The CNN challenge

Deep Learning

Jelena Duric
Esteban Aspe
Nicoleta Roman

Process

01

Tried 5 architectures

02

Tried data augmentation

03

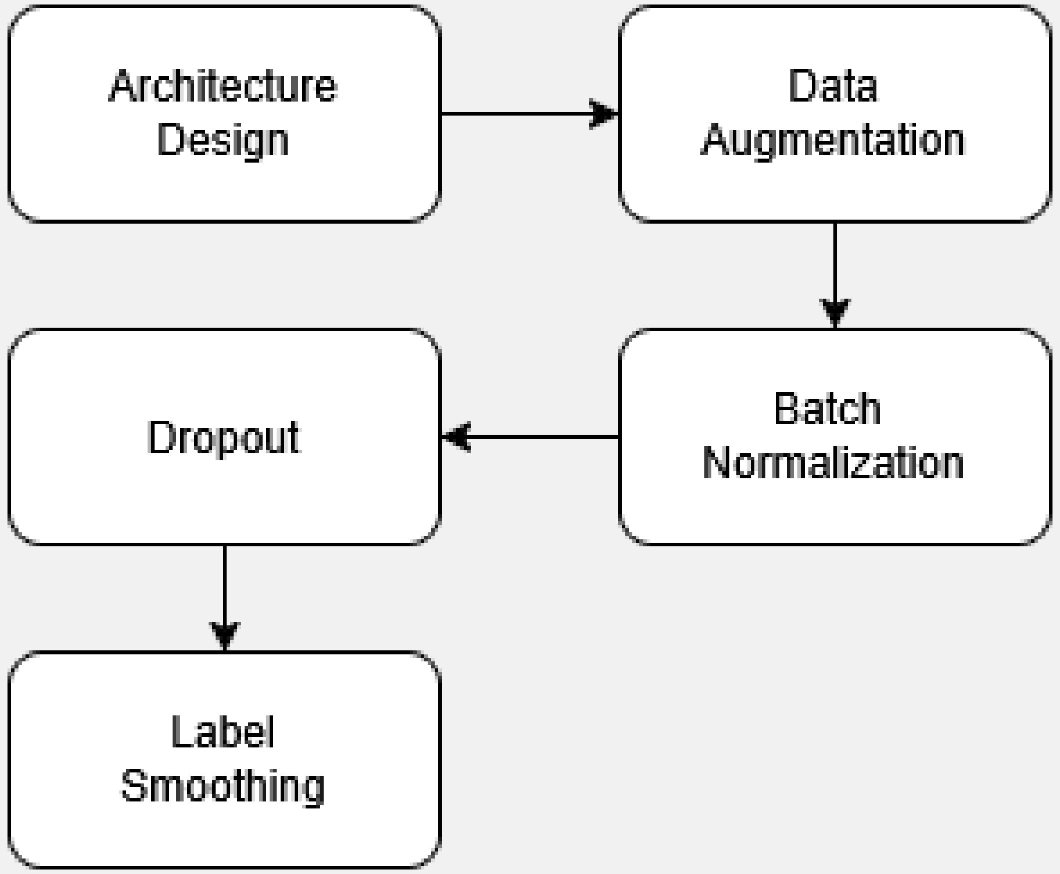
Tried with Batch Normalization

04

Tried with different Dropout rates

05

Tried with label smoothing



Architecture 1

- **Simple CNN**

Conv(32 filters, 3×3`) + leaky ReLU
Conv(32 filters, 3×3) + leaky ReLU
MaxPooling(2×2)
Conv(64 filters, 3×3) + leaky ReLU
Conv(64 filters, 3×3) + Leaky ReLU
MaxPooling(2×2)
Fully connected (128 neurons) +leaky ReLU

Architecture 2

- **Deeper CNN (VGG-style)**

Conv(64 filters, 3×3) + leaky ReLU
Conv(64 filters, 3×3) + leaky ReLU
MaxPooling(2×2)
Conv(128 filters, 3×3) + leaky ReLU
Conv(128 filters, 3×3) + ReLU
MaxPooling(2×2)
Conv(256 filters, 3×3) + leaky ReLU
Conv(256 filters, 3×3) + leaky ReLU
MaxPooling(2×2)
Fully connected (256 neurons) + leaky ReLU

Architecture 3

- **Residual CNN (ResNet Inpired, but smaller)**

Conv(64 filters, 3×3) + leaky ReLU
Residual Block:
Conv(64, 3×3) + leaky ReLU
Conv(64, 3×3) (skip connection)
MaxPooling(2×2)
Conv(128 filters, 3×3) + leaky ReLU
Residual Block:
Conv(128, 3×3) + leaky ReLU
Conv(128, 3×3) (skip connection)
MaxPooling(2×2)
Fully connected (256 neurons) + leaky ReLU

Architecture 4

- **Dilated CNN**

Conv(64 filters, 3×3, dilation=2) + leaky ReLU
Conv(64 filters, 3×3, dilation=2) + leaky ReLU
MaxPooling(2×2)
Conv(128 filters, 3×3, dilation=2) + leaky ReLU
Conv(128 filters, 3×3, dilation=2) + leaky ReLU
MaxPooling(2×2)
Fully connected (128 neurons) + leaky ReLU

Architecture 5

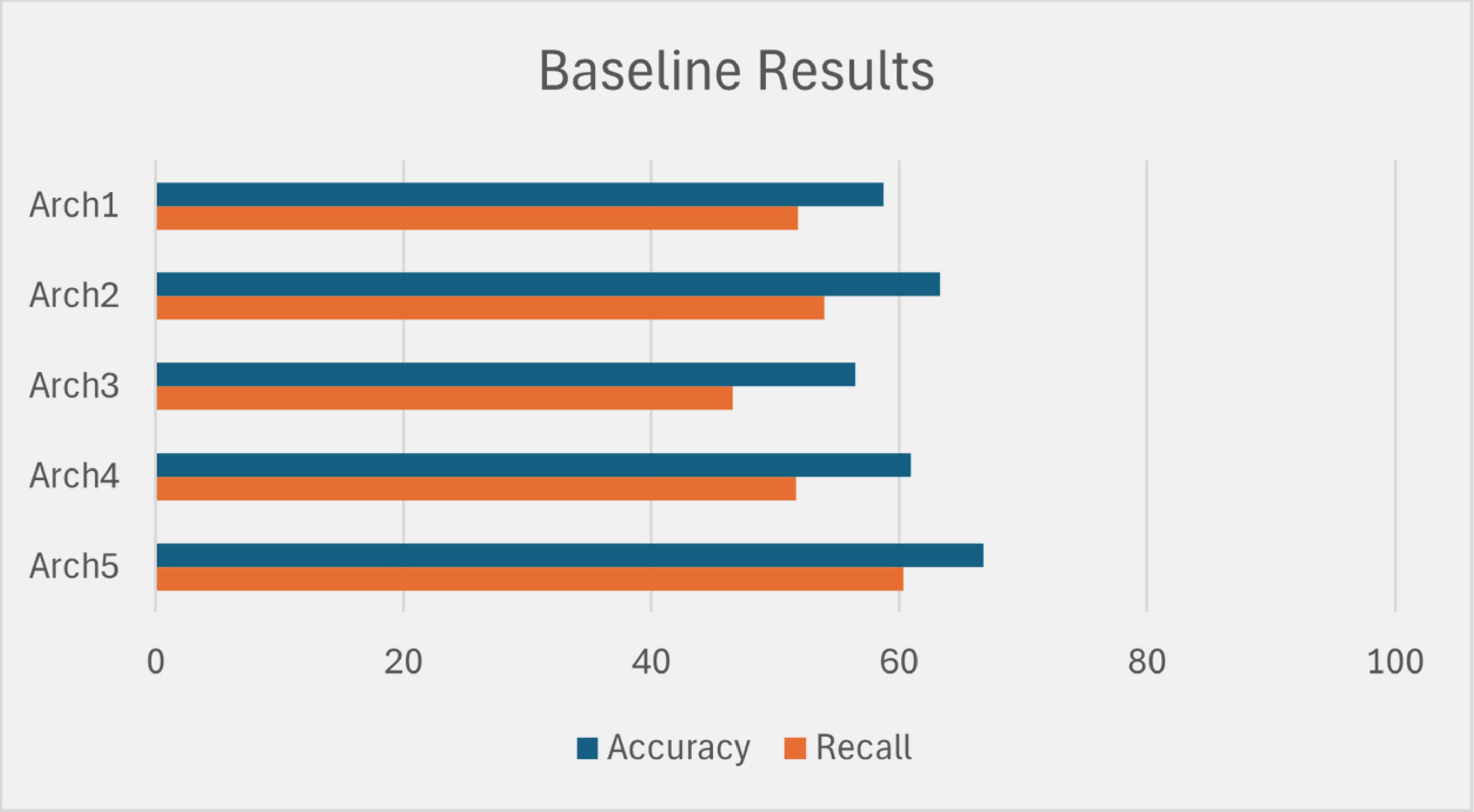
- **Extra-Deep CNN (VGG-like)**

Conv layer (64 filters, 3×3) + LeakyReLU
Conv layer (64 filters, 3×3) + LeakyReLU
MaxPooling (2×2)
Conv layer (128 filters, 3×3) + LeakyReLU
Conv layer (128 filters, 3×3) + LeakyReLU
MaxPooling (2×2)
Conv layer (256 filters, 3×3) + LeakyReLU
Conv layer (256 filters, 3×3) + LeakyReLU
MaxPooling (2×2)
Conv layer (512 filters, 3×3) + LeakyReLU
Conv layer (512 filters, 3×3) + LeakyReLU
MaxPooling (2×2)
Dense (256 neurons) + LeakyReLU
Dense (256 neurons) + LeakyReLU

Baseline Results

Architectures used, all of them with leakyReLu and Adam:

1. Simple CNN
2. Deeper CNN (VGG-style)
3. Residual CNN (ResNet-style but smaller)
4. Dilated CNN
5. Extra-Deep CNN (VGG-style)



Discussion. Data augmentation

- Applied it on training data to architectures 2 and 5
- Applied several image transformations to make the model more flexible and prevent overfitting
- It randomly rotates images up to 20 degrees, shifts them left, right, up, or down by 20%, and slightly tilts (shears) them. It also zooms in and out by 20% and flips images horizontally. If any empty spaces appear due to these changes, they are filled using nearby pixel values
- Kept training on 20 epochs, considering that the overfitting was happening pretty early in the previous network (around third and fourth epoch)
- Results on both network were great, overfitting was almost non existent

Discussion. Batch normalization

- Tried batch normalization on architecture 5 with data augmentation, to prevent overfitting even more
- Used 100 epochs and lowered batch size to 64, considering that with 128 we ran out of memory on the GPU
- Very good results and with no overfitting

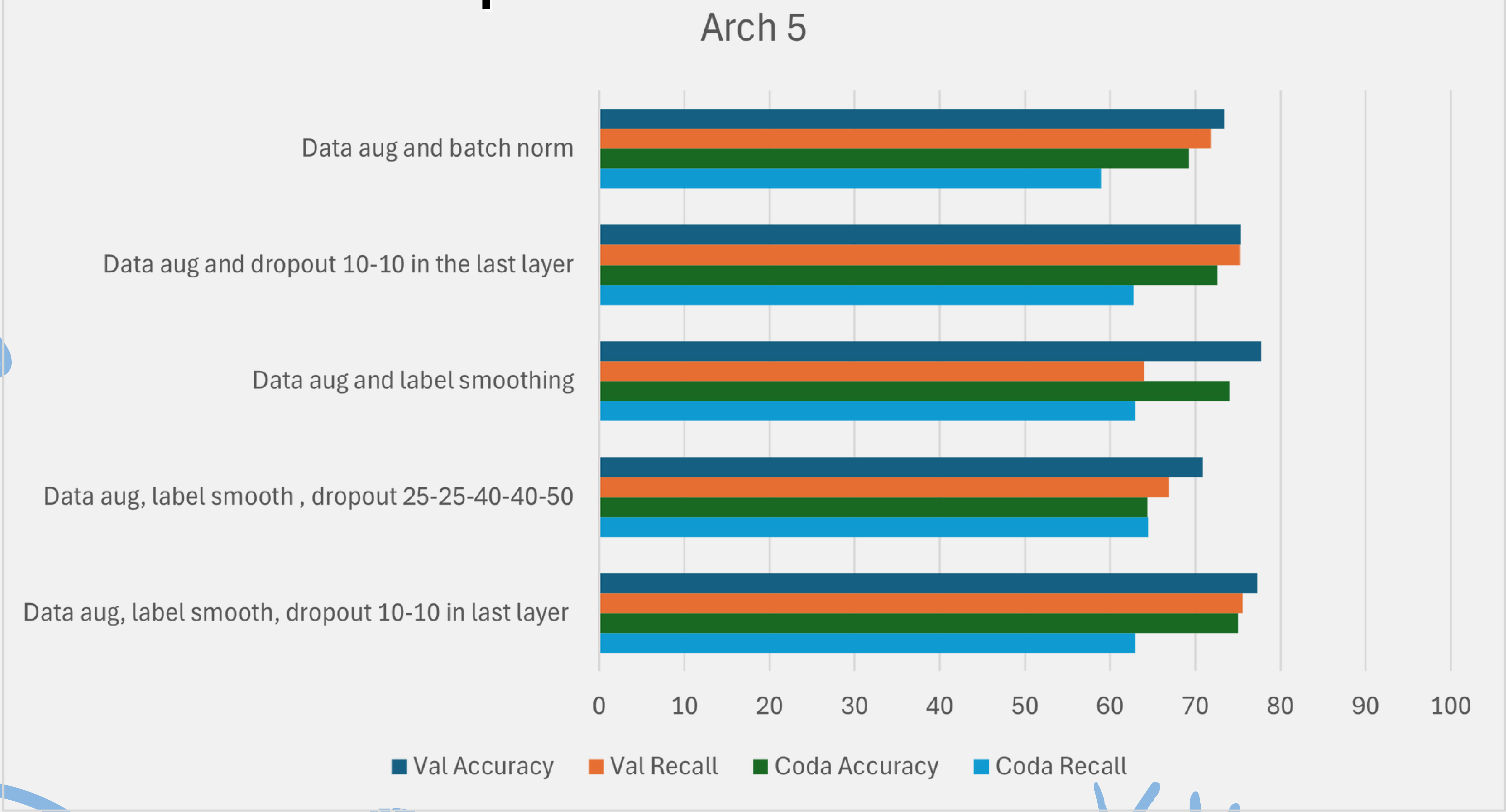
Discussion. Dropout

- First used with data augmentation, great results with no overfitting
- Possibly putting more epochs and get even better results

Discussion. Label Smoothing

- Applied to architecture 2 with 20 epochs combined with data augmentation, resulting in a relatively bad performance overall.
- Architecture 5 was taken with 100 epochs and data augmentation as well, performing significantly better, however it tended to overfit.
- A dropout (25-25-40-40-50) was added, which gave worse results than without the dropout, nevertheless it overfitted less.
- Also, a smaller dropout (10) in the dense layers, gave a very similar result with the experiment tried without dropout at all.
- A bigger dropout (30) (+ batch normalization) resulted in a decrease of 15% in validation accuracy.

Discussion. Comparison

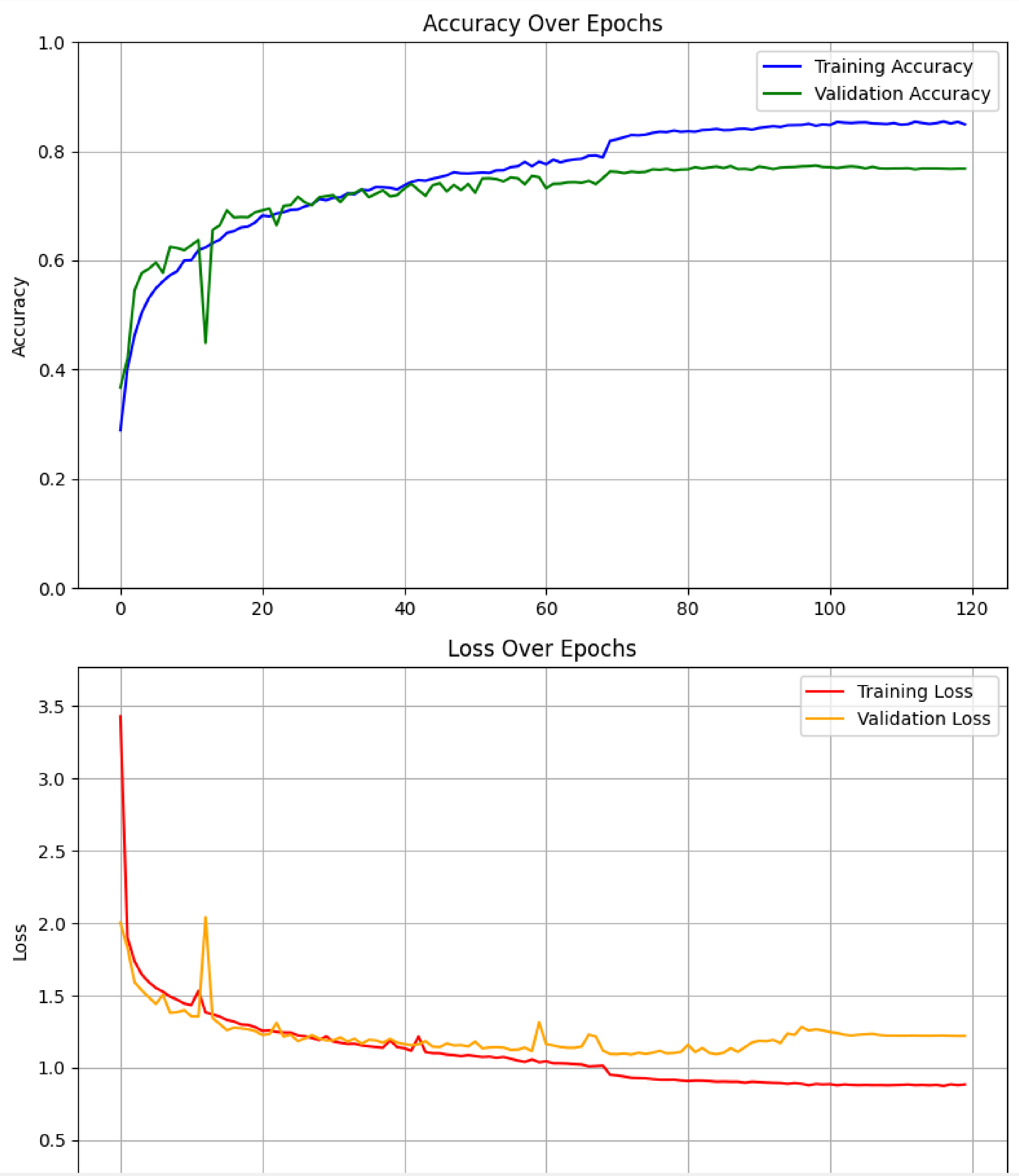


Conclusions

The **best performing neural networks** were:

- network 5 with data augmentation and label smoothing
- network 5 with data augmentation, label smoothing and dropout in the dense layers

Network	Mean Accuracy	Mean Precision	Mean Recall
without dropout	74.72	63.99	63.48
with dropout	75.07	73.87	63.69



Network 5 without dropout

Thank you