Group 3 week 3

Adrian, Cristina, Kira, and Mariana

Overview—Executive summary

- CIFAR-10 dataset
- Winning model:
  - Transfer Learning with VGG16 as base model
  - 86% accuracy on testing dataset

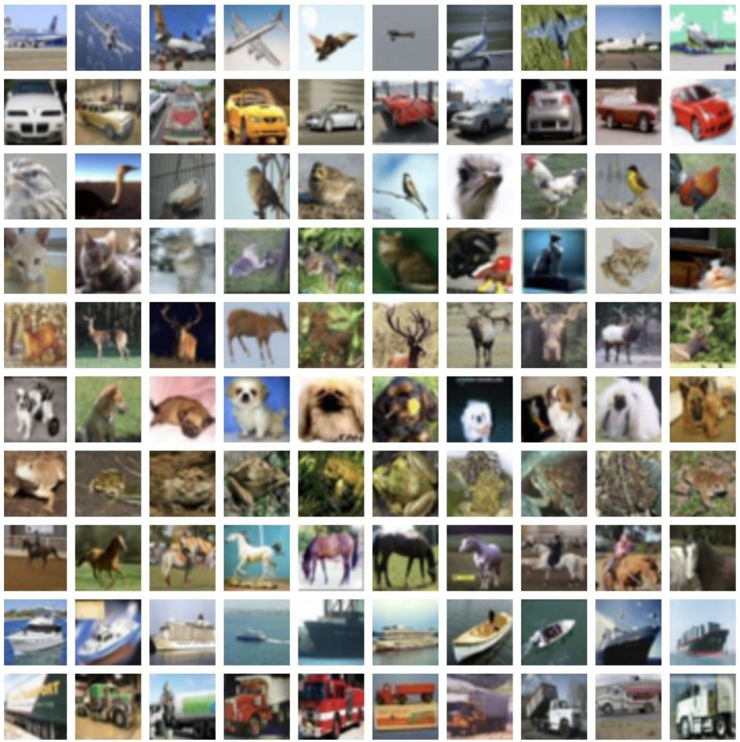
Dataset chosen

CIFAR-10 dataset:

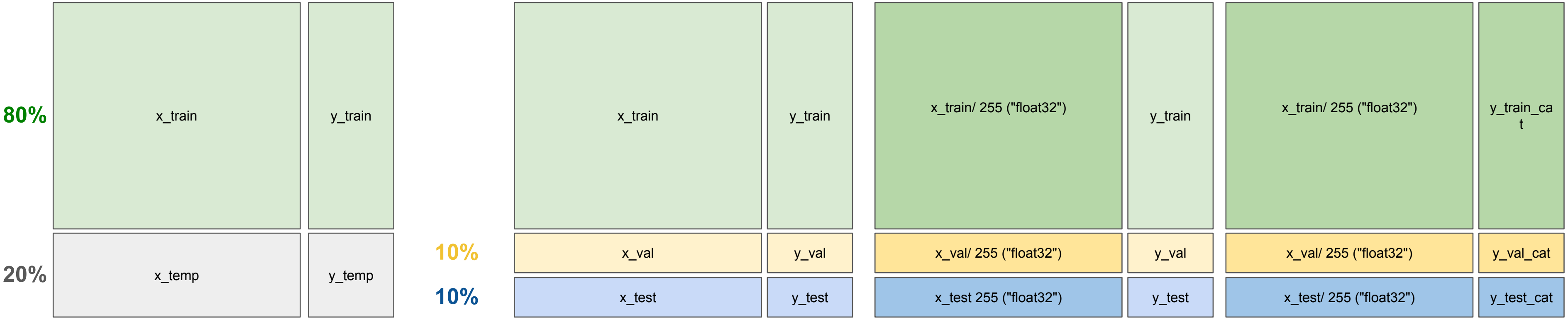
60,000 images (32×32 pixels, RGB, 10 classes like airplane, car, cat, dog).

Why?

- Familiarity: we already used it in a lab.
- Size: This is a bigger dataset (more training data).
- Standardization: all images in the same shape and pixel scale [0, 255]



Data Preprocessing



80%

20%

x_train

y_train

x_temp

y_temp

10%

10%

x_train

y_train

x_val

y_val

x_test

y_test

x_train/ 255 ("float32")

y_train

x_val/ 255 ("float32")

y_val

x_test 255 ("float32")

y_test

x_train/ 255 ("float32")

y_train_cat

x_val/ 255 ("float32")

y_val_cat

x_test/ 255 ("float32")

y_test_cat

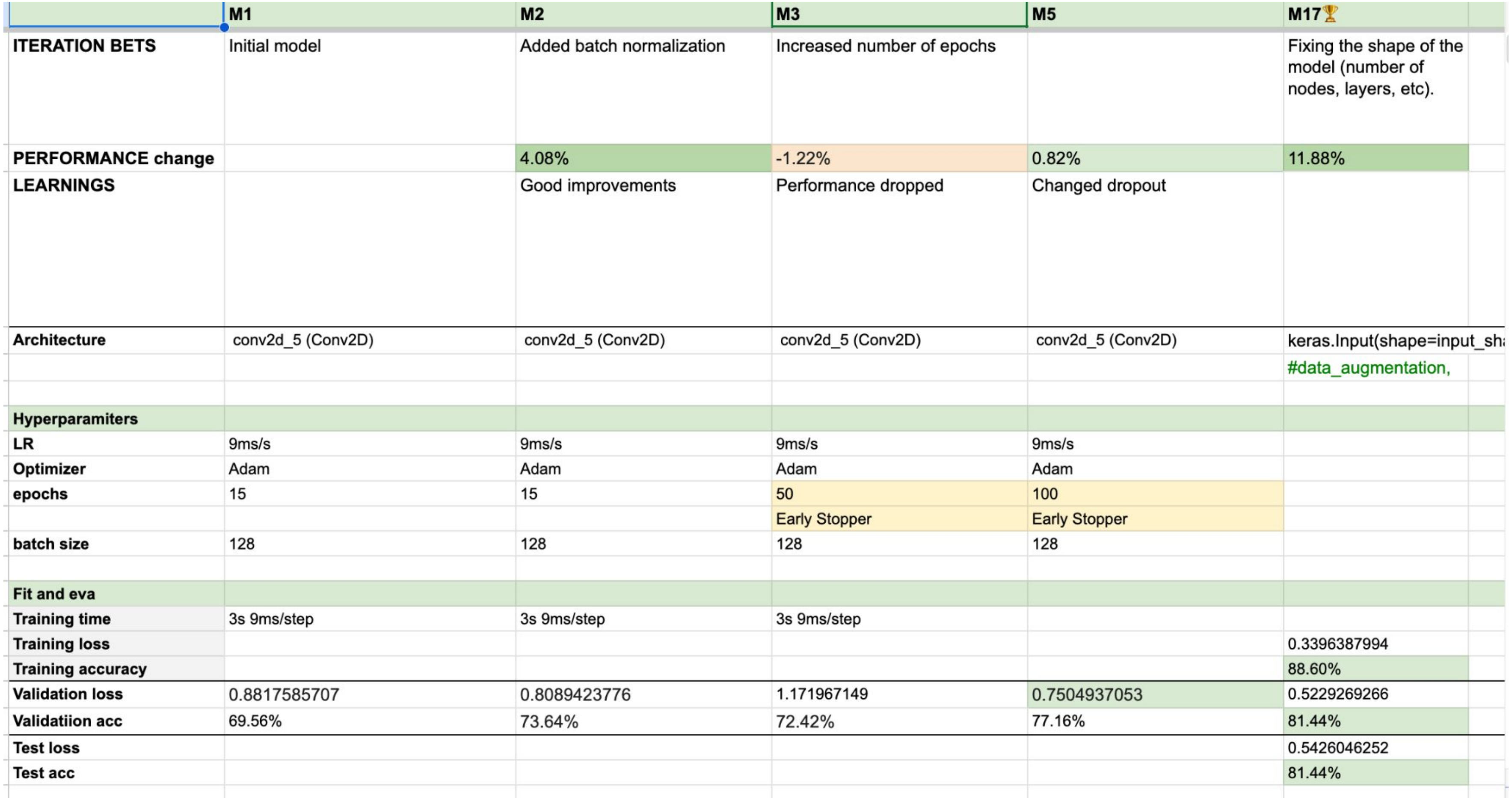
Original data from the dataset: already split into 2 groups.

Splitting data to keep some unused "real" examples

Scale images to the [0, 1]

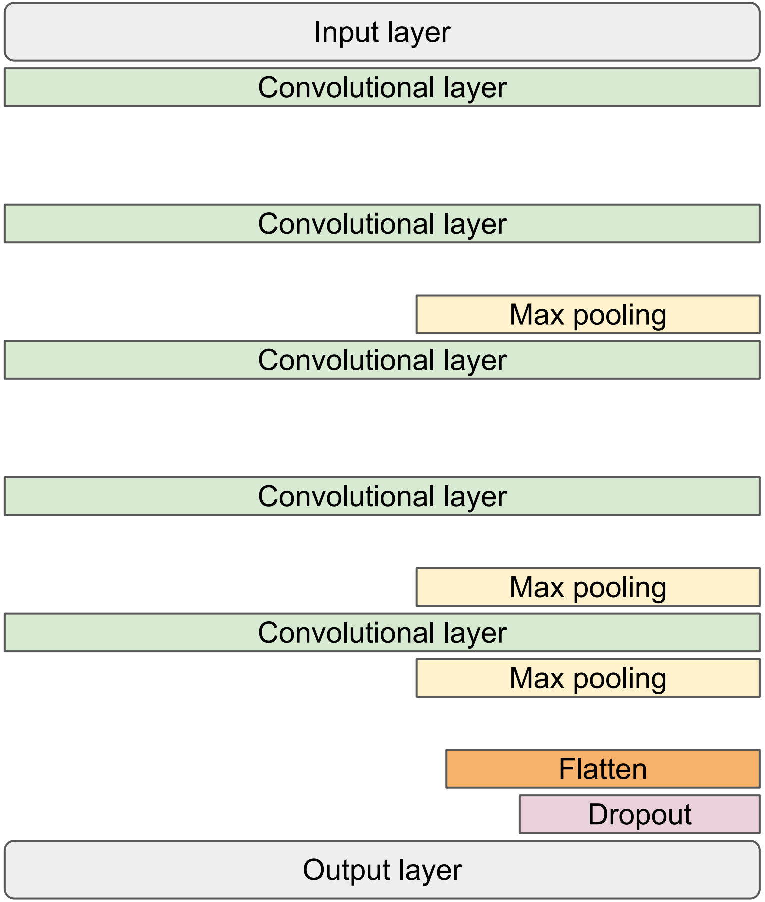
convert classes into categories

Learning Process



	M1	M2	M3	M5	M17🏆
ITERATION BETS	Initial model	Added batch normalization	Increased number of epochs		Fixing the shape of the model (number of nodes, layers, etc).
PERFORMANCE change		4.08%	-1.22%	0.82%	11.88%
LEARNINGS		Good improvements	Performance dropped	Changed dropout	
Architecture	conv2d_5 (Conv2D)	conv2d_5 (Conv2D)	conv2d_5 (Conv2D)	conv2d_5 (Conv2D)	keras.Input(shape=input_sha
					#data_augmentation,
Hyperparamiters					
LR	9ms/s	9ms/s	9ms/s	9ms/s	
Optimizer	Adam	Adam	Adam	Adam	
epochs	15	15	50	100	
			Early Stopper	Early Stopper	
batch size	128	128	128	128	
Fit and eva					
Training time	3s 9ms/step	3s 9ms/step	3s 9ms/step		
Training loss					0.3396387994
Training accuracy					88.60%
Validation loss	0.8817585707	0.8089423776	1.171967149	0.7504937053	0.5229269266
Validatiion acc	69.56%	73.64%	72.42%	77.16%	81.44%
Test loss					0.5426046252
Test acc					81.44%

Initial model



Input layer

Convolutional layer

Convolutional layer

Max pooling

Convolutional layer

Convolutional layer

Max pooling

Convolutional layer

Max pooling

Flatten

Dropout

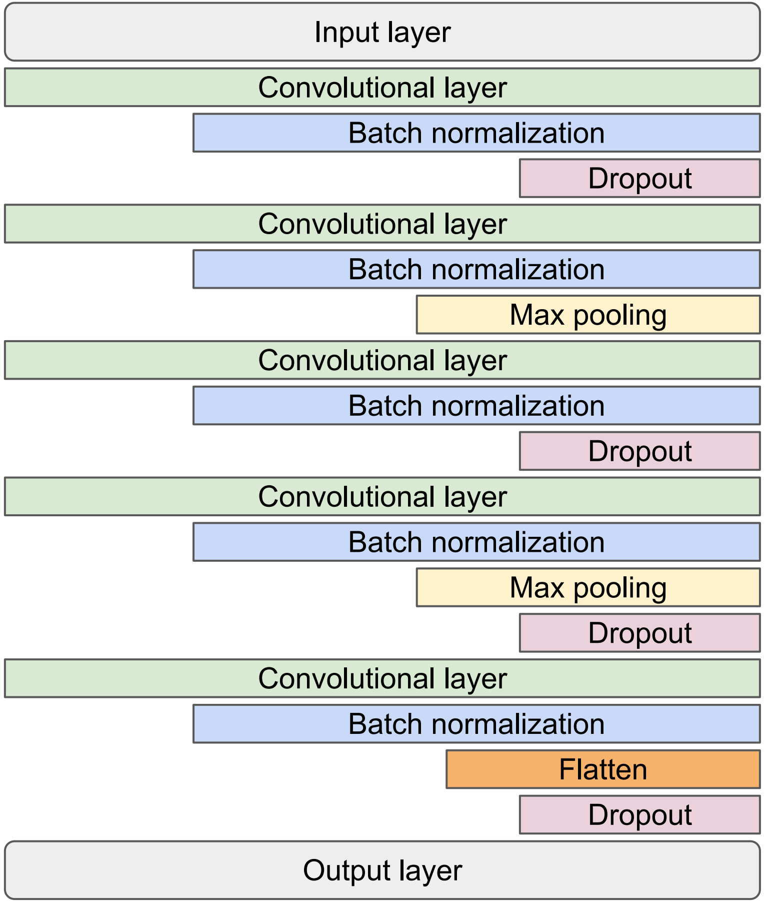
Output layer

Validation loss: 0.8817585707
Validatiion acc: 69.56%

Tested 19 different models:

- 😐 More nodes in dense layers.
- 😊 Addition of batching normalization layers.
- 😊 Increasing number of epochs with an early stopper.
- 😐 Data augmentation.
- 😊 Additional dropout layers.
- 😊 Decreasing the dropout.
- 😐 Transfer learning did not work… at the beginning.

Best in-house model


Input layer

Convolutional layer

Batch normalization

Dropout

Convolutional layer

Batch normalization

Max pooling

Convolutional layer

Batch normalization

Dropout

Convolutional layer

Batch normalization

Max pooling

Dropout

Convolutional layer

Batch normalization

Flatten

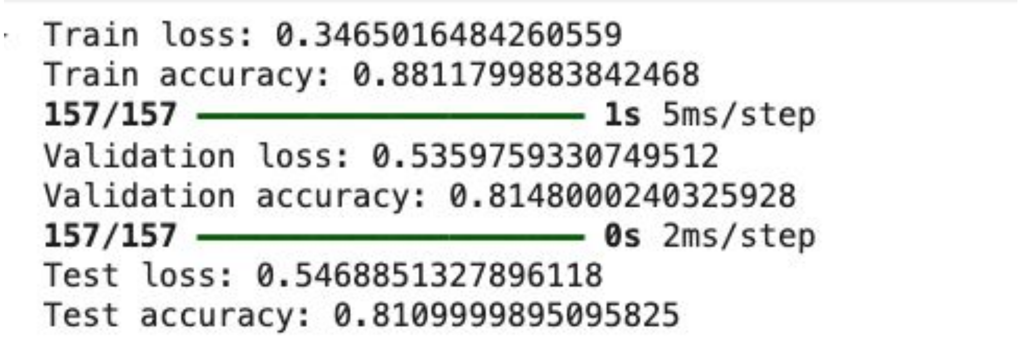
Dropout

Output layer

- **Avoid overfitting** (it delivers similar performance for the different dataset evaluations) by using dropout.

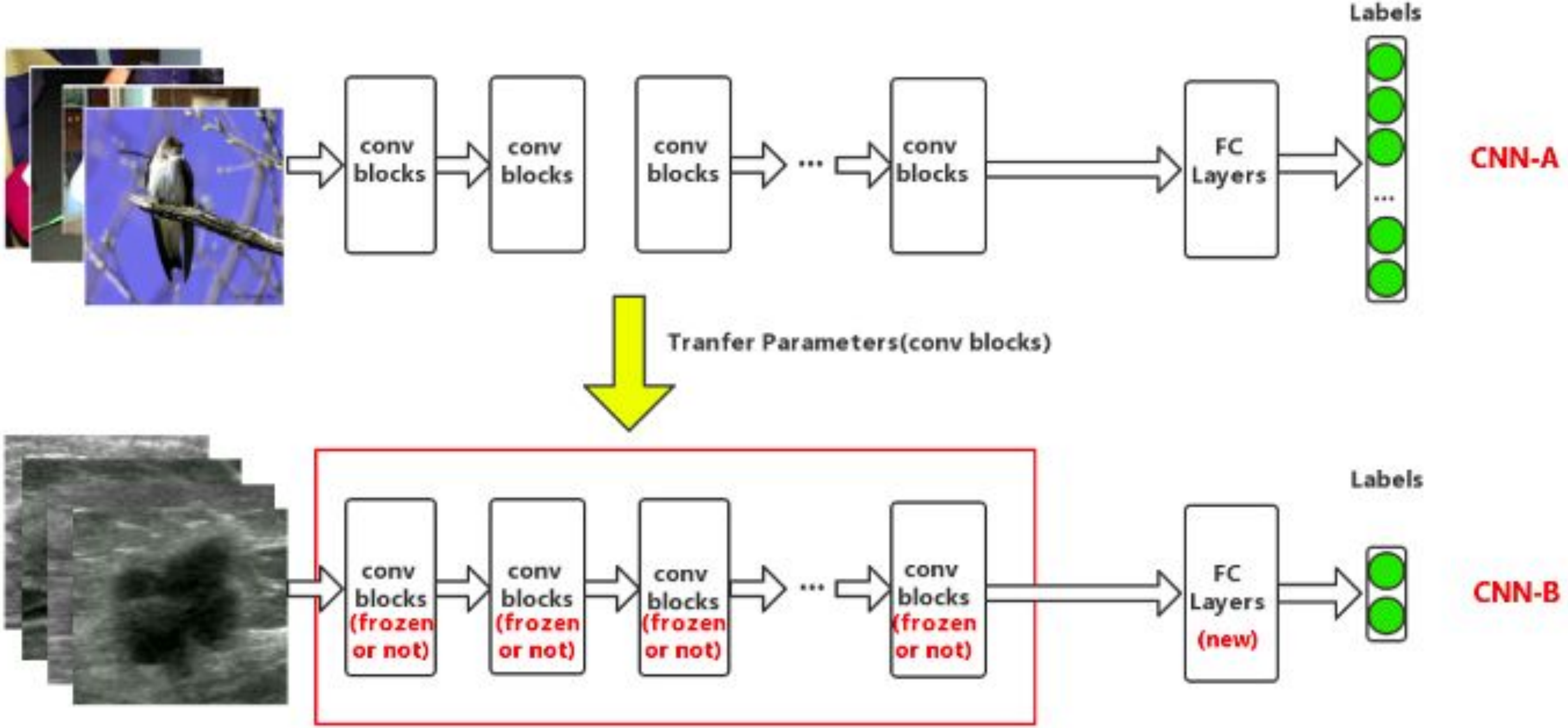
- Data augmentation did not improve our accuracy; our theory is that smaller-sized images can distort features so much that the model sees noise instead of useful patterns.

- The main performance increments came from the **batch normalization** and increment in **epochs** (from 15 to 100 with early stopper).



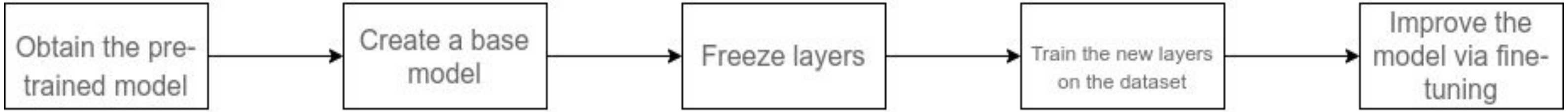
```
Train loss: 0.3465016484260559
Train accuracy: 0.8811799883842468
157/157 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step
Validation loss: 0.5359759330749512
Validation accuracy: 0.8148000240325928
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
Test loss: 0.5468851327896118
Test accuracy: 0.8109999895095825
```

Transfer learning

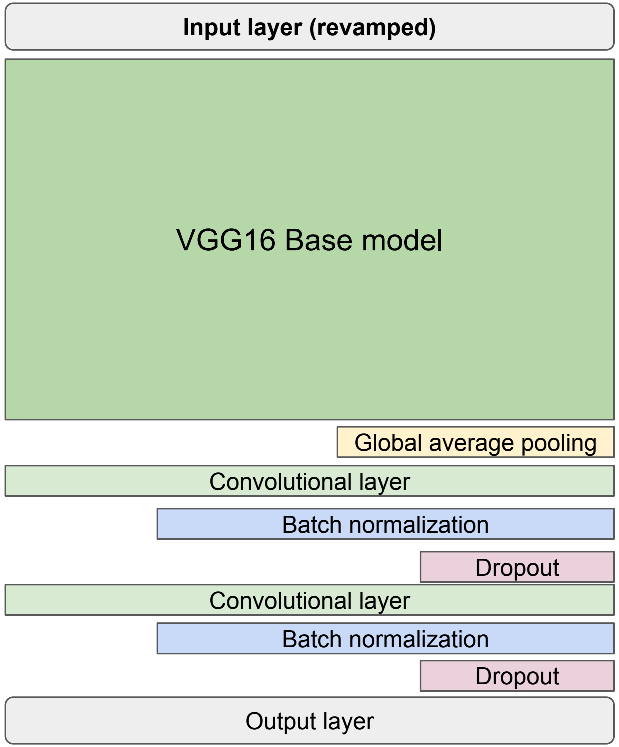


Labels
conv blocks
conv blocks
conv blocks
...
conv blocks
FC Layers
CNN-A
Tranfer Parameters(conv blocks)
Labels
conv blocks (frozen or not)
conv blocks (frozen or not)
conv blocks (frozen or not)
...
conv blocks (frozen or not)
FC Layers (new)
CNN-B

Winner model, aka Transfer Learning



Obtain the pre-trained model → Create a base model → Freeze layers → Train the new layers on the dataset → Improve the model via fine-tuning



Input layer (revamped)

VGG16 Base model

Global average pooling

Convolutional layer

Batch normalization

Dropout

Convolutional layer

Batch normalization

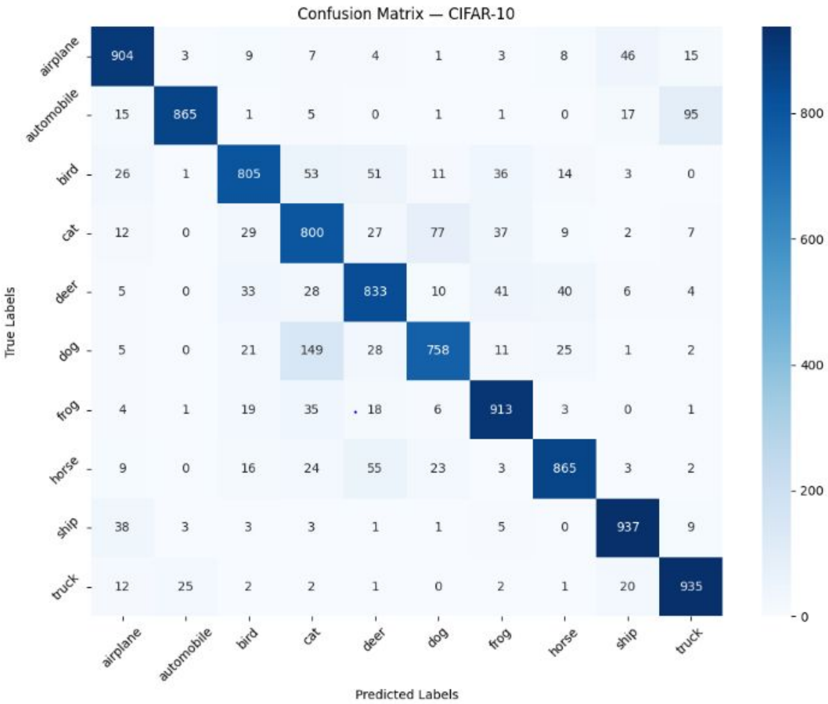
Dropout

Output layer

Validation loss: 0.34
Validation acc: 86.15%*
(*this model had only testing and training split.)

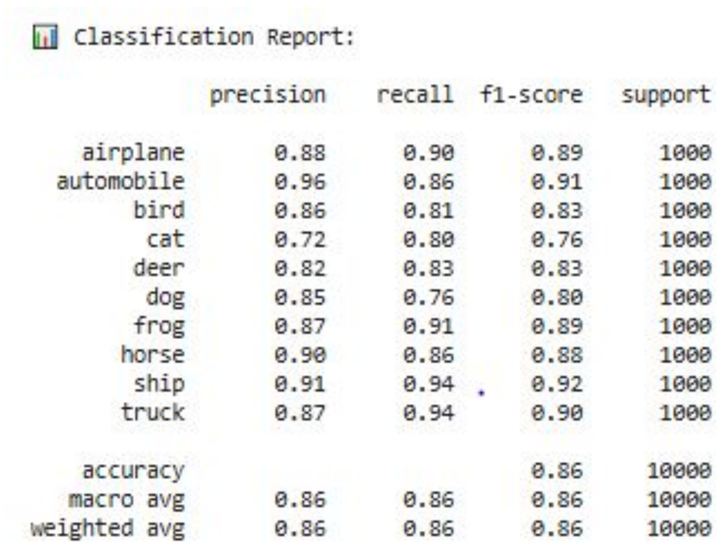
- **Preprocessing data** to fit base model
- **Global average pooling** (to convert the feature maps into vectors)
- Adding extra layers (**batch normalization** and **dropout**)
- Base model selection:
-
    1. MobileNetV2
    2. EfficientNETB0
    3. **VGG16**

Evaluation



Confusion Matrix — CIFAR-10

	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	904	3	9	7	4	1	3	8	46	15
automobile	15	865	1	5	0	1	1	0	17	95
bird	26	1	805	53	51	11	36	14	3	0
cat	12	0	29	800	27	77	37	9	2	7
deer	5	0	33	28	833	10	41	40	6	4
dog	5	0	21	149	28	758	11	25	1	2
frog	4	1	19	35	18	6	913	3	0	1
horse	9	0	16	24	55	23	3	865	3	2
ship	38	3	3	3	1	1	5	0	937	9
truck	12	25	2	2	1	0	2	1	20	935



```
📊 Classification Report:

              precision    recall  f1-score   support

    airplane       0.88      0.90      0.89      1000
  automobile       0.96      0.86      0.91      1000
        bird       0.86      0.81      0.83      1000
         cat       0.72      0.80      0.76      1000
        deer       0.82      0.83      0.83      1000
         dog       0.85      0.76      0.80      1000
        frog       0.87      0.91      0.89      1000
       horse       0.90      0.86      0.88      1000
        ship       0.91      0.94      0.92      1000
       truck       0.87      0.94      0.90      1000

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000
```

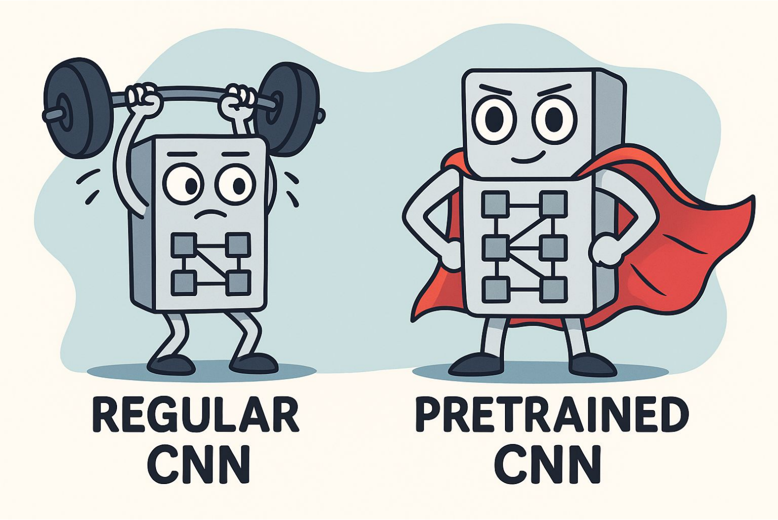
Learnings

- Iterative approach by changing one item at a time:
  - An increase of accuracy was achieved by adding batch normalization and dropout layers.
  - Avodied overfitting by adding dropout layers and early stopping.

- Selecting the dataset, we did not consider the challenges related to the sizing of images:
  - It affected the viability to use data augmentation.

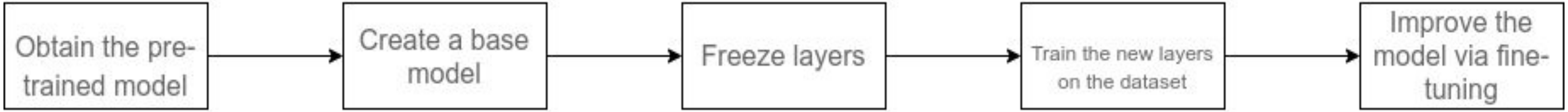
- Selecting the correct base model in the data model: this is a trial and error to find the model that works better.

Thank you for your attention!



Mariana Borssatto, Cristina Insignares, Adrián A. H., Kira Redberg

Transfer learning process



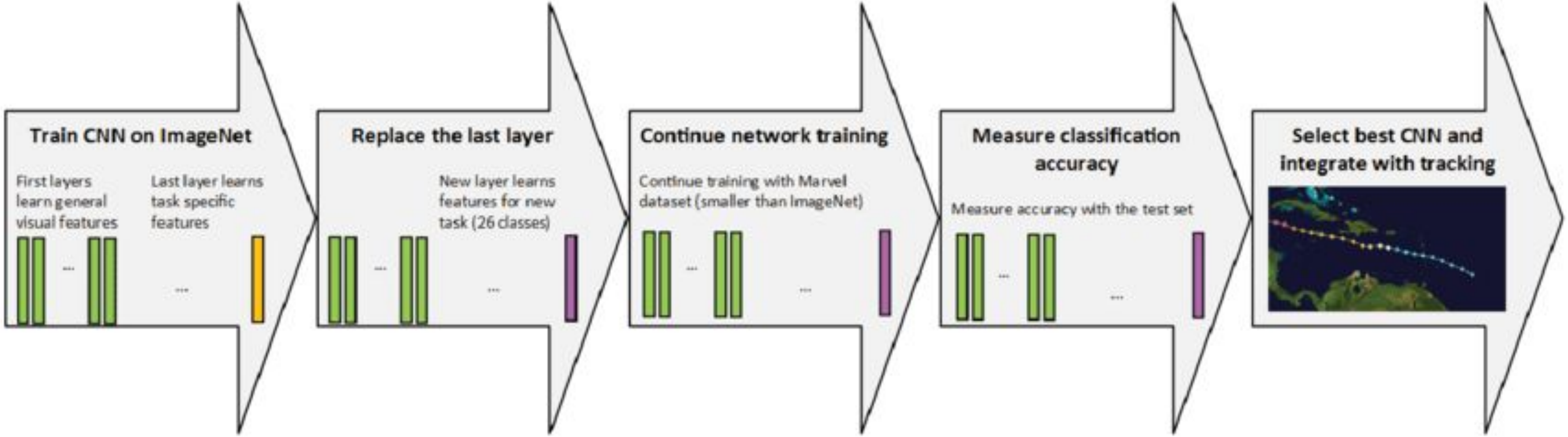
Obtain the pre-trained model

Create a base model

Freeze layers

Train the new layers on the dataset

Improve the model via fine-tuning



Train CNN on ImageNet

First layers learn general visual features

Last layer learns task specific features

Replace the last layer

New layer learns features for new task (26 classes)

Continue network training

Continue training with Marvel dataset (smaller than ImageNet)

Measure classification accuracy

Measure accuracy with the test set

Select best CNN and integrate with tracking