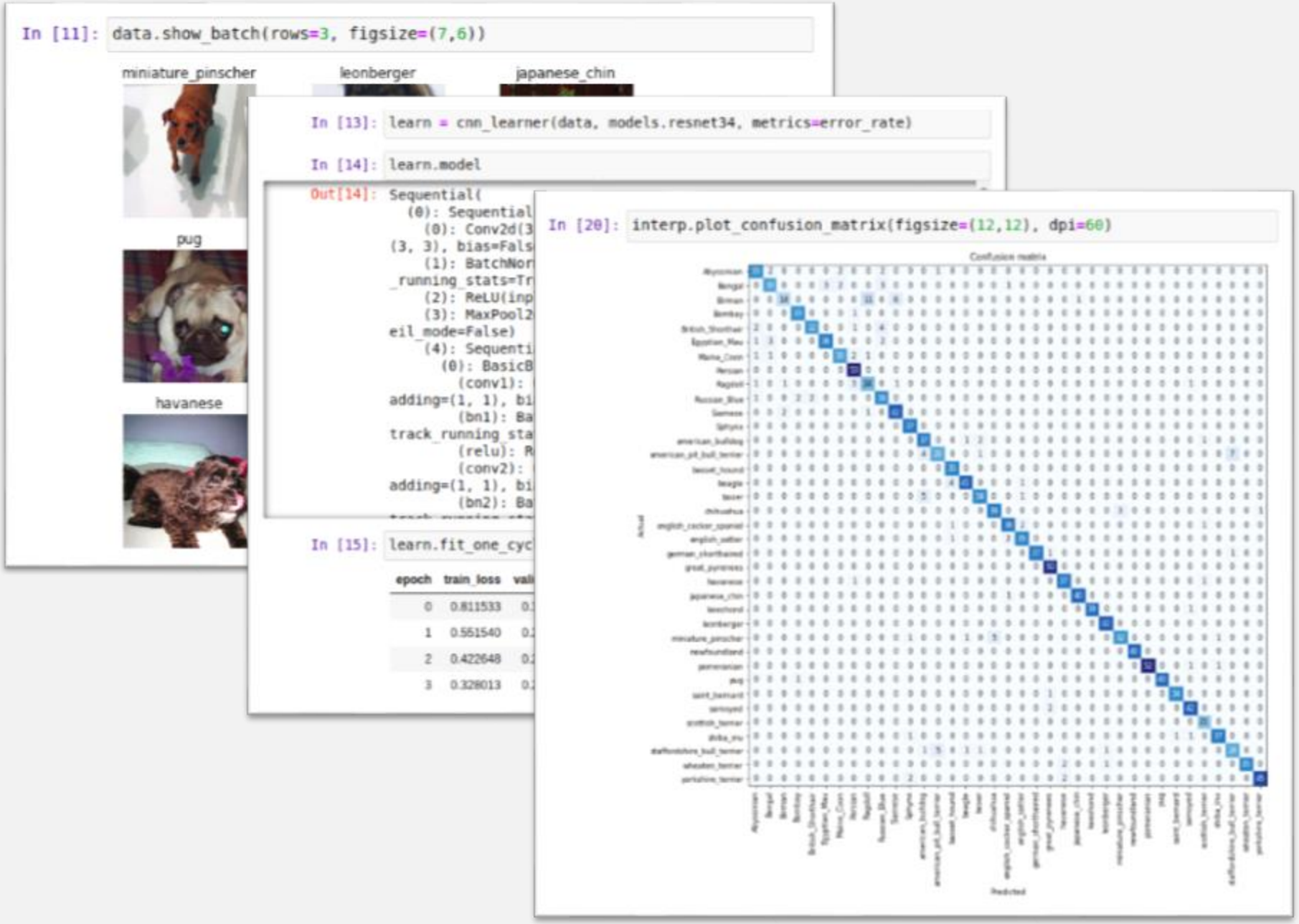


fast.ai

Making neural nets uncool again

CAIR

Josiah Laivins

February 23, 2020

Making Deep Learning Accessible

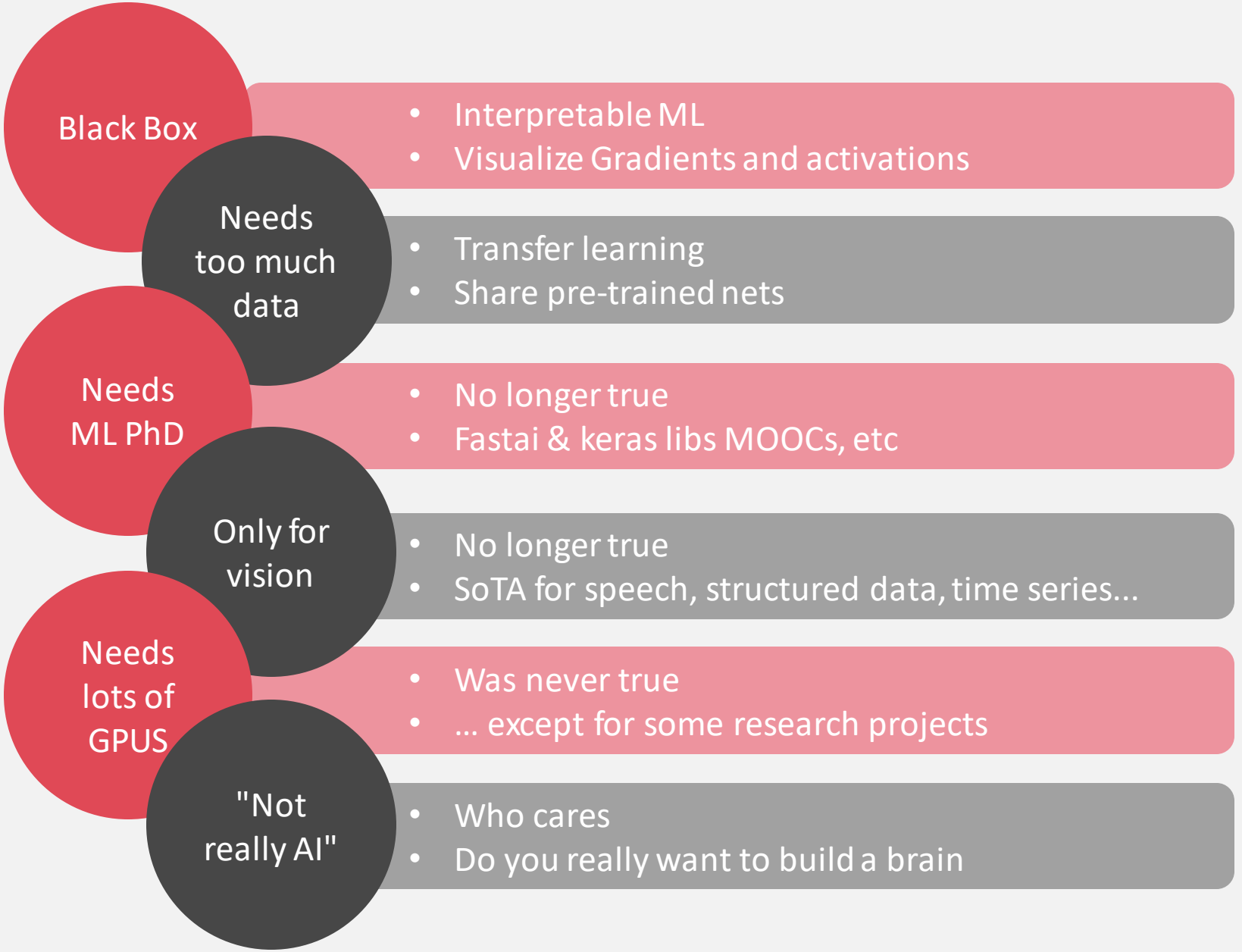
Software
To make these available touse quickly, reliably, and with minimal code

Community
So that we can all help each other

Research
Ways to make state of the art deep learning techniques more accessible

Research
So that as many people as possible use these

Description from: https://course.fast.ai/videos/?lesson=1



Description from: https://course.fast.ai/videos/?lesson=1

fast.ai
Vision models

Computer Vision models zoo

The fastai library includes several pretrained models from torchvision, namely:

- resnet18, resnet34, resnet50, resnet101, resnet152
- squeezenet1_0, squeezenet1_1
- densenet121, densenet169, densenet201, densenet161
- vgg16_bn, vgg19_bn
- alexnet

On top of the models offered by torchvision, fastai has implementations for the following models:

- Darknet architecture, which is the base of Yolo v3
- Unet architecture based on a pretrained model. The original unet is described here, the model implementation is detailed in `models.unet`
- Wide resnets architectures, as introduced in this article

Ref: https://docs.fast.ai/vision.models.html

Darknet implimentation example

```python
from ...torch_core import *
from ...layers import *

__all__ = ['Darknet', 'ResLayer']

def conv_bn_lrelu(ni:int, nf:int, ks:int=3, stride:int=1)->nn.Sequential:
    "Create a seuence Conv2d->BatchNorm2d->LeakyReLu layer."
    return nn.Sequential(
        nn.Conv2d(ni, nf, kernel_size=ks, bias=False, stride=stride, padding=ks//2),
        nn.BatchNorm2d(nf),
        nn.LeakyReLU(negative_slope=0.1, inplace=True))

class ResLayer(Module):
    "Resnet style layer with `ni` inputs."
    def __init__(self, ni:int):
        self.conv1 = conv_bn_lrelu(ni, ni//2, ks=1)
        self.conv2 = conv_bn_lrelu(ni//2, ni, ks=3)

    def forward(self, x): return x + self.conv2(self.conv1(x))

class Darknet(Module):
    "https://github.com/pjreddie/darknet"
    def make_group_layer(self, ch_in:int, num_blocks:int, stride:int=1):
        "starts with conv layer - `ch_in` channels in - then has `num_blocks` `ResLayer`"
        return [conv_bn_lrelu(ch_in, ch_in*2,stride=stride)
                ] + [(ResLayer(ch_in*2)) for i in range(num_blocks)]

    def __init__(self, num_blocks:Collection[int], num_classes:int, nf=32):
        "create darknet with `nf` and `num_blocks` layers"
        layers = [conv_bn_lrelu(3, nf, ks=3, stride=1)]
        for i,nb in enumerate(num_blocks):
            layers += self.make_group_layer(nf, nb, stride=2)
            nf *= 2
        layers += [nn.AdaptiveAvgPool2d(1), Flatten(), nn.Linear(nf, num_classes)]
        self.layers = nn.Sequential(*layers)

    def forward(self, x): return self.layers(x)
```

Ref: https://github.com/fastai/fastai/blob/master/fastai/vision/models/darknet.py

I wanna make my own model!

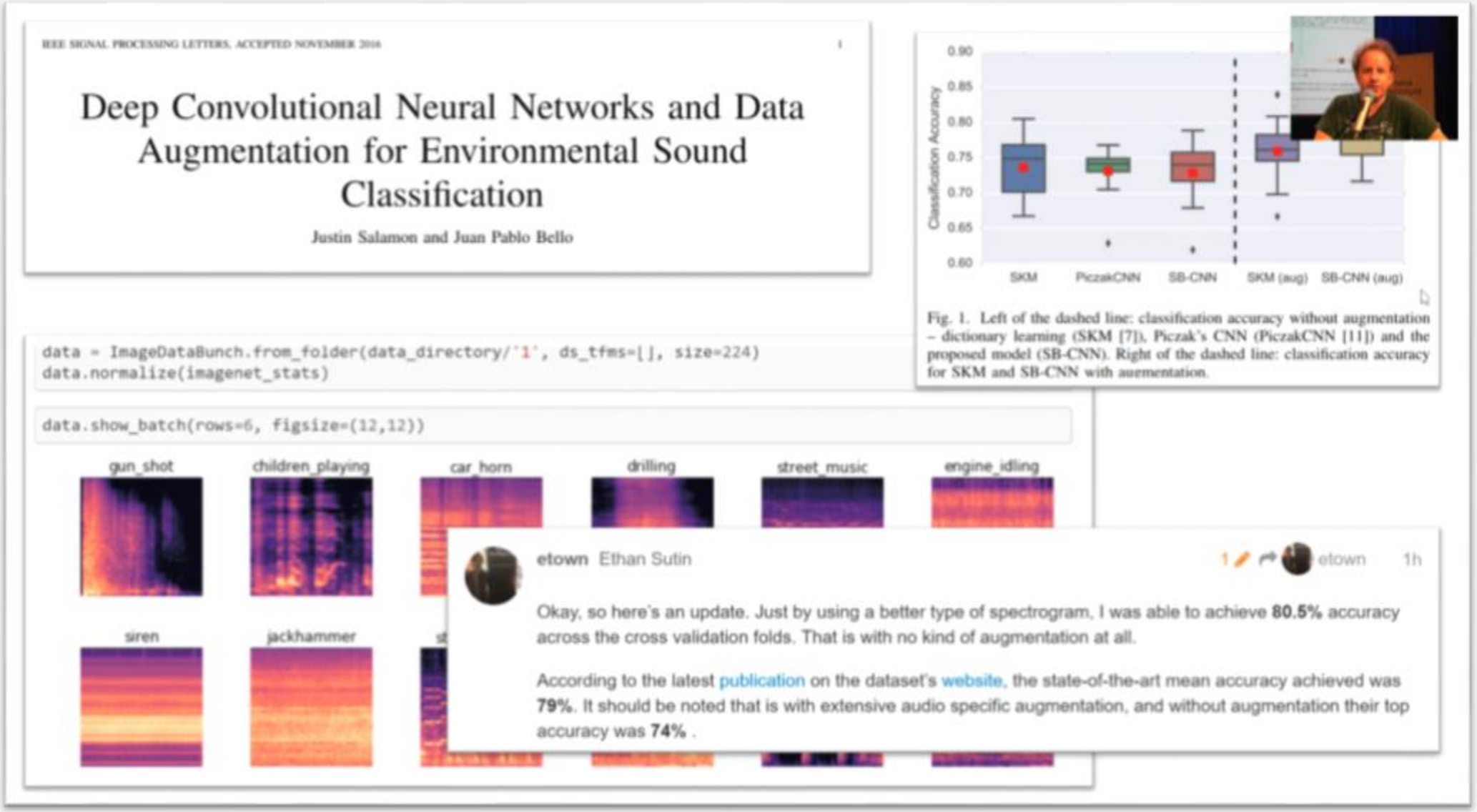
Trainer
If your model involves a novel way of training.

Learner
Needed if you have a new model and new Training method.

Model
If your model your model is simply a pytorch Module object.

DataBunch
If there is a new way of handling data.

[1]: https://github.com/fastai/fastai/blob/master/fastai/vision/gan.py
[2]: https://docs.fast.ai/vision.gan.html
[3]: https://github.com/josiahls/fast-reinforcement-learning/blob/master/fast_rl/agents/dqn.py



[1]: https://github.com/hiromis/notes/blob/master/Lesson2.md
[2]: https://forums.fast.ai/t/share-your-work-here/27676/215

State of the art on DHCD (देवनागरी)

Suvash Thapaliya @suvash · Oct 27

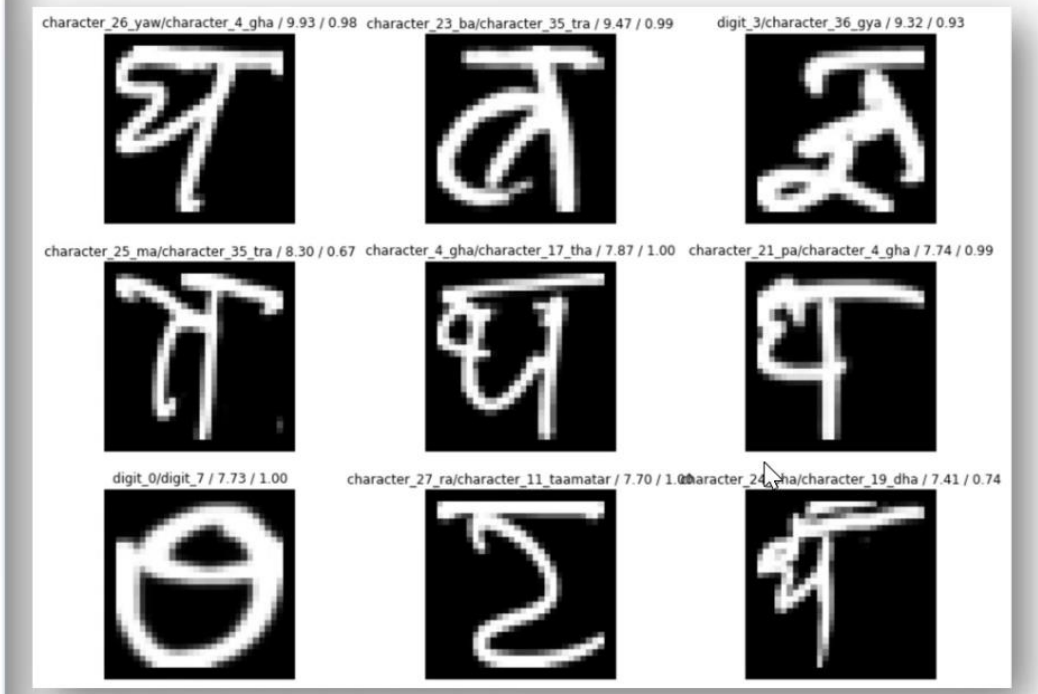
After lots of intentional tuneups, I can now announce that SoTA for accuracy on #DHCD #देवनागरी dataset is 99.02% (previously 98.50%). All thanks to @fastdotai library & courses, and the dataset creators @pk_gyawali amongst others. #resnet34 #devanagari #deeplearning

lel fitting (transfer learning) on

_loss	valid_loss	accuracy
342	0.037567	0.989493
461	0.035617	0.990072
426	0.034761	0.990290

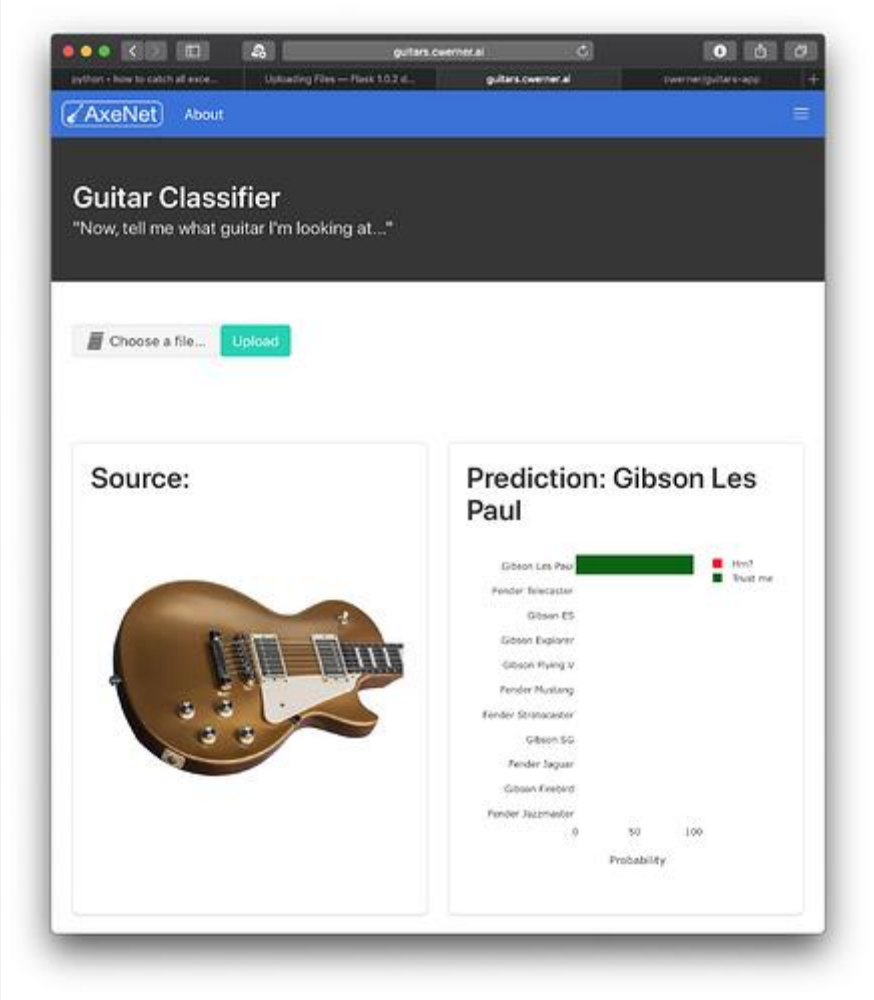
aining cycle (Accuracy = 99.02

3 12 63

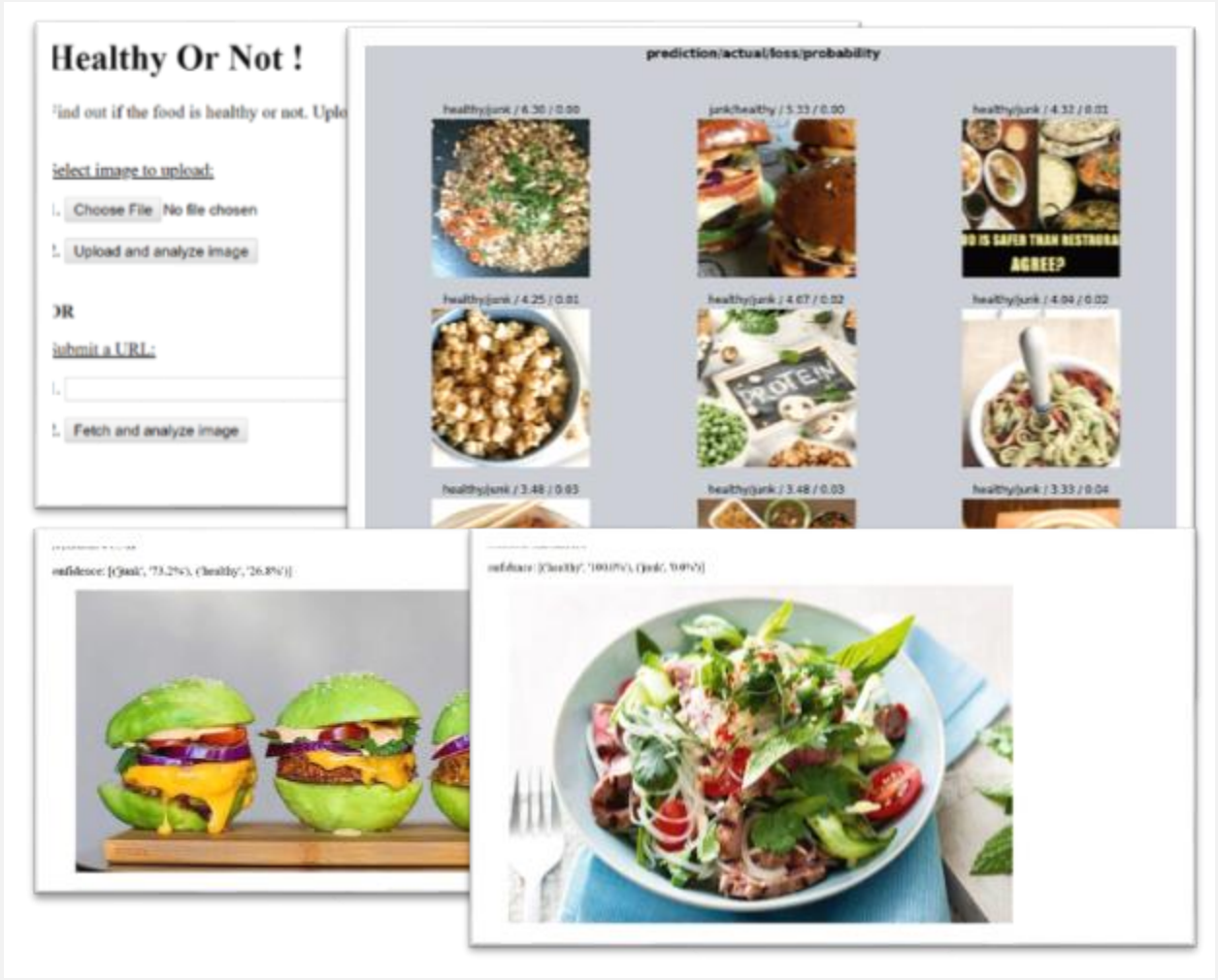


[1]: https://github.com/hiromis/notes/blob/master/Lesson2.md
[2]: https://forums.fast.ai/t/share-your-work-here/27676/38

Guitar Classifier



Junk or Health Food Classifier

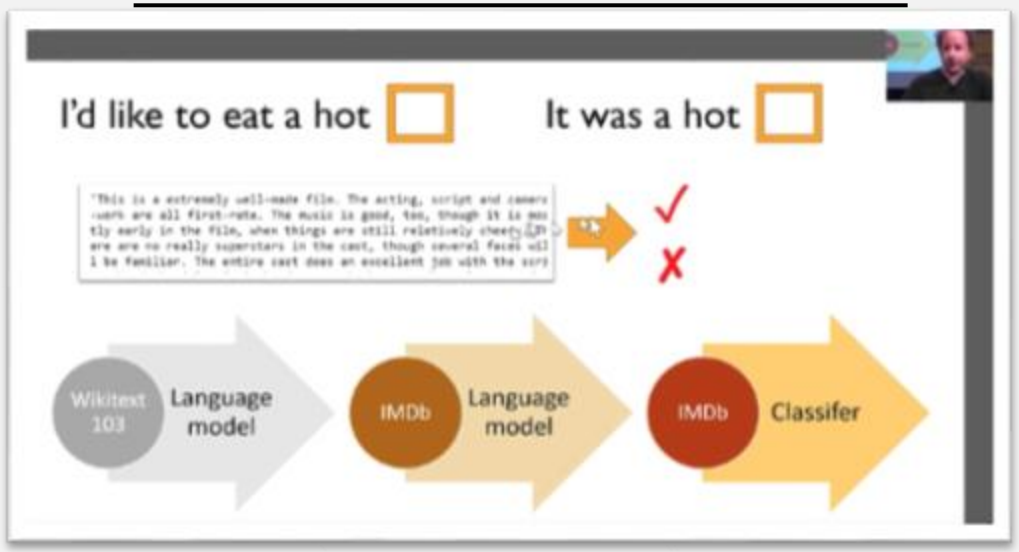


[1]: https://github.com/hiromis/notes/blob/master/Lesson3.md
[2]: https://forums.fast.ai/t/share-your-work-here/27676/399
[2]: https://forums.fast.ai/t/share-your-work-here/27676/340

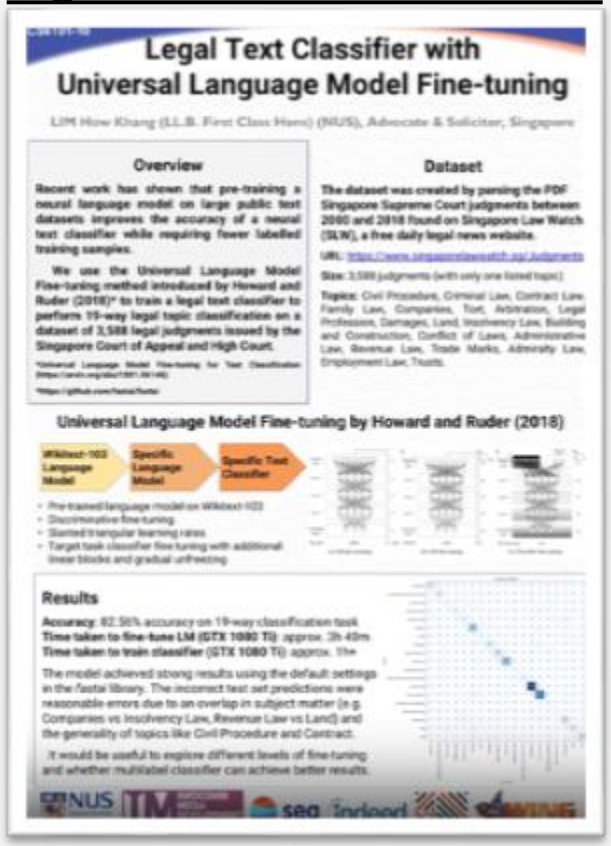
fast.ai not just classification

NLP

Movie Review Sentiment Classification



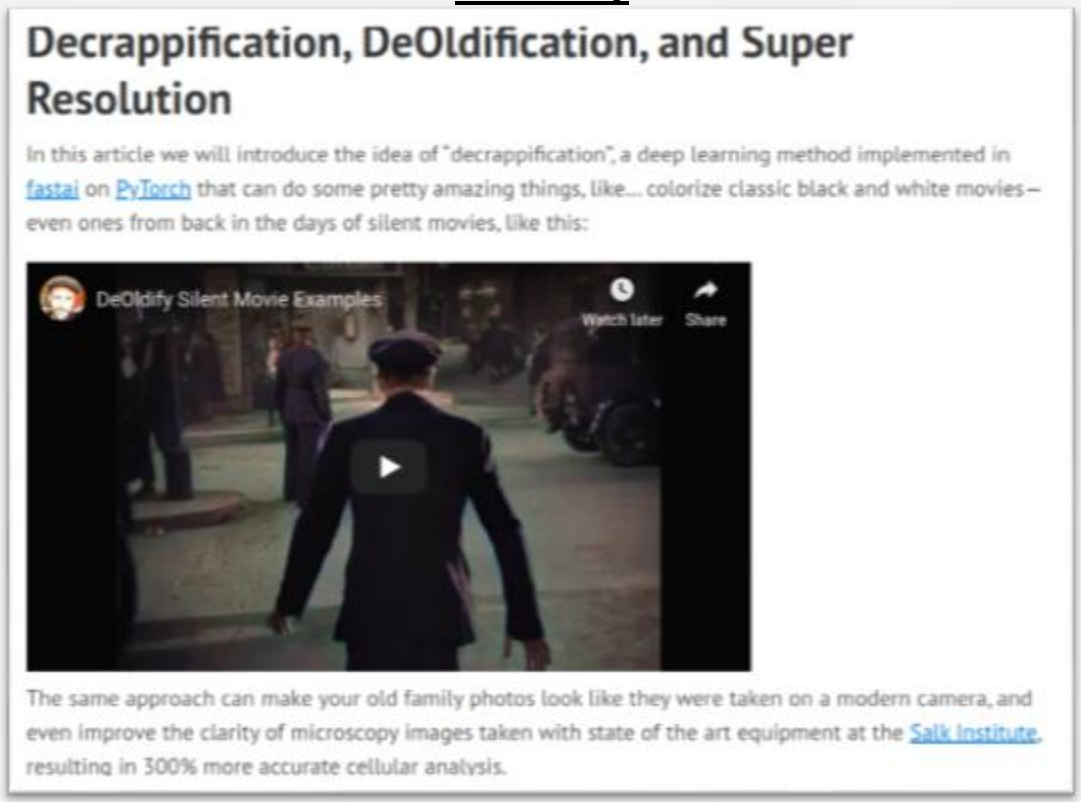
Legal Document Classification



[1]: https://forums.fast.ai/t/deep-learning-lesson-4-notes/30983

GANs

DeOldify

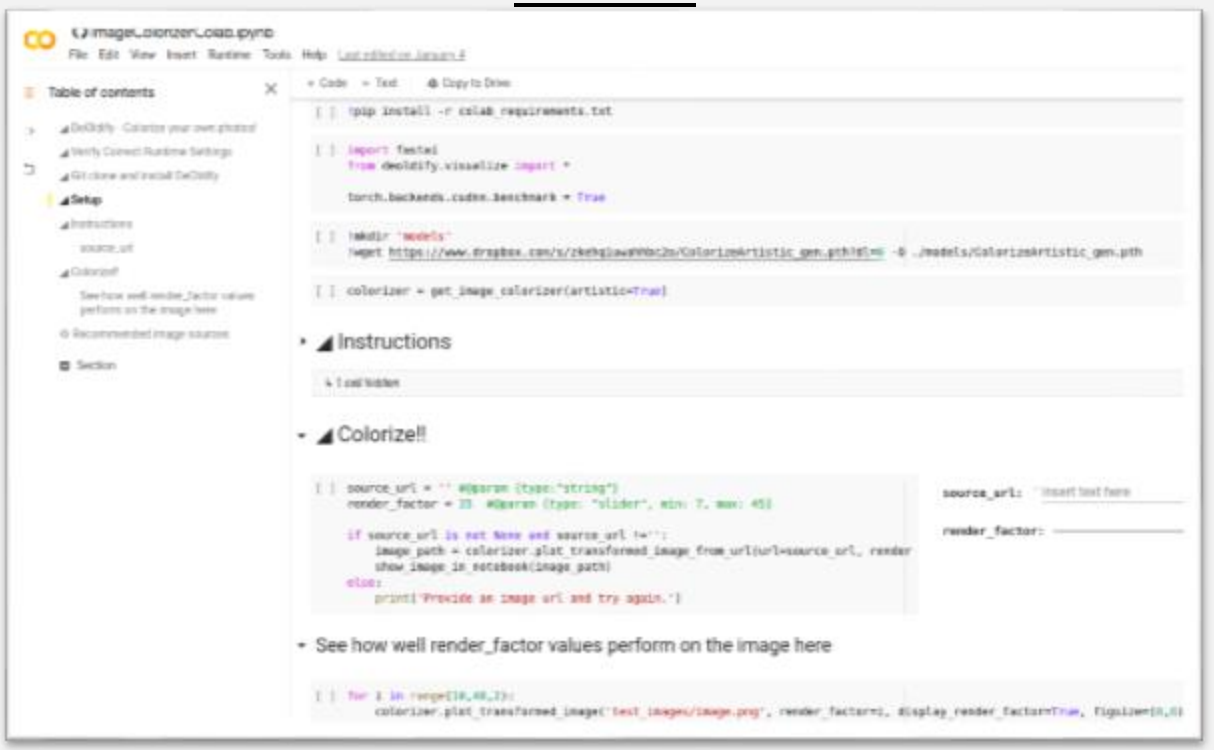


Decrappification, DeOldification, and Super Resolution

In this article we will introduce the idea of "decrappification", a deep learning method implemented in fastai on PyTorch that can do some pretty amazing things, like... colorize classic black and white movies—even ones from back in the days of silent movies, like this:

The same approach can make your old family photos look like they were taken on a modern camera, and even improve the clarity of microscopy images taken with state of the art equipment at the Salk Institute, resulting in 300% more accurate cellular analysis.

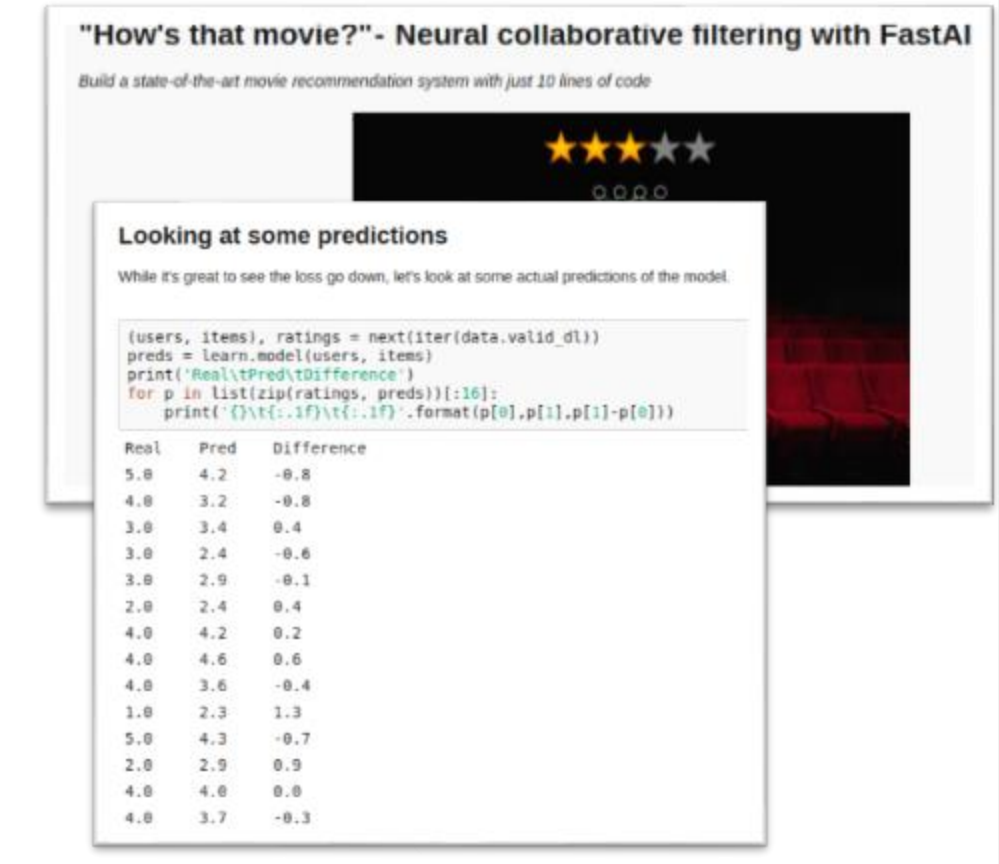
Colorizer



[1]: https://www.fast.ai/2019/05/03/decrappify/
[2]: https://colab.research.google.com/github/jantic/DeOldify/blob/master/ImageColorizerColab.ipynb#scrollTo=LHfUPH42O_iK

Collaborative Filtering

Movie Recommendation System



Book Recommendation System

```
1  learn = collab_learner(data, n_factors=40, y_range=(1, 5), wd=1e-1)
```

```
Output:
Top idx:
array(['5000', '3315', '3313', '3312', '3311', '3309', '3308',
'3307', '3306', '3304'], dtype='<U21')

Top names:
array(['Passion Unleashed (Demonica #3)', 'My Story', 'The Gargoyle',
'Pretty Baby', ...,
       'Top Secret Twenty-One (Stephanie Plum, #21)', 'The Warrior
Heir (The Heir Chronicles, #1)', 'Stone Soup',
       'The Sixth Man (Sean King & Michelle Maxwell, #5)'],
dtype='<U144')

Most negative bias:
[(tensor(-0.1021), 'The Almost Moon', 2.49),
 (tensor(-0.0341), 'Skinny Bitch', 2.9),
 (tensor(-0.0325), 'Bergdorf Blondes', 3.0),
 (tensor(-0.0316), 'The Particular Sadness of Lemon Cake', 2.93),
 (tensor(-0.0148), 'The Weird Sisters', 3.08)]
...
```


[1]: https://course.fast.ai/videos/?lesson=4
[2]: https://towardsdatascience.com/collaborative-filtering-with-fastai-3dbdd4ef4f00
[3]: https://jovian.ml/aakashns/5bc23520933b4cc187cfe18e5dd7e2ed

Reinforcement Learning (experimental)

[1]: https://course.fast.ai/videos/?lesson=4