Develop Your First Neural Network in Python With Keras Step-By-Step

by **Jason Brownlee** on May 24, 2016 in **Deep Learning**

Keras is a powerful easy-to-use Python library for developing and evaluating deep learning models.

It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in a few short lines of code.

In this post, you will discover how to create your first neural network model in Python using Keras.

Let's get started.

- **Update Feb/2017**: Updated prediction example so rounding works in Python 2 and Python 3.
- **Update Mar/2017**: Updated example for Keras 2.0.2, TensorFlow 1.0.1 and Theano 0.9.0.

Develop Your First Neural Network in Python With Keras Step-By-Step
Photo by Phil Whitehouse, some rights reserved.

Tutorial Overview

There is not a lot of code required, but we are going to step over it slowly so that you will know how to create your own models in the future.

The steps you are going to cover in this tutorial are as follows:

1. Load Data.
2. Define Model.
3. Compile Model.
4. Fit Model.
5. Evaluate Model.
6. Tie It All Together.

This tutorial has a few requirements:

1. You have Python 2 or 3 installed and configured.
2. You have SciPy (including NumPy) installed and configured.
3. You have Keras and a backend (Theano or TensorFlow) installed and configured.

If you need help with your environment, see the tutorial:

- How to Setup a Python Environment for Machine Learning and Deep Learning with Anaconda

Create a new file called **keras_first_network.py** and type or copy-and-paste the code into the file as you go.

1. Load Data

Whenever we work with machine learning algorithms that use a stochastic process (e.g. random numbers), it is a good idea to set the random number seed.

This is so that you can run the same code again and again and get the same result. This is useful if you need to demonstrate a result, compare algorithms using the same source of randomness or to debug a part of your code.

You can initialize the random number generator with any seed you like, for example:

```
1 from keras.models import Sequential
2 from keras.layers import Dense
3 import numpy
4 # fix random seed for reproducibility
5 numpy.random.seed(7)
```

Now we can load our data.

In this tutorial, we are going to use the Pima Indians onset of diabetes dataset. This is a standard machine learning dataset from the UCI Machine Learning repository. It describes patient medical record data for Pima Indians and whether they had an onset of diabetes within five years.

As such, it is a binary classification problem (onset of diabetes as 1 or not as 0). All of the input variables that describe each patient are numerical. This makes it easy to use directly with neural networks that expect numerical input and output values, and ideal for our first neural network in Keras.

Download the Pima Indian dataset from the UCI Machine Learning repository and place it in your local working directory, the same as your python file. Save it with the file name:

```
pima-indians-diabetes.csv
```

You can now load the file directly using the NumPy function **loadtxt()**. There are eight input variables and one output variable (the last column). Once loaded we can split the dataset into input variables (X) and the output class variable (Y).

```
# load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
```

We have initialized our random number generator to ensure our results are reproducible and loaded our data. We are now ready to define our neural network model.

2. Define Model

Models in Keras are defined as a sequence of layers.

We create a Sequential model and add layers one at a time until we are happy with our network topology.

The first thing to get right is to ensure the input layer has the right number of inputs. This can be specified when creating the first layer with the **input_dim** argument and setting it to 8 for the 8 input variables.

How do we know the number of layers and their types?

This is a very hard question. There are heuristics that we can use and often the best network structure is found through a process of trial and error experimentation. Generally, you need a network large enough to capture the structure of the problem if that helps at all.

In this example, we will use a fully-connected network structure with three layers.

Fully connected layers are defined using the Dense class. We can specify the number of neurons in the layer as the first argument, the initialization method as the second argument as **init** and specify the activation function using the **activation** argument.

In this case, we initialize the network weights to a small random number generated from a uniform distribution ('**uniform**'), in this case between 0 and 0.05 because that is the default uniform weight initialization in Keras. Another traditional alternative would be '**normal'** for small random numbers

generated from a Gaussian distribution.

We will use the rectifier ('**relu**') activation function on the first two layers and the sigmoid function in the output layer. It used to be the case that sigmoid and tanh activation functions were preferred for all layers. These days, better performance is achieved using the rectifier activation function. We use a sigmoid on the output layer to ensure our network output is between 0 and 1 and easy to map to either a probability of class 1 or snap to a hard classification of either class with a default threshold of 0.5.

We can piece it all together by adding each layer. The first layer has 12 neurons and expects 8 input variables. The second hidden layer has 8 neurons and finally, the output layer has 1 neuron to predict the class (onset of diabetes or not).

```
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

3. Compile Model

Now that the model is defined, we can compile it.

Compiling the model uses the efficient numerical libraries under the covers (the so-called backend) such as Theano or TensorFlow. The backend automatically chooses the best way to represent the network for training and making predictions to run on your hardware, such as CPU or GPU or even distributed.

When compiling, we must specify some additional properties required when training the network. Remember training a network means finding the best set of weights to make predictions for this problem.

We must specify the loss function to use to evaluate a set of weights, the optimizer used to search through different weights for the network and any optional metrics we would like to collect and report during training.

In this case, we will use logarithmic loss, which for a binary classification problem is defined in Keras as "**binary_crossentropy**". We will also use the efficient gradient descent algorithm "**adam**" for no other reason that it is an efficient default. Learn more about the Adam optimization algorithm in the paper "Adam: A Method for Stochastic Optimization".

Finally, because it is a classification problem, we will collect and report the classification accuracy as the metric.

```
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

4. Fit Model

We have defined our model and compiled it ready for efficient computation.

Now it is time to execute the model on some data.

We can train or fit our model on our loaded data by calling the **fit()** function on the model.

The training process will run for a fixed number of iterations through the dataset called epochs, that we must specify using the **nepochs** argument. We can also set the number of instances that are evaluated before a weight update in the network is performed, called the batch size and set using the **batch_size** argument.

For this problem, we will run for a small number of iterations (150) and use a relatively small batch size of 10. Again, these can be chosen experimentally by trial and error.

```
# Fit the model
model.fit(X, Y, epochs=150, batch_size=10)
```

This is where the work happens on your CPU or GPU.

5. Evaluate Model

We have trained our neural network on the entire dataset and we can evaluate the performance of the network on the same dataset.

This will only give us an idea of how well we have modeled the dataset (e.g. train accuracy), but no idea of how well the algorithm might perform on new data. We have done this for simplicity, but ideally, you could separate your data into train and test datasets for training and evaluation of your model.

You can evaluate your model on your training dataset using the **evaluate()** function on your model and pass it the same input and output used to train the model.

This will generate a prediction for each input and output pair and collect scores, including the average loss and any metrics you have configured, such as accuracy.

```
# evaluate the model
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

6. Tie It All Together

You have just seen how you can easily create your first neural network model in Keras.

Let's tie it all together into a complete code example.

```
# Create your first MLP in Keras
from keras.models import Sequential
from keras.layers import Dense
import numpy
# fix random seed for reproducibility
numpy.random.seed(7)
# load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, epochs=150, batch_size=10)
# evaluate the model
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

Running this example, you should see a message for each of the 150 epochs printing the loss and accuracy for each, followed by the final evaluation of the trained model on the training dataset.

It takes about 10 seconds to execute on my workstation running on the CPU with a Theano backend.

```
...
Epoch 145/150
768/768 [==============================] - 0s - loss: 0.5105 - acc: 0.7396
Epoch 146/150
768/768 [==============================] - 0s - loss: 0.4900 - acc: 0.7591
Epoch 147/150
768/768 [==============================] - 0s - loss: 0.4939 - acc: 0.7565
Epoch 148/150
768/768 [==============================] - 0s - loss: 0.4766 - acc: 0.7773
Epoch 149/150
768/768 [==============================] - 0s - loss: 0.4883 - acc: 0.7591
Epoch 150/150
768/768 [==============================] - 0s - loss: 0.4827 - acc: 0.7656
 32/768 [>.............................] - ETA: 0s
acc: 78.26%
```

Note: If you try running this example in an IPython or Jupyter notebook you may get an error. The reason is the output progress bars during training. You can easily turn these off by setting **verbose=0** in the call to **model.fit()**.

7. Bonus: Make Predictions

The number one question I get asked is:

> *After I train my model, how can I use it to make predictions on new data?*

Great question.

We can adapt the above example and use it to generate predictions on the training dataset, pretending it is a new dataset we have not seen before.

Making predictions is as easy as calling **model.predict()**. We are using a sigmoid activation function on the output layer, so the predictions will be in the range between 0 and 1. We can easily convert them into a crisp binary prediction for this classification task by rounding them.

The complete example that makes predictions for each record in the training data is listed below.

```
# Create first network with Keras
from keras.models import Sequential
from keras.layers import Dense
import numpy
# fix random seed for reproducibility
seed = 7
numpy.random.seed(seed)
# load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))
model.add(Dense(8, init='uniform', activation='relu'))
model.add(Dense(1, init='uniform', activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, nb_epoch=150, batch_size=10,  verbose=2)
# calculate predictions
predictions = model.predict(X)
# round predictions
rounded = [round(x[0]) for x in predictions]
print(rounded)
```

Running this modified example now prints the predictions for each input pattern. We could use these predictions directly in our application if needed.

```
[1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 1.0, 0.0, 0.
```

Summary

In this post, you discovered how to create your first neural network model using the powerful Keras Python library for deep learning.

Specifically, you learned the five key steps in using Keras to create a neural network or deep learning model, step-by-step including:

1. How to load data.
2. How to define neural network in Keras.
3. How to compile a Keras model using the efficient numerical backend.
4. How to train a model on data.
5. How to evaluate a model on data.

Do you have any questions about Keras or about this tutorial?
Ask your question in the comments and I will do my best to answer.

Related Tutorials

Are you looking for some more Deep Learning tutorials with Python and Keras?

Take a look at some of these:

- 5 Step Life-Cycle for Neural Network Models in Keras
- How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras
- Time Series Prediction With Deep Learning in Keras
- Multi-Class Classification Tutorial with the Keras Deep Learning Library
- Regression Tutorial with the Keras Deep Learning Library in Python

About Jason Brownlee

Dr. Jason Brownlee is a husband, proud father, academic researcher, author, professional developer and a machine learning practitioner. He is dedicated to helping developers get started and get good at applied machine learning. Learn more.

View all posts by Jason Brownlee →

‹ Evaluate the Performance of Machine Learning Algorithms in Python using Resampling

Metrics To Evaluate Machine Learning Algorithms in Python ›

227 Responses to *Develop Your First Neural Network in Python With Keras Step-By-Step*

Saurav May 27, 2016 at 11:08 pm #

REPLY

The input layer doesn't have any activation function, but still activation="relu" is mentioned in the first layer of the model. Why?

Jason Brownlee May 28, 2016 at 6:32 am #

REPLY

Hi Saurav,

The first layer in the network here is technically a hidden layer, hence it has an activation function.

sam Johnson December 21, 2016 at 2:44 am #

REPLY

Why have you made it a hidden layer though? the input layer is not usually represented as a hidden layer?

Jason Brownlee December 21, 2016 at 8:41 am #

REPLY

Hi sam,

Note this line:

```
1 model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))
```

It does a few things.

It defines the input layer as having 8 inputs.

It defines a hidden layer with 12 neurons, connected to the input layer that use relu activation function.

It initializes all weights using a sample of uniform random numbers.

Does that help?

Geoff May 29, 2016 at 6:18 am #

REPLY

Can you explain how to implement weight regularization into the layers?

Jason Brownlee June 15, 2016 at 5:50 am #

REPLY

Yep, see here:

http://keras.io/regularizers/

KWC June 14, 2016 at 12:08 pm #

REPLY

Import statements if others need them:

from keras.models import Sequential

from keras.layers import Dense, Activation

Jason Brownlee June 15, 2016 at 5:49 am #

REPLY

Thanks.

I had them in Part 6, but I have also added them to Part 1.

Aakash Nain June 29, 2016 at 6:00 pm #

REPLY

If there are 8 inputs for the first layer then why we have taken them as ’12’ in the following line :

model.add(Dense(12, input_dim=8, init=’uniform’, activation=’relu’))

Jason Brownlee June 30, 2016 at 6:47 am #

REPLY

Hi Aakash.

The input layer is defined by the input_dim parameter, here set to 8.

The first hidden layer has 12 neurons.

Joshua July 2, 2016 at 12:04 am #

REPLY

I ran your program and i have an error:
ValueError: could not convert string to float:
what could be the reason for this, and how may I solve it.
thanks.
great post by the way.

Jason Brownlee July 2, 2016 at 6:20 am #

REPLY

It might be a copy-paste error. Perhaps try to copy and run the whole example listed in section 6?

cheikh brahim July 5, 2016 at 7:40 pm #

REPLY

thank you for your simple and useful example.

Jason Brownlee July 6, 2016 at 6:22 am #

REPLY

You're welcome cheikh.

Nikhil Thakur July 6, 2016 at 6:39 pm #

REPLY

Hello Sir, I am trying to use Keras for NLP , specifically sentence classification. I have given the model building part below. It's taking quite a lot time to execute. I am using Pycharm IDE.

batch_size = 32
nb_filter = 250

```
filter_length = 3
nb_epoch = 2
pool_length = 2
output_dim = 5
hidden_dims = 250

# Build the model

model1 = Sequential()

model1.add(Convolution1D(nb_filter, filter_length ,activation='relu',border_mode='valid',
input_shape=(len(embb_weights),dim), weights=[embb_weights]))

model1.add(Dense(hidden_dims))
model1.add(Dropout(0.2))
model1.add(Activation('relu'))

model1.add(MaxPooling1D(pool_length=pool_length))

model1.add(Dense(output_dim, activation='sigmoid'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)

model1.compile(loss='mean_squared_error',
optimizer=sgd,
metrics=['accuracy'])
```

Jason Brownlee July 7, 2016 at 7:31 am #

REPLY

You may want a larger network. You may also want to use a standard repeating structure like CNN->CNN->Pool->Dense.

See this post on using a CNN:
http://machinelearningmastery.com/handwritten-digit-recognition-using-convolutional-neural-networks-python-keras/

Later, you may also want to try some stacked LSTMs.

Andre Norman July 15, 2016 at 10:40 am #

REPLY

Hi Jason, thanks for the awesome example. Given that the accuracy of this model is 79.56%. From here on, what steps would you take to improve the accuracy?

Given my nascent understanding of Machine Learning, my initial approach would have been:

Implement forward propagation, then compute the cost function, then implement back propagation, use

gradient checking to evaluate my network (disable after use), then use gradient descent.

However, this approach seems arduous compared to using Keras. Thanks for your response.

Jason Brownlee July 15, 2016 at 10:52 am #

REPLY

Hi Andre, indeed Keras makes working with neural nets so much easier. Fun even!

We may be maxing out on this problem, but here is some general advice for lifting performance.
– data prep – try lots of different views of the problem and see which is best at exposing the structure of the problem to the learning algorithm (data transforms, feature engineering, etc.)
– algorithm selection – try lots of algorithms and see which one or few are best on the problem (try on all views)
– algorithm tuning – tune well performing algorithms to get the most out of them (grid search or random search hyperparameter tuning)
– ensembles – combine predictions from multiple algorithms (stacking, boosting, bagging, etc.)

For neural nets, there are a lot of things to tune, I think there are big gains in trying different network topologies (layers and number of neurons per layer) in concert with training epochs and learning rate (bigger nets need more training).

I hope that helps as a start.

Andre Norman July 18, 2016 at 7:19 am #

REPLY

Awesome! Thanks Jason =)

Jason Brownlee July 18, 2016 at 8:03 am #

REPLY

You're welcome Andre.

Romilly Cocking July 21, 2016 at 12:31 am #

REPLY

Hi Jason, it's a great example but if anyone runs it in an IPython/Jupyter notebook they are likely to encounter an I/O error when running the fit step. This is due to a known bug in IPython.

The solution is to set verbose=0 like this

```
# Fit the model
model.fit(X, Y, nb_epoch=40, batch_size=10, verbose=0)
```

Jason Brownlee July 21, 2016 at 5:36 am #

REPLY

Great, thanks for sharing Romilly.

Anirban July 23, 2016 at 10:20 pm #

REPLY

Great example. Have a query though. How do I now give a input and get the output (0 or 1). Can you pls give the cmd for that.
Thanks

Jason Brownlee July 24, 2016 at 6:53 am #

REPLY

You can call model.predict() to get predictions and round on each value to snap to a binary value.

For example, below is a complete example showing you how to round the predictions and print them to console.

```
# Create first network with Keras
from keras.models import Sequential
from keras.layers import Dense
import numpy
# fix random seed for reproducibility
seed = 7
numpy.random.seed(seed)
# load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))
model.add(Dense(8, init='uniform', activation='relu'))
model.add(Dense(1, init='uniform', activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, nb_epoch=150, batch_size=10,  verbose=2)
# calculate predictions
predictions = model.predict(X)
# round predictions
rounded = [round(x) for x in predictions]
print(rounded)
```

Debanjan March 27, 2017 at 12:04 pm #

REPLY

Hi, Why you are not using any test set? You are predicting from the training set , I think.

Jason Brownlee March 28, 2017 at 8:19 am #

REPLY

Correct, it is just an example to get you started with Keras.

Anirban July 23, 2016 at 10:52 pm #

REPLY

I am not able to get to the last epoch. Getting error before that:
Epoch 11/150
390/768 [==============>……]Traceback (most recent call last):.6921

ValueError: I/O operation on closed file

I could resolve this by varying the epoch and batch size.

Now to predict a unknown value, i loaded a new dataset and used predict cmd as below :
dataset_test = numpy.loadtxt(“pima-indians-diabetes_test.csv”,delimiter=”,”) –has only one row

X = dataset_test[:,0:8]
model.predict(X)

But I am getting error :
X = dataset_test[:,0:8]

IndexError: too many indices for array

Can you help pls.

Thanks

Jason Brownlee July 24, 2016 at 6:55 am #

REPLY

I see problems like this when you run from a notebook or from an IDE.

Consider running examples from the console to ensure they work.

Consider tuning off verbose output (verbose=0 in the call to fit()) to disable the progress bar.

David Kluszczynski July 28, 2016 at 12:42 am #

REPLY

Hi Jason!
Loved the tutorial! I have a question however.
Is there a way to save the weights to a file after the model is trained for uses, such as kaggle?
Thanks,
David

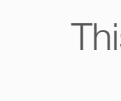
Jason Brownlee July 28, 2016 at 5:47 am #

REPLY

Thanks David.

You can save the network weights to file by calling model.save_weights("model.h5")

You can learn more in this post:
http://machinelearningmastery.com/save-load-keras-deep-learning-models/



Alex Hopper July 29, 2016 at 5:45 am #

REPLY

Hey, Jason! Thank you for the awesome tutorial! I've use your tutorial to learn about CNN. I have one question for you… Supposing I want to use Keras to classicate images and I have 3 or more classes to classify, How could my algorithm know about this classes? You know, I have to code what is a cat, a dog and a horse. Is there any way to code this? I've tried it:

target_names = ['class 0(Cats)', 'class 1(Dogs)', 'class 2(Horse)']
print(classification_report(np.argmax(Y_test,axis=1), y_pred,target_names=target_names))

But my results are not classifying correctly.

precision recall f1-score support
class 0(Cat) 0.00 0.00 0.00 17
class 1(Dog) 0.00 0.00 0.00 14
class 2(Horse) 0.99 1.00 0.99 2526

avg / total 0.98 0.99 0.98 2557

Jason Brownlee July 29, 2016 at 6:41 am #

REPLY

Great question Alex.

This is an example of a multi-class classification problem. You must use a one hot encoding on the

output variable to be able to model it with a neural network and specify the number of classes as the number of outputs on the final layer of your network.

I provide a tutorial with the famous iris dataset that has 3 output classes here: http://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/

Alex Hopper August 1, 2016 at 1:22 am #

REPLY

Thank you.
I'll check it.

Jason Brownlee August 1, 2016 at 6:25 am #

REPLY

No problem Alex.

Anonymouse August 2, 2016 at 11:28 pm #

REPLY

This was really useful, thank you

I'm using keras (with CNNs) for sentiment classification of documents and I'd like to improve the performance, but I'm completely at a loss when it comes to tuning the parameters in a non-arbitrary way. Could you maybe point me somewhere that will help me go about this in a more systematic fashion? There must be some heuristics or rules-of-thumb that could guide me.

Jason Brownlee August 3, 2016 at 8:09 am #

REPLY

I have a tutorial coming out soon (next week) that provide lots of examples of tuning the hyperparameters of a neural network in Keras, but limited to MLPs.

For CNNs, I would advise tuning the number of repeating layers (conv + max pool), the number of filters in repeating block, and the number and size of dense layers at the predicting part of your network. Also consider using some fixed layers from pre-trained models as the start of your network (e.g. VGG) and try just training some input and output layers around it for your problem.

I hope that helps as a start.

Shopon August 14, 2016 at 5:04 pm #

REPLY

Hello Jason , My Accuracy is : 0.0104 , but yours is 0.7879 and my loss is : -9.5414 . Is there any problem with the dataset ? I downloaded the dataset from a different site .

REPLY

Jason Brownlee August 15, 2016 at 12:36 pm #

I think there might be something wrong with your implementation or your dataset. Your numbers are way out.

REPLY

mohamed August 15, 2016 at 9:30 am #

after training, how i can use the trained model on new sample

REPLY

Jason Brownlee August 15, 2016 at 12:36 pm #

You can call model.predict()

See an above comment for a specific code example.

REPLY

Omachi Okolo August 16, 2016 at 10:21 pm #

Hi Jason,
i'm a student conducting a research on how to use artificial neural network to predict the business viability of potential software projects.
I intend to use python as a programming language. The application of ANN fascinates me but i'm new to machine learning and python. Can you help suggest how to go about this.
Many thanks

REPLY

Jason Brownlee August 17, 2016 at 9:51 am #

Consider getting a good grounding in how to work through a machine learning problem end to end in python first.

Here is a good tutorial to get you started:
http://machinelearningmastery.com/machine-learning-in-python-step-by-step/

Agni August 17, 2016 at 6:23 am #

REPLY

Dear Jeson, this is a great tutorial for beginners. It will satisfy the need of many students who are looking for the initial help. But I have a question. Could you please light on a few things: i) how to test the trained model using test dataset (i.e., loading of test dataset and applied the model and suppose the test file name is test.csv) ii) print the accuracy obtained on test dataset iii) the o/p has more than 2 class (suppose 4-class classification problem).
Please show the whole program to overcome any confusion.
Thanks a lot.

Jason Brownlee August 17, 2016 at 10:03 am #

REPLY

I provide an example elsewhere in the comments, you can also see how to make predictions on new data in this post:
http://machinelearningmastery.com/5-step-life-cycle-neural-network-models-keras/

For an example of multi-class classification, you can see this tutorial:
http://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/

Doron Vetlzer August 17, 2016 at 9:29 am #

REPLY

I am trying to build a Neural Network with some recursive connections but not a full recursive layer, how do I do this in Keras?

Doron Vetlzer August 17, 2016 at 9:31 am #

REPLY

I could print a diagram of the network but what I want Basically is that each neuron in the current time frame to know only its own previous output and not the output of all the neurons in the output layer.

Jason Brownlee August 17, 2016 at 10:04 am #

REPLY

I don’t know off hand Doron.

Doron Veltzer August 23, 2016 at 2:28 am #

REPLY

Thanks for replying though, have a good day.

sairam August 30, 2016 at 8:49 am #

REPLY

Hello Jason,

This is a great tutorial . Thanks for sharing.

I am having a dataset of 100 finger prints and i want to extract minutiae of 100 finger prints using python (Keras). Can you please advise where to start? I am really confused.

Jason Brownlee August 31, 2016 at 8:43 am #

REPLY

If your fingerprints are images, you may want to consider using convolutional neural networks (CNNs) that are much better at working image data.

See this tutorial on digit recognition for a start:
http://machinelearningmastery.com/handwritten-digit-recognition-using-convolutional-neural-networks-python-keras/

CM September 1, 2016 at 4:23 pm #

REPLY

Hi Jason,

Thanks for the great article. But I had 1 query.

Are there any inbuilt functions in keras that can give me the feature importance for the ANN model?

If not, can you suggest a technique I can use to extract variable importance from the loss function? I am considering an approach similar to that used in RF which involves permuting the values of the selected variable and calculating the relative increase in loss.

Regards,
CM

Jason Brownlee September 2, 2016 at 8:07 am #

REPLY

I don’t believe so CM.

I would suggest using a wrapper method and evaluate subsets of features to develop a feature importance/feature selection report.

I talk a lot more about feature selection in this post:
http://machinelearningmastery.com/an-introduction-to-feature-selection/

I provide an example of feature selection in scikit-learn here:
http://machinelearningmastery.com/feature-selection-machine-learning-python/

I hope that helps as a start.

Kamal September 7, 2016 at 2:09 am #

REPLY

Dear Jason, I am new to Deep learning. Being a novice, I am asking you a technical question which may seem silly. My question is that- can we use features (for example length of the sentence etc.) of a sentence while classifying a sentence (suppose the o/p are +ve sentence and -ve sentence) using deep neural network?

Jason Brownlee September 7, 2016 at 10:27 am #

REPLY

Great question Kamal, yes you can. I would encourage you to include all such features and see which give you a bump in performance.

Saurabh September 11, 2016 at 12:42 pm #

REPLY

Hi, How would I use this on a dataset that has multiple outputs? For example a dataset with output A and B where A could be 0 or 1 and B could be 3 or 4 ?

Jason Brownlee September 12, 2016 at 8:30 am #

REPLY

You could use two neurons in the output layer and normalize the output variables to both be in the range of 0 to 1.

This tutorial on multi-class classification might give you some ideas:
http://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/

Tom_P September 17, 2016 at 1:47 pm #

REPLY

Hi Jason,

The tutorial looks really good but unfortunately I keep getting an error when importing Dense from keras.layers, I get the error : AttributeError: module ‘theano’ has no attribute ‘gof’
I have tried reinstalling Theano but it has not fixed the issue.

Best wishes
Tom

Jason Brownlee September 18, 2016 at 7:57 am #

REPLY

Hi Tom, sorry to hear that. I have not seen this problem before.

Have you searched google? I can see a few posts and it might be related to your version of scipy or similar.

Let me know how you go.

shudhan September 21, 2016 at 5:54 pm #

REPLY

Hey Jason,

Can you please make a tutorial on how to add additional train data into the already trained model? This will be helpful for the bigger data sets. I read that warm start is used for random forest. But not sure how to implement as algorithm. A generalised version of how to implement would be good. Thank You!

Jason Brownlee September 22, 2016 at 8:08 am #

REPLY

Great question Shudhan!

Yes, you could save your weights, load them later into a new network topology and start training on new data again.

I’ll work out an example in coming weeks, time permitting.

Joanna September 22, 2016 at 1:09 am #

REPLY

Hi Jason,
first of all congratulations for this amazing work that you have done!
Here is my question:
What about if my .csv file includes also both nominal and numerical attributes?
Should I change my nominal values to numerical?

Thank you in advance

Jason Brownlee September 22, 2016 at 8:19 am #

REPLY

Hi Joanna, yes.

You can use a label encoder to convert nominal to integer, and then even convert the integer to one hot encoding.

This post will give you code you can use:
http://machinelearningmastery.com/data-preparation-gradient-boosting-xgboost-python/

ATM October 2, 2016 at 5:47 am #

REPLY

A small bug:-
Line 25 : rounded = [round(x) for x in predictions]

should have numpy.round instead, for the code to run!
Great tutorial, regardless. The best i've seen for intro to ANN in python. Thanks!

Jason Brownlee October 2, 2016 at 8:20 am #

REPLY

Perhaps it's your version of Python or environment?

In Python 2.7 the round() function is built-in.

AC January 14, 2017 at 2:11 am #

REPLY

If there is comment for python3, should be better.
#use unmpy.round instead, if using python3,

Jason Brownlee January 15, 2017 at 5:24 am #

REPLY

Thanks for the note AC.

Ash October 9, 2016 at 1:36 am #

REPLY

This is simple to grasp! Great post! How can we perform dropout in keras?

Jason Brownlee October 9, 2016 at 6:49 am #

REPLY

Thanks Ash.

You can learn about drop out with Keras here:
http://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/

Homagni Saha October 14, 2016 at 4:15 am #

REPLY

Hello Jason,
You are using model.predict in the end to predict the results. Is it possible to save the model somewhere in the harddisk and transfer it to another machine(turtlebot running on ROS for my instance) and then use the model directly on turtlebot to predict the results?
Please tell me how
Thanking you
Homagni Saha

Jason Brownlee October 14, 2016 at 9:07 am #

REPLY

Hi Homagni, great question.

Absolutely!

Learn exactly how in this tutorial I wrote:
http://machinelearningmastery.com/save-load-keras-deep-learning-models/

Rimi October 16, 2016 at 8:21 pm #

REPLY

Hi Jason,
I implemented you code to begin with. But I am getting an accuracy of 45.18% with the same parameters and everything.
Cant figure out why.
Thanks

Jason Brownlee October 17, 2016 at 10:29 am #

REPLY

There does sound like a problem there Rimi.

Confirm the code and data match exactly.

Ankit October 26, 2016 at 8:12 pm #

REPLY

Hi Jason,
I am little confused with first layer parameters. You said that first layer has 12 neurons and expects 8 input variables.

Why there is a difference between number of neurons, input_dim for first layer.

Regards,
Ankit

Jason Brownlee October 27, 2016 at 7:45 am #

REPLY

Hi Ankit,

The problem has 8 input variables and the first hidden layer has 12 neurons. Inputs are the columns of data, these are fixed. The Hidden layers in general are whatever we design based on whatever capacity we think we need to represent the complexity of the problem. In this case, we have chosen 12 neurons for the first hidden layer.

I hope that is clearer.

Tom October 27, 2016 at 3:04 am #

REPLY

Hi,
I have a data , IRIS like data but with more colmuns.
I want to use MLP and DBN/CNNClassifier (or any other Deep Learning classificaiton algorithm) on my data to see how correctly it does classified into 6 groups.

Previously using DEEP LEARNING FOR J, today first time see KERAS.
does KERAS has examples (code examples) of DL Classification algorithms?

Kindly,
Tom

Jason Brownlee October 27, 2016 at 7:48 am #

REPLY

Yes Tom, the example in this post is an example of a neural network (deep learning) applied to a classification problem.

Rumesa October 30, 2016 at 1:57 am #

REPLY

I have installed theano but it gives me the error of tensorflow.is it mendatory to install both packages? because tensorflow is not supported on wndows.the only way to get it on windows is to install virtual machine

Jason Brownlee October 30, 2016 at 8:57 am #

REPLY

Keras will work just fine with Theano.

Just install Theano, and configure Keras to use the Theano backend.

More information about configuring the Keras backend here:
http://machinelearningmastery.com/introduction-python-deep-learning-library-keras/

Rumesa October 31, 2016 at 4:36 am #

REPLY

hey jason I have run your code but got the following error.Although I have aready installed theano backend.help me out.I just stuck.

Using TensorFlow backend.
Traceback (most recent call last):
File “C:\Users\pc\Desktop\first.py”, line 2, in
from keras.models import Sequential
File “C:\Users\pc\Anaconda3\lib\site-packages\keras__init__.py”, line 2, in
from . import backend
File “C:\Users\pc\Anaconda3\lib\site-packages\keras\backend__init__.py”, line 64, in
from .tensorflow_backend import *
File “C:\Users\pc\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py”, line 1, in
import tensorflow as tf
ImportError: No module named ‘tensorflow’
>>>

Jason Brownlee October 31, 2016 at 5:34 am #

REPLY

Change the backend used by Keras from TensorFlow to Theano.

You can do this either by using the command line switch or changing the Keras config file.

See the link I posted in the previous post for instructions.

Maria January 6, 2017 at 1:05 pm #

REPLY

Hello Rumesa!
Have you solved your problem? I have the same one. Everywhere is the same answer with keras.json file or envirinment variable but it doesn't work. Can you tell me what have worked for you?

Jason Brownlee January 7, 2017 at 8:20 am #

REPLY

Interesting.

Maybe there is an issue with the latest version and a tight coupling to tensorflow? I have not seen this myself.

Perhaps it might be worth testing prior versions of Keras, such as 1.1.0?

Try this:

```
1 pip install --upgrade --no-deps keras==1.1.0
```

Alexon November 1, 2016 at 6:54 am #

REPLY

Hi Jason,

First off, thanks so much for creating these resources, I have been keeping an eye on your newsletter for a while now, and I finally have the free time to start learning more about it myself, so your work has been really appreciated.

My question is: How can I set/get the weights of each hidden node?

I am planning to create several arrays randomized weights, then use a genetic algorithm to see which weight array performs the best and improve over generations. How would be the best way to go about this, and if I use a "relu" activation function, am I right in thinking these randomly generated weights should be between 0 and 0.05?

Many thanks for your help 🙂
Alexon

Jason Brownlee November 1, 2016 at 8:05 am #

REPLY

Thanks Alexon,

You can get and set the weights from a network.

You can learn more about how to do this in the context of saving the weights to file here:
http://machinelearningmastery.com/save-load-keras-deep-learning-models/

I hope that helps as a start, I'd love to hear how you go.

Alexon November 6, 2016 at 6:36 am #

REPLY

Thats great, thanks for pointing me in the right direction.
I'd be happy to let you know how it goes, but might take a while as this is very much a "when I can find the time" project between jobs 🙂

Cheers!

Arnaldo Gunzi November 2, 2016 at 10:17 pm #

REPLY

Nice introduction, thanks!

Jason Brownlee November 3, 2016 at 7:59 am #

REPLY

I'm glad you found it useful Arnaldo.

Abbey November 14, 2016 at 11:05 pm #

REPLY

Good day

I have a question, how can I represent a character as a vector that could be an input for the neural network to predict the word meaning and trained using LSTM

For instance, I have bf to predict boy friend or best friend and similarly I have 2mor to predict tomorrow. I

need to encode all the input as a character represented as vector, so that it can be train with RNN/LSTM to predict the output.

Thank you.

Kind Regards

Jason Brownlee November 15, 2016 at 7:54 am #

REPLY

Hi Abbey, You can map characters to integers to get integer vectors.

Abbey November 15, 2016 at 6:17 pm #

REPLY

Thank you Jason, if i map characters to integers value to get vectors using English Alphabets, numbers and special characters

The question is how will LSTM predict the character. Please example in more details for me.

Regards

Jason Brownlee November 16, 2016 at 9:27 am #

REPLY

Hi Abbey,

If your output values are also characters, you can map them onto integers, and reverse the mapping to convert the predictions back to text.

Abbey November 16, 2016 at 8:39 pm #

The output value of the characters encoding will be text

Abbey November 15, 2016 at 6:22 pm #

REPLY

Thank you, Jason, if I map characters to integers value to get vectors representation of the informal text using English Alphabets, numbers and special characters

The question is how will LSTM predict the character or words that have close meaning to the input value. Please example in more details for me. I understand how RNN/LSTM work based on your

tutorial example but the logic in designing processing is what I am stress with.

Regards

Ammar November 27, 2016 at 10:35 am #

REPLY

hi Jason,
i am trying to implement CNN one dimention on my data. so, i bluit my network.
the issue is:
def train_model(model, X_train, y_train, X_test, y_test):
X_train = X_train.reshape(-1, 1, 41)
X_test = X_test.reshape(-1, 1, 41)

numpy.random.seed(seed)
model.fit(X_train, y_train, validation_data=(X_test, y_test), nb_epoch=100, batch_size=64)
# Final evaluation of the model
scores = model.evaluate(X_test, y_test, verbose=0)
print("Accuracy: %.2f%%" % (scores[1] * 100))
this method above does not work and does not give me any error message.
could you help me with this please?

Jason Brownlee November 28, 2016 at 8:40 am #

REPLY

Hi Ammar, I'm surprised that there is no error message.

Perhaps run from the command line and add some print() statements to see exactly where it stops.

KK November 28, 2016 at 6:55 pm #

REPLY

Hi Jason
Great work. I have another doubt. How can we apply this to text mining. I have a csv file containing review document and label. I want to apply classify the documents based on the text available. Can U do this favor.

Jason Brownlee November 29, 2016 at 8:48 am #

REPLY

I would recommend converting the chars to ints and then using an Embedding layer.

Alex M November 30, 2016 at 10:52 pm #

REPLY

Mr Jason, this is great tutorial but I am stack with some errors.

First I can't load data set correctly, tried to correct error but can't make it. (FileNotFoundError: [Errno 2] No such file or directory: 'pima-indians-diabetes.csv').

Second: While trying to evaluate the model it says (X is not defined) May be this is because uploading failed.

Thanks!

Jason Brownlee December 1, 2016 at 7:29 am #

REPLY

You need to download the file and place it in your current working directory Alex.

Does that help?

Alex M December 1, 2016 at 6:45 pm #

REPLY

Sir, it is now successful….

Thanks!

Jason Brownlee December 2, 2016 at 8:15 am #

REPLY

Glad to hear it Alex.

Bappaditya December 2, 2016 at 7:35 pm #

REPLY

Hi Jason,

First of all a special thanks to you for providing such a great tutorial. I am very new to machine learning and truly speaking i had no background in data science. The concept of ML overwhelmed me and now i have a desire to be an expert of this field. I need your advice to start from a scratch. Also i am a PhD student in Computer Engineering (computer hardware)and i want to apply it as a tool for fault detection and testing for ICs.Can you provide me some references on this field?

Jason Brownlee December 3, 2016 at 8:29 am #

REPLY

Hi Bappaditya,

My best advice for getting started is here:
http://machinelearningmastery.com/start-here/#getstarted

I believe machine learning and deep learning are good tools for use on problems in fault detection. A good place to find references is here http://scholar.google.com

Best of luck with your project.

Alex M December 3, 2016 at 8:00 pm #

REPLY

Well as usual in our daily coding life errors happen, now I have this error how can I correct it? Thanks!

” ———————————————————————

NoBackendError Traceback (most recent call last)
in ()
16 import librosa.display
17 audio_path = (‘/Users/MA/Python Notebook/OK.mp3’)
—> 18 y, sr = librosa.load(audio_path)

C:\Users\MA\Anaconda3\lib\site-packages\librosa\core\audio.py in load(path, sr, mono, offset, duration, dtype)
107
108 y = []
–> 109 with audioread.audio_open(os.path.realpath(path)) as input_file:
110 sr_native = input_file.samplerate
111 n_channels = input_file.channels

C:\Users\MA\Anaconda3\lib\site-packages\audioread__init__.py in audio_open(path)
112
113 # All backends failed!
–> 114 raise NoBackendError()

NoBackendError:

”

That is the error I am getting just when trying to load a song into librosa…
Thanks!! @Jason Brownlee

Jason Brownlee December 4, 2016 at 5:30 am #

REPLY

Sorry, this looks like an issue with your librosa library, not a machine learning issue. I can’t give

you expert advice, sorry.

Alex M December 4, 2016 at 10:30 pm #

REPLY

Thanks I have managed to correct the error…

Happy Sunday to you all……

Jason Brownlee December 5, 2016 at 6:49 am #

REPLY

Glad to hear it Alex.

Lei December 4, 2016 at 10:52 pm #

REPLY

Hi, Jason, thank you for your amazing examples.
I run the same code on my laptop. But I did not get the same results. What could be the possible reasons?
I am using windows 8.1 64bit+eclipse+anaconda 4.2+theano 0.9.4+CUDA7.5
I got results like follows.

… …
Epoch 145/150

10/768 [……] – ETA: 0s – loss: 0.3634 – acc: 0.8000
80/768 [==>……] – ETA: 0s – loss: 0.4066 – acc: 0.7750
150/768 [====>……] – ETA: 0s – loss: 0.4059 – acc: 0.8067
220/768 [=======>……] – ETA: 0s – loss: 0.4047 – acc: 0.8091
300/768 [==========>……] – ETA: 0s – loss: 0.4498 – acc: 0.7867
380/768 [============>……] – ETA: 0s – loss: 0.4595 – acc: 0.7895
450/768 [===============>……] – ETA: 0s – loss: 0.4568 – acc: 0.7911
510/768 [================>……..] – ETA: 0s – loss: 0.4553 – acc: 0.7882
580/768 [==================>……..] – ETA: 0s – loss: 0.4677 – acc: 0.7776
660/768 [=====================>…..] – ETA: 0s – loss: 0.4697 – acc: 0.7788
740/768 [========================>..] – ETA: 0s – loss: 0.4611 – acc: 0.7838
768/768 [==============================] – 0s – loss: 0.4614 – acc: 0.7799
Epoch 146/150

10/768 [……] – ETA: 0s – loss: 0.3846 – acc: 0.8000
90/768 [==>……] – ETA: 0s – loss: 0.5079 – acc: 0.7444
170/768 [=====>……] – ETA: 0s – loss: 0.4500 – acc: 0.7882
250/768 [========>……] – ETA: 0s – loss: 0.4594 – acc: 0.7840

330/768 [==========>……] – ETA: 0s – loss: 0.4574 – acc: 0.7818
400/768 [==========>……] – ETA: 0s – loss: 0.4563 – acc: 0.7775
470/768 [==========>……] – ETA: 0s – loss: 0.4654 – acc: 0.7723
540/768 [==========>……] – ETA: 0s – loss: 0.4537 – acc: 0.7870
620/768 [==========>……] – ETA: 0s – loss: 0.4615 – acc: 0.7806
690/768 [==========>….] – ETA: 0s – loss: 0.4631 – acc: 0.7739
750/768 [==========>.] – ETA: 0s – loss: 0.4649 – acc: 0.7733
768/768 [==========] – 0s – loss: 0.4636 – acc: 0.7734
Epoch 147/150

10/768 [……] – ETA: 0s – loss: 0.3561 – acc: 0.9000
90/768 [==>……] – ETA: 0s – loss: 0.4167 – acc: 0.8556
170/768 [=====>……] – ETA: 0s – loss: 0.4824 – acc: 0.8059
250/768 [========>……] – ETA: 0s – loss: 0.4534 – acc: 0.8080
330/768 [==========>……] – ETA: 0s – loss: 0.4679 – acc: 0.7848
400/768 [==========>……] – ETA: 0s – loss: 0.4590 – acc: 0.7950
460/768 [==========>……] – ETA: 0s – loss: 0.4619 – acc: 0.7913
530/768 [==========>……] – ETA: 0s – loss: 0.4562 – acc: 0.7868
600/768 [==========>……] – ETA: 0s – loss: 0.4497 – acc: 0.7883
680/768 [==========>….] – ETA: 0s – loss: 0.4525 – acc: 0.7853
760/768 [==========>.] – ETA: 0s – loss: 0.4568 – acc: 0.7803
768/768 [==========] – 0s – loss: 0.4561 – acc: 0.7812
Epoch 148/150

10/768 [……] – ETA: 0s – loss: 0.4183 – acc: 0.9000
80/768 [==>……] – ETA: 0s – loss: 0.3674 – acc: 0.8750
160/768 [=====>……] – ETA: 0s – loss: 0.4340 – acc: 0.8250
240/768 [========>……] – ETA: 0s – loss: 0.4799 – acc: 0.7583
320/768 [==========>……] – ETA: 0s – loss: 0.4648 – acc: 0.7719
400/768 [==========>……] – ETA: 0s – loss: 0.4596 – acc: 0.7775
470/768 [==========>……] – ETA: 0s – loss: 0.4475 – acc: 0.7809
540/768 [==========>……] – ETA: 0s – loss: 0.4545 – acc: 0.7778
620/768 [==========>……] – ETA: 0s – loss: 0.4590 – acc: 0.7742
690/768 [==========>….] – ETA: 0s – loss: 0.4769 – acc: 0.7652
760/768 [==========>.] – ETA: 0s – loss: 0.4748 – acc: 0.7658
768/768 [==========] – 0s – loss: 0.4734 – acc: 0.7669
Epoch 149/150

10/768 [……] – ETA: 0s – loss: 0.3043 – acc: 0.9000
90/768 [==>……] – ETA: 0s – loss: 0.4913 – acc: 0.7111
170/768 [=====>……] – ETA: 0s – loss: 0.4779 – acc: 0.7588
250/768 [========>……] – ETA: 0s – loss: 0.4794 – acc: 0.7640
320/768 [==========>……] – ETA: 0s – loss: 0.4957 – acc: 0.7562
370/768 [==========>……] – ETA: 0s – loss: 0.4891 – acc: 0.7703
450/768 [==========>……] – ETA: 0s – loss: 0.4737 – acc: 0.7867

520/768 [==========>……] – ETA: 0s – loss: 0.4675 – acc: 0.7865
600/768 [==========>……] – ETA: 0s – loss: 0.4668 – acc: 0.7833
680/768 [==========>….] – ETA: 0s – loss: 0.4677 – acc: 0.7809
760/768 [==========>.] – ETA: 0s – loss: 0.4648 – acc: 0.7803
768/768 [==========] – 0s – loss: 0.4625 – acc: 0.7826
Epoch 150/150

10/768 [……] – ETA: 0s – loss: 0.2751 – acc: 1.0000
100/768 [==>……] – ETA: 0s – loss: 0.4501 – acc: 0.8100
170/768 [=====>……] – ETA: 0s – loss: 0.4588 – acc: 0.8059
250/768 [========>……] – ETA: 0s – loss: 0.4299 – acc: 0.8200
310/768 [==========>……] – ETA: 0s – loss: 0.4298 – acc: 0.8129
380/768 [==========>……] – ETA: 0s – loss: 0.4365 – acc: 0.8053
460/768 [==========>……] – ETA: 0s – loss: 0.4469 – acc: 0.7957
540/768 [==========>……] – ETA: 0s – loss: 0.4436 – acc: 0.8000
620/768 [==========>……] – ETA: 0s – loss: 0.4570 – acc: 0.7871
690/768 [==========>….] – ETA: 0s – loss: 0.4664 – acc: 0.7783
760/768 [==========>.] – ETA: 0s – loss: 0.4617 – acc: 0.7789
768/768 [==========] – 0s – loss: 0.4638 – acc: 0.7773

32/768 [>……] – ETA: 0s
448/768 [==========>……] – ETA: 0sacc: 79.69%

Jason Brownlee December 5, 2016 at 6:50 am #

REPLY

There is randomness in the learning process that we cannot control for yet.

See this post:
http://machinelearningmastery.com/randomness-in-machine-learning/

Nanya December 10, 2016 at 2:55 pm #

REPLY

Hello Jason Brownlee,Thx for sharing~
I’m new in deep learning.And I am wondering can what you dicussed here:”Keras” be used to build a CNN in tensorflow and train some csv fiels for classification.May be this is a stupid question,but waiting for you reply.I’m working on my graduation project for Word sense disambiguation with cnn,and just can’t move on.Hope for your heip~Bese wishes!

Jason Brownlee December 11, 2016 at 5:22 am #

REPLY

Sorry Nanya, I'm not sure I understand your question. Are you able to rephrase it?

Anon December 16, 2016 at 12:51 am #

REPLY

I've just installed Anaconda with Keras and am using python 3.5.
It seems there's an error with the rounding using Py3 as opposed to Py2. I think it's because of this change:
https://github.com/numpy/numpy/issues/5700

I removed the rounding and just used print(predictions) and it seemed to work outputting floats instead.

Does this look correct?

…
Epoch 150/150
0s – loss: 0.4593 – acc: 0.7839
[[0.79361773]
[0.10443526]
[0.90862554]
…,
[0.33652252]
[0.63745886]
[0.11704451]]

Jason Brownlee December 16, 2016 at 5:44 am #

REPLY

Nice, it does look good!

Florin Claudiu Mihalache December 19, 2016 at 2:37 am #

REPLY

Hi Jason Brownlee
I tried to modified your exemple for my problem (Letter Recognition
,http://archive.ics.uci.edu/ml/datasets/Letter+Recognition).
My data set look like http://archive.ics.uci.edu/ml/machine-learning-databases/letter-recognition/letter-recognition.data (T,2,8,3,5,1,8,13,0,6,6,10,8,0,8,0,8) .I try to split the data in input and ouput like this :

X = dataset[:,1:17]
Y = dataset[:,0]
but a have some error (something related that strings are not recognized) .
I tried to modified each letter whit the ASCII code (A became 65 and so on).The string error disappeared.
The program compiles now but the output look like this :

17445/20000 [=========================>….] – ETA: 0s – loss: -1219.4768 – acc:0.0000e+00
17605/20000 [=========================>….] – ETA: 0s – loss: -1219.4706 – acc:0.0000e+00
17730/20000 [=========================>….] – ETA: 0s – loss: -1219.4566 – acc:0.0000e+00
17890/20000 [=========================>….] – ETA: 0s – loss: -1219.4071 – acc:0.0000e+00
18050/20000 [==========================>…] – ETA: 0s – loss: -1219.4599 – acc:0.0000e+00
18175/20000 [==========================>…] – ETA: 0s – loss: -1219.3972 – acc:0.0000e+00
18335/20000 [==========================>…] – ETA: 0s – loss: -1219.4642 – acc:0.0000e+00
18495/20000 [==========================>…] – ETA: 0s – loss: -1219.5032 – acc:0.0000e+00
18620/20000 [==========================>…] – ETA: 0s – loss: -1219.4391 – acc:0.0000e+00
18780/20000 [==========================>..] – ETA: 0s – loss: -1219.5652 – acc:0.0000e+00
18940/20000 [==========================>..] – ETA: 0s – loss: -1219.5520 – acc:0.0000e+00
19080/20000 [==========================>..] – ETA: 0s – loss: -1219.5381 – acc:0.0000e+00
19225/20000 [==========================>..] – ETA: 0s – loss: -1219.5182 – acc:0.0000e+00
19385/20000 [==========================>.] – ETA: 0s – loss: -1219.6742 – acc:0.0000e+00
19535/20000 [==========================>.] – ETA: 0s – loss: -1219.7030 – acc:0.0000e+00
19670/20000 [==========================>.] – ETA: 0s – loss: -1219.7634 – acc:0.0000e+00
19830/20000 [==========================>.] – ETA: 0s – loss: -1219.8336 – acc:0.0000e+00
19990/20000 [==========================>.] – ETA: 0s – loss: -1219.8532 – acc:0.0000e+00
20000/20000 [==============================] – 1s – loss: -1219.8594 – acc: 0.0000e+00
18880/20000 [==========================>..] – ETA: 0sacc: 0.00%

I do not understand why. Can you please help me

Anon December 26, 2016 at 6:44 am #

REPLY

What version of Python are you running?

karishma sharma December 22, 2016 at 10:03 am #

REPLY

Hi Jason,

Since the epoch is set to 150 and batch size is 10, does the training algorithm pick 10 training examples at random in each iteration, given that we had only 768 total in X. Or does it sample randomly after it has finished covering all.

Thanks

Jason Brownlee December 23, 2016 at 5:27 am #

REPLY

Good question,

It iterates over the dataset 150 times and within one epoch it works through 10 rows at a time before doing an update to the weights. The patterns are shuffled before each epoch.

I hope that helps.

Kaustuv January 9, 2017 at 4:57 am #

REPLY

Hi Jason
Thanks a lot for this blog. It really helps me to start learning deep learning which was in a planning state for last few months. Your simple enrich blogs are awsome. No questions from my side before completing all tutorials.
One question regarding availability of your book. How can I buy those books from India ?

Jason Brownlee January 9, 2017 at 7:53 am #

REPLY

All my books and training are digital, you can purchase them from here:
http://machinelearningmastery.com/products

Stephen Wilson January 15, 2017 at 4:00 pm #

REPLY

Hi Jason, firstly your work here is a fantastic resource and I am very thankful for the effort you put in.
I am a slightly-better-than-beginner at python and an absolute novice at ML, I wonder if you could help me classify my problem and find an angle to work at it from.

My data is thus:
Column Names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, Result
Values: 4, 4, 6, 6, 3, 2, 5, 5, 0, 0, 0, 0, 0, 0, 0, 4

I want to find the percentage chance of each Column Names category being the Result based off the configuration of all the values present from 1-15. Then if need be compare the configuration of Values with another row of values to find the same, Resulting in the total needed calculation as:

Column Names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, Result
Values: 4, 4, 6, 6, 3, 2, 5, 5, 0, 0, 0, 0, 0, 0, 0, 4
Values2: 7, 3, 5, 1, 4, 8, 6, 2, 9, 9, 9, 9, 9, 9, 9

I apologize if my explanation is not clear, and appreciate any help you can give me thank you.

Jason Brownlee January 16, 2017 at 10:39 am #

REPLY

Hi Stephen,

This process might help you work through your problem:
http://machinelearningmastery.com/start-here/#process

Specifically the first step in defining your problem.

Let me know how you go.

Rohit January 16, 2017 at 10:37 pm #

REPLY

Thanks Jason for such a nice and concise example.

Just wanted to ask if it is possible to save this model in a file and port it to may be an Android or iOS device? If so, what are the libraries available for the same?

Thanks

Rohit

Jason Brownlee January 17, 2017 at 7:38 am #

REPLY

Thanks Rohit,

Here's an example of saving a Keras model to file:
http://machinelearningmastery.com/save-load-keras-deep-learning-models/

I don't know about running Keras on an Android or iOS device. Let me know how you go.

Hsiang January 18, 2017 at 3:35 pm #

REPLY

Hi, Jason

Thank you for your blog! It is wonderful!

I used tensorflow as backend, and implemented the procedures using Jupyter.
I did "source activate tensorflow" -> "ipython notebook".
I can successfully use Keras and import tensorflow.

However, it seems that such environment doesn't support pandas and sklearn.
Do you have any way to incorporate pandas, sklearn and keras?

(I wish to use sklearn to revisit the classification problem and compare the accuracy with the deep learning method. But I also wish to put the works together in the same interface.)

Thanks!

Jason Brownlee January 19, 2017 at 7:24 am #

REPLY

Sorry, I do not use notebooks myself. I cannot offer you good advice.

Hsiang January 19, 2017 at 12:53 pm #

REPLY

Thanks, Jason!
Actually the problem is not on notebooks. Even I used the terminal mode, i.e. doing “source activate tensorflow” only. It failed to import sklearn. Does that mean tensorflow library is not compatible with sklearn? Thanks again!

Jason Brownlee January 20, 2017 at 10:17 am #

REPLY

Sorry Hsiang, I don’t have experience using sklearn and tensorflow with virtual environments.

Hsiang January 21, 2017 at 12:46 am #

Thank you!

Jason Brownlee January 21, 2017 at 10:34 am #

You’re welcome Hsiang.

keshav bansal January 24, 2017 at 12:45 am #

REPLY

hello sir,
A very informative post indeed . I know my question is a very trivial one but can you please show me how to predict on a explicitly mentioned data tuple say v=[6,148,72,35,0,33.6,0.627,50]

thanks for the tutorial anyway

Jason Brownlee January 24, 2017 at 11:04 am #

REPLY

Hi keshav,

You can make predictions by calling model.predict()

CATRINA WEBB January 25, 2017 at 9:06 am #

REPLY

When I rerun the file (without predictions) does it reset the model and weights?

Ericson January 30, 2017 at 8:04 pm #

REPLY

excuse me sir, i wanna ask you a question about this paragraph”dataset = numpy.loadtxt(“pima-indians-diabetes.csv”,delimiter=’,’)”, i used the mac and downloaded the dataset,then i exchanged the text into csv file. Running the program

,hen i got:{Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 12:39:47)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type “copyright”, “credits” or “license()” for more information.
>>>
========== RESTART: /Users/luowenbin/Documents/database_test.py ==========
Using TensorFlow backend.

Traceback (most recent call last):
File “/Users/luowenbin/Documents/database_test.py”, line 9, in
dataset = numpy.loadtxt(“pima-indians-diabetes.csv”,delimiter=’,’)
File “/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-packages/numpy/lib/npyio.py”, line 985, in loadtxt
items = [conv(val) for (conv, val) in zip(converters, vals)]
File “/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-packages/numpy/lib/npyio.py”, line 687, in floatconv
return float(x)
ValueError: could not convert string to float: book
>>> }
How can i solve this problem? give me a hand thank you!

Jason Brownlee February 1, 2017 at 10:22 am #

REPLY

Hi Ericson,

Confirm that the contents of “pima-indians-diabetes.csv” meet your expectation of a list of CSV lines.

Sukhpal February 7, 2017 at 9:00 pm #

REPLY

excuse me sir,when i run this code for my data set ,I encounter this problem…please help me finding solution to this problem
runfile(‘C:/Users/sukhpal/.spyder/temp.py’, wdir=’C:/Users/sukhpal/.spyder’)
Using TensorFlow backend.
Traceback (most recent call last):

File “”, line 1, in
runfile(‘C:/Users/sukhpal/.spyder/temp.py’, wdir=’C:/Users/sukhpal/.spyder’)

File “C:\Users\sukhpal\Anaconda2\lib\site-packages\spyder\utils\site\sitecustomize.py”, line 866, in runfile
execfile(filename, namespace)

File “C:\Users\sukhpal\Anaconda2\lib\site-packages\spyder\utils\site\sitecustomize.py”, line 87, in execfile
exec(compile(scripttext, filename, ‘exec’), glob, loc)

File “C:/Users/sukhpal/.spyder/temp.py”, line 1, in
from keras.models import Sequential

File “C:\Users\sukhpal\Anaconda2\lib\site-packages\keras__init__.py”, line 2, in
from . import backend

File “C:\Users\sukhpal\Anaconda2\lib\site-packages\keras\backend__init__.py”, line 67, in
from .tensorflow_backend import *

File “C:\Users\sukhpal\Anaconda2\lib\site-packages\keras\backend\tensorflow_backend.py”, line 1, in
import tensorflow as tf

ImportError: No module named tensorflow

Jason Brownlee February 8, 2017 at 9:34 am #

REPLY

This is a change with the most recent version of tensorflow, I will investigate and change the example.

For now, consider installing and using an older version of tensorflow.

Will February 14, 2017 at 5:33 am #

REPLY

Great tutorial! Amazing amount of work you've put in and great marketing skills (I also have an email list, ebooks and sequence, etc). I ran this in Jupyter notebook… I noticed the 144th epoch (acc .7982) had more accuracy than at 150. Why is that?

P.S. i did this for the print: print(numpy.round(predictions))
It seems to avoid a list of arrays which when printing includes the dtype (messy)

Jason Brownlee February 14, 2017 at 10:07 am #

REPLY

Thanks Will.

The model will fluctuate in performance while learning. You can configure triggered check points to save the model if/when conditions like a decrease in train/validation performance is detected. Here's an example:
http://machinelearningmastery.com/check-point-deep-learning-models-keras/

Sukhpal February 14, 2017 at 3:50 pm #

REPLY

Please help me to find out this error
runfile('C:/Users/sukhpal/.spyder/temp.py', wdir='C:/Users/sukhpal/.spyder')ERROR: execution aborted

Jason Brownlee February 15, 2017 at 11:32 am #

REPLY

I'm not sure Sukhpal.

Consider getting code working from the command line, I don't use IDEs myself.

Kamal February 14, 2017 at 5:15 pm #

REPLY

please help me to find this error find this error
Epoch 194/195
195/195 [==============================] – 0s – loss: 0.2692 – acc: 0.8667
Epoch 195/195
195/195 [==============================] – 0s – loss: 0.2586 – acc: 0.8667
195/195 [==============================] – 0s

Traceback (most recent call last):

Jason Brownlee February 15, 2017 at 11:32 am #

REPLY

What was the error exactly Kamal?

Kamal February 15, 2017 at 3:24 pm #

REPLY

sir when i run the code on my data set
then it doesnot show overall accuracy although it shows the accuracy and loss for the whole iterations

Jason Brownlee February 16, 2017 at 11:06 am #

REPLY

I'm not sure I understand your question Kamal, please you could restate it?

Val February 15, 2017 at 9:00 pm #

REPLY

Hi Jason, im just starting deep learning in python using keras and theano. I have followed the installation instructions without a hitch. Tested some examples but when i run this one line by line i get a lot of exceptions and errors once i run the "model.fit(X,Y, nb_epochs=150, batch_size=10"

Jason Brownlee February 16, 2017 at 11:06 am #

REPLY

What errors are you getting?

CrisH February 17, 2017 at 8:12 pm #

REPLY

Hi, how do I know what number to use for random.seed() ? I mean you use 7, is there any reason for that? Also is it enough to use it only once, in the beginning of the code?

Jason Brownlee February 18, 2017 at 8:38 am #

REPLY

You can use any number CrisH. The fixed random seed makes the example reproducible.

You can learn more about randomness and random seeds in this post:
http://machinelearningmastery.com/randomness-in-machine-learning/

kk February 18, 2017 at 1:53 am #

REPLY

am new to deep learning and found this great tutorial. keep it up and look forward!!

Jason Brownlee February 18, 2017 at 8:41 am #

REPLY

Thanks!

Iqra Ameer February 21, 2017 at 5:20 am #

REPLY

HI, I have a problem in execution the above example as it. It seems that it's not running properly and stops at Using TensorFlow backend.

Epoch 147/150
768/768 [==============================] – 0s – loss: 0.4709 – acc: 0.7878
Epoch 148/150
768/768 [==============================] – 0s – loss: 0.4690 – acc: 0.7812
Epoch 149/150
768/768 [==============================] – 0s – loss: 0.4711 – acc: 0.7721
Epoch 150/150
768/768 [==============================] – 0s – loss: 0.4731 – acc: 0.7747
32/768 [>……..] – ETA: 0sacc: 76.43%

I am new in this field, could you please guide me about this error.
I also executed on another data set, it stops with the same behavior.

Jason Brownlee February 21, 2017 at 9:39 am #

REPLY

What is the error exactly? The example hangs?

Maybe try the Theano backend and see if that makes a difference. Also make sure all of your libraries are up to date.

Iqra Ameer February 22, 2017 at 5:47 am #

REPLY

Dear Jason,
Thank you so much for your valuable suggestions. I tried Theano backend and also updated all my libraries, but again it hanged at:

768/768 [==============================] – 0s – loss: 0.4656 – acc: 0.7799
Epoch 149/150
768/768 [==============================] – 0s – loss: 0.4589 – acc: 0.7826
Epoch 150/150
768/768 [==============================] – 0s – loss: 0.4611 – acc: 0.7773
32/768 [>……..] – ETA: 0sacc: 78.91%

Jason Brownlee February 22, 2017 at 10:05 am #

REPLY

I'm sorry to hear that, I have not seen this issue before.

Perhaps a RAM issue or a CPU overheating issue? Are you able to try different hardware?

frd March 8, 2017 at 2:50 am #

REPLY

Hi!

Were you able to find a solution for that?

I'm having exactly the same problem

(…)
Epoch 149/150
768/768 [==============================] – 0s – loss: 0.4593 – acc: 0.7773
Epoch 150/150
768/768 [==============================] – 0s – loss: 0.4586 – acc: 0.7891
32/768 [>……..] – ETA: 0sacc: 76.69%

Bhanu February 23, 2017 at 1:51 pm #

REPLY

Hello sir,
i want to ask wether we can convert this code to deep learning wid increasing number of layers..

Jason Brownlee February 24, 2017 at 10:12 am #

REPLY

Sure you can increase the number of layers, try it and see.

Ananya Mohapatra February 28, 2017 at 6:40 pm #

REPLY

hello sir,
could you please tell me how do i determine the no.of neurons in each layer, because i am using a different datset and am unable to know the no.of neurons in each layer

Jason Brownlee March 1, 2017 at 8:33 am #

REPLY

Hi Ananya, great question.

Sorry, there is no good theory on how to configure a neural net.

You can configure the number of neurons in a layer by trial and error. Also consider tuning the number of epochs and batch size at the same time.

Ananya Mohapatra March 1, 2017 at 4:42 pm #

REPLY

thank you so much sir. It worked ! 🙂

Jason Brownlee March 2, 2017 at 8:11 am #

REPLY

Glad to here it Ananya.

Jayant Sahewal February 28, 2017 at 8:11 pm #

REPLY

Hi Jason,

really helpful blog. I have a question about how much time does it take to converge?

I have a dataset with around 4000 records, 3 input columns and 1 output column. I came up with the following model

def create_model(dropout_rate=0.0, weight_constraint=0, learning_rate=0.001, activation='linear'):
# create model

```
model = Sequential()
model.add(Dense(6, input_dim=3, init='uniform', activation=activation,
W_constraint=maxnorm(weight_constraint)))
model.add(Dropout(dropout_rate))
model.add(Dense(1, init='uniform', activation='sigmoid'))
# Optimizer
optimizer = Adam(lr=learning_rate)
# Compile model
model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])
return model

# create model
model = KerasRegressor(build_fn=create_model, verbose=0)
# define the grid search parameters
batch_size = [10]
epochs = [100]
weight_constraint = [3]
dropout_rate = [0.9]
learning_rate = [0.01]
activation = ['linear']
param_grid = dict(batch_size=batch_size, nb_epoch=epochs, dropout_rate=dropout_rate, \
weight_constraint=weight_constraint, learning_rate=learning_rate, activation=activation)
grid = GridSearchCV(estimator=model, param_grid=param_grid, n_jobs=-1, cv=5)
grid_result = grid.fit(X_train, Y_train)
```

I have a 32 core machine with 64 GB RAM and it does not converge even in more than an hour. I can see all the cores busy, so it is using all the cores for training. However, if I change the input neurons to 3 then it converges in around 2 minutes.

Keras version: 1.1.1
Tensorflow version: 0.10.0rc0
theano version: 0.8.2.dev-901275534cbfe3fbbe290ce85d1abf8bb9a5b203

It's using Tensorflow backend. Can you help me understand what is going on or point me in the right direction? Do you think switching to theano will help?

Best,
Jayant

Jason Brownlee March 1, 2017 at 8:36 am #

REPLY

This post might help you tune your deep learning model:
http://machinelearningmastery.com/improve-deep-learning-performance/

I hope that helps as a start.

Animesh Mohanty March 1, 2017 at 9:21 pm #

REPLY

hello sir,
could you please tell me how can i plot the results of the code on a graph . I made a few adjustments to the code so as to run it on a different dataset.

Jason Brownlee March 2, 2017 at 8:16 am #

REPLY

What do you want to plot exactly Animesh?

Animesh Mohanty March 2, 2017 at 4:56 pm #

REPLY

Accuracy vs no.of neurons in the input layer and the no.of neurons in the hidden layer

param March 2, 2017 at 12:15 am #

REPLY

sir can u plz explain
the different attributes used in this statement
print(“%s: %.2f%%” % (model.metrics_names[1], scores[1]*100))

param March 2, 2017 at 12:16 am #

REPLY

precisely,what is model.metrics_names

Jason Brownlee March 2, 2017 at 8:22 am #

REPLY

model.metrics_names is a list of names of the metrics collected during training.

More details here:
https://keras.io/models/sequential/

Jason Brownlee March 2, 2017 at 8:20 am #

REPLY

Hi param,

It is using string formatting. %s formats a string, %.2f formats a floating point value with 2 decimal places, %% includes a percent symbol.

You can learn more about the print function here:
https://docs.python.org/3/library/functions.html#print

More info on string formatting here:
https://pyformat.info/

Vijin K P March 2, 2017 at 4:01 am #

REPLY

Hi Jason,

It was an awesome post. Could you please tell me how to we decide the following in a DNN 1. number of neurons in the hidden layers
2. number of hidden layers

Thanks.
Vijin

Jason Brownlee March 2, 2017 at 8:22 am #

REPLY

Great question Vijin.

Generally, trial and error. There are no good theories on how to configure a neural network.

Vijin K P March 3, 2017 at 5:23 am #

REPLY

We do cross validation, grid search etc to find the hyper parameters in machine algorithms. Similarly can we do anything to identify the above parameters??

Jason Brownlee March 3, 2017 at 7:46 am #

REPLY

Yes, we can use grid search and tuning for neural nets.

The stochastic nature of neural nets means that each experiment (set of configs) will have to be run many times (30? 100?) so that you can take the mean performance.

More general info on tuning neural nets here:
http://machinelearningmastery.com/improve-deep-learning-performance/

More on randomness and stochastic algorithms here:
http://machinelearningmastery.com/randomness-in-machine-learning/

Bogdan March 2, 2017 at 11:48 pm #

REPLY

Jason, Please tell me about these lines in your code:

seed = 7
numpy.random.seed(seed)

What do they do? And why do they do it?

One more question is why do you call the last section Bonus:Make a prediction?
I thought this what ANN was created for. What the point if your network's output is just what you have already know?

Jason Brownlee March 3, 2017 at 7:44 am #

REPLY

They seed the random number generator so that it produces the same sequence of random numbers each time the code is run. This is to ensure you get the same result as me.

I'm not convinced it works with Keras though.

More on randomness in machine learning here:
http://machinelearningmastery.com/randomness-in-machine-learning/

I was showing how to build and evaluate the model in this tutorial. The part about standalone prediction was an add-on.

Sounak sahoo March 3, 2017 at 7:39 pm #

REPLY

what exactly is the work of "seed" in the neural network code? what does it do?

Jason Brownlee March 6, 2017 at 10:44 am #

REPLY

Seed refers to seeding the random number generator so that the same sequence of random numbers is generated each time the example is run.

The aim is to make the examples 100% reproducible, but this is hard with symbolic math libs like Theano and TensorFlow backends.

For more on randomness in machine learning, see this post:
http://machinelearningmastery.com/randomness-in-machine-learning/

Priya Sundari March 3, 2017 at 10:19 pm #

REPLY

hello sir
could you plz tell me what is the role of optimizer and binary_crossentropy exactly? it is written that optimizer is used to search through the weights of the network which weights are we talking about exactly?

Jason Brownlee March 6, 2017 at 10:48 am #

REPLY

Hi Priya,

You can learn more about the fundamentals of neural nets here:
http://machinelearningmastery.com/neural-networks-crash-course/

Bogdan March 3, 2017 at 10:23 pm #

REPLY

If I am not mistaken, those lines I commented about used when we write

init = ‘uniform’

?

Bogdan March 3, 2017 at 10:44 pm #

REPLY

Could you explain in more details what is the batch size?

Jason Brownlee March 6, 2017 at 10:50 am #

REPLY

Hi Bogdan,

Batch size is how many patterns to show to the network before the weights are updated with the accumulated errors. The smaller the batch, the faster the learning, but also the more noisy the learning (higher variance).

Try exploring different batch sizes and see the effect on the train and test performance over each epoch.

Mohammad March 7, 2017 at 6:50 am #

REPLY

Dear Jason
Firstly, thanks for your great tutorials.
I am trying to classify computer networks packets using first 500 bytes of every packet to identify its protocol. I am trying to use 1d convolution. for simpler task,I just want to do binary classification and then tackle multilabel classification for 10 protocols. Here is my code but the accuracy which is like .63. how can I improve the performance? should I Use RNNs?

```
########
model=Sequential()
model.add(Convolution1D(64,10,border_mode='valid',
activation='relu',subsample_length=1, input_shape=(500, 1)))
#model.add(Convolution2D(32,5,5,border_mode='valid',input_shape=(1,28,28),))
model.add(MaxPooling1D(2))
model.add(Flatten())
model.add(Dense(200,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',
optimizer='adam',metrics=['accuracy'])
model.fit(train_set, y_train,
batch_size=250,
nb_epoch=30,
show_accuracy=True)
#x2= get_activations(model, 0,xprim )
#score = model.evaluate(t, y_test, show_accuracy = True, verbose = 0)
#print(score[0])
```

Jason Brownlee March 7, 2017 at 9:37 am #

REPLY

This post lists some ideas to try an lift performance:
http://machinelearningmastery.com/improve-deep-learning-performance/

Damiano March 7, 2017 at 10:13 pm #

REPLY

Hi Jason, thank you so much for this awesome tutorial. I have just started with python and machine learning.
I am joking with the code doing few changes, for example i have changed..

this:

```
# create model
model = Sequential()
model.add(Dense(250, input_dim=8, init='uniform', activation='relu'))
model.add(Dense(200, init='uniform', activation='relu'))
model.add(Dense(200, init='uniform', activation='relu'))
model.add(Dense(1, init='uniform', activation='sigmoid'))
```

and this:

```
model.fit(X, Y, nb_epoch=250, batch_size=10)
```

then i would like to pass some arrays for prediction so…

```
new_input = numpy.array([[3,88,58,11,54,24.8,267,22],[6,92,92,0,0,19.9,188,28],
[10,101,76,48,180,32.9,171,63], [2,122,70,27,0,36.8,0.34,27], [5,121,72,23,112,26.2,245,30]])

predictions = model.predict(new_input)
print predictions # [1.0, 1.0, 1.0, 0.0, 1.0]
```

is this correct? In this example i used the same series of training (that have 0 class), but i am getting wrong results. Only one array is correctly predicted.

Thank you so much!

Jason Brownlee March 8, 2017 at 9:41 am #

REPLY

Looks good. Perhaps you could try changing the configuration of your model to make it more skillful?

See this post:
http://machinelearningmastery.com/improve-deep-learning-performance/

ANJI March 13, 2017 at 8:48 pm #

REPLY

hello sir,
could you please tell me to rectify my error below it is raised while model is training:

str(array.shape))
ValueError: Error when checking model input: expected convolution2d_input_1 to have 4 dimensions, but got array with shape (68, 28, 28).

Jason Brownlee March 14, 2017 at 8:17 am #

REPLY

It looks like you are working with CNN, not related to this tutorial.

Consider trying this tutorial to get familiar with CNNs:
http://machinelearningmastery.com/handwritten-digit-recognition-using-convolutional-neural-networks-python-keras/

Rimjhim March 14, 2017 at 8:21 pm #

REPLY

I want a neural that can predict sin values. Further from a given data set i need to determine the function(for example if the data is of tan or cos, then how to determine that data is of tan only or cos only)

Thanks in advance

Sudarshan March 15, 2017 at 11:19 pm #

REPLY

Keras just updated to Keras 2.0. I have an updated version of this code here:
https://github.com/sudarshan85/keras-projects/tree/master/mlm/pima_indians

Jason Brownlee March 16, 2017 at 7:59 am #

REPLY

Nice work.

subhasish March 16, 2017 at 5:09 pm #

REPLY

hello sir,
can we use PSO (particle swarm optimisation) in this? if so can you tell how?

Jason Brownlee March 17, 2017 at 8:25 am #

REPLY

Sorry, I don't have an example of PSO for fitting neural network weights.

Ananya Mohapatra March 16, 2017 at 10:03 pm #

REPLY

hello sir,
what type of neural network is used in this code? as there are 3 types of Neural network that are…
feedforward, radial basis function and recurrent neurak network.

Jason Brownlee March 17, 2017 at 8:28 am #

REPLY

A multilayer perceptron (MLP) neural network. A classic type from the 1980s.

Diego March 17, 2017 at 3:58 am #

REPLY

got this error while compiling..

sigmoid_cross_entropy_with_logits() got an unexpected keyword argument 'labels'

Jason Brownlee March 17, 2017 at 8:30 am #

REPLY

Perhaps confirm that your libraries are all up to date (Keras, Theano or TensorFlow)?

Rohan March 20, 2017 at 5:20 am #

REPLY

Hi Jason!

I am trying to use two odd frames of a video to predict the even one. Thus I need to give two images as input to the network and get one image as output. Can you help me with the syntax for the first model.add()? I have X_train of dimension (190, 2, 240, 320, 3) where 190 are the number of odd pairs, 2 are the two odd images, and (240,320,3) are the (height, width, depth) of each image.

Herli Menezes March 21, 2017 at 8:33 am #

REPLY

Hello, Jason,

Thanks for your good tutorial. However i found some issues:
Warnings like these:

1 – Warning (from warnings module):
File “/usr/lib/python2.7/site-packages/keras/legacy/interfaces.py”, line 86
‘ call to the Keras 2 API: ' + signature)
UserWarning: Update your Dense call to the Keras 2 API: Dense(12, activation=”relu”, kernel_initializer=”uniform”, input_dim=8)

2 - Warning (from warnings module):
File "/usr/lib/python2.7/site-packages/keras/legacy/interfaces.py", line 86
' call to the Keras 2 API: ‘ + signature)
UserWarning: Update your Dense call to the Keras 2 API: Dense(8, activation="relu", kernel_initializer="uniform")

3 – Warning (from warnings module):
File “/usr/lib/python2.7/site-packages/keras/legacy/interfaces.py”, line 86
‘ call to the Keras 2 API: ' + signature)
UserWarning: Update your Dense call to the Keras 2 API: Dense(1, activation=”sigmoid”, kernel_initializer=”uniform”)

3 - Warning (from warnings module):
File "/usr/lib/python2.7/site-packages/keras/models.py", line 826
warnings.warn('The nb_epoch argument in fit '
UserWarning: The nb_epoch argument in fit has been renamed epochs`.

I think these are due to some package update..

But, the output of predictions was an array of zeros…
such as: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ….0.0]

I am running in a Linux Machine, Fedora 24,
Python 2.7.13 (default, Jan 12 2017, 17:59:37)
[GCC 6.3.1 20161221 (Red Hat 6.3.1-1)] on linux2

Why?

Thank you!

Jason Brownlee March 21, 2017 at 8:45 am #

REPLY

These look like warnings related to the recent Keras 2.0 release.

They look like just warning and that you can still run the example.

I do not know why you are getting all zeros. I will investigate.

Ananya Mohapatra March 21, 2017 at 6:21 pm #

REPLY

hello sir,
can you please help me build a recurrent neural network with the above given dataset. i am having a bit trouble in building the layers…

Jason Brownlee March 22, 2017 at 7:56 am #

REPLY

Hi Ananya ,

The Pima Indian diabetes dataset is a binary classification problem. It is not appropriate for a Recurrent Neural Network as there is no sequence information to learn.

Ananya Mohapatra March 22, 2017 at 8:04 pm #

REPLY

sir so could you tell on which type of dataset would the recurrent neural network accurately work? i have the dataset of EEG signals of epileptic patients…will recurrent network work on this?

Jason Brownlee March 23, 2017 at 8:49 am #

REPLY

It may if it is regular enough.

LSTMs are excellent at sequence problems that have regularity or clear signals to detect.

Shane March 22, 2017 at 5:18 am #

REPLY

Hi Jason, I have a quick question related to an error I am receiving when running the code in the tutorial…

When I run

Compile model
```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=
['accuracy'])
```

Python returns the following error:

sigmoid_cross_entropy_with_logits() got an unexpected keyword argument ‘labels’

Jason Brownlee March 22, 2017 at 8:09 am #

REPLY

Sorry, I have not seen this error Shane.

Perhaps check that your environment is up to date with the latest versions of the deep learning libraries?

Tejes March 24, 2017 at 1:04 am #

REPLY

Hi Jason,
Thanks for this awesome post.
I ran your code with tensorflow back end, just out of curiosity. The accuracy returned was different every time I ran the code. That didn’t happen with Theano. Can you tell me why?

Thanks in advance!

Jason Brownlee March 24, 2017 at 7:56 am #

REPLY

You will get different accuracy each time you run the code because neural networks are stochastic.

This is not related to the backend (I expect).

More on randomness in machine learning here:
http://machinelearningmastery.com/randomness-in-machine-learning/

Saurabh Bhagvatula March 27, 2017 at 9:49 pm #

REPLY

Hi Jason,
I’m new to deep learning and learning it from your tutorials, which previously helped me understand Machine Learning very well.
In the following code, I want to know why the number of neurons differ from input_dim in first layer of Nueral Net.

```
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))
```

```
model.add(Dense(8, init='uniform', activation='relu'))
model.add(Dense(1, init='uniform', activation='sigmoid'))
```

Jason Brownlee March 28, 2017 at 8:22 am #

REPLY

You can specify the number of inputs via “input_dim”, you can specify the number of neurons in the first hidden layer as the first parameter to Dense().

Saurabh Bhagvatula March 28, 2017 at 4:15 pm #

REPLY

Thanx a lot.

Jason Brownlee March 29, 2017 at 9:05 am #

REPLY

You’re welcome.

Nalini March 29, 2017 at 2:52 am #

REPLY

Hi Jason

while running this code for k fold cross validation it is not working.please give the code for k fold cross validation in binary class

Jason Brownlee March 29, 2017 at 9:10 am #

REPLY

Generally neural nets are too slow/large for k-fold cross validation.

Nevertheless, you can use a sklearn wrapper for a keras model and use it with any sklearn resampling method:
http://machinelearningmastery.com/evaluate-performance-machine-learning-algorithms-python-using-resampling/

trangtruong March 29, 2017 at 7:04 pm #

REPLY

Hi Jason, why i use function evaluate to get accuracy score my model with test dataset, it return result >1, i can't understand.

enixon April 3, 2017 at 3:08 am #

REPLY

Hey Jason, thanks for this great article! I get the following error when running the code above:

TypeError: Received unknown keyword arguments: {'epochs': 150}

Any ideas on why that might be? I can't get 'epochs', nb_epochs, etc to work…

Jason Brownlee April 4, 2017 at 9:07 am #

REPLY

You need to update to Keras version 2.0 or higher.

Ananya Mohapatra April 5, 2017 at 9:30 pm #

REPLY

def baseline_model():
# create model
model = Sequential()
model.add(Dense(10, input_dim=25, init='normal', activation='softplus'))
model.add(Dense(3, init='normal', activation='softmax'))
# Compile model
model.compile(loss='mean_squared_error', optimizer='adam', metrics=['accuracy'])
return model
sir here mean_square_error has been used for loss calculation. Is it the same as LMS algorithm. If not, can we use LMS , NLMS or RLS to calculate the loss?

Ahmad Hijazi April 5, 2017 at 10:19 pm #

REPLY

Hello Jason, thank you a lot for this example.

My question is, after I trained the model and an accuracy of 79.2% for example is obtained successfully, how can I test this model on new data?

for example if a new patient with new records appear, I want to guess the result (0 or 1) for him, how can I do that in the code?

Jason Brownlee April 9, 2017 at 2:36 pm #

REPLY

You can fit your model on all available training data then make predictions on new data as follows:

```
yhat = model.predict(X)
```

Perick Flaus April 6, 2017 at 12:16 am #

REPLY

Thanks Jason, how can we test if new patient will be diabetic or no (0 or 1) ?

Jason Brownlee April 9, 2017 at 2:36 pm #

REPLY

Fit the model on all training data and call:

```
yhat = model.predict(X)
```

Gangadhar April 12, 2017 at 1:28 am #

REPLY

Dr Jason,

In compiling the model i got below error

TypeError: compile() got an unexpected keyword argument ‘metrics’

unable to resolve the below error

Jason Brownlee April 12, 2017 at 7:53 am #

REPLY

Ensure you have the latest version of Keras, v2.0 or higher.

Omogbehin Azeez April 13, 2017 at 1:48 am #

REPLY

Hello sir,

Thank you for the post. A quick question, my dataset has 24 input and 1 binary output(170 instances, 100 epoch , hidden layer=6 and 10 batch, kernel_initializer=’normal’) . I adapted your code using Tensor flow and keras. I am having an accuracy of 98 to 100 percent. I am scared of over-fitting in my model. I need

your candid advice. Kind regards sir

Sethu Baktha April 13, 2017 at 5:19 am #

REPLY

Hi Jason,
If I want to use the diabetes dataset (NOT Pima) https://archive.ics.uci.edu/ml/datasets/Diabetes to predict Blood Glucose which tutorials and e-books of yours would I need to start with…. Also, the data in its current format with time, code and value is it usable as is or do I need to convert the data in another format to be able to use it.

Thanks for your help

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Welcome to Machine Learning Mastery

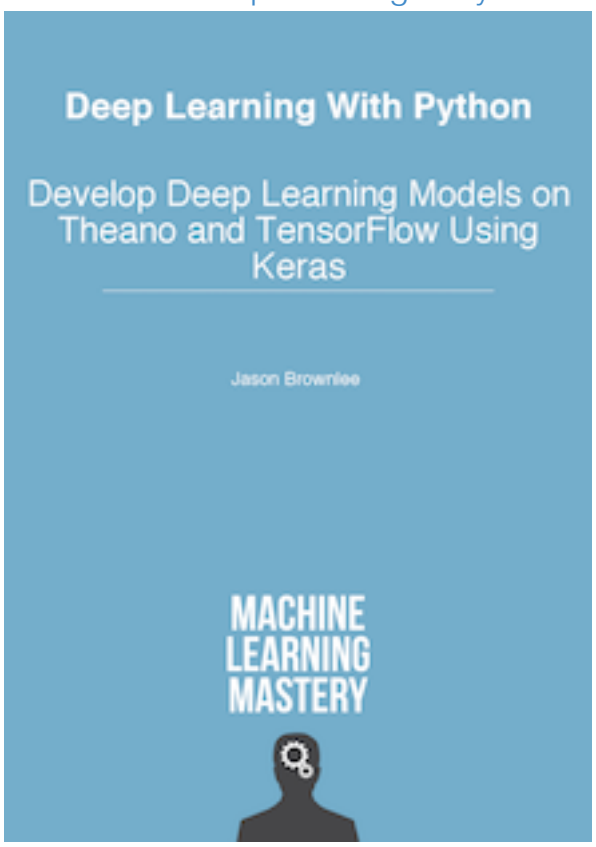
Hi, I'm Dr. Jason Brownlee.
My goal is to make practitioners like YOU awesome at applied machine learning.

Read More

Finally Get Started With Deep Learning

Sick of the fancy math and need for super computers?
Looking for step-by-step tutorials?
Want end-to-end projects?

Get Started With Deep Learning in Python Today!



POPULAR

Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras
JULY 21, 2016

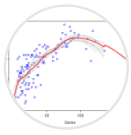
Develop Your First Neural Network in Python With Keras Step-By-Step
MAY 24, 2016

Your First Machine Learning Project in Python Step-By-Step
JUNE 10, 2016

How to Run Your First Classifier in Weka
FEBRUARY 17, 2014

Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras

JULY 26, 2016



A Tour of Machine Learning Algorithms

NOVEMBER 25, 2013

Tutorial To Implement k-Nearest Neighbors in Python From Scratch

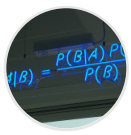
SEPTEMBER 12, 2014

Multi-Class Classification Tutorial with the Keras Deep Learning Library

JUNE 2, 2016

Regression Tutorial with the Keras Deep Learning Library in Python

JUNE 9, 2016



How To Implement Naive Bayes From Scratch in Python

DECEMBER 8, 2014