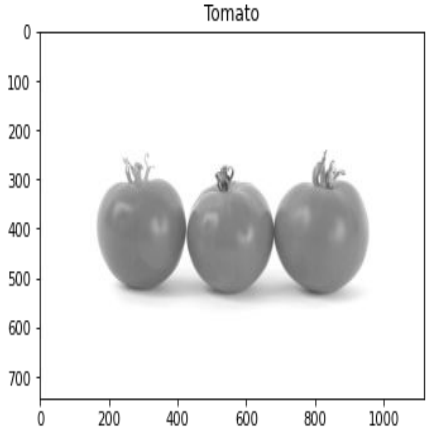
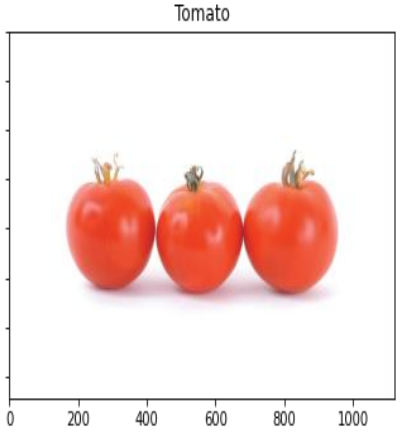
Image Colorization

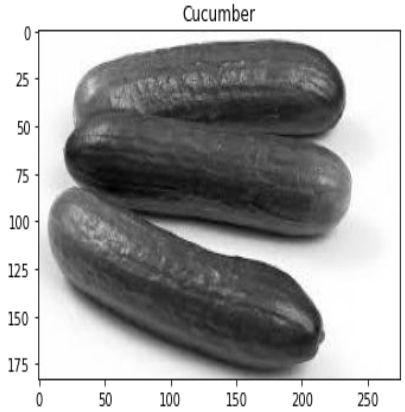
Dominik Macko

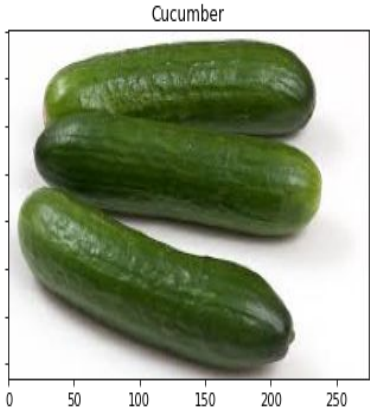
Preliminaries

- convert grayscale image to rgb image
- Natural-Color Dataset (NCD) from paper Image Colorization: Survey and Dataset (August 2020) – hard benchmark dataset







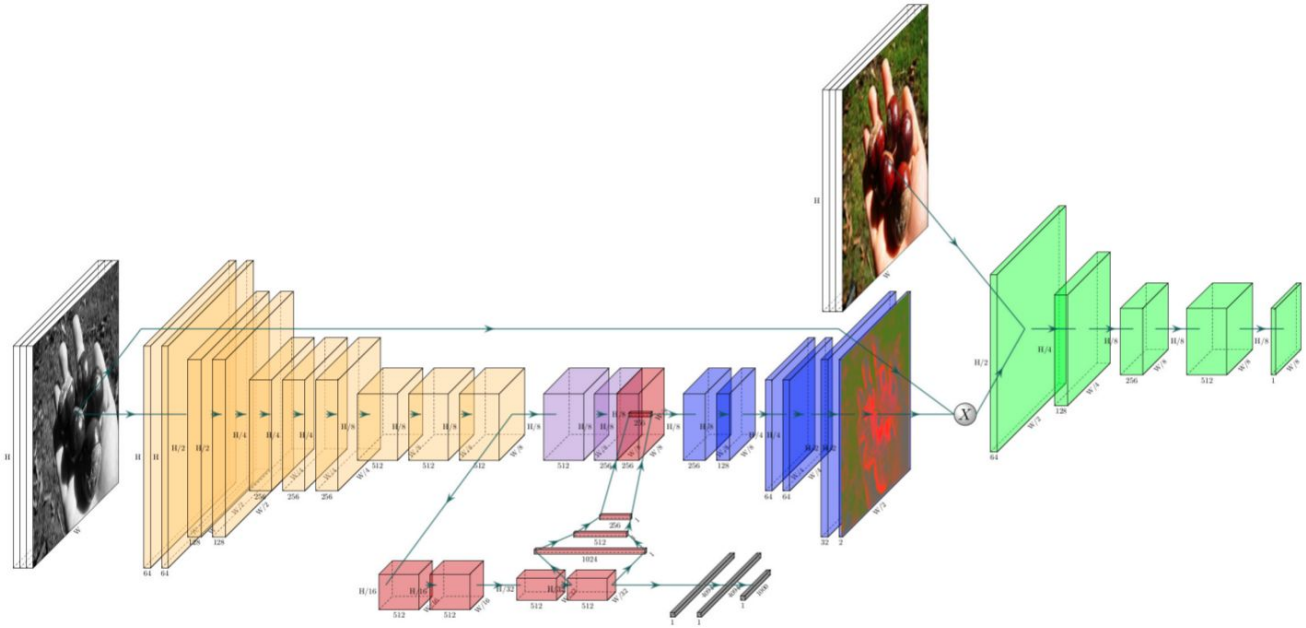


Models

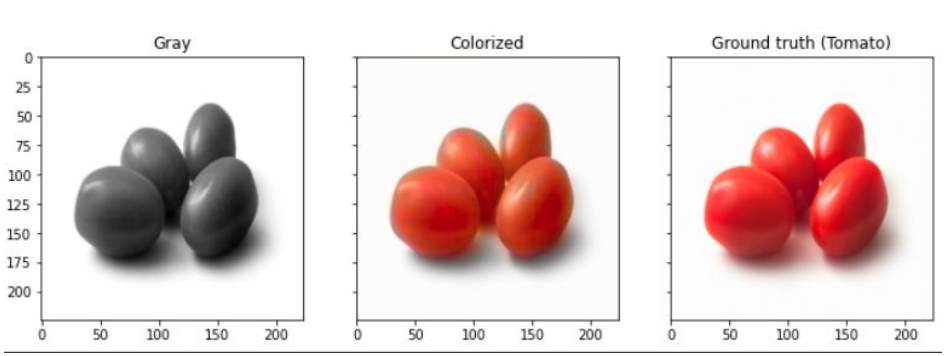
- segmentation (various CNN architectures),
- generative models (GAN, WGAN, ..)
- Transformers (Colorization Transformer)

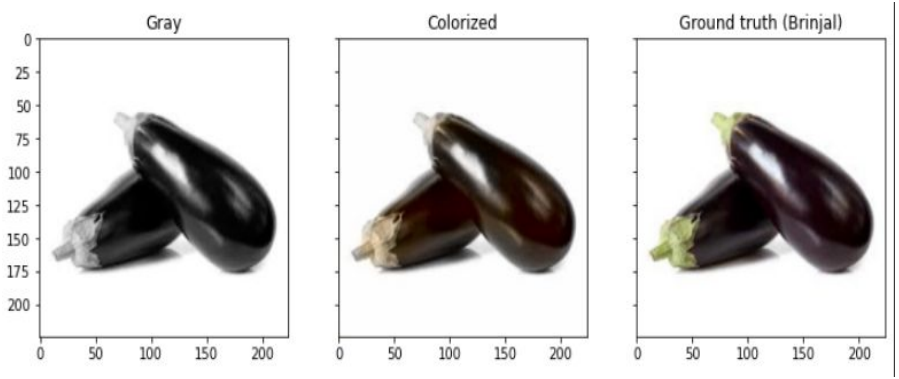
ChromaGAN

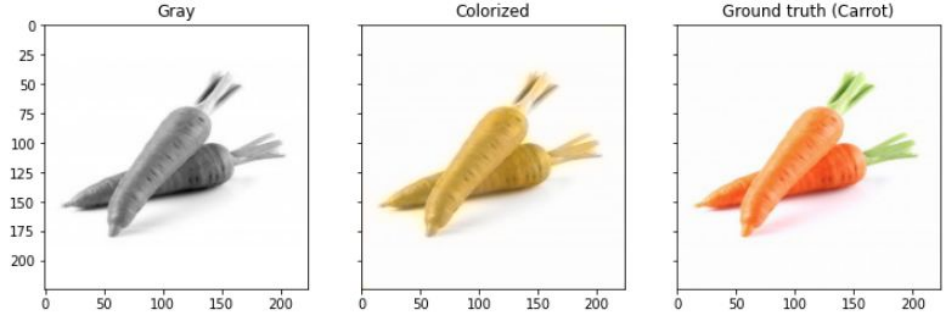
- generator: VGG16 backbone, outputs: class, 2 color channels
- discriminator: looks at patches, and determines realness
- Lab color space (Lightness, a-axis, b-axis)

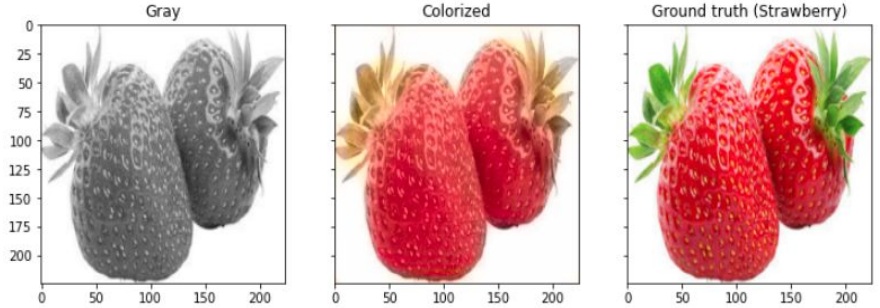


Results


Gray
Colorized
Ground truth (Tomato)


Gray
Colorized
Ground truth (Brinjal)


Gray
Colorized
Ground truth (Carrot)


Gray
Colorized
Ground truth (Strawberry)

Lessons taken

- Model definition took me longer than
- GradientPenaltyLoss implementation usable in Functional API in TF2 is very hard to implement (I had to go back to TF1)
- TensorFlow is so hard to debug

```
    TypeError: You are passing KerasTensor(type_spec=TensorSpec(shape=(16, 224, 224, 2), dtype=tf.float32, name=Non
e), name='random_weighted_average/add:0', description="created by layer 'random_weighted_average'"), an intermediat
e Keras symbolic input/output, to a TF API that does not allow registering custom dispatchers, such as `tf.cond`, `
tf.function`, gradient tapes, or `tf.map_fn`. Keras Functional model construction only supports TF API calls that *
do* support dispatching, such as `tf.math.add` or `tf.reshape`. Other APIs cannot be called directly on symbolic Ke
rasinputs/outputs. You can work around this limitation by putting the operation in a custom Keras layer `call` and
calling that layer on this symbolic input/output.
```

Thank you for your attention