Internship Report

By

Miguel Ibrahim

Supervisor: Prof. Filip De Turck

Supervisor at Kapernikov: Doc. Stef Vandermeeren

Ghent University

Master of Science in Computer Science Engineering

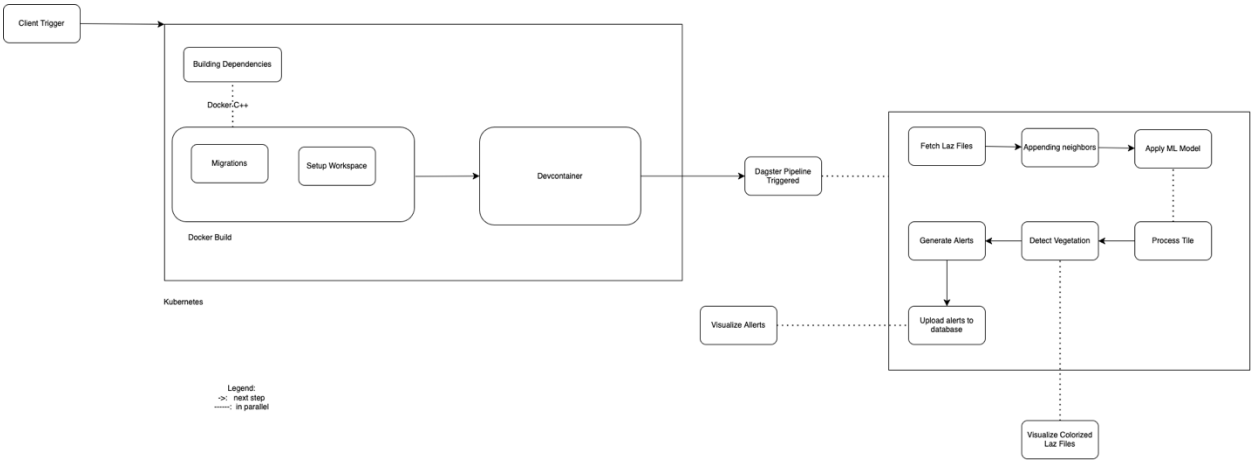
Data Engineering Internship at Kapernikov

Table of Contents

I. Introduction

Kapernikov is a 14-year-old company specialized in handling big data applications, analyzing those data inputs, and applying Artificial Intelligence Principles to solve clients' issues. Infrabel-SNCB is one of Kapernikov's biggest clients and has been working together for more than 5 years. Their goal is to make track maintenance and safety more efficient by applying a Machine Learning model on a point cloud. Today, Kapernikov is reaching that goal quickly and fine-tuning its models to be more robust on rough edges. Note: some details cannot be documented and mentioned in this internship report due to a signed NDA.

II. Architectural Overview



Dagster pipeline: Python

Analyzing results using the custom visualizer tool.

Database: SQL, Postgres

Database Visualizing Tool: DBeaver

Extras: Data Analytics using ipynb files and Excel

III. Quantitative Results:

- Old Model Accuracy: ~89%
- New Model Accuracy: ~98%
- Improved thresholds reduced false positives by 35%.
- Runtime per tile: ~90s
- Pipeline throughput: ~51 tiles in 1.5 hours
- Successfully integrated new model into pipeline

IV. My Role

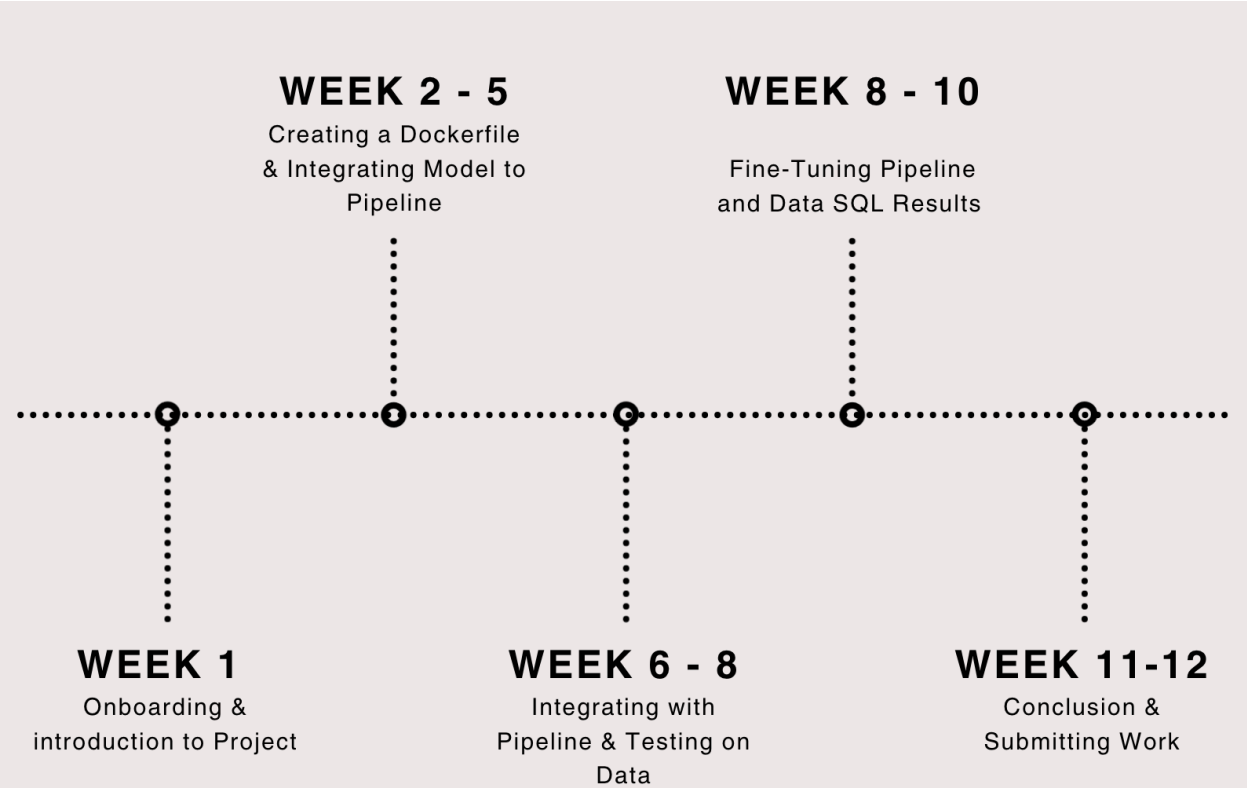
My Role as a Data Engineer Intern is to configure the newer fine-tuned model and integrate it into their pre-existing Dagster pipeline. To do this, I need to use Docker, Dagster, Python, CMAKE, C++, Linux commands, and Kubernetes.

This model is a custom-made C++ classification model that is trained on a point cloud and outputs probabilities for certain points in this point cloud, and classifies them as vegetation. This helps Infrabel-SNCB detect vegetation that starts to enter the track area, potentially causing a safety concern, such as ingrown trees, vegetation on tracks (which may cause the train to slip), vegetation in the catenary (wires), which may cause a fire, etc.

This new model is important since it reduces runtime, is more accurate, takes less energy during inference, and does not require too much memory and/or CPU power. This model is also

important since Infrabel-SNCB is planning to use this technology to expand its sphere of influence throughout Europe and put Belgium on the map as the leader for rail efficiency and safety.

V. Internship Timeline



Actual: I completed the requirements by week 5. After week 5, I will improve my work and train the ML model to increase accuracy from ~90% to ~100%.

A. Week 1: Onboarding & Introduction to Project

Start Date: Thursday, February 6, 2025. The first day at the internship involved setting up an Ubuntu distribution laptop that took half the day due to administration issues, as well as 2FA requirements that had been previously set. Day one also included an introduction to the project by team members, and a deeper dive into what sections of code and project(s) I will be working on during the interval of the internship. I then set up my environment and began understanding the code written in C++, CMAKE, Python, Docker, etc.

Day 2: 10 am included a sync with my team members, I updated them on the situation & what happened the day before and told them I will start working on creating a Docker file to automate the Linux commands for running the classification model and quickly started to collect all the commands needed to run the inference and compiled them into a Docker file. At this point, I asked one of my team members for help with gathering the commands needed, and they very nicely provided those.

B. Week 2: Creating Docker file & Integrating Model to Pipeline

Day 1: 10 am, there was a sync with all the team members where I updated them on the current situation of the project and requested help from one of my team members to reduce the installation time required by the project packages. Then on Day 1, I completed the Docker file and asked for one of my team members' help with debugging the code due to a recurring error, and it was quickly squashed. This error referred to a line in the Docker file that was not currently in the right folder. This was then pointed to the right folder, and the commands then worked perfectly and outputted a probability file.

After this issue was solved, I then added the commands necessary for compiling the model into the main pipeline Docker file, of which I made the command shorter and linked it to another RUN operation to reduce program complexity and space complexity.

Day 2: 10 am, there was a sync with all the team members where I updated them on the current situation of the project, and they introduced me to the pipeline structure and how to initialize the pipeline using a devcontainer. This devcontainer opened a Dagster website that showed the orchestration of the Python files, Docker files, and well as sensors. After the sync, I started to understand the structure of the pipeline and to understand how files interacted with each other. I also tried to run the pipeline on my company machine, with some issues after adding the model to the pipeline. I then notified one of my team members, and they helped me solve that issue quickly.

i. Week 3

Day 1: Updating the team about the integration of the classification model into the devcontainer. This means the classification model can now be run in the devcontainer as well as be called from functions in the program. I also learned about the structure of the project as well as the risks I need to keep in mind when manipulating data. I also sat in a meeting with the stakeholders to discuss the current issues with the program and ideas they want to add to the project.

Day 2: Updating the team about the successful integration of the executable in the dev container, I also expressed my concerns about some aspects in the project, such as redundant processes, inefficient code, and lack of parallelization. On this day, I met with a

Data Engineer to discuss the structure of the pipeline and how everything is orchestrated. They also gave me access to the database and showed me around with the structure and the optimal layout to perfectly process data and query it in the database.

Skills Learned till Now

Refreshed SCRUM Agile Process. Dived Deeper into Docker, Dockerfiles, devcontainer: running multiple images, running migrations on a Docker image, executing operations inside Docker images. Linux, Linux command lines, C/C++, CMAKE. Refreshed Python programming language. Refreshed SQL syntax, ER diagram, data dependencies, foreign tables, used PostgreSQL, etc. Introduced to Dagster pipeline orchestration. Introduced to Big Data Structures & Algorithms. Using Parallel algorithms to solve matrix-matrix multiplications (BLAS). Introduced to Microsoft DevOps. Introduced to Atlassian Confluence. Introduced to Slack. Introduced to DBeaver to visualize the database structure and data.

Indirect things learnt teamwork skills, problem solving, communication, work culture, corporate language, etc.

ii. Week 4

Day 1: Updating the team about the capability of running the model in the devcontainer directly, which means that the model can now be integrated into the pipeline. To do this, I need to create an execution function in Python that calls the executable in the Docker image. On this day, I was later introduced to the Bucket, which is an online database that stores multiple items. I later created a Python

function that called the model and outputs the probability to find a vegetation point into an output, directly later used by the pipeline.

I later had a meeting with the team leader about the scope of the project, They told me that I need to refactor what I did in the devcontainer to the pipeline Docker files. There are 3 Dockerfiles in the pipeline configuration that all depend on each other, this is done to reduce the compile time of all Dockerfiles. I completed the migration of all devcontainer commands in the pipeline Dockerfiles.

Day 2: Day two had a lot of processing. I started to add the new AI model into the pipeline, and it worked very quickly after some debugging. This debugging included being very specific with folders as well as modifying YAML files, which proved hard to modify through Python. Afterwards, I pulled the main branch, which included many fixes. At this point, I learned to use a new command line for git, which is –mine or –their, which chooses which version to use. After successful refactoring, I modified the code to now use the new devcontainer and compiled the pipeline, which seemed to work at first, but it wasn't saving the information in a database.

iii. Week 5:

Day 1: On this day, I moved some of the files to the right places for best practices so they can reuse code for production. After moving the files around, it made my problem disappear: files not found, unnecessary compilation in the devcontainer,

and unnecessary folders. After doing that, I moved on to create a function that can publish the results of the AI model to the database. After many debugs later, I reran the Dagster pipeline, and I ran into issues related to permissions: sometimes Docker has root access, which is not good because you don't want to use sudo commands. The issue stemmed from creating a folder from inside the pipeline, which requires root access. To solve that, use an existing high-privilege folder. After all of that is solved, finally, the new model is in the pipeline, and the next steps are to set up for production.

I am then allocated a new task, which is to train the Machine Learning model by annotating the .laz files. Check if the laz files are predicted correctly, compare to the actual photo of the area, or use domain knowledge (point cloud).

Day 2: Generating the files. The campaign has 40 files of which are tiles (sections of an entire laz file). Each tile takes about 10 minutes to run fully on the pipeline => Wait 400 minutes to get the files to fully generate. In the meantime, I started to create a script in Python to call the new and old models, and to compare them with SOTA algorithms used to detect vegetation on a 3D point cloud.

iv. Week 6:

Automating the script so it can process multiple inputs instead of one at a time. This file is used to process one file only, and I had to change the file name within the Python code. Now the code takes a file input from the pipeline (if needed) and runs the commands to compute the Vegetation Probabilities on multiple different model types. I have also created an IPython notebook to analyse the Vegetation

Probabilities and to compare with the actual output probability file. This file must read binary data and transform it into floats between 0 and 1. I then compare the results from the inference data to the binary file in the actual labeled file.

I have also created a function that parses through the thresholds because values >0.5 probabilities are the original threshold. I then compute the accuracy, recall, etc., for this threshold to choose the best one while limiting overfitting.

On day 2, I have discovered that one model has an accuracy of 98% vs the old accuracy of 89%, so I made it my goal to use that model now in the pipeline.

v. Week 7 (March 20 -21)

Day 1: I have presented my findings to the team and am starting to work on understanding the best thresholds to know why other models failed. I have created an IPython notebook to understand the best thresholds by testing multiple thresholds against the true value, and found that thresholds for the old files were very low (0.2) to achieve high accuracies compared to the 0.76 threshold on the most optimal model signifying that it had many false positives instead of false negatives.

Day 2: Filtering, Day two is to create a executable script that runs the algorithm on multiple files and view them to see what is going on and what sections of the file it is making mistakes on, this took some time since the algorithm had to run on 51 files of which took on average 90 seconds on each file.

vi. Week 8 (March 27 – 28)

Day 1: I have presented my findings in these files, but I also mentioned that it is important that I focus on formalizing the integration of the model in the production branch, which did not go many. I then showed the importance of switching from the old model to the more accurate model, even if it takes 10x the amount of time. This day was a day to review the comments made on my pull request, of which many were just some function name changes, and to remove some comments.

Day 2: I did some research on the state-of-the-art models on how to detect vegetation in a 3D LiDAR point cloud.

vii. Week 9 (April 7 – 8)

Day 1: Creating some files with low thresholds (0.35) compared to the default 0.5, I only did the algorithm for 5 files and compared to the default threshold, instead there are many false positives, and we learned that the model does not perform well on flat surfaces near trees or under trees such as barriers.

Day 2: This day I spent entirely trying to refactor the code, checking if the pipeline still works with the better model – it did not work, so some functions needed to be fixed and some code needed to be optimized.

viii. Week 10 (April 10 – 11)

Day 1: Presenting my final findings with the new model and modifying the files in an annotation tool to be trained again, making sure that the model is more robust against these issues, which are mostly undetected trees, as well as barriers being detected as vegetation when vegetation is nearby. I also mentioned that it is important to test the implementation on production.

Day 2: New employees are being interviewed in person, and they asked us what it is like working at Kapernikov and how to prepare for future interviews, as well as what to expect when working here.

ix. Week 11 (April 17 – 18 End)

Testing the model on production, this process needed some editing, but in the end, the model worked perfectly on the production branch. On the last day I investigated the potential to integrate a hybrid model between the two models, one maximizes true positives the other maximizes true negatives, in the end this did not work as one model will pull harshly in one direction (true positive) while the other will pull the model in the other direction (true negative). This resulted in a model that performed poorly in the real world. This again is a practice of Machine learning principles that if a model performs well on training data, it does not mean it will perform well on real-world data, signifying an overfit on the training data.

Conclusion

My internship at Kapernikov has been an incredibly enriching experience, both technically and personally. Over the course of eleven weeks, I had the opportunity to contribute meaningfully to a real-world data engineering pipeline, working with cutting-edge technologies like Docker, Dagster, and Kubernetes, while also deepening my understanding of machine learning, point cloud classification, and large-scale data orchestration.

From integrating a more efficient and accurate C++ classification model into a production-level pipeline to optimizing Docker images and automating processing scripts, I was consistently

challenged to adapt, problem-solve, and innovate. The process of refactoring code for scalability, debugging complex pipelines, and tuning ML models against real-world edge cases has taught me not only technical skills but also the critical importance of software robustness and production-readiness.

This internship has also underscored the importance of communication and teamwork in a professional setting. Regular syncs with teammates, collaborative debugging, and constructive feedback on pull requests all played a vital role in accelerating my learning and improving the overall quality of the work delivered. I'm especially grateful for the guidance and support of my mentors, Prof. Filip De Turck and Doc. Stef Vandermeeren, Louis, and Przemeck, whose insights were instrumental throughout the journey.

Looking ahead, this internship has not only reaffirmed my passion for data engineering and machine learning but also equipped me with the hands-on experience and confidence to tackle complex technical problems in the industry. I leave Kapernikov with a sense of accomplishment and enthusiasm for future challenges, knowing that my work has contributed, however modestly, to the broader goal of enhancing railway safety and efficiency in Belgium and beyond.