**­­­**

**VILNIUS UNIVERSITY**

**ŠIAULIAI ACADEMY**

BACHELOR PROGRAMME SOFTWARE ENGINEERING

**Artificial Intelligence**

**Report on**

**“Investigation of pretrained computer vision models” task**

Student: Anna Kutova

Lecturer: Dr. Gintautas Daunys

Šiauliai, 2023

*Table of Contents*

[*Table of Contents* 2](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151820)

[1. My dataset 3](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151821)

[2. Models I used 3](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151822)

[3. Code I used 3](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151823)

[4. Code explanation 4](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151824)

[5. Accuracy of each pretrained model 5](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151825)

[▪ Resnet18 5](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151826)

[▪ Inception-v3 6](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151827)

[▪ Efficientnet\_b0 6](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151828)

[6. About my results 7](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151829)

[7. Conclusion 7](applewebdata://58971528-F723-43B4-94F5-1397041A427B#_Toc153151830)

1. **My dataset**

For this task I’ve got a dataset, which contains a lot of magnificent goldfish pictures

1. **Models I used**

In this task, I used three models, which are:

* Resnet18
* Inception-v3
* Efficientnet\_b0

Here is a part of code, where I have loaded them:

# Loading the pre-trained model  
resnet = torchvision.models.resnet18(weights='DEFAULT').eval()  
efficientnet = torchvision.models.efficientnet\_b0(weights='DEFAULT').eval()  
inception = inception\_v3(pretrained=True).eval()

1. **Code I used**

from torchvision.transforms.functional import normalize  
from torchvision.io.image import read\_image  
from sklearn.metrics import accuracy\_score  
from torchvision.models import inception\_v3  
import matplotlib.pyplot as plt  
import torchvision  
import numpy as np  
import json  
import os  
  
# Loading the pre-trained model  
resnet = torchvision.models.resnet18(weights='DEFAULT').eval()  
efficientnet = torchvision.models.efficientnet\_b0(weights='DEFAULT').eval()  
inception = inception\_v3(pretrained=True).eval()  
  
# Loadin the imagenet\_class\_index.json file to map class index to label  
with open('imagenet\_class\_index.json') as labels\_file:  
 labels = json.load(labels\_file)  
  
# Defining the path to the folder containing the images  
folder\_path = 'goldfish'  
  
# Initializing lists to collect the predicted and actual labels  
temp\_predicted\_labels = []  
actual\_labels = []  
  
# Creating a list of tuples, each containing a model and its name  
models = [  
 ('ResNet-18', resnet),  
 ('Inception-v3', inception),  
 ('EfficientNet-B0', efficientnet)  
]  
  
def test\_model(model\_name, model):  
  
 # Iterating over each image file in the folder and apply the script  
 for filename in os.listdir(folder\_path):  
 if filename.endswith('.jpg') or filename.endswith('.jpeg') or filename.endswith('.png'):  
  
 # Loading the image and transform it to match the format required by the model  
 image\_path = os.path.join(folder\_path, filename)  
 img = read\_image(image\_path)  
  
 plt.imshow(img.permute(1, 2, 0))  
 input\_tensor = normalize(img / 255., [0.485, 0.456, 0.406], [0.229, 0.224, 0.225])  
 out = model(input\_tensor.unsqueeze(0))  
 out = out.detach().cpu().numpy()  
 idx = np.argmax(out, axis=1)  
 str\_idx = str(idx[0])  
 predicted\_label = labels[str\_idx]  
 print(f" {model\_name} : {predicted\_label}")  
  
 actual\_label = 'goldfish'  
  
 # Collecting the predicted and actual labels  
 temp\_predicted\_labels.append(predicted\_label[1])  
 actual\_labels.append(actual\_label)  
  
 predicted\_labels = []  
 for label in temp\_predicted\_labels:  
 if 'goldfish' in label:  
 predicted\_labels.append('goldfish')  
 else:  
 predicted\_labels.append(label)  
  
 accuracy = accuracy\_score(actual\_labels, predicted\_labels)  
 print('Accuracy: {:.2f}%\n'.format(accuracy \* 100))  
  
for model\_name, model in models:  
 test\_model(model\_name, model)

1. **Code explanation**

Firstly, the code begins by importing necessary modules such as the torchvision library for working with computer vision models, sklearn for evaluating accuracy, and other standard libraries.

Next, the pre-trained models, ResNet-18, EfficientNet-B0, and Inception-v3, are loaded and set to evaluation mode using the **.eval()** method.

The script then loads the **imagenet\_class\_index.json** file, which maps class indices to human-readable labels, using the json library.

The path to the folder containing the images ('goldfish' in this case) is defined.

Subsequently, two lists, **temp\_predicted\_labels** and **actual\_labels**, are initialized to collect predicted and actual labels during the testing process.

A list of tuples, **models**, is created, each containing a model name and its corresponding pretrained model.

The **test\_model** function is defined to evaluate each model. It iterates over each image file in the specified folder, loads the image, and transforms it into the format required by the model. The model then predicts the label, and the results are printed with the model name. The actual label is set as 'goldfish' in this case.

After iterating through the dataset, the script checks if the predicted labels contain the word 'goldfish', considering it a valid prediction. Finally, it calculates and prints the accuracy using the sklearn **accuracy\_score** function.

Lastly, a loop iterates over the list of models, calling the **test\_model** function for each model, and providing the model name and the model itself as arguments.

In summary, the code loads pretrained models, tests them on a dataset of goldfish images, and evaluates their accuracy, considering 'goldfish' as the expected label. The results are printed for each model, showing the predicted labels and the overall accuracy.

1. **Accuracy of each pretrained model**

* Resnet18

ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n04606251', 'wreck']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01930112', 'nematode']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n03950228', 'pitcher']  
 ResNet-18 : ['n01698640', 'American\_alligator']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01498041', 'stingray']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01950731', 'sea\_slug']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n04275548', 'spider\_web']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n04209239', 'shower\_curtain']  
 ResNet-18 : ['n01443537', 'goldfish']  
 ResNet-18 : ['n01443537', 'goldfish']

Accuracy: 84.00%

* Inception-v3

Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01632777', 'axolotl']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01737021', 'water\_snake']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']  
 Inception-v3 : ['n01443537', 'goldfish']

Accuracy: 90.00%

* Efficientnet\_b0

EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n09332890', 'lakeside']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01950731', 'sea\_slug']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01737021', 'water\_snake']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']  
 EfficientNet-B0 : ['n01443537', 'goldfish']

Accuracy: 91.33

1. **Explanation of my results**

The evaluation of the pretrained computer vision models on the goldfish image dataset demonstrated notable performance.

ResNet-18 achieved an accuracy of 84.00%, Inception-v3 demonstrated improved accuracy at 90.00%, and EfficientNet-B0 showcased the highest accuracy of 91.33%.

These results underscore the effectiveness of these models in goldfish image recognition, with EfficientNet-B0 exhibiting the most robust performance.

1. **Conclusion**

*To sum up, this task represented a comprehensive exploration of pretrained computer vision models in the domain of image recognition. The investigation employed the PyTorch framework and focused on a dataset specifically comprised of images of goldfish.*

*Throughout the task, meticulous attention was devoted to three pretrained models: ResNet-18, Inception-v3, and EfficientNet, assessing their proficiency in identifying goldfish images.*

*The results unveiled commendable performance across all three models, with EfficientNet demonstrating a subtle advantage in recognizing goldfish images compared to ResNet-18 and Inception-v3. Noteworthy is the impact of fine-tuning on the models, highlighting its role in optimizing their performance for the specific characteristics of the goldfish dataset.*

*This project not only underscores the efficacy of pretrained computer vision models in image recognition tasks but also emphasizes the nuanced role of fine-tuning in tailoring these models to specialized datasets.*

*These insights hold significant applications, ranging from automated quality control in manufacturing processes to the development of robust environmental monitoring systems. Moreover, they can play a crucial role in improving the accuracy of facial recognition technology, ensuring its reliability in diverse security applications.*