Using frequency information to improve accuracy of object detectors

Petar Ulev, Attila Lengyel, Silvia Pintea

TU Delft

June 27, 2021

Abstract

This research paper analyses the effect that using frequency information can have in object detectors. The latter are complex networks that learn information about objects from images and are then able to predict the location of these objects in new, unseen images. There are, however, certain datasets that are hard to learn on, partly because the environment in which images are taken is diverse and complex, and also because the objects to detect can appear in fairly different shapes. The dataset considered in this paper is called the Global Wheat Head Dataset (GWHD, provided by a Kaggle competition). An object detector is run on the original GWHD images and then the performance is compared to running the detector on a frequency filtered version of the images. A mathematical transform called Fourier Transform is used to map images from their spatial (pixel) domain to a new domain called the frequency domain, where certain non-informative frequencies are filtered out and then the images are mapped back to their spatial domain. Two experiments were conducted and results show that with this specific filtering methodology, no improvement is found on the GWHD dataset using an object detector called YoloV5. A pipeline was developed which allows for custom filtering strategy implementations and customs datasets. Similar work has shown that images in their frequency domain can speed up computational time and also increase the accuracy of an object detector, so this paper also gives the opportunity for further experiments with the created pipeline.

1 Introduction

Deep learning has gained a lot of popularity in recent years because of the large increase in computing power and information. Many industries make use of, directly or indirectly, machine learning (ML) and data analysis algorithms. A subfield of ML that is vastly used in the industry is computer vision (CV). Object detection is a CV technique that is used to locate objects in images and videos and surround them with bounding boxes. It is used in iris verification in mobile phones, face recognition and detection, medical imaging, self-driving cars - detect lanes, surrounding cars, recognize signs, and then communicate with other software to make decision for optimal action, and others. Object detection can also be used in the wheat industry. A Kaggle competition called Global Wheat Detection offers the challenging task to accurately locate wheat heads in images. It provides a labeled dataset that can be used to train an object detector, and there are also images on which the trained model can be tested on. The dataset provided is called the Global Whead Head Dataset (GWHD). The GWHD is difficult to predict on given that there are different environments in which images are taken, and the wheat heads are often crossing each other making it difficult for a network to learn and predict. In this work, we investigate whether prior knowledge from the frequency domain can be used to improve the performance of an object detector on such a difficult task. Representing images in frequency domain can be done using a mathematical transform called Fourier Transform (FT), then remove certain frequencies from the images that are considered noisy or non-informative, and map back to spatial (pixel) domain. This technique will be used together with an object detector to analyse the impact that Fourier Transform has on the accuracy of the object detector that is otherwise run on normal, unfiltered images.

The FT has immense application in signal processing. For example, Shazam relies on FT[1] to remove background noise when making a prediction for a song. The transform maps functions of spatial or time domain into functions depending on frequencies of these domains[2]. When applied on image, the input is the spatial domain - formed out of pixels, and the output is the frequency domain. Having an object into its frequency domain can be used for solely analyzing what specific frequencies compose the original signal, but also filter out some of them.
Frequency filtered images can be interpreted by any object detector. To be able to predict objects in an image or video, the object detector first needs to be trained on a dataset that consists of the objects to be predicted in some environment with their coordinates. The model is then trained and knows how to predict the coordinates of new, unseen images with objects. Some datasets may be harder to learn on if, for example, they are more diverse and contain object with complex shapes, as does the GWHD, which is exactly what makes it an appropriate dataset to use frequency analysis on, because filtering unnecessary frequencies in the noisy and diverse images can create a new, cleaner dataset that may be easier interpreted by detectors. Using FT, each image is mapped to its frequency spatial domain, frequencies that are not so prominent in the wheat heads are filtered, and then the image is mapped back to the spatial domain. Since the image spatial domain is finite, Discrete Fourier Transform can be used. An efficient implementation of the Discrete Fourier Transform called Fast Fourier Transform is used to increase speed. Additional low-pass, high-pass, and band-pass filters are used to eliminate too low and too high frequencies. After eliminating certain frequencies that are far from the ones in the wheat heads, the object detector may be more efficient and capable of capturing patterns and information in the heads. The aim is to decrease the degree to which the model is confused by the background frequencies and mostly focus on analyzing the frequencies on the wheat heads.
This paper will present the results and analyses of multiple experiments that are run with a pipeline containing an object detector and a FFT implementation that is run on the wheat head images. The findings of the conducted analysis of the frequency information of the wheat dataset do not show improvement in any of the metrics that are normally used to evaluate object detectors. What is interesting is that, although in some of the filtered images, wheat heads can barely be seen by the human eye, the detector is able to recognize most of them despite the fact that there is no increase in any metric. Key contributions of the conducted experiments and analyses are:

- Remove background frequencies from the original images using Fast Fourier Transform algorithm and run an object detector on the original and filtered images. The difference in the training, validation, and test dataset accuracy of original and filtered images can then be observed and conclusions can be drawn about how and why the frequency information is beneficial when used for the GWHD and suggest further research on similar datasets.

- Suggest other methods for filtering frequencies that could improve the detector's accuracy. They require more research and are harder to implement compared to the method described in this report, but are more sophisticated and can indeed result in improvement for the GWHD and can be used for other datasets as well.

- Create a generic, usable pipeline that can perform Frequency Information analysis on an arbitrary dataset that can be interpreted by the YOLOv5 object detector. It also allows for custom Frequency analysis implementations and aims for easy integration. The aim of the pipeline is to allow researchers to easily perform frequency analysis experiments and further allow for integration of novel frequency filter methods.

Using a more sophisticated frequency filter, as suggested in the discussion, may allow for an increase in all metrics for the current dataset and be tested also on other datasets easily. The rest of the report is organized as follows: the next section will describe more in detail what object detectors are, how they work, and why and how frequency information could improve detectors' accuracy, supported by literature, section 3 will give an overview of the methods used to perform the whole experiment, section 4 will expose the experimental details and also the results of the comparisons and the output of the detector.Section 5 will describe the experiment's hyperparameters and show that the detector's performance is objectively evaluated on completely new, unseen data that has not been neither in the training nor validation process. In section 6 I put forward an alternative algorithm for filtering out frequencies, and in the last section I summarise and conclude my work.

2 Related work

Object detection. Object detectors can vary in architecture, training strategy and optimization function[3]. Some of them focus more on the accuracy, and others more on maintainability, reproducibility and speed. There are generally two types of object detectors. Two-stage object detectors, that generate regions of interest in the first stage and in the second stage - find the coordinates of the bounding boxes using regression. These models are usually fast, but rather slow. There are also one-stage object detectors - these try to directly predict the bounding box coordinates and are usually faster, but less accurate[4]. An example of a one-stage object detector is YOLO ("You only look once")[5]. There have been multiple improvements in the YOLO family, various YOLO detectors have been created and their main advantage is the speed. An example of a two-stage object detector is R-CNN. Although it tries to bypass the problem of selecting many regions of interest, it is still relatively slow. To address this problem, Fast-RCNN[6] and Faster-RCNN[7] have been created. They are faster due to the fact that they encompass convolution operations that are done only once per image[8]. Still, none of them can compare to the speed of YOLO. Another famous object detector is EfficientDet. Its ConvNet is built by Google Brain and it performs fast relative to other detectors. There is no single best object detector, each one has its own benefits and is appropriate for certain problems. The choice of a detector is highly dependent on the specific use case - in medical imaging, it is safer to use a more accurate detector since accurate predictions are what is valuable, and in a traffic observation system it may be more appropriate to use a faster detector because of the large number of cars that are constantly moving.

Fast Fourier Transform and its application in images In programming, a two-dimensional image can be represented in various ways depending on a specific use case. For example, an image can be represented in vector space and this allows for learning techniques in computer vision[9]. Transform-based representations provide a sparse representation of smooth images and this is beneficial in approximation and compression[10]. Of course, the classical way to represent an image is a 2D array of pixels, because it is convenient for storage and processing[11]. The matrix representation of an image can be used to visualize the whole image (as well as different parts of it), and feed into the machine learning models where it is also transformed into tensors. FFT can be used to represent an image in the frequency domain[12]. FFT implementations allow for very fast mapping to the frequency domain of an image. An image represented in its frequency domain can have various computational advantages. For example, novel blur invariant features that are used for object recognition are shown to be practically computed faster using the FFT algorithm[13]. Representing an image in its frequency domain also has use case in Deep Neural Networks (DNNs): a simpler and more lightweight DNN model can be created, useful information from the images can be extracted, and improvements on the results can be made. This also rises the question whether DNNs can use a combination of spatial and frequency features to achieve better performance[14]. Images in the frequency domain can also be efficiently interpreted by Convolutional Neural Networks (CNNs). Most modern object detectors use ConvNets in their architectures. They are is a type of Artificial Neural Network (ANN), the main difference being that the CNN is mainly used for pattern recognition tasks within images[15]. CNNs make use of filters to detect patterns (edges, shapes, figures). There are more filters located deeper in the network that detect complex patterns. This network is ideal for object detection since it can detect objects' shapes and patterns.

It is known that a convolution in the spatial domain in an image becomes multiplication in the frequency domain which means that the images can be processed directly in the frequency spectrum for efficiency. Although the Fast Fourier Transform method for convolution is not used very much in practice because of the memory growth of the coefficients storage[16], still an intriguing question would be: what if a CNN is designed to work specifically on the frequency domain[14]? FFT can be used to achieve computational advantages when a CNN is directly fed with data that is in the frequency domain.

Frequency domain and CNNs. An example use case of the frequency domain of the images can be filtering out certain frequencies to decrease the network overhead while at the same time improve the accuracy over a classical spatial downsampling approach. The network is directly fed with the frequency-domain information. It is demonstrated that this technique achieves 1.6% and 0.63% top-1 accuracy improvement on the ImageNet dataset using ResNet-50 and MobileNetV2, respectively. Trimming the input size in half also shows an improvement - 1.42% using ResNet-50[17]. The difference between the method mentioned and the approach of this paper is that in this paper the CNN

is not directly fed with the data in the frequency domain. The images are first transformed to their respective frequency domains using Fourier Transform, filtering then happens, and then the frequency domain is mapped back to the respective space domain and then the image is fed into the CNN in spatial domain. The inputting is not directly done, because object detectors have many preprocessing steps before feeding into the CNN, but the preprocessing happens on the 2D vector representation of the images, while the method mentioned in the paper feeds the frequency domain information in the CNN directly. Similar literature has shown a definite time and accuracy increase of the CNN when using Fast Fourier Transform-based U-Net. As expected, the image convolution costs are reduced and the overall computation costs are reduced. This approach was run on the BBBC dataset and achieved training time between 400 and 500 per step, compared to the non-FFT approach, which previously was between 600 700 ms per training step. What is interesting is that this approach also improved the accuracy of the model[18].

3 Methodology

Fast Fourier Transform[19] is an efficient algorithm that computes the discrete Fourier Transform, which is a Fourier Transform on discrete, uniformly sampled data[20]. An image, which originally is in the spatial domain, is transformed into its frequency domain, where non-informative frequencies will be removed. After that, Inverse Fast Fourier Transform is applied to map back to the spatial domain, where some frequencies have been removed. The removal of these frequencies is relative and problem-dependent. Below is explained, together with the whole problem-solution process, the methodology used to filter frequencies.

The *Fast Fourier Transform* is used to decompose the images into real and imaginary components, and the image in this representation is said to be in *spatial frequency domain*. The *Inverse Fast Fourier Transform* is used to map the spatial frequency function to the spatial pixel domain. An intuitive understanding of the Fourier transform and its inverse can be wrapping a specific composite signal around a circle with a rotating vector with a specific rotation speed. At each point in time, the vector will have a length equal to the frequency intensity in that point in time. Now define a new point which will be the center of mass of the vector graph. As the rotation speed of the vector increases, the center of mass also changes. There is a number of specific rotation speeds for which the center of mass will be fairly far from the 0 x coordinate, and these rotation speeds are the frequencies that ultimately compose the final composite frequency[21]. When thinking about circles, it is also useful to think about the complex plane, and this is where the i complex number is also present in the formula for calculating the FFT for a signal. The FFT and inverse FFT can be computed using the following equations[22]:

$$F(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{-\left(i\times 2\times\pi\left(x\frac{m}{M}+y\frac{n}{N}\right)\right)} \tag{1}$$

$$f(x,y) = \frac{1}{M.N}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} F(m,n)e^{\left(i\times 2\times\pi\left(x\frac{m}{M}+y\frac{n}{N}\right)\right)} \tag{2}$$

Here, $f(m,n)$ is a pixel with coordinates (m,n), $F(m,n)$ represents the image at position (m,n) in the frequency domain of the $M \times N$ image.
To choose which frequencies to filter out, a frequency spectrum is created that is the subtraction of the average frequency spectrum of all the images and the average frequency spectrum of the bounding boxes in all images. The *low-pass* and *high-pass* filters are then applied to the difference of the two spectrums: the high-pass filter removes high frequencies, and the low-pass filter removes the low frequencies. This combination and the two filters is called a *band-pass* filter. The obtained mask is then applied on each image to remove frequencies in all images. The YoloV5 object detector is run on the original wheat dataset and frequency filtered images and a comparison is made on the two results.

4 Experimental Setup and Results

4.1 Object detector

The object detector that is used is YoloV5 - it is a lot faster than it's predecessor YoloV4, it also produces very small weights file with a small accuracy decrease compared to YoloV4[23]. It is also suggested by the authors of Yolo to use v5 if integration time is a concern. Other object detectors, such as FasterRCNN, EfficientDet and PP-Yolo, can also be used for this experiement. The architecture of the object detector can be seen in Figure 1[24].

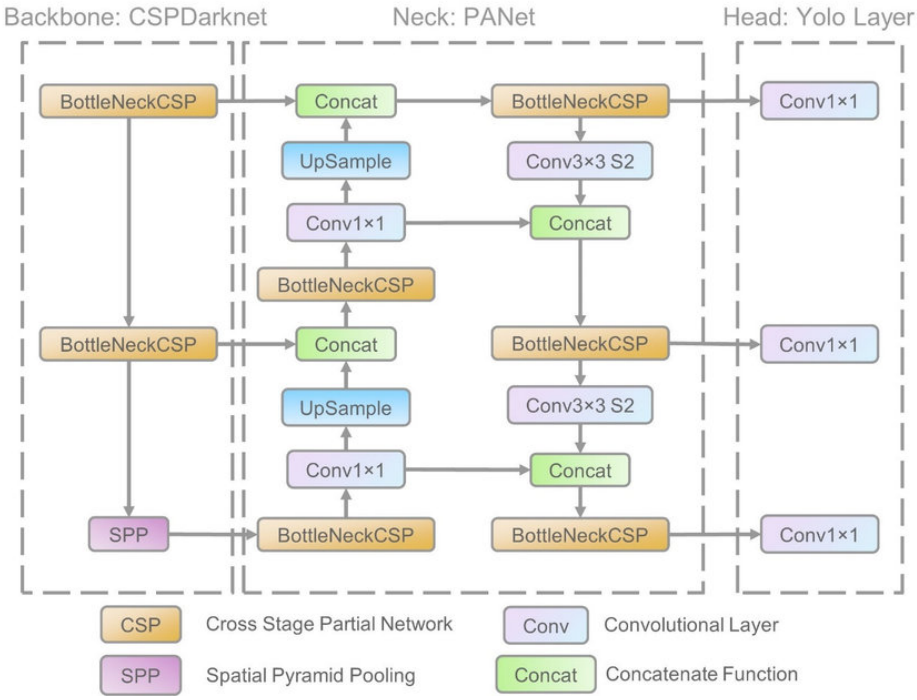


Figure 1: YoloV5 architecture[25]

4.2 Pipeline description

To make the experiment reproducible, scalable and self-contained, a configurable pipeline was created that runs all of the necessary methods and computations to achieve the final results. It can be described as follows:

1. Set up YoloV5, configure parameters, create project architecture. Run a pre-processing step on the provided Kaggle data, so it can be used by the detector[26].
2. Take a sample of the images and split it into train and test data. Train Yolov5 on the train data and test it on the test data with the learned weights on the train data and save results.
3. Use the mask filter to remove non-informative frequencies in all images
4. Repeat steps 2. and 3. on the filtered images

The pipeline can be configured to run with custom parameters: the YoloV5 can be configured to run with a specific number of training epochs, batch size and number of images. The low-pass and high-pass filters can also be customly set.

4.3 Error metrics

This subsection defines the most widely used error metrics in object detection:

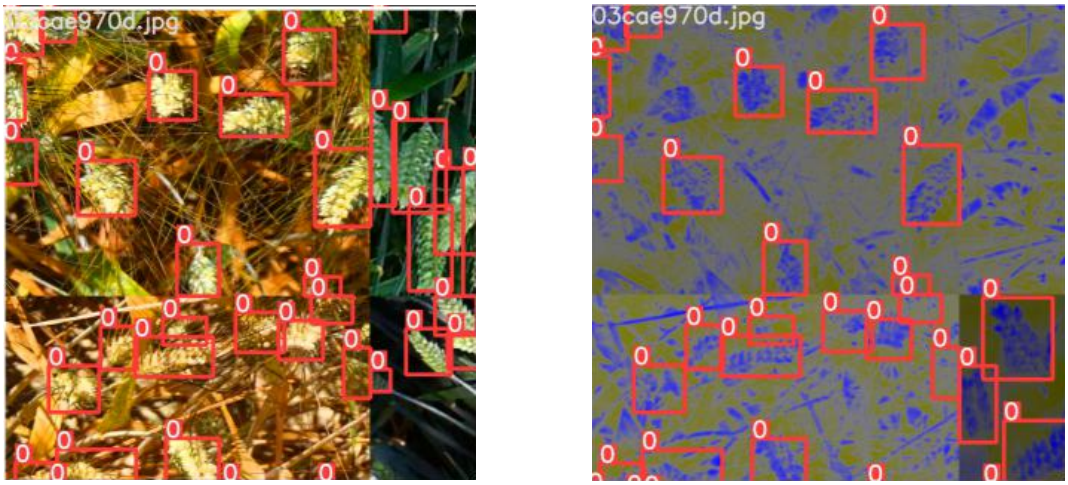
Table 1: Metric on the test data in Experiment 1 showing that non-fft method outperforms the fft method in all metrics

metrics						
	images	*labels*	*precision*	*recall*	*mAP_0.5*	*mAP_0.5 : .95*
$non - fft$	553	23929	0.943	0.916	0.944	0.56
fft	553	23929	0.939	0.893	0.928	0.539

- **IoU(A, B)** $= \dfrac{A \cap B}{A \cup B}$
- **mAP** - mean average precision, it is very often used as an evaluation metric in object detection. It is the average of the average precision and/or IoU of each class[27].
  For example, map@0.5 means the mean Average Precision at **IoU** = 0.5 is calculated for all pictures for each category, and then each category is averaged. mAP@0.5:0.95 represents the mean Average Precision at various **IoU** thresholds varying from 0.5 to 0.95 at intervals of 0.05[28].
- **Precision** $= \dfrac{tp}{tp + fp}$, where **tp** is a true positive and **fp** is a false positive.
- **Recall** $= \dfrac{tp}{tp + fn}$, where **fn** is false negative
- **Box loss (box_loss)** - a loss that measures how tight are the predicted bounding boxes around the true objects. It is usually a regression loss[29].

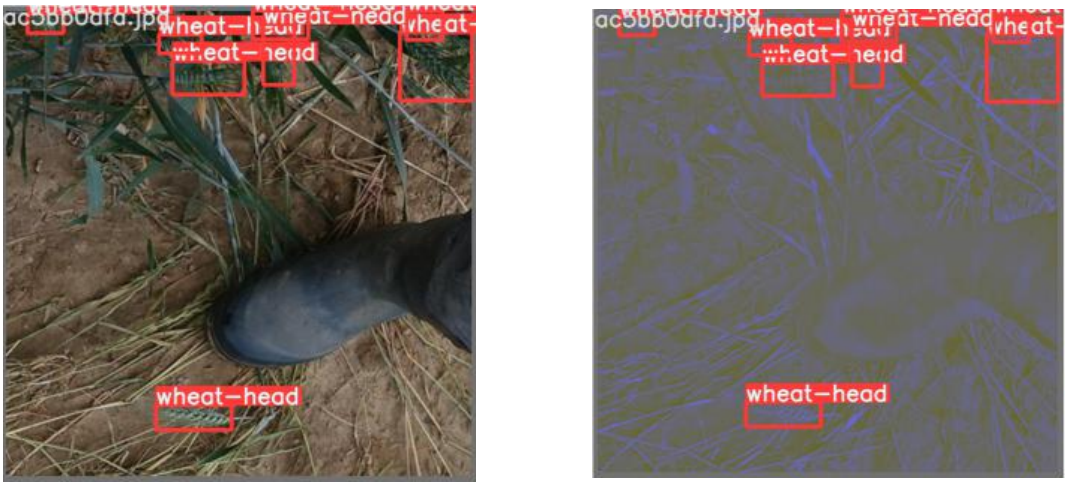
4.4 Experiment 1

Now that the error metrics and the pipeline functionality are described, this subsection presents the results of running the pipeline with the following parameters: low-pass filter: 0.2, high-pass filter: 120, YOLO training epochs: 200, number of images: 3422. Training and validation data can be seen in Figures 1 and 2. In Figure 2, subfigures (a) and (b) present a training, labeled image that the detector trains on. In Figure 3, subfigures (a) and (b) present a validation image to fine tune the detector's hyperparameters. In Figure 4 can be seen precision, map and recall metrics on the train set. Figure 5 and table 1 present validation and test results, respectively.



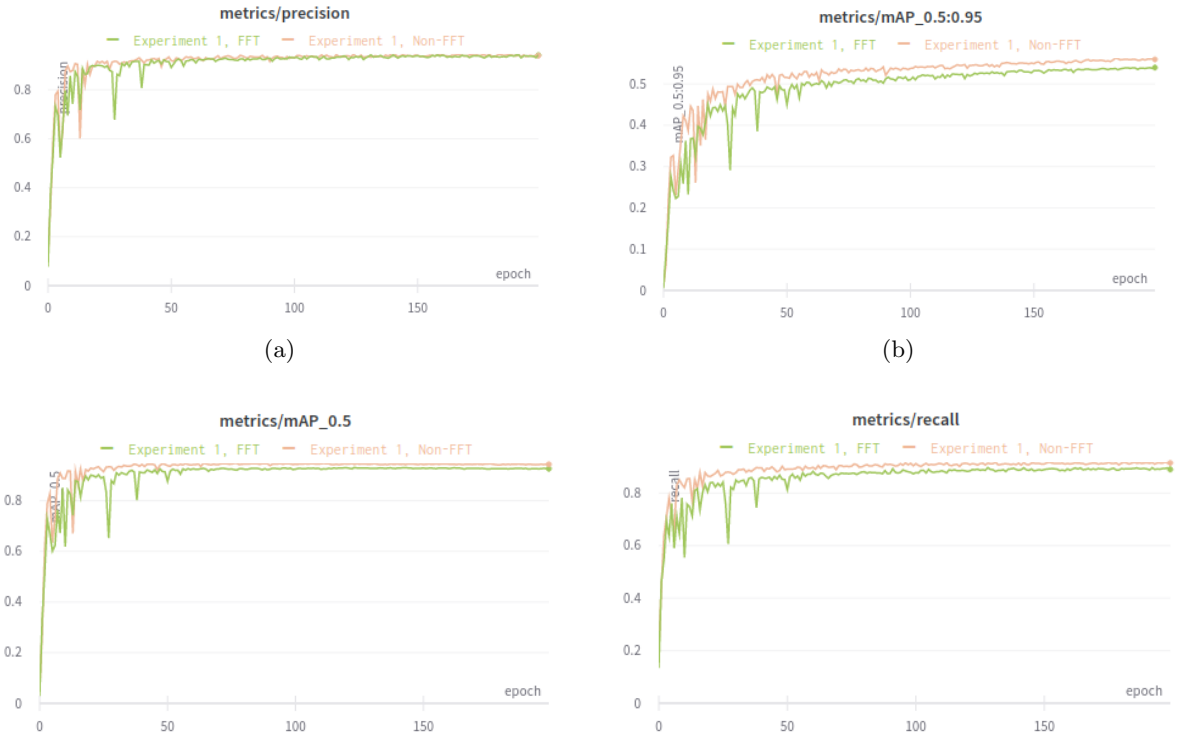
(a) (b)

Figure 2: YoloV5 labelled training image, FFT image has more sharp and prominent frequencies because of the frequency mask and the environment that the image was taken in



(a) (b)

Figure 3: YoloV5 labelled validation image in Experiment 1, FFT image has softer and not so distinctive frequencies in the wheat heads because of the frequency mask and the environment that the image was taken in



(a) (b)

(c) (d)

Figure 4: YoloV5 learning curve on training data in Experiment 1, fft and non-fft methods have similar learning curves

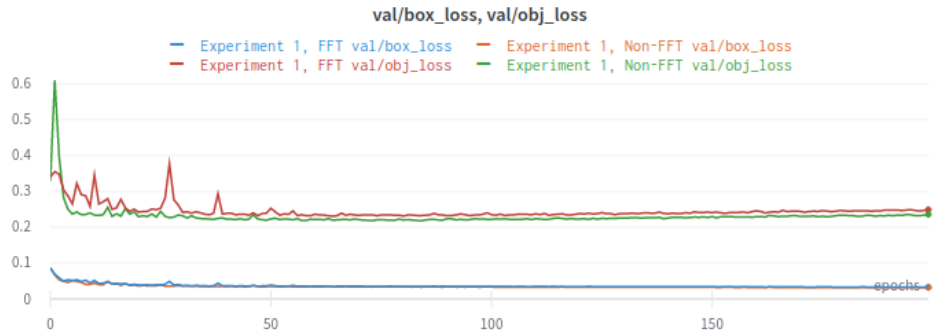


Figure 5: YoloV5 validation data performance curves are similar in Experiment 1

Table 2: Metric on the test data in Experiment 2 showing that non-FFT method again outperforms the FFT method in all metrics

metrics						
	images	*labels*	*precision*	*recall*	$mAP\_0.5$	$mAP\_0.5:.95$
$non-FFT$	507	22200	0.947	0.929	0.955	0.578
FFT	507	22200	0.943	0.91	0.939	0.559

The graphs suggest that, for this experiment and specific set of hyperparameters, the network does not perform better when run on the transformed images. The validation data which is used for hyperparameter tuning also shows no improvement in the box_loss and obj_loss metrics. Ultimately, when the trained and tuned network is run on new, unseen test data, the FFT method does not show any improvement in any of the metrics, as can be seen in Table 1. The results of the two runs show that the behaviour of the detector is similar on both FFT images and non-FFT images with a slight decrease in accuracy when FFT method is used. This suggests that YoloV5 does not learn better on that specific frequency filter and on this specific wheat head dataset. For this specific frequency filter and dataset, the object detector does not learn better when certain frequencies are filtered out.

4.5 Experiment 2

This experiment is similar to Experiment 1, but with different hyperparameters are used. The parameters that are changed for this experiment are: the low-pass filter is set to 0.15 and the high-pass filter is set to 125, also the number of images considered for the experiment is 3000. Decreasing the low-pass filter and increasing the high pass filter means that more low and more high frequencies will be allowed, that is, fewer frequencies will be filtered in total. This means that the combined band-pass filter will not be so restrictive and a different frequency spectrum will be created than the one in Experiment 1. The results of this experiment are similar to those shown in Experiment 1 and confirm the theory that the YoloV5 detector does not improve accuracy when run on frequency filtered images. Figure 6 shows the learning curves of the both approaches on the train data, and Figure 7 and Table 2 show that the using FFT does not improve any of the metrics for the validation and test data - although results are similar, there is no improvement when the frequency filter is applied on the images. This experiment shows similar results to those in Experiment 1.

5 Responsible Research

5.1 Ethics

All the presented results were obtained by directly running the code on the data. Instructions can be found below on how to run the code, so you can verify that the results are very similar to the presented ones (if using the same configuration). It is clear that the results will almost never be exactly the same - this is due to the non-deterministic nature of neural networks - after all, random weight initialization

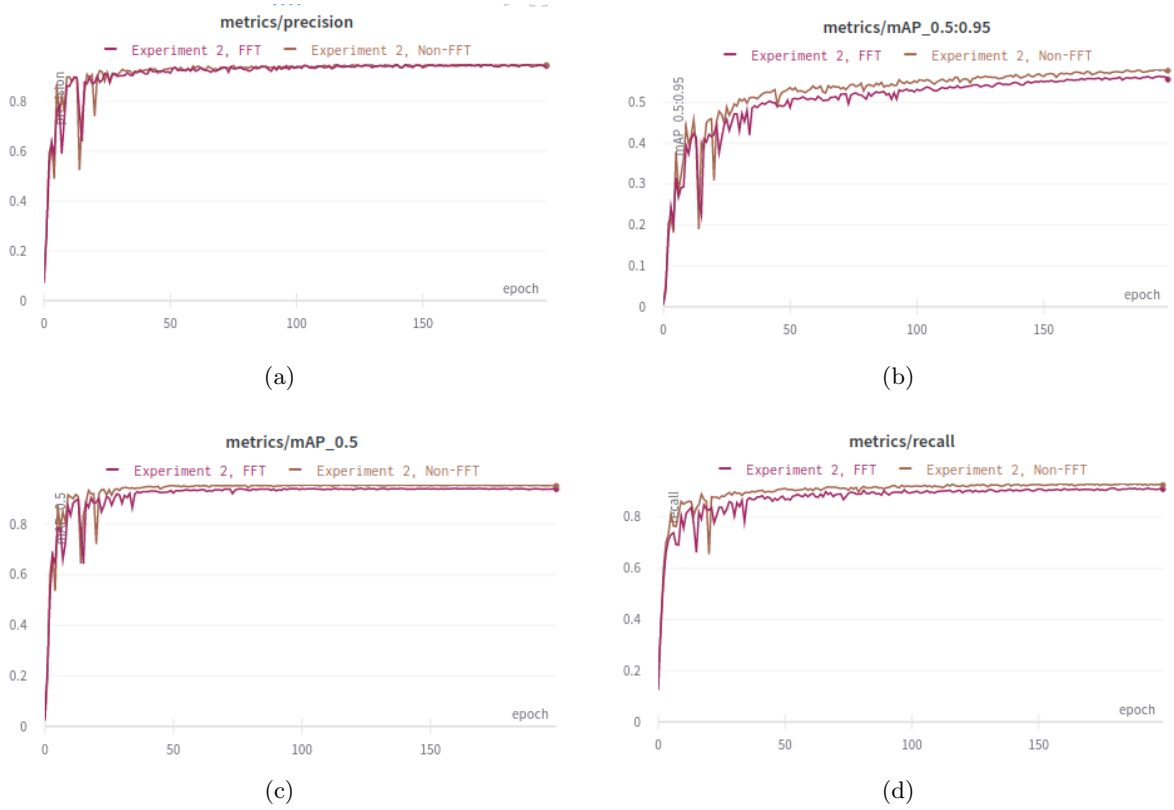


Figure 6: YoloV5 learning curve on training data in Experiment 2, FFT and non-FFT methods have similar learning curves

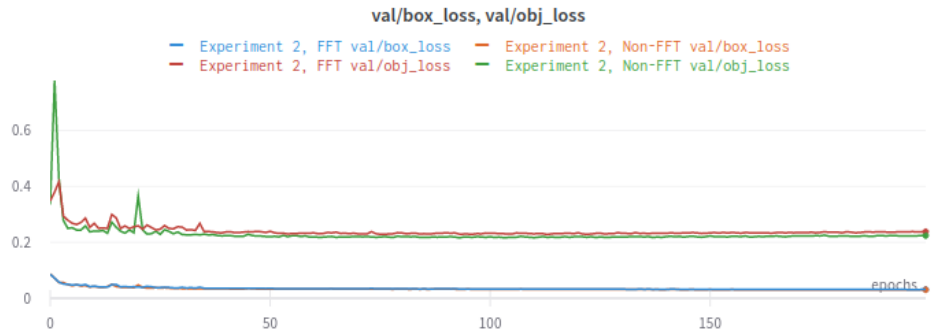


Figure 7: YoloV5 validation data performance curves are similar in Experiment 2

is created in the beginning, so there will probably almost always be a small offset of difference, but the metrics will not be very different from a run to run. Note that in the experiments, the dataset is split in three subsets - training set, validation set and test set. The training set is used to train the model, the validation set is used to observe the accuracy of the model after a specific training period and then fine-tune the hyper parameters of the model to increase the accuracy, and the test set is unseen data by the model that is purely used to evaluate the performance of the model. The wheat head images are responsibly and randomly split into all three of the mentioned subsets randomly - there is no bias in the splitting, and each time that an experiment is run results are shown for all of the three sets and analysis can be done accordingly. For the current analysis, the following split was used: 70% for the training dataset, 20% for the validation dataset and 10% for the test dataset. This is a reasonable split in this case since 10% of 3000 images is 300 test images. Considering the fact that there are usually many wheat heads in the images, 300 of them is enough for an objective evaluation of the model.

5.2 Reproducibility

All the code used to create this experiment is available on https://gitlab.ewi.tudelft.nl/cse3000/2020-2021/rp-group-50/rp-group-50-common/-/tree/yolov5_and_wheat_preprocessing . Before running the code, it is recommended to install WanDB with pip or pip3, then create an account in the https://wandb.ai/ website. Lastly, the command *wandb login* should be run in the project root directory to connect to your remote newly created WanDB account. Having set up WanDB, you can check all of the results of your experiments represented in a visual manner, but also check the logs of the program. To start the program, run *python3 main.py*. The program will start with the default values of the parameters that are described below:

Arguments

- **n_images**: number of images to sample, default value: 50
- **batch**: training batch size for YoloV5, default value: 1
- **epochs**: number of training epochs, default value: 15
- **threshold**: FFT the low-pass filter, default value: 0.2
- **exclude**: FFT high-pass filter, default value: 120

It is recommended to use higher number of images and epochs for a better accuracy. An example is:
python3 main.py –n_images 2500 –batch 16 –epochs 200 –threshold 0.2 –exclude 120
The documentation can also be found in the *main.py* method and also in the README.txt file.

6 Discussion

Object detectors have a really large use case - one of them being self-driving cars, for example. Such an example makes it clear how important it is to have a very accurate network that can predict the best possible move in each possible situation - people's lives depend on that software. This is just to underline the importance of having a decent object detector. In many cases, it is not possible to create a great detector because of the nature of the training data. The wheat head data set is such a data set - its variety makes it so hard to generalize to new data. That's why it's important to discover a way to efficiently extract the most useful frequencies in the images so that the detectors can detect where that frequency information is present and detect the object at that position.
A possible, better frequency filtering technique would be using an additional dataset containing only the objects of interest without any background. This can be achieved by either manually cutting of the objects out of images, or using instance segmentation or semantic segmentation for this task. It is important to note that both methods have their cons - manually preparing such dataset would be cumbersome and slow, objects could be missed and this solution is not scalable. Semantic segmentation deals with this problem well, but it also prone to mistakes. Having the raw objects as a separate dataset, the current (or another) FFT method can be run only on the objects of interest. This will eliminate any background information that the bounding boxes usually contain.
Further investigating frequency filtering techniques, a potential approach would be to run a gradient

descent optimization on the low-pass and high-pass filters, on one or more evaluation metrics. This method would be slow, since running the FFT algorithm and the object detector both take a lot of time. This option can be viable if using a machine with a strong GPU or Multi-GPU architecture. Most object detectors are implemented to run on the GPU.

7 Conclusions and Future Work

Deep learning techniques are being employed more and more since the advances in computing power have made it possible to run complex algorithms on big data. Object detectors, as such, can be very useful in the industry because they allow for automation, scalability and mainainability of a system. Complex datasets, such as the wheat head dataset, make it harder to achieve a high accuracy and thus new methods should be considered that facilitate the object detectors' interpretation of the complex images. Using frequency information is known to have benefits in computation and accuracy in image recognition, and it gives a foundation of the experiments in this report. The Fourier Transform allows to represent images in their frequency domain, where certain frequencies that are considered as noisy can be removed, then images are mapped back to the spatial domain using the Inverse Fourier Transform, so that an object detector can only consider the most prominent and important frequencies that are in the wheat heads. For the experiments conducted in this work, the object detector YoloV5 does not show improvement in accuracy when run on frequency filtered images of the Global Wheat Head Dataset neither in the training, validation nor test datasets. This result may be due to the specific frequency filter method or the GWHD. A dedicated customizable pipeline in Python was created that allows for easy integration of custom frequency filter implementations and datasets. It can be used for further research on this topic.

References

[1] G. Palmirotta. *A study of Shazam's Audio Recognition*. Dec. 2016.

[2] *Fourier transform*. June 2021. URL: https://en.wikipedia.org/wiki/Fourier_transform.

[3] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu. "Object Detection with Deep Learning: A Review". In: *CoRR* abs/1807.05511 (2018). arXiv: 1807.05511. URL: http://arxiv.org/abs/1807.05511.

[4] P. Soviany and R. T. Ionescu. "Optimizing the Trade-off between Single-Stage and Two-Stage Object Detectors using Image Difficulty Prediction". In: *CoRR* abs/1803.08707 (2018). arXiv: 1803.08707. URL: http://arxiv.org/abs/1803.08707.

[5] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *CoRR* abs/1506.02640 (2015). arXiv: 1506.02640. URL: http://arxiv.org/abs/1506.02640.

[6] R. Girshick. "Fast R-CNN". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1440–1448.

[7] S. Ren, K. He, R. B. Girshick, and J. Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[8] R. Gandhi. July 2018. URL: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

[9] D. Beymer and T. Poggio. "Image Representations for Visual Learning". In: *Science (New York, N.Y.)* 272 (July 1996), pp. 1905–9.

[10] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P. Dragotti. "Image representation and compression using directionlets - art. no. 67010N". In: (Oct. 2007).

[11] R. Navarro, A. Tabernero, G. Cristobal, E. Peter, and W. Hawkes. "Image Representation With Gabor Wavelets And Its". In: (June 1999).

[12] N. M. Singh, J. E. Iglesias, E. Adalsteinsson, A. V. Dalca, and P. Golland. "Joint Frequency- and Image-Space Learning for Fourier Imaging". In: *CoRR* abs/2007.01441 (2020). arXiv: 2007.01441. URL: https://arxiv.org/abs/2007.01441.

[13] V. Ojansivu and J. Heikkil. "Object Recognition Using Frequency Domain Blur Invariant Features". In: ().

[14] J. A. Stuchi, L. Boccato, and R. Attux. "Frequency learning for image classification". In: *CoRR* abs/2006.15476 (2020). arXiv: 2006.15476. URL: https://arxiv.org/abs/2006.15476.

[15] K. O'Shea and R. Nash. "An Introduction to Convolutional Neural Networks". In: *CoRR* abs/1511.08458 (2015). arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

[16] M. Mody, C. Ghone, M. Mathew, and J. Jones. "Efficient frequency domain CNN algorithm". In: *2017 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*. 2017, pp. 22–25.

[17] K. Xu, M. Qin, F. Sun, Y. Wang, Y.-K. Chen, and F. Ren. "Learning in the Frequency Domain". In: *CoRR* abs/2002.12416 (2020). arXiv: 2002.12416. URL: https://arxiv.org/abs/2002.12416.

[18] V. Nair, M. Chatterjee, N. Tavakoli, A. S. Namin, and C. Snoeyink. "Fast Fourier Transformation for Optimizing Convolutional Neural Networks in Object Recognition". In: *CoRR* abs/2010.04257 (2020). arXiv: 2010.04257. URL: https://arxiv.org/abs/2010.04257.

[19] *Fast Fourier transform*. May 2021. URL: https://en.wikipedia.org/wiki/Fast_Fourier_transform.

[20] *Discrete Fourier transform*. May 2021. URL: https://en.wikipedia.org/wiki/Discrete_Fourier_transform.

[21] 3Blue1Brown. *But what is the Fourier Transform? A visual introduction.* Youtube. 2018. URL: https://www.youtube.com/watch?v=spUNpyF58BY&t=339s.

[22] V. Nair, M. Chatterjee, N. Tavakoli, A. S. Namin, and C. Snoeyink. "Fast Fourier Transformation for Optimizing Convolutional Neural Networks in Object Recognition". In: *CoRR* abs/2010.04257 (2020). arXiv: 2010.04257. URL: https://arxiv.org/abs/2010.04257.

[23] C. Supeshala. *YOLO v4 or YOLO v5 or PP-YOLO?* Aug. 2020. URL: https://towardsdatascience.com/yolo-v4-or-yolo-v5-or-pp-yolo-dad8e40f7109.

[24] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu. "A Forest Fire Detection System Based on Ensemble Learning". In: *Forests* 12 (Feb. 2021), p. 217.

[25] Y. Ding, Z. Li, and D. Yastremsky. *Real-time Face Mask Detection in Video Data*. May 2021.

[26] V. Balakrishna Kumar. *Yolo-V5 Object Detection on a Custom Dataset*. URL: https://deepscopy.com/Yolo-V5_Object_Detection_on_a_Custom_Dataset?fbclid=IwAR1QtH272v2ec77ptRnl9Mp7qaT97mEAVOWORY

[27] T. C. Arlen. *Understanding the mAP Evaluation Metric for Object Detection*. Mar. 2018. URL: https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3.

[28] *The meaning of mAP@0.5 and mAP@0.5:0.95, YOLO - Programmer Sought*. URL: https://www.programmersought.com/article/84866344358/.

[29] R.-R. 59377. *Obj loss*. URL: https://stackoverflow.com/questions/54977311/what-is-loss-cls-and-loss-bbox-and-why-are-they-always-zero-in-training.