Traditional programming = data + rules = answer, Modern programming = data + answer = rules

A neural network is basically a set of functions which can learn patterns. Simple NN is contains one neuron. In keras, you use the word dense to defines a layer of connected neurons. Example:

Model = keras.Sequential([keras.layers.Dense(unit = 1, input.shape= [1])])

Model.compile(optimizer = ‘sgd’, loss = ‘mean\_squared\_error’)

Xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype = float)

ys = np.array([-3.0, 1.0, 1.0, 3.0, 5.0, 7.0], dtype = float)

model.fit(xs, ys, epoch = 500)

linear case gunakan optimizer: stokastik gradient descent (sgd) dan loss mean squared error (mse)

FMNIST : loss function : categorical cross entropy, optimizer: adam, activation function: relu

Horse vs Human : activation function : sigmoudi (binary class), loss fuction: binary crossentropy, optimizer: rmsprop

Jaringan saraf tidak tahu hubungan antara X dan Y, sehingga jaringan saraf akan membuat beberapa tebakan dan mengukur seberapa baik atau seberapa buruk tebakan tersebut menggunakan loss function, lalu optimizer digunakan untuk melihat mana tebakan terbaik.

Fashion MNIST yang memberikan 70 ribu gambar yang tersebar di 10 item pakaian yang berbeda. Gambar-gambar ini telah diperkecil menjadi 28 kali 28 piksel. Sekarang biasanya, semakin kecil semakin baik karena komputer memiliki lebih sedikit pemrosesan yang harus dilakukan. Tetapi tentu saja, Anda perlu menyimpan informasi yang cukup untuk memastikan bahwa fitur dan objek masih dapat dibedakan. Gambar juga dalam skala abu-abu, sehingga jumlah informasi juga berkurang. Setiap piksel dapat direpresentasikan dalam nilai dari nol hingga 255 sehingga hanya satu byte per piksel. Dengan 28 x 28 piksel dalam sebuah gambar, hanya 784 byte yang diperlukan untuk menyimpan seluruh gambar.

Dalam labelling, ada dua alasan utama kenapa memilih menggunakan integer sebagai label. Pertama, komputer bekerja lebih baik pada angka daripada teks. Kedua, bisa membantu mengurangi bias. Jika kita menamakannya sebagai ankle boot (nama suatu benda), tentu saja kita akan bias terhadap penutur bahasa Inggris.

Model = keras.Sequential(keras.layers.Flatten(input\_shape=(28, 28), keras.layers.Dense(128, activation = tf.nn.relu),

keras.layers.Dense(10, activation=tf .nn.softmax)))

Lapisan terakhir memiliki 10 neuron di dalamnya karena kami memiliki sepuluh kelas pakaian dalam dataset. Lapisan pertama adalah flatten layer dengan input 28 x 28. Sekarang, jika Anda ingat gambar kami berukuran 28x28, jadi kami menentukan bahwa ini adalah bentuk yang kami harapkan dari data. Flatten mengambil ini 28 x 28 persegi dan mengubahnya menjadi array linier sederhana. Hal menarik terjadi di lapisan tengah, kadang disebut juga hidden layer. Ini adalah 128 neuron di dalamnya, dan saya ingin Anda menganggapnya sebagai variabel dalam suatu fungsi. Mungkin menyebutnya x1, x2 x3, dll. Sekarang, ada aturan yang menggabungkan semua ini yang mengubah nilai 784 dari ankle boot menjadi nilai sembilan, dan serupa untuk semua 70.000 lainnya. Ini adalah fungsi yang terlalu rumit untuk Anda lihat dengan memetakan gambar sendiri, tetapi itulah yang dilakukan jaringan saraf. Jadi, misalnya, jika Anda kemudian mengatakan fungsinya adalah y sama dengan w1 dikali x1, ditambah w2 dikali x2, ditambah w3 dikali x3, hingga w128 dikali x128. Dengan mencari tahu nilai w, maka y akan menjadi sembilan, ketika Anda memiliki nilai input sepatu.

Gambar kami memiliki nilai dari nol hingga 255, tetapi jaringan saraf bekerja lebih baik dengan data yang dinormalisasi. Jadi, mari kita ubah menjadi antara nol dan satu hanya dengan membagi setiap nilai dengan 255. Seperti yang dijelaskan sebelumnya, ada lapisan input dalam bentuk data dan lapisan output dalam bentuk kelas, dan satu lapisan tersembunyi yang mencoba mencari tahu peran di antara mereka. Sekarang kami mengkompilasi model untuk menemukan loss function dan optimizer. Hasilnya kurang baik karena memiliki nilai loss yang besar dengan menggunakan model tsb.

Pertanyaan dari programmer khususnya ketika bereksperimen dengan jumlah epoch yang berbeda adalah, Bagaimana saya bisa berhenti berlatih ketika saya mencapai titik yang saya inginkan? Nah, kabar baiknya adalah, loop pelatihan mendukung callbacs. Jadi di setiap zaman, Anda dapat callbacks ke fungsi kode, setelah memeriksa metrik. Jika itu yang ingin Anda katakan, maka Anda dapat membatalkan pelatihan pada saat itu. Berikut kodenya. Ini diimplementasikan sebagai kelas terpisah, tetapi itu bisa sejalan dengan kode Anda yang lain. Tidak perlu dalam file terpisah. Di dalamnya, kita akan mengimplementasikan fungsi on\_epoch\_end, yang dipanggil oleh callback setiap kali epoch berakhir. Itu juga mengirimkan objek log yang berisi banyak informasi bagus tentang status pelatihan saat ini. Misalnya, loss saat ini tersedia di log. di sini saya memeriksa apakah kerugiannya kurang dari 0,4 dan membatalkan pelatihan itu sendiri.

Class myCallback(tf.keras.callbacks.Callback):

Def on\_epoch\_end(self, epoch, logs ={}):

If (logs.get(‘loss’) < 0.4)

Print(‘\nLoss is low so cancelling training!’)

Self.model.stop\_training = True

Pada contoh sebelumnya, Anda belajar bagaimana membuat jaringan saraf yang disebut DNN untuk mencocokkan pola dengan sekumpulan gambar item mode dengan label. Hanya dalam beberapa menit, Anda dapat melatihnya untuk mengklasifikasikan dengan akurasi yang cukup tinggi pada set pelatihan, tetapi sedikit kurang pada set pengujian. Sekarang, satu hal yang mungkin Anda lihat adalah ada banyak ruang kosong di setiap gambar. Meskipun hanya ada 784 piksel, akan menarik untuk melihat apakah ada cara untuk memadatkan gambar hingga fitur penting yang membedakan objek. Di situlah konvolusi masuk Jadi, apa konvolusi? Nah, jika Anda pernah melakukan pemrosesan gambar, biasanya melibatkan filter dan melewatkan filter itu di atas gambar untuk mengubah gambar di bawahnya. Prosesnya bekerja seperti ini. Untuk setiap piksel, ambil nilainya, dan lihat nilai tetangganya. Jika filter kami tiga kali tiga, maka kami dapat melihat tetangga terdekat, sehingga Anda memiliki kisi tiga kali tiga yang sesuai. Kemudian untuk mendapatkan nilai baru untuk piksel, kita cukup mengalikan setiap tetangga dengan nilai yang sesuai di filter.. Idenya di sini adalah bahwa beberapa konvolusi akan mengubah gambar sedemikian rupa sehingga fitur-fitur tertentu dalam gambar ditekankan. Jadi, misalnya, jika Anda melihat filter ini, maka garis vertikal pada gambar benar-benar menonjol. Dengan filter ini, garis horizontal muncul. Ketika konvolusi dikombinasikan dengan sesuatu yang disebut penyatuan (pooling), mereka bisa menjadi sangat kuat. Tapi sederhananya, pooling adalah cara mengompresi gambar. Cara cepat dan mudah untuk melakukannya, adalah dengan menelusuri gambar empat piksel sekaligus, yaitu piksel saat ini dan tetangganya di bawah dan di sebelah kanannya. Dari keempatnya, pilih nilai terbesar dan pertahankan saja. Kita tidak perlu melakukan semua perhitungan untuk memfilter dan mengompresi, kita cukup mendefinisikan lapisan convolutional dan pooling sbb:

Model = tf.keras.models.Sequential([

tf.keras.layers.Conv2D(64, (3,3), activation = ‘ relu’, input\_shape = (28,28,1)),

tf.keras.layers.MaxPooling2D(2,2),

tf.keras.layers.Conv2D(64, (3,3), activation = ‘relu’),

tf.keras.layers.MaxPooling2D(2,2),

tf.keras.layers.Flatten(),

tf.keras.Dense(128, activation=’relu’),

tf.keras.layers.Dense(10, activation = ‘sofftmax’)])

Di sini kita menentukan konvolusi pertama. Kami meminta keras untuk menghasilkan 64 filter untuk kami. Filter ini 3 kali 3, aktivasinya adalah relu, yang berarti nilai negatif akan dibuang, dan akhirnya bentuk inputnya seperti sebelumnya, 28 kali 28. Tambahan 1 itu berarti kita menghitung menggunakan satu byte untuk kedalaman warna. Seperti yang kita lihat sebelumnya, gambar kita adalah skala abu-abu, jadi kita hanya menggunakan satu byte. Kalau RGB =3. Baris kode berikutnya ini kemudian akan membuat lapisan penyatuan. Ini max-pooling karena kita akan mengambil nilai maksimum. Kami mengatakan itu adalah kumpulan dua kali dua, jadi untuk setiap empat piksel, yang terbesar akan bertahan seperti yang ditunjukkan sebelumnya. Jadi, pada saat gambar di flatten layer untuk masuk ke dense layer, itu sudah jauh lebih kecil. Itu dibagi empat, dan kemudian dibagi lagi. Jadi, isinya telah sangat disederhanakan, tujuannya adalah agar konvolusi akan menyaringnya ke fitur yang menentukan output. Metode yang sangat berguna pada model adalah metode model.summary. Ini memungkinkan Anda untuk memeriksa lapisan model, dan melihat perjalanan gambar melalui konvolusi, dan inilah hasilnya.

|  |  |  |
| --- | --- | --- |
| Layer (type) | Output Shape | Param # |
| Conv2D\_12 (Conv2D) | (None, 26, 26, 64) | 640 |
| Max\_pooling2d\_12 (maxpooling) | (None, 13, 13, 64) | 0 |
| Conv2D\_13 (Conv2D) | (None, 11, 11, 64) | 36928 |
| Max\_pooling2d\_13 (maxpooling) | (None, 5, 5, 64) | 0 |
| Flatten\_5 (flatten) | (None, 1600) | 0 |
| Dense\_10 (Dense) | (None, 128) | 204928 |
| Dense\_11 (Dense) | (None, 10) | 1290 |

Ini adalah tabel yang menunjukkan beberapa detail tentang lapisan-lapisannya dan bentuk keluarannya. Kuncinya adalah mengingat bahwa filternya adalah filter tiga kali tiga. Pertimbangkan apa yang terjadi ketika Anda mulai memindai gambar mulai dari kiri atas. Jadi, misalnya dengan gambar anjing di sebelah kanan ini, Anda dapat melihat diperbesar menjadi piksel di sudut kiri atas. Anda tidak dapat menghitung filter untuk piksel di kiri atas, karena tidak ada tetangga di atasnya atau di kirinya. Jadi, karena piksel ini tentu saja memiliki delapan tetangga yang dibutuhkan oleh filter tiga kali tiga, berarti Anda tidak dapat menggunakan margin satu piksel di sekitar gambar, sehingga output dari konvolusi akan menjadi dua piksel lebih kecil pada x, dan dua piksel lebih kecil pada y. Jika filter Anda berukuran lima kali lima untuk alasan yang sama, output Anda akan menjadi empat lebih kecil pada x, dan empat lebih kecil pada y. Jadi,, keluaran kami dari gambar 28x28, sekarang menjadi 26x 26, kami telah menghapus satu piksel pada x dan y, dan masing-masing batas.

Selanjutnya adalah lapisan max-pooling yang pertama. ingat kita menetapkannya menjadi 2x2, sehingga mengubah empat piksel menjadi satu. Jadi, sekarang output kita berkurang dari 26x26, menjadi 13x 13 (dibagi 2). Konvolusi kemudian akan beroperasi pada itu, dan tentu saja, kita kehilangan margin satu piksel seperti sebelumnya, jadi kita turun menjadi 11x11, tambahkan yang lain 2x2 max-pooling memiliki pembulatan ini ke bawah, dan turun, ke gambar 5x5. Jadi, sekarang dense layer kita sama seperti sebelumnya, tetapi diisi dengan gambar lima kali lima, bukan 28 kali 28. Tapi ingat, ini bukan hanya satu kompres gambar lima kali lima, bukan yang asli 28 kali 28, ada sejumlah konvolusi per gambar yang kami tentukan, dalam hal ini 64. Jadi, ada 64 gambar baru 5x5 yang telah dimasukkan. Ratakan (flatten) itu dan Anda memiliki 25 piksel dikalikan 64, yaitu 1600. Jadi, Anda dapat melihat bahwa lapisan datar yang baru memiliki 1.600 elemen di dalamnya, berbeda dengan 784 yang Anda miliki sebelumnya. Jumlah ini dipengaruhi oleh parameter yang Anda tetapkan saat menentukan lapisan 2D konvolusi.

Image data generator

Satu batasan dari konvolusi sebelumnya adalah ia menggunakan data gambar yang sangat seragam. Gambar berada dalam ukuran 28x 28. Namun apa yang terjadi bila Anda menggunakan gambar yang lebih besar dan di mana fitur tersebut mungkin berada di lokasi yang berbeda? Secara khusus image generator di TensorFlow dapat menangani masalah tersebut. Salah satu fitur image generator adalah dapat mengarahkannya ke direktori dan kemudian sub-direktori secara otomatis menghasilkan label untuk Anda. Jadi misalnya, pertimbangkan struktur direktori ini. Anda memiliki direktori gambar dan di dalamnya, Anda memiliki sub-direktori untuk pelatihan dan validasi. Saat Anda meletakkan sub-direktori untuk kuda dan manusia ini dan menyimpan gambar yang diperlukan di sana, generator gambar dapat membuat feeder untuk gambar-gambar itu dan memberi label otomatis untuk Anda.

Jadi mari kita lihat ini dalam kode. Class image generator tersedia di Keras.preprocessing.image. Saya akan memberikan rescale untuk menormalkan data. Anda kemudian dapat memanggil metode flow from directory di atasnya untuk memuat gambar dari direktori itu dan sub-direktori.. Nama sub-direktori akan menjadi label untuk gambar Anda yang ada di dalamnya. Jadi, pastikan direktori yang Anda tuju adalah direktori yang benar. Sekarang, gambar mungkin dalam berbagai bentuk dan ukuran, sayangnya untuk melatih jaringan saraf semua data input harus berukuran sama sehingga gambar perlu diubah ukurannya agar konsisten. Hal yang menyenangkan tentang kode ini adalah gambar diubah ukurannya untuk Anda saat dimuat. Keuntungan melakukannya saat runtime seperti ini adalah Anda kemudian dapat bereksperimen dengan berbagai ukuran tanpa memengaruhi data sumber . Meskipun set data kuda dan manusia sudah berukuran 300 kali 300, saat Anda menggunakan set data lain, ukurannya mungkin tidak selalu seragam. Jadi ini sangat berguna untuk Anda. Gambar akan dimuat untuk pelatihan dan validasi dalam batch yang lebih efisien daripada melakukannya satu per satu. Akhirnya ini adalah pengklasifikasi biner yaitu memilih antara dua hal yang berbeda; kuda dan manusia.

From tensorflow.keras.optimizers import RMSprop

Model.compile(los = ‘binary\_crossentropy’, optimizer=RMSprop(lr=0.001), metrics=[‘accuracy’])

History = model.fit( train\_generator, steps\_per\_epoch = 8, epoch = 15,

validation\_data=validation\_generator, validation\_steps = 8, verbose=2)

Pelatihan sekarang terlihat sedikit berbeda dari sebelumnya karena kami menggunakan generator, bukan kumpulan data. Ingat image generator dari sebelumnya, mari kita lihat setiap parameter secara detail. Parameter pertama adalah train\_generator yang mengalirkan gambar dari direktori pelatihan. Ingat ukuran batch yang Anda gunakan saat membuatnya? Ada 1.024 gambar di direktori pelatihan, jadi kami memuatnya di 128 sekaligus. Jadi untuk memuat semuanya, kita perlu melakukan 8 batch. Jadi kami mengatur steps\_per\_epoch.

Di sini kita hanya mengatur sebanyak 15 dalam kasus ini. Dan sekarang kita tentukan set validasi yang berasal dari validation\_generator yang juga kita buat sebelumnya. Itu memiliki 256 gambar, dan kami ingin menanganinya dalam kumpulan 32, jadi kami akan melakukan 8 langkah.

Dan parameter verbose menentukan berapa banyak yang akan ditampilkan saat pelatihan sedang berlangsung. Dengan verbose diatur ke 2, kita akan mendapatkan sedikit animasi yang menyembunyikan kemajuan epoch.

The impact of compressing images in input\_shape conv2d layer is overfitting.

Quiz 1

What do we call the process of telling the computer what the data represents (i.e. this data is for walking, this data is for running)? Labelling the Data

What is a Dense layer? A layer of connected neurons

How do you measure how good the current ‘guess’ is? Using the Loss function

What does the optimizer do? Generates a new and improved guess

What is Convergence? The process of getting very close to the correct answer

What is overfitting? An analysis that corresponds too closely or exactly to a particular set of data.

What does model.fit do? It trains the neural network to fit one set of values to another

Quiz 2

What is the resolution of o the 70,000 images from the Fashion MNIST dataset? 28x28 Greyscale

Why are there 10 output neurons in the Neural Network used as an example for the Computer Vision Problem? There are 10 different labels

What does Relu do? It only returns x if x is greater than zero

Why do you split data into training and test sets? To test a network with previously unseen data

True or False: The on\_epoch\_end function sends a logs object with lots of great information about the current state of training at the start of every epoch. False

Why do you set the callbacks= parameter in your fit function? So, on every epoch you can call back to a code function

Quiz 3

How do Convolutions improve image recognition? They isolate features in images

What does the Pooling technique do to the images? Reduces information in them while maintaining some features

True or False. If you pass a 28x28 image through a 3x3 filter the output will be 26x26 True

After max pooling a 26x26 image with a 2x2 filter, the output will be 56x56 False

How does using Convolutions in our Deep neural network impact training? Its impact will depend on other factors.

Quiz 4

What method on the Image Generator is used to normalize the image? rescale

How did we specify the training size for the images? The target\_size parameter on the training generator

When we specify the input\_shape to be (300, 300, 3), what does that mean? Every Image will be 300x300 pixels, with 3 bytes to define color

Convolutional Neural Networks are better for classifying images like horses and humans because:

* In these images, the features may be in different parts of the frame
* There’s a wide variety of horses
* There’s a wide variety of humans

If your training data is close to 1.000 accuracy, but your validation data isn’t, what’s the risk here? You’re overfitting on your training data

After reducing the size of the images, the training results were different. Why? We removed some convolutions to handle the smaller images

Using Image Generator, how do you label images? It’s based on the directory the image is contained in

Now that you’ve seen how an ImageDataGenerator can flow images from a directory and perform operations such as resizing them on the fly, the next thing to do is design the neural network to handle these more complex images.

Now that you’ve learned how to download and process the horses and humans dataset, you’re ready to train. When you defined the model, you saw that you were using a new loss function called ‘[Binary Crossentropy](https://gombru.github.io/2018/05/23/cross_entropy_loss/)’, and a new [optimizer](http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf) called [RMSProp](https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/RMSprop). If you want to learn more about the type of binary classification we are doing here, check out [this](https://www.youtube.com/watch?v=eqEc66RFY0I&t=6s) great video from Andrew!

Course 2

ImageDataGenerator.flow\_from\_directory Mengambil jalur ke direktori & menghasilkan kumpulan data tambahan. Sementara tipe pengembaliannya juga berbeda tetapi perbedaan utamanya adalah flow\_from\_directory adalah metode ImageDataGenerator sedangkan image\_dataset\_from\_directory adalah fungsi preprocessing untuk membaca direktori bentuk gambar.

Quiz 1

What does flow\_from\_directory give you on the ImageDataGenerator? The ability to easily load images for training, to pick the size of training images, to automatically label images based on their directory name

If my Image is sized 150x150, and I pass a 3x3 Convolution over it, what size is the resulting image? 148x148

If my data is sized 150x150, and I use Pooling of size 2x2, what size will the resulting image be? 75x75

If I want to view the history of my training, how can I access it? Create a variable ‘history’ and assign it to the return of model.fit or model.fit\_generator

What’s the name of the API that allows you to inspect the impact of convolutions on the images? The model.layers API

When exploring the graphs, the loss levelled out at about .75 after 2 epochs, but the accuracy climbed close to 1.0 after 15 epochs. What's the significance of this? There was no point training after 2 epochs, as we overfit to the training data

Why is the validation accuracy a better indicator of model performance than training accuracy? The validation accuracy is based on images that the model hasn't been trained with, and thus a better indicator of how the model will perform with new images.

Why is overfitting more likely to occur on smaller datasets? Because there's less likelihood of all possible features being encountered in the training process.

Image Augmentation is a very simple, but very powerful tool to help you avoid overfitting your data. The concept is very simple though: If you have limited data, then the chances of you having data to match potential future predictions is also limited, and logically, the less data you have, the less chance you have of getting accurate predictions for data that your model hasn't yet seen. To put it simply, if you are training a model to spot cats, and your model has never seen what a cat looks like when lying down, it might not recognize that in future.

Augmentation simply amends your images on-the-fly while training using transforms like rotation. So, it could 'simulate' an image of a cat lying down by rotating a 'standing' cat by 90 degrees. As such you get a cheap way of extending your dataset beyond what you have already. This week, you looked at the really useful tool that TensorFlow gives you with image augmentation. With it, you can effectively simulate a larger dataset from a smaller one with tools to move images around the frame, skew them, rotate them, and more. This can be an effective tool in fixing overfitting.

Quiz 2

**How do you use Image Augmentation in TensorFLow? Using parameters to the ImageDataGenerator**

**If my training data only has people facing left, but I want to classify people facing right, how would I avoid overfitting? Use the ‘horizontal\_flip’ parameter**

**When training with augmentation, you noticed that the training is a little slower. Why? Because the image processing takes cycles**

**What does the fill\_mode parameter do? It attempts to recreate lost information after a transformation like a shear**

When using Image Augmentation with the ImageDataGenerator, what happens to your raw image data on-disk? Nothing, all augmentation is done in-memory

How does Image Augmentation help solve overfitting? It manipulates the training set to generate more scenarios for features in the images

When using Image Augmentation my training gets... Slower

Using Image Augmentation effectively simulates having a larger data set for training. True

The idea behind it is to remove a random number of neurons in your neural network. This works very well for two reasons: The first is that neighboring neurons often end up with similar weights, which can lead to overfitting, so dropping some out at random can remove this. The second is that often a neuron can over-weigh the input from a neuron in the previous layer, and can over specialize as a result. Thus, dropping out can break the neural network out of this potential bad habit!

This week you've learned a lot of great new concepts!

You saw Transfer Learning, and how you can take an existing model, freeze many of its layers to prevent them being retrained, and effectively 'remember' the convolutions it was trained on to fit images.

You then added your own DNN underneath this so that you could retrain on your images using the convolutions from the other model.

You learned about regularization using dropout to make your network more efficient in preventing over-specialization and this overfitting.

Quiz 3

If I put a dropout parameter of 0.2, how many nodes will I lose? 20% of them

Why is transfer learning useful? Because I can use the features that were learned from large datasets that I may not have access to

How did you lock or freeze a layer from retraining? layer.trainable = false

How do you change the number of classes the model can classify when using transfer learning? (i.e. the original model handled 1000 classes, but yours handles just 2) When you add your DNN at the bottom of the network, you specify your output layer with the number of classes you want

Can you use Image Augmentation with Transfer Learning Models? Yes, because you are adding new layers at the bottom of the network, and you can use image augmentation when training these

Why do dropouts help avoid overfitting? Because neighbor neurons can have similar weights, and thus can skew the final training

What would the symptom of a Dropout rate being set too high? The network would lose specialization to the effect that it would be inefficient or ineffective at learning, driving accuracy down

Which is the correct line of code for adding Dropout of 20% of neurons using TensorFlow? tf.keras.layers.Dropout(0.2),

Quiz 4

The diagram for traditional programming had Rules and Data In, but what came out? Answers

Why does the DNN for Fashion MNIST have 10 output neurons? The dataset has 10 classes

What is a Convolution? A technique to extract features from an image

Applying Convolutions on top of a DNN will have what impact on training? It depends on many factors. It might make your training faster or slower, and a poorly designed Convolutional layer may even be less efficient than a plain DNN!

What method on an ImageGenerator is used to normalize the image? Rescale

When using Image Augmentation with the ImageDataGenerator, what happens to your raw image data on-disk. Nothing

Can you use Image augmentation with Transfer Learning? Yes. It's pre-trained layers that are frozen. So you can augment your images as you train the bottom layers of the DNN with them

When training for multiple classes what is the Class Mode for Image Augmentation? class\_mode='categorical'

Course 3

Quiz 1

What is the name of the object used to tokenize sentences? Tokenizer

What is the name of the method used to tokenize a list of sentences? fit\_on\_texts(sentences)

Once you have the corpus tokenized, what’s the method used to encode a list of sentences to use those tokens? texts\_to\_sequences(sentences)

When initializing the tokenizer, how do you specify a token to use for unknown words? oov\_token=<Token>

If you don’t use a token for out of vocabulary words, what happens at encoding? The word isn’t encoded, and is skipped in the sequence

If you have a number of sequences of different lengths, how do you ensure that they are understood when fed into a neural network? Use the pad\_sequences function from the tensorflow.keras.preprocessing.sequence namespace

If you have a number of sequences of different length, and call pad\_sequences on them, what’s the default result? They’ll get padded to the length of the longest sequence by adding zeros to the beginning of shorter ones

When padding sequences, if you want the padding to be at the end of the sequence, how do you do it? Pass padding=’post’ to pad\_sequences when initializing it

Quiz 2

What is the name of the TensorFlow library containing common data that you can use to train and test neural networks? TensorFlow Datasets

How many reviews are there in the IMDB dataset and how are they split? 50,000 records, 50/50 train/test split

How are the labels for the IMDB dataset encoded? Reviews encoded as a number 0-1

What is the purpose of the embedding dimension? It is the number of dimensions for the vector representing the word encoding

When tokenizing a corpus, what does the num\_words=n parameter do? It specifies the maximum number of words to be tokenized, and picks the most common ‘n-1’ words

To use word embeddings in TensorFlow, in a sequential layer, what is the name of the class? tf.keras.layers.Embedding

IMDB Reviews are either positive or negative. What type of loss function should be used in this scenario? Binary crossentropy

When using IMDB Sub Words dataset, our results in classification were poor. Why? Sequence becomes much more important when dealing with subwords, but we’re ignoring word positions

Quiz 3

Why does sequence make a large difference when determining semantics of language? Because the order in which words appear dictate their impact on the meaning of the sentence

How do Recurrent Neural Networks help you understand the impact of sequence on meaning? They carry meaning from one cell to the next

How does an LSTM help understand meaning when words that qualify each other aren’t necessarily beside each other in a sentence? Values from earlier words can be carried to later ones via a cell state

What keras layer type allows LSTMs to look forward and backward in a sentence? Bidirectional

What’s the output shape of a bidirectional LSTM layer with 64 units? (None, 128)

When stacking LSTMs, how do you instruct an LSTM to feed the next one in the sequence? Ensure that return\_sequences is set to True only on units that feed to another LSTM

If a sentence has 120 tokens in it, and a Conv1D with 128 filters with a Kernal size of 5 is passed over it, what’s the output shape? None, 116, 128)

What’s the best way to avoid overfitting in NLP datasets? None of the above

Quiz 4

When predicting words to generate poetry, the more words predicted the more likely it will end up gibberish. Why? It doesn’t, the likelihood of gibberish doesn’t change

What is a major drawback of word-based training for text generation instead of character-based generation? Because there are far more words in a typical corpus than characters, it is much more memory intensive

What are the critical steps in preparing the input sequences for the prediction model? Pre-padding the subprhases sequences. Generating subphrases from each line using n\_gram\_sequences.

In natural language processing, predicting the next item in a sequence is a classification problem.Therefore, after creating inputs and labels from the subphrases, we one-hot encode the labels. What function do we use to create one-hot encoded arrays of the labels?

True or False: When building the model, we use a sigmoid activated Dense output layer with one neuron per word that lights up when we predict a given word. False

Course 4

Quiz 1

**What is an example of a Univariate time series? Hour by hour temperature**

**What is an example of a Multivariate time series? Hour by hour weather**

**What is imputed data? A projection of unknown (usually past or missing) data**

A sound wave is a good example of time series data. True

**What is Seasonality? A regular change in shape of the data**

**What is a trend? An overall direction for data regardless of direction**

In the context of time series, what is noise? **Unpredictable changes in time series data**

**What is autocorrelation? Data that follows a predictable shape, even if the scale is different**

**What is a non-stationary time series? One that has a disruptive event breaking trend and seasonality**

Quiz 2

**What is a windowed dataset? A fixed-size subset of a time series**

**What does ‘drop\_remainder=true’ do? It ensures all rows in the data window are the same length by cropping data**

**What’s the correct line of code to split an n column window into n-1 columns for features and 1 column for a label? dataset = dataset.map(lambda window: (window[:-1], window[-1:]))**

What does MSE stand for? **Mean Squared error. What does MAE stand for? Mean Absolute Error**

If time values are in time[], series values are in series[] and we want to split the series into training and validation at time 1000, what is the correct code?

time\_train = time[:split\_time]

x\_train = series[:split\_time]

time\_valid = time[split\_time:]

x\_valid = series[split\_time:]

**If you want to inspect the learned parameters in a layer after training, what’s a good technique to use? Assign a variable to the layer and add it to the model using that variable. Inspect its properties after training**

**How do you set the learning rate of the SGD optimizer? Use the lr property**

**If you want to amend the learning rate of the optimizer on the fly, after each epoch, what do you do? Use a LearningRateScheduler object in the callbacks namespace and assign that to the callback**

Quiz 3

**If X is the standard notation for the input to an RNN, what are the standard notations for the outputs? Y(hat) and H**

What is a sequence to vector if an RNN has 30 cells numbered 0 to 29? **The Y(hat) for the last cell**

**What does a Lambda layer in a neural network do? Allows you to execute arbitrary code while training**

**What does the axis parameter of tf.expand\_dims do? Defines the dimension index at which you will expand the shape of the tensor**

**A new loss function was introduced in this module, named after a famous statistician. What is it called? Huber loss**

**What’s the primary difference between a simple RNN and an LST? In addition to the H output, LSTMs have a cell state that runs across all cells**

**If you want to clear out all temporary variables that tensorflow might have from previous sessions, what code do you run? tf.keras.backend.clear\_session()**

What happens if you define a neural network with these two layers?

**tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32)),**

**tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32)),**

**tf.keras.layers.Dense(1),**

**Your model will fail because you need return\_sequences=True after the first LSTM layer**

Quiz 4

**How do you add a 1 dimensional convolution to your model for predicting time series data? Use a Conv1D layer**

**What’s the input shape for a univariate time series to a Conv1D? [None, 1]**

**You used a sunspots dataset that was stored in CSV. What’s the name of the Python library used to read CSVs? CSV**

**If your CSV file has a header that you don’t want to read into your dataset, what do you execute before iterating through the file using a ‘reader’ object? Next(reader)**

**When you read a row from a reader and want to cast column 2 to another data type, for example, a float, what’s the correct syntax? float(row[2])**

What was the sunspot seasonality? **11 or 22 years depending on who you ask**

After studying this course, what neural network type do you think is best for predicting time series like our sunspots dataset? **Convolutions, RNN/LSTM, DNN**

**Why is MAE a good analytic for measuring accuracy of predictions for time series? It doesn’t heavily punish larger errors like square errors do**

**STRUCTURING ML**

You are delighted because this list of criteria will speed development and provide guidance on how to evaluate two different algorithms. False

Suggest to them that they define which criterion is to be optimized. Then, set thresholds for the other two. True

The city revises its criteria to:

* "We need an algorithm that can let us know a bird is flying over Peacetopia as accurately as possible."
* "We want the trained model to take no more than 10 sec to classify a new image.”
* “We want the model to fit in 10MB of memory.”

Given models with different accuracies, runtimes, and memory sizes, how would you choose one? Find the subset of models that meet the runtime and memory criteria. Then, choose the highest accuracy.

The city asks for your help in further defining the criteria for accuracy, runtime, and memory. How would you suggest they identify the criteria? Suggest to them that they define which criterion is to be optimized. Then, set thresholds for the other two.

The essential difference between an optimizing metric and satisficing metrics is the priority assigned by the stakeholders. False

You propose a 95/2.5%/2.5% for train/dev/test splits to the City Council. They ask for your reasoning. Which of the following best justifies your proposal? With a dataset comprising 10M individual samples, 2.5% represents 250k samples, which should be more than enough for dev and testing to evaluate bias and variance.

You train a system, and the train/dev set errors are 3.5% and 4.0% respectively. You decide to try regularization to close the train/dev accuracy gap. Do you agree? No, because you do not know what the human performance level

Which of the following best expresses how to evaluate the next steps in your project when your results for human-level performance, train, and dev set error are 0.1%, 2.0%, and 2.1% respectively? Based on differences between the three levels of performance, prioritize actions to decrease bias and iterate.

Which of the below shows the optimal order of accuracy from worst to best?Human-level performance -> the learning algorithm’s performance -> Bayes error.

After working on your algorithm you have to decide the next steps. Currently, human-level performance is 0.1%, training is at 2.0% and the dev set is at 2.1%. Which statement below best describes your thought process?

Address bias first through a larger model to get closest to human level error. One more answer

After running your model with the test set you find it is a 7.0% error compared to a 2.1% error for the dev set and 2.0% for the training set. What can you conclude? (Choose all that apply) You have overfitted to the dev set, You should try to get a bigger dev set.

After working on this project for a year, you finally achieve: Human-level performance, 0.10%, Training set error, 0.05%, Dev set error, 0.05%. Which of the following are true? (Check all that apply.) All or almost all of the avoidable bias has been accounted for. The model has recognized emergent features that humans cannot. (Chess and Go for example)

It turns out Peacetopia has hired one of your competitors to build a system as well. Your system and your competitor both deliver systems with about the same running time and memory size. However, your system has higher accuracy! However, when Peacetopia tries out your and your competitor’s systems, they conclude they actually like your competitor’s system better, because even though you have higher overall accuracy, you have more false negatives (failing to raise an alarm when a bird is in the air). What should you do? Rethink the appropriate metric for this task, and ask your team to tune to the new metric.

The City Council thinks that having more Cats in the city would help scare off birds. They are so happy with your work on the Bird detector that they also hire you to build a Cat detector. You have a huge dataset of 100,000,000 cat images. Training on this data takes about two weeks. Which of the statements do you agree with? (Check all that agree.) Lowering the number of images will reduce training time and likely allow for an acceptable tradeoff between iteration speed and accuracy., This significantly impacts iteration speed.