

Deep Learning Adventures

TensorFlow In Practice -

Presentation 6

A chat with Laurence Moroney, AI Lead at Google, after
completing Coursera's Tensorflow in Practice Specialization

Robert Kraig, David Patton, George Zoto

<https://www.meetup.com/Deep-Learning-Adventures>

In the beginning...

Meetup

The screenshot shows the homepage of the Deep Learning Adventures Meetup group. At the top, there's a photo of two men sitting on a blue couch. Below the photo, the group's name "Deep Learning Adventures" is displayed in large, bold letters. To the right of the name are links for "Start a new group", "Log in", and "Sign up". Under the group name, it says "Washington, DC", "377 members · Public group", and "Organized by George Z. and 2 others". There are sharing options for Facebook, Twitter, and LinkedIn. A red "Join this group" button is prominent. Below the main header, there are sections for "What we're about", "Upcoming events (2)", and "Members (377)". The "What we're about" section contains a detailed paragraph about the group's mission and history. The "Upcoming events" section lists an event with Laurence Moroney on July 3rd. The "Members" section shows a grid of member profiles.

Deep Learning Adventures

Washington, DC · 377 members · Public group · Organized by George Z. and 2 others

Share: [Facebook](#) [Twitter](#) [LinkedIn](#)

[Join this group](#)

What we're about

Deep Learning Adventures is a welcoming group for anyone interested in learning more about deep learning, its foundations, its strengths and weaknesses and ever growing applications that best serve humanity and help those in need throughout the world. After participating in hundreds of meetups in the area, we have taken many lessons learned and incorporated them into this group. This group is also startup oriented in the sense that we are open minded and ready to pivot to new directions as our community and needs around the world guide us....

[Read more](#)

Upcoming events (2)

FRI, JUL 3, 7:30 PM EDT

A chat with Laurence Moroney, AI Lead at Google

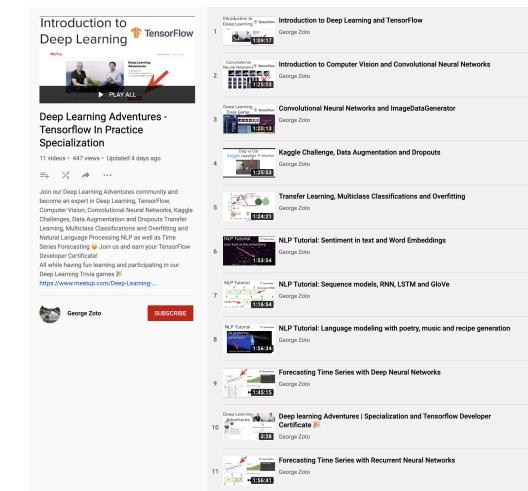
Online event

Join us for a fun conversation with Laurence Moroney, AI Lead at Google (<https://www.linkedin.com/in/laurence-moroney/>) and developer of our TensorFlow in Practice and TensorFlow: Data and Deployment Specializations! We plan to...

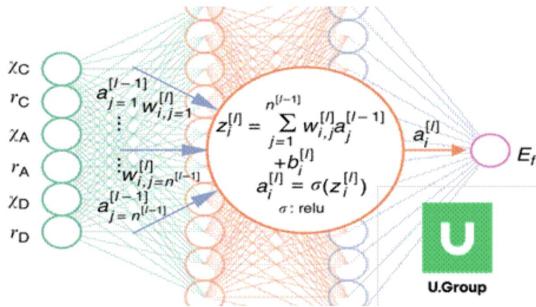
Members (377)

See all

- **Meetup Link**
<https://www.meetup.com/Deep-Learning-Adventures>
- **GitHub repository**
<https://github.com/georgezoto/Deep-Learning-Adventures>
<https://github.com/georgezoto/TensorFlow-in-Practice>
- **Join us on Slack**
https://join.slack.com/t/deeplearninga-nmk8930/shared_invite/zt-d52h9mm9-h~Q0ZXw5PXsTDzPIINivoq
- **Need a refresher or new to Deep Learning?**
- **YouTube recordings of all our Meetups 😊**
<https://bit.ly深深学习-tf>



Deep Learning and Data Science Partner Communities

[Start a new group](#)[Explore](#)[Messages](#)[Notifications](#)

DC Deep Learning Working Group

Washington, DC

1,631 members · Public group

Organized by Kathleen Perez-Lopez and 2 others

Share: [Facebook](#) [Twitter](#) [LinkedIn](#)

About Events Members Photos Discussions More

You're a member

What we're about

The DC Deep Learning Working Group (DC DLWG) is a not-for-profit, educational group of DC-metro machine learning enthusiasts looking to...

[Read more](#)

Past events (69)

[See all](#)

TUE, FEB 18, 6:00 PM EST

Using LSTMs to Predict Stock Price

Central Parking

15 attendees

Went

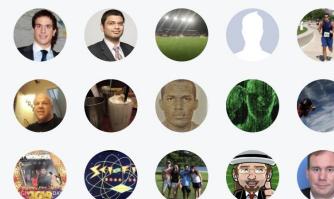
Organizers



Kathleen Perez-Lopez and 2 others
[Message](#)

Members (1,631)

[See all](#)



Source: <https://www.meetup.com/DC-Deep-Learning-Working-Group>

Deep Learning and Data Science Partner Communities

Meetup

Start a new group

Explore

Messages

Notifications



NOVA Deep Learning

Ashburn, VA

2,210 members · Public group

Organized by Data Community DC (DC2) and 1 other



Share: [f](#) [t](#) [in](#)

About Events Members Photos Discussions More

You're a member

What we're about

NOVA Deep Learning is a group for anyone interested in meeting to discuss the concepts and technologies associated with Deep Learning. It will be a...

[Read more](#)

Past events (58)

[See all](#)

THU, APR 2, 6:00 PM EDT

Virtual Event : Session 3 - [Udacity] Intro to Deep Learning using PyTorch

[Virtual event](#)

62 attendees

Went

Organizers

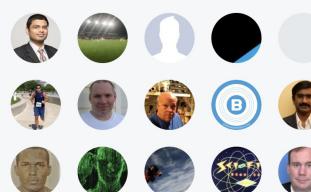


Data Community DC (DC2) and 1 other

[Message](#)

Members (2,210)

[See all](#)



Source: <https://www.meetup.com/novadeeplearning>

Deep Learning and Data Science Partner Communities

Meetup

Start a new group Explore Messages Notifications 

Machine Learning Paper Club

Arlington, VA
397 members - Public group
Organized by Elsa Schaefer and 1 other

Share:   

About Events Members Photos Discussions More You're a member ▾

What we're about
We're a small group that meets once or twice a month to review recent research in machine learning. We've recently been focused on computer...
[Read more](#)

Upcoming events (3) See all

THU, JUL 2, 7:30 PM EDT
NEW DATE! End-to-End Object Detection with Transformers: Just can't get enough
 Online event
We've covered the set-up and the loss function, and next we will dive into the implementation and results. Details A new paper out from Facebook AI Research (FAIR) this week shows how transformers can be used for object detection. Let's...
28 attendees  3 

Organizers
 Elsa Schaefer and 1 other
[Message](#)

Members (397) See all

See all



Source: <https://www.meetup.com/Machine-Learning-Paper-Club>

Bethesda Artificial Intelligence Meetup

A meetup for all levels of people interested in AI, Machine Learning, and related fields.

Centered in the Bethesda area, and meet biweekly Saturdays from 10AM to noon.
(We have free coffee when we're not virtual.)

<https://www.meetup.com/Bethesda-Artificial-Intelligence-Meetup/>

Source: <https://www.meetup.com/Bethesda-Artificial-Intelligence-Meetup>



The image shows a screenshot of the Meetup.com website for the "Bethesda Artificial Intelligence Meetup". The header includes the Meetup logo and links for "Start a new group", "Explore", "Messages", and "Notifications". The main title "Bethesda Artificial Intelligence Meetup" is displayed in bold black text. Below it, a sub-header indicates "1,105 members · Public group · Organized by Marc Pickett and 2 others". A large banner image at the top features a robotic hand reaching towards a human hand against a futuristic, glowing background. Navigation links "About", "Events", "Members", "Photos", "Discussions", and "More" are visible. A "Share" button with icons for Facebook, Twitter, and LinkedIn is present. A message box says "You're a member". Below the banner, sections include "What we're about" (describing the group as for AI and Machine Learning enthusiasts), "Upcoming events (2)" (listing an "AI Meet & Greet" on July 3, 2020, at 7:30 PM EDT, described as an online event), and "Organizers" (listing Marc Pickett and 2 others). The "Members (1,105)" section shows a grid of member profiles. A "See all" link is located on the right side of the page.

Deep Learning and Data Science Partner Communities



UpSkill > ML

UpSkill ML operates at the intersection of machine learning, data science and deep learning

Our goal is to **empower** technical professionals to **solve** their organizations most challenging **problems**

 https://www.meetup.com/deep_learning_doj/
<https://www.upskillml.com>

Data Education DC is a non-profit professional and educational group that meets bi-monthly to learn and discuss Data Science concepts and applications. Members learn to use Machine Learning, Data Science and Data Visualization tools that they can apply immediately following our meetups!

Source: https://www.meetup.com/deep_learning_doj
<https://www.meetup.com/Data-Education-DC>

UpSkill ML is a meetup that teaches applied machine learning concepts to data science and machine learning practitioners in the DMV area. Our sessions are in-person workshops that focus on applications and advances in the field, presenting them in a digestible way. Our goal is to present you with the math and intuition behind the ideas along with starter code for your projects.



<https://www.meetup.com/Data-Education-DC>

DATA EDUCATION DC'S GOAL IS TO DEMOCRATIZE DATA SCIENCE THROUGH HIGH QUALITY, ACCESSIBLE AND RELEVANT MEETUPS



NeuroTechDC

Washington, DC
363 members · Public group
Organized by NeuroTechX USA and 2 others

Share



You're a member

About

Events

Members

Photos

Discussions

More

See all



What we're about

Hello brains of the world! NeuroTechDC is the Washington D.C. chapter of NeuroTechX, a global network of engineers, designers, scientists, and hackers.

Read more

Upcoming events (1)

See all

Fri, Jul 3, 7:00 PM EDT
A chat with Laurence Moroney, AI Lead at Google

A short talk on learning Motivation, AI lead at Google for Deep Learning. Advertisers
Friday, July 3, 2020 7:00 PM to 8:30 PM EDT on the event. [REGISTER ON THE MEETUP PAGE HERE!](#) [Cross-post: Friday, July 3, 2020 7:00 PM to 8:30 PM EDT](#)

Attend

See all attendees

Attend

NeuroTechX is
an international community that facilitates
the advancement of neurotechnology



NeuroTechLIMA

Coming in: Mid 2020

20

CHAPTERS
WORLDWIDE

200

MEETUP
EVENTS

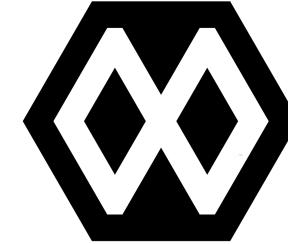
13,000

MEETUP
NEUROHACKERS

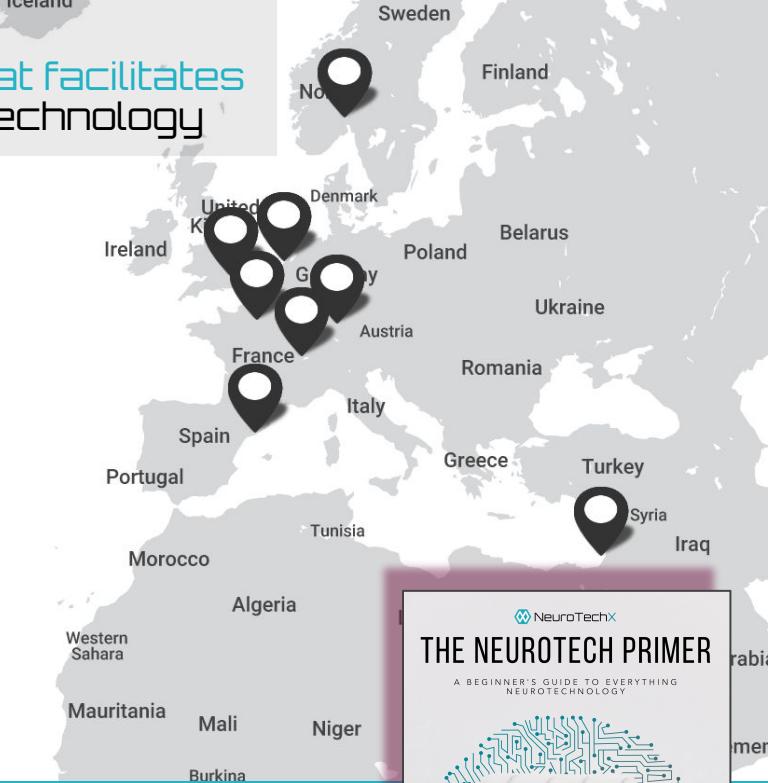
3500

ONLINE COMMUNITY
MEMBERS

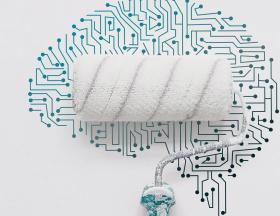
Source: <https://www.meetup.com/NeuroTechDC>



NEUROTECHX

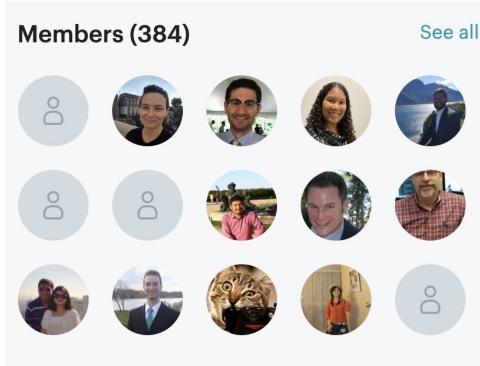


 NeuroTechX
THE NEUROTECH PRIMER
A BEGINNER'S GUIDE TO EVERYTHING NEUROTECHNOLOGY



Coming in:
Mid 2020

Get to know our community



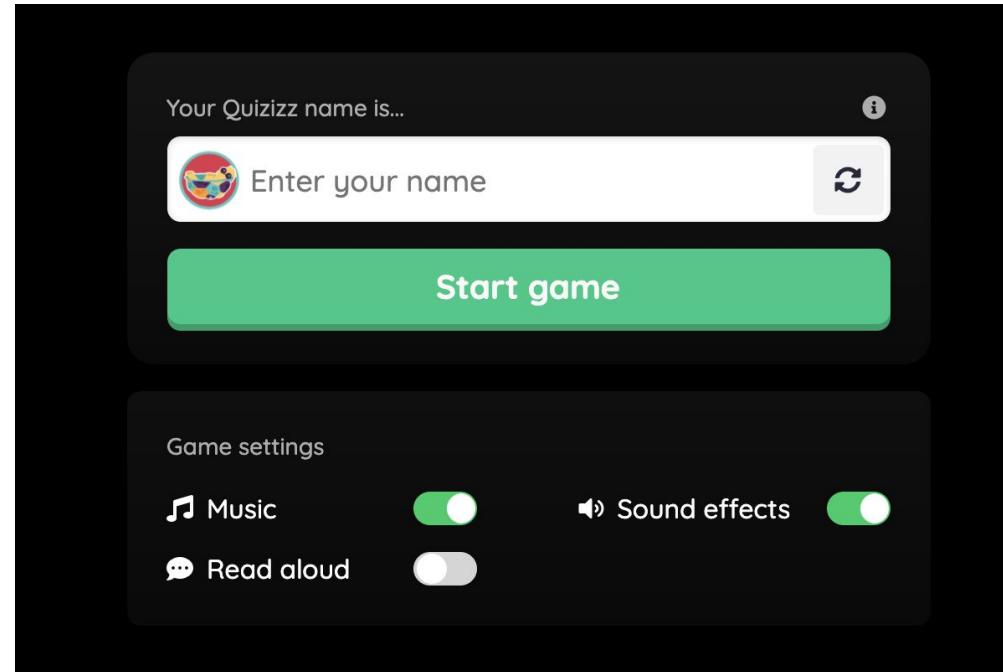
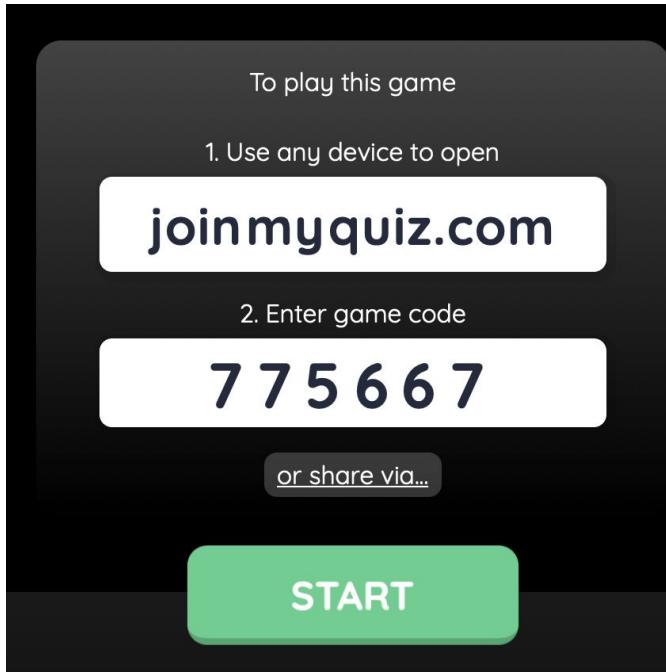
Hello, my name is ____ and I am a new/existing/regular member of this meetup.

I enjoy ____ and I am interested in learning more about

Training set 😊

Hello, my name is **George** and I am a **regular** member of this meetup. I enjoy **applying deep learning to solve interesting problems** and I am interested in learning more about **time series and forecasting**.

Not a typical Meetup... Get ready for a fun game 😊🎉



Time for a fun game



Practice here or use Flashcards:

<https://quizizz.com/join/quiz/5e87bdbc07fa7f001b120404/start?from=soloLinkShare&referrer=5d921444d0fa99001a135336>

The image shows two side-by-side screenshots. On the left is the Quizizz game dashboard for a game with code 432 236. It displays a list of 22 players with their names, profile icons, scores, and progress bars. A video feed of a person is shown in the top right corner. On the right is a slide from a presentation titled "Deep-Learning-Adventures-Chapter-1-Presentation-1 - Google Slides". The slide has a purple header with the question "How do you import Tensorflow in your Python code". Below the question are four numbered options: 1. import tf from ai, 2. import tensorflow from google, 3. expot tensorflow as tf, and 4. import tensorflow as tf.

Rank	Name	Score
6	Anil	8630
6	rek	8630
7	Charles Stockman	7450
8	Martin	7290
9	Angel	7230
10	Chuba	7050
11	Peter	6880
12	Dean	6840
13	Melissa	6505
14	Sanjay	5710
15	Hasson	4750
16	AH	4470
16	Kottie	4470
17	v	4010
18	KL	3910
19	Tony	3240
20	SS	2130
21	George	2100
22	Thomas	0

Attribution to Coursera and deeplearning.ai



A screenshot of the deeplearning.ai website homepage. The header features the deeplearning.ai logo and a navigation menu with links for "Courses", "Workers", "The Batch", "Events", "Forums", "Blog", and "Company". The main section has a dark background with the title "Break Into AI" in large white letters. Below the title, a paragraph reads: "Whether you want to build algorithms or build a company, deeplearning.ai's courses will teach you key concepts and applications of AI." A red button at the bottom says "Take the Deep Learning Specialization". To the right, there's a graphic of a laptop screen showing two images: a "Content Image" of the Golden Gate Bridge and a "Generated Image" of the same bridge in a painterly style, labeled "Art Generation with Deep Learning".

Source:

<https://www.coursera.org/about/terms>
<https://www.coursera.org/>
<https://www.deeplearning.ai/>

TensorFlow: An end-to-end open source machine learning platform

[TensorFlow](#) [Install](#) [Learn](#) [API](#) [Resources](#) [Community](#) [Why TensorFlow](#) [Search](#) [English](#) [GitHub](#) [Sign in](#)

Solutions to common ML problems

Simple step-by-step walkthroughs to solve common ML problems with TensorFlow.



For beginners

Your first neural network

Train a neural network to classify images of clothing, like sneakers and shirts, in this fast-paced overview of a complete TensorFlow program.



For experts

Generative adversarial networks

Train a generative adversarial network to generate images of handwritten digits, using the Keras Subclassing API.

Find | Compare

TensorFlow
Software

Keras
Search term

PyTorch
Computer application

+ Add comparison

United States

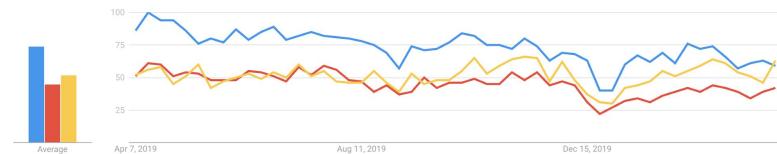
Past 12 months

All categories

Web Search

LEARN MORE

Interest over time



Source: <https://www.tensorflow.org/>

Chapter 1 - TensorFlow in Practice Specialization

About this Specialization

199,621 recent views

Discover the tools software developers use to build scalable AI-powered algorithms in TensorFlow, a popular open-source machine learning framework.

In this four-course Specialization, you'll explore exciting opportunities for AI applications. Begin by developing an understanding of how to build and train neural networks. Improve a network's performance using convolutions as you train it to identify real-world images. You'll teach machines to understand, analyze, and respond to human speech with natural language processing systems. Learn to process text, represent sentences as vectors, and input data to a neural network. You'll even train an AI to create original poetry!

AI is already transforming industries across the world. After finishing this Specialization, you'll be able to apply your new TensorFlow skills to a wide range of problems and projects.

Looking for more advanced TensorFlow content? Check out the new [TensorFlow: Data and Deployment Specialization](#).

Chapter 1 - TensorFlow in Practice Specialization

There are 4 Courses in this Specialization

COURSE

1

Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning

★★★★★ 4.7 6,196 ratings • 1,282 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This course is part of the upcoming Machine Learning in Tensorflow Specialization and will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

[SHOW ALL](#)



6 hours to complete

A New Programming Paradigm

Welcome to this course on going from Basics to Mastery of TensorFlow. We're excited you're here! In week 1 you'll get a soft introduction to what Machine Learning and Deep Learning are, and how they offer you a new programming paradigm, giving you a new set of tools to open previously unexplored scenarios. All you need to know is some very basic SHOW ALL



4 videos (Total 16 min), 5 readings, 3 quizzes [SEE ALL](#)

COURSE

2

Convolutional Neural Networks in TensorFlow

★★★★★ 4.7 2,751 ratings • 410 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This course is part of the upcoming Machine Learning in Tensorflow Specialization and will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

[SHOW ALL](#)



7 hours to complete

Introduction to Computer Vision

Welcome to week 2 of the course! In week 1 you learned all about how Machine Learning and Deep Learning is a new programming paradigm. This week you're going to take that to the next level by beginning to solve problems of computer vision with just a few lines of code! SHOW ALL



7 videos (Total 15 min), 6 readings, 3 quizzes [SEE ALL](#)

COURSE

3

Natural Language Processing in TensorFlow

★★★★★ 4.6 2,037 ratings • 277 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This Specialization will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

[SHOW ALL](#)



8 hours to complete

Enhancing Vision with Convolutional Neural Networks

Welcome to week 3! In week 2 you saw a basic Neural Network for Computer Vision. It did the job nicely, but it was a little naive in its approach. This week we'll see how to make it better, as discussed by Laurence and Andrew here. SHOW ALL



6 videos (Total 19 min), 6 readings, 3 quizzes [SEE ALL](#)

COURSE

4

Sequences, Time Series and Prediction

★★★★★ 4.6 1,374 ratings • 223 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This Specialization will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

[SHOW ALL](#)



9 hours to complete

Using Real-world Images

Last week you saw how to improve the results from your deep neural network using convolutions. It was a good start, but the data you used was very basic. What happens when your images are larger, or if the features aren't always in the same place? Andrew and Laurence discuss this to prepare you for what you'll learn this week: handling complex images! SHOW ALL

Source: <https://www.coursera.org/specializations/tensorflow-in-practice>

TensorFlow in Practice Specialization

Prerequisites

- Programming in Python
- High school level math

Helpful but not required

- Basic linear algebra and calculus
- Basic deep learning concepts

Setup



Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. You can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.

<https://colab.research.google.com>

Discussion with Laurence

- Goals and challenges developing TensorFlow in Practice
- User engagement in this remote setup
- Other events you found interesting/cool
- Other deep learning frameworks
- Other specializations in mind - Data and Deployment
- TensorFlow Developer Exam: PyCharm vs Colab

Course 1: Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning

Week 1: A New Programming Paradigm

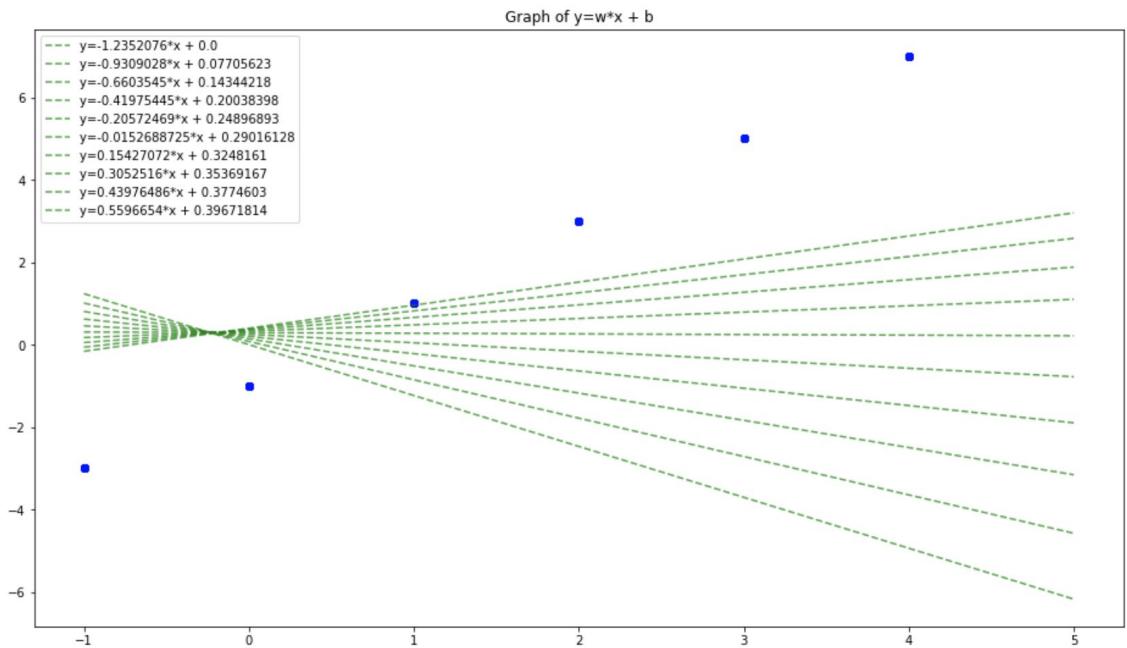
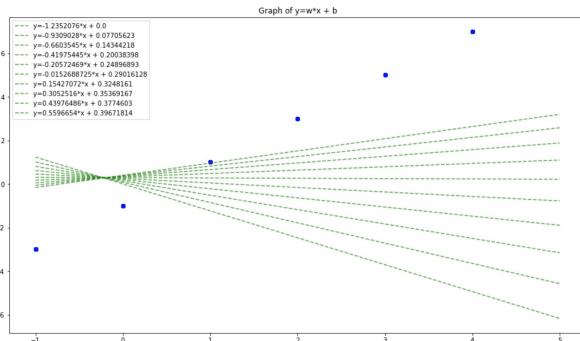
Week 2: Introduction to Computer Vision

Week 3: Enhancing Vision with Convolutional Neural Networks

Week 4: Using Real-world Images

Content added after our 1st session by George

```
: model.fit(xs, ys, epochs=10, callbacks=[get_weight_and_bias()])  
  
Epoch 1/10  
-1.2352076 0.0  
1/1 [=====] - 0s 2ms/step - loss: 45.3716  
Epoch 2/10  
-0.9309028 0.07705623  
1/1 [=====] - 0s 2ms/step - loss: 36.0725  
Epoch 3/10  
-0.6603545 0.14344218  
1/1 [=====] - 0s 2ms/step - loss: 28.7486  
Epoch 4/10  
-0.41975445 0.20038398  
1/1 [=====] - 0s 1ms/step - loss: 22.9789  
Epoch 5/10  
-0.20572469 0.24896893  
1/1 [=====] - 0s 1ms/step - loss: 18.4323  
Epoch 6/10  
-0.0152688725 0.29016128  
1/1 [=====] - 0s 1ms/step - loss: 14.8479  
Epoch 7/10  
0.15427072 0.3248161  
1/1 [=====] - 0s 1ms/step - loss: 12.0208  
Epoch 8/10  
0.3952516 0.35369167  
1/1 [=====] - 0s 1ms/step - loss: 9.7895  
Epoch 9/10  
0.43976486 0.3774603  
1/1 [=====] - 0s 1ms/step - loss: 8.0273  
Epoch 10/10  
0.5596654 0.39671814  
1/1 [=====] - 0s 1ms/step - loss: 6.6342  
: <tensorflow.python.keras.callbacks.History at 0x7f77aae0da20>
```



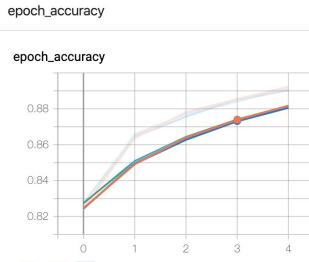
Our code is available on GitHub and on our Slack channel:
https://github.com/georgezoto/TensorFlow-in-Practice/blob/master/Course-1-Introduction-to-TensorFlow/C1W1_A_new_programming_paradigm_1.ipynb

#tensorflow-in-practice-specialization

You created this channel on March 31st. This is the very beginning of the #tensorflow-in-practice-specialization channel.

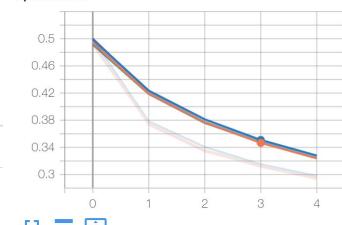
An Introduction to computer vision

```
# Define the Keras TensorBoard callback.  
logdir = "logs/scalars/" + datetime.now().strftime("%Y%m%d-%H%M%S")  
tensorboard_callback = keras.callbacks.TensorBoard(log_dir=logdir)  
  
training_history = model.fit(training_images,  
    training_labels,  
    batch_size = 64,  
    verbose = 0,  
    epochs = 5,  
    validation_data = (test_images, test_labels),  
    callbacks=[tensorboard_callback])
```



Name	Smoothed	Value	Step	Time	Relative
20200410-165543/train	0.8738	0.8854	3	Fri Apr 10, 12:56:03	13s
20200410-171047/train	0.873	0.8852	3	Fri Apr 10, 13:11:06	12s
20200410-171729/train	0.8731	0.884	3	Fri Apr 10, 13:17:47	12s
20200410-175850/train	0.8738	0.8855	3	Fri Apr 10, 13:59:09	12s
20200410-180421/train	0.8741	0.8855	3	Fri Apr 10, 14:04:39	12s

Name	Smoothed	Value	Step	Time	Relative
20200410-165543/train	0.3466	0.3113	3	Fri Apr 10, 12:56:03	13s
20200410-171047/train	0.3509	0.3152	3	Fri Apr 10, 13:11:06	12s
20200410-171729/train	0.3489	0.3152	3	Fri Apr 10, 13:17:47	12s
20200410-175850/train	0.3475	0.3128	3	Fri Apr 10, 13:59:09	12s
20200410-180421/train	0.3474	0.3139	3	Fri Apr 10, 14:04:39	12s

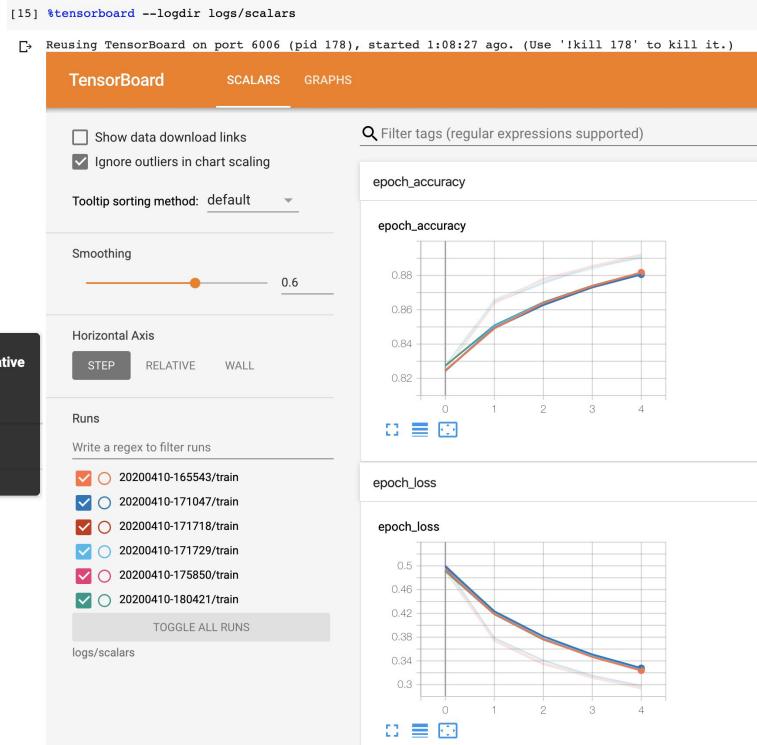


Extra Content - TensorBoard - issues in Colab

https://colab.research.google.com/github/tensorflow/tensorboard/blob/master/docs/scalars_and_keras.ipynb

Op-level graph

Start TensorBoard and wait a few seconds for the UI to load. Select the Graphs dashboard by tapping "Graphs" at the top.



Course 2: Convolutional Neural Networks in TensorFlow

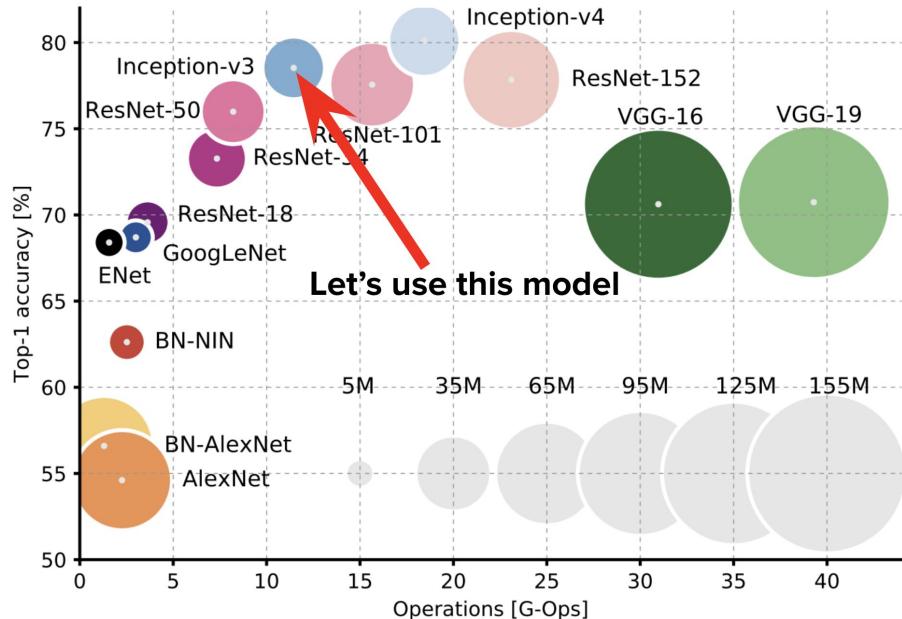
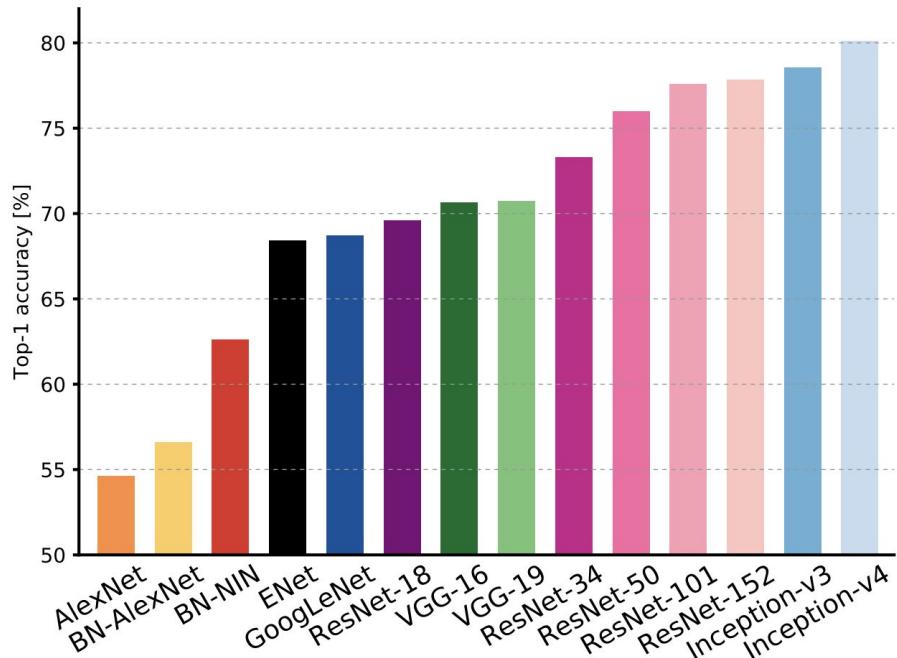
Week 1: Exploring a Larger Dataset

Week 2: Augmentation: A technique to avoid overfitting

Week 3: Transfer Learning

Week 4: Multiclass Classifications

Transfer Learning - Extra Content - Classic Networks by David



Extra Content - Regularization

Logistic regression

$$\min_{w,b} J(w,b) \quad w \in \mathbb{R}^n, b \in \mathbb{R}$$

$$J(w,b) = \frac{1}{m} \sum_{i=1}^m L(y_i, \hat{y}_i) + \frac{\lambda}{2m} \|w\|_2^2$$

λ = regularization parameter
lambda
lambda

L₂ regularization $\|w\|_2^2 = \sum_{j=1}^n w_j^2 = w^T w \leftarrow$

L₁ regularization $\frac{\lambda}{2m} \sum_{j=1}^n |w_j| = \frac{\lambda}{2m} \|w\|_1$

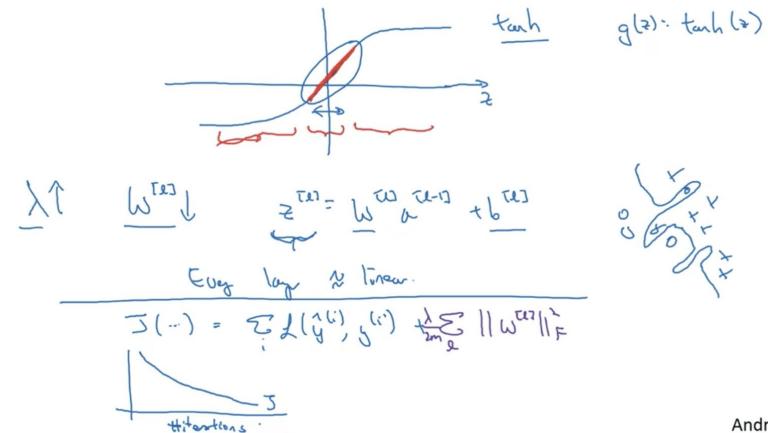
w will be sparse

Andrew Ng

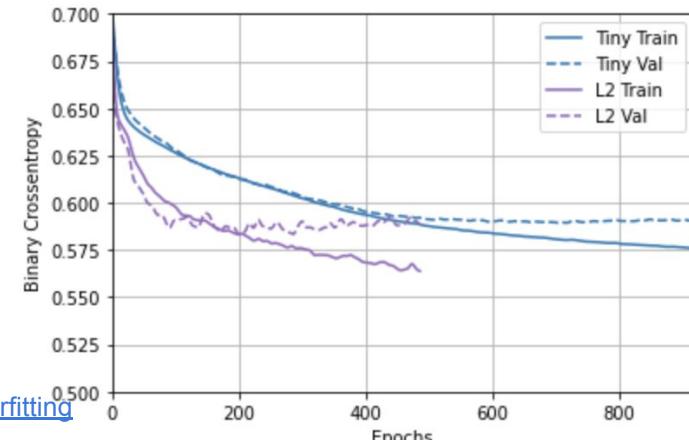
```
from tensorflow.keras import regularizers
```

```
l2_model = tf.keras.Sequential([
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001),
                input_shape=(FEATURES,)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(1)
])
```

How does regularization prevent overfitting?



Andrew Ng



Source: <https://www.coursera.org/learn/deep-neural-network/lecture/Srsrc/regularization>

<https://www.coursera.org/learn/deep-neural-network/lecture/T6OJj/why-regularization-reduces-overfitting>

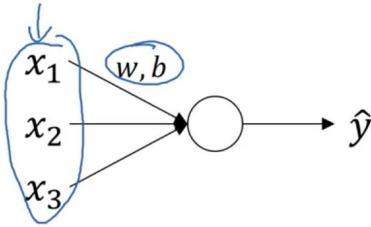
https://www.tensorflow.org/tutorials/keras/overfit_and_underfit

Extra Content - Batch Normalization

Normalizing inputs to speed up learning

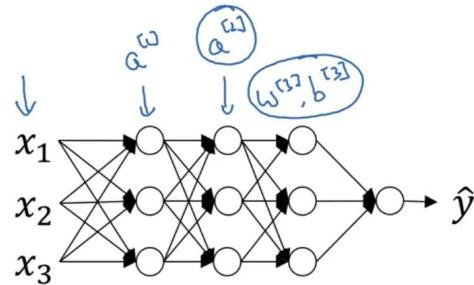
TensorFlow > API > TensorFlow Core v2.1.0 > Python

`tf.keras.layers.BatchNormalization`



$$\begin{aligned}\mu &= \frac{1}{m} \sum_i x^{(i)} \\ X &= X - \mu \\ \sigma^2 &= \frac{1}{m} \sum_i (x^{(i)} - \mu)^2 \quad \leftarrow \text{element-wise} \\ Z &= X / \sigma^2\end{aligned}$$

```
graph LR; Z((Z)) --> Y_hat((y-hat))
```



Can we normalize $a^{[2]}$ so
as to train $w^{[2]}, b^{[2]}$ faster
Normalize $\underline{z}^{[2]}$.

Normalize the activations of the previous layer at each batch, i.e. applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

Batch normalization differs from other layers in several key aspects:

Source:

<https://www.coursera.org/learn/deep-neural-network/lecture/4ptp2/normalizing-activations-in-a-network>

https://www.tensorflow.org/api_docs/python/tf/keras/layers/BatchNormalization

Andrew Ng

Extra Content - Overfitting in Neural Networks

To recap: here are the most common ways to prevent overfitting in neural networks:

- Get more training data.
- Reduce the capacity of the network.
- Add weight regularization.
- Add dropout.

Two important approaches not covered in this guide are:

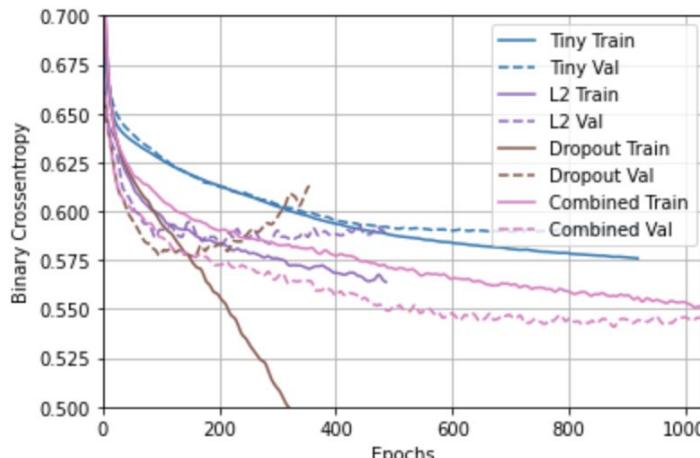
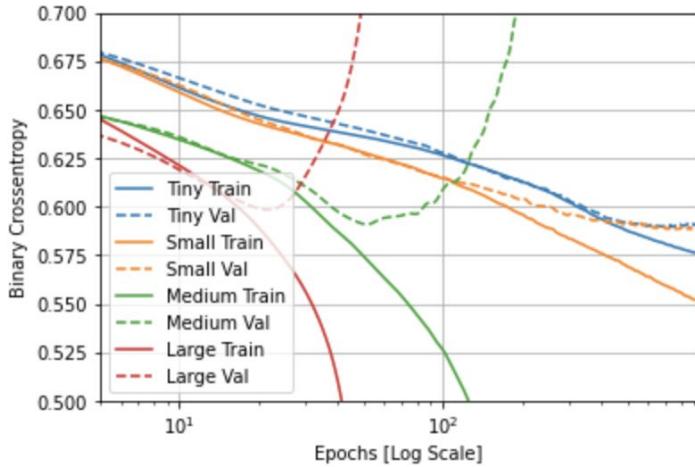
- data-augmentation
- batch normalization

Remember that each method can help on its own, but often combining them can be even more effective.

- Data Augmentation
- Dropout
- Early Stopping
- Ensembling
- Injecting Noise
- L1 Regularization
- L2 Regularization

The Higgs Dataset:
11M samples, 28
features, binary class

Total params:
Tiny: 0.5K
Small: 0.7K
Medium: 10K
Large: 800K



Source: https://www.tensorflow.org/tutorials/keras/overfit_and_underfit
<https://ml-cheatsheet.readthedocs.io/en/latest/regularization.html>

Course 3: Natural Language Processing in TensorFlow

Week 1: Sentiment in text

Week 2: Word Embeddings

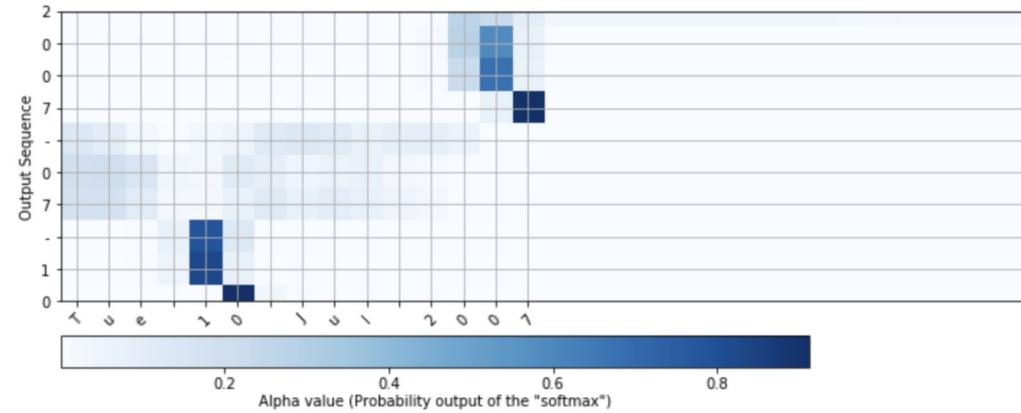
Week 3: Sequence models

Week 4: Sequence models and literature

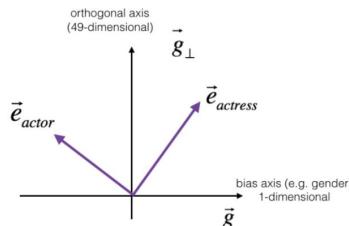
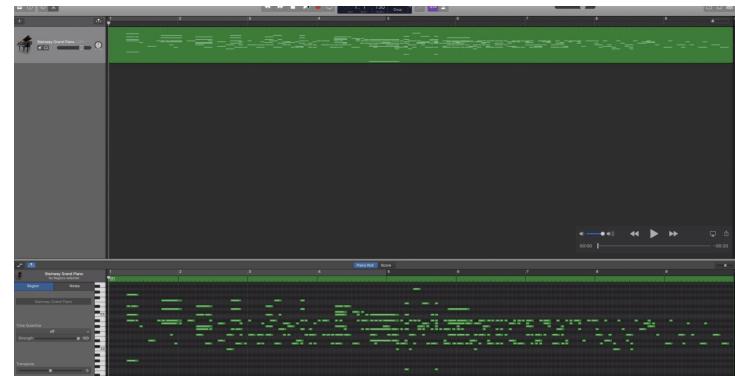
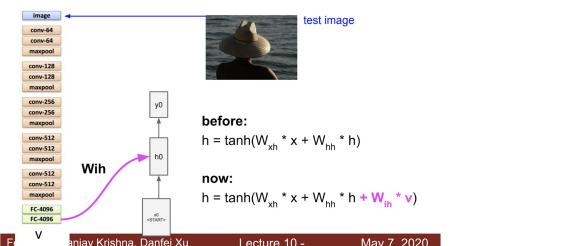
Extra Content - cool ways to use RNNs, Bias, Attention, Text and Vision by George

```
text = EXAMPLES[3]
pre_plot(model, human_vocab, inv_machine_vocab, text, num = 7, n_s = 64);
```

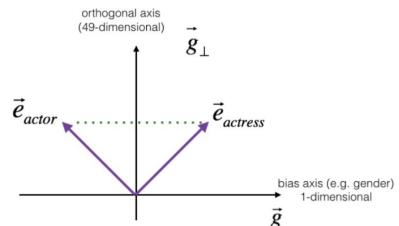
Input Tue 10 Jul 2007
Output 2007-07-10



Sequence models and computer vision



before equalizing,
"actress" and "actor" differ
in many ways beyond the
direction of \vec{g}



after equalizing,
"actress" and "actor" differ
only in the direction of \vec{g} , and further
are equal in distance from \vec{g}_\perp

Source: <https://github.com/georgezoto/RNN-LSTM-NLP-Sequence-Models>

Extra Content - Deep Learning for Structured Data

TensorFlow

Install Learn API Resources Community Why TensorFlow

Search

TensorFlow tutorials
Quickstart for beginners
Quickstart for experts

BEGINNER

ML basics with Keras

- Basic image classification
- Text classification with TF Hub
- Text classification with preprocessed text

Regression

- Overfit and underfit
- ... and more

```
def build_model():
    model = keras.Sequential([
        layers.Dense(64, activation='relu', input_shape=[len(train_features)]),
        layers.Dense(64, activation='relu'),
        layers.Dense(1)
    ])

    optimizer = tf.keras.optimizers.RMSprop(0.001)

    model.compile(loss='mse',
                  optimizer=optimizer,
                  metrics=['mae', 'mse'])
    return model
```

	MPG	Cylinders	Displacement	Horsepower	Weight	Acceleration	Model Year	Origin
393	27.0	4	140.0	86.0	2790.0	15.6	82	1
394	44.0	4	97.0	52.0	2130.0	24.6	82	2
395	32.0	4	135.0	84.0	2295.0	11.6	82	1
396	28.0	4	120.0	79.0	2625.0	18.6	82	1
397	31.0	4	119.0	82.0	2720.0	19.4	82	

Source: <https://www.tensorflow.org/tutorials/keras/regression>

Course 4: Sequences, Time Series and Prediction

Week 1: Sequences and Prediction

Week 2: Deep Neural Networks for Time Series

Week 3: Recurrent Neural Networks for Time Series

Week 4: Real-world time series data

Laurence's LR optimization? by David

For consistent results:

```
import random  
seed = 51  
tf.random.set_seed(seed)  
random.seed = seed
```

TensorFlow > API > TensorFlow Core v2.2.0 > Python

tf.keras.callbacks.ReduceLROnPlateau

```
tf.keras.callbacks.ReduceLROnPlateau(  
    monitor='val_loss', factor=0.1, patience=10, verbose=0, mode='auto',  
    min_delta=0.0001, cooldown=0, min_lr=0, **kwargs  
)
```

Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

Set all runs to epochs=500

Restart runtime and run all each test.

Source: https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ReduceLROnPlateau

Laurence's LR optimization results (seed=51)

1. `optimizer=tf.keras.optimizers.SGD(lr=1e-6, momentum=0.9)`
`mae 4.500836`

2. `optimizer = tf.keras.optimizers.SGD(lr=4e-6, momentum=0.9)`
`mae 4.8579106`

3. `optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)`
`mae 5.3804765`

4. `ReduceLROnPlateau(monitor='loss', patience=25, verbose=1)`
`optimizer = tf.keras.optimizers.Adam(learning_rate=0.01)`
`mae 4.490887`

Recurrent Neural Networks for Time Series by Robert

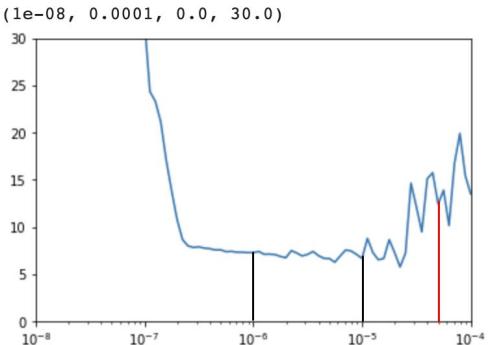
```
# Create the series
series = baseline + trend(time, slope) + seasonality(time, period=365, amplitude=10)
# Update with noise
series += noise(time, noise_level, seed=42)

split_time = 1000
time_train = time[:split_time]
x_train = series[:split_time]
time_valid = time[split_time:]
x_valid = series[split_time:]

window_size = 20
batch_size = 32
shuffle_buffer_size = 1000

my_epoch = 100

plt.semilogx(history.history["lr"], history.history["loss"])
plt.axis([1e-8, 1e-4, 0, 30])
```

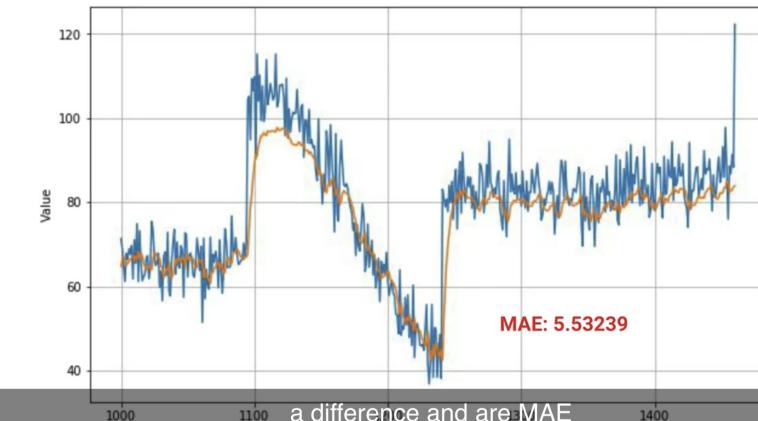


```
tf.keras.backer
tf.random.set_
np.random.seed(42)

train_set = windowed_
model = tf.keras.
tf.keras.laye
tf.keras.laye
tf.keras.laye
tf.keras.laye
])

lr_schedule = t
lambda epoch
optimizer = tf.
model.compile(
c
n
history = model
```

Coding LSTMs



deeplearning.ai

```
optimizer = tf.keras.optimizers.SGD(lr=5e-5, momentum=0.9)
model.compile(loss=tf.keras.losses.Huber(),
              optimizer=optimizer,
              metrics=[ "mae" ])
history = model.fit(dataset,epochs=my_epoch*4)
```

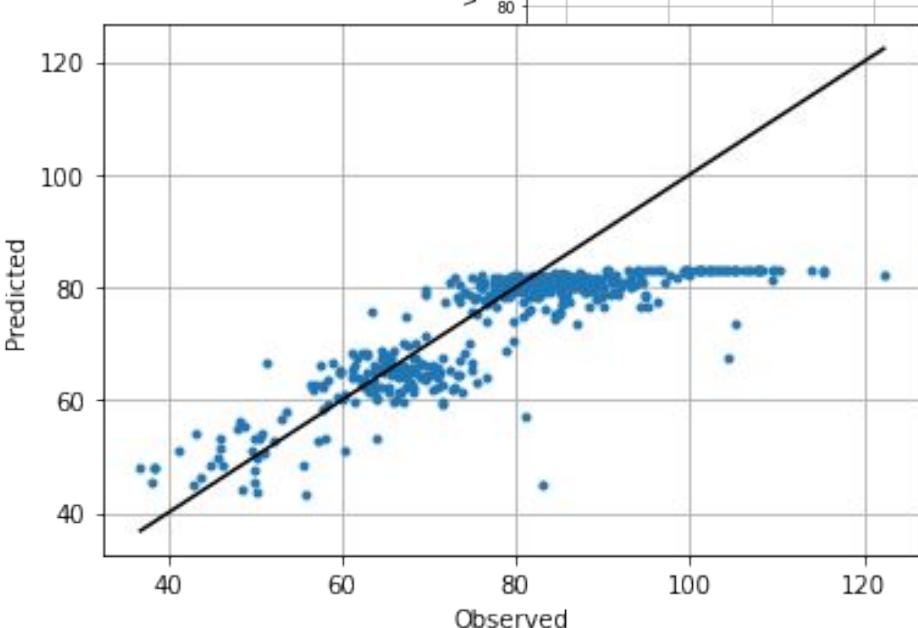
Source: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction>

Errata 5*10-6: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction/lecture/5W1Rw/rnn>

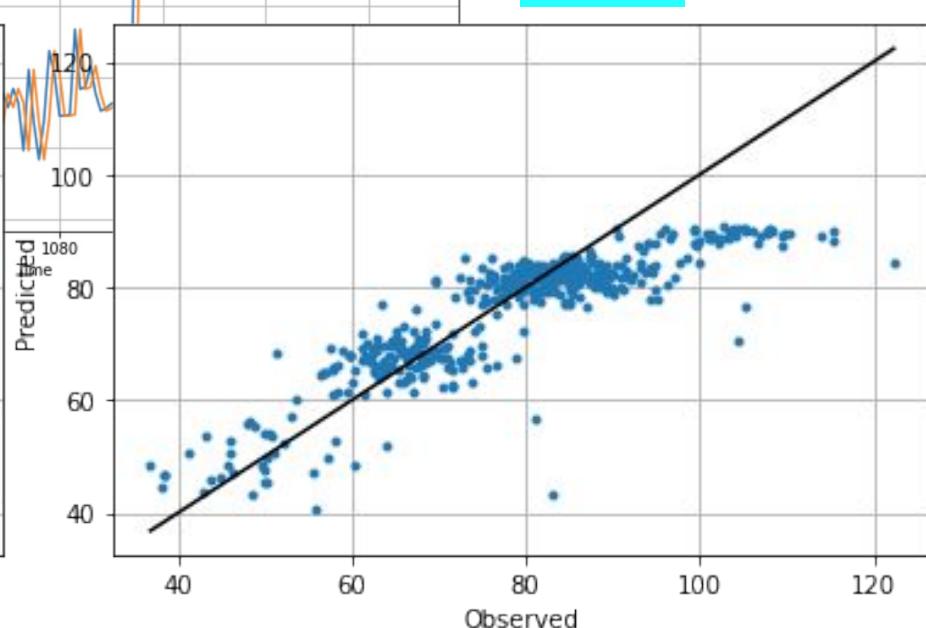
Errata MAE has gone down: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction/lecture/8u3wq/coding-lstms>

Recurrent Neural Networks for Time Series

LR: 5e-5



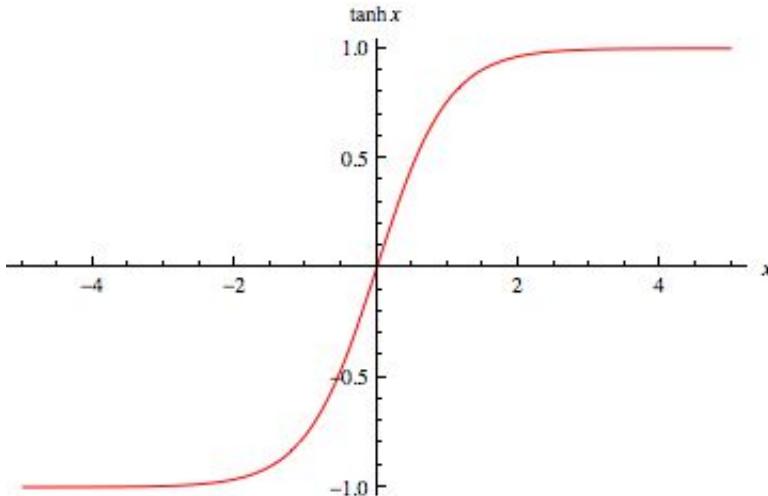
LR: 3e-6



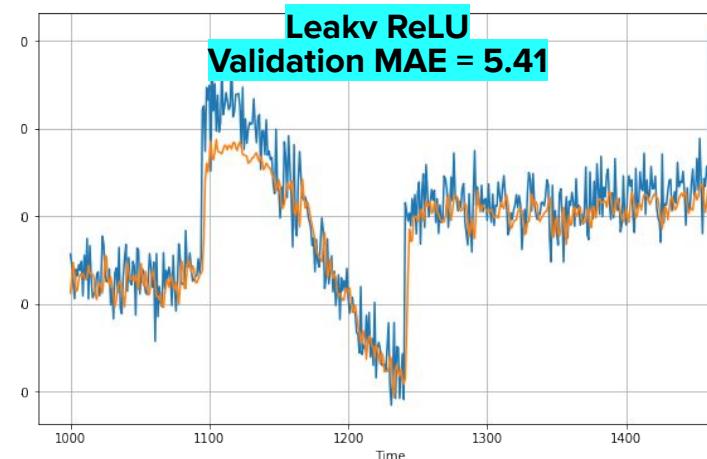
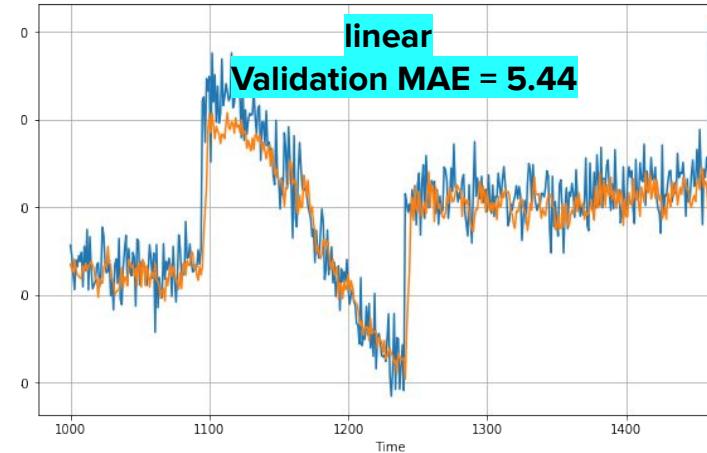
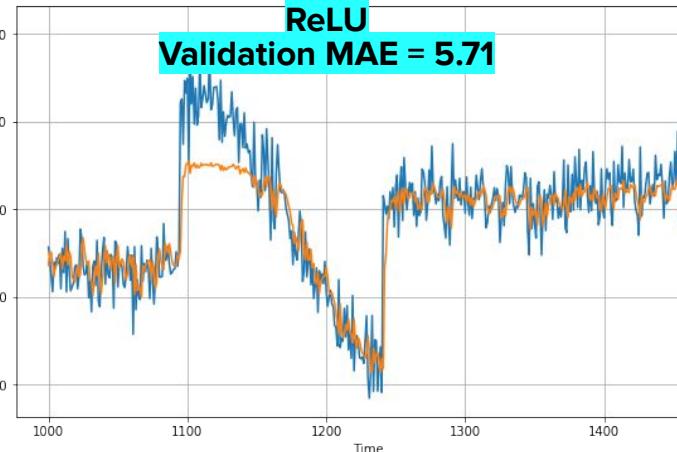
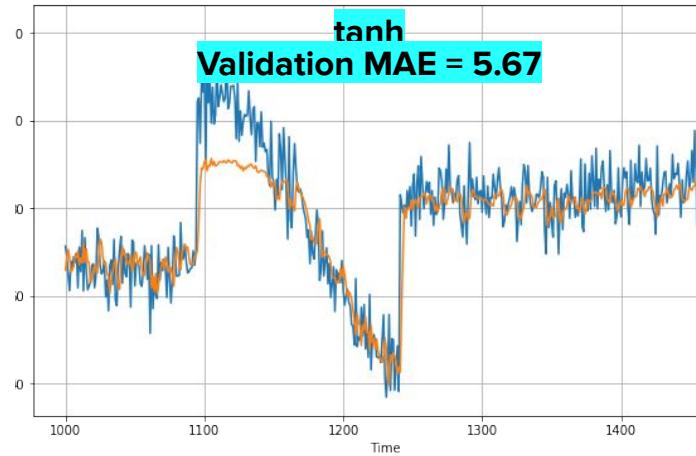
Recurrent Neural Networks for Time Series



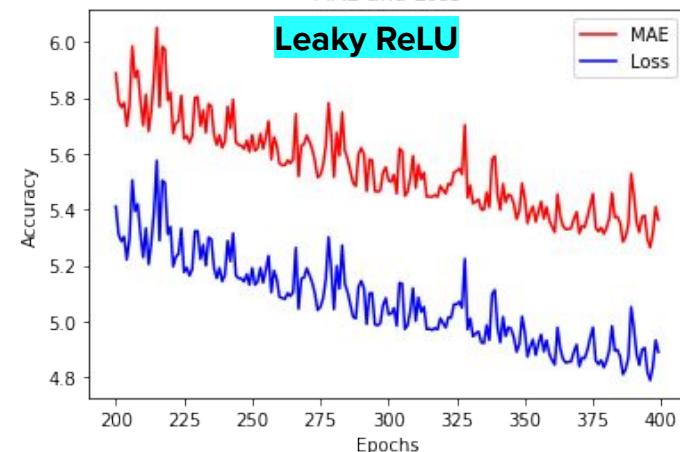
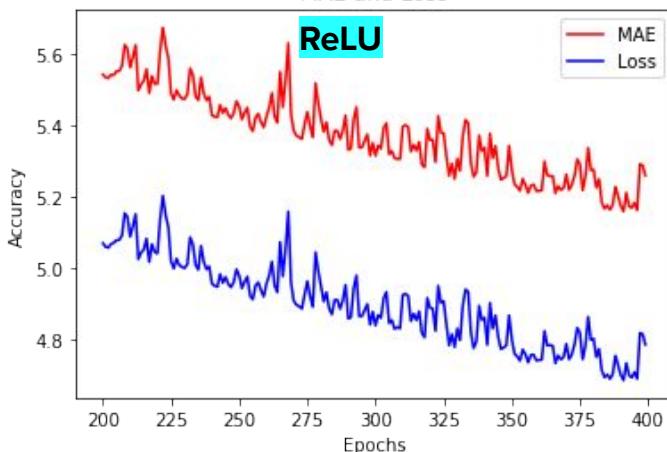
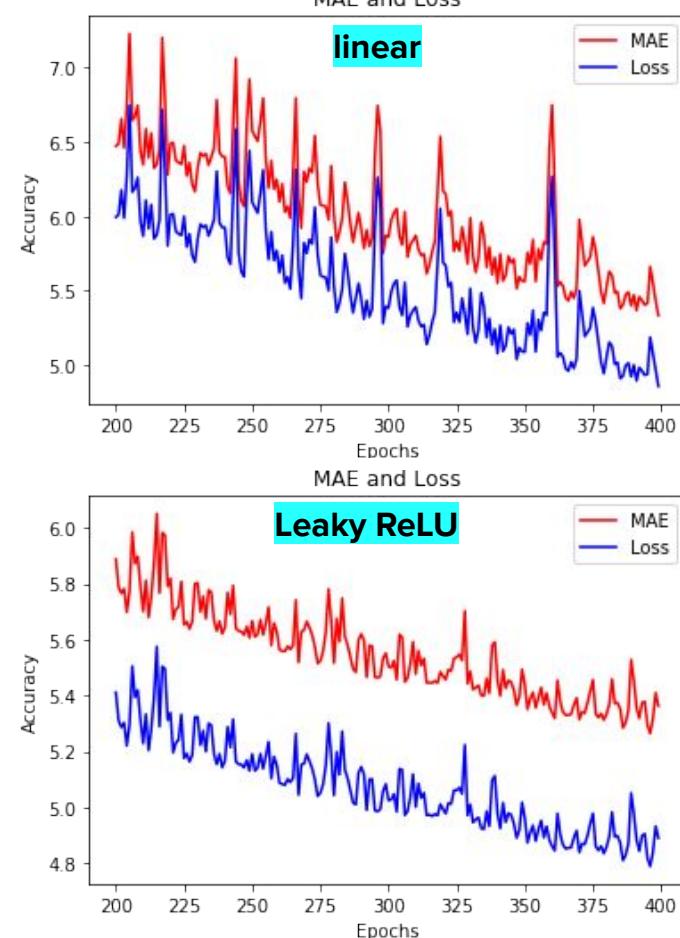
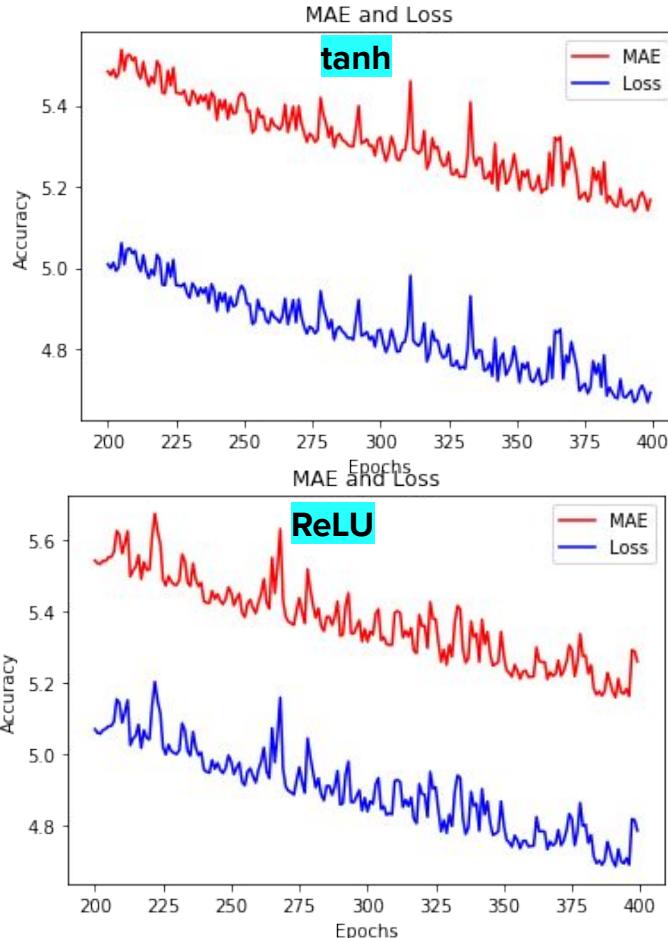
```
tf.keras.layers.SimpleRNN(  
    units, activation='tanh', use_bias=True, kernel_initializer='glor  
    recurrent_initializer='orthogonal', bias_initializer='zeros',  
    kernel_regularizer=None, recurrent_regularizer=None, bias_regularizer  
    activity_regularizer=None, kernel_constraint=None, recurrent_cons
```



Recurrent Neural Networks for Time Series



Recurrent Neural Networks for Time Series



Appendix:

A very quick overview of

TensorFlow in Practice Specialization

Course 1: Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning

Week 1: A New Programming Paradigm

Week 2: Introduction to Computer Vision

Week 3: Enhancing Vision with Convolutional Neural Networks

Week 4: Using Real-world Images

A New Programming Paradigm

```
model = keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
model.compile(optimizer='sgd', loss='mean_squared_error')

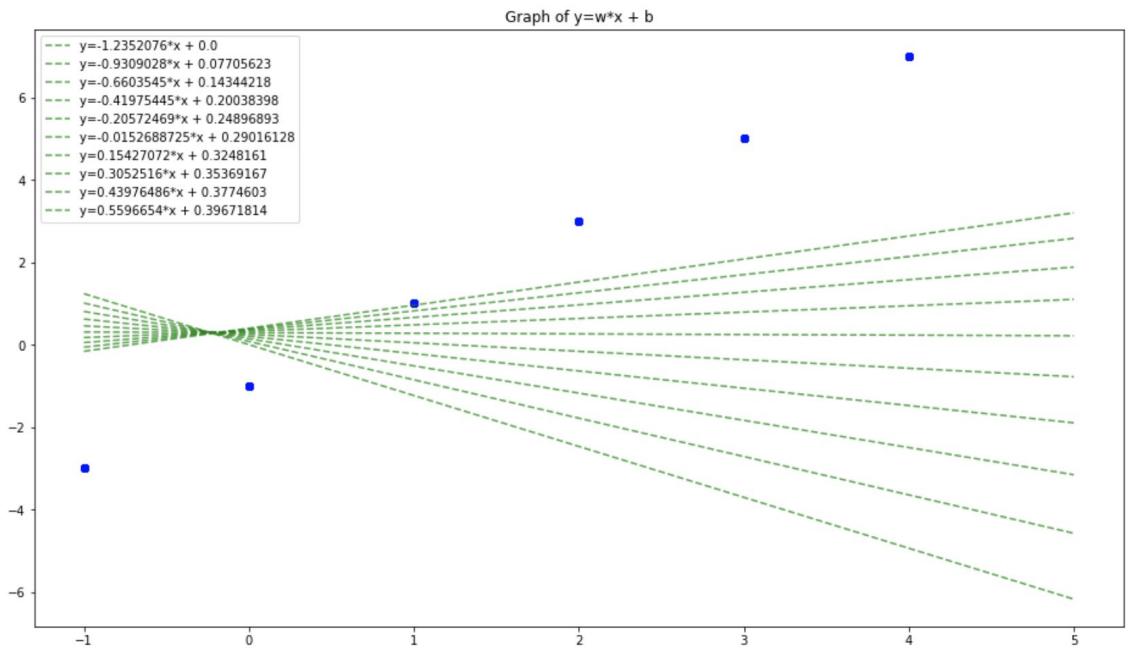
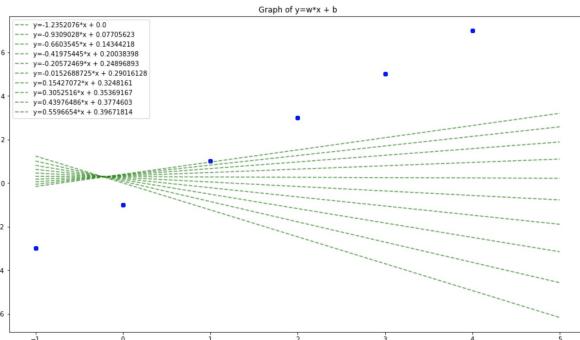
xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype=float)

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

print(model.predict([10.0]))
```

Content added after our 1st session, based on group feedback

```
: model.fit(xs, ys, epochs=10, callbacks=[get_weight_and_bias()])  
  
Epoch 1/10  
-1.2352076 0.0  
1/1 [=====] - 0s 2ms/step - loss: 45.3716  
Epoch 2/10  
-0.9309028 0.07705623  
1/1 [=====] - 0s 2ms/step - loss: 36.0725  
Epoch 3/10  
-0.6603545 0.14344218  
1/1 [=====] - 0s 2ms/step - loss: 28.7486  
Epoch 4/10  
-0.41975445 0.20038398  
1/1 [=====] - 0s 1ms/step - loss: 22.9789  
Epoch 5/10  
-0.20572469 0.24896893  
1/1 [=====] - 0s 1ms/step - loss: 18.4323  
Epoch 6/10  
-0.0152688725 0.29016128  
1/1 [=====] - 0s 1ms/step - loss: 14.8479  
Epoch 7/10  
0.15427072 0.3248161  
1/1 [=====] - 0s 1ms/step - loss: 12.0208  
Epoch 8/10  
0.3952516 0.35369167  
1/1 [=====] - 0s 1ms/step - loss: 9.7895  
Epoch 9/10  
0.43976486 0.3774603  
1/1 [=====] - 0s 1ms/step - loss: 8.0273  
Epoch 10/10  
0.5596654 0.39671814  
1/1 [=====] - 0s 1ms/step - loss: 6.6342  
: <tensorflow.python.keras.callbacks.History at 0x7f77aae00d0>
```



Our code is available on GitHub and on our Slack channel:
https://github.com/georgezoto/TensorFlow-in-Practice/blob/master/Course-1-Introduction-to-TensorFlow/C1W1_A_new_programming_paradigm_1.ipynb

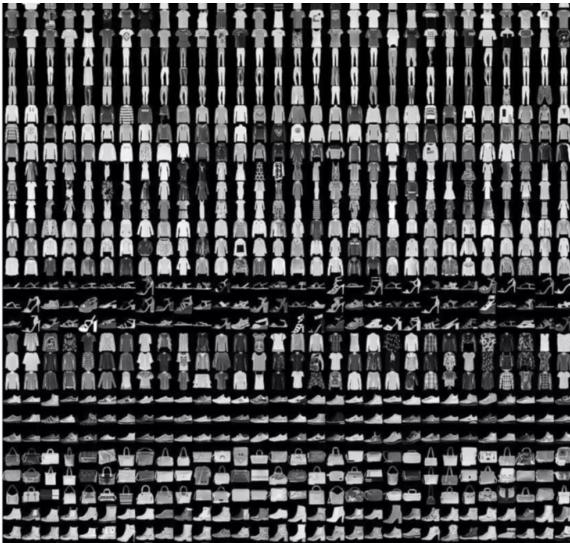
#tensorflow-in-practice-specialization

You created this channel on March 31st. This is the very beginning of the #tensorflow-in-practice-specialization channel.

An Introduction to computer vision

Fashion MNIST

- 70k Images
- 10 Categories
- Images are 28x28
- Can train a neural net!



deeplearning.ai

Fashion MNIST: <https://github.com/zalandoresearch/fashion-mnist>

[zalandoresearch / fashion-mnist](https://github.com/zalandoresearch/fashion-mnist)

Watch 289 ★ Unstar 7.5k Fork 1.7k

Code

Issues 19

Pull requests 2

Actions

Security

Insights

A MNIST-like fashion product database. Benchmark <http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/>

[mnist](#) [deep-learning](#) [benchmark](#) [machine-learning](#) [dataset](#) [computer-vision](#) [fashion](#) [fashion-mnist](#) [gan](#) [zalando](#)
[convolutional-neural-networks](#)

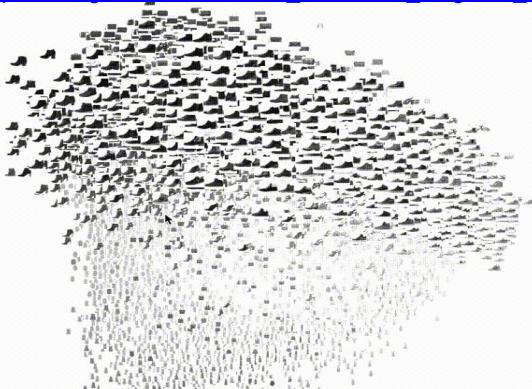
Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot



deeplearning.ai

Visualization: t-SNE

https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding



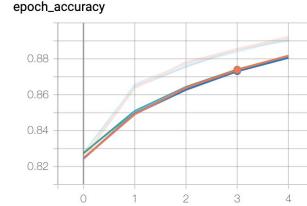
An Introduction to computer vision

```
# Define the Keras TensorBoard callback.
```

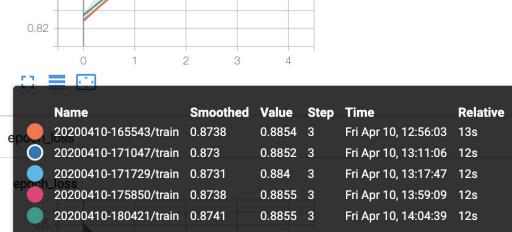
```
logdir = "logs/scalars/" + datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = keras.callbacks.TensorBoard(log_dir=logdir)
```

```
training_history = model.fit(training_images,
    training_labels,
    #batch_size = 64,
    #verbose = 0,
    epochs = 5,
    #validation_data = (test_images, test_labels),
    callbacks=[tensorboard_callback])
```

epoch_accuracy



epoch_loss



Extra Content - TensorBoard - issues in Colab

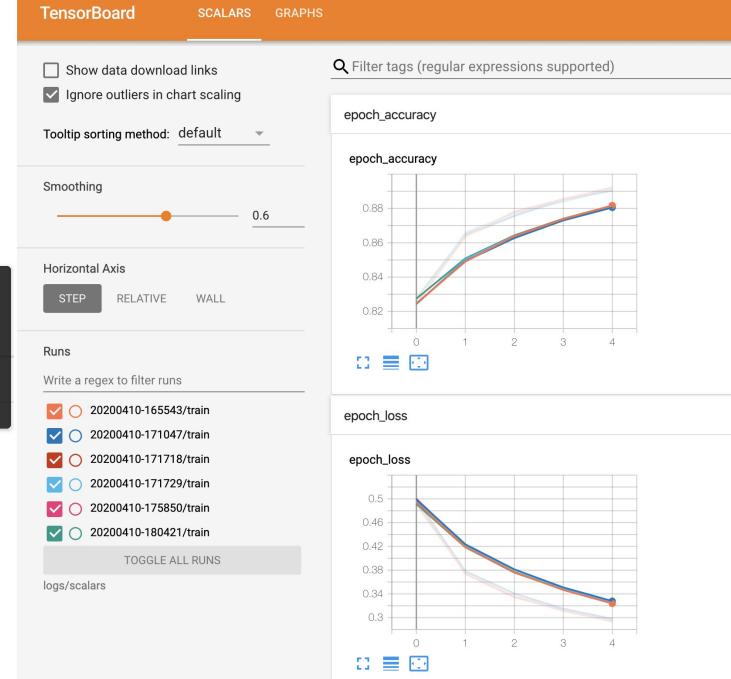
https://colab.research.google.com/github/tensorflow/tensorboard/blob/master/docs/scalars_and_keras.ipynb

Op-level graph

Start TensorBoard and wait a few seconds for the UI to load. Select the Graphs dashboard by tapping "Graphs" at the top.

```
[15] *tensorboard --logdir logs/scalars
```

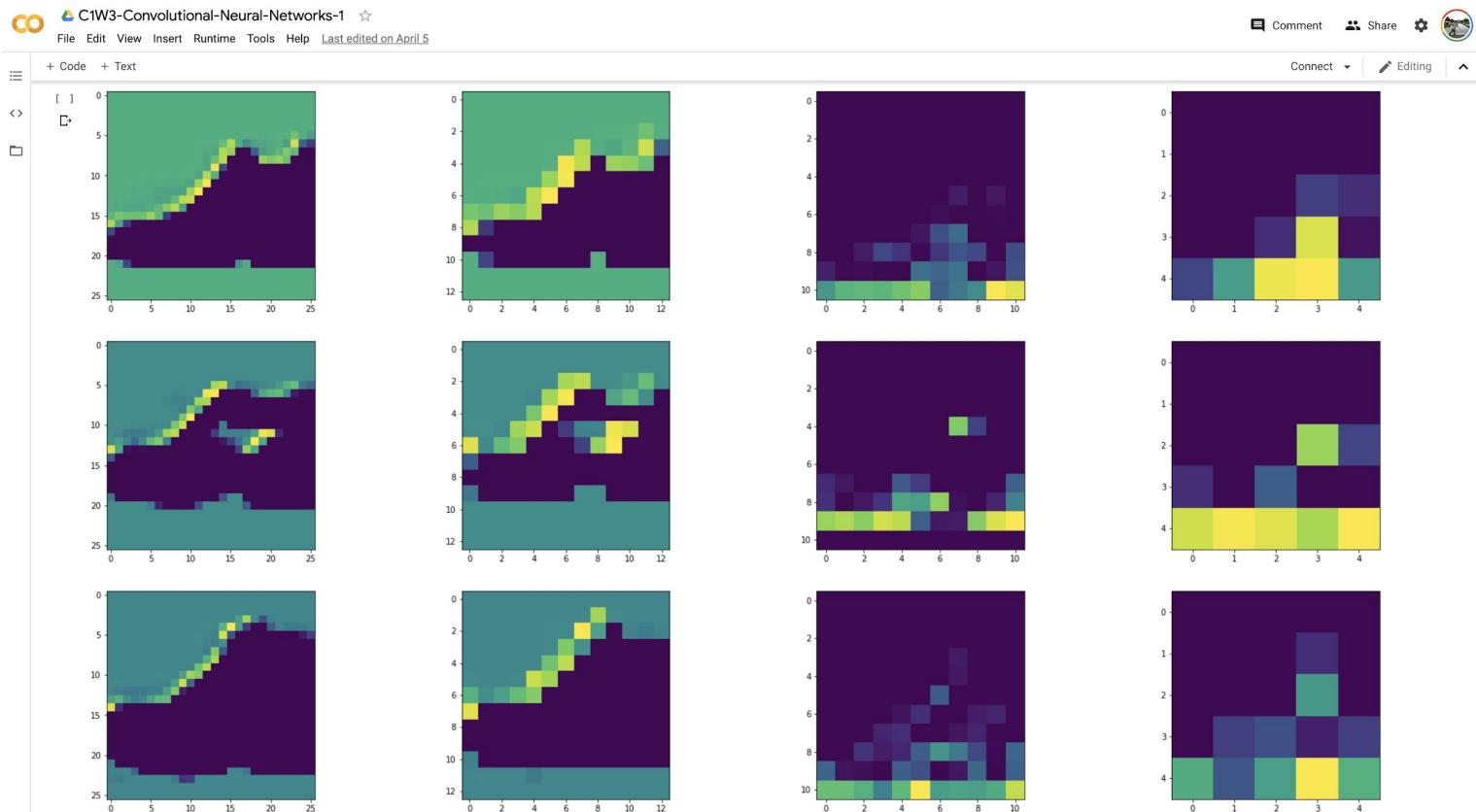
↳ Reusing TensorBoard on port 6006 (pid 178), started 1:08:27 ago. (Use '!kill 178' to kill it.)



As you can see we go from the raw pixels of the images to increasingly abstract and compact representations. The representations downstream start highlighting what the network pays attention to, and they show fewer and fewer features being "activated"; most are set to zero. This is called "sparsity". Representation sparsity is a key feature of deep learning.

These representations carry increasingly less information about the original pixels of the image, but increasingly refined information about the class of the image. You can think of a convnet (or a deep network in general) as an **information distillation pipeline**.

Enhancing Vision with Convolutional Neural Networks - Colab



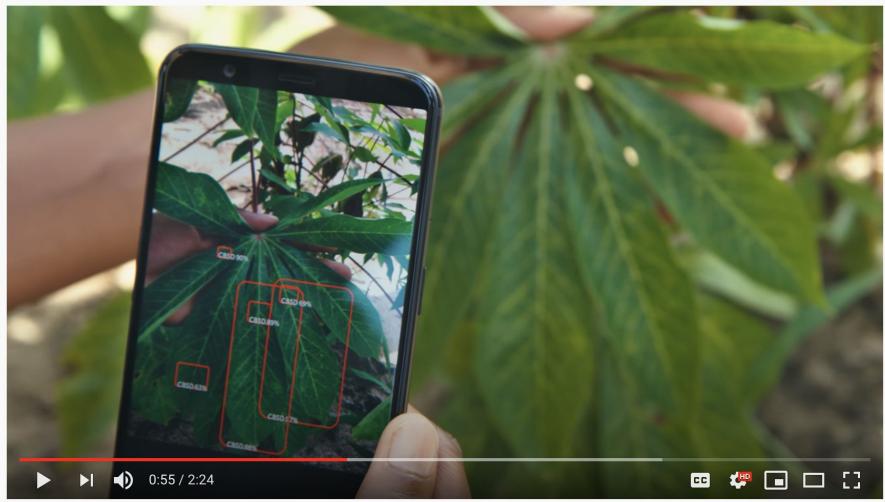
Check out these resources



TensorFlow: an ML platform for solving impactful and challenging problems

105,225 views • Mar 30, 2018

2.3K 25 SHARE SAVE ...



TensorFlow: an ML platform for solving impactful and challenging problems

105,225 views • Mar 30, 2018

2.3K 25 SHARE SAVE ...

Source: <https://www.youtube.com/watch?v=NIpS-DhayQA>

<https://plantvillage.psu.edu/>

<https://www.iita.org/>

<https://www.blog.google/technology/ai/ai-takes-root-helping-farmers-identify-diseased-plants/>

Course 2: Convolutional Neural Networks in TensorFlow

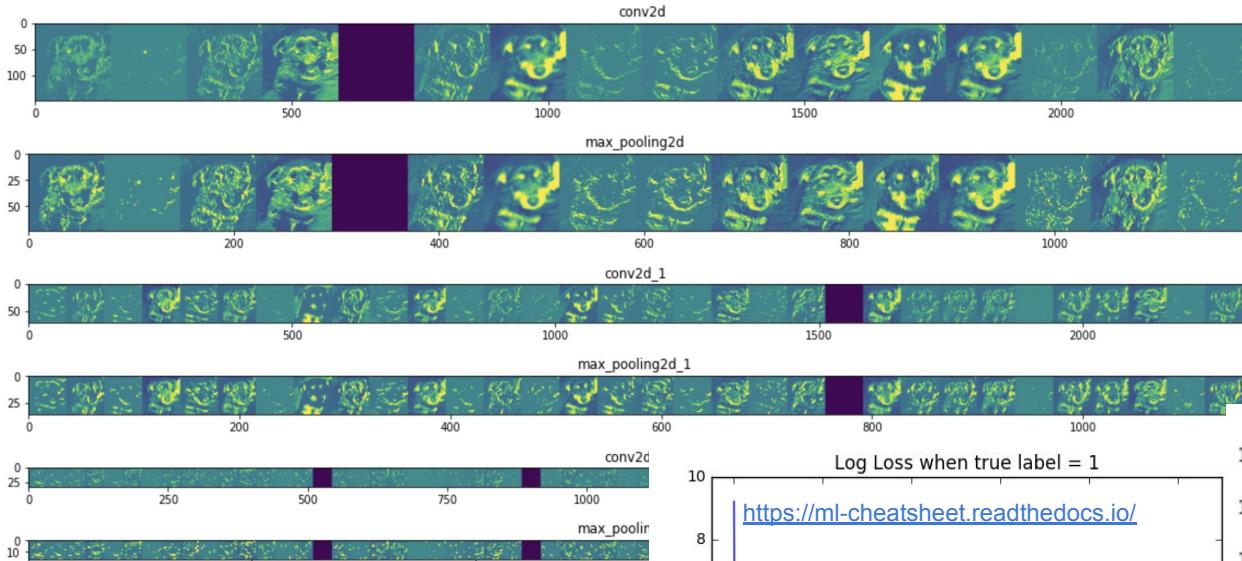
Week 1: Exploring a Larger Dataset

Week 2: Augmentation: A technique to avoid overfitting

Week 3: Transfer Learning

Week 4: Multiclass Classifications

Exploring a Larger Dataset

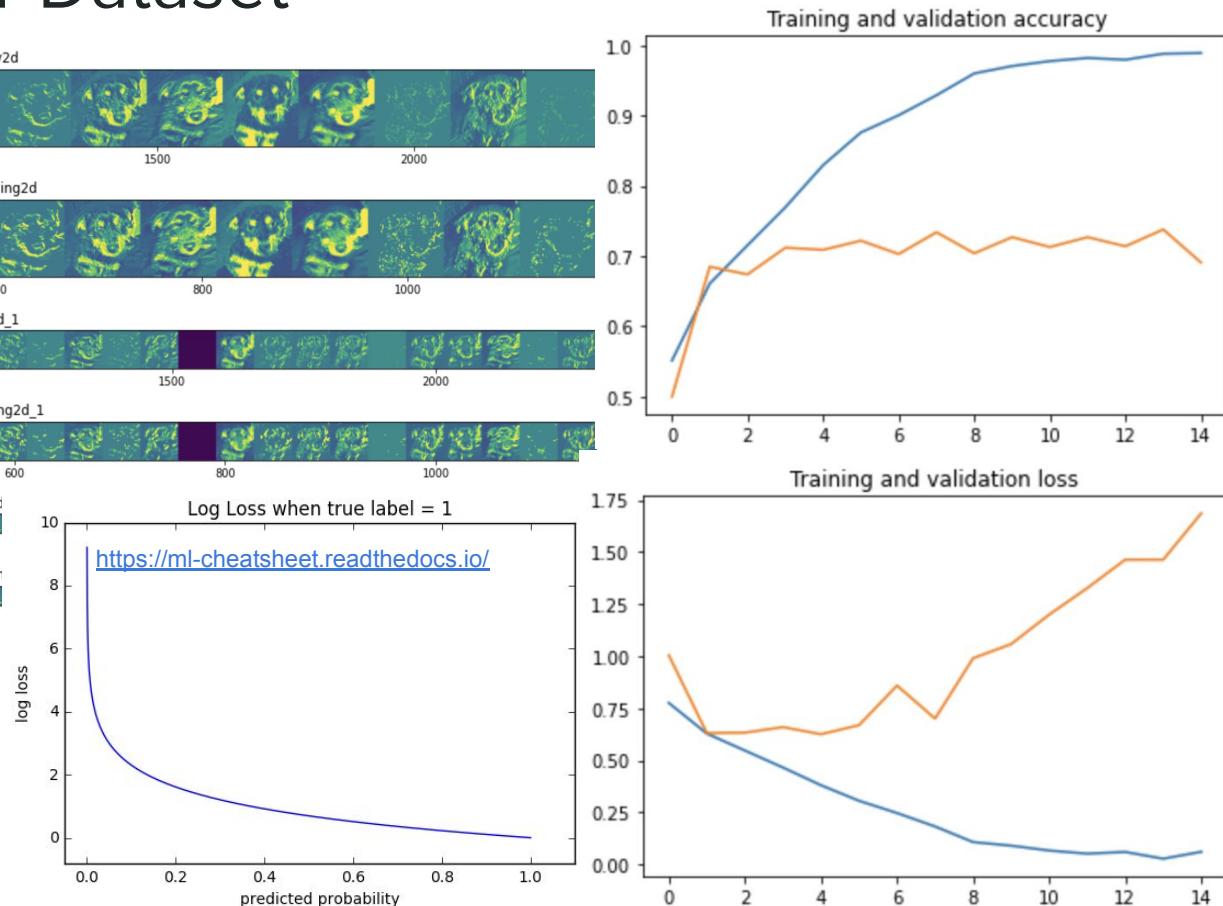


Bug fix: [layer.output for layer in model.layers[1:]] X

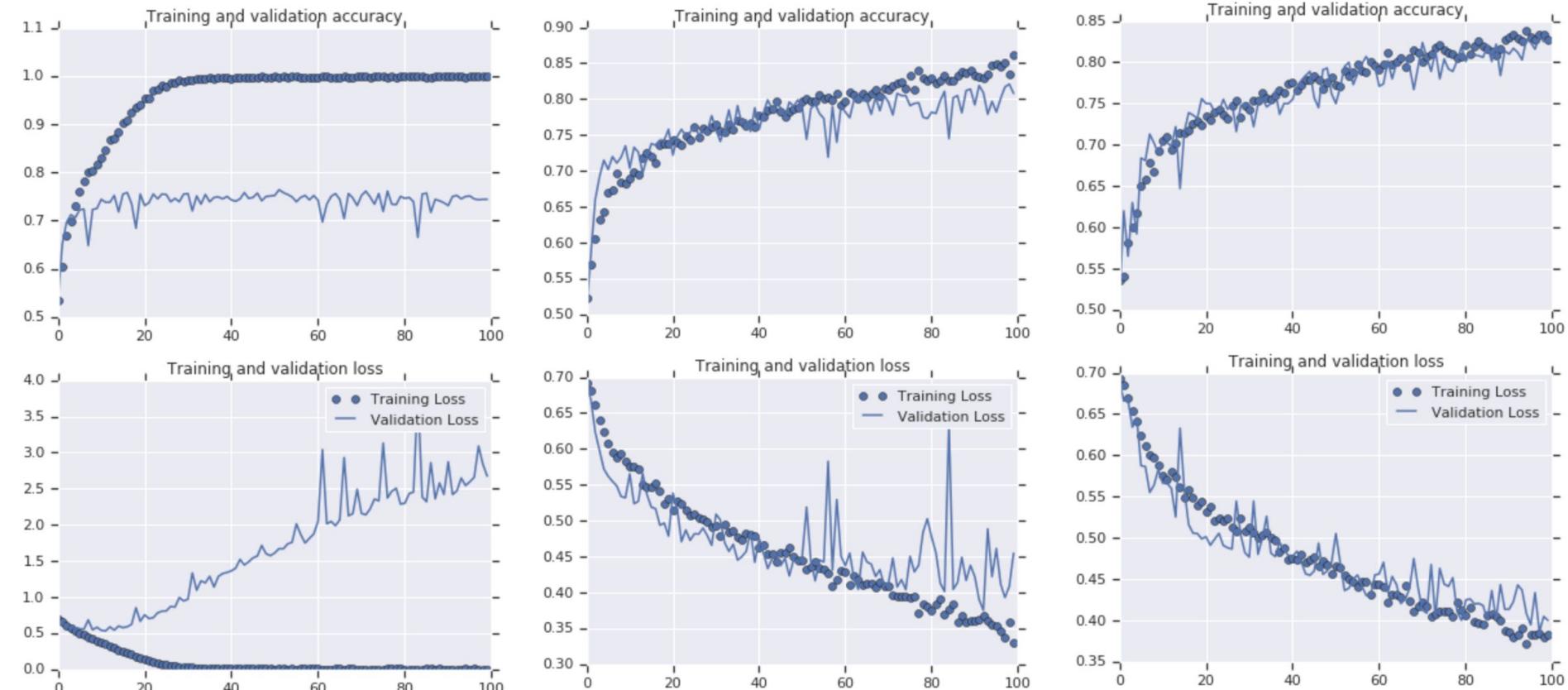
Model is badly overfitting!

Training Loss Decreasing

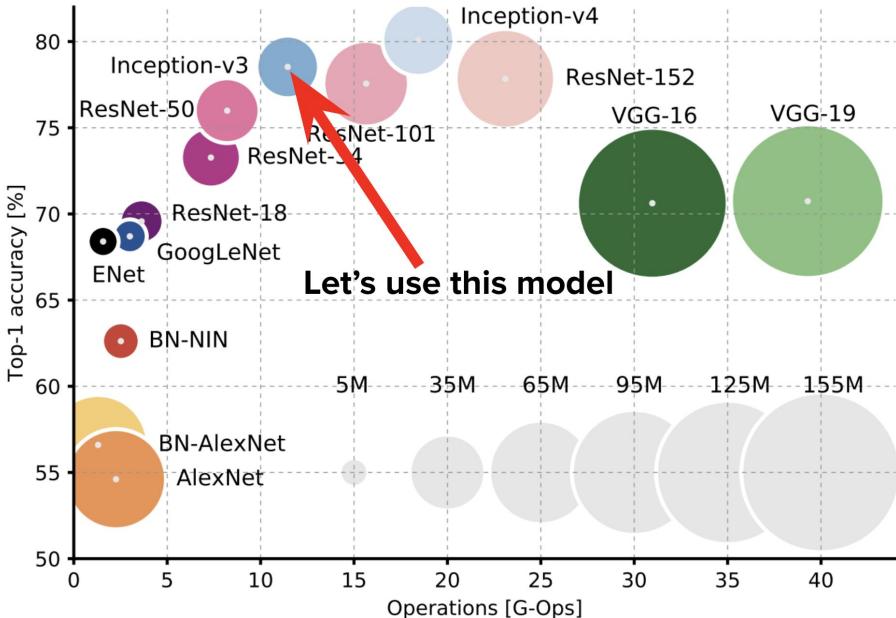
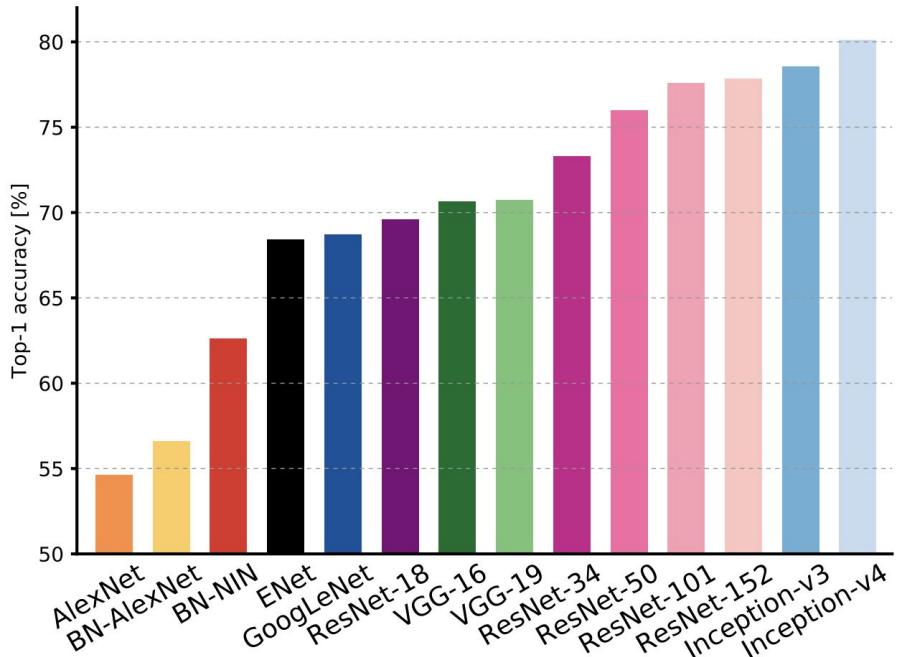
Validation Loss Increasing



Augmentation and Dropout: Techniques to avoid overfitting



Transfer Learning - Extra Content - Classic Networks



Extra Content - Regularization

Logistic regression

$$\min_{w,b} J(w,b) \quad w \in \mathbb{R}^n, b \in \mathbb{R}$$

$$J(w,b) = \frac{1}{m} \sum_{i=1}^m L(y_i, \hat{y}_i) + \frac{\lambda}{2m} \|w\|_2^2$$

L₂ regularization $\|w\|_2^2 = \sum_{j=1}^n w_j^2 = w^T w \leftarrow$

L₁ regularization $\frac{\lambda}{2m} \sum_{j=1}^n |w_j| = \frac{\lambda}{2m} \|w\|_1$

λ = regularization parameter
 lambda
 lambd
 onit

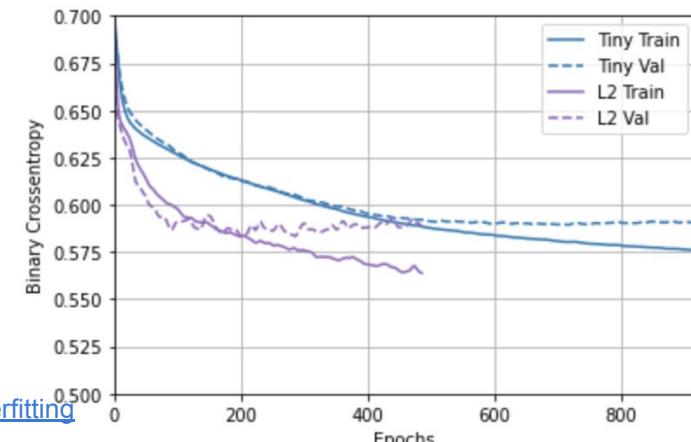
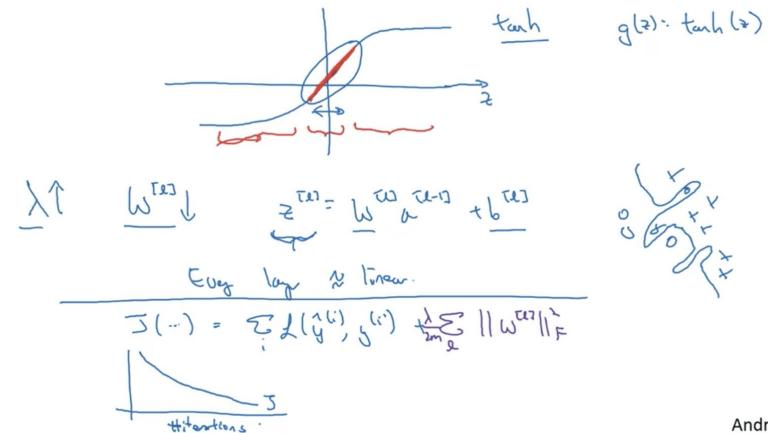
w will be sparse

Andrew Ng

```
from tensorflow.keras import regularizers
```

```
l2_model = tf.keras.Sequential([
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001),
                input_shape=(FEATURES,)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(512, activation='elu',
                kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(1)
])
```

How does regularization prevent overfitting?



Source: <https://www.coursera.org/learn/deep-neural-network/lecture/Srsrc/regularization>

<https://www.coursera.org/learn/deep-neural-network/lecture/T6OJj/why-regularization-reduces-overfitting>

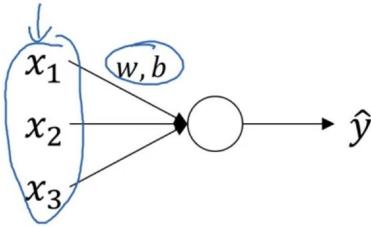
https://www.tensorflow.org/tutorials/keras/overfit_and_underfit

Extra Content - Batch Normalization

Normalizing inputs to speed up learning

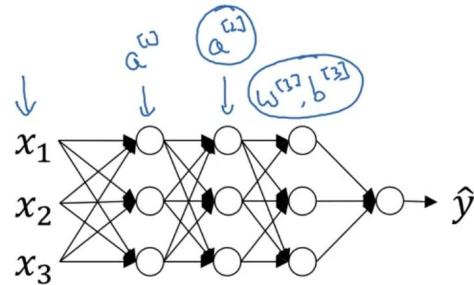
TensorFlow > API > TensorFlow Core v2.1.0 > Python

`tf.keras.layers.BatchNormalization`



$$\begin{aligned}\mu &= \frac{1}{m} \sum_i x^{(i)} \\ X &= X - \mu \\ \sigma^2 &= \frac{1}{m} \sum_i (x^{(i)} - \mu)^2 \quad \leftarrow \text{element-wise} \\ Z &= X / \sigma^2\end{aligned}$$

```
graph LR; Z((Z)) --> Y_hat((y-hat))
```



Can we normalize $a^{[2]}$ so
as to train $w^{[2]}, b^{[2]}$ faster
Normalize $\underline{z}^{[2]}$.

Normalize the activations of the previous layer at each batch, i.e. applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

Batch normalization differs from other layers in several key aspects:

Source:

<https://www.coursera.org/learn/deep-neural-network/lecture/4ptp2/normalizing-activations-in-a-network>

https://www.tensorflow.org/api_docs/python/tf/keras/layers/BatchNormalization

Andrew Ng

Extra Content - Overfitting in Neural Networks

To recap: here are the most common ways to prevent overfitting in neural networks:

- Get more training data.
- Reduce the capacity of the network.
- Add weight regularization.
- Add dropout.

Two important approaches not covered in this guide are:

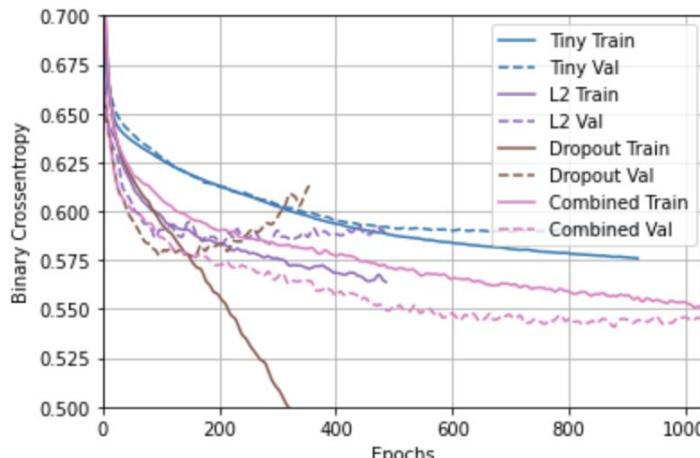
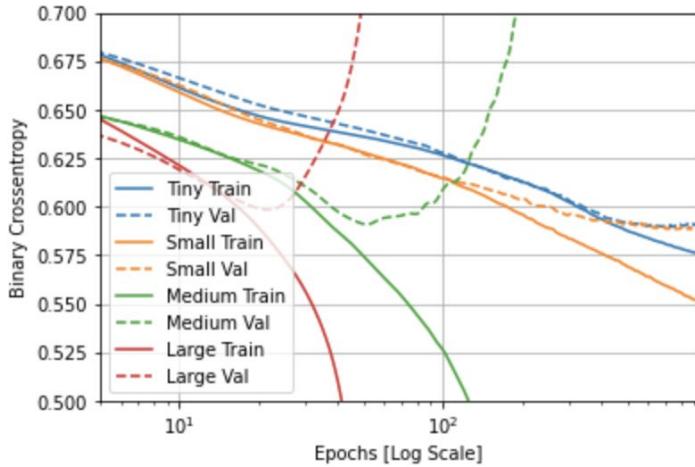
- data-augmentation
- batch normalization

Remember that each method can help on its own, but often combining them can be even more effective.

- Data Augmentation
- Dropout
- Early Stopping
- Ensembling
- Injecting Noise
- L1 Regularization
- L2 Regularization

The Higgs Dataset:
11M samples, 28
features, binary class

Total params:
Tiny: 0.5K
Small: 0.7K
Medium: 10K
Large: 800K



Source: https://www.tensorflow.org/tutorials/keras/overfit_and_underfit
<https://ml-cheatsheet.readthedocs.io/en/latest/regularization.html>

Course 3: Natural Language Processing in TensorFlow

Week 1: Sentiment in text

Week 2: Word Embeddings

Week 3: Sequence models

Week 4: Sequence models and literature

Word Embeddings

audio	"fashion_mnist"	text
	"horses_or_humans"	
	"image_label_folder"	"cnn_dailymail"
	"imagenet2012"	"glue"
image	"imagenet2012_corrupted"	"imdb_reviews"
	"kmnist"	"lm1b"
	"lsun"	"multi_nli"
"abstract_reasoning"	"mnist"	"squad"
"caltech101"	"omniglot"	"wikipedia"
"cats_vs_dogs"	"open_images_v4"	"xnli"
"celeb_a"	"oxford_iit_pet"	
"celeb_a_hq"	"quickdraw_bitmap"	translate
"cifar10"	"rock_paper_scissors"	
"cifar100"	"shapes3d"	"flores"
"cifar10_corrupted"	"smallnorb"	"para_crawl"
"coco2014"	"sun397"	"ted_hrlr_translate"
"colorectal_histology"	"svhn_cropped"	"ted_multi_translate"
"cycle_gan"	"tf_flowers"	"wmt15_translate"
"diabetic_retinopathy..."		"wmt16_translate"
"dsprites"		"wmt17_translate"
"dtd"		"wmt18_translate"
"emnist"		"wmt19_translate"
		structured
	"higgs"	
	"iris"	
	"titanic"	

Source:

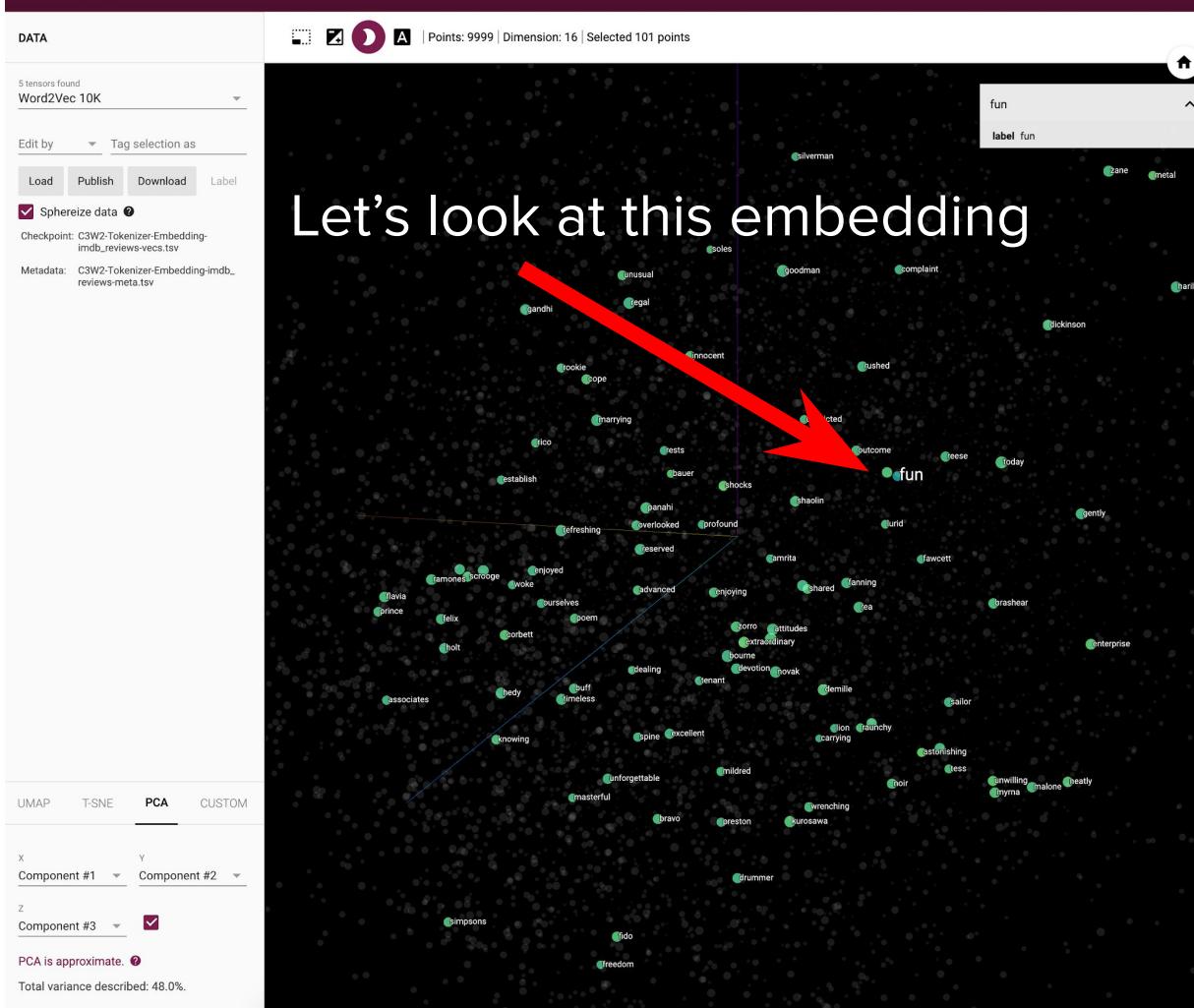
<https://www.tensorflow.org/datasets/catalog/overview>

<https://github.com/tensorflow/datasets>

https://www.tensorflow.org/datasets/catalog/imdb_reviews

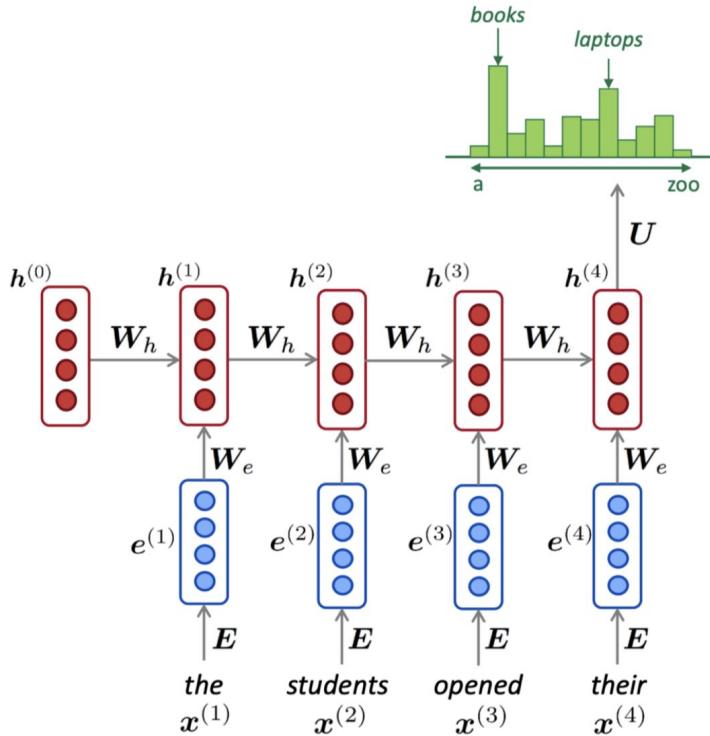
TensorFlow Datasets

Text	blimp
	c4 (manual)
	cfq
	civil_comments
	cos_e
	definite_pronoun_resolution
	eraser_multi_rc
	esnli
	gap
	glue
	imdb_reviews
	librispeech_lm
	lm1b
	math_dataset
	movie rationales
	multi_nli
	multi_nli_mismatch
	natural_questions
	qa4mre
	scan
	scicite
	snli
	squad
	super_glue
	tiny_shakespeare
	trivia_qa
	web_questions
	wiki40b
	wikipedia
	xnli
	yelp_polarity_reviews

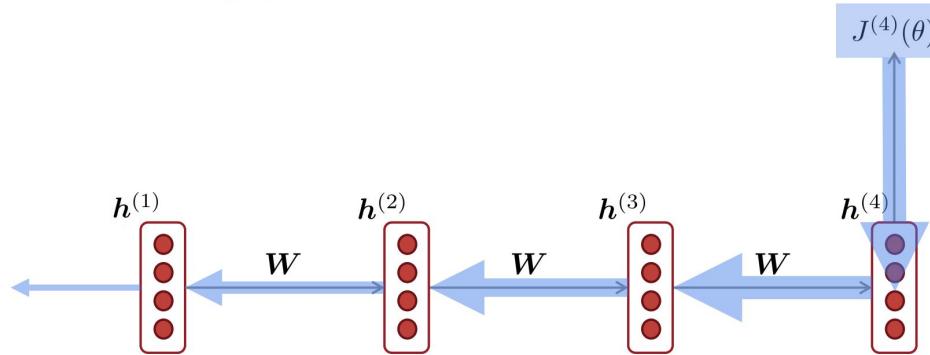


Sequence models - new

$$\hat{y}^{(4)} = P(\mathbf{x}^{(5)} | \text{the students opened their})$$



Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial h^{(4)}}{\partial h^{(3)}} \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

What happens if these are small?

Vanishing gradient problem:
When these are small, the gradient signal gets smaller and smaller as it backpropagates further

Sequence models - new

Long Short-Term Memory (LSTM)

We have a sequence of inputs $x^{(t)}$, and we will compute a sequence of hidden states $h^{(t)}$ and cell states $c^{(t)}$. On timestep t :

Forget gate: controls what is kept vs forgotten, from previous cell state

Input gate: controls what parts of the new cell content are written to cell

Output gate: controls what parts of cell are output to hidden state

New cell content: this is the new content to be written to the cell

Cell state: erase ("forget") some content from last cell state, and write ("input") some new cell content

Hidden state: read ("output") some content from the cell

Sigmoid function: all gate values are between 0 and 1

$$f^{(t)} = \sigma(W_f h^{(t-1)} + U_f x^{(t)} + b_f)$$

$$i^{(t)} = \sigma(W_i h^{(t-1)} + U_i x^{(t)} + b_i)$$

$$o^{(t)} = \sigma(W_o h^{(t-1)} + U_o x^{(t)} + b_o)$$

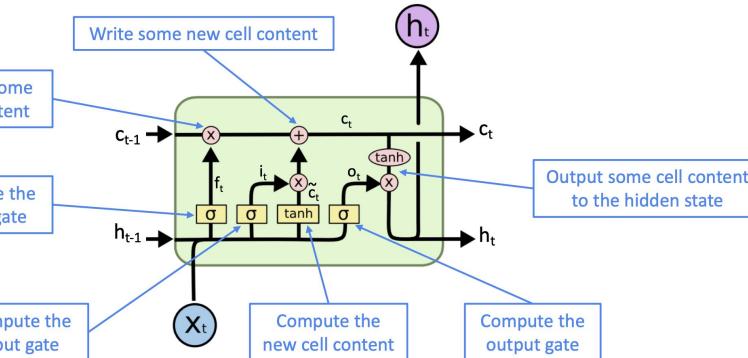
$$\tilde{c}^{(t)} = \tanh(W_c h^{(t-1)} + U_c x^{(t)} + b_c)$$

$$c^{(t)} = f^{(t)} \circ c^{(t-1)} + i^{(t)} \circ \tilde{c}^{(t)}$$

$$h^{(t)} = o^{(t)} \circ \tanh c^{(t)}$$

Gates are applied using element-wise product

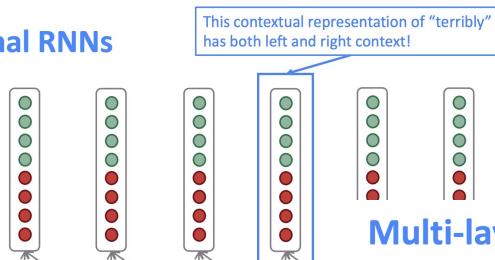
All these are vectors of same length n



Sequence models - new

Bidirectional RNNs

Concatenated hidden states



Backward RNN

Forward RNN

37

Multi-layer RNNs

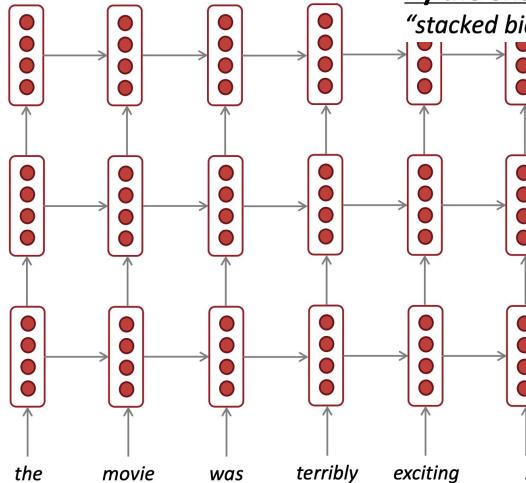
RNN layer 3

RNN layer 1

42

The hidden states from RNN layer i are the inputs to RNN layer $i+1$

By the end of the course: You will understand phrases like
"stacked bidirectional LSTM with residual connections and self-attention"



Source: <http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture07-fancy-rnn.pdf>
<http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture06-rnnlm.pdf>

A note on terminology

The RNN described in this lecture = simple/vanilla/Elman RNN

Next lecture: You will learn about other RNN flavors



Sequence models - new

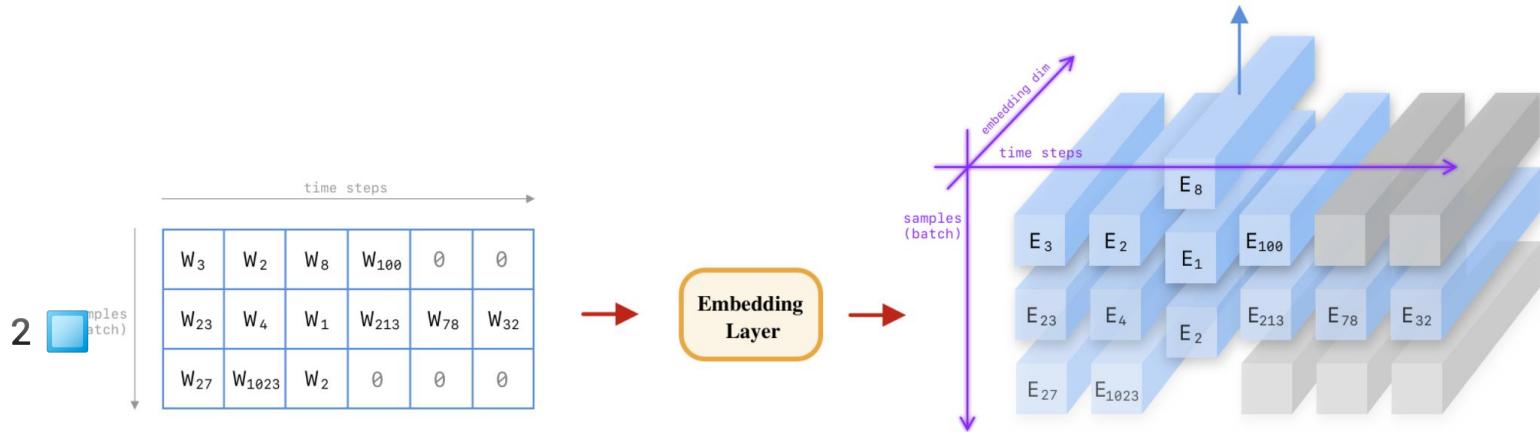


Figure 2. E_i is the embedding vector for word W_i. This tensor has [batch = 3, max_len = 6, features = 300] shape, and is ready to be fed into an RNN network directly.

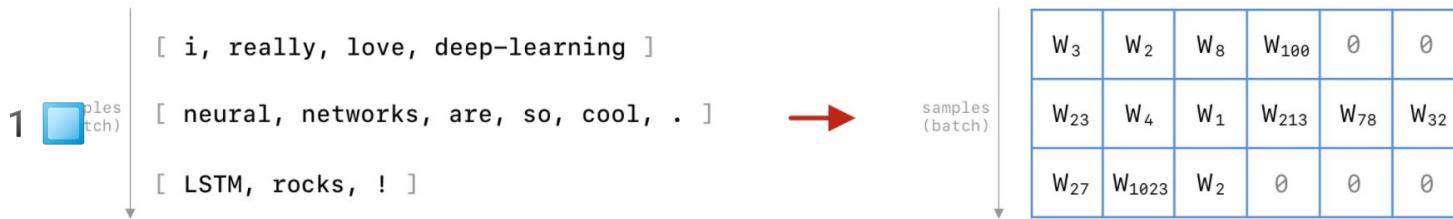
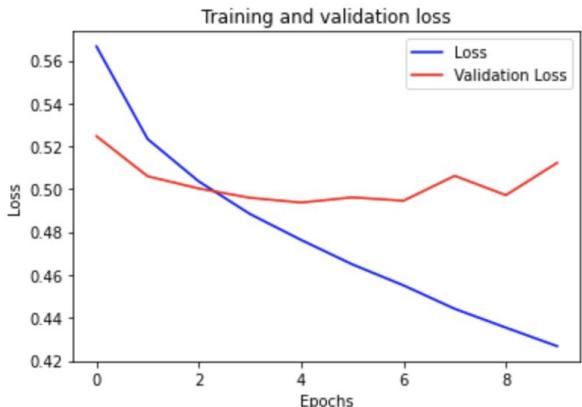
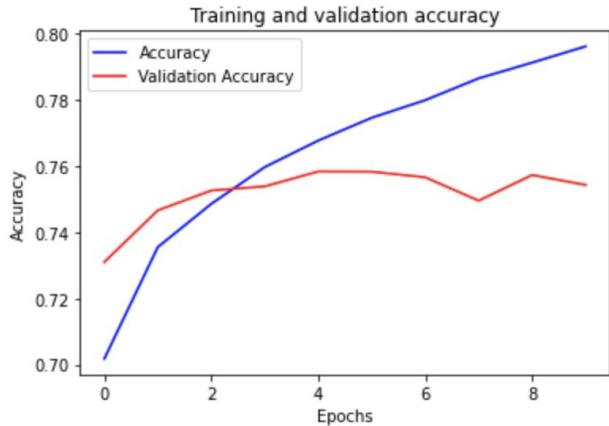


Figure 1. Batch size = 3, and W_i = Word ID assigned by the vocabulary, and 0 is reserved for padding; Here max_len = 6, and sentence 1 and 3 are padded to match up with the longest sentence

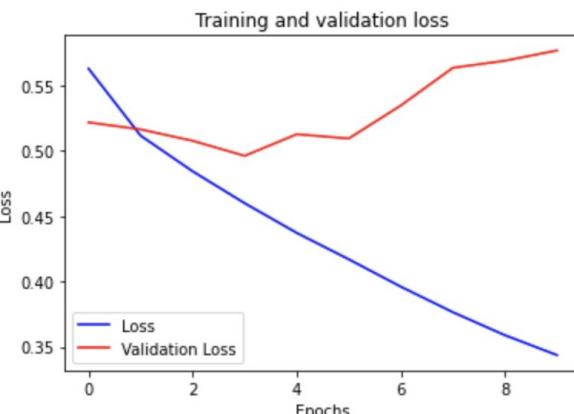
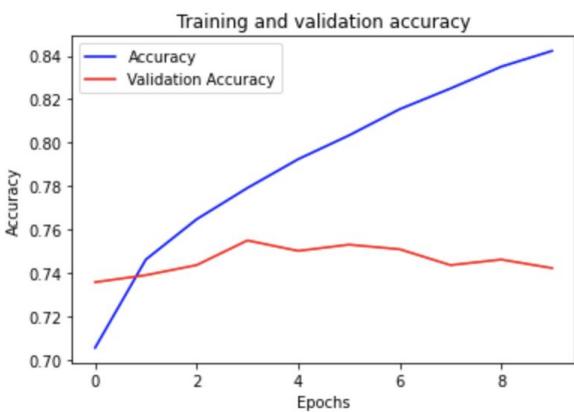
Source: https://iust-deep-learning.github.io/972/static_files/assignments/assignment_05_preview.html

Sequence models

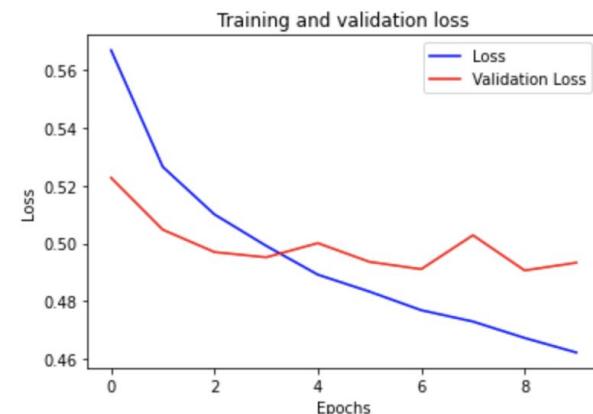
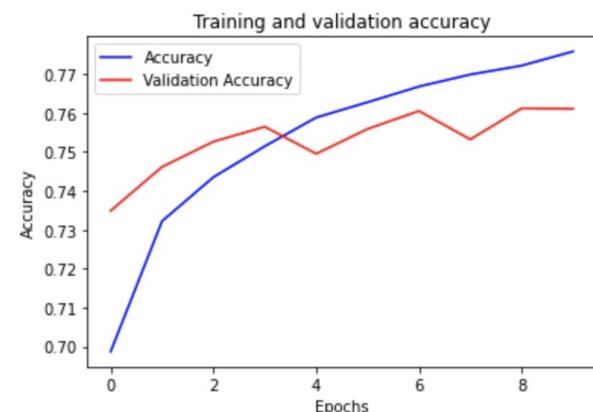
Conv1D, GlobalAveragePooling1D()



2 x Conv1D, GAP1D(), Dropout



Dropout, Conv1D, MaxP1D(), LSTM(64)

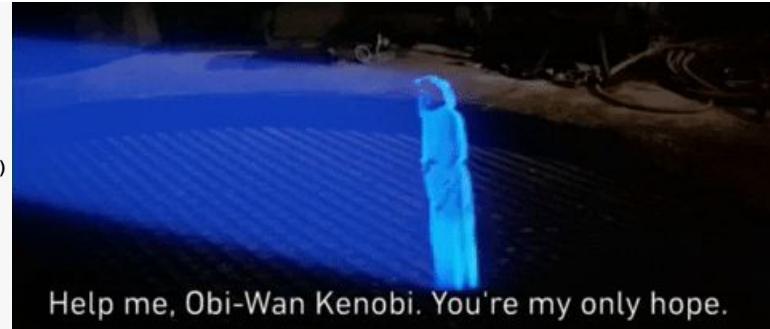


Sequence models and literature

```
seed_text = "Help me Obi Wan Kenobi, you're my only hope"
next_words = 100

for _ in range(next_words):
    token_list = tokenizer.texts_to_sequences([seed_text])[0]
    token_list = pad_sequences([token_list], maxlen=max_sequence_len-1, padding='pre')
    predicted = model.predict_classes(token_list, verbose=0)
    output_word = ""
    for word, index in tokenizer.word_index.items():
        if index == predicted:
            output_word = word
            break
    seed_text += " " + output_word
print(seed_text)

tf.keras.preprocessing.sequence.pad_sequences(
    sequences, maxlen=None, dtype='int32', padding='pre', truncating='pre',
    value=0.0
)
```



Help me, Obi-Wan Kenobi. You're my only hope.

shakespeare thyself away desired be cross afford twain

All Images News Shopping Videos More Settings Tools

About 1,530,000 results (0.85 seconds)

www.opensourceshakespeare.org > views > sonnets > sonnet_view ▾

[View Shakespeare sonnets \(OpenSourceShakespeare.org\)](#)

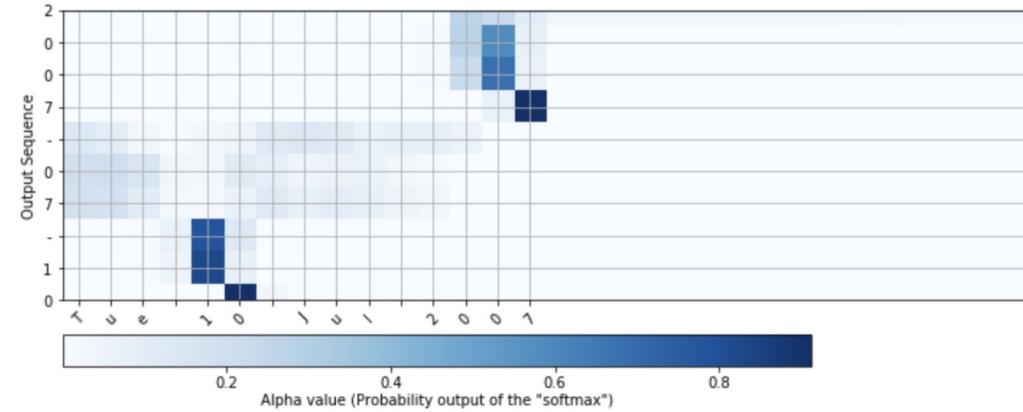
SONNET I. From fairest creatures we desire increase, ... So thou, thyself out-going in thy noon,
Unlook'd ... To him that bears the strong offence's cross. ... And that thou teachest how to make
one twain, ... And found it in thy cheek; he can afford

Help me Obi Wan Kenobi, you're my only hope thyself away desired be cross afford twain date of men ' doth groan ' be free free live away away your days here brought of one one more must grow of free ride hits green thee burn pleasure pleasure me of days much brought of pleasure pleasure of pleasure green die burn'd hits dwells beard ward fall by days another rolling ' doth ride ride ride ride ' her view free grew to ride cherish plight grow thus days did tell away away night exchanged mad so meant free his senses can alive pleasure ride ride light decease translate must

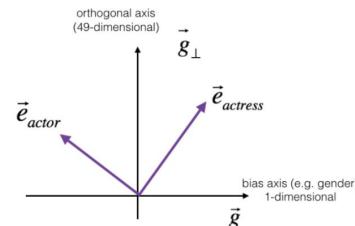
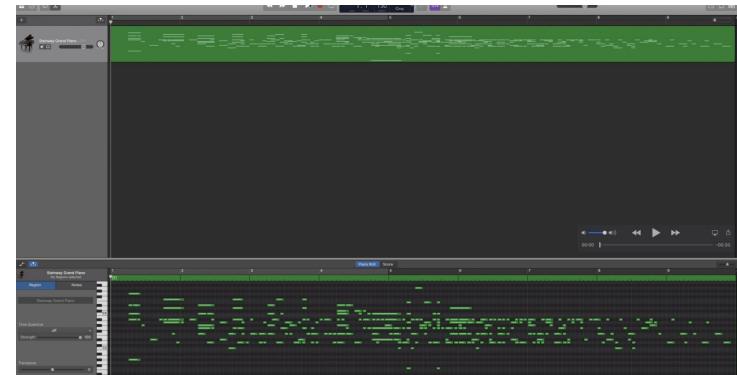
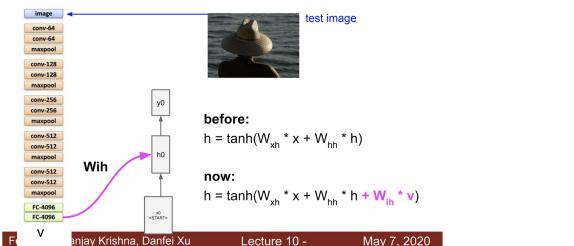
Extra Content - cool ways to use RNNs, Bias, Attention, Text and Vision

```
text = EXAMPLES[3]
pre_plot(model, human_vocab, inv_machine_vocab, text, num = 7, n_s = 64);
```

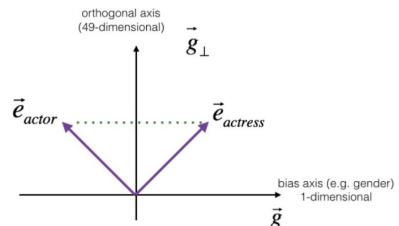
Input Tue 10 Jul 2007
Output 2007-07-10



Sequence models and computer vision



before equalizing,
"actress" and "actor" differ
in many ways beyond the
direction of \vec{g}



after equalizing,
"actress" and "actor" differ
only in the direction of \vec{g} , and further
are equal in distance from \vec{g}_\perp

Course 4: Sequences, Time Series and Prediction

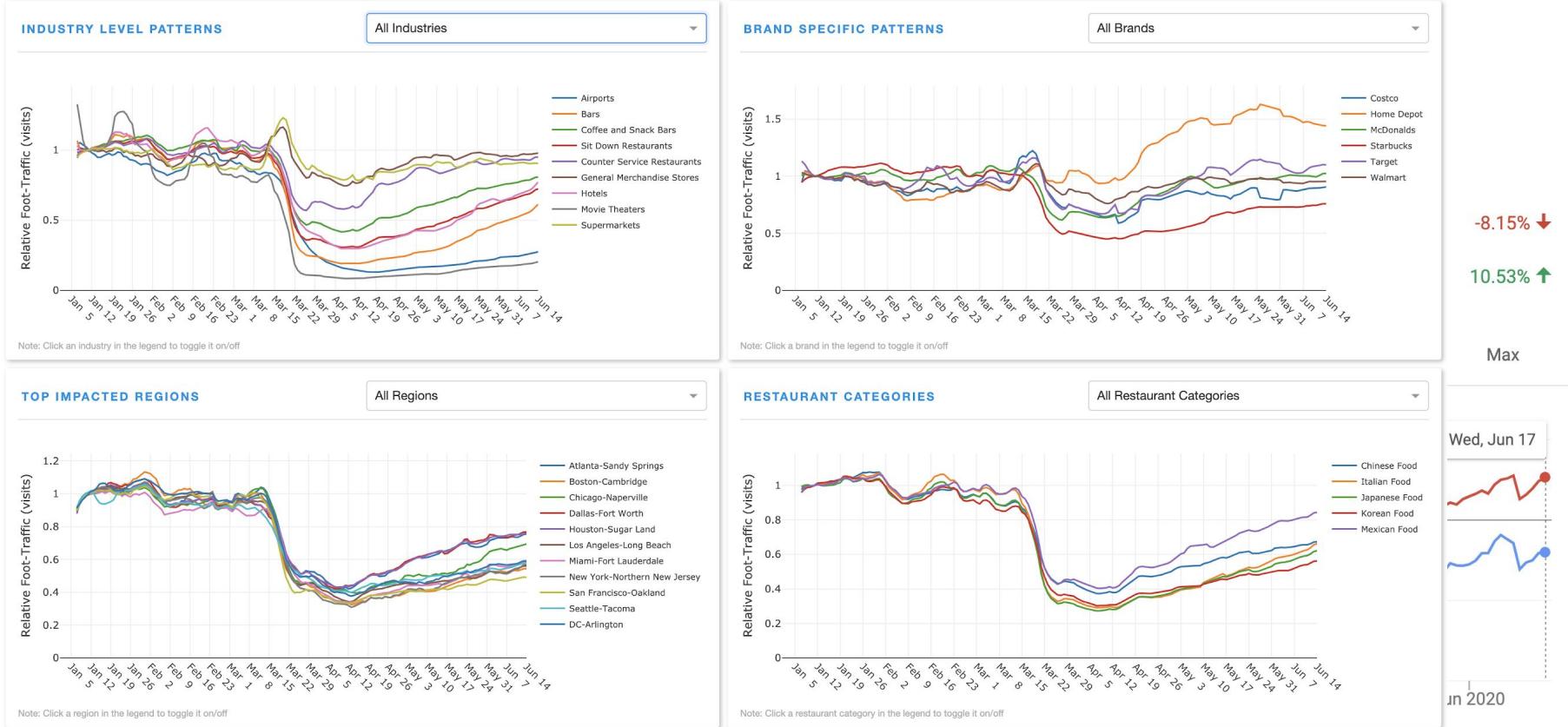
Week 1: Sequences and Prediction

Week 2: Deep Neural Networks for Time Series

Week 3: Recurrent Neural Networks for Time Series

Week 4: Real-world time series data

Sequences and Prediction



Source: <https://www.safegraph.com/dashboard/covid19-commerce-patterns>
<https://finance.google.com/finance>

Sequences and Prediction

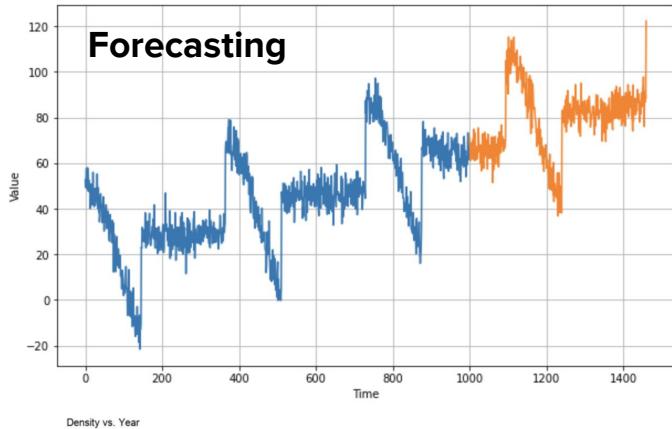


Chart Created by Imoroney@

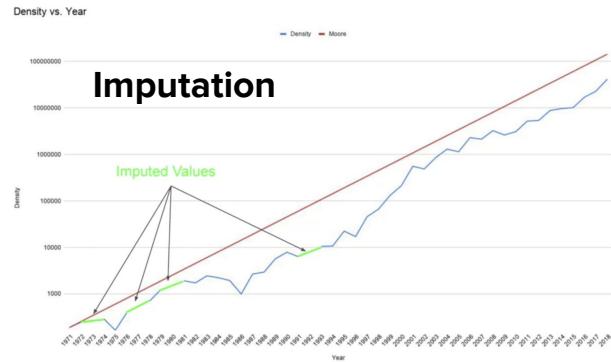
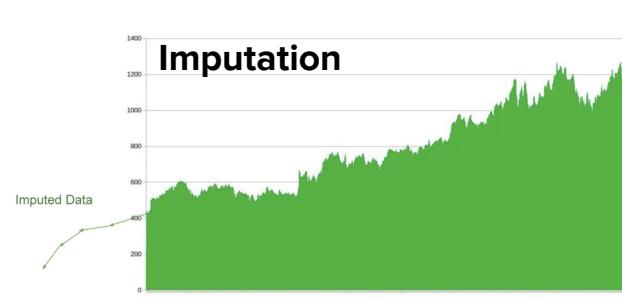
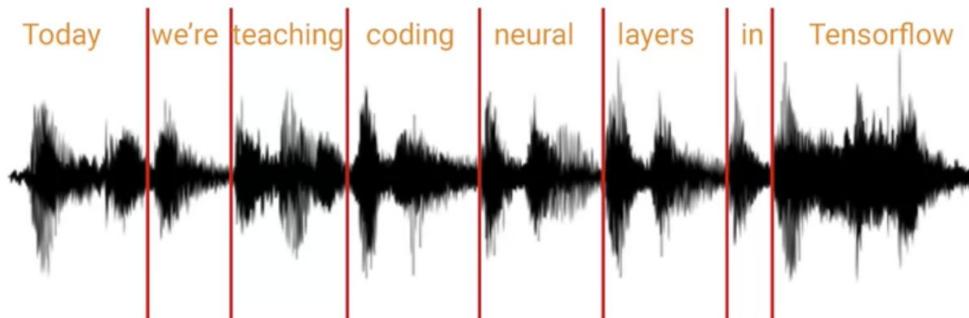


Chart Created by Imoroney@



Patterns / time series analysis



Lawrence's LR optimization?

For consistent results:

```
import random  
seed = 51  
tf.random.set_seed(seed)  
random.seed = seed
```

TensorFlow > API > TensorFlow Core v2.2.0 > Python

tf.keras.callbacks.ReduceLROnPlateau

```
tf.keras.callbacks.ReduceLROnPlateau(  
    monitor='val_loss', factor=0.1, patience=10, verbose=0, mode='auto',  
    min_delta=0.0001, cooldown=0, min_lr=0, **kwargs  
)
```

Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

Set all runs to epochs=500

Restart runtime and run all each test.

Source: https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ReduceLROnPlateau

Lawrence's LR optimization results (seed=51)

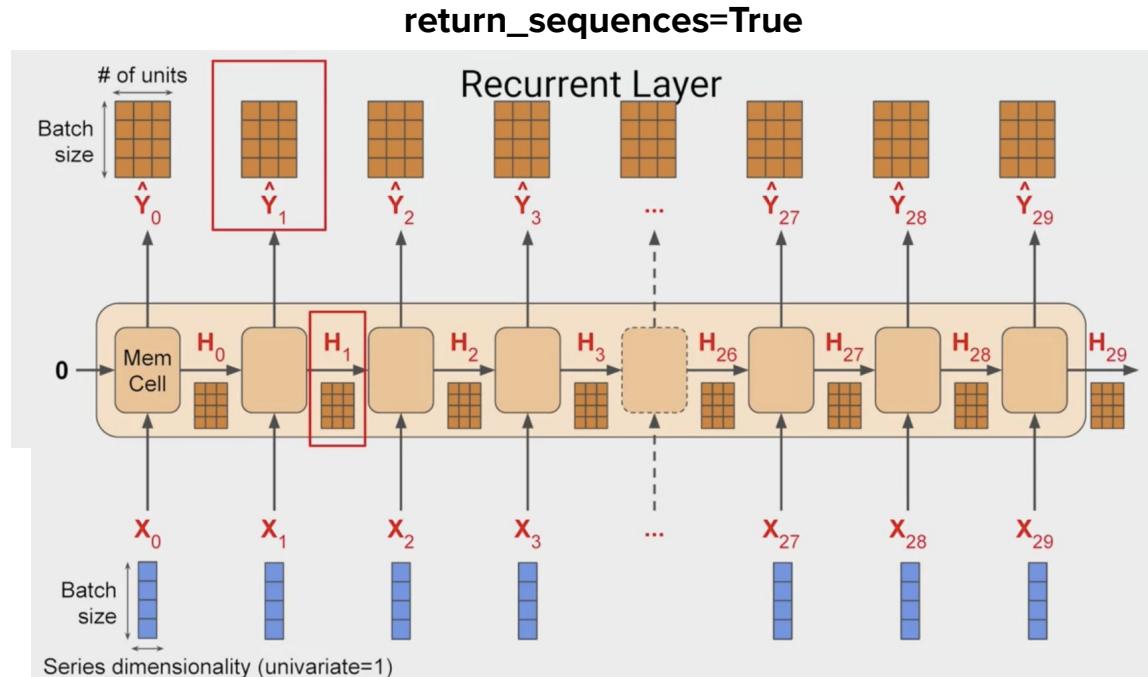
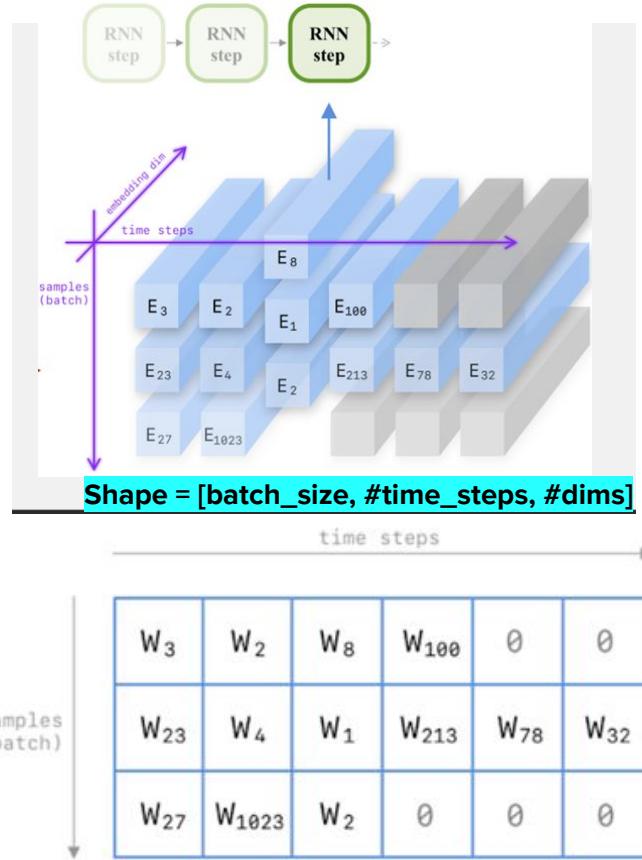
1. `optimizer=tf.keras.optimizers.SGD(lr=1e-6, momentum=0.9)`
`mae 4.500836`

2. `optimizer = tf.keras.optimizers.SGD(lr=4e-6, momentum=0.9)`
`mae 4.8579106`

3. `optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)`
`mae 5.3804765`

4. `ReduceLROnPlateau(monitor='loss', patience=25, verbose=1)`
`optimizer = tf.keras.optimizers.Adam(learning_rate=0.01)`
`mae 4.490887`

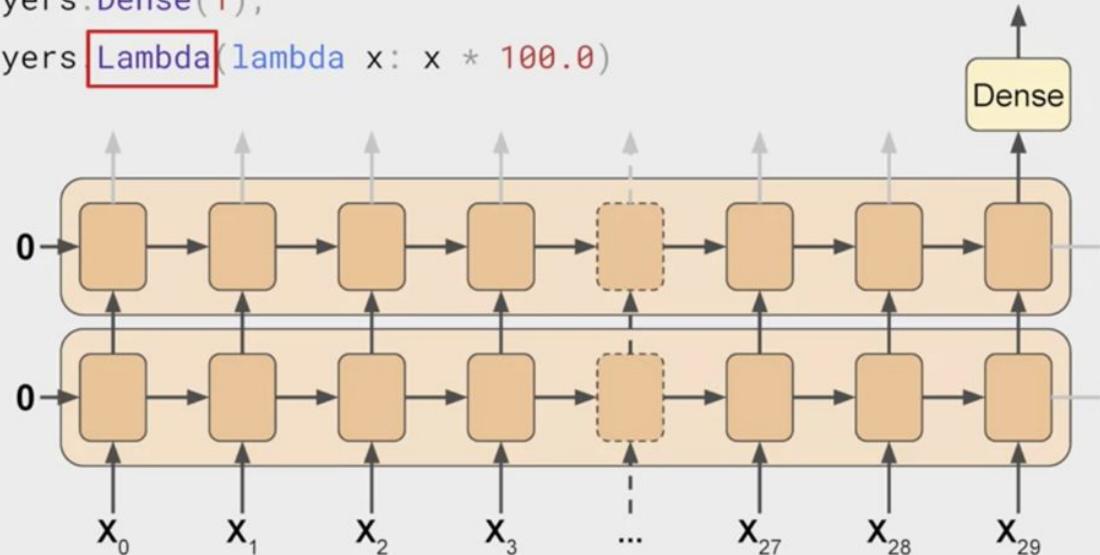
Recurrent Neural Networks for Time Series



Sources: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction>
<http://karpathy.github.io/2015/05/21/rnn-effectiveness>
https://iust-deep-learning.github.io/972/static_files/assignments/assignment_05_preview.html

Recurrent Neural Networks for Time Series

```
model = keras.models.Sequential([
    keras.layers Lambda(lambda x: tf.expand_dims(x, axis=-1),
                        input_shape=[None]),
    keras.layers.SimpleRNN(20, return_sequences=True),
    keras.layers.SimpleRNN(20),
    keras.layers.Dense(1),
    keras.layers Lambda(lambda x: x * 100.0)
])
```



Recurrent Neural Networks for Time Series

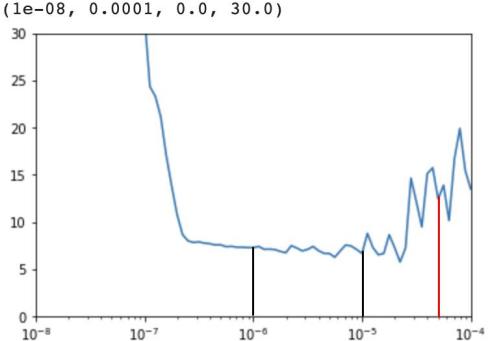
```
# Create the series
series = baseline + trend(time, slope) + seasonality(time, period=365, amplitude=10)
# Update with noise
series += noise(time, noise_level, seed=42)

split_time = 1000
time_train = time[:split_time]
x_train = series[:split_time]
time_valid = time[split_time:]
x_valid = series[split_time:]

window_size = 20
batch_size = 32
shuffle_buffer_size = 1000

my_epoch = 100

plt.semilogx(history.history["lr"], history.history["loss"])
plt.axis([1e-8, 1e-4, 0, 30])
```



```
tf.keras.backer
tf.random.set_s
np.random.seed(42)

train_set = windowed_dataset(...)

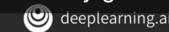
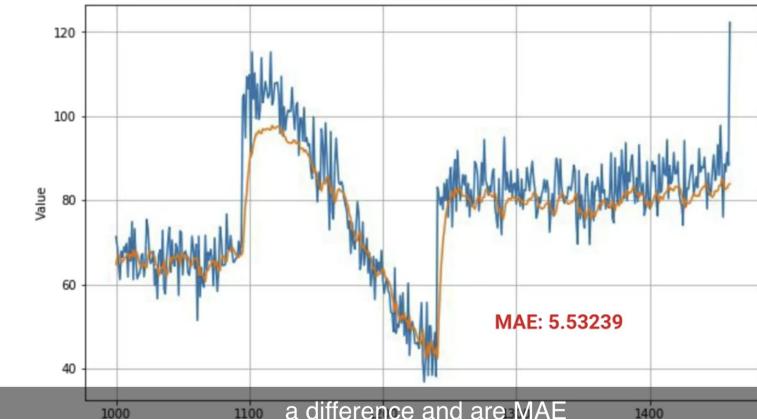
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation="relu"),
    tf.keras.layers.Dense(64, activation="relu"),
    tf.keras.layers.Dense(64, activation="relu"),
    tf.keras.layers.Dense(1)
])

lr_schedule = tf.keras.callbacks.LearningRateScheduler(
    lambda epoch: ...)

optimizer = tf.keras.optimizers.Adam(learning_rate=lr_schedule)
model.compile(loss="mse", optimizer=optimizer)
```

```
history = model.fit(train_set, epochs=my_epoch*4)
```

Coding LSTMs

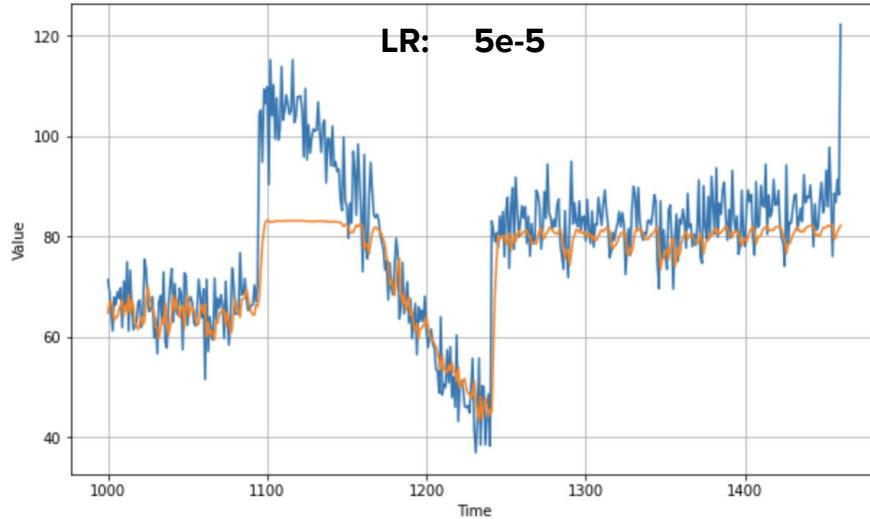


Source: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction>

Errata 5*10-6: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction/lecture/5W1Rw/rnn>

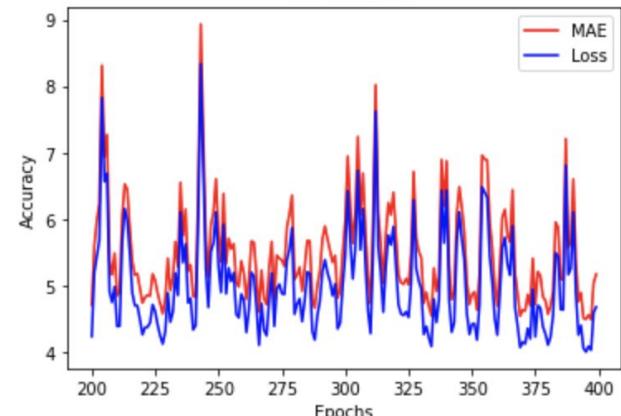
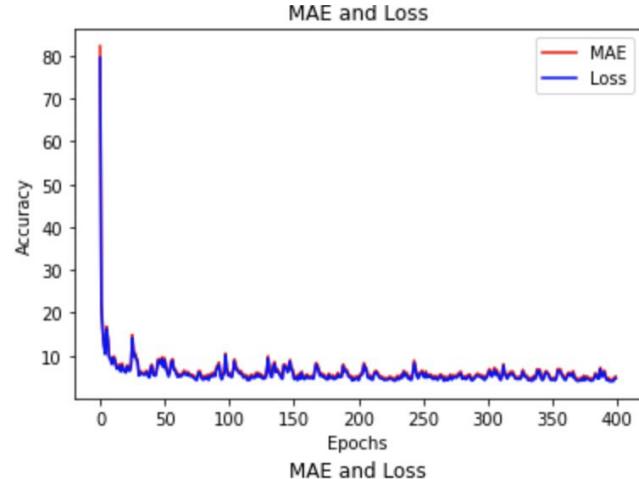
Errata MAE has gone down: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction/lecture/8u3wq/coding-lstms>

Recurrent Neural Networks for Time Series



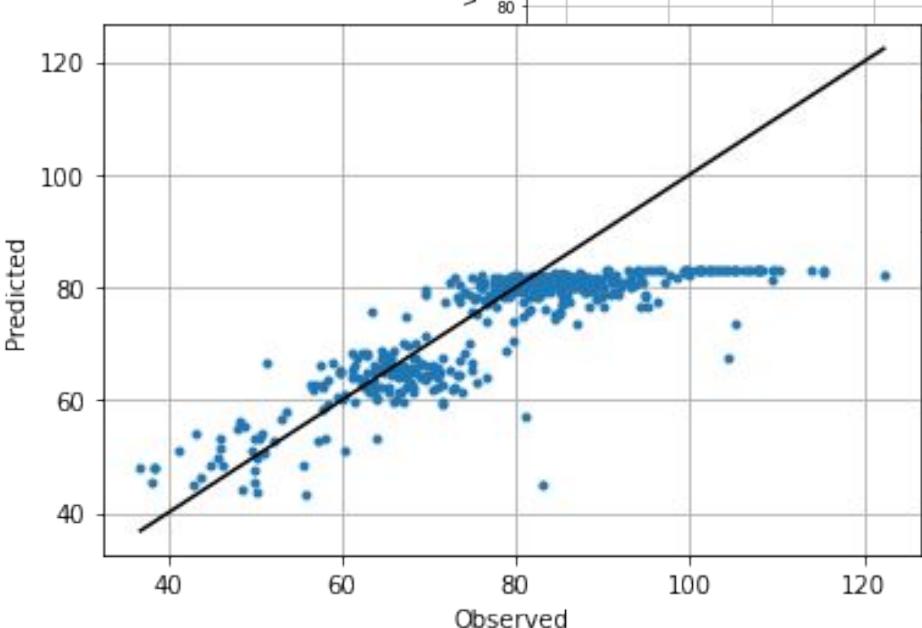
```
tf.keras.metrics.mean_absolute_error(x_valid, results).numpy()
```

6.824173

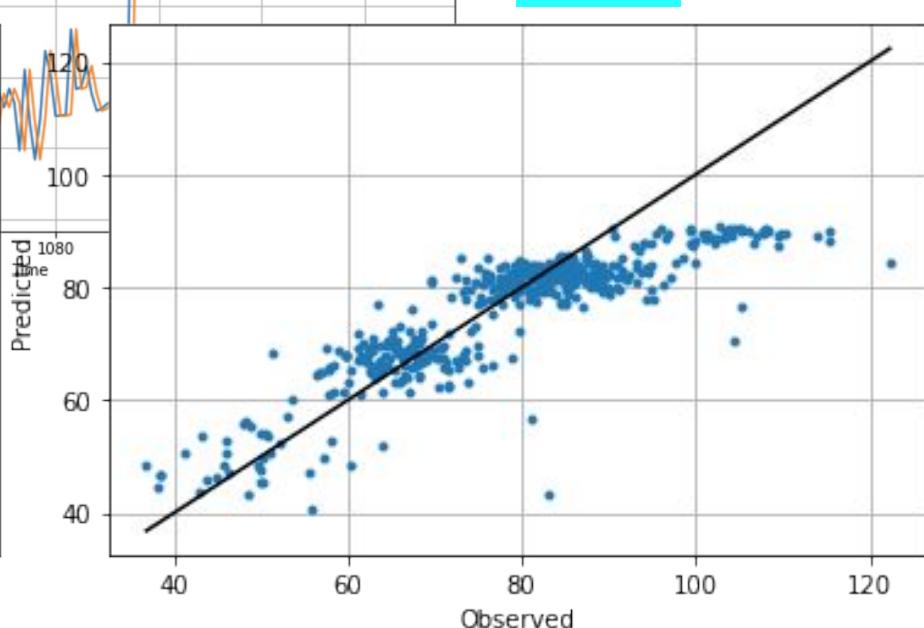


Recurrent Neural Networks for Time Series

LR: 5e-5



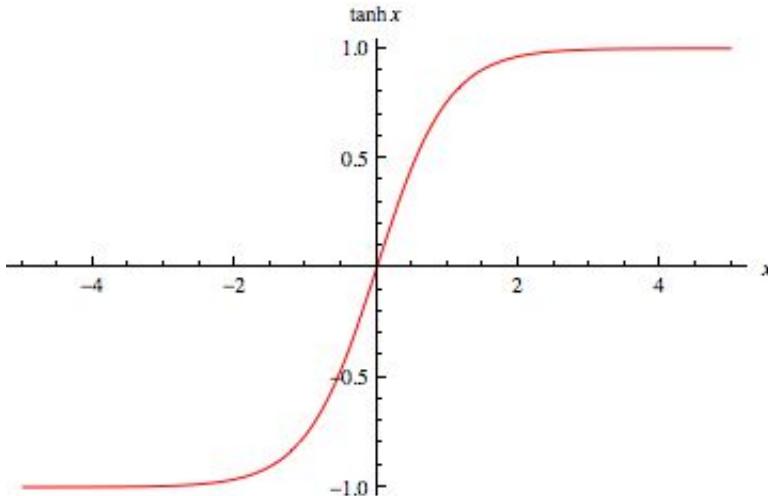
LR: 3e-6



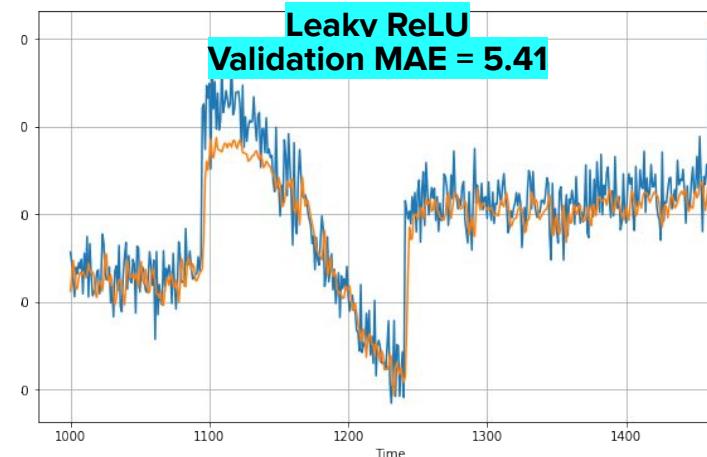
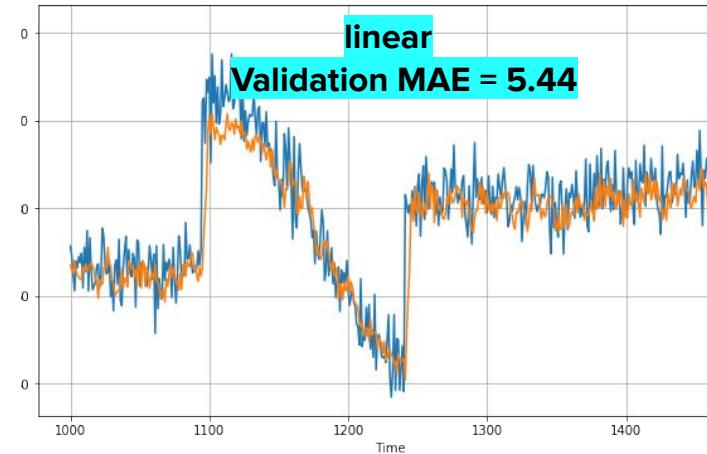
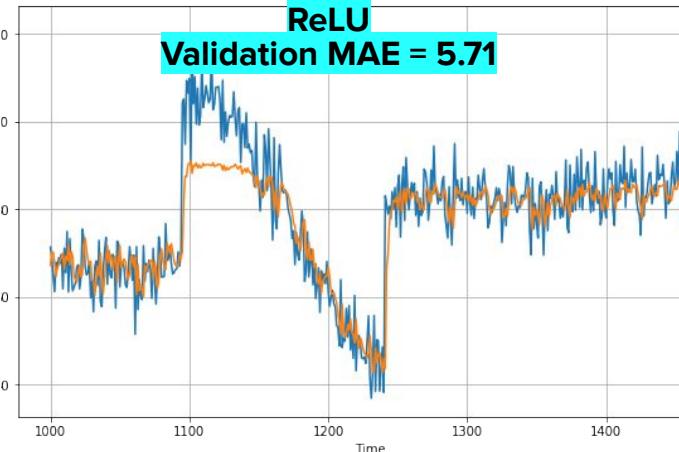
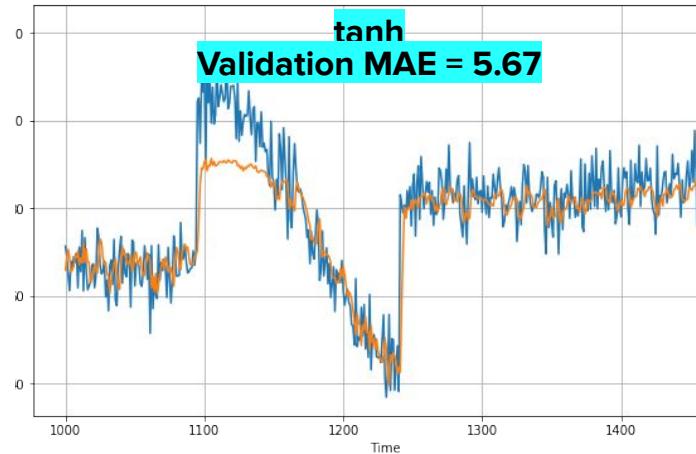
Recurrent Neural Networks for Time Series



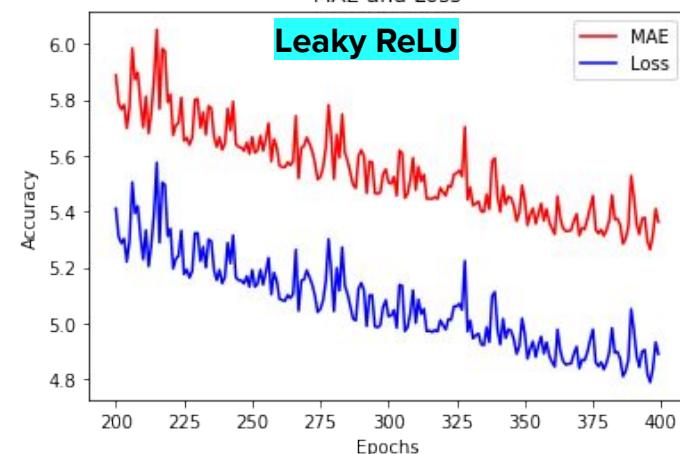
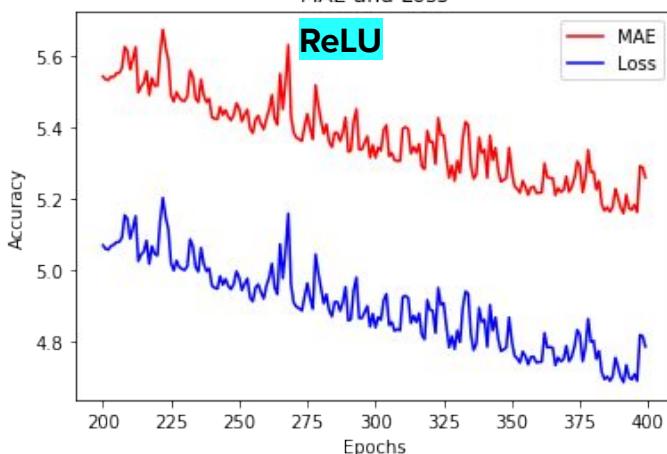
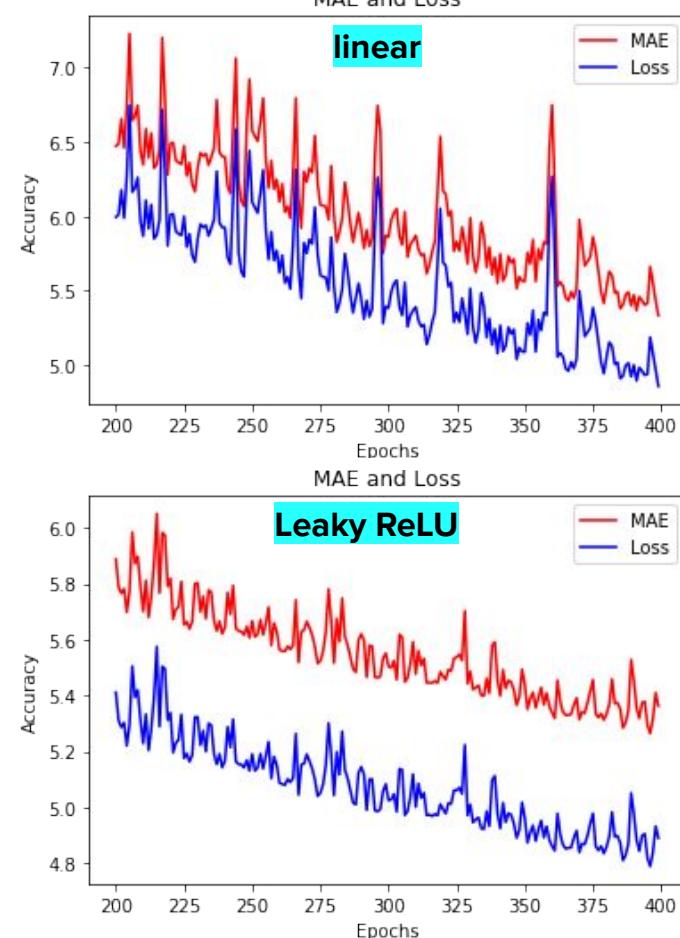
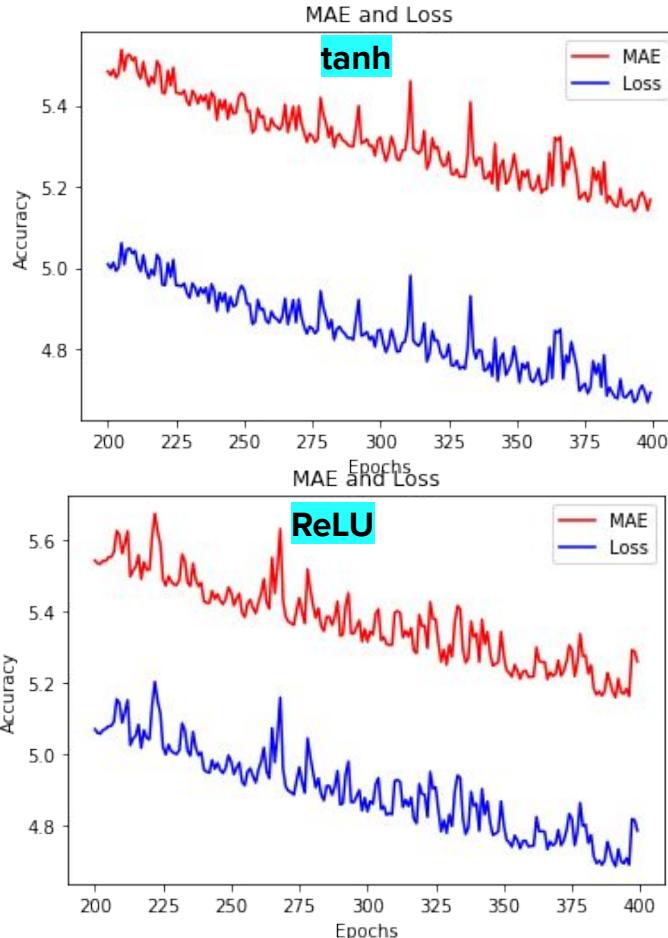
```
tf.keras.layers.SimpleRNN(  
    units, activation='tanh', use_bias=True, kernel_initializer='glor  
    recurrent_initializer='orthogonal', bias_initializer='zeros',  
    kernel_regularizer=None, recurrent_regularizer=None, bias_regularizer  
    activity_regularizer=None, kernel_constraint=None, recurrent_cons
```



Recurrent Neural Networks for Time Series



Recurrent Neural Networks for Time Series



Real-world time series data

Dataset

Sunspots

Monthly Mean Total Sunspot Number - from 1749 to July 2018

Especuloide • updated 5 months ago (Version 2)

Data Tasks Kernels (19) Discussion Activity Metadata

Download (70 KB) New Notebook

Usability 8.2 License CC0: Public Domain Tags natural and physical sciences, astronomy

Description

Context

Sunspots are temporary phenomena on the Sun's photosphere that appear as spots darker than the surrounding areas. They are regions of reduced surface temperature caused by concentrations of magnetic field flux that inhibit convection. Sunspots usually appear in pairs of opposite magnetic polarity. Their number varies according to the approximately 11-year solar cycle.

Source: <https://en.wikipedia.org/wiki/Sunspot>

Content :

Monthly Mean Total Sunspot Number, from 1749/01/01 to 2017/08/31

< **Sunspots.csv** (69.52 KB)

Detail Compact Column

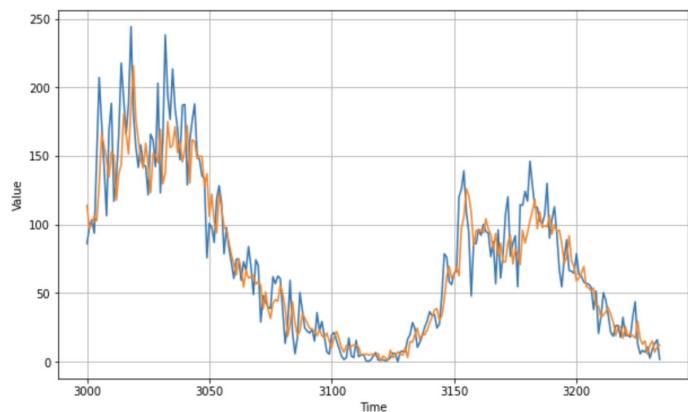
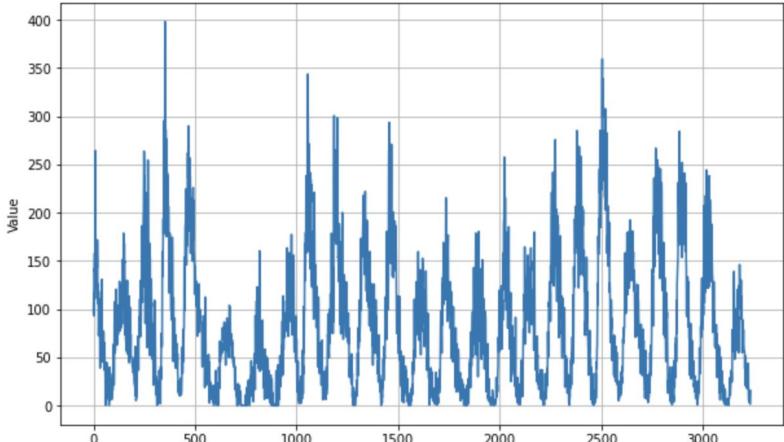
About this file

Sunspots - Monthly Mean Total Sunspot Number

#	Date	Monthly Mean T...
0	3251	398
0	1749-01-31	96.7
1	1749-02-28	104.3
2	1749-03-31	116.7
3	1749-04-30	92.8
4	1749-05-31	141.7
5	1749-06-30	139.2
6	1749-07-31	158.0
7	1749-08-31	110.5
8	1749-09-30	126.5
9	1749-10-31	125.8
10	1749-11-30	264.3

Source: <https://www.kaggle.com/robervalt/sunspots>

Real-world time series data



Source: <https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction>

```
dataset = windowed_dataset_dnn(x_train, window_size, batch_size, shuffle_buffer_size)

model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(20, input_shape=[window_size], activation="relu"),
    tf.keras.layers.Dense(10, activation="relu"),
    tf.keras.layers.Dense(1)
])

model.compile(loss="mse", optimizer=tf.keras.optimizers.SGD(lr=1e-7, momentum=0.9))

model.summary()

Model: "sequential_1"
-----  

Layer (type)           Output Shape        Param #
-----  

dense_3 (Dense)        (None, 20)          620  

dense_4 (Dense)        (None, 10)          210  

dense_5 (Dense)        (None, 1)           11  

-----  

Total params: 841  

Trainable params: 841  

Non-trainable params: 0  

-----  

model.fit(dataset, epochs=100, verbose=1)  

-----  

tf.keras.metrics.mean_absolute_error(x_valid, results).numpy()  

-----  

14.931627
```

Real-world time series data

Screenshot of a GitHub repository page titled "Datasets". The repository has 26 stars and 402 forks. It contains machine learning datasets used in tutorials on MachineLearningMastery.com, with a link to the website (<http://MachineLearningMastery.com>). The repository includes categories for Binary Classification Datasets, Multiclass Classification Datasets, and Regression Datasets.

- Binary Classification Datasets**
 - Breast Cancer (Wisconsin) ([breast-cancer-wisconsin.csv](#))
 - Breast Cancer (Yugoslavia) ([breast-cancer.csv](#))
 - Breast Cancer (Haberman's) ([haberman.csv](#))
 - Bank Note Authentication ([banknote_authentication.csv](#))
 - Horse Colic ([horse-colic.csv](#))
 - Ionosphere ([ionosphere.csv](#))
 - Pima Indians Diabetes ([pima-indians-diabetes.csv](#))
 - Sonar Returns ([sonar.csv](#))
 - German Credit ([german.csv](#))
 - Credit Card Fraud ([creditcard.csv.zip](#))
 - Adult Income ([adult-all.csv](#))
 - Mammography ([mammography.csv](#))
 - Oil Spill ([oil-spill.csv](#))
 - Phoneme ([phoneme.csv](#))
- Multiclass Classification Datasets**
 - Glass Identification ([glass.csv](#))
 - Iris Flower Species ([iris.csv](#))
 - Wheat Seeds ([wheat-seeds.csv](#))
 - Wine ([wine.csv](#))
 - Ecoli ([ecoli.csv](#))
 - Thyroid Gland ([new-thyroid.csv](#))
- Regression Datasets**
 - Boston Housing ([housing.csv](#))
 - Auto Insurance Total Claims ([auto-insurance.csv](#))
 - Auto Imports Prices ([auto_imports.csv](#))
 - Abalone Age ([abalone.csv](#))
 - Wine Quality Red ([winequality-red.csv](#))
 - Wine Quality White ([winequality-white.csv](#))

Source: <https://github.com/jbrownlee/Datasets>

Univariate Time Series Datasets

- Daily Minimum Temperatures in Melbourne ([daily-min-temperatures.csv](#))
- Daily Maximum Temperatures in Melbourne ([daily-max-temperatures.csv](#))
- Daily Female Births in California ([daily-total-female-births.csv](#))
- Monthly International Airline Passengers ([monthly-airline-passengers.csv](#))
- Monthly Armed Robberies in Boston ([monthly-robberies.csv](#))
- Monthly Sunspots ([monthly-sunspots.csv](#))
- Monthly Champagne Sales ([monthly_champagne_sales.csv](#))
- Monthly Shampoo Sales ([monthly-shampoo-sales.csv](#))
- Monthly Car Sales ([monthly-car-sales.csv](#))
- Monthly Mean Temperatures in Nottingham Castle ([monthly-mean-temp.csv](#))
- Monthly Specialty Writing Paper Sales ([monthly-writing-paper-sales.csv](#))
- Yearly Water Usage in Baltimore ([yearly-water-usage.csv](#))

Multivariate Time Series Datasets

- Hourly Pollution Levels in Beijing
- Minutely Individual Household Electric Power Consumption
- Human Activity Recognition Using Smartphones
- Indoor Movement Prediction
- Yearly Longley Economic Employment

Check out these resources



Get more from Colab

UPGRADE NOW

\$9.99/month

Recurring billing • Cancel anytime

Restrictions apply, learn more here



Faster GPUs

Priority access to faster GPUs and TPUs means you spend less time waiting while code is running. [Learn more](#)



Longer runtimes

Longer running notebooks and fewer idle timeouts mean you disconnect less often. [Learn more](#)



More memory

More RAM means better performance, and less running out of memory. [Learn more](#)

[See what Colab Pro benefits would look like](#)



Install

Learn

API

Resources

Community

Why TensorFlow

Search

TensorFlow Core

Overview

Tutorials

Guide

TF 1

TensorFlow tutorials

Quickstart for beginners
Quickstart for experts

BEGINNER

ML basics with Keras

Load and preprocess data

Estimator

ADVANCED

Customization

Distributed training

Images

Text

Structured data

Text

Word embeddings

Text classification with an RNN

Text generation with an RNN

Neural machine translation with attention

Image captioning

Transformer model for language understanding

The TensorFlow tutorials are written as Jupyter notebooks and run directly in Google Colab—a hosted notebook environment that requires no setup. Click the [Run in Google Colab](#) button.

For beginners

The best place to start is with the user-friendly Keras sequential API. Build models by plugging together building blocks. After these tutorials, read the [Keras guide](#).

Beginner quickstart

This "Hello, World!" notebook shows the Keras Sequential API and `model.fit`.

Keras basics

This notebook collection demonstrates basic machine learning tasks using Keras.

Load data

These tutorials use `tf.data` to load various data formats and build input pipelines.

For experts

The Keras functional and subclassing APIs provide a define-by-run interface for customization and advanced research. Build your model, then write the forward and backward pass. Create custom layers, activations, and training loops.

Advanced quickstart

This "Hello, World!" notebook uses the Keras subclassing API and a custom training loop.

Customization

This notebook collection shows how to build custom layers and training loops in TensorFlow.

Distributed training

Distribute your model training across multiple GPUs, multiple machines or TPUs.

The Advanced section has many instructive notebook examples, including [Neural machine translation](#), [Transformers](#), and [CycleGAN](#).

Source:

<https://colab.research.google.com/signup>

<https://www.tensorflow.org/tutorials/>

Check out these AI for Healthcare resources



Enroll for Free
Starts May 22

About How It Works Courses Instructors Enrollment Options FAQ

There are 3 Courses in this Specialization

COURSE

AI for Medical Diagnosis

1

★★★★★ 4.6 410 ratings • 102 reviews

AI is transforming the practice of medicine. It's helping doctors diagnose patients more accurately, make predictions about patients' future health, and recommend better treatments. As an AI practitioner, you have the opportunity to join in this transformation of modern medicine. If you're already familiar with some of the math and coding behind AI

[SHOW ALL](#)

COURSE

AI for Medical Prognosis

2

★★★★★ 4.6 113 ratings • 27 reviews

AI is transforming the practice of medicine. It's helping doctors diagnose patients more accurately, make predictions about patients' future health, and recommend better treatments. This Specialization will give you practical experience in applying machine learning to concrete problems in medicine.

[SHOW ALL](#)

COURSE

AI For Medical Treatment

3

AI is transforming the practice of medicine. It's helping doctors diagnose patients more accurately, make predictions about patients' future health, and recommend better treatments. This Specialization will give you practical experience in applying machine learning to concrete problems in medicine.

[SHOW ALL](#)

AI for Healthcare

Learn to build, evaluate, and integrate predictive models that have the power to transform patient outcomes. Begin by classifying and segmenting 2D and 3D medical images to augment diagnosis and then move on to modeling patient outcomes with electronic health records to optimize clinical trial testing decisions. Finally, build an algorithm that uses data collected from wearable devices to estimate the wearer's pulse rate in the presence of motion.

PREREQUISITE KNOWLEDGE

Intermediate Python, and Experience with Machine Learning See detailed requirements.

[— HIDE DETAILS](#)



Applying AI to 2D Medical Imaging Data

Learn the fundamental skills needed to work with 2D medical imaging data and how to use AI to derive clinically-relevant insights from data gathered via different types of 2D medical imaging such as x-ray, mammography, and digital pathology. Extract 2D images from DICOM files and apply the appropriate tools to perform exploratory data analysis on them. Build different AI models for different clinical scenarios that involve 2D images and learn how to position AI tools for regulatory approval.

PNEUMONIA DETECTION FROM CHEST X-RAYS

Applying AI to 3D Medical Imaging Data

Learn the fundamental skills needed to work with 3D medical imaging datasets and frame insights derived from the data in a clinically relevant context. Understand how these images are acquired, stored in clinical archives, and subsequently read and analyzed. Discover how clinicians use 3D medical images in practice and where AI holds most potential in their work with these images. Design and apply machine learning algorithms to solve the challenging problems in 3D medical imaging and how to integrate the algorithms into the clinical workflow.

HIPPOCAMPUS VOLUME QUANTIFICATION FOR ALZHEIMER'S PROGRESSION

Applying AI to EHR Data

Learn the fundamental skills to work with EHR data and build and evaluate compliant, interpretable models. You will cover EHR data privacy and security standards, how to analyze EHR data and avoid common challenges, and cover key industry code sets. By the end of the course, you will have the skills to analyze an EHR dataset, transform it to the right level, build powerful features with TensorFlow, and model the uncertainty and bias with TensorFlow Probability and Aequitas.

PATIENT SELECTION FOR DIABETES DRUG TESTING

Applying AI to Wearable Device Data

Learn how to build algorithms that process the data collected by wearable devices and surface insights about the wearer's health. Cover the sensors and signal processing foundation that are critical for success in this domain, including IMU, PPG, and ECG that are common to most wearable devices, and learn how to build three algorithms from real-world sensor data.

MOTION COMPENSATED PULSE RATE ESTIMATION

Source:

<https://www.coursera.org/specializations/ai-for-medicine>

<https://www.udacity.com/course/ai-for-healthcare-nanodegree--nd320>

Check out this certification and books

TensorFlow Core

Introducing the TensorFlow Developer Certificate!

March 12, 2020



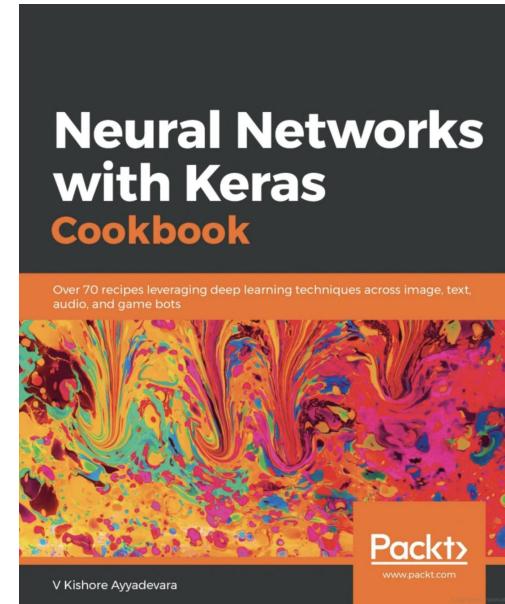
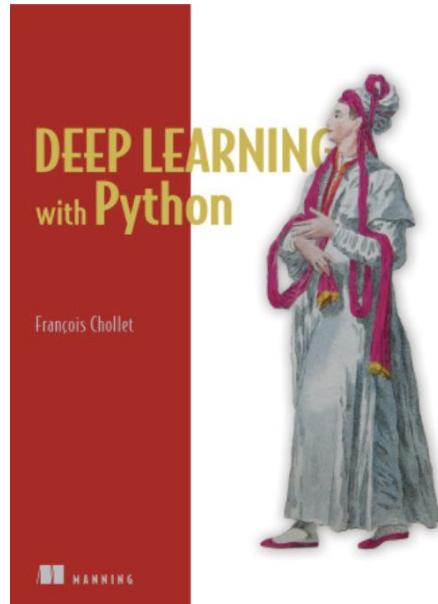
Posted by Alina Shinkarsky, on behalf of the TensorFlow Team

In the AI world today, more and more companies are looking to hire machine learning talent, and simultaneously, an increasing number of students and developers are looking for ways to gain and showcase their ML knowledge with formal recognition. In addition to the courses and learning resources available online, we want to help developers showcase their ML proficiency and help companies hire ML developers to solve challenging problems.



Source:

- <https://blog.tensorflow.org/2020/03/introducing-tensorflow-developer-certificate.html?m=1>
- <https://www.manning.com/books/deep-learning-with-python>
- https://books.google.com/books?id=5quLDwAAQBAJ&printsec=frontcover&source=gbs_qe_summary_r&cad=0#v=onepage&q&f=false
- <https://github.com/PacktPublishing/Neural-Networks-with-Keras-Cookbook/tree/master/Chapter11>

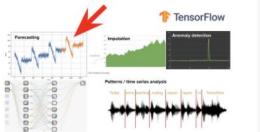


Let's continue our Time Series adventure 😊

FRI, JUN 19, 7:30 PM EDT

TensorFlow in Practice - C4 Week 1,2 - Forecasting and DNN for Time...

Online event



Join us for our 9th adventure in Deep Learning! Just bring your curiosity and get ready to meet our growing community 😊 We are taking Course 4 of TensorFlow in Practice Specialization available at:...



39 attendees



FRI, JUL 3, 7:30 PM EDT

A chat with Laurence Moroney, AI Lead at Google

Online event



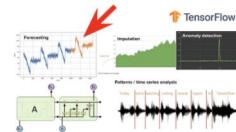
Join us for a fun conversation with Laurence Moroney, AI Lead at Google (<https://www.linkedin.com/in/laurence-moroney>) and developer of our TensorFlow in Practice and TensorFlow: Data and Deployment Specializations 🎉 We plan to...



71 attendees

2

Attend



FRI, JUN 26, 7:30 PM EDT

TensorFlow in Practice - C4 Week 3,4 - Forecasting & RNN, Conv1D f...

Online event

Join us for our 10th adventure in Deep Learning! Just bring your curiosity and get ready to meet our growing community 😊 We are taking our last Course 4 of TensorFlow in Practice Specialization available at:...

Attend



FRI, JUL 10, 7:30 PM EDT

How to prepare for and pass the TensorFlow Developer Certificate 🎉

Online event



Join us for our 11th adventure in Deep Learning! Just bring your curiosity and get ready to meet our growing community 😊 Join Zoom Meeting: <https://us02web.zoom.us/j/84402592502?...>



21 attendees

Attend

Questions

Discussion

2 Deep Learning representations

For representations:

- nodes represent inputs, activations or outputs
- edges represent weights or biases

Here are several examples of Standard deep learning representations

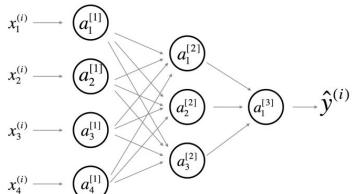


Figure 1: Comprehensive Network: representation commonly used for Neural Networks. For better aesthetic, we omitted the details on the parameters ($w_{ij}^{[l]}$ and $b_i^{[l]}$ etc...) that should appear on the edges

Standard notations for Deep Learning

This document has the purpose of discussing a new standard for deep learning mathematical notations.

1 Neural Networks Notations.

General comments:

- superscript (i) will denote the i^{th} training example while superscript [l] will denote the l^{th} layer

Sizes:

$\cdot m$: number of examples in the dataset

$\cdot n_x$: input size

$\cdot n_y$: output size (or number of classes)

$\cdot n_h^{[l]}$: number of hidden units of the l^{th} layer

In a for loop, it is possible to denote $n_x = n_h^{[0]}$ and $n_y = n_h^{[\text{number of layers} + 1]}$.

$\cdot L$: number of layers in the network.

Objects:

$\cdot X \in \mathbb{R}^{n_x \times m}$ is the input matrix

$\cdot x^{(i)} \in \mathbb{R}^{n_x}$ is the i^{th} example represented as a column vector

$\cdot Y \in \mathbb{R}^{n_y \times m}$ is the label matrix

$\cdot y^{(i)} \in \mathbb{R}^{n_y}$ is the output label for the i^{th} example

$\cdot W^{[l]} \in \mathbb{R}^{\text{number of units in next layer} \times \text{number of units in the previous layer}}$ is the weight matrix, superscript [l] indicates the layer

$\cdot b^{[l]} \in \mathbb{R}^{\text{number of units in next layer}}$ is the bias vector in the l^{th} layer

$\cdot \hat{y} \in \mathbb{R}^{n_y}$ is the predicted output vector. It can also be denoted $a^{[L]}$ where L is the number of layers in the network.

Common forward propagation equation examples:

$a = g^{[l]}(W_x x^{(i)} + b_1) = g^{[l]}(z_1)$ where $g^{[l]}$ denotes the l^{th} layer activation function

$\hat{y}^{(i)} = \text{softmax}(W_h h + b_2)$

· General Activation Formula: $a_j^{[l]} = g^{[l]}(\sum_k w_{jk}^{[l]} a_k^{[l-1]} + b_j^{[l]}) = g^{[l]}(z_j^{[l]})$

· $J(x, W, b, y)$ or $J(\hat{y}, y)$ denote the cost function.

Examples of cost function:

$\cdot J_{CE}(\hat{y}, y) = -\sum_{i=0}^m y^{(i)} \log \hat{y}^{(i)}$

$\cdot J_1(\hat{y}, y) = \sum_{i=0}^m |y^{(i)} - \hat{y}^{(i)}|$

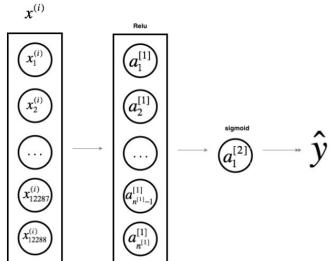


Figure 2: Simplified Network: a simpler representation of a two layer neural network, both are equivalent.