

MARIA PIOTROWSKA 38671 & KAMIL PODWORSKI 38672

# ADVANCED PHYTON PROJECT

“NEURAL STYLE TRANSFER”

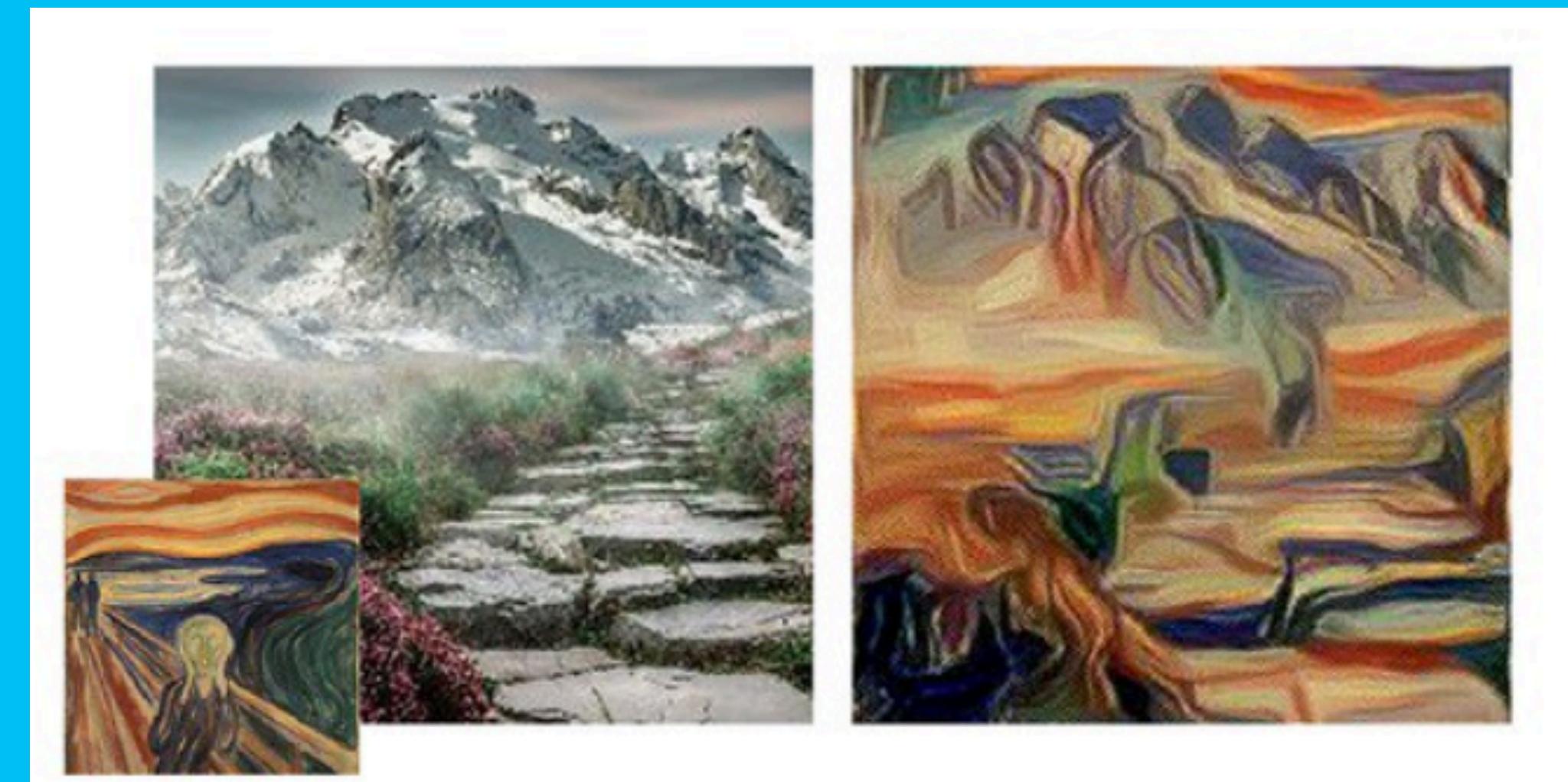
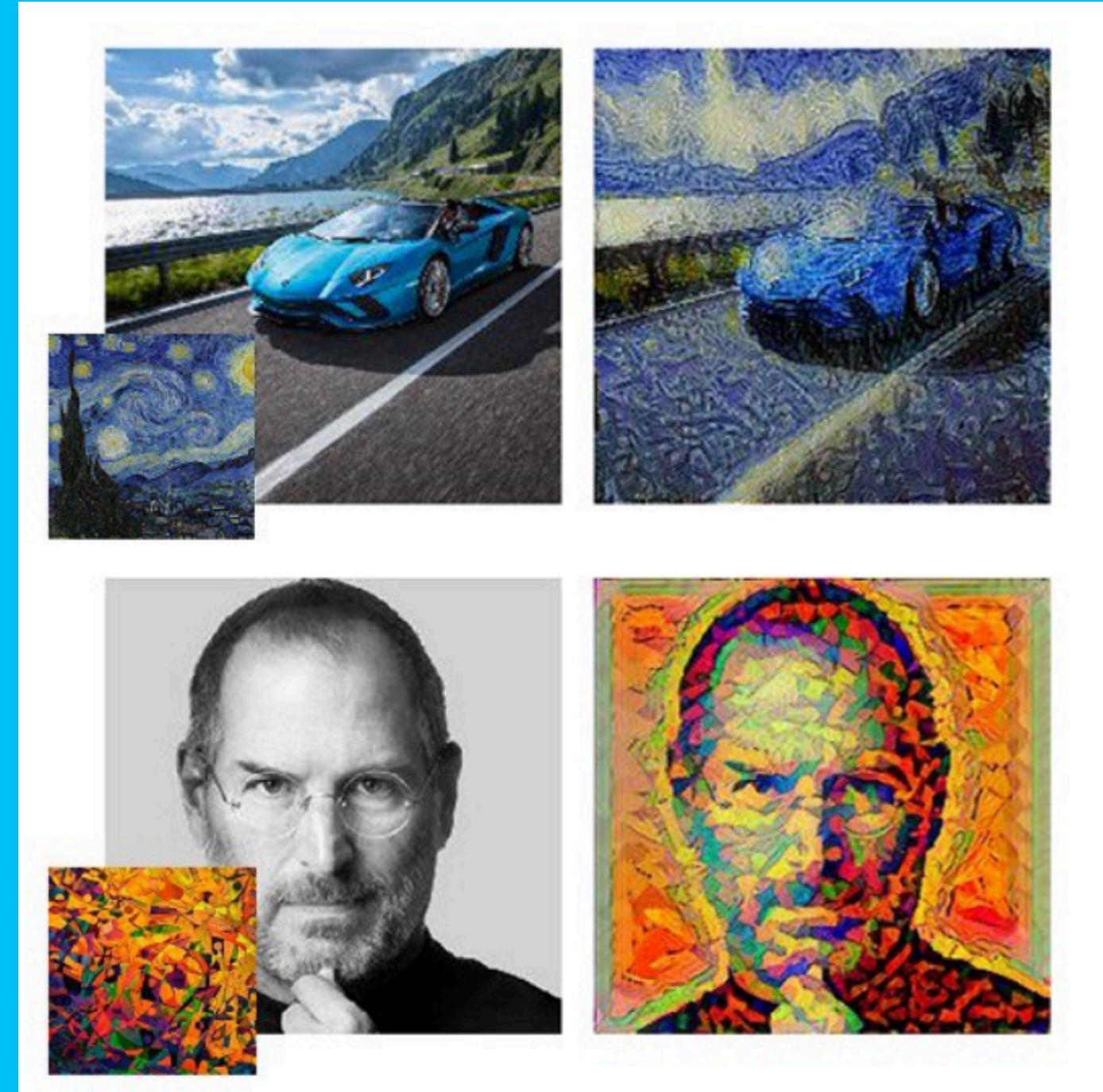
# AT THE BEGINNING

We searched through various websites to look for:

- inspiration
- knowledge
- tips

Link:

<https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398>



# FIRST ATTEMPT

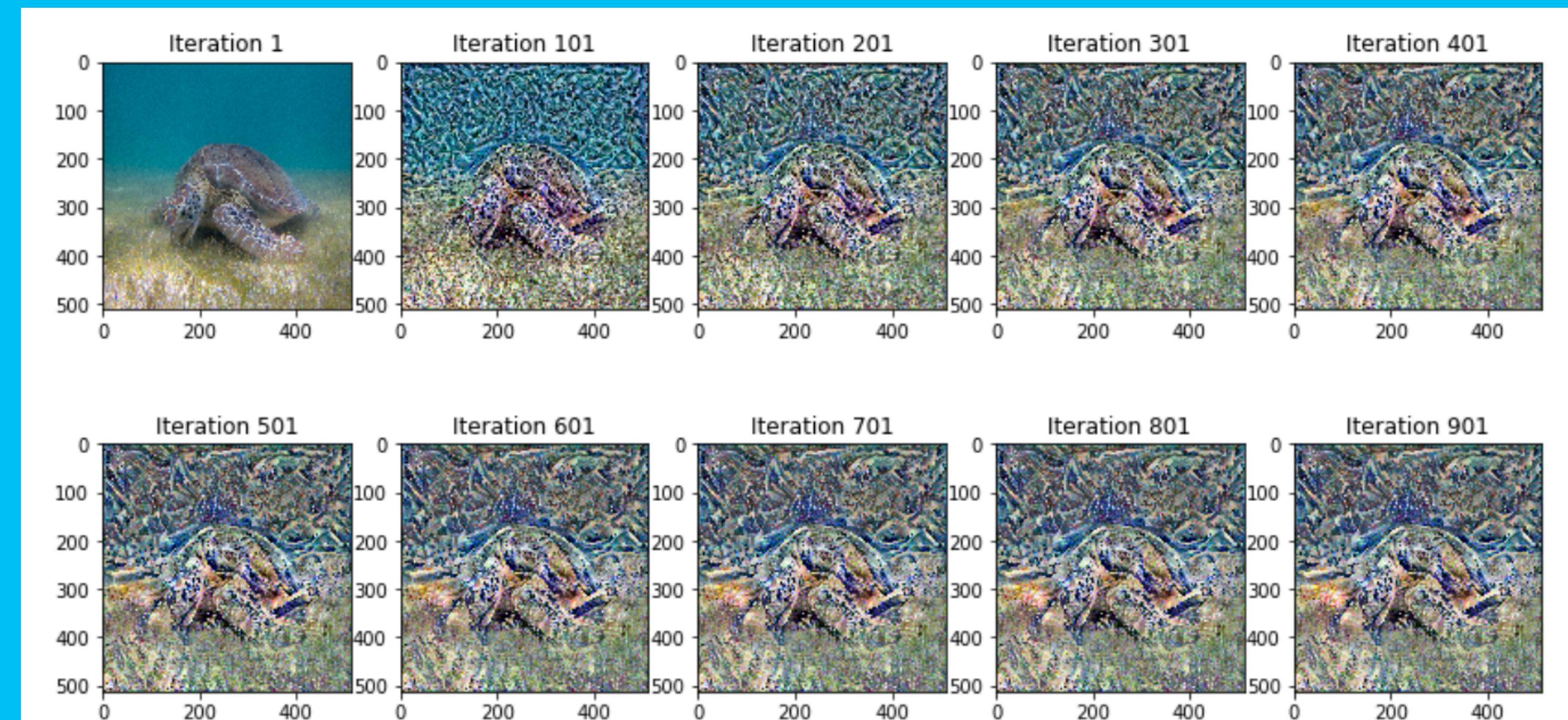
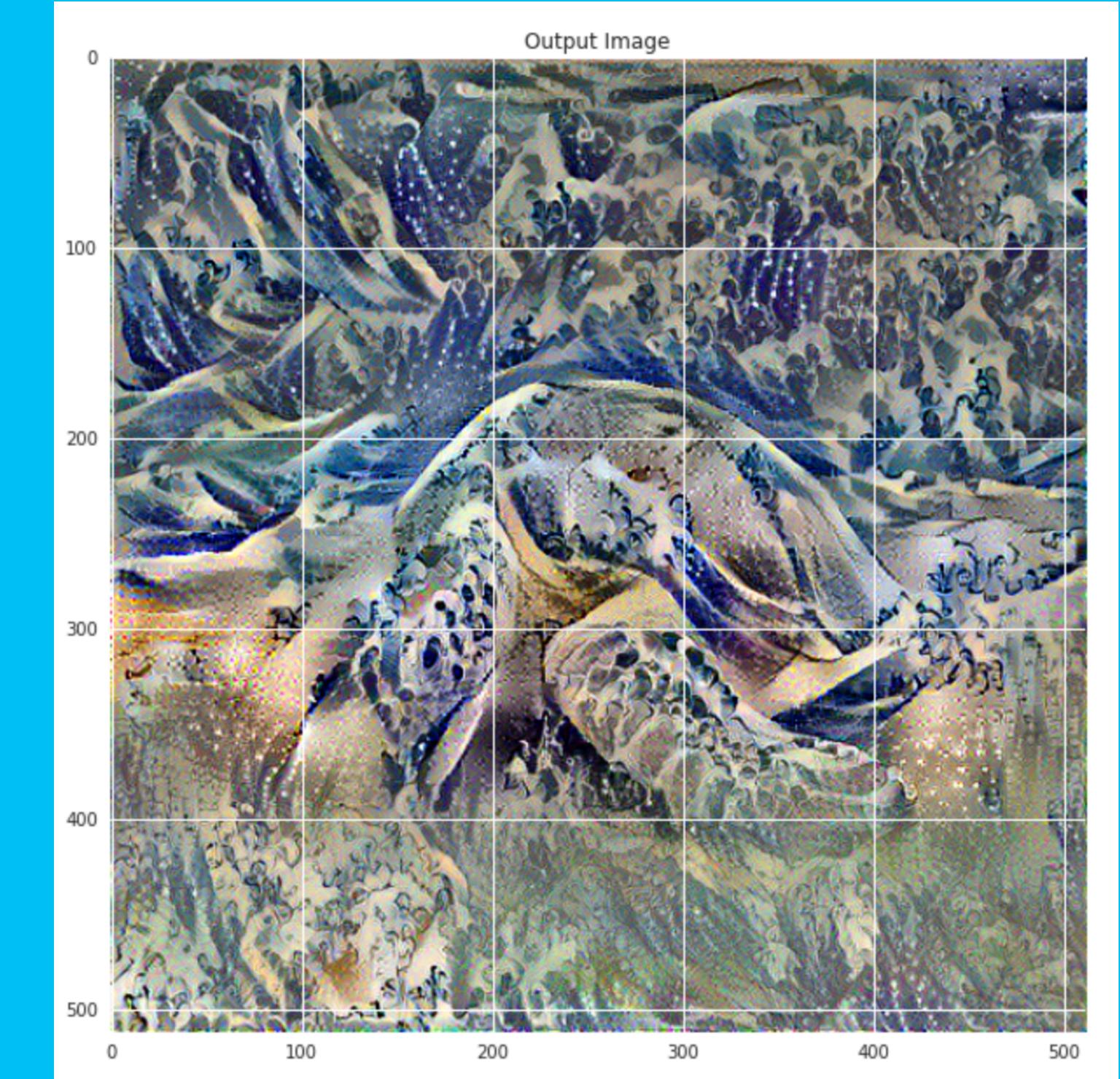
Was very difficult, cause we were ambitious!

But we had a keras problem on both computers.

We decided to look for something else.

Link:

<https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398>



# SECOND ATTEMPT

This instruction helped us solve a few problems.

We understood more, but the code was too complicated to tamper with it.

Link:

<https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-neural-style-transfer-ef88e46697ee>

The screenshot shows a blog post from TensorFlow.org. At the top, there's a navigation bar with links for ANNOUNCEMENTS, KERAS, TENSORFLOW.JS, MOBILE, RESPONSIBLE AI, ALL STORIES, and TENSORFLOW.ORG. A large, bold title 'Neural Style Transfer: Creating Art with Deep Learning using tf.keras and eager execution' is centered above the content. Below the title is a profile picture of the author, Raymond Yuan, with the text 'TensorFlow Follow Aug 3, 2018 · 10 min read'. The main text of the post begins with a paragraph about neural style transfer and its historical context, mentioning Leon A. Gatys' paper. The entire screenshot is framed by a thick blue border.

## Neural Style Transfer: Creating Art with Deep Learning using tf.keras and eager execution

TensorFlow Follow Aug 3, 2018 · 10 min read

By Raymond Yuan, Software Engineering Intern

In this [tutorial](#), we will learn how to use deep learning to compose images in the style of another image (ever wish you could paint like Picasso or Van Gogh?). This is known as **neural style transfer!** This is a technique outlined in [Leon A. Gatys' paper, A Neural Algorithm of Artistic Style](#), which is a great read, and you should definitely check it out.

A FEW ATTEMPTS LATER

# ANOTHER ATTEMPT

From tensorflow.org we took our primary code.

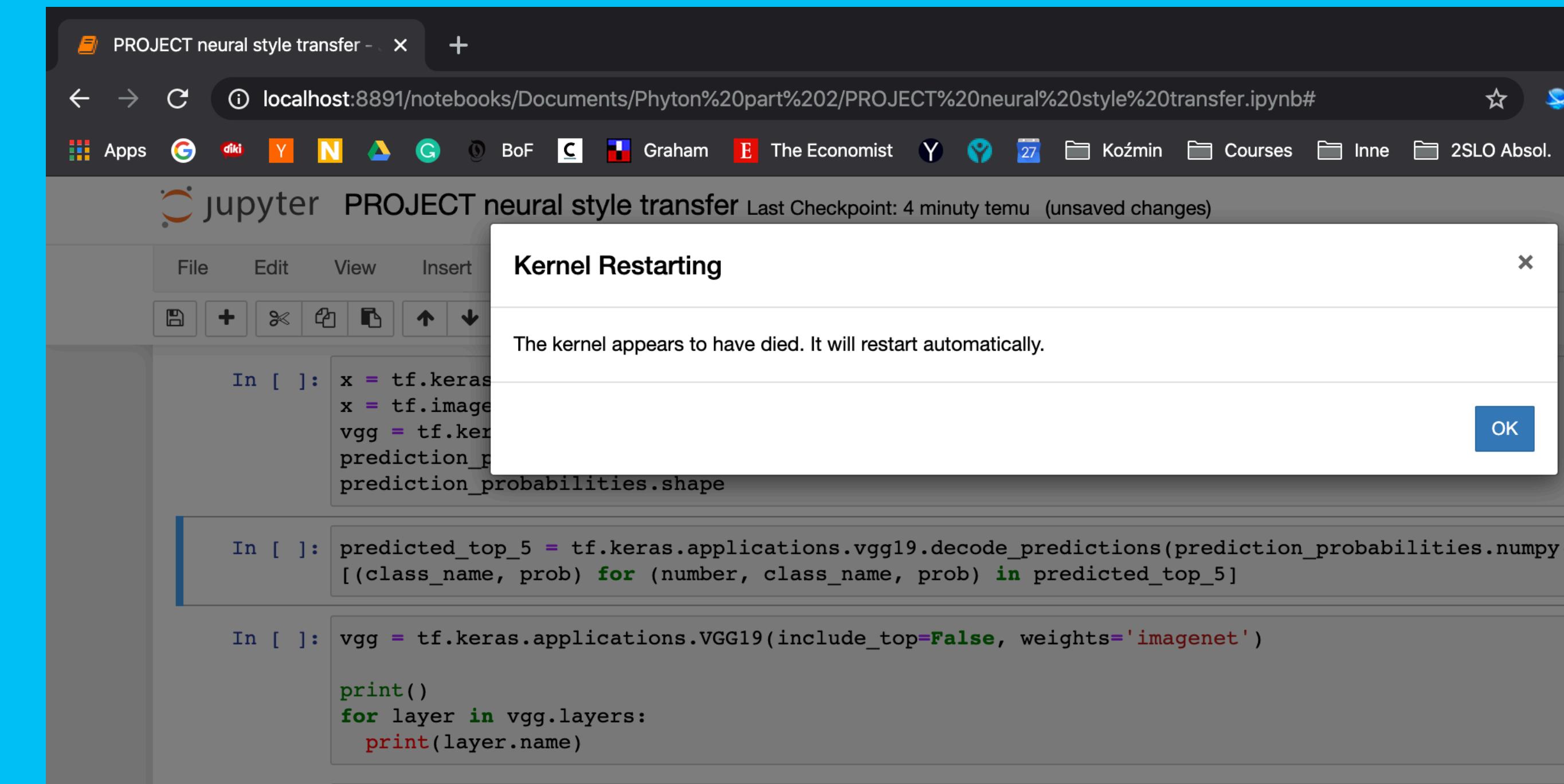
We changed it.

But we had a problem.

Even switching from new tensorflow to tensorflow 1.15.0 did not help.

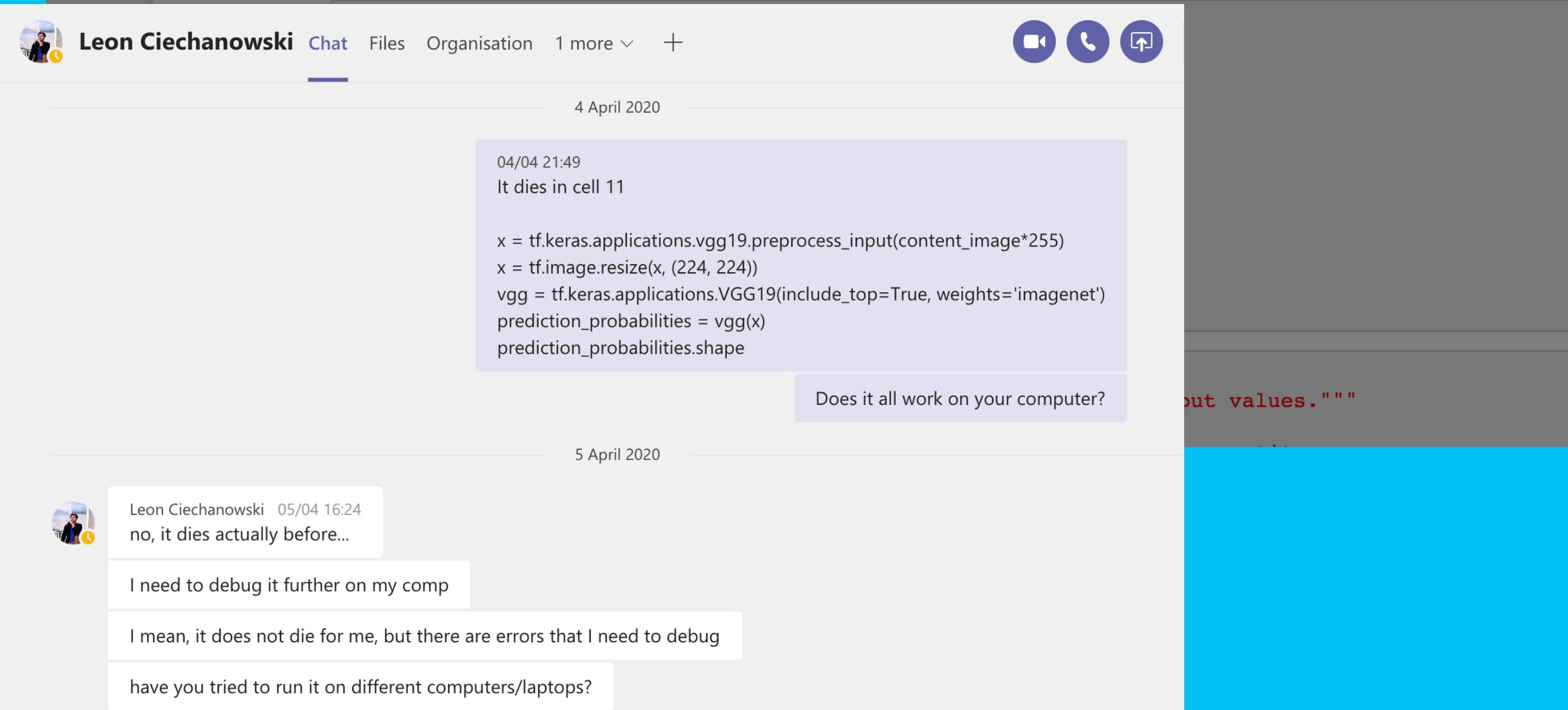
Link:

[https://www.tensorflow.org/tutorials/generative/style\\_transfer](https://www.tensorflow.org/tutorials/generative/style_transfer)



The screenshot shows a Jupyter Notebook interface with a 'Kernel Restarting' dialog box. The dialog states: 'The kernel appears to have died. It will restart automatically.' A blue 'OK' button is visible in the bottom right corner of the dialog. Below the dialog, the notebook code cells show the following Python code:

```
In [ ]: x = tf.keras  
x = tf.image  
vgg = tf.keras  
prediction_p  
prediction_probabilities.shape  
  
In [ ]: predicted_top_5 = tf.keras.applications.vgg19.decode_predictions(prediction_probabilities.numpy()  
[(class_name, prob) for (number, class_name, prob) in predicted_top_5]  
  
In [ ]: vgg = tf.keras.applications.VGG19(include_top=False, weights='imagenet')  
  
print()  
for layer in vgg.layers:  
    print(layer.name)
```



The screenshot shows a Microsoft Teams chat window with Leon Ciechanowski. The message history includes:

- Leon Ciechanowski 04/04 21:49: It dies in cell 11
- Leon Ciechanowski 04/04 21:49: 

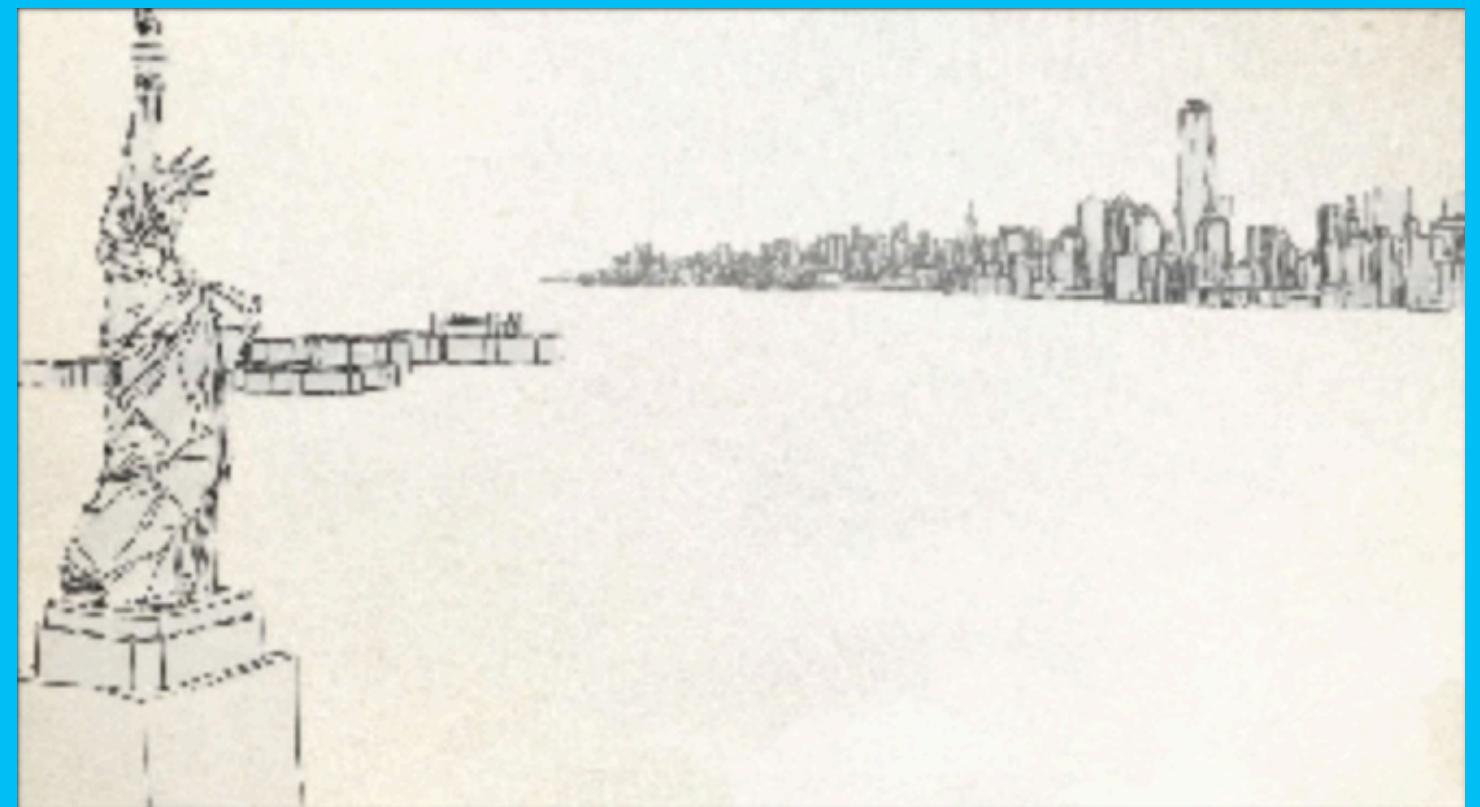
```
x = tf.keras.applications.vgg19.preprocess_input(content_image*255)  
x = tf.image.resize(x, (224, 224))  
vgg = tf.keras.applications.VGG19(include_top=True, weights='imagenet')  
prediction_probabilities = vgg(x)  
prediction_probabilities.shape
```
- Leon Ciechanowski 05/04 16:24: Does it all work on your computer?
- Leon Ciechanowski 05/04 16:24: 04/04 21:49: It dies in cell 11
- Leon Ciechanowski 05/04 16:24: no, it dies actually before...
- Leon Ciechanowski 05/04 16:24: I need to debug it further on my comp
- Leon Ciechanowski 05/04 16:24: I mean, it does not die for me, but there are errors that I need to debug
- Leon Ciechanowski 05/04 16:24: have you tried to run it on different computers/laptops?

**BUT FINALLY WE MANAGED TO...**

**DO THIS**



**FROM THIS**



# BACKGROUND

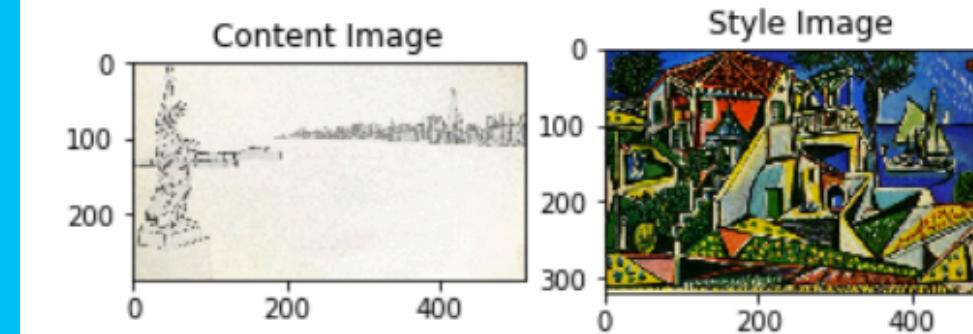
Because of a previously mentioned problem with tensorflow, we run just a part of the code.

But luckily we could see the results of neural style transfer.

```
content_image = load_img(content_path)
style_image = load_img(style_path)

plt.subplot(1, 2, 1)
imshow(content_image, 'Content Image')

plt.subplot(1, 2, 2)
imshow(style_image, 'Style Image')
```



```
import tensorflow_hub as hub
hub_module = hub.load('https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/1')
stylized_image = hub_module(tf.constant(content_image), tf.constant(style_image))[0]
tensor_to_image(stylized_image)
```



```
x = tf.keras.applications.vgg19.preprocess_input(content_image*255)
x = tf.image.resize(x, (224, 224))
vgg = tf.keras.applications.VGG19(include_top=True, weights='imagenet')
prediction_probabilities = vgg(x)
prediction_probabilities.shape
```

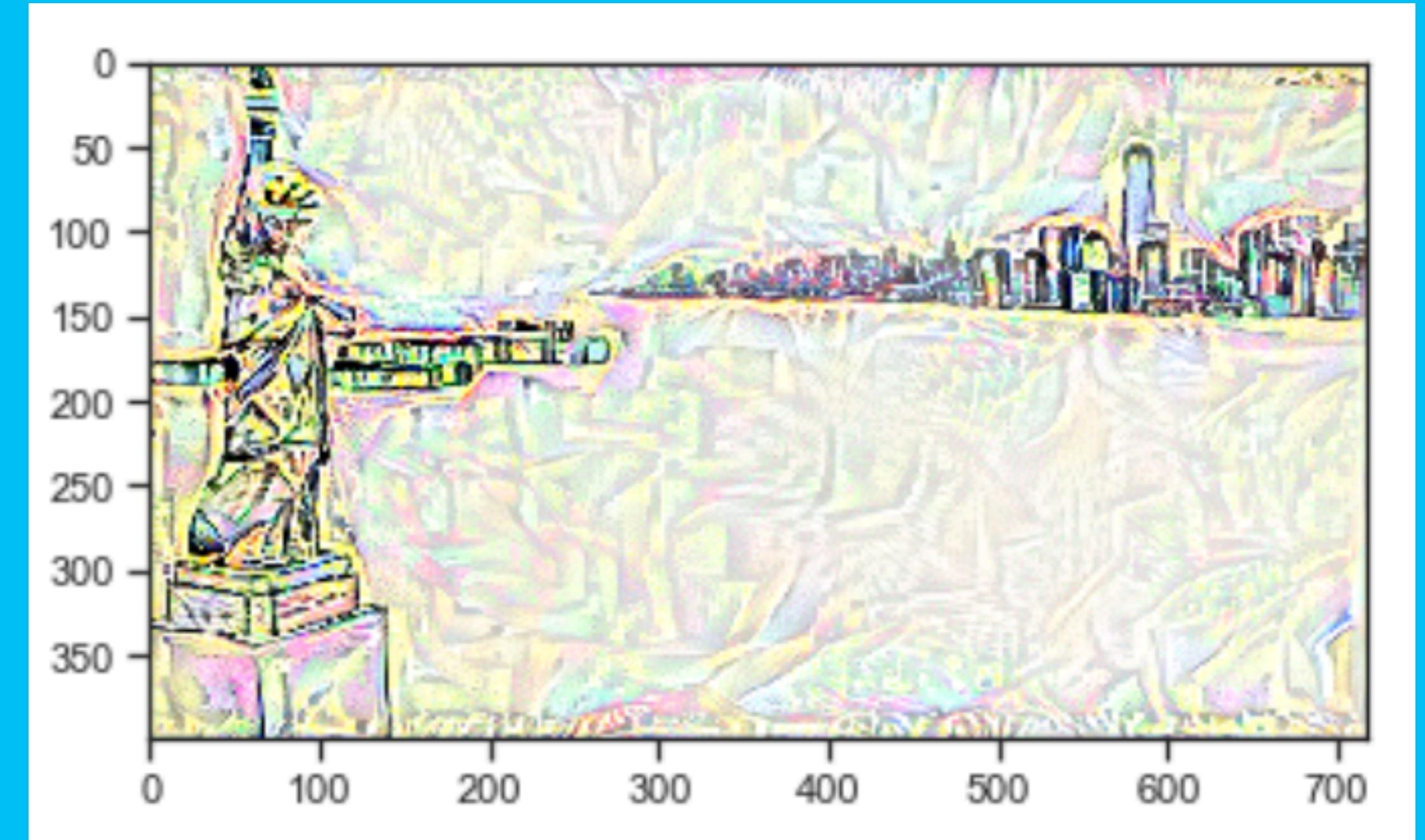
```
predicted_top_5 = tf.keras.applications.vgg19.decode_predictions(prediction_probabilities.numpy())[0]
[(class_name, prob) for (number, class_name, prob) in predicted_top_5]
```

# ALSO...

## AFTER NEURAL STYLE TRANSFER CLASSES:

We adjusted the code from our classes to our photos.

However the result was not as good as with the previous code.



# THANK YOU FOR READING!



MARIA PIOTROWSKA 38671 & KAMIL PODWORSKI 38672