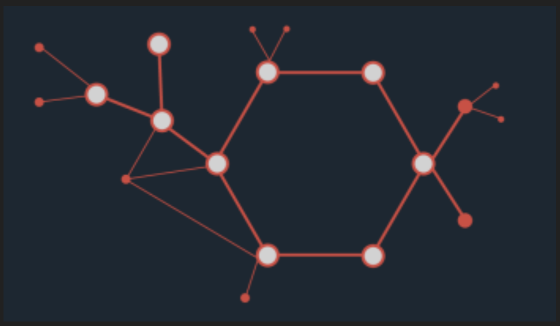
(https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/?share=reddit&nb=1&nb=1)

Like this:

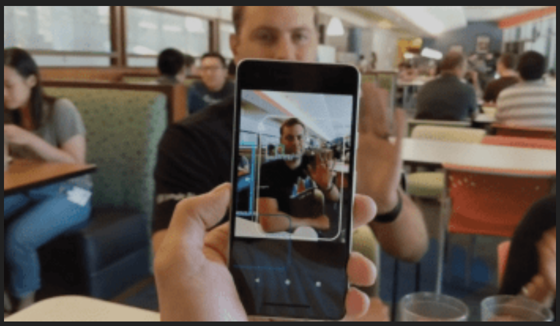
Loading...

Related Articles

(https://www.analyticsvidhya.com/blog/2019/03/pretrained-models-get-started-nlp/)
8 Excellent Pretrained Models to get you Started with Natural Language Processing (NLP) (https://www.analyticsvidhya.com/blog/2019/03/pretrained-models-get-started-nlp/)
March 18, 2019
In "Deep Learning"



(https://www.analyticsvidhya.com/blog/2019/03/deep-learning-frameworks-comparison/)
5 Amazing Deep Learning Frameworks Every Data Scientist Must Know! (with Illustrated Infographic) (https://www.analyticsvidhya.com/blog/2019/03/deep-learning-frameworks-comparison/)
March 14, 2019
In "Deep Learning"



(https://www.analyticsvidhya.com/blog/2018/07/build-computer-vision-model-tensorflow-object-detection-api/)
Build your own Computer Vision Model with the Latest TensorFlow Object Detection API Update (https://www.analyticsvidhya.com/blog/2018/07/build-computer-vision-model-tensorflow-object-detection-api/)
July 16, 2018
In "AVbytes"

TAGS : COMPUTER VISION (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/COMPUTER-VISION/), KERAS (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/KERAS/), LIST OF PRETRAINED MODELS (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LIST-OF-PRETRAINED-MODELS/), NEURAL NETWORKS (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/NEURAL-NETWORKS/), OBJECT DETECTION (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/OBJECT-DETECTION/), PRETRAINED MODEL (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/PRETRAINED-MODEL/)

NEXT ARTICLE

Making AI Systems Faster and More Efficient with Google Edge TPUs

(https://www.analyticsvidhya.com/blog/2018/07/google-unveils-ai-chips-on-device-machine-learning/)

•••

PREVIOUS ARTICLE

Infographic – 13 Common Mistakes Amateur Data Scientists Make and How to Avoid Them

(https://www.analyticsvidhya.com/blog/2018/07/infographic-common-mistakes-amateur-data-scientists-make-how-avoid-them/)

(https://www.analyticsvidhya.com/blog/author/datascience22

Pranav Dar (Https://Www.analyticsvidhya.com/Blog/Author/Datascience22/)

Senior Editor at Analytics Vidhya. Data visualization practitioner who loves reading and delving deeper into the data science and machine learning arts. Always looking for new ways to improve processes using ML and AI.

This article is quite old and you might not get a prompt response from the author. We request you to post this comment on Analytics Vidhya's **Discussion portal** (https://discuss.analyticsvidhya.com/) to get your queries resolved

ONE COMMENT

ARELIS GUZMÁN

Reply

October 15, 2018 at 4:24 am (https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/#comment-155314)

Thank you for insteresting article. I would like have the part II.

POPULAR POSTS

- 8 Useful R Packages for Data Science You Aren't Using (But Should!) (https://www.analyticsvidhya.com/blog/2019/04/8-useful-r-packages-data-science/)
- 24 Ultimate Data Science Projects To Boost Your Knowledge and Skills (& can be accessed freely) (https://www.analyticsvidhya.com/blog/2018/05/24-ultimate-data-science-projects-to-boost-your-knowledge-and-skills/)
- Understanding Support Vector Machine algorithm from examples (along with code) (https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/)
- A Complete Tutorial to Learn Data Science with Python from Scratch (https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-learn-data-science-python-scratch-2/)
- 7 Types of Regression Techniques you should know! (https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/)
- Essentials of Machine Learning Algorithms (with Python and R Codes) (https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/)
- Introduction to k-Nearest Neighbors: Simplified (with implementation in Python) (https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/)
- A Simple Introduction to ANOVA (with applications in Excel) (https://www.analyticsvidhya.com/blog/2018/01/anova-analysis-of-variance/)

RECENT POSTS

Statistics for Data Science: Introduction to the Central Limit Theorem (with implementation in R) (https://www.analyticsvidhya.com/blog/2019/05/statistics-101-introduction-central-limit-theorem/)

MAY 3, 2019

Don't Miss these 5 Data Science GitHub Projects and Reddit Discussions (April Edition) (https://www.analyticsvidhya.com/blog/2019/05/5-data-science-github-reddit-april/)

MAY 1, 2019

Last Chance to Register on our Flagship Program – AI & ML BlackBelt (https://www.analyticsvidhya.com/blog/2019/05/last-chance-register-course-ai-ml-blackbelt/)

MAY 1, 2019

Winning Solutions and Approaches from the Machine Learning Hikeathon: Feature Engineering Special! (https://www.analyticsvidhya.com/blog/2019/04/ml-hikeathon-winning-solution-approaches/)

APRIL 30, 2019

- Computer Vision Model in 10 min