Where is Waldo?

Project AI

Team members:

- Miguel Haest
- Dries Augustyns
- Niels Baptist

Link to Github repo (https://github.com/driaug/where-is-waldo) .

Introduction

We chose the challenge of finding Waldo in a large puzzle filled with Waldo lookalikes, because we were convinced this would be a straightforward assignment. We looked at the different options that existed to solve this problem. Some of the approaches we took included the usage of YOLO, working with Haar cascades and training our own model. We were however met with an incoherent mess of struggles and a lack of finding Waldo. This is why we finally settled on a different approach: we take an existing puzzle of 'Where is Waldo' and slice it up in chunks of 256x256, for each little image, we use a model that we trained using transfer learning to decide wether Waldo is on the image or not. We use flask to host a little web application so that we can demo our solution.

Doing the necessary imports

In [ ]:

```
import fastbook
fastbook.setup_book()
from fastbook import *
from fastai.vision.widgets import *
from PIL import Image
from fastai.vision.all import *
```

Preparing the images

To start, we had to link the path to our images and verify that all of the files in the images/256 folder are supported image formats.

In [ ]:

```
path = '../images/256/'
filenames = get_image_files(path)
filenames
```

Out[ ]:

```
(#373) [Path('../images/256/notwaldo/10_0_0.jpg'),Path('../images/256/notw
aldo/10_0_1.jpg'),Path('../images/256/notwaldo/10_0_2.jpg'),Path('../image
s/256/notwaldo/10_0_3.jpg'),Path('../images/256/notwaldo/10_1_0.jpg'),Path
('../images/256/notwaldo/10_1_2.jpg'),Path('../images/256/notwaldo/10_1_3.
jpg'),Path('../images/256/notwaldo/10_2_0.jpg'),Path('../images/256/notwal
do/10_2_1.jpg'),Path('../images/256/notwaldo/10_2_2.jpg')...]
```

Checking wether there are any corrupted images

In [ ]:

```
failed = verify_images(filenames)
failed
```

Out[ ]:

```
(#0) []
```

We split the dataset and provided a generous 40% validation set, since we had few images to work with and we wanted an estimate of the performance with which our model performed.

In [ ]:

```
img = DataBlock(
    blocks=(ImageBlock, CategoryBlock),
    get_items=get_image_files,
    splitter=RandomSplitter(valid_pct=0.4, seed=42),
    get_y=parent_label
    # item_tfms=[Resize(64)]
    )
```

In [ ]:

```
dls = img.dataloaders(path)
```

```
Due to IPython and Windows limitation, python multiprocessing isn't availa
ble now.
So `number_workers` is changed to 0 to avoid getting stuck
```

Performing data augmentation

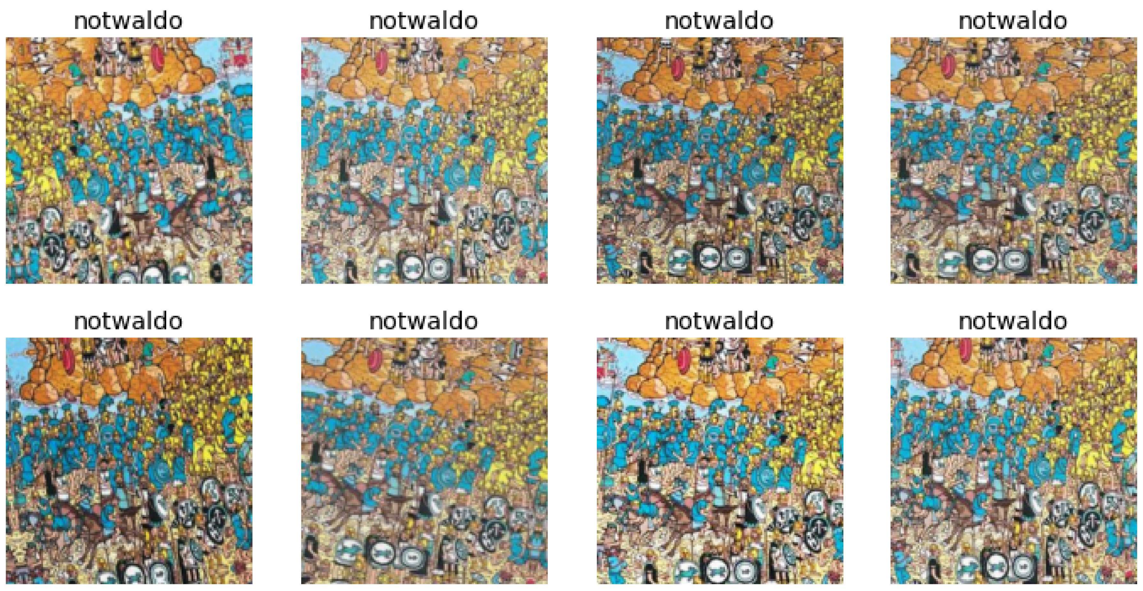
Because we were left with rather few images to work with, we decided to a perform a number of augmentation techniques that enhanced the accuracy of our model. The augmentations that we decided to go with are:

- rotate
- flip
- brightness
- contrast

In [ ]:

```
img = img.new(item_tfms=RandomResizedCrop(256, min_scale=0.8), batch_tfms=[Rotate(), Fl
ip(), Brightness(), Contrast()])
dls = img.dataloaders(path)
dls.train.show_batch(max_n=8, nrows=2, unique=True)
```

```
Due to IPython and Windows limitation, python multiprocessing isn't availa
ble now.
So `number_workers` is changed to 0 to avoid getting stuck
```



Transfer learning

In [ ]:

```
metrics = [accuracy, error_rate]
our_out_of_the_box_model = cnn_learner(dls, resnet152, loss_func=CrossEntropyLossFlat
(), metrics=metrics)
```

Retraining the model for Waldo recognition

Even though our accuracy drops throughout the different epochs, we noticed something very important: upon using different models it came to our attention that each time, the model had difficulties recognizing Waldo, where it didn't have difficulties recognizing 'not waldo'. This all changed when we started using Resnet152, as suddenly, our model started recognizing Waldo much better and we got better results in our real-life tests with our application.

In [ ]:

```
our_out_of_the_box_model.fine_tune(5)
```

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.966022	0.480307	0.785235	0.214765	03:41

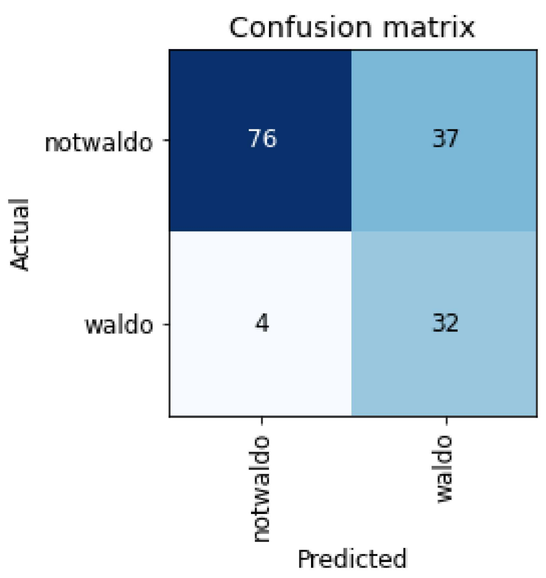
epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.542572	0.321277	0.879195	0.120805	04:08
1	0.516291	0.554498	0.758389	0.241611	04:18
2	0.457921	0.864789	0.657718	0.342282	04:06
3	0.391280	0.879220	0.664430	0.335570	03:56
4	0.359327	0.675475	0.724832	0.275168	03:49

Confusion matrix

As explained above, the fact that our model is good at recognizing waldos is crucial to our decision making when it comes to choosing the model. We blame the large amounts of false predictions concerning 'notwaldos' to the similarity of the different figures and patterns.

In [ ]:

```
interp = ClassificationInterpretation.from_learner(our_out_of_the_box_model)
interp.plot_confusion_matrix()
```



Saving the model

In [ ]:

```
our_out_of_the_box_model.save('model')
our_out_of_the_box_model.export()
```

Running the application

With your terminal in the 'app' folder, use the command **python ./main.py** to run the web application. Give it a moment to spin up and you can find it on Localhost:4000 (http://Localhost:4000) .

Conclusion

If we step back and look at the journey of which this project existed, it's fair to say that we underestimated the difficulty of the assignment. However, with each setback or challenge that we faced, we put our heads together and worked to solve it as a team, with everybody working together to solve it. Despite the fact that there are still a number of things we feel can be improved on our solution, we are proud to have come to this working result. In the end, it's easy to forget that the whole point of the 'Where-is-waldo'-series is to disguise Waldo amongst different figures that closely resemble him.

Possible improvements

- The usage of a larger dataset containing more images of Waldo
- Speeding up the model, which is slow right now due to the nature of cutting up the image into smaller pieces and evaluating each image, one at a time.
- The general accuracy and the occurence of 'false positives' in challenging puzzles.

We were exited to work together, as we previously had a great time with Data Science project. We are confident with what we are handing in and look forward to our presentation very much!