Deteksi Masker Menggunakan YOLO

Steven A S (c14180049)
Kevin Giovanni (c14180086)
Vincent Natanael (c14180141)
August Berlin T (c14180003)
Eric Saputra L (c14180048)

Latar Belakang

- Pandemi COVID-19, pertama di era modern
- Penekanan tingkat penyebaran -> masker
- Pengawasan protokol kesehatan

Tentang YOLO

- Real-time Object Detection
- Membagi image menjadi beberapa bagian -> Deteksi Objek
- Menghitung klasifikasi objek jika diasumsikan objek -> berjalan paralel
- Sampai pada hari ini YOLO memiliki 5 versi dan versi yang dipakai adalah

  YOLO versi 4 dan YOLO versi 3

YOLO v2

- Resolusi image pada training sama dengan testing dan pemakaian umum
- Menggunakan anchor box hasil K-means clustering dan menghitung offset untuk memprediksi bounding box
- Jumlah layer YOLO v2 diperbanyak

YOLO v3

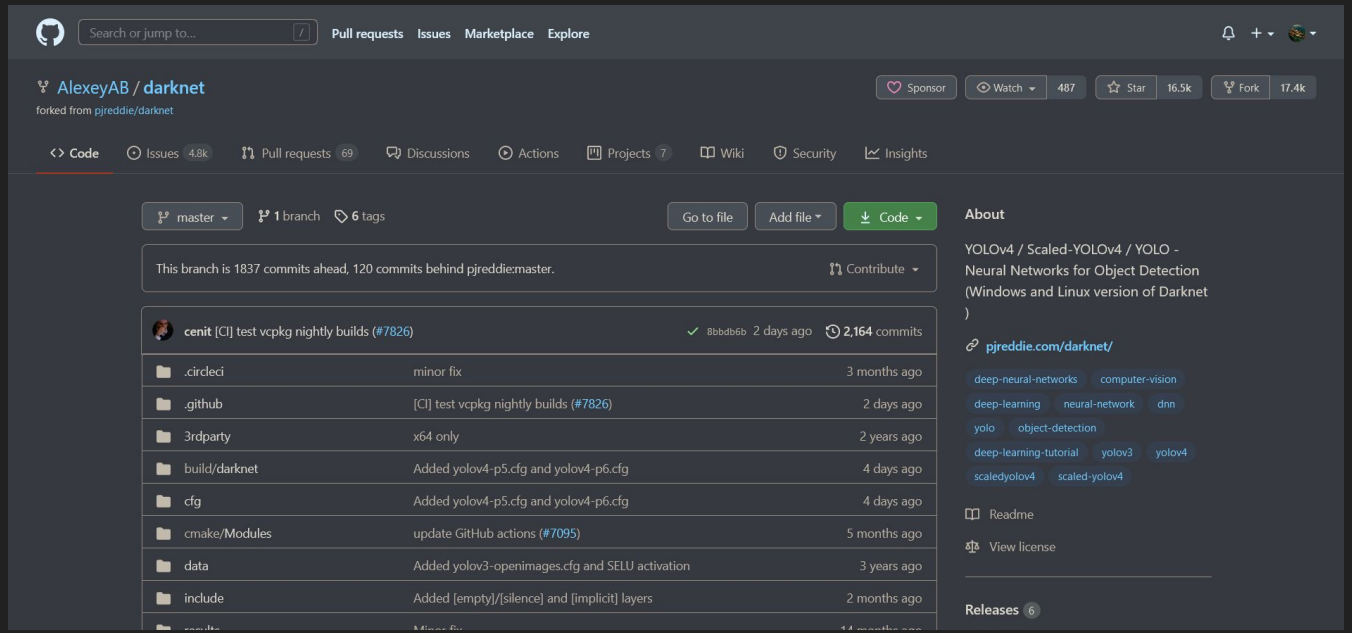
- Prediksi bounding box memakai logical regression
- Class prediction diganti dari softmax menjadi independent logistic classifier karena lebih cocok untuk multi class identifier dan softmax tidak berpengaruh pada performa YOLO
- Feature extraction memakai Darknet-53, hasil modifikasi dari Darknet-19 milik YOLO v2
- Memiliki kelemahan mendeteksi benda berukuran sedang dan besar

YOLO v4

- Struktur sama dengan YOLO v3
- Resolusi lebih besar, jumlah layer dan jumlah parameter lebih banyak
- Berbagai fungsi-fungsi yang digunakan dalam YOLO v3 digantikan dengan fungsi lain yang sudah diteliti lebih lanjut
- Lebih cepat dan akurat dibandingkan YOLO v3

Setup

- Git clone https://github.com/AlexeyAB/darknet



Build DarkNet

Mengubah flag dalam file Makefile

- OPENCV = 1
- CUDNN = 1
- GPU = 1
- CUDNN_HALF = 1

GPU lebih cepat dari CPU, karena itu kita memakai GPU flag. Selain itu, tidak ada modifikasi tambahan.

Training

- Framework: AlexeyAB’s Darknet
- File-file yang disiapkan:
  - Obj.data (letak file names, train, test, folder backup)
  - Obj.names (berisi nama-nama class yang di train)
  - Train.txt dan test.txt (path dari file yang digunakan untuk training dan testing)
- Image preparation
  - 1 txt file untuk 1 image
  - Txt file berisi satu atau lebih label (kelas, titik pusat box, ukuran box)
  - Titik pusat dan ukuran relatif terhadap ukuran image aslinya

Dataset

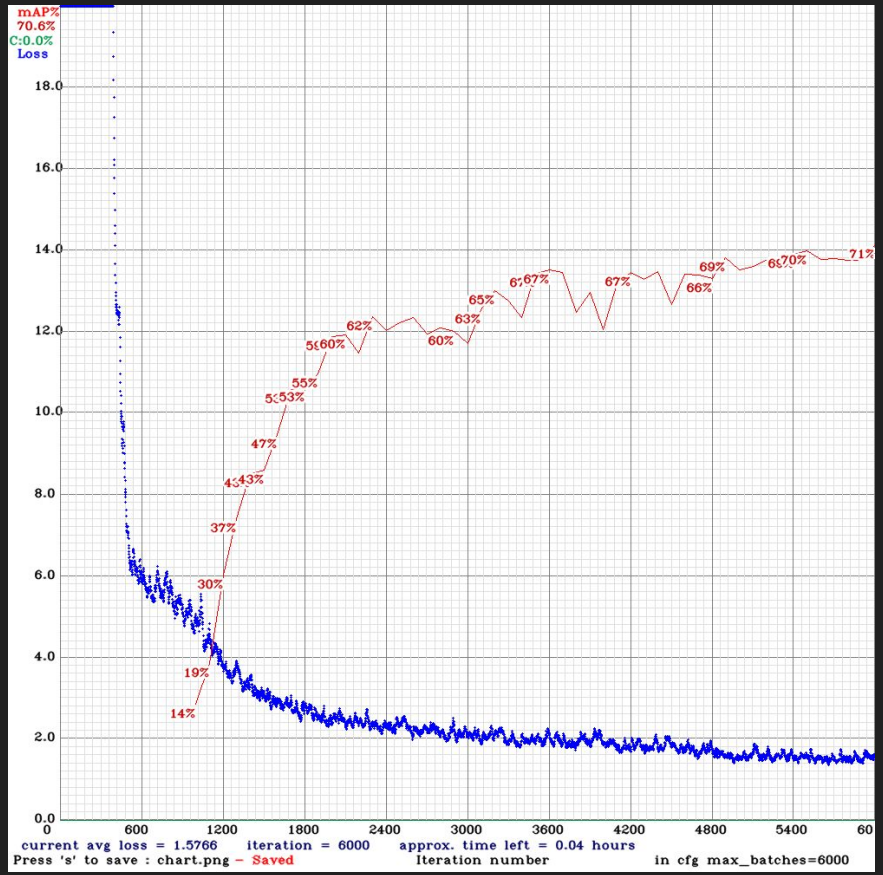
- Face Mask Dataset (YOLO Format)
- File structure:
  - Train folder:
    - 610 gambar jpg
    - 82 file jpeg
    - 8 file png
  - Test folder:
    - 105 file jpg
    - 12 file jpeg
    - 3 file png

Settings

- Jumlah class = 2 class
- Jumlah maksimum step = 6000 steps
- line_steps = 4800, 5400
- Batch size = 64
- Subdivision = 16
- Input size = 416 x 416
- Platform: Google Colab (GPU: NVIDIA T4, CUDA 11.0, CuDNN 7.6.5)
- Compile flag: CUDNN = 1, CUDNN_HALF = 1, GPU = 1, OPENCV = 1

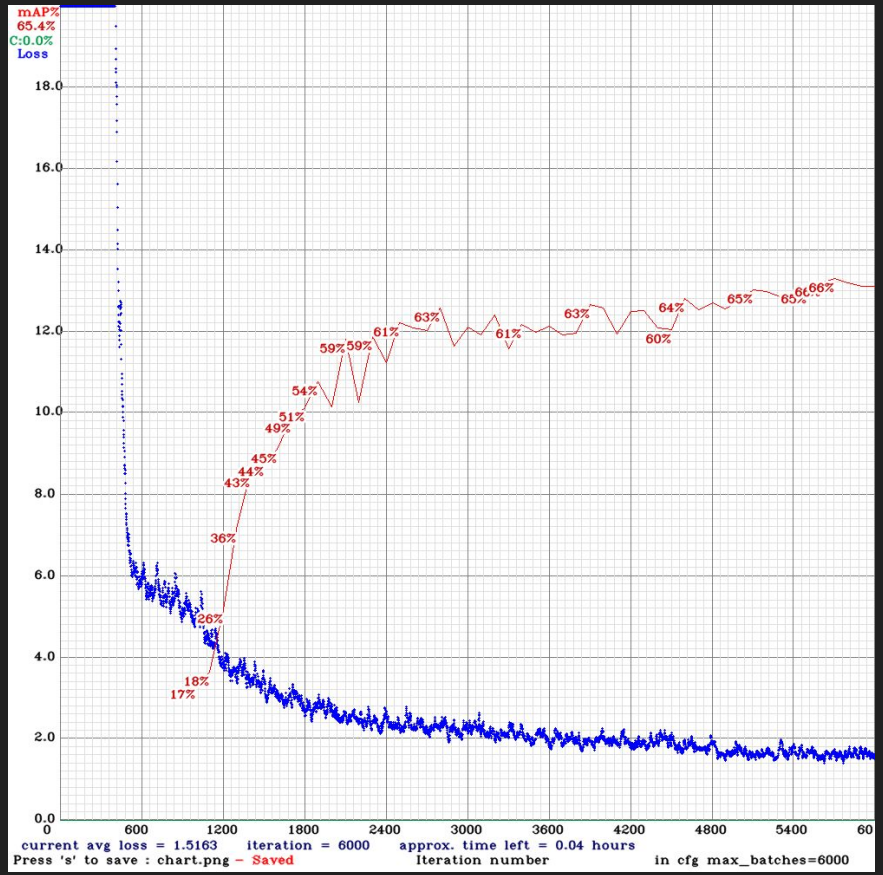
Training 1

- Yolo v3 Tiny
- 610 jpg train
- 105 jpg test
- mAP: 70.6%
- Train time: 3 jam 35 menit



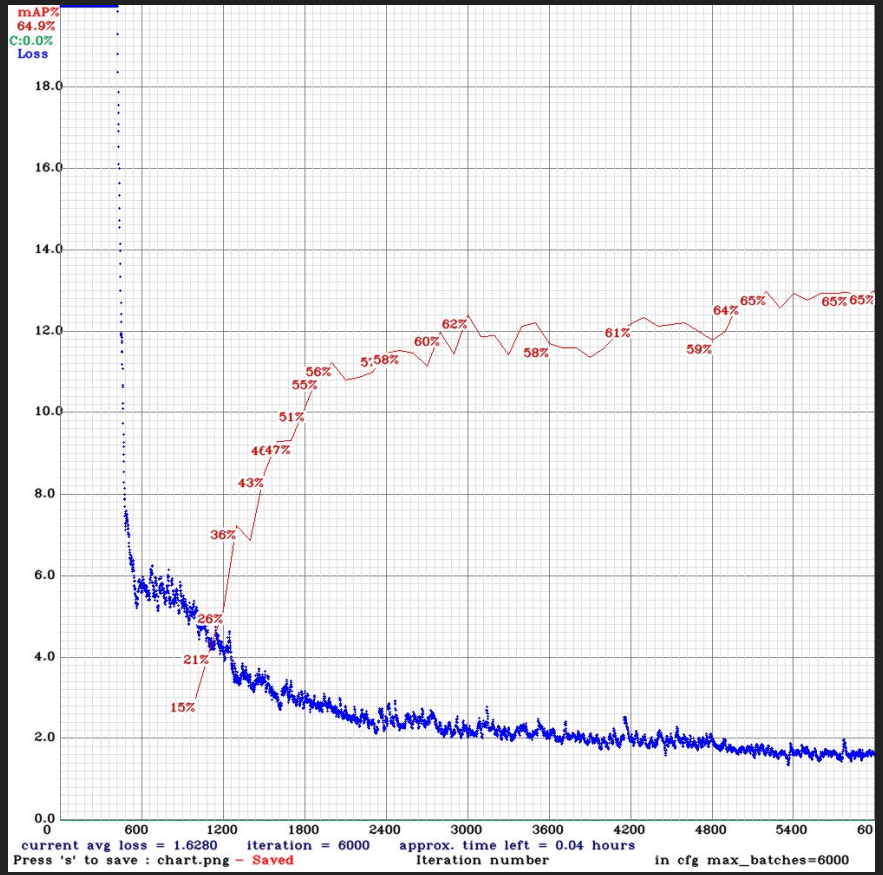
Training 2

- Yolo v3 Tiny
- 700 jpg train
- 120 jpg test
- mAP: 65.4%
- Train time: 2 jam 44 menit



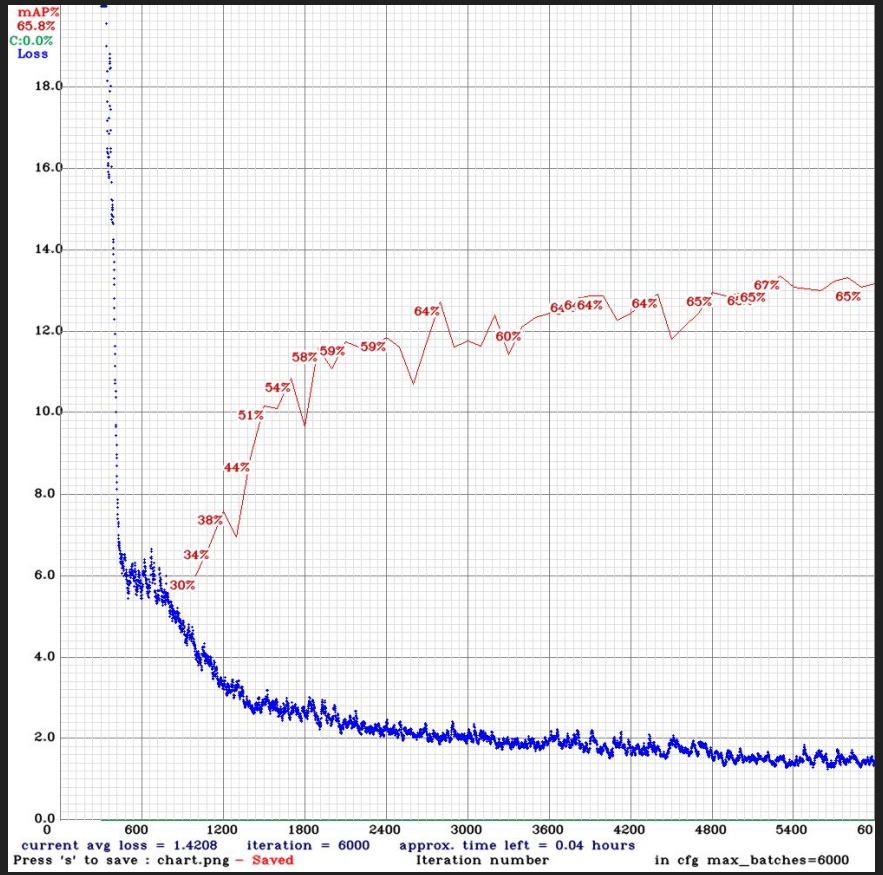
Training 3

- Yolo v3 Tiny
- 700 jpg train (histogram equalized)
- 120 jpg test (histogram equalized)
- mAP: 64.9%
- Train time: 2 jam 22 menit



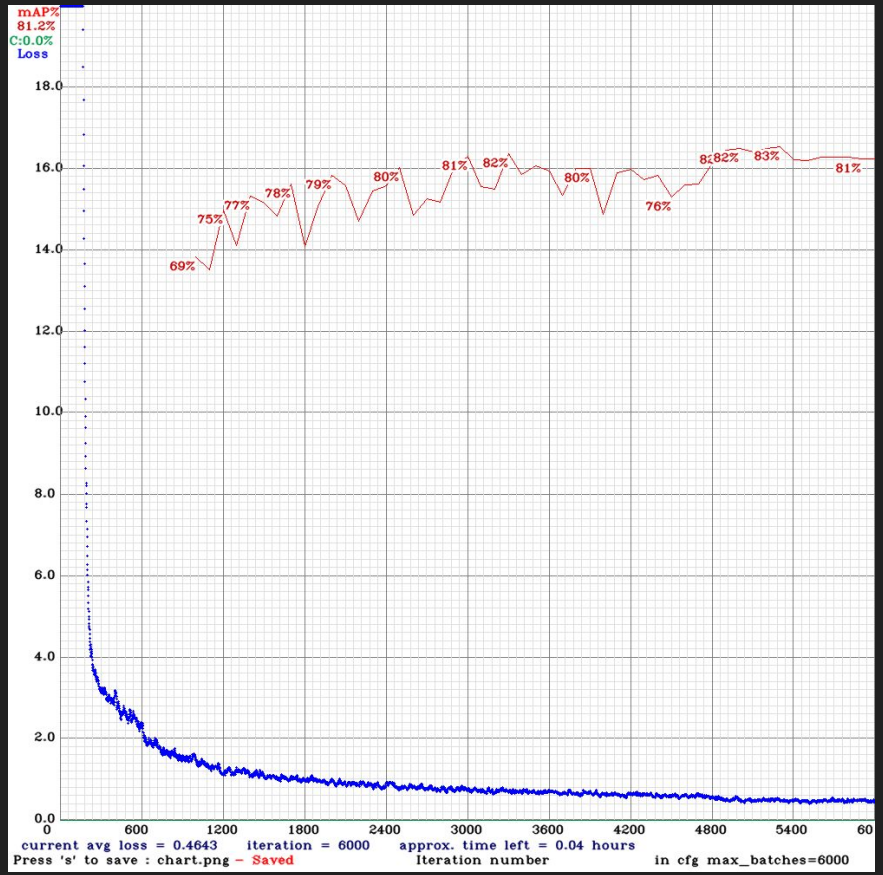
Training 4

- Yolo v3 Tiny
- Learning rate = 0.002
- 700 jpg train
- 120 jpg test
- mAP: 65.8%
- Train time: 2 jam 34 menit



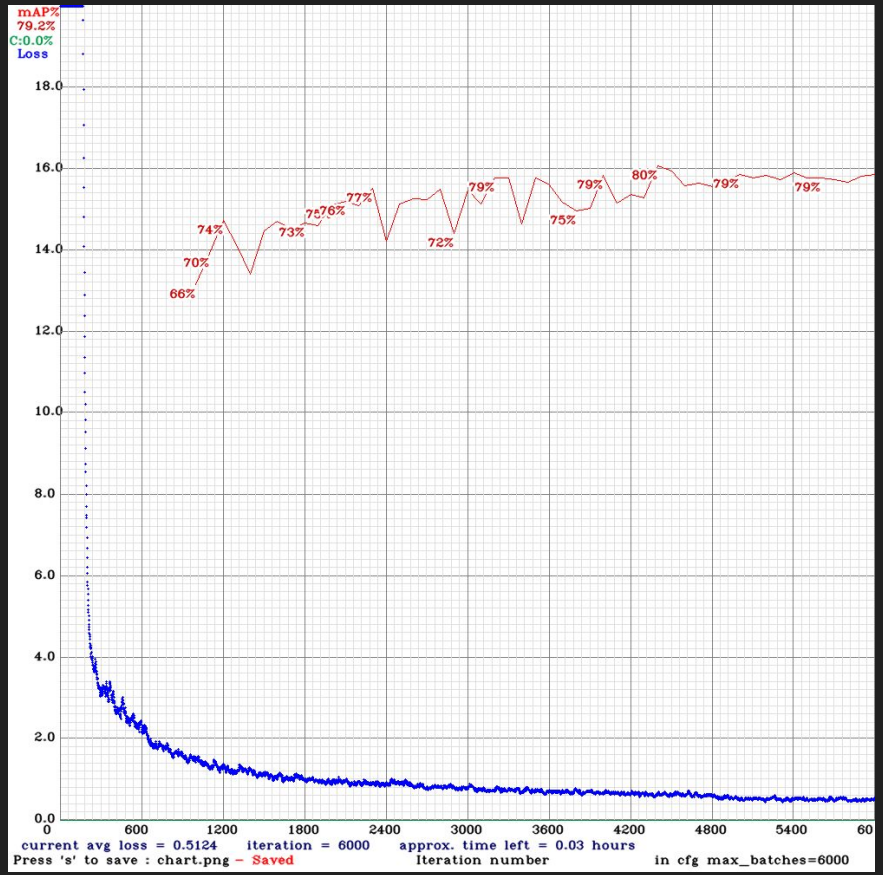
Training 5

- Yolo v4 Tiny
- Learning rate = 0.002
- 700 jpg train
- 120 jpg test
- mAP: 81.2%
- Train time: 2 jam 14 menit



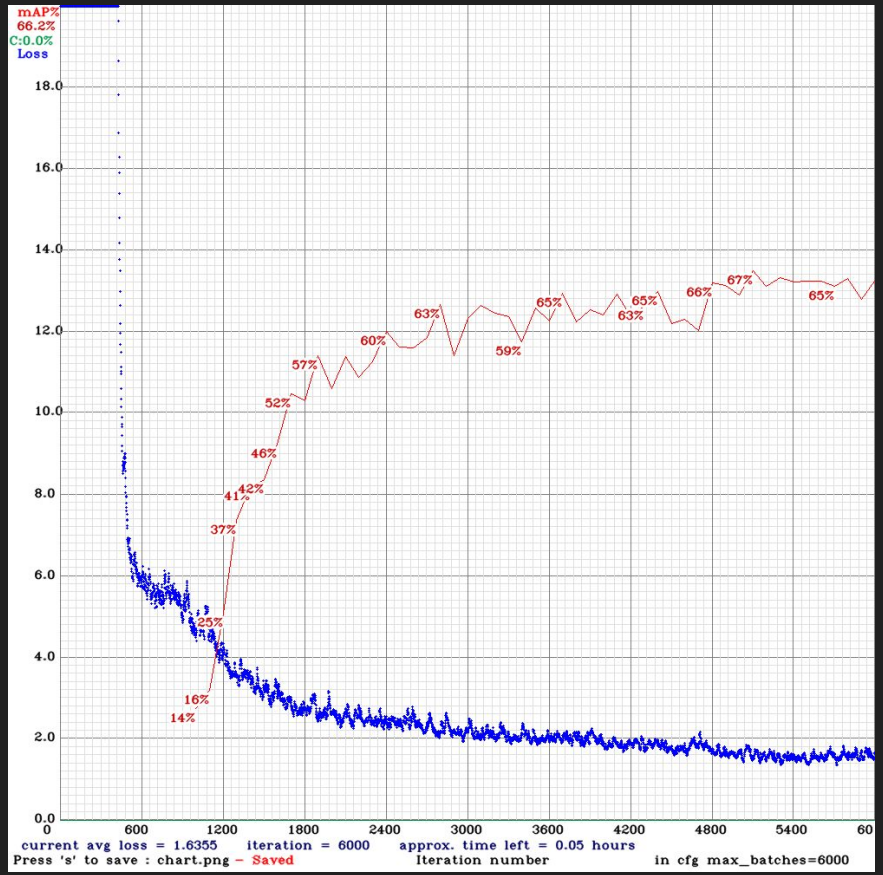
Training 6

- Yolo v4 Tiny
- Learning rate = 0.002
- 700 jpg train (histogram equalized)
- 120 jpg test (histogram equalized)
- mAP: 79.2%
- Train time: 1 jam 55 menit



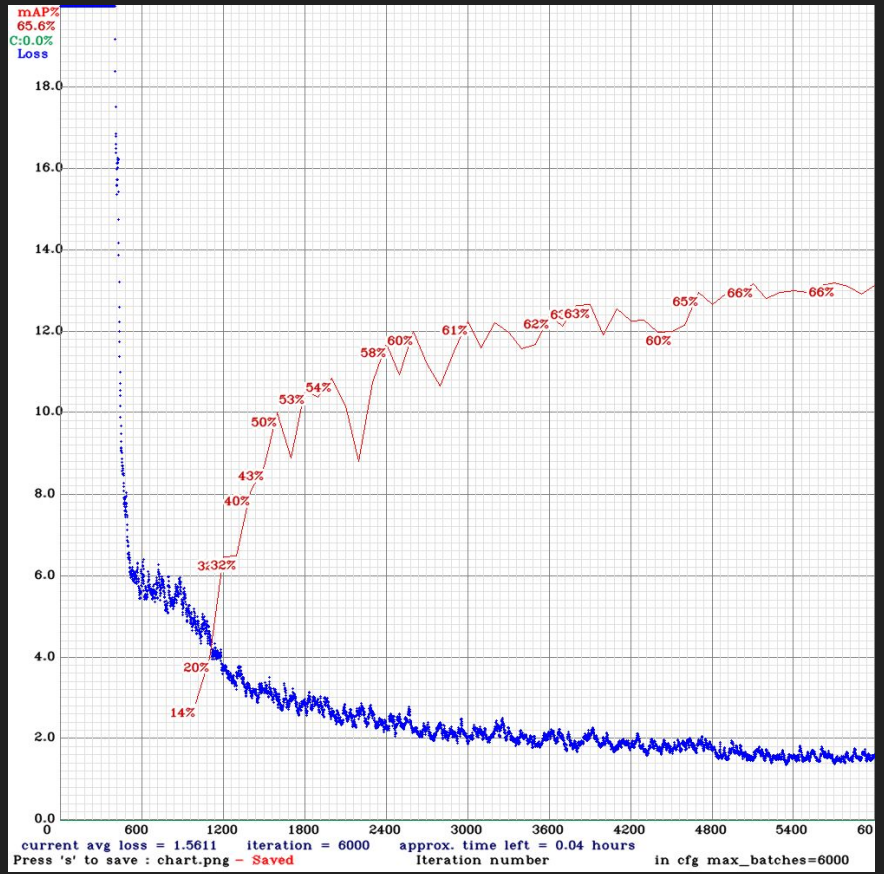
Training 7

- Sama seperti eksperimen ketiga
- Learning rate: 0.001
- Lama training: 3 jam 30 menit
- mAP: 66.2%



Training 8

- Menggunakan obj2 fully equalized
- No random training
- Learning rate: 0.001
- Lama training: 2 jam 43 menit
- mAP: 65.6%



Kesimpulan

- Keseimbangan Dataset berpengaruh pada hasil training
- Learning rate berpengaruh pada akurasi awal
- Yolo v4 memiliki akurasi yang lebih tinggi dari Yolo v3