GLOBAL WHEAT DETECTION

ABSTRACT

In the past, wheat breeding progress was key for the food security of emerging countries. Among all traits, wheat head number per unit ground area is a major yield component and is still manually evaluated in breeding trials, which is labor-intensive and leads to measurement errors around 10%. Thus developing sensing-based methods to increase the throughput and accuracy of wheat heads counting in the field is highly desired to help breeders manipulating the balance between yield components.

Given outdoor images of wheat plants, our task was to identify bounding boxes around wheat heads. We started off with data exploration, calculating the average number of wheat heads per image, and checking for the presence of weeds and irrelevant plants. We then went on to train our Faster-RCNN Pytorch model with a ResNet-152 backbone. We achieved a final mean average precision at different IoU (Intersection over Union) thresholds as 0.587 on the test set.

TEAM MEMBERS

- Aakanksha Nallabothula Surya
- Akanksha .

PROJECT OBJECTIVE

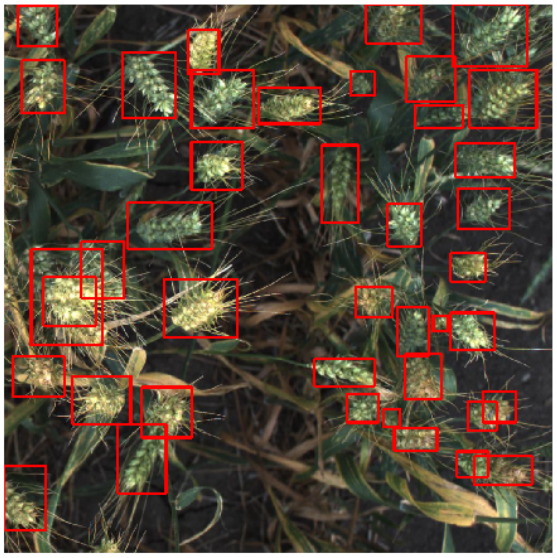
To predict bounding boxes around each wheat head in images that have them

Kaggle competition link: https://www.kaggle.com/c/global-wheat-detection

DATA

Training data consists of 3422 image files, totaling around 613 Mb.

Train.csv has columns such as image_id, width and height of the image, and bounding box dimension [xmin, ymin, width, height]



Sample Image from the training set consisting of bounding boxes around wheat heads

Accurate wheat head detection in outdoor field images can be visually challenging. There is often overlap of dense wheat plants, and the wind can blur the photographs. Both make it difficult to identify single heads. Additionally, appearances vary due to maturity, color, genotype, and head orientation. Finally, because wheat is grown worldwide, different varieties, planting densities, patterns, and field conditions must be considered. To better gauge the performance for unseen genotypes, environments, and observational conditions, the training dataset covers multiple regions. The training set consists of more than 3,000 images from Europe (France, UK, Switzerland) and North America (Canada). The test data includes about 1,000 images from Australia, Japan, and China.

EVALUATION

The evaluation criterion is the mean average precision at the different intersection over union (IoU) thresholds. The metric sweeps over a range of IoU thresholds, at each point calculating an average precision value. The threshold values range from 0.5 to 0.75 with a step size of 0.05. At each threshold value (t), a precision value is calculated based on the number of true positives (TP), false negatives (FN), and false positives (FP) resulting from comparing the predicted object to all ground-truth objects. The average precision of a single image is calculated as the mean of the above precision values at each IoU threshold.

REFERENCE CODE AND CHANGES MADE

REFERENCE CODE: https://www.kaggle.com/pestipeti/pytorch-starter-fasterrcnn-train

OUR MODIFICATIONS:

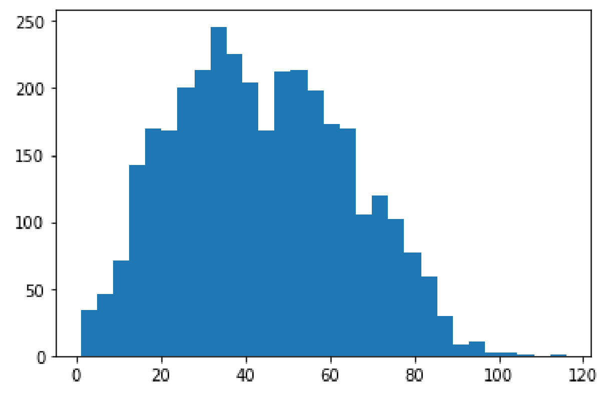
- Included more data augmentations
- Tried ResNet-34, ResNet-50, and ResNet-152 backbone
- Performed Exploratory Data Analysis

MACHINE LEARNING METHODS AND EXPERIMENTAL RESULTS

DATA EXPLORATION:

The images have been collected from various sources. We inspected the distribution of images by source and found that more than 50% of the images belong to ethz_1 and arvalis_1.

Images can contain 0 or more wheat heads. So, we looked at the distribution of the number of wheat heads per image and an average of 43 wheat heads are present per image.



Distribution of number of wheat heads per image

On exploring the images we drew out the following observations:

- There are plenty of overlapping bounding boxes.
- All images have been taken vertically
- The wheat heads have different colors based on the source.

- Wheat heads themselves are seen from very different angles of view relevant to the observer

DATA AUGMENTATION:

To ensure our model does not overfit to the training set and generalizes well, we performed the following augmentations while training:

- Randomly changing the hue
- Vertical and horizontal flipping
- Random crop
- Converting image to greyscale
- Randomly changing brightness and contrast of the image

MODEL:

For our model, we've used a pre-trained Faster R-CNN model in Pytorch with a ResNet-152 architecture.

The architecture of Faster R-CNN is complex because it has several moving parts. It all starts with an image, from which we want to obtain:

- A list of bounding boxes.
- A label assigned to each bounding box.
- A probability for each label and bounding box.

The input images are represented as Height×Width×Depth tensors (multidimensional arrays), which are passed through a pre-trained CNN up until an intermediate layer, ending up with a convolutional feature map. We use this as a feature extractor for the next part.

Next, we have what is called a Region Proposal Network (RPN). Using the features that the CNN computed, it is used to find up to a predefined number of regions (bounding boxes), which may contain objects. The hardest issue with using Deep Learning (DL) for object detection is generating a variable-length list of bounding boxes.

The variable-length problem is solved in the RPN by using anchors: fixed-sized reference bounding boxes which are placed uniformly throughout the original image. Instead of having to detect where objects are, we model the problem into two parts. For every anchor, we ask:

1. Does this anchor contain a relevant object?
2. How would we adjust this anchor to better fit the relevant object?

Using the features extracted by the CNN and the bounding boxes with relevant objects, we apply Region of Interest (RoI) Pooling and extract those features which would correspond to the relevant objects into a new tensor. Finally, comes the R-CNN module, which uses that information to:

- Classify the content in the bounding box (or discard it, using “background” as a label).
- Adjust the bounding box coordinates (so it better fits the object).

RESULTS:

Training results

Backbone	**Mean Average Precision at IoU**
RESNET-34	0.680
RESNET-50	0.701
RESNET-152	0.711

Test results

Backbone	**Mean Average Precision at IoU**
RESNET-34	0.515
RESNET-50	0.587
RESNET-152	0.672

We achieved a final mean average precision at different IoU (Intersection over Union) thresholds as 0.672 on the test set.

CONCLUSIONS AND LESSONS LEARNED:

Although we did not get to spend as much as time as we'd ideally need for such a huge problem, we have quite a few learnings:

- Dealing with Images consisting of multiple bounding boxes.
- Using Kaggle notebooks, making submissions to notebook only competitions.
- There are lots of useful public notebooks on Kaggle.

POINTERS TO OUR CODE:

Exploratory Data Analysis:
https://github.com/aakanksha-ns/wheat-detection/blob/master/Wheat_Detection_EDA.ipynb

Faster RCNN attempt 1:
https://github.com/aakanksha-ns/wheat-detection/blob/master/FasterRCNN_attempt1.ipynb

Final Training notebook:
https://github.com/aakanksha-ns/wheat-detection/blob/master/faster_r-CNN_Train.ipynb

Final Inference notebook:
https://github.com/aakanksha-ns/wheat-detection/blob/master/faster_r-CNN_Inference.ipynb

TEAM RESPONSIBILITIES

- **Aakanksha Nallabothula Surya** : Data processing, Exploratory data analysis, Modeling, Project report
- **Akanksha** . : Data processing, Exploratory data analysis, Modeling, Project report

REFERENCES

https://www.kaggle.com/c/global-wheat-detection

https://arxiv.org/pdf/2005.02162.pdf

https://towardsdatascience.com/review-faster-r-cnn-object-detection-f5685cb30202