IT3915 Master Preparatory Project - Image Processing For Better Sheep Retrieval Using Drone Images

Written by Thomas Bjerke and Trym Grande

14.12.22

Introduction

We chose this problem as our Masters preparatory project for a few different reasons. We have both had experience with machine learning algorithms, and object detection. It seemed like an interesting topic with a good combination of different areas of both software engineering and machine learning. Also, Trym had some experience with drone technology that might turn out to be helpful.

This report documents what has been done throughout the project, as well as which choices have been made along the way. We have been working mostly separately, and have therefore decided to split the report into two parts, describing each of our contributions to the project. Specifically, Trym has been working on project structure and architecture, as well as image augmentation. Thomas has been working on hyperparameter tuning and image clipping.

We would like to thank Svein-Olaf Hvasshovd at NTNU for being a supervisor throughout the project. Here is the original problem description (Hvasshovd), written in norwegian:
Gjenfinning av sau ved hjelp av drone
Gjenfinning av de siste sauene på høsten er ofte en av en sauebondes større mareritt. Han kan ofte legge ned flere hundre timer i felten uten å finne de. Dette er en oppgave som han meget gjerne kan tenke seg teknologisk støtte til å utføre.
Vi foreslår anvendelse av drone utstyrt med normalt kamera og infrarødt kamera til denne typen søkejobb. Dronen vil fly et veldefinert søkemønster. Kameraene tar bilder som så analyseres på dronen for om det befinner seg sau i bildet. Man vil se sau dels basert på farge og dels basert på at de er vesentlig varmere enn terrenget.
Vi ønsker her en gruppe på to eller tre studenter. Den ene studenten vil arbeide med utviklingen av bildeanalysen. Den andre vil utføre sammenstilling av de to typer av bilder der det i minst ett av de er detektert mulig sau. Den tredje studenten vil utvikle presentasjonssystemet overfor bønder som viser kartmessig hvor i terrenget sau er funnet.
Vi har her behov for to studenter som har kunnskap i bildeanalyse og bilde samstilling. Den tredje studenten har generell informasjonsteknologi-bakgrunn.

We have focused on the image processing part of the problem. This was mentioned by the supervisor as an important aspect of the problem. We therefore agreed to take the course - TDT4195 - Visual Computing Fundamentals (NTNU) as a supplementation. In addition, we interviewed a well experienced farmer from Oppdal, where we got insights into some important things to think about when developing such a system. A written report in Norwegian from this is attached.

Content

2 Project structure/architecture and image augmentation

Trym was mainly responsible for this part of the project. This section involves what has been done in order to get the project set up, as well as making the different components interact with each other. In addition to this, it includes the method used for applying different augmentation transformations to improve dataset diversity.

2.1 Research and project setup

I started by thinking about what we would solve, how to solve it, and potential problems surrounding this. Inference in real time on the drone vs. inference from a ground station. Running from the drone in real time would require an external attached module. Both running in real time and on the ground might be relevant. The state of the art machine learning algorithm for object detection was YOLO (“YOLO: Real-Time Object Detection”), with the most recent version being YOLOv7 (Kin-Yiu), which had two sub-versions available. “YOLOv7-tiny” is a compressed version, and would work well for real time inference on a small module, or for running on a laptop on the ground after landing. “YOLOv7” (non-tiny) could work well for running on a computer on the ground or even a powerful external server. This depends on the performance of the external module and on the ground computer. Since the supervisor mentioned a laptop likely being the most viable option, I opted for the YOLOv7-tiny version to begin with. I read through previous master theses while reflecting over the types of problems and potential solutions.

I then proceeded to install the latest version of the state of the art object detection algorithm - YOLOv7. I set this up as a “Git submodule” (“IT3915-master-preparatory-project-old”), since this allows for an updated version of the algorithm while also being able to store other files in parallel. I later decided to discontinue this in favor of using a forked git repository from the official YOLOv7 repository (Kin-Yiu), since submodules can be complicated to work with compared to a fork. However, this did not allow other files/projects to be stored in parallel, which would have to be stored in a second repository.

I tried uploading parts of a dataset to the repository, but this failed because of large file sizes. I therefore looked into alternative image storing or downloading methods, but did not find any immediate solutions. I therefore decided to make a minimized sample of the dataset in the meanwhile that was small enough to be pushed to the repository. I also set up configuration files required for training, and divided this sample into training and testing splits.

I successfully trained and tested the model with a dataset sample, and then started working with “Roboflow” using an initial dataset (“Sheep v4 (all images) Dataset”), consisting of around 2 thousand images, which was imported into a new private roboflow project I created (“IT3915masterpreparatoryproject”). Thomas later merged another dataset (“Sheep detection 2 Image Dataset”) with around 4 thousand images into a new dataset (“Merged sheep dataset Image Dataset”), consisting of around 6 thousand images in total. This dataset was uploaded and used later, towards the end of the project.

It was beneficial to use a jupyter notebook template ("Training YOLOv7 on Custom Data - Colaboratory"), found in an article ("How to Train YOLOv7 on a Custom Dataset") for performing most of the operations needed to get started. This included downloading a dataset from Roboflow, then performing training and inference on it, and exporting the result. This was able to run on my personal computer, but should be implemented on a GPU cluster for faster runtimes. Thomas was already working on getting access to this, so I had to wait until then. The source code was now established and I could begin working on customizing and adapting it more towards the project.

I made a new Github repository ("IT3915-master-preparatory-project") to replace the old forked one, containing only the jupyter notebook script. This solution was the best of both worlds since it can automatically download everything it needs on runtime, while being easy to maintain, and will always download the latest data - both in terms of the YOLOv7 algorithm and the Roboflow dataset. This also means that a second repository is not needed anymore.

2.2 Augmentation using personal computer

I did some research into the best image augmentation tool, when I found the "Albumentations" library ("Albumentations Documentation") - the same one used in a previous master's thesis (Johannessen). I started implementing this into the code by using a Github repository (Persson) and the official documentation for the Albumentation library ("Albumentations Documentation").

A problem occurred with augmentations where certain transformations that affect the shape of the image would require their bounding boxes to be transformed as well. This meant that the labels representing bounding boxes had to be processed and sent to the Albumentations library in a specific way. I was eventually able to get the label transformations to work by following the documentation ("Bounding boxes augmentation for object detection").

After this, I began experimenting by running tests both with and without augmentations to measure the effect. I measured the mean average precision at 50% recall (map@0.5), and ran training for 20 epochs to save time. Without augmentations, I measured a map@0.5 of 0.08, and 0.1 with image augmentations - a slight improvement. Something I later realized was that these measurements did not take into account that the dataset with augmentations was larger, and therefore effectively got more training since the number of epochs stayed the same. In addition, the results were from the training split of the dataset. This more or less invalidated the results, and had to be taken into account when doing further tests in the future. From this point on, I started using our newest dataset ("Merged sheep dataset Image Dataset"). I also set up the repository in Google Collab, which looked like a good option for running and also live code collaboration. This ended up not being used.

2.3 Attempting to utilize GPU Cluster

After Thomas eventually received access to the Idun GPU Cluster, I followed the instructions for getting started on Idun (“Getting started on Idun – High Performance Computing Group”) by connecting to a virtual machine through SSH. From there, I set up the project, and ran the code remotely. I also helped Thomas connect and set up everything the same way. One problem we were now facing was that the training phase failed with a GPU memory allocation error stating that it was out of VRAM. This did not make sense, however, since the GPU being utilized had plenty of available memory. We tried debugging this, but could not get it working.

Thomas found out that it worked with a very low image resolution. This seemed to solve the memory issue, but it still took an unreasonable amount of time to perform the training. I reverted back to using my own computer since the runtime seemed to be better than that of the Virtual Machine being used in the GPU Cluster. This worked better, but also took a long time. I therefore sent out an email regarding access to more and/or more powerful GPUs since the ones we previously had access to seemed to not be enough. I continued working on my personal computer in the meantime, and also set up an SSH server on it to allow Thomas to work remotely using my computer.

As the supervisor requested, I also messaged another student, Sindre Langaard, regarding whether they had had similar problems with the GPU cluster, which they had not. Also at the request of the supervisor, we contacted the GPU cluster personnel at their office, which got us instructions for properly running code on the GPUs. They mentioned specific instructions that were listed on the website that needed to be done in order to actually queue up a “job” on a compute node. This was a critical mistake that we had overlooked, as we had been trying to run the training program directly on the login node instead. This was confusing since this node did have 2 GPUs that were listed in the hardware section of the GPU cluster (“Hardware – High Performance Computing Group”), being 2 x Tesla P100 GPUs, but these were not intended for use. This was obviously what had caused the weird memory issues, and was a turning point considering how much time was lost due to this problem.

2.4 Augmentation using GPU Cluster

After this breakthrough of getting proper access to the GPU cluster, I disregarded some of the recent work on my personal machine, and started on a script for running a simple “job” on a compute node from the cluster. I tried out the different GPUs that were available for executing jobs, and spent some time reading the documentation on this, as well as writing new scripts. I learned that there were some occupation problems with different GPUs at different times of the day, which meant that I had to be flexible with the type of GPU used, and when they were used. I opted mostly for multiple of the “smaller” V100, since there were many of them, and with a reasonable amount of VRAM. Eventually I got everything integrated and automated. There was a problem with running augmentations on many images. I was only able to set the number of augmented images to 3250 images - around half the size of the dataset, or else the job would kill the execution of the python script for some reason.

In the end, a list of augmentation transformations are listed in augmentation_transformations.py, and a job for each of them can be queued up. Each of these jobs would run in parallel, with its own set of files and directories, allowing for completely separated datasets that can be augmented and trained on without affecting each other. The results would then be copied and merged into a common folder labeled with the job and an augmentation ID that represents a specific set of augmentations from the mentioned list. These are also printed in the log file for each job.

3 Hyperparameter tuning and image clipping

Thomas was mainly responsible for this part. It involves the different methods I tried for hyperparameter tuning and image clipping.

3.1 Hyperparameter tuning

Hyperparameters are parameters that the model uses during training. YoloV7 uses about 30 of these parameters, and they need to be tuned for optimal performance. There are several different ways to do this, and I started by implementing a grid search.

A grid search trains the model multiple times with different hyperparameters each time, and compares the performance to find the best combination of hyperparameter values. The problem with this approach was that it was intractable due to the high dimensional search space. It took too long to run, so I decided to look for other approaches instead.

What I found as the best solution was hyperparameter evolution. This uses a genetic algorithm for optimization. This means that it uses concepts inspired by the process of natural selection, and the parameters are tuned by using mutations, selection, and crossover.

3.2 Image clipping

To be able to increase the resolution during training, we decided to clip the images into 9 parts. This implies both splitting the images with overlap so that no sheep are cut in half, and also resizing the annotation boxes.

4 Results

4.1 Source code

The source code (“IT3915-master-preparatory-project”) has been uploaded and maintained on Github throughout the duration of the project. This provides all the files necessary to reproduce the results. In addition, the README file provides instructions on how to install and run the project.

4.2 Achievements

- Obtained and implemented a dataset with 6142 aerial images of sheep (“Merged sheep dataset Image Dataset”).
- Implemented YOLOv7 (Kin-Yiu) - a state of the art machine learning algorithm for object detection on the dataset.
- Implemented image augmentation using the Python library Albumentations (“Albumentations Documentation”). This implementation uses the provided labels from the dataset, which are transformed according to each corresponding image transformation. This means that the bounding boxes are preserved after an image transformation in the case of e.g. a flip or rotation of the image.
- Hyperparameter evolution with genetic algorithm
- Image clipping so that we can increase the image resolution.
- A working pipeline that downloads the newest version of the dataset and the machine learning algorithm, and then applies custom image augmentation and training.
- Some results with testing a few sets of specific augmentation transformations. These are listed according to the sets of transformations from augmentation_transformations.py, and can be looked up there. The test numbers are extracted from the training results, but should be tested on the test split of the dataset in the future instead. The resulting numbers are from the output found in runs/run<job_id>-<augmentation_id>/train/results.txt in the 11th column, describing mean average precision at 50% recall (map@0.5), and in the 100th and final row, meaning 100 epochs of training:
  0: 0.942, 1: 0.9542, 2: 0.9477, 3: 0.912, 4: 0.9029, 5: 0.952, 6: 0.9464,
  7: 0.9466, 8: 0.9489, 9: 0.9506, 10: 0.9459, 11: 0.9419.
  From this, we can, among other things, see that flipping the images has the best result so far.

4.3 Limitations

- Late access to the Idun GPU Cluster meant that limited time could be spent actually performing experimentation.
- Limited access to the compute nodes in the middle of the day, likely due to high traffic, meant that training had to be limited to other time periods of the day, sometimes causing delayed results.
- Inconsistencies when running jobs. Jobs have randomly failed sometimes. This was solved by simply running them again, but has led to some time loss.
- The number of training epochs was limited to 100 in order to save time.
- The test results are from the training split of the dataset. This should be tested on the test split instead to avoid the problem of overfitting. The test results are therefore considered technically inconclusive as of now.
- The number of augmented images was limited to around 3 thousand - half the size of the dataset, or else the job would kill the execution of the python script for some reason.

5 Further work

- Test more augmentation transformations. This can be implemented according to the Albumentations API (“Full API Reference”), as long as the transformation supports bounding boxes. This can be checked in the diagram within the Albumentations API Reference (“Spatial-level transforms”).
- Test augmentations using test split instead of the training split by using the test.py script documented in the YOLOv7 repository (“YOLO: Real-Time Object Detection”).
- Perform tests with more than 100 training epochs, e.g. closer to 300, for higher accuracy.
- Implementing a regression model for adjusting augmentation transformation parameters, allowing for a highly optimized augmentation transformation could be viable.
- Improve the overall pipeline flow and work out any bugs.

Intervju med sauebonde Terje

- Vi ble forklart av veileder at sauesanking består av tre faser; først der man henter flokken (mesteparten), deretter mindre flokker, og til slutt utstikkere. Omtrent hvor mange slike utstikkere pleier det å være i din saueflokk?

I flokken til Terje på rundt 400 sauer, er det vanligvis rundt 10 stk igjen i siste fase, og de tar ca. 7-10 dager å samle inn. Terje skryter av kollegene sine mtp hvor nøye de er med å prøve å hente inn alle dyrene.

- Hvor stort er et typisk beiteområde?

400 km^2 i sunndal, men dette varierer veldig. Sau i Norge har generelt veldig store områder å bevege seg på, og rundt 70% av Norge er beiteområde.

- Hvor mye beveger sauen seg? Hvor lang tid tenker du at det kan gå mellom at sauen blir observert til bonden får beskjed om hvor den er?

De som har råd bruker i dag ofte chips og satellittbilder. Da går det 3 timer mellom bildene, som er akseptabelt. Sauen kan ha tempo som mennesker og bevege seg like fort over f.eks. 4-5 timer dersom den først har bestemt seg for det. Sauen flytter seg fra gjerne fra dag til dag. Spesielt i september-oktober beveger sauen seg mye for å lete etter beite. Terje forklarer at dronesystemet vil ha stor nytte likevel om det kan gå litt tid mellom sauen blir observert og til bonden får sauens posisjon.

- Er det sånn at når man går ut for å finne utstikkere og man finner en sau, så tar man med seg den sauen videre for å finne flere, eller er det mer vanlig å gå tilbake med den sauen først og så dra ut og lete på nytt?

Det spørs om man er høyt oppe på fjellet eller ikke. Man tar gjerne med flere istedenfor å vente til neste dag.

- Vi har tenkt litt på å legge inn funksjonalitet for å få opp optimale veibane for å få hentet alle sauene. Problemet er at hvis det er mye krevende terreng, så kan man som oftest ikke kan følge en rett linje fra ett punkt på kartet til neste punkt, noe som gjør det vanskelig å finne optimal veibane. Tror du likevel at det kan være vits i visse tilfeller hvor det for eksempel er flatt og lite krevende terreng å få opp optimal veibane som i hovedsak er basert på avstand og ikke terrenget?

Bonden er ofte godt kjent med terrenget, så det er nok ikke nødvendig. I tillegg ville det vært vanskelig å ta med alle signifikante variabler om terrenget i en sånn beregning.

- Hva om en bonde skal bruke drone til å finne utstikkerne sine, men ender opp med å finne andre sine sauer i stedet?

Dette er ikke et problem, ettersom bondelagene pleier å gjøre det sånn at visse personer har ansvar for visse områder. Så hvis noen finner andre sine sauer, så tar de likevel sauene med seg. Terje sier at det å finne andre sine sauer bare er gøy - da bidrar man til fellesskapet og det skaper litt kameraderi blant bøndene.

- Kunne du tenkt deg å delta under testingen?

Terje kunne tenkt seg å bidra til testing, men har kanskje ikke så mange sauer ute. Blir mest aktuelt om høsten (september/oktober). Han forklarer at han har et fint område på Nerskogen for testing. Der er det nokså flatt, og en del skog. Da får man blant annet testet om det infrarøde kameraet klarer å fange opp sauer i skogen.

- Forslag til funksjonalitet?

Infrarødt fungerer bra i skog. Noen på Sunndalsøra har kjøpt infrarød kikkert til over 50 000 kr. Terje har blitt fortalt at den fungerer svært bra.

Nofence - Et norsk selskap som utvikler halsbånd som gir strøm dersom sauen går utenfor bestemt geografisk område - problemet er prisen. Kontaktperson: Hovde. Terje har tenkt mye på at Nofence kan brukes til sauesanking ved å innskrenke det geografiske området til den veien man vil at sauen skal gå. Nofence er godkjent av myndighetene og teller ikke som dyreplageri.

Rovdyr tar mer enn man tror - finnes død-deteksjon.

- Noe vi burde ha i bakhodet?

Vind er en viktig faktor. Ofte mye vind i fjellområder, noe som kan gjøre det vanskelig å fly dronen.

På Oppdal har man pleid annethvert år å spleise på leie av småfly. Da sitter en bonde på småflyet og informerer folk på bakken om hvor han ser sau.

Utstikkere befinner seg ofte på omtrent samme område fra sesong til sesong. Det betyr at selv om beiteområdet totalt kan være ekstremt stort, så trenger man ikke nødvendigvis å scanne et særlig stort område for å finne de man leter etter.

Sauen dropper fjellet når det er snø/is.

En viktig faktor er landskapsvern. Hvor har man lov til å fly droner?

Dato: 25.08.22

Works Cited (MLA 8th ed.)

"Albumentations Documentation." *Albumentations*, https://albumentations.ai/docs/. Accessed 14 December 2022.

"Bounding boxes augmentation for object detection." *Albumentations*, https://albumentations.ai/docs/getting_started/bounding_boxes_augmentation/. Accessed 14 December 2022.

"Full API Reference." *Albumentations*, https://albumentations.ai/docs/api_reference/full_reference/. Accessed 14 December 2022.

"Getting started on Idun – High Performance Computing Group." *High Performance Computing Group*, https://www.hpc.ntnu.no/idun/getting-started-on-idun/. Accessed 14 December 2022.

"Hardware – High Performance Computing Group." *High Performance Computing Group*, https://www.hpc.ntnu.no/idun/hardware/. Accessed 14 December 2022.

"How to Train YOLOv7 on a Custom Dataset." *Roboflow Blog*, Roboflow, 13 July 2022, https://blog.roboflow.com/yolov7-custom-dataset-training-tutorial/. Accessed 14 December 2022.

Hvasshovd, Svein-Olaf. "Oppgaveforslag at IDI NTNU." *Department of Computer Science (IDI)*, https://www.idi.ntnu.no/education/oppgaveforslag.php?oid=2179. Accessed 14 December 2022.

"IT3915masterpreparatoryproject." *RoboFlow*, 18 september 2022, https://app.roboflow.com/it3915masterpreparatoryproject. Accessed 14 December 2022.

"IT3915-master-preparatory-project." *GitHub*, https://github.com/trymgrande/IT3915-master-preparatory-project. Accessed 14 December 2022.

"IT3915-master-preparatory-project-old." *GitHub*, Trym Grande, https://github.com/trymgrande/IT3915-master-preparatory-project-old. Accessed 14 December 2022.

Johannessen, Kari Meling. "Towards Improved Sheep Roundup - Using Deep Learning-Based Detection on MultiChannel RGB and Infrared UAV Imagery." *NTNU Open*, https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2779322. Accessed 14 December 2022.

Kin-Yiu, Wong. "YOLOv7." *GitHub*, https://github.com/WongKinYiu/yolov7. Accessed 14 December 2022.

"Merged sheep dataset Image Dataset." *RoboFlow*, 1 November 2022, https://app.roboflow.com/it3915masterpreparatoryproject/merged-sheep-dataset/3. Accessed 14 December 2022.

NTNU. "Course - Visual Computing Fundamentals - TDT4195." *ntnu.edu*, https://www.ntnu.edu/studies/courses/TDT4195#tab=omEmnet. Accessed 14 December 2022.

Persson, Aladdin. "Machine-Learning-Collection/ML/Pytorch/Basics/albumentations_tutorial/." *GitHub*, 29 September 2022, https://github.com/aladdinpersson/Machine-Learning-Collection/tree/master/ML/Pytorch/Basics/albumentations_tutorial. Accessed 14 December 2022.

"Sheep detection 2 Image Dataset." *RoboFlow*, https://app.roboflow.com/it3915masterpreparatoryproject/sheep-detection-2/3. Accessed 14 December 2022.

"Sheep v4 (all images) Dataset." *RoboFlow*, SAU, February 2022, https://universe.roboflow.com/sau-cixmv/sheep-v4--all-images/dataset/1. Accessed 14 December 2022.

“Spatial-level transforms.” *Albumentations*, https://albumentations.ai/docs/api_reference/full_reference/#spatial-level-transforms. Accessed 14 December 2022.

“Training YOLOv7 on Custom Data - Colaboratory.” *Google Colab*, Roboflow, 13 July 2022, https://colab.research.google.com/drive/1X9A8odmK4k6l26NDviiT6dd6TgR-piOa. Accessed 14 December 2022.