

Laporan Machine Learning

Image Similarity Cifar-10

Kelompok 9

1. Nama Anggota dan Pembagian Kerja

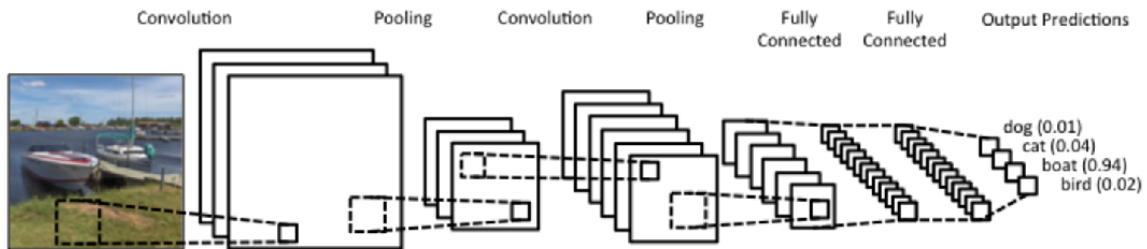
Nama	NRP	Pembagian Kerja
David Riski Tanoto	C14190003	Tes hasil apabila kelas yang diuji adalah 7 kelas
Ryan Jonathan S.	C14190008	Tes hasil apabila kelas yang diuji adalah 4 kelas
Rensis Yehuda	C14190111	Tes hasil apabila kelas yang diuji adalah 10 kelas

2. Teori Model dan Dataset

a. Teori Model

Pada project "*Image Similarity*" menggunakan machine learning CNN atau *Convolutional Neural Network*. CNN merupakan salah satu jenis neural network yang dapat digunakan untuk mendeteksi dan mengenali object (gambar) tersebut. Terdapat 2 proses selama proses mendeteksi yaitu *Convolution Layer* dan *Classification*. Pada Convolution layer dibagi 2 yaitu convolution dan pooling. Proses convolution ini adalah proses encoding ke dalam bentuk angka-angka (array 2 dimensi). Convolution layer ini digunakan untuk menyusun sedemikian rupa gambar dengan memfilter dalam bentuk 2 dimensi. Pada tiap pergeseran, input dan nilai dari filter dihitung menggunakan operasi dot sehingga dapat menghasilkan sebuah output. Setelah melakukan perhitungan "dot" dan memiliki nilai output, output ini akan dihitung kembali dan mencari nilai terbaik. Pada project ini, kami menggunakan `averagepooling2D()` yang artinya pada saat pergeseran,

akan menghitung nilai average pada 2Dnya. Setelah melakukan nilai average, diperlukan fase *reshape* bentuk gambarnya menjadi sebuah vektor agar bisa digunakan sebagai input. Berikut secara gambar menjelaskan cara kerja CNN.



Pada project ini, kami menggunakan model softmax dengan modifikasi menggunakan *metric learning* AnchorPositive.

AnchorPositive merupakan metode *learning* untuk melihat kemiripan. Anchor adalah random data dari sebuah kelas, sedangkan Positive adalah merupakan random data yang memiliki kemiripan dengan data Anchor. Fungsi softmax merupakan fungsi untuk urutan indeks kelas pada tiap gambar yang akan di training dan akan menghitung loss dari data yang akan di training. Berikut *code* untuk mempraktikkan AnchorPositive tersebut ke dalam modelling.

```
class EmbeddingModel(keras.Model):
    def train_step(self, data):
        # Note: Workaround for open issue, to be removed.
        if isinstance(data, tuple):
            data = data[0]
            anchors, positives = data[0], data[1]

        with tf.GradientTape() as tape:
            # Run both anchors and positives through model.
            anchor_embeddings = self(anchors, training=True)
            positive_embeddings = self(positives, training=True)

            # Calculate cosine similarity between anchors and positives. As they have
            # been normalised this is just the pair wise dot products.
            similarities = tf.einsum(
```

```

        "ae,pe->ap", anchor_embeddings, positive_embeddings
    )

    # Since we intend to use these as logits we scale them by a temperature.
    # This value would normally be chosen as a hyper parameter.
    temperature = 0.2
    similarities /= temperature

    # We use these similarities as logits for a softmax. The labels for
    # this call are just the sequence [0, 1, 2, ..., num_classes] since we
    # want the main diagonal values, which correspond to the anchor/positive
    # pairs, to be high. This loss will move embeddings for the
    # anchor/positive pairs together and move all other pairs apart.
    sparse_labels = tf.range(num_class_train)
    loss = self.compiled_loss(sparse_labels, similarities)

    # Calculate gradients and apply via optimizer.
    gradients = tape.gradient(loss, self.trainable_variables)
    self.optimizer.apply_gradients(zip(gradients, self.trainable_variables))

    # Update and return metrics (specifically the one for the loss value).
    self.compiled_metrics.update_state(sparse_labels, similarities)
    return {m.name: m.result() for m in self.metrics}

```

Modelling ini untuk mencari nilai *similarities* dari konsep AnchorPositives. Nilai *similarities* ini akan dijadikan sebagai bahan untuk nilai probabilitas (seperti penggunaan *Softmax*). Probabilitas inilah untuk menentukan label gambar yang di train dan akan digunakan untuk proses testing.

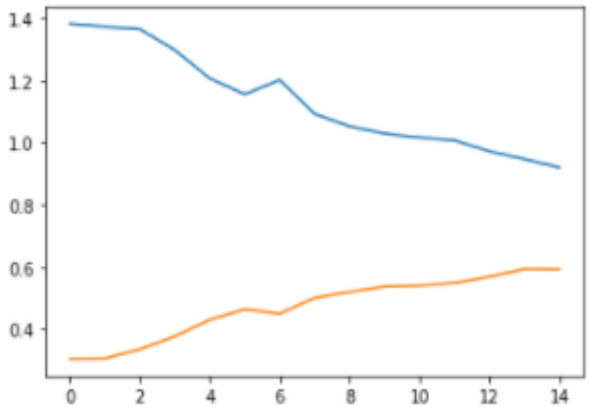
b. Dataset

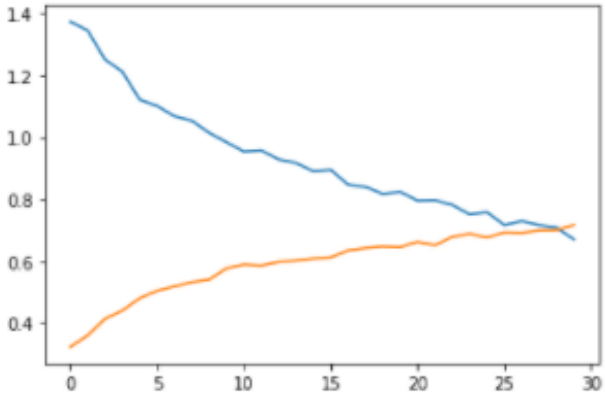
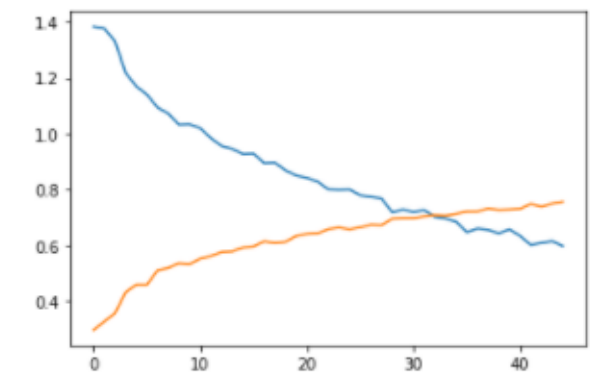
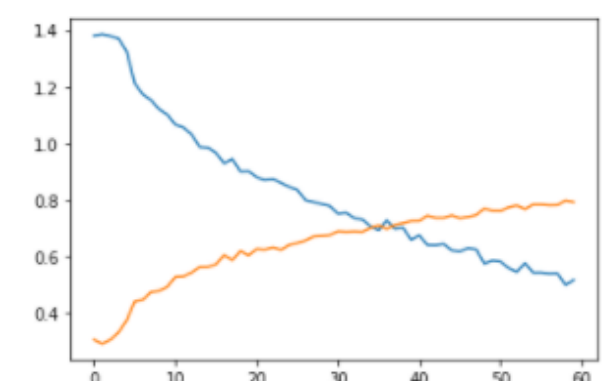
Dataset yang kami gunakan adalah dataset dari CIFAR-10. Pada CIFAR-10 memiliki 60000 data gambar yang terdiri dari 50000 data training dan 10000 data testing serta terdapat 10 jenis klasifikasi. Setiap kelas memiliki jumlah 6000 data gambar yang terdiri 5000 data training dan 1000 data testing.

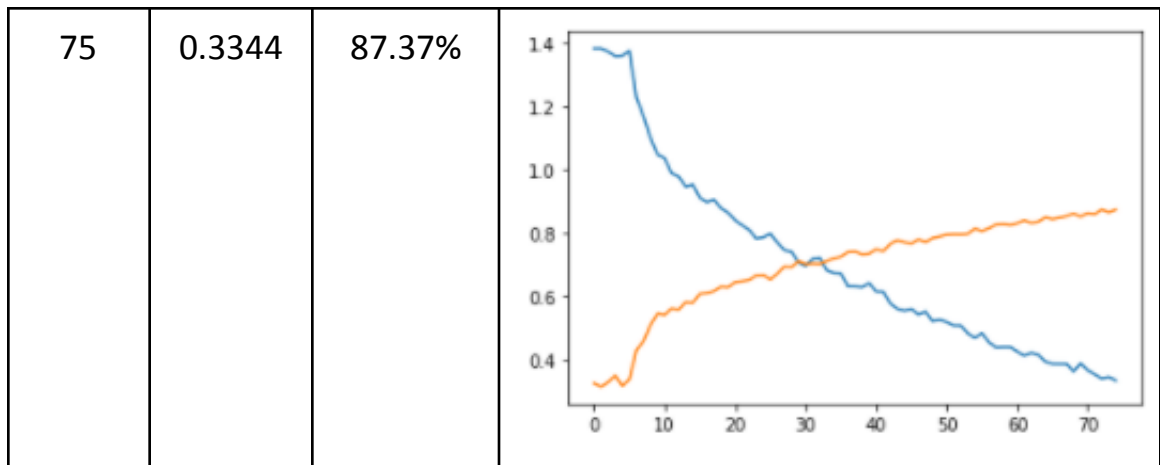
3. Hasil Pengujian dan Analisa (Numbatch = 1000)

a. Tes hasil apabila kelas yang diuji adalah 4 kelas

Tabel akurasi dan loss pada epoch:

Epoch	Loss	Akurasi	Graphic
15	0.92	59.25%	

30	0.67	71.72%	 <p>A line graph with the x-axis ranging from 0 to 30 and the y-axis from 0.4 to 1.4. A blue curve starts at approximately 1.35 at x=0 and decreases steadily to about 0.7 at x=30. An orange curve starts at approximately 0.3 at x=0 and increases steadily to about 0.7 at x=30. The two curves intersect at approximately x=28.</p>
45	0.5978	75.67%	 <p>A line graph with the x-axis ranging from 0 to 45 and the y-axis from 0.4 to 1.4. A blue curve starts at approximately 1.35 at x=0 and decreases to about 0.6 at x=45. An orange curve starts at approximately 0.3 at x=0 and increases to about 0.75 at x=45. The two curves intersect at approximately x=35.</p>
60	0.5146	79.00%	 <p>A line graph with the x-axis ranging from 0 to 60 and the y-axis from 0.4 to 1.4. A blue curve starts at approximately 1.35 at x=0 and decreases to about 0.5 at x=60. An orange curve starts at approximately 0.3 at x=0 and increases to about 0.8 at x=60. The two curves intersect at approximately x=38.</p>

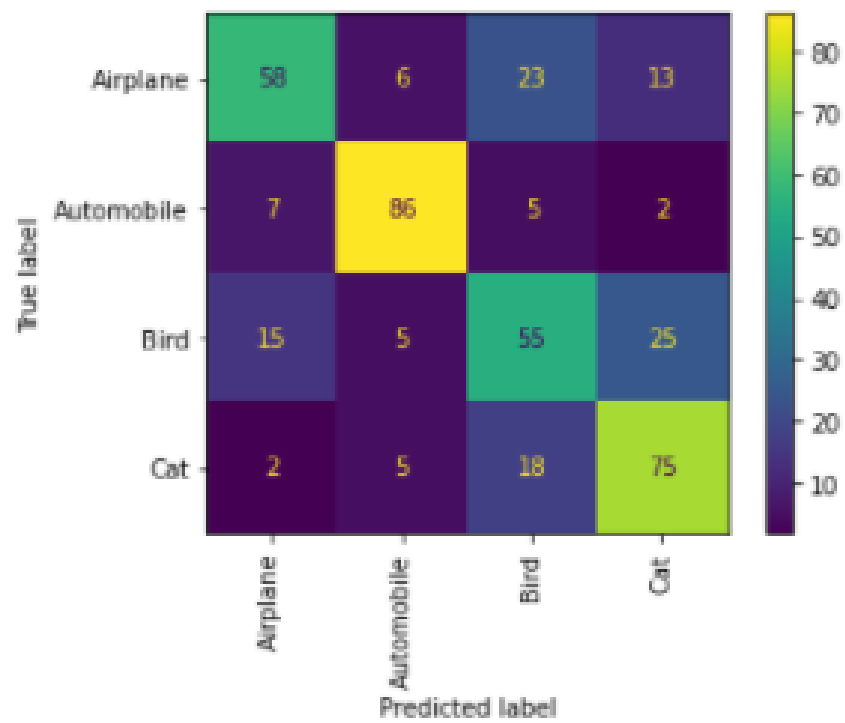


* garis biru merupakan garis loss dan garis kuning merupakan garis akurasi

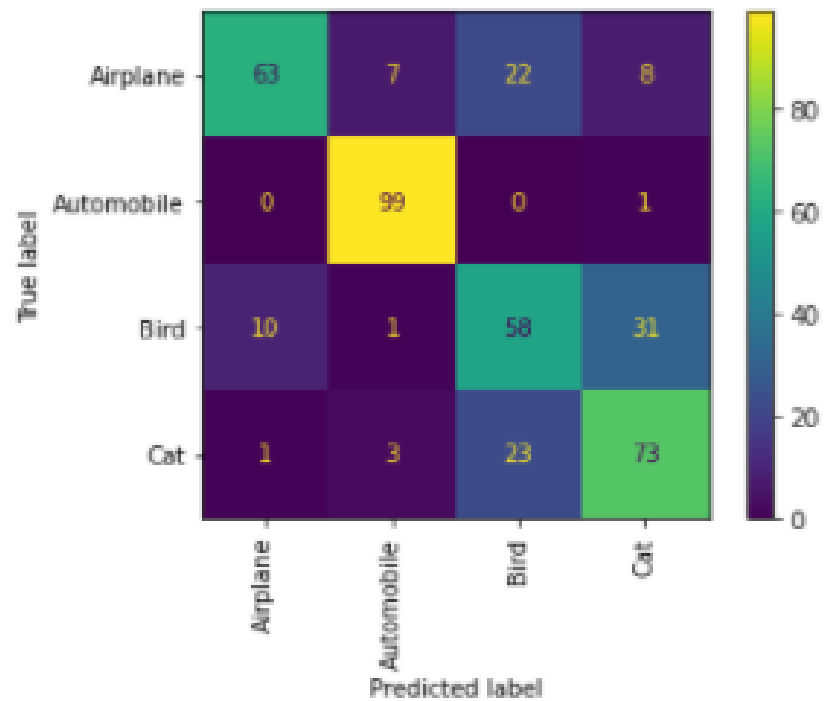
Tabel hasil confusion matrix:

Epoch	Hasil Confusion Matrix
-------	------------------------

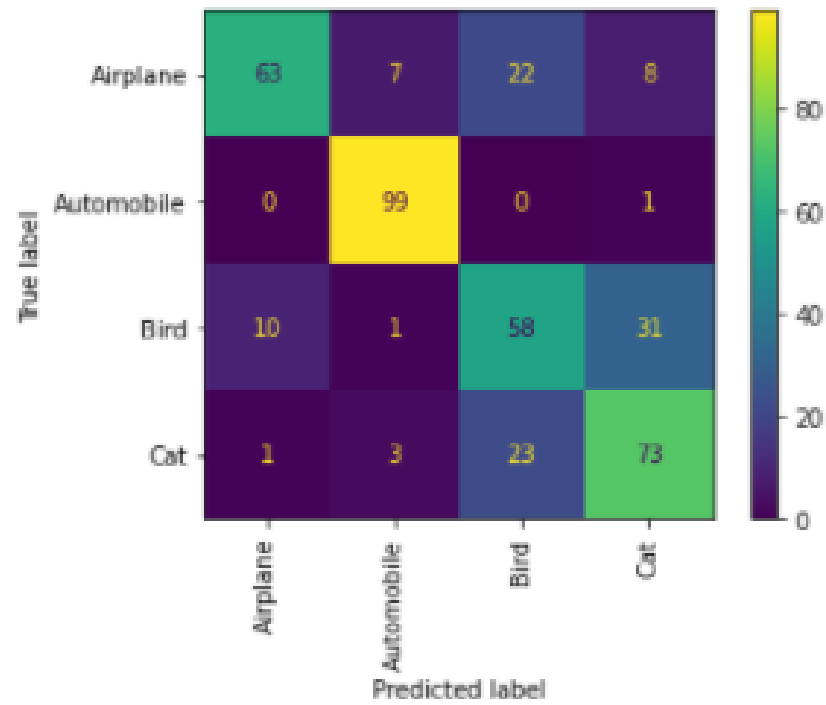
15



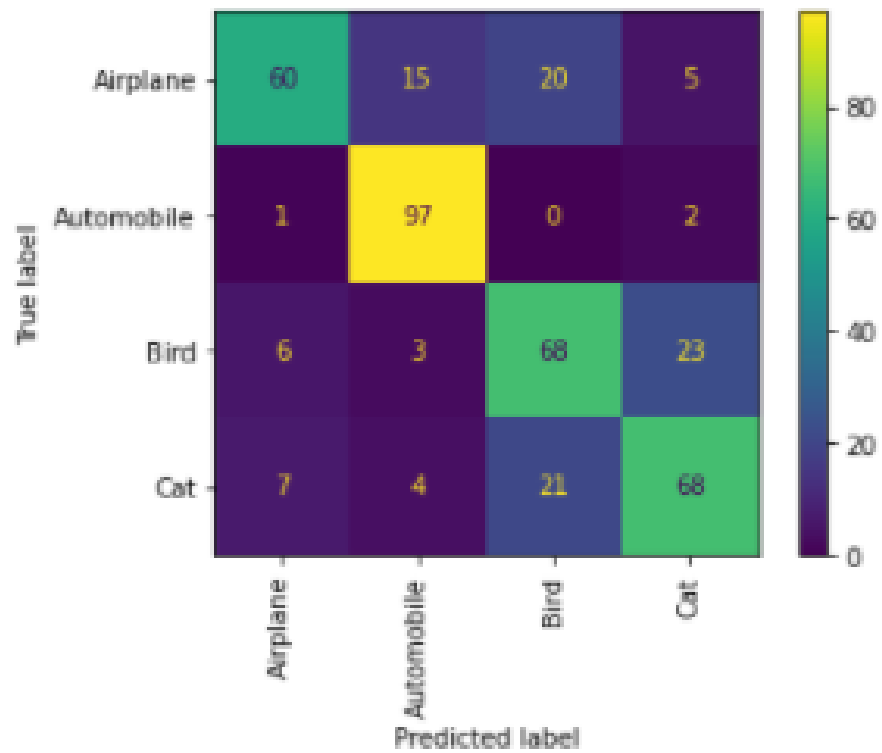
30

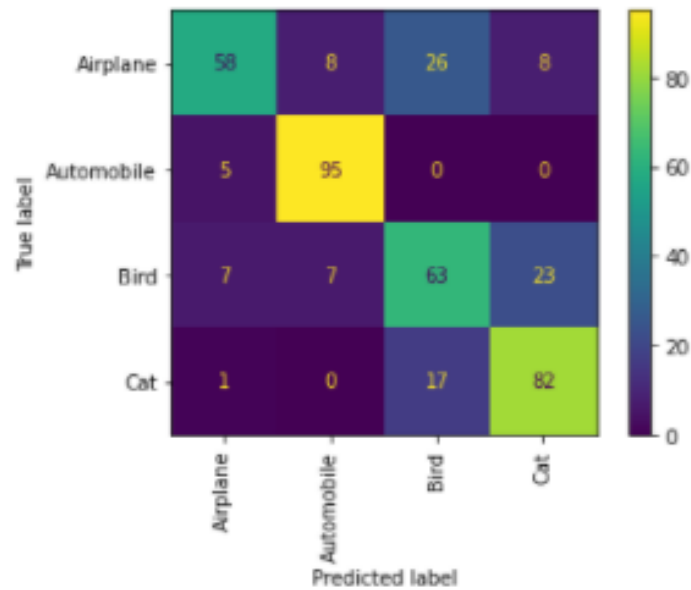


45



60

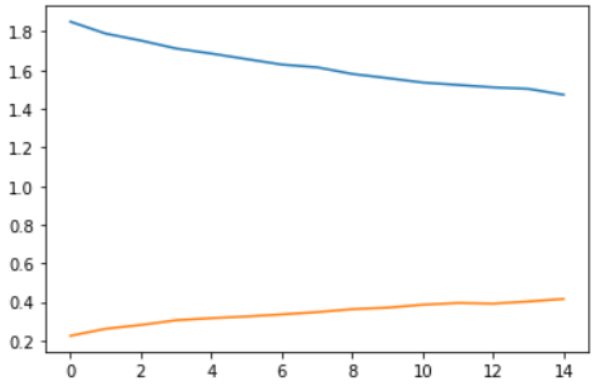
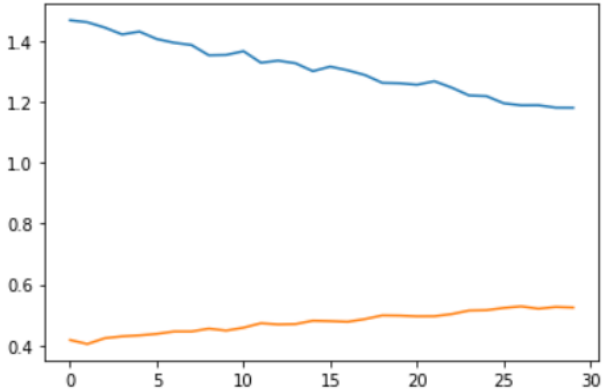
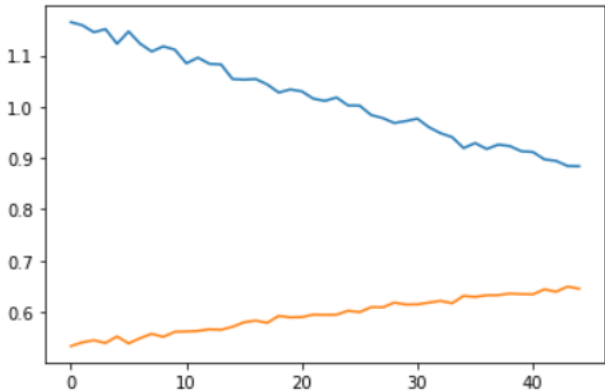


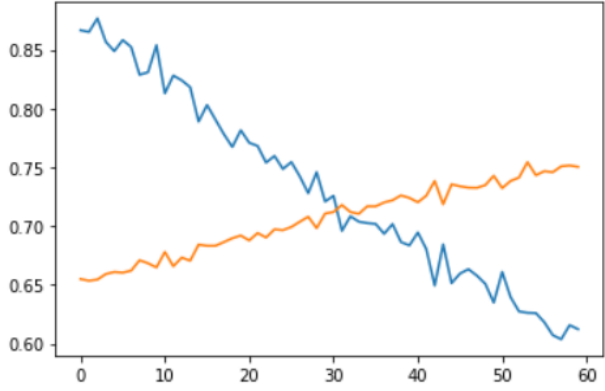
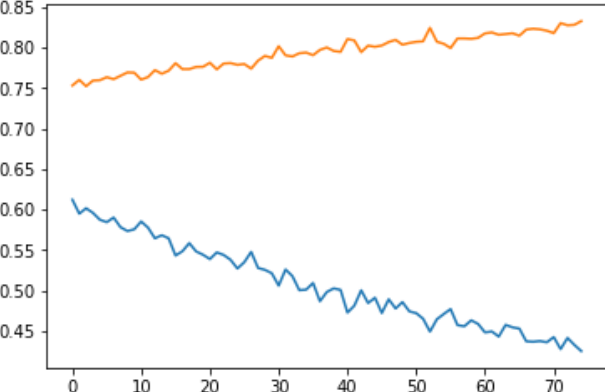


- i. Dari hasil uji coba dengan menggunakan 4 *class* yaitu *airplane*, *automobile*, *bird*, *cat* dapat disimpulkan misalnya dengan *class* yang memiliki 7 *class train* dan 10 *class train* itu pada saat *epoch* ke 15 terlihat ada perbedaan yang cukup signifikan dimana hasil dari *loss* yang semakin naik dengan perbandingan jumlah *class train*. Berpengaruh juga ke table akurasi yang diikuti dengan penurunan seiring naiknya dari jumlah *class* yang ada. Hal ini mengartikan bahwa semakin kecil data *class* yang akan di *train* dan *epoch* yang semakin besar akan menghasilkan akurasi yang baik namun tidak menutup kemungkinan akan terjadi dimana akurasi yang tinggi (*epoch* yang semakin besar) namun hasil prediksi yang lebih rendah dibandingkan percobaan *epoch* sebelumnya yang disebabkan oleh *overfitting*

b. Hasil tes menggunakan variasi 7 kelas

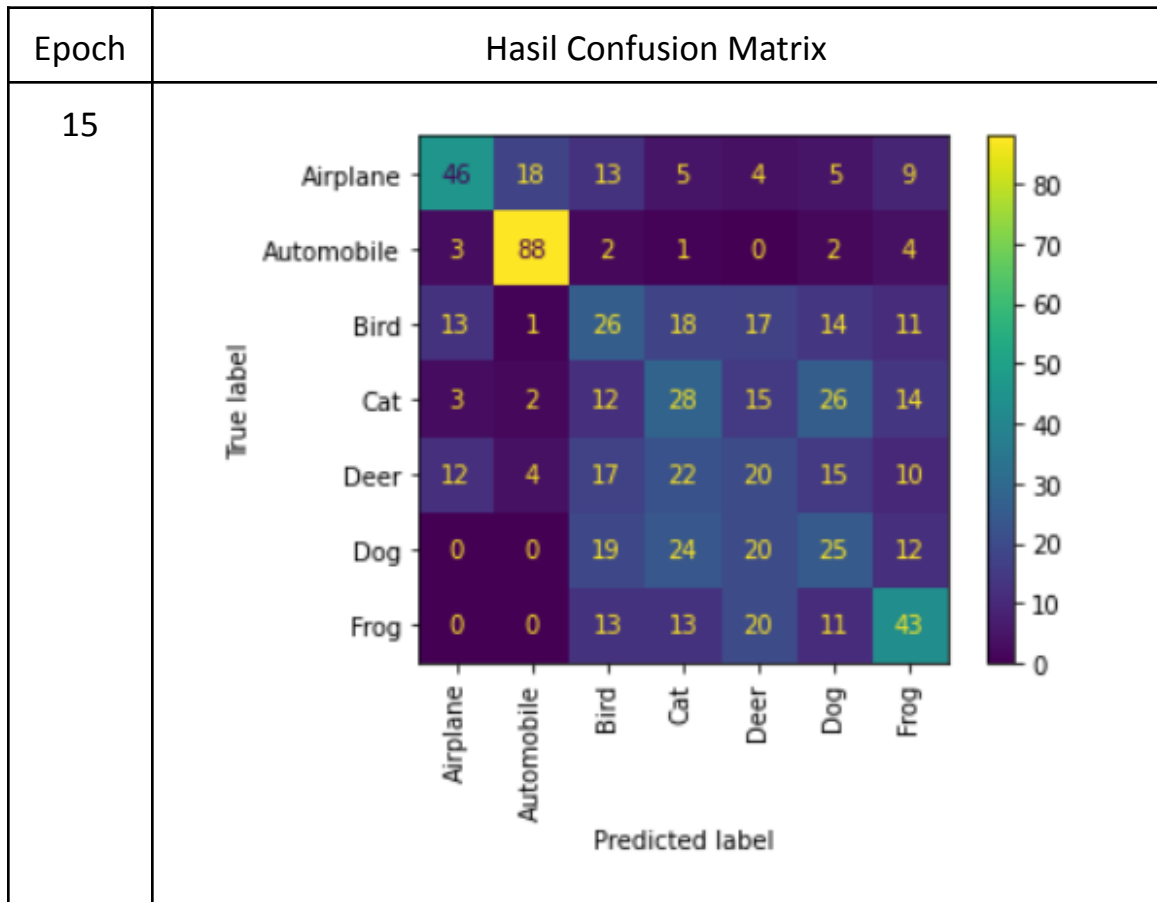
Tabel akurasi dan loss pada epoch:

Epoch	Loss	Akurasi	Graphic
15	1.4724	41.59%	 <p>This line graph shows the training progress over 15 epochs. The x-axis represents epochs from 0 to 14, and the y-axis represents values from 0.2 to 1.8. A blue line representing loss starts at approximately 1.8 and decreases steadily to about 1.5. An orange line representing accuracy starts at approximately 0.2 and increases steadily to about 0.4.</p>
30	1.1810	52.46%	 <p>This line graph shows the training progress over 30 epochs. The x-axis represents epochs from 0 to 30, and the y-axis represents values from 0.4 to 1.4. A blue line representing loss starts at approximately 1.5 and decreases with some fluctuations to about 1.2. An orange line representing accuracy starts at approximately 0.4 and increases with some fluctuations to about 0.55.</p>
45	0.8839	64.51%	 <p>This line graph shows the training progress over 45 epochs. The x-axis represents epochs from 0 to 45, and the y-axis represents values from 0.6 to 1.1. A blue line representing loss starts at approximately 1.1 and decreases with some fluctuations to about 0.9. An orange line representing accuracy starts at approximately 0.55 and increases with some fluctuations to about 0.65.</p>

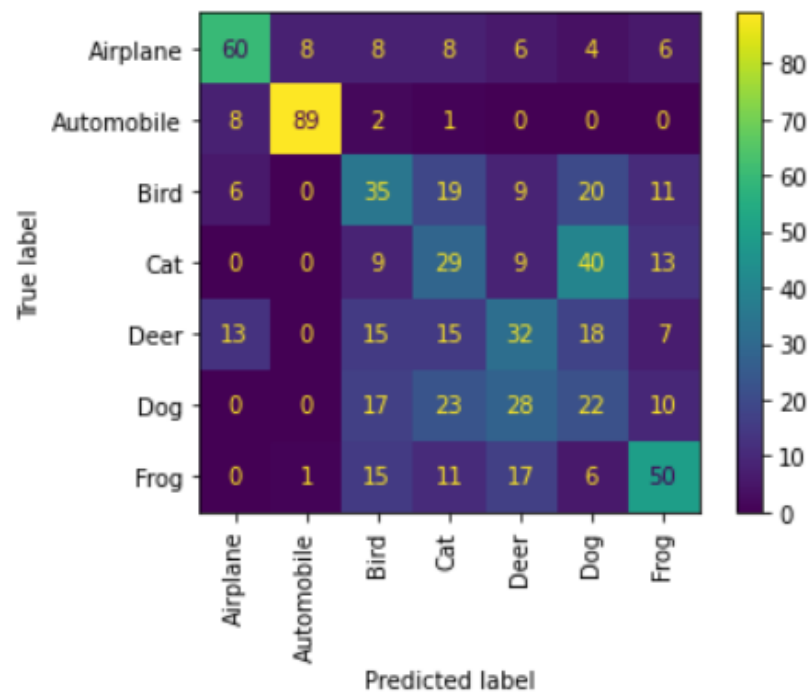
60	0.6124	75.04%	 <p>A line graph with the x-axis ranging from 0 to 60 and the y-axis from 0.60 to 0.85. A blue line representing loss starts at approximately 0.85 and trends downwards with some fluctuations, ending at about 0.61. An orange line representing accuracy starts at approximately 0.65 and trends upwards, ending at about 0.75. The two lines intersect around iteration 30.</p>
75	0.4252	83.30%	 <p>A line graph with the x-axis ranging from 0 to 75 and the y-axis from 0.45 to 0.85. A blue line representing loss starts at approximately 0.61 and trends downwards, ending at about 0.43. An orange line representing accuracy starts at approximately 0.75 and trends upwards, ending at about 0.83. The lines are more distinct than in the first graph, with less overlap.</p>

* garis biru merupakan garis loss dan garis kuning merupakan garis akurasi

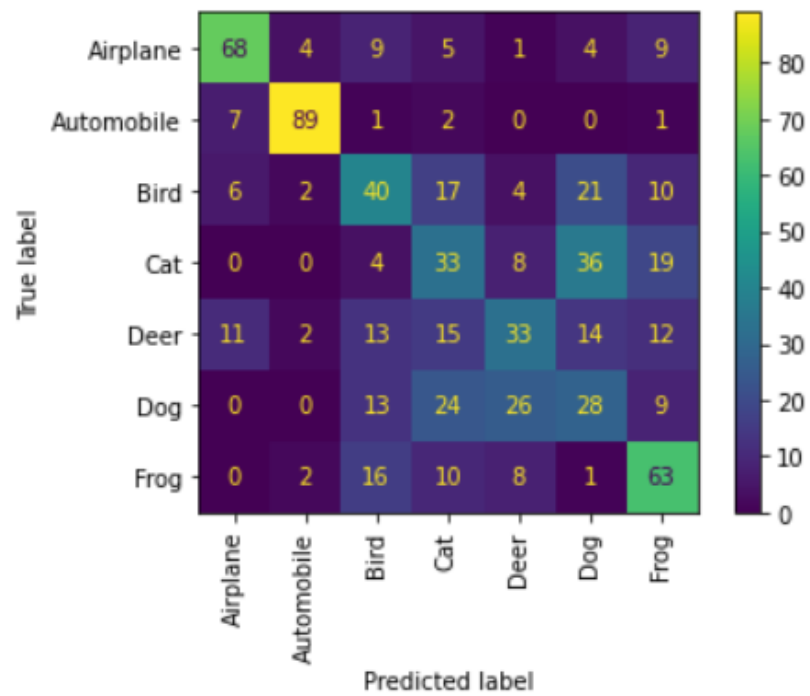
Tabel hasil confusion matrix:



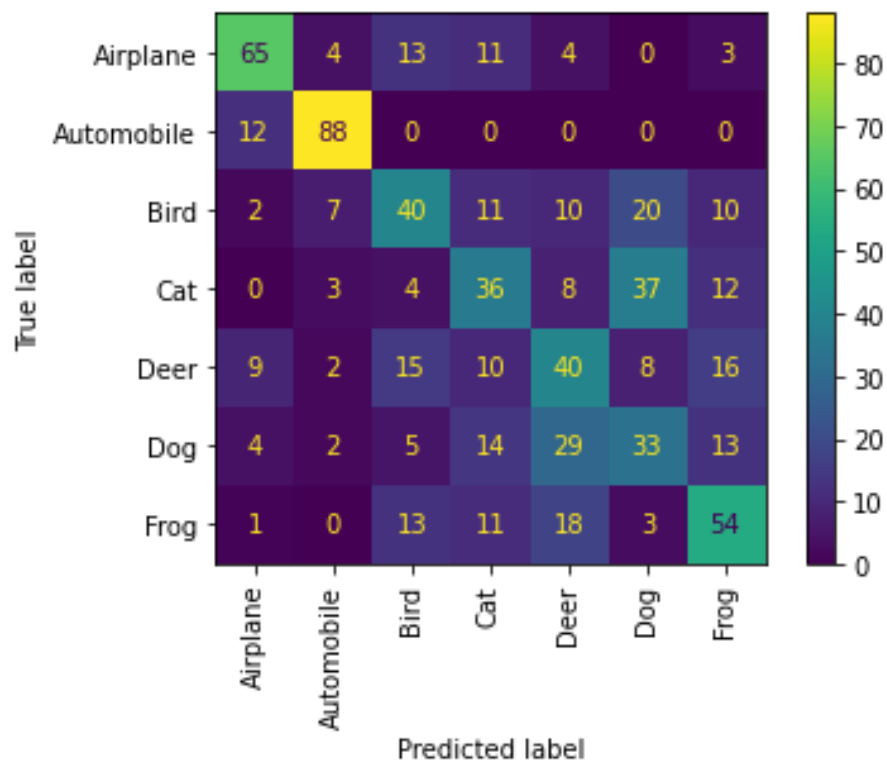
30



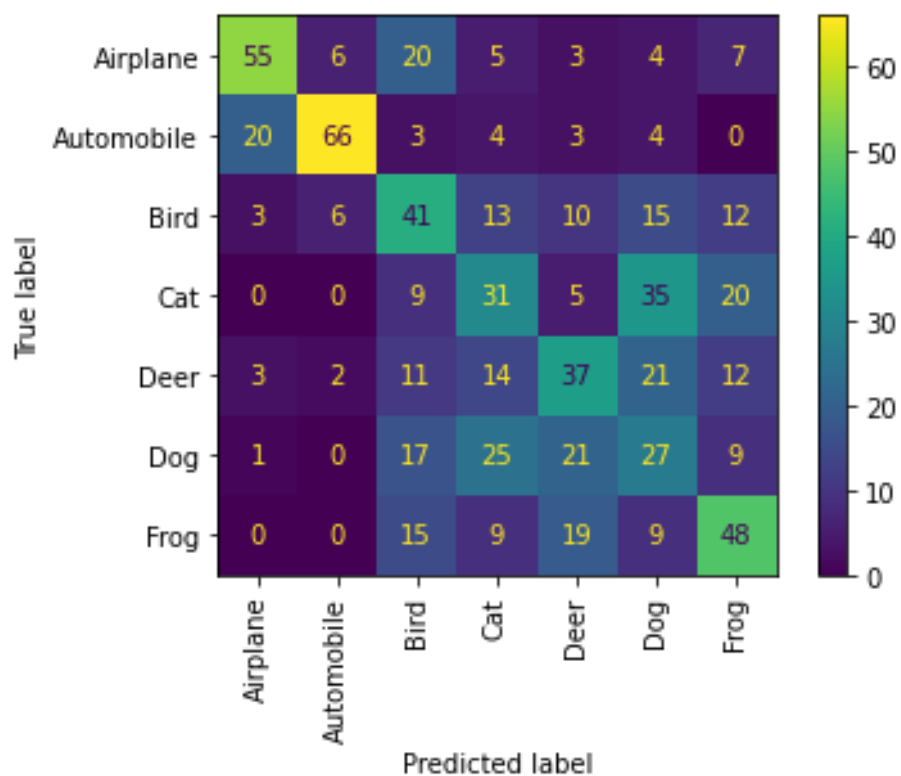
45



60



75

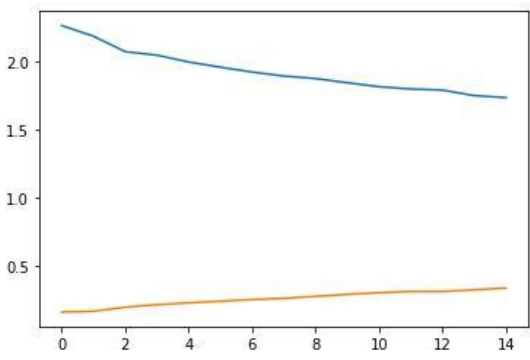
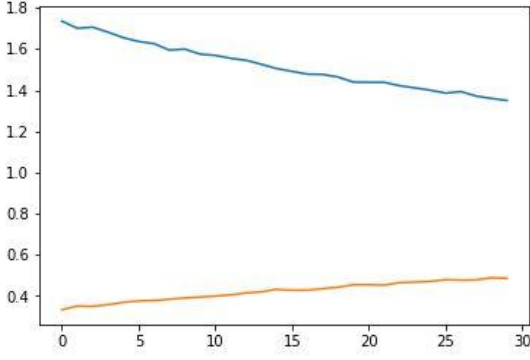
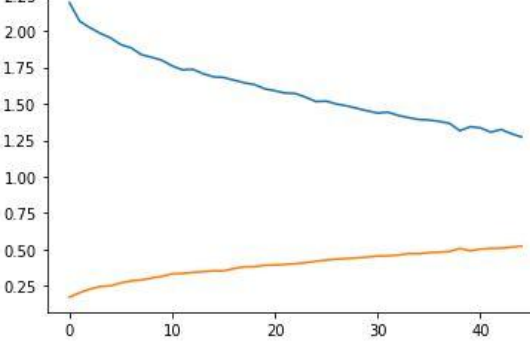


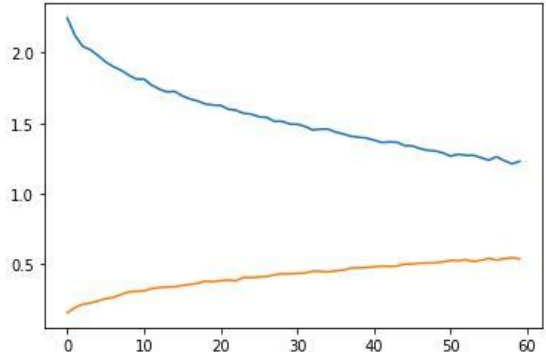
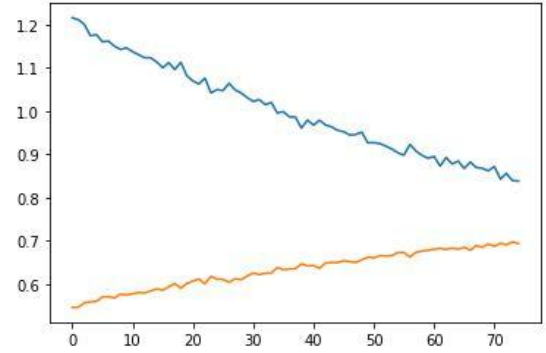
- i. Hasil data diatas menggunakan data *train* 7 kelas. Kelas tersebut adalah *Airplane, Automobile, Bird, Cat, Deer, Dog, dan Frog* dengan dilakukan training menggunakan *epoch* yang berbeda. Terlihat dari perbedaan *epoch*, value *prediction label* terhadap *true* label yang sama memiliki nilai yang semakin tinggi dibandingkan dengan value *prediction label* lainnya. Pada grafik *loss* dan akurasi, terlihat bahwa semakin tinggi *epoch* yang dilakukan, grafik akurasinya semakin naik sedangkan grafik *loss* turun. Tetapi jika dilihat dari grafik *loss* dan akurasi dari jumlah kelas yang lebih kecil dari 7, grafik akurasi pada jumlah kelas dibawah 7 lebih tinggi pada *epoch* yang lebih sedikit daripada grafik akurasi pada jumlah kelas 7 dengan jumlah *epoch* yang sama. Hal ini dikarenakan bahwa jumlah data *training* yang dimiliki berbeda. Pada jumlah kelas 7 memiliki jumlah data *training* yang lebih besar daripada jumlah data *training* pada jumlah kelas 4.

Selain itu, ada titik tertentu (jumlah *epoch* tertentu) yang dimana akurasi pada *epoch* tersebut tinggi, tetapi pada hasil *prediction label* terhadap *true* label value nya yang sama memiliki jumlah yang berbeda dari hasil *confusion matrix* dari *epoch* sebelumnya. Ini karena adanya penyebab *overfitting*.

c. Tes hasil apabila kelas yang diuji adalah 10 kelas

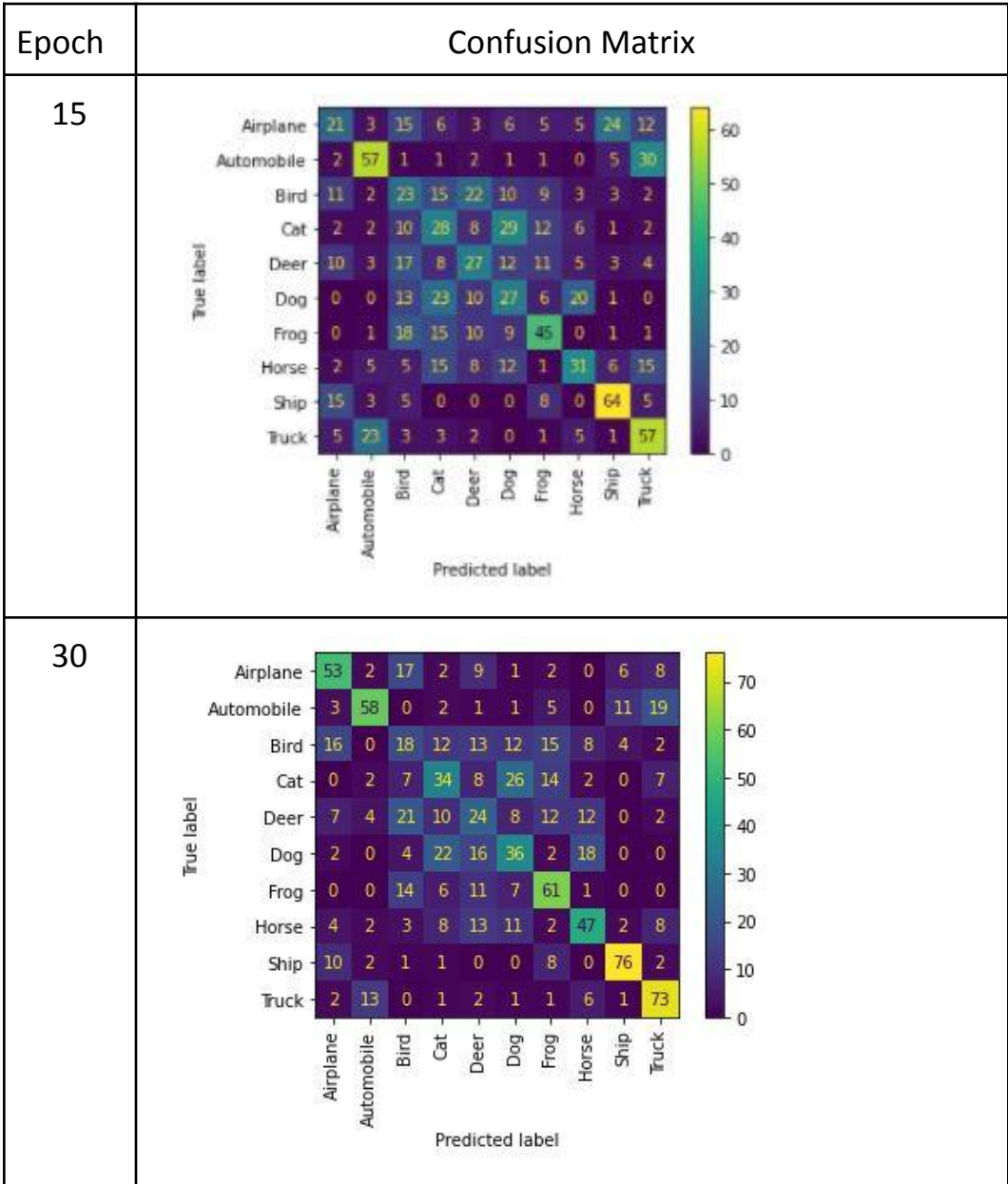
Tabel hasil akurasi dan loss:

Epoch	Loss	Accuracy	Graphic
15	1.7384	33,75%	
20	1.3882	50,51%	
45	1.2712	52,24%	

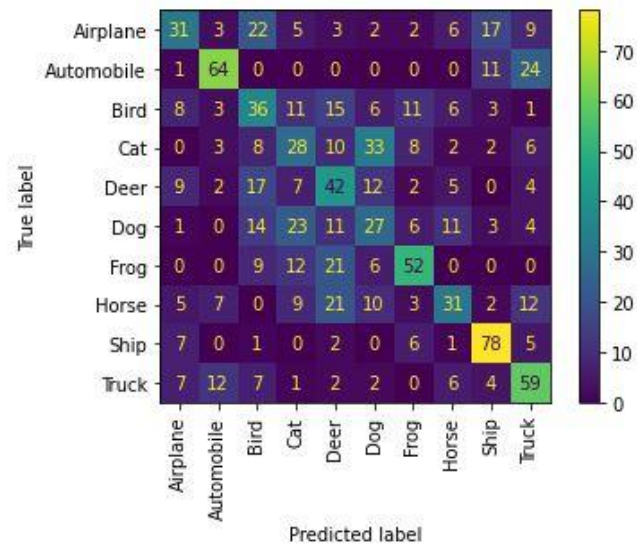
60	1.2303	53,89%	
75	0.8380	69,40%	

* garis biru merupakan garis loss dan garis kuning merupakan garis akurasi

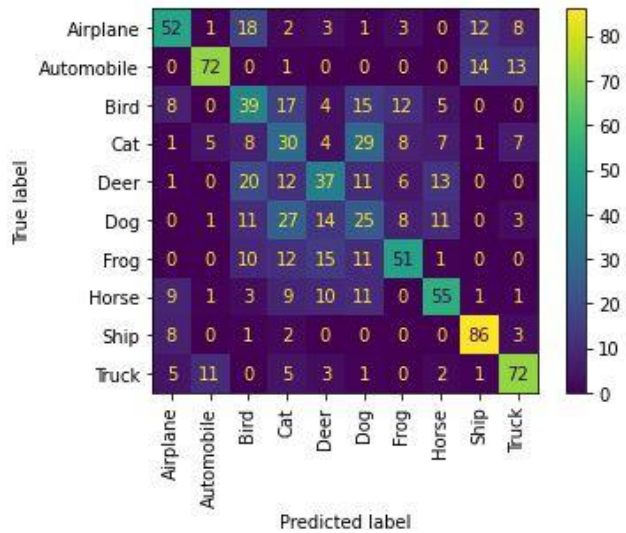
Tabel confusion matrix:



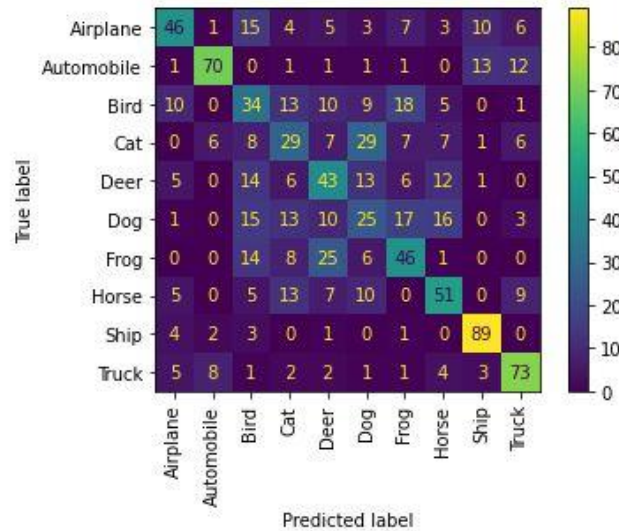
45



60



75



I. Pengujian ini menggunakan 10 class (semua class yang ada). Hasilnya berupa *confusion matrix* yang terlampir. Dari warna tiap bloknya bisa dilihat, semakin ungu semakin sedikit gambar yang dilabeli oleh label tersebut, semakin kuning semakin banyak gambar yang dilabeli. Hasil *confusion matrix* antar *epoch* bervariasi. Terkadang dengan *epoch* lebih kecil lebih banyak memiliki *prediction label* yang benar daripada dengan *epoch* yang besar di class tertentu seperti *airplane*, *deer*, *dog*. Karena jumlah *class* dari data yang di *train* memang lebih banyak, itu mengapa dengan *epoch* sekian belum bisa sama seperti dengan 4 *class* dan 7 *class*, dimana rata-rata di *epoch* 60 dan 75, grafik *accuracy* sudah di atas grafik *loss*.

1. Tabel Hasil Rangkuman Loss

Jumlah Kelas	Epoch				
	15	30	45	60	75
4	0.92	0.67	0.5978	0.5146	0.3344
7	1.4724	1.1810	0.8839	0.6124	0.4252
10	1.7384	1.3882	1.2712	1.2303	0.8380

2. Tabel Hasil Rangkuman Accuracy

Jumlah Kelas	Epoch				
	15	30	45	60	75
4	59.25%	71.72%	75.67%	79.00%	87.37%
7	41.59%	52.46%	64.51%	75.04%	83.30%
10	33,75%	50,51%	52,24%	53,89%	69,40%

4. Kesimpulan

Dari percobaan diatas, kami menyimpulkan bahwa semakin besar epoch yang dipakai maka semakin tinggi juga akurasi prediksi labelnya. Namun dalam kondisi spesifik, hasil prediksi tidak sebanding dengan akurasinya. Akurasinya tinggi, namun pada kenyataannya, hasil prediksi gambar menurun dari *epoch* sebelumnya. Hal ini disebut dengan *Overfitting*, dimana model yang dibuat tidak bisa secara benar melakukan prediksi karena ia menangkap data "noise", yang mana seharusnya tidak ditangkap. Selain overfitting, jumlah class data yang di train juga mempengaruhi hasil accuracy. Semakin banyak jumlah class data yang di train, epochnya juga harus semakin besar supaya loss semakin kecil dan akurasi semakin besar.