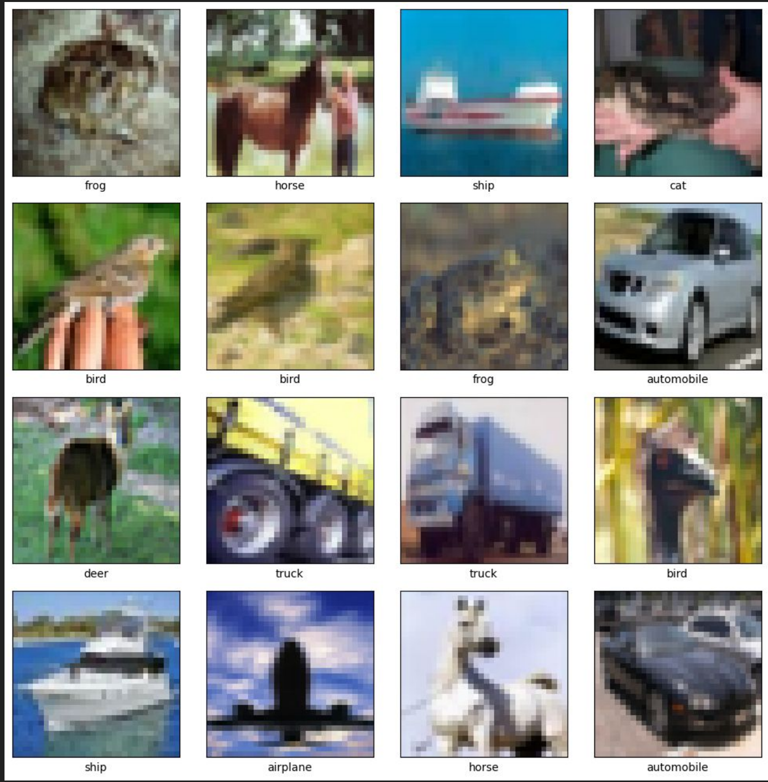
Computer Vision

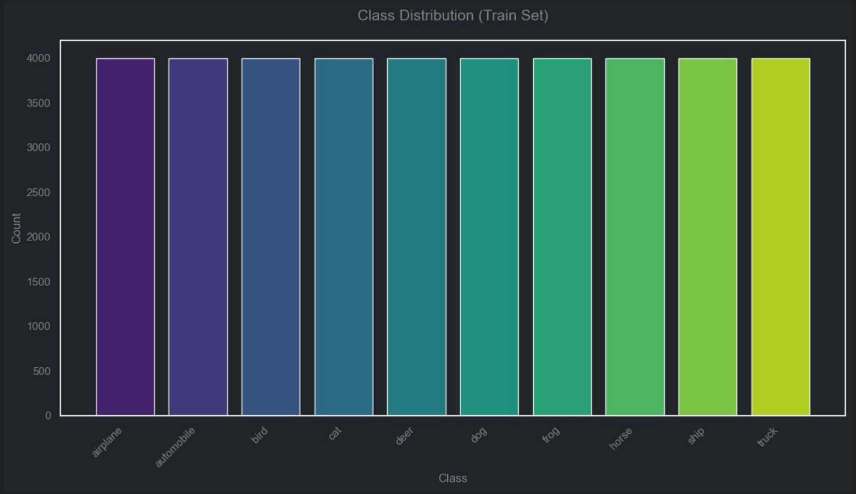
Alexandre, Janete, Isis

Feb 2026

CIFAR10 small images classification dataset



- [cifar10 dataset from keras]()
- 60,000 32x32 colour images in 10 classes
  - training: 50,000 images
  - test: 10,000 images
- 6,000 images per class



Data Preparation

data splitting:
cifar10 is pre-split into 50K train and 10K test images, but we adjusted to

- 40,000 train
- 10,000 validation
- 10,000 test

Images

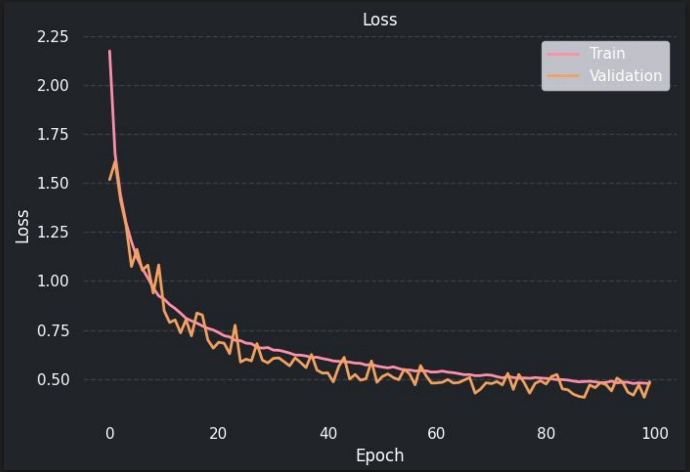
- Normalization

Target

- One-Hot-Encoding

CNN - base model

- 3 blocks
  - Conv2D 32 (x2) ; 64 (x2) ; 128
  - activation “relu”
  - Dropouts (0.25,0.25,0.4)
- Classifier
  - Dense 256
  - Dropout 0.5
  - activation “softmax”
- Compiler:
  - adam, with adjusted learning rate
  - loss="sparse_categorical_crossentropy"
- Early Stop:
  - monitor val_loss
  - patience = 10
- ReduceLROnPlateau:
  - monitor val_loss
  - patience = 5



Confusion Matrixes CNN Models

Confusion Matrix (CNN Base)

True \ Predicted	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	881	12	17	2	3	0	11	4	44	26
automobile	4	953	0	0	0	0	2	0	4	37
bird	49	2	771	14	30	28	86	11	7	2
cat	26	4	34	614	45	107	113	19	17	21
deer	11	1	30	15	836	11	76	15	3	2
dog	7	7	22	67	28	795	37	26	2	9
frog	7	0	10	8	3	2	965	1	3	1
horse	16	2	11	8	34	20	14	887	2	6
ship	24	8	2	1	0	0	3	1	949	12
truck	17	29	2	0	1	1	1	2	13	934

Confusion Matrix - Custom VGG Model

True \ Predicted	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	871	10	17	3	10	0	5	5	42	37
automobile	3	960	0	0	0	0	0	0	5	32
bird	22	2	830	12	27	30	51	9	2	15
cat	5	7	24	703	37	80	85	16	15	28
deer	4	3	24	16	860	12	54	21	3	3
dog	5	5	13	90	28	793	32	21	2	11
frog	2	2	14	8	3	1	966	1	1	2
horse	6	3	5	17	12	13	9	921	3	11
ship	18	9	0	2	1	1	5	0	942	22
truck	4	32	1	2	0	1	1	0	4	955

Transfer Learning: EfficientNetB0 (04 Model)

EfficientNetB0

Adjustments:
- resized images to (96,96)
- randomly adjusted brightness, contrast, hue, saturation, random crop
- ‘unnormalised’ data, since image normalisation happens inside EfficientNet

EfficientNetB0 - Layers Unfrozen

Adjustments:
- same as before
- Unfroze half of the layers

Transferlearning: EfficientNetV2B1 (Model 06)

EfficientNetV2B1 

Adjustments:
- resized images to (96,96)
- randomly adjusted brightness, contrast, hue, saturation, random crop
- ‘unnormalised’ data, since image normalisation happens inside EfficientNet

EfficientNetV2B1 - Layers Unfrozen

Adjustments:
- same as before
- Unfroze half of the layers

Transferlearning: EfficientNetV2S (Model 07)

PyTorch

Adjustments:

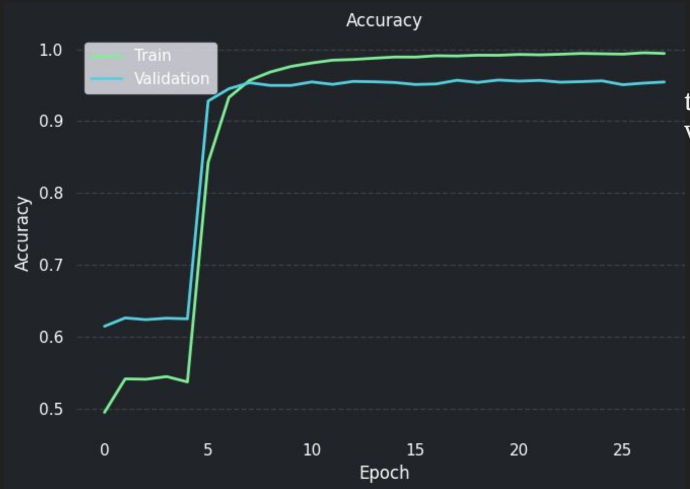
- using Pytorch library instead of keras
- same as 04 model
- Normalized by the mean and std

PyTorch - Layers Unfrozen

Adjustments:

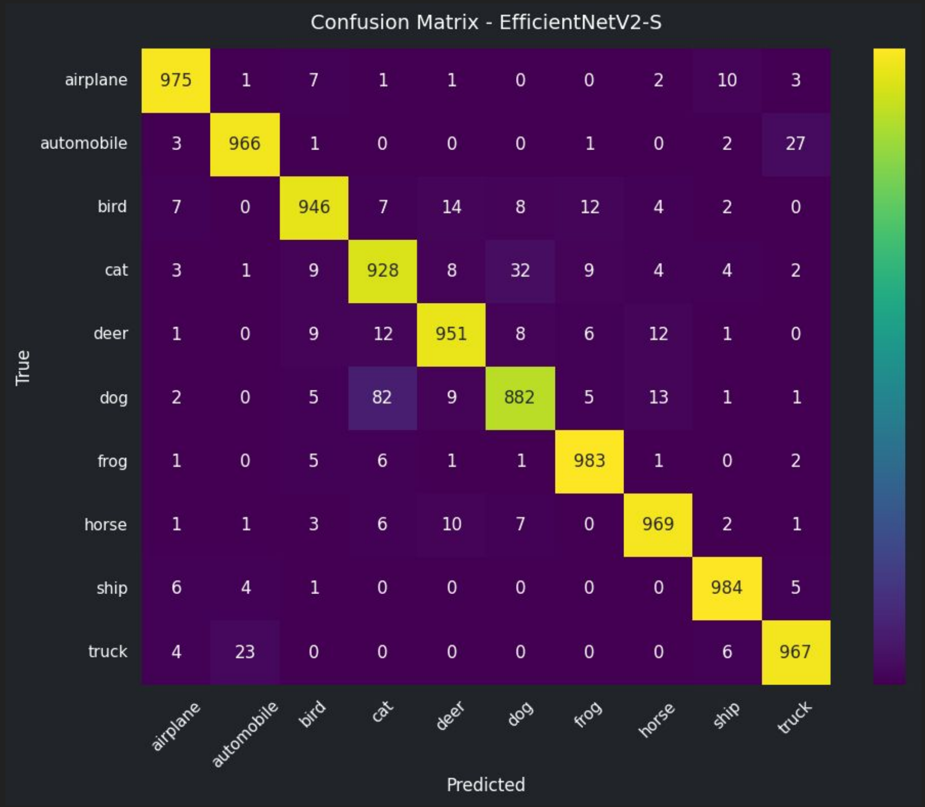
- Unfroze half of the layers
- Smaller learning rate and label smoothing
- Increase the patience

Best Model: PyTorch: Performance

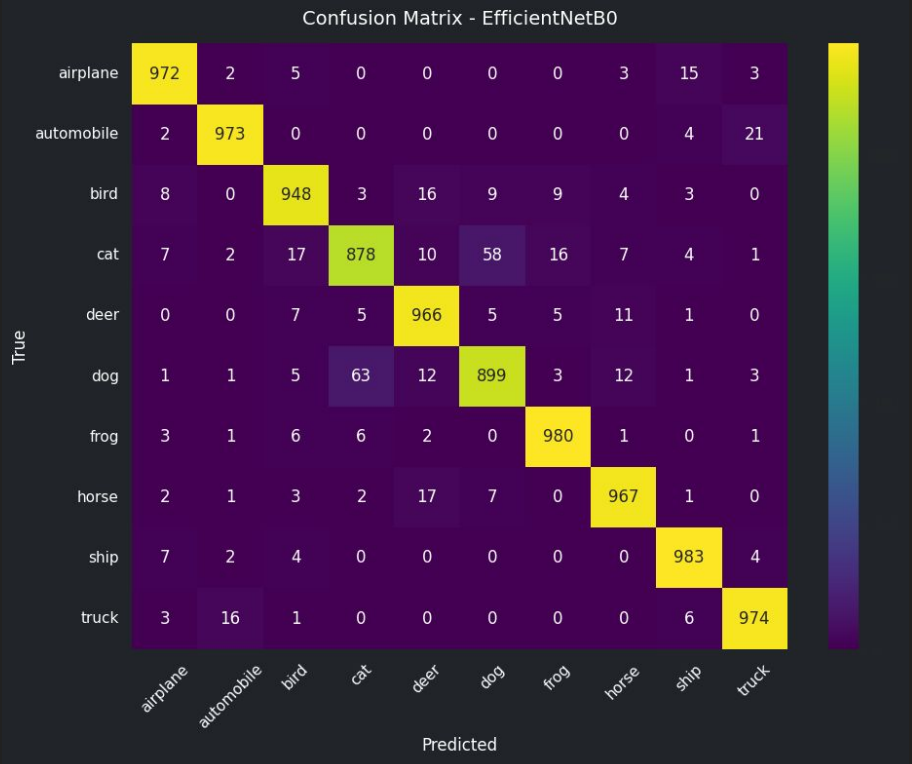


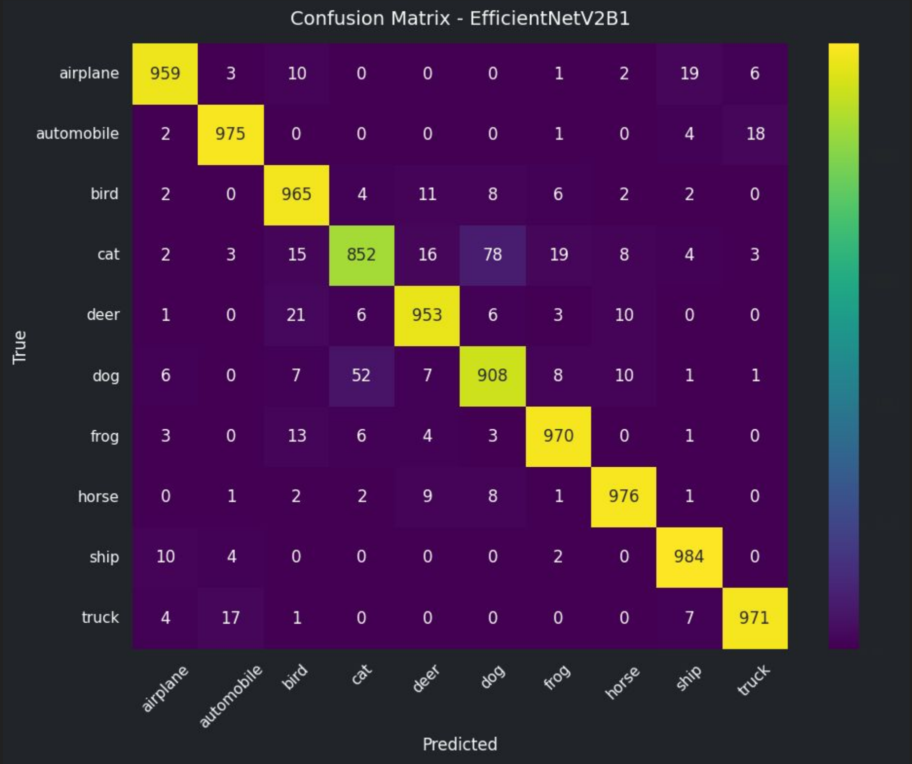
train_acc=0.9941
val_acc=0.9545

train_loss=0.3010
val_loss=0.4271

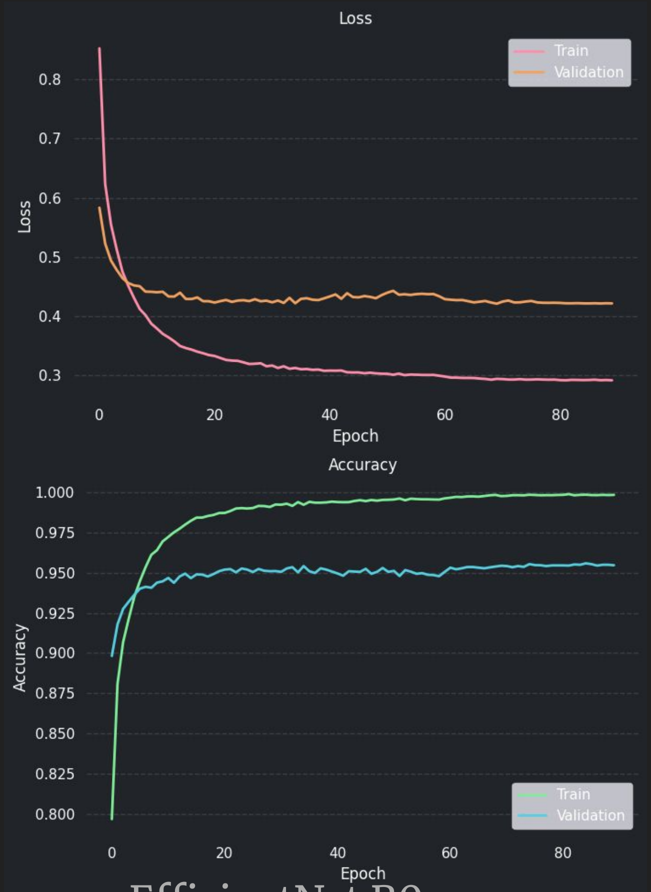


Transfer Learning: Confusion Matrixes





Transfer Learning Curves



EfficientNet B0

EfficientNet V2B1

Overview: Adjustments made on...

Images:

- rotation
- adjusting saturation / hue / brightness
- cutting random holes into the images
- flipping
- random crop

Model Architecture:

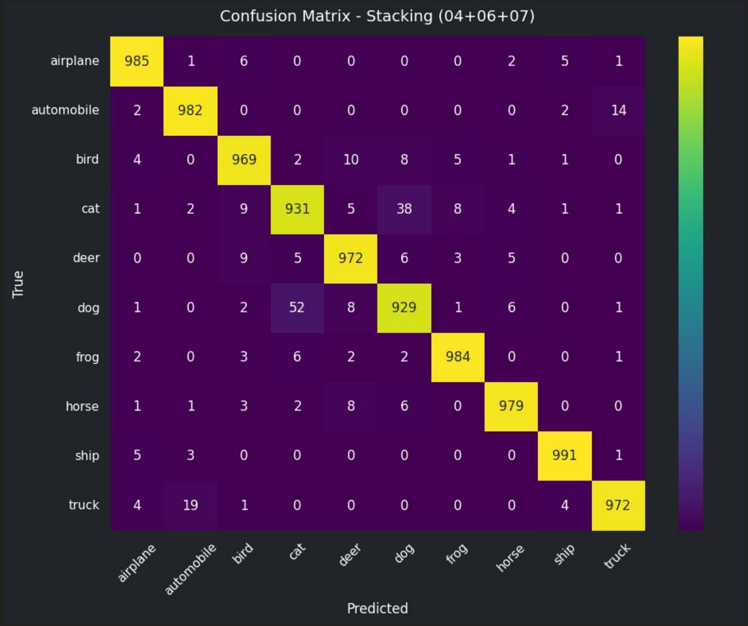
- add layers
- batch normalisation
- replicating the architecture of the VGGBN model on our CNN model

Model Training

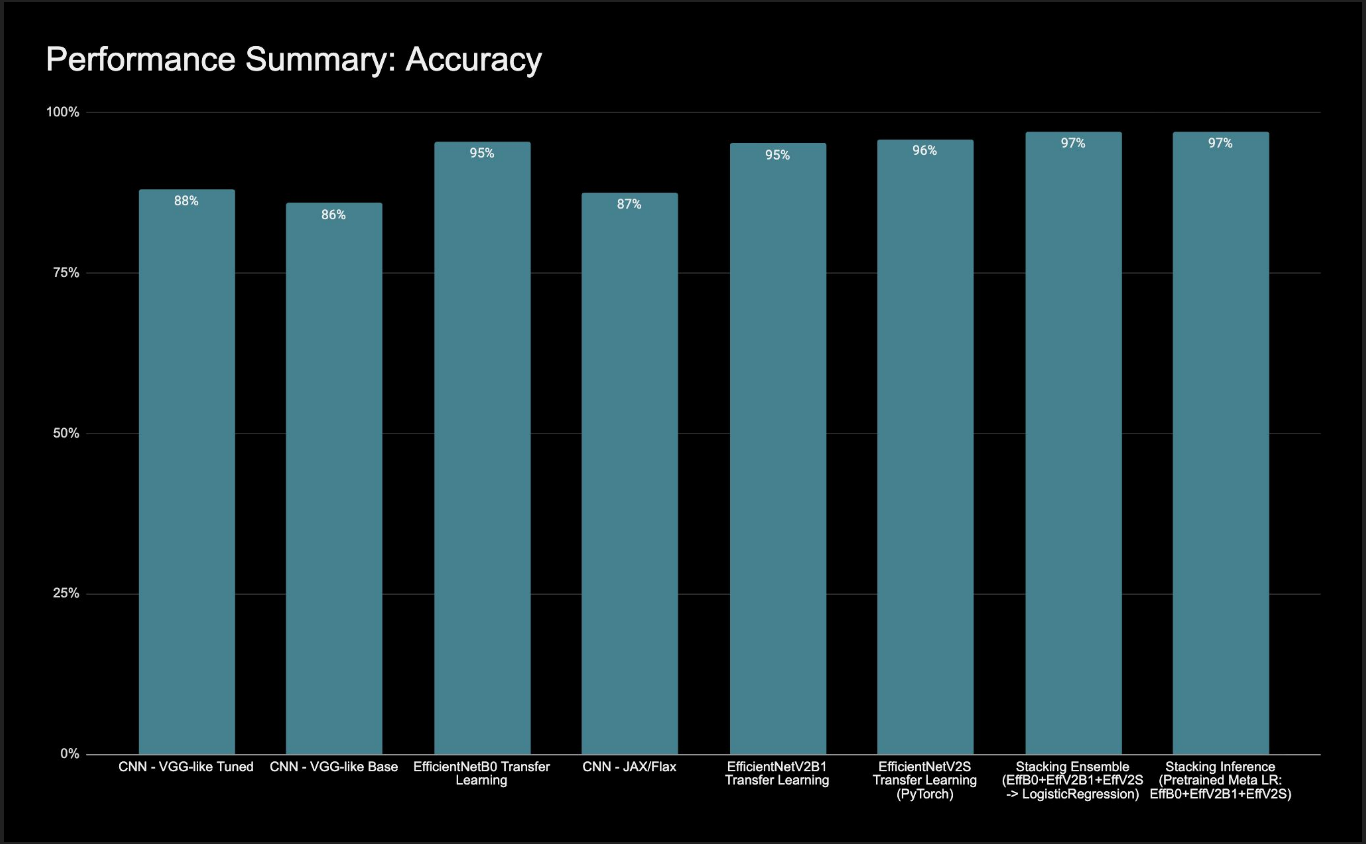
- learning rate scheduler:
  - reduces learning rate as you progress
- label smoothing:
  - accepts less confident predictions, improving performance

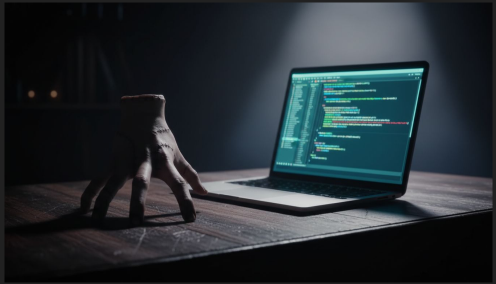
Very Best Model: Franceshina Model

- Stacking Models 04 + 06 + 07
- using a logistic regression to stack these models
- F1 Score: 96.94%



Evaluation Overview





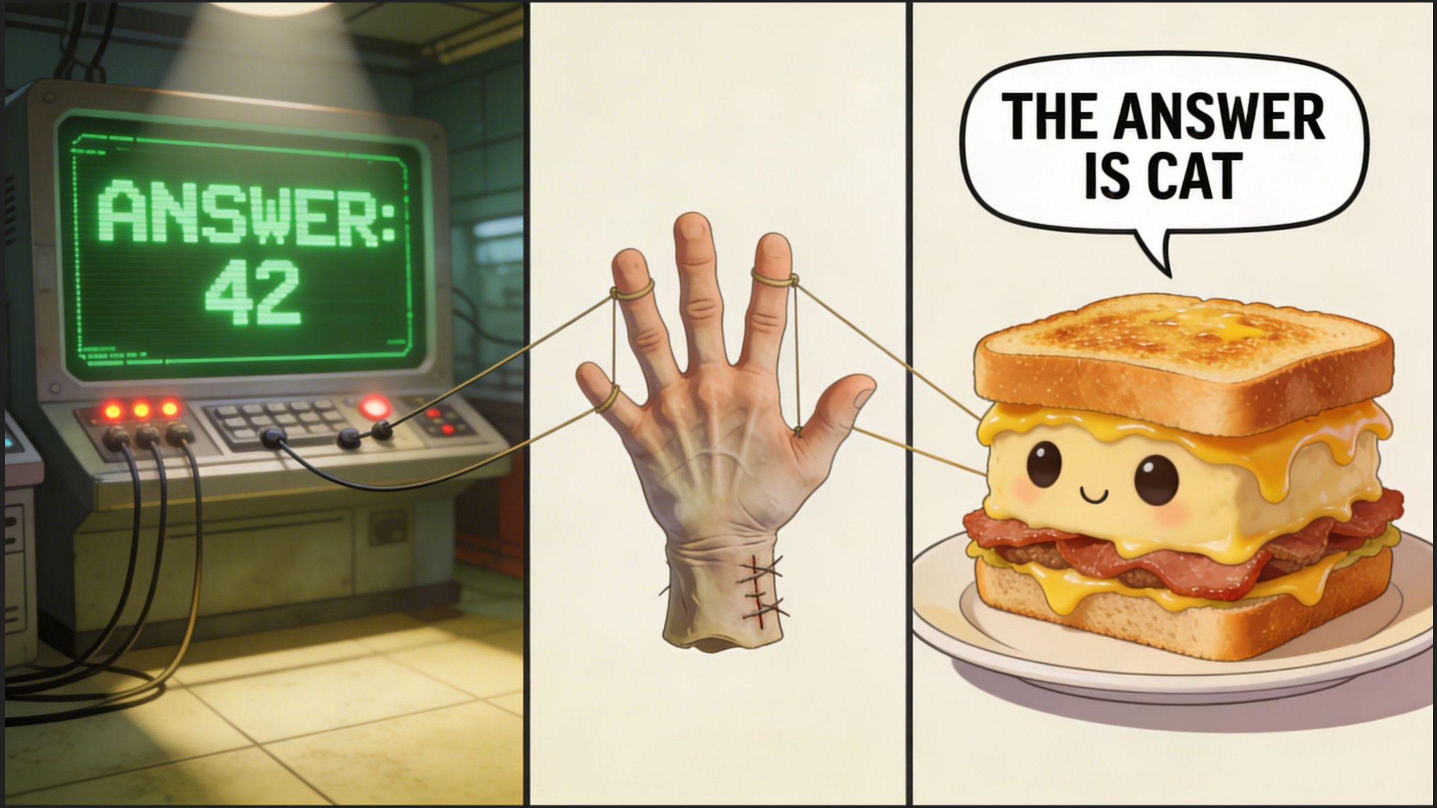
Conclusion

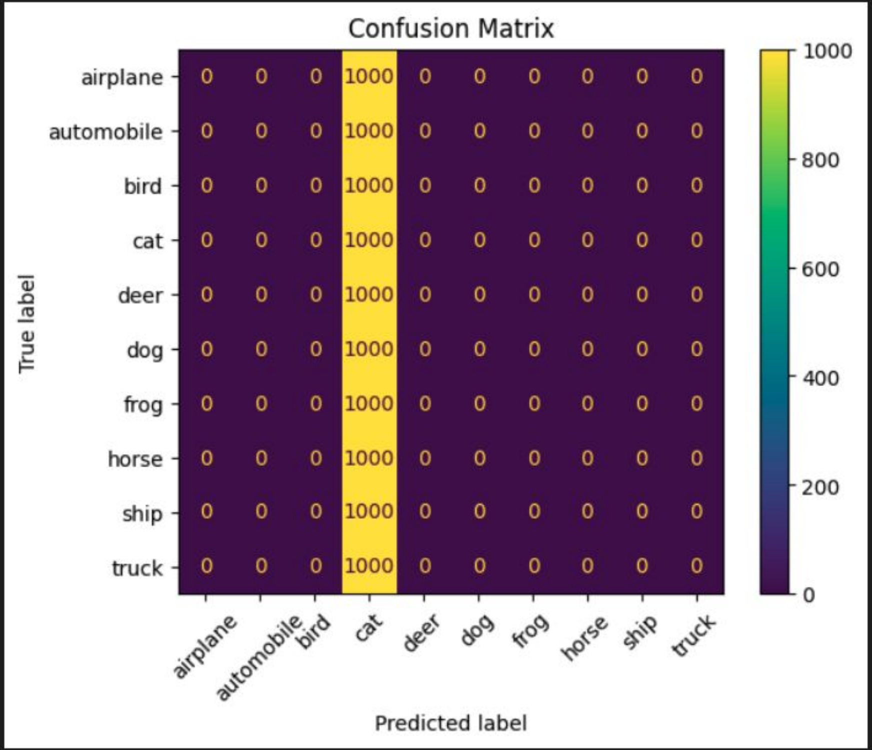
- It helps to challenge your model by manipulating the data input: think of it as giving your model the really harsh and difficult physics teacher → the model is going to do really well on the test
- Pre-trained models achieve better results, quicker, but have more overfitting.

Recommendations & Learnings

- Saving and Evaluating
  - Take some time together to think about how to save your models, and evaluations
- Create multiple notebooks to keep a better overview over your code
- Take a step back if it gets frustrating
- Pick a job that doesn’t require computer vision if you hate it

What we learned?





Questions?

Yes, everything *IS* cats (not 42).

Early stopping…

train_time_sec
35802.59
3620.99
6744.72

triggered by Google Colab.

Recommendations & Learnings

- Saving and Evaluating
  - Take some time together to think about how to save your models, and evaluations
- Create multiple notebooks to keep a better overview over your code
- Take a step back if it gets frustrating
- Pick a job that doesn’t require computer vision if you hate it