03. List of tasks

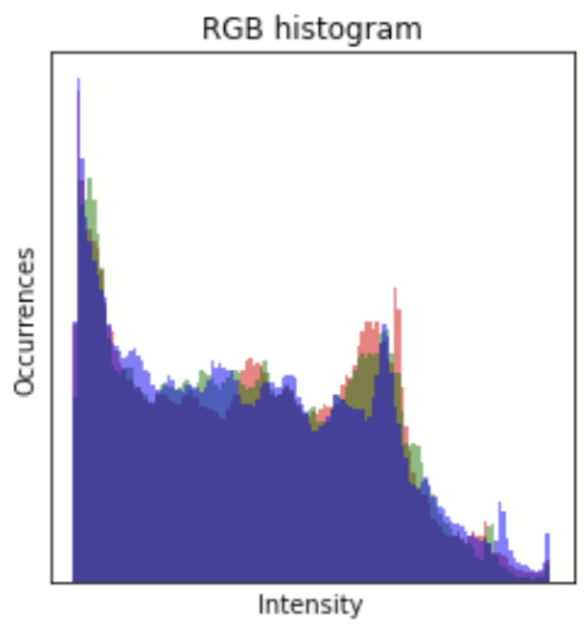
1. Data exploration and feature extraction

In this first task of the project, you will explore the Swissroads dataset and extract a set of **high-level features** for each image in it. To achieve this, you can use the MobileNet v2 pretrained ConvNet which extracts 1280 high-level features.

These high-level features should then be used for all of the subsequent tasks in this project, except for when it is stated otherwise. In other words, all models in tasks 2-7 (except for the Convolutional Neural Network of task 8) should use these high-level features. And in the case where we ask you to visualize the images, we of course mean to visualize the raw images with their pixel values.

Start by doing the following:

- Load the images from the three sets (train, validation, and test). Since the images will be fed to a pretrained model for feature extraction, make sure to set the scale of the pixels and the image size as expected by the model (you can check the expected scale and size in the official documentation page)
- Plot a few images from each category (bike, car, motorcycle, other, truck, van).
- Use an appropriate visualization to show the proportions of each category in the three sets (train, validation, and test).
- (Optional) For each of the 6 categories in the training dataset, plot the color histogram. An example of a color histogram is depicted below. For colored RGB images, as in our case, the color histogram shows the statistics of the three colors (red, green, and blue) as a function of their intensities in a given image or dataset. You are required to produce 6 such plots, each one should represent the statistics of three colors in all the images belonging to a specific category. What can you comment on these histograms? Do you spot distinctive pattern(s) in them across the different categories? Do you think such statistics are enough to distinguish the 6 categories?



- Extract the high-level features for all the images in the three datasets using MobileNet v2. Consider storing the extracted **high-level features**, e.g. in npz files, for quickly reloading them into each of the following notebooks.

50 hours Bookmark
NEXT
Reach out for personalized support:
BOOK A 1-TO-1
More options
Course 42%
04. Applied machine learning 2
Subject 0%
01. Welcome to Applie... ✓
02. K-nearest neighbors ✓
03. Bias-variance trad... ✓
04. Logistic regressions ✓
05. Decision trees and... ✓
06. Clustering and dim... ✓
07. Introduction to dee... ✓
08. Multilayer and con... ✓
09. Tensorflow ecosys... ✓
10. Machine learning ... ✓
11. Course summary ✓
12. Course project
Unit
01. Welcome ✓
02. What to remember... ✓
03. List of tasks
04. Final checklist

Tips and Tricks

To extract high-level features you can follow the example provided in the unit *Transfer learning* of the subject *Tensorflow ecosystem*

- (Optional) For each of the 6 categories in the training dataset, use a heatmap to visualize the intensity of the 1280 features for all the samples belonging to a specific category. Comment on your finding:
  - Do the high-level features behave the same across the different categories?
- For each of the 6 categories in the training dataset, answer the following questions:
  - What are the top 5 features that have the highest mean value?
  - Which categories share the same most active feature (i.e. have highest means)?

Note

All your models should be **trained** on the training set, and the fine tuning of your hyperparameters should be **validated** on the validation set. The final test set should only be used for the final comparison to **test** the accuracies of your models on a new dataset.

2. PCA analysis and clustering

You will further explore the high-level features by doing the following:

- Apply PCA analysis on the training dataset. Make a scree plot, how many PCA components explain 10%, 20%, ..., 90% and 100% of the variance?
- After transforming the training dataset **using the first two PCA components**, visualize the transformed data on a 2D-plot and use 6 different colors to designate the 6 categories. What can you say about the results?
- On the 2D transformed training data, apply k-means clustering with k equals 6. Plot the same 2D-plot as above but this time use the colors to distinguish between the six clusters obtained by k-means. Do the 6 clusters of k-means align with the 6 categories from above?
- Using the first two PCA components obtained form the training dataset, transform the test dataset. Visualize the transformed data on a 2D-plot and use 6 different colors to designate the 6 categories. Based on this visualization, how many sample in the test dataset you think might be hard to classify?

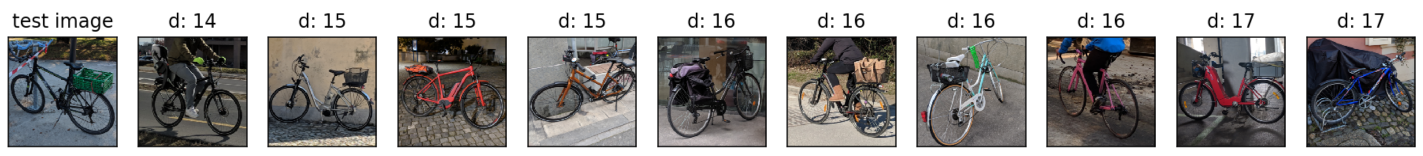
Note

In some of the following tasks you are asked to tune the model hyperparameters. **For all tuned models** you should display the training and validation scores, plot the corresponding curves and state your observations regarding overfitting. Moreover **for all models** provide a classification report and visualize the confusion matrix of the test set.

3. Visual search with k-NN

The idea here is to implement a little search engine based on the high-level image representation and k-nearest neighbors

- Fit and tune a k-NN classifier. Visualize the train and validation curves
- Provide a classification report and visualize the confusion matrix of the test dataset
- Pick an image which was correctly classified in the test set and plot its 10 nearest neighbors from the training set. An example is depicted below
- Pick an image which was misclassified in the test set and plot its 10 neighbors from the training set



Tips and Tricks

Take a look at the `kneighbors()` method from Scikit-learn k-NN estimators.

4. Logistic regression

Next let's explore linear models, more specifically logistic regression. Unlike the k-NN model, it has internal parameters that need to be optimized.

- Train and evaluate a logistic regression model (without any regularization `penalty="none"` and without any hyperparameters tuning).
- Get the model coefficients using the `coef_` attribute of your estimator and visualize them using a heatmap.
  - What are the largest 5 coefficients for each category (i.e. the indices of these coefficients)?
  - Are these results consistent with your observations during the data exploration in the last question of Task 1?
- Set an "l2" regularization and tune the regularization strength parameter of the model with cross-validated grid-search.

Note

For cross-validation, you can merge the train and validation set into one bigger dataset and use this for model fitting. For info on computational issues see our comment at the very end of the project.

- Report the result of cross-validated grid-search as a dataframe and interpret the result. In particular, briefly explain what are the `mean_train_score` , `mean_test_score` , `std_train_score` and `std_test_score` :
  - How are they obtained?
  - What do they measure?
- Do the training and validation curves indicate overfitting?

5. Decision trees and random forest

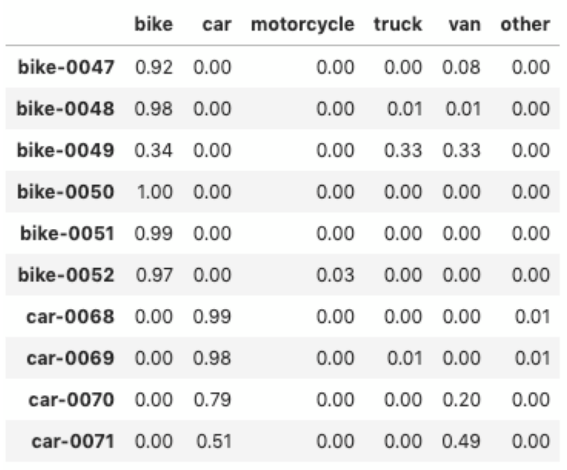
Can you do better than the logistic regression with tree methods which are nonlinear?

- What accuracy can you achieve using a decision trees with a depth of 3? Plot the corresponding decision tree with `plot_tree()` (see here for more)
- Tune the depth of your decision tree. Does it improve the accuracy?
- (Optional) Do you get better results with your decision tree if you reduce the number of dimensions with PCA first?
- Try a random forest model and tune the number of trees and their depth. Does increasing the number of trees help? Is there an optimal depth and how does it compare to your decision tree above?
- The random forest estimator provides a ranking of the features according to their importance in the classification task via the `feature_importances_` attribute. Find the the top 5 important features and visualize their importance scores.
- Are these top features similar to the ones you found based on the coefficients in the logistic regression task, and also during the data exploration in the last question of Task 1?

6. Support vector machine (optional)

Try another nonlinear classifier: SVM. Can you do better than the models from above?

- Try with an SVM classifier. Does the RBF kernel perform better than the linear one? You don't need to tune the hyperparameters.
- Pick ten images and compute the probability for each category using the `predict_proba()` function of the best SVM estimator you found in the previous question. Plot the results for each image and comment your findings. For example, you can plot the ten images in subplots and collect the probabilities in a DataFrame.



	bike	car	motorcycle	truck	van	other
bike-0047	0.92	0.00	0.00	0.00	0.08	0.00
bike-0048	0.98	0.00	0.00	0.01	0.01	0.00
bike-0049	0.34	0.00	0.00	0.33	0.33	0.00
bike-0050	1.00	0.00	0.00	0.00	0.00	0.00
bike-0051	0.99	0.00	0.00	0.00	0.00	0.00
bike-0052	0.97	0.00	0.03	0.00	0.00	0.00
car-0068	0.00	0.99	0.00	0.00	0.00	0.01
car-0069	0.00	0.98	0.00	0.01	0.00	0.01
car-0070	0.00	0.79	0.00	0.00	0.20	0.00
car-0071	0.00	0.51	0.00	0.00	0.49	0.00

- Finally, vary the regularization strength of your SVM estimator (e.g. once using a C value of 0.0001 and once of 1000). What is the effect on the probabilities? Write your observations in a markdown cell.

7. Dense network

Finally, try with neural networks

- 1-layer dense network i.e. no hidden layer, just the input and output ones
- 2-layer dense network i.e. one hidden layer

Note

It is up to you if you want to implement your neural networks for this and the next task with TensorFlow's low-level API or the Keras API. Our recommendation is to practice your knowledge of the Keras API, as this approach is quicker to implement, more straight forward to read and most importantly, the Keras API is since TensorFlow version 2.0 also the preferred API of TensorFlow itself.

8. Convolutional neural network

You tested above different models with the set of high-level features extracted from a pretrained neural network. However, can you get similar results by creating a ConvNet from scratch and using the pixel values from the original images to train the model?

- What accuracy can you achieve?
- Can you get good results? - If not, why?

9. Final Comparison

Collect the test accuracy of all of the models from above in a `09 Results.ipynb` notebook. You can use a DataFrame to store your results for instance

	model	test_accuracy
0	k-NN	...
1	decision tree	...
2	logistic	...
3	random forest	...
4	svm linear	...
5	svm rbf	...
6	1-layer nn	...
7	2-layer nn	...
8	cnn	...

Include a final visualization which summarizes the test accuracy of all of the models from above. For instance, you could use a bar chart.

Note

Running a cross-validation on top of a grid search can computationally be very demanding. Under certain conditions, this might lead to a "never ending" model fit. We recommend that you use `GridSearchCV` together with `verbose=1` to better understand how the computation is progressing. Also, if your computer provides parallel computation, you can use `n_jobs=2` (or `4`, `8` or `-1`) to specify how many parallel computation you want to allow.

QUESTIONS Ask a Question

rosybrown-plum · Learner · a year ago ✓ 1 Answer

3. Visual search with k-NN

Hi,
For the 3rd part, I am not sure what "high-level image representation" means. Should I apply the k-NN classifier on high level features that I extracted in part 1 or on original images?
Thank you

POST ANSWER

Amir Khalilzadeh · Teacher · a year ago ✓ 0

Hi, thanks for this question. The high-level image representation refers to the high level features extracted in part one. All models except the CNN should use high-level features as data. By the end of the project, you will be able to compare the performance of relatively simple models trained on high level information with a complex model (CNN) trained on the original images.

Mandy · Learner · a year ago ✓ 3 Answers

heatmap of coefficients on task 4.2 never ends

Hello,
I was trying to plot the heatmap of the coefficients obtained from the logistic regression (without hyperparameters, taks 4.2) but the plot never ends and times out as the shape of the coefficients is:

```
Shape of coefficients:(6, 150528)
```

I anyway am getting the largest 5 coefficients for each category and the indices of those coefficients, though I am sure from the heatmap I could have a better view.

Is there any way around the problem described above?

POST ANSWER

Amir Khalilzadeh · Teacher · a year ago ✓ 1

Hi, the shape of coefficients for each class should follow the shape of the features used to estimate those coefficients. For instance, if you use *area* and *quality* to predict house prices, then you are estimating two parameters. In your case, you seem to have around 150k coefficients which is not sensible.

Mandy · Learner · a year ago ✓ 0

Thank you Amir for your answer. The above shape is a result of the reshaping of the dataset from 4D (280, 224, 224, 3) to 2D (280, 150528) so that I could fit the StandardScaler and the LogisticRegression models. I guess that then this is not the correct was to do it. Ok I will search for alternatives

Amir Khalilzadeh · Teacher · a year ago ✓ 1

Please note that all models except the CNN should use high-level features as data.

PlWeb · Learner · 2 years ago
Should we '``sparse'`` or ``'categorical'`` the labels?

✓ 1
Answer

Hi,
Should we ' `sparse'` or `'categorical'` the labels?
`ImageDataGenerator` `'categorical"` by default why is that? Should we no avoid one hot encoded for the target value?

POST ANSWER

ChristianLuebbe · Teacher · 2 years ago
That is a good observation and question. You can use both, but `"sparse"` is easier to work with in this project.
`class_mode='categorical'` is indeed the default and gives 2D one-hot encoded labels, here 6 binary columns.
`class_mode="sparse"` on the other hand returns `y` as a vector of integer labels, here 0 - 5. This is easier to work with than the OHE labels.

✓ 1

flovertaco · Learner · 2 years ago
Models impressive performance

✓ 1
Answer

Thank you for this very interesting course project!
I have a side question. I'm actually quite amazed by the level of performance we get, even with simple models such as k-NN. Hence my question: did you filter the set of images that you provided in the 3 sets to ensure high levels of accuracy, or is it just random images and MobileNet feature extractor really does magic?

POST ANSWER

Amir Khalilzadeh · Teacher · 2 years ago
Hi, thanks for your message. No, the images are not selected to ensure high accuracy. But they're checked to have good quality (e.g. blurriness, lighting condition, etc). Most of them clearly indicate the target class, so the high-level features can't go wrong about them.

✓ 1

cype · Learner · 2 years ago
Dying kernels when extracting the features

✓ 2
Answers

Hi there

In task 1 I try to extract the features with MobileNet v2. But in every attempt my kernel dies. I tried different machines (MacBook Pro 2017, iMac 2019, MacBook Air 2021), always the same result: kernel R.I.P. First, the machine works really hard and writes about 50 GB temporary data to the disc. But then always the death of the kernel.

Jupyter Lab doesn't provide any more information about it. It just informs me that the kernel has died. Also in debug-mode.

Kernel Restarting
The kernel for 01 Data exploration and feature extraction.ipynb appears to have died. It will restart automatically.

Any idea?

POST ANSWER

ChristianLuebbe · Teacher · 2 years ago
Could you please push a copy of your notebook to your project git-repo? That would allow us to take a closer look on our side. Thanks

✓ 1

ChristianLuebbe · Teacher · 2 years ago
The problem was resolved in the 1-1, but just posting the issue here in case someone else encounters a similar problem.

The data you feed to your feature extractor must be the image component of your batch, i.e. you must first extract images and labels with `trainset.next()` . If you feed `trainset` , i.e. the object generated by `train_generator.flow_from_directory(...)` , directly to the feature extractor then there is no warning message that indicates the source of the problem. Instead the kernel gets overloaded until it crashes.

✓ 1

favrat-marie · Learner · 2 years ago
Extracting features on all sets or only training?

✓ 2
Answers

Hello, the guidelines are a bit confusing when reading them right now.
First we have ; Extract the high-level features for all the images in the three datasets, and then in the optional task just after we are asked to create a heatmap on the trainset by visualizing the intensity of the 1280 features just extracted.

So my question is :

- shall I extract the features on all set? If yes then on the next question, shall I just cut off the extracted features up till index 280 to only have train set images?

Thank you in advance for your answer.
Marie.

POST ANSWER

Amir Khalilzadeh · Teacher · 2 years ago
Hi, thanks for the question. Here is a summary:

- **Extract the high-level features for all the images in the three datasets**: you should do this per set, i.e. you will do the feature extraction separately for each of the train, validation and test sets. You can also save them separately, as it'll be more convenient to load and use them for other tasks.
- **Heatmap**: take only the extracted features from the train set.

hope this helps

✓ 0

favrat-marie · Learner · 2 years ago
Hello, thank you for your prompt answer!

Make sense to do it per set, thank you very much !
Marie.

✓ 1

Tabea · Learner · 2 years ago ✓ 1 Answer

extracted features - 6 categories

I'm having troubles with the the extracted high-level features. I used the code from the lesson on "Transfer Learning" with the url from MobileNet v2. It looks like this worked, I got arrays of numbers. But how do I know which of the extracted features belong to which of the 6 categories?

POST ANSWER

Panagiota Xydi · Teacher · 2 years ago ✓ 0

Hello!

Starting from the pixels for each image, MobileNet should have allowed you to extract new, more meaningful representations for each of your images with regard to image classification tasks.

The first important thing to observe is the shape of the matrix you extracted with MobileNet; it should be something like (number of images) x (number of high-level features). Therefore, each "row" in this matrix represents an image, and each "column" represents a high-level feature.

This matrix doesn't hold any information about the category of each image. Instead, you should extract the images' labels separately, for example, using the `flow_from_directory` of the `ImageDataGenerator` object as shown in the previous unit "Tensorflow ecosystem" -> "Advanced Keras".

I hope this helps!

Joëlle · Learner · 3 years ago ✓ 1 Answer

Heatmap on features in first exercice

Hello, I don't understand on what data I must create the heatmaps... I have tried to build them directly on the features data but it creates a huge heatmap (with 1280 units) totally unreadable. I suppose it is not correct?

Thanks for help :)

POST ANSWER

Panagiota Xydi · Teacher · 3 years ago ✓ 0

Hello!

This exercise aims to determine which high-level features are the most active for each category in the target variable. For example, you can plot one heatmap per category (car, bike, etc.) using the high-level features. Therefore, for each class, the heatmap "shape" should be: `number of samples in category` x 1'280.

Seaborn's heatmap function has two parameters: `vmin` and `vmax` that can be useful in this case. For example, the `vmin` parameter can help highlight the most active high-level features in a heatmap; it allows setting a lower threshold below which values aren't shown in the heatmap.

Finally, you can use the `figsize` parameter to change the width and height of the heatmaps to be aesthetically pleasing.

I hope this helps!

marta · Learner · 3 years ago

problem with loading model via model_url

✓ 2 Answers

Hello. I tried to rerun my first notebook this morning and suddenly the model does not load. I had no problem with this during the entire week since I started working on the project. The error I get when running this code:

```python
model_url = "https://tfhub.dev/google/imagenet/mobilenet_v2_100_224/
feature_vector/5"

feature_extractor = hub.load(model_url)
```
is:

```
OSError Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_3140/3947694757.py in
2 model_url = "https://tfhub.dev/google/imagenet/mobilenet_v2_100_224/
feature_vector/5"
3
----> 4 feature_extractor = hub.load(model_url)

~\anaconda3\envs\adsml\lib\site-packages\tensorflow_hub\module_v2.py in
load(handle, tags, options)
104 module_path, tags=tags, options=options)
105 else:
--> 106 obj = tf.compat.v1.saved_model.load_v2(module_path, tags=tags)
107 obj._is_hub_module_v1 = is_hub_module_v1 # pylint: disable=protected-access
108 return obj

~\anaconda3\envs\adsml\lib\site-
packages\tensorflow\python\saved_model\load.py in load(export_dir, tags, options)
862 """
863 metrics.IncrementReadApi(_LOAD_V2_LABEL)
--> 864 result = load_internal(export_dir, tags, options)["root"]
865 metrics.IncrementRead()
866 return result

~\anaconda3\envs\adsml\lib\site-
packages\tensorflow\python\saved_model\load.py in load_internal(export_dir, tags,
options, loader_cls, filters)
876 tags = nest.flatten(tags)
877 saved_model_proto, debug_info = (
--> 878 loader_impl.parse_saved_model_with_debug_info(export_dir))
879
880 if (len(saved_model_proto.meta_graphs) == 1 and

~\anaconda3\envs\adsml\lib\site-
packages\tensorflow\python\saved_model\loader_impl.py in
parse_saved_model_with_debug_info(export_dir)
58 parsed. Missing graph debug info file is fine.
59 """
---> 60 saved_model = _parse_saved_model(export_dir)
61
62 debug_info_path = os.path.join(

~\anaconda3\envs\adsml\lib\site-
packages\tensorflow\python\saved_model\loader_impl.py in
parse_saved_model(export_dir)
116 raise IOError("Cannot parse file %s: %s." % (path_to_pbtxt, str(e)))
117 else:
--> 118 raise IOError(
119 "SavedModel file does not exist at: %s%s{%s|%s}" %
120 (export_dir, os.path.sep, constants.SAVED_MODEL_FILENAME_PBTXT,
```

OSError: SavedModel file does not exist at: C:
\Users\marta\AppData\Local\Temp\tfhub_modules\5a2fc00b609485c742fba2c65fa
saved_model.pb}

POST ANSWER

ChristianLuebbe · Teacher · 3 years ago ✓ 2

Hello,

Yes, this is a problem with the temporary folders and files in the background as described in this issue.
Please delete the folder mentioned in the final line of your error message and try to re-execute the code. Hopefully, this should resolve the problem.

If the problem persists could you please let us know your OS-version as well as the current version of tensorflow and tensorflow hub you are using.

marta · Learner · 3 years ago ✓ 1

Thank you very much! Deleting the folder helped!

marta · Learner · 3 years ago ✓ 3 Answers

rescaling vs standardizing data

Hello, when I load the data at the beginning of the project, I also rescale them by using:

```
# Create image generator
dataset_generator = ImageDataGenerator(rescale=1 / 255)
```

I wonder if using the StandardScaler() in any of the following tasks makes sense since the data are already rescaled?

POST ANSWER

Panagiota Xydi · Teacher · 3 years ago ✓ 1

Hello!

Thanks for the question!

Scaling the pixels to the [0,1] range before feeding them to any pre-trained model (MobileNet or Inception Net) to extract the high-level features is correct. However, scaling high-level features after extraction might not be helpful unless you have a good reason. In fact, scaling high-level features could affect the quality of the prediction. The intuition here is that the range and scale of the extracted features learned by the model can hold some information. By standardizing them, we might lose information related to their scale/mean.

I hope this helps!

marta · Learner · 3 years ago ✓ 1

Interesting, thank you for the fast reply. I am curious now what an example of a good reason would be to still apply the StandardScaler(). I have my 1:1 tomorrow, so will ask about this.

Panagiota Xydi · Teacher · 3 years ago ✓ 1

Hello!

Something useful here would be to check whether the high-level features are more or less on the same scale.

marta · Learner · 3 years ago

Plotting neighbours in Part 03

✓ 1 Answer

Hello, I have got my indices and distances and so can choose the 10 nearest neighbours by index from my training set. However, my images have now the shape (1,1280) instead of (224x224x3). How can I reshape them at this point for plotting? (PS. I tried to just use the original training data with low-level features, but since the set is shuffled during training, I do not get what I should get, as the indexes do not match).

POST ANSWER

ChristianLuebbe · Teacher · 3 years ago

✓ 1

Hi,

We are just passing the data through the model to convert it into another format, so we don't need to shuffle our training images for the feature extraction process.
You can set `shuffle` to False inside `flow_from_dataframe` and extract the images in the order they were stored.

Shady · Learner · 3 years ago

Method of loading dataset?

✓ 3 Answers

Hi,

Can you please clarify through what method are we expected to load the images? I attempted to load them as directory iterators (via function flow_from_directory). However, I can't seem to run a for loop to either plot the histograms or even extract the high-level features from the dataset. Nor can I retrieve the labels from the numpy data.

I also attempted to load them via Image.open and cv2 but then I have to manually go through each folder to do so. Through this method I manage to extract the high-level features of a single image, but couldn't figure out how to run it across all images.

Regarding the question related to plotting histogram of the RGB colors of the different categories, how do we combine the colors of all the images of a specific set? I can't seem to get my head around it? Is it by summing the RGB colors of each image of a particular category respectively or through taking the average or via another method?

POST ANSWER

Amir Khalilzadeh · Teacher · 3 years ago

✓ 0

Hi, sorry for the delay. Let's discuss your question in our 1-1 today.

locke_lamora · Learner · 2 years ago

✓ 0

My question is similar to the first question here. We were shown how we get all the images from flow_to_directory into sets using ImageDataGenerator, we were also shown how one can use pretrained models to extract features, but how one connects those two was never shown. That seems like a sizeable omission

All my current solutions right now are work-arounds that require me to only use the filepaths from the flow_to_directory and then read the images back in before using them as batches. That seems like an ugly work-around and hyper frustrating. This approach also requires me to change their resolution again i.e. why did i even say I wanted 224x224 in the image generator.

Amir Khalilzadeh · Teacher · 2 years ago

✓ 0

Hi, it helps to better understand your question if you could be more specific. What do you mean by *how one connects those two*?
The previous note by shady contains several questions.

Dr. Phil · Learner · 3 years ago

✓ 1 Answer

KNN on a 2048 dimensional dataset

Hi,

the featureset I extracted with the NN has 2048 dimensions. Does it make sense to apply KNN onto such a high-dimensional dataset? KNN is distance based and will run into curse of dimensionality issues?
Or should I train the KNN on a PCA-reduced dataset with less dimensions?

Thank you,

POST ANSWER

Mohamad Dia · Teacher · 3 years ago

✓ 1

Hello,

Very good question!

You are right, theoretically with such a dimension on a small dataset we can easily run into curse of dimensionality issues. Especially with KNN where the notion of distance becomes less informative in a very high dimension (almost all points become equidistant if we randomly sample the points **along all the axes**). However, note that the NN used here to extract the features was trained on the ImageNet dataset with more than 1 million images labeled in **1000 different classes** (which include vehicles, animals, plants, food, instruments…). Hence, the features extracted in this project for **6 classes of vehicles only** might share many similarities along the 2048 dimensions, and we might have a **few active dimensions where the distance varies** (e.g. the feature that captures if there is a wing in the image is always inactive here). Therefore, we expect the curse of dimensionality to be less severe in this special case. Of course, if you want to run the KNN on the original 1000 classes then this might be more problematic for the reason mentioned above.

You should train the KNN on the whole features, then you can try if you want to apply the PCA and repeat the experiment (for your own experimentation). Such kind of investigation is already required in the decision tree task later on in the project.

Best,

Dr. Phil · Learner · 3 years ago

✓ 1
Answer

Sensibility of ConvNets to image size, filter size and objects in image

Hi,

Are there any recommendations when we are building the ConvNet, regarding the size of the image (I am considering resampling its size when I create the generators), and/ or the size of the filter o kernel relative to a) the image size or b) the displayed objects size? I ask because the the highest accuracy I've obtained so far with my ConvNet is 57% (most changes I've done to those parameters have given about a constant 50% accuracy).
I'm not sure if my net would need more data points or more complexity, to increase the accuracy. How can I tell?

Also, there's a discussion in the section about ConvNets around using 3D networks to capture the RGB dimension, but if I understand correctly, when we are using the Keras API, `keras.layers.Conv2D` already handles the color dimension (3), is that right? The 2D in the name is confusing.

POST ANSWER

Mohamad Dia · Teacher · 3 years ago

✓ 0

Hello,

The kernel size and also the image size can be considered part of the hyperparameters to be tuned. Typically, the kernel size varies between 3 and 7 in visual recognition problems. This depends on the task and nature of images (whether there are fine textures and low-level variations). The larger the kernel size the bigger the number of parameters to be trained. This has some effects on the complexity of the model and hence on the amount of the required training data. The size of the image (i.e. the number of features) also affects the complexity/deepness of the architecture to be learned and the training data required.

In this example, a few-layers CNN architecture is already complex enough. The real bottleneck is the amount of training data which is relatively small. Hence, the performance you have obtained is expected for such a small dataset. If you want to improve it further, you can try some data augmentation techniques and additional regularization (this is not required though). You can also try different image sizes to see the effect. Decreasing the image size has a double effect: on one side it makes the number of features smaller and hence requires less training data, on the other side we lose some spatial information by decreasing the resolution.

Concerning the `Conv2D` layer whether in TensorFlow or Keras API, it handles three-dimensional data (whatever number of colors 1, 2, 3, ...). You are right about this! For 3-d inputs, the kernel will automatically be 3-d in these libraries. The name `Conv2D` is to reflect the fact that the sliding of the kernel is done on a 2-d grid. Whether the kernel is a 3-d cube or a 2-d square, the sliding is done horizontally and vertically, not in the third dimension. Hence the output, or activation map, of a single kernel is always a 2-d grid. By having many kernels, we can concatenate more activation maps.

I hope this helps.

Dr. Phil · Learner · 3 years ago

✓ 1
Answer

Question about the initial PCA task

I have my high-level features (1280 of them) and do the PCA analysis with n_components=None. It works, no warnings are given but I notice when checking the explained variances that I only have 280 components rather then the expected 1280. As a consequence the scree plot is also only taking 280 components. Now 280 is the number of images in the train set, so I got confused and thought I might have used a transposed input matrix, but no. Then I discovered that setting the n_components=None actually is not taking all features, but the min(n_samples, n_features) I guess due to the underlying maths that need at least as many samples as features. So I'm stuck and quite surprised no-one asked this question yet.
I could create a few copies of the samples so that to have more rows than features, but not sure if this would be mathematically sound concerning the final result…

POST ANSWER

Mohamad Dia · Teacher · 3 years ago

✓ 2

Nice catch! Sure, this is indeed the case. The number of components can not be more than `min(n_samples, n_features)` . Mathematically speaking, the components are obtained by computing the eigenvectors and eigenvalues of the data covariance matrix XX^T (usually by running the SVD algorithm). The covariance matrix has `n_features` eigenvalues. If `n_samples < n_features` as in this case, the covariance matrix will only have `n_samples` non-zero eigenvalues while the other ones will be null. Therefore, there will be only `n_samples` components that make sense in this case. Hence, you only need to consider principle components up to 280.

Repeating the data is a nice hack to make the algorithm work beyond 280. However, this won't help the analysis since all the explained variances beyond 280 components will be null. The cumulative sum will saturate at the 280th component.

Terms of Use Privacy Policy FAQs

EPFL