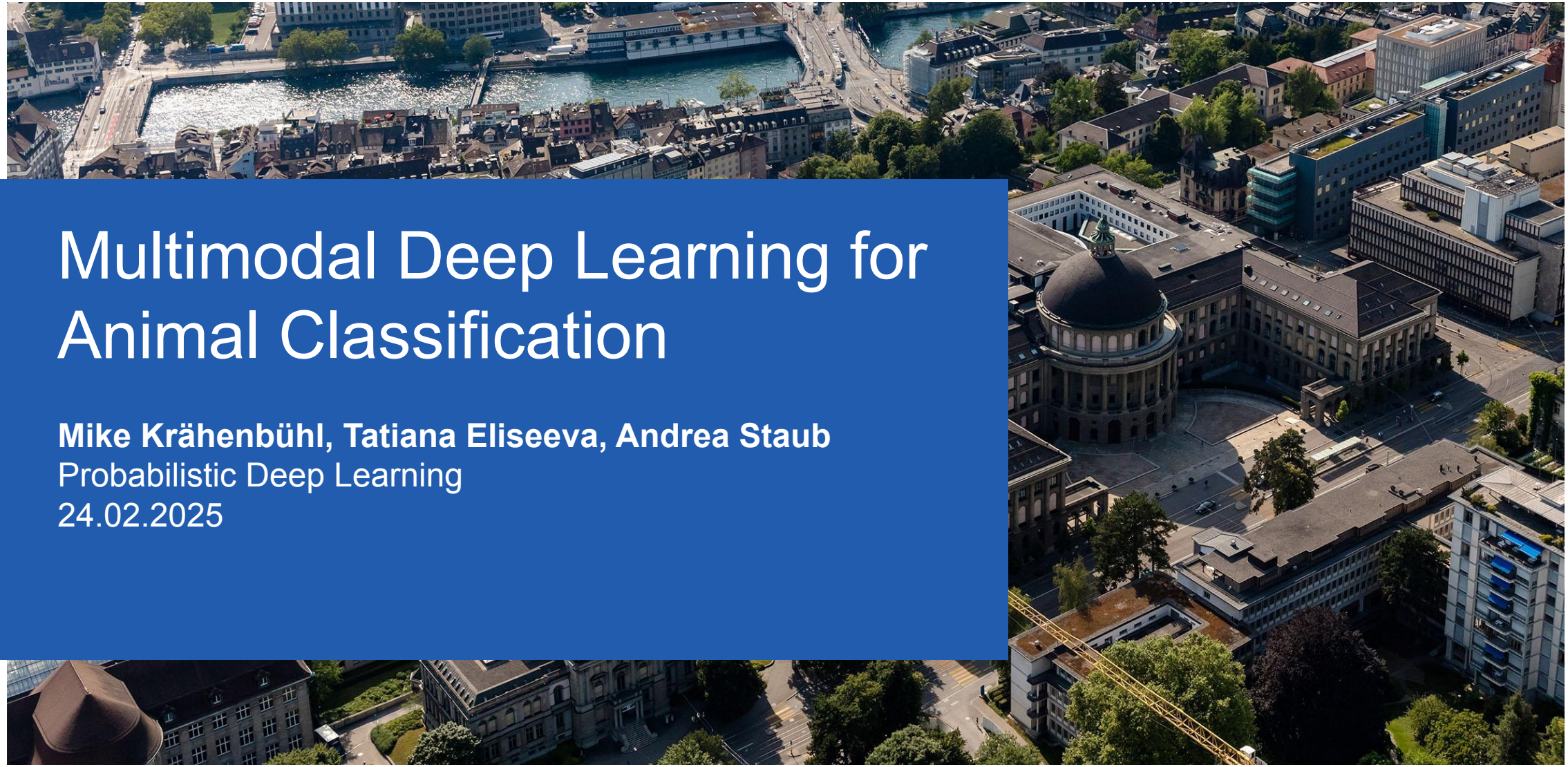
ETH zürich



Multimodal Deep Learning for Animal Classification

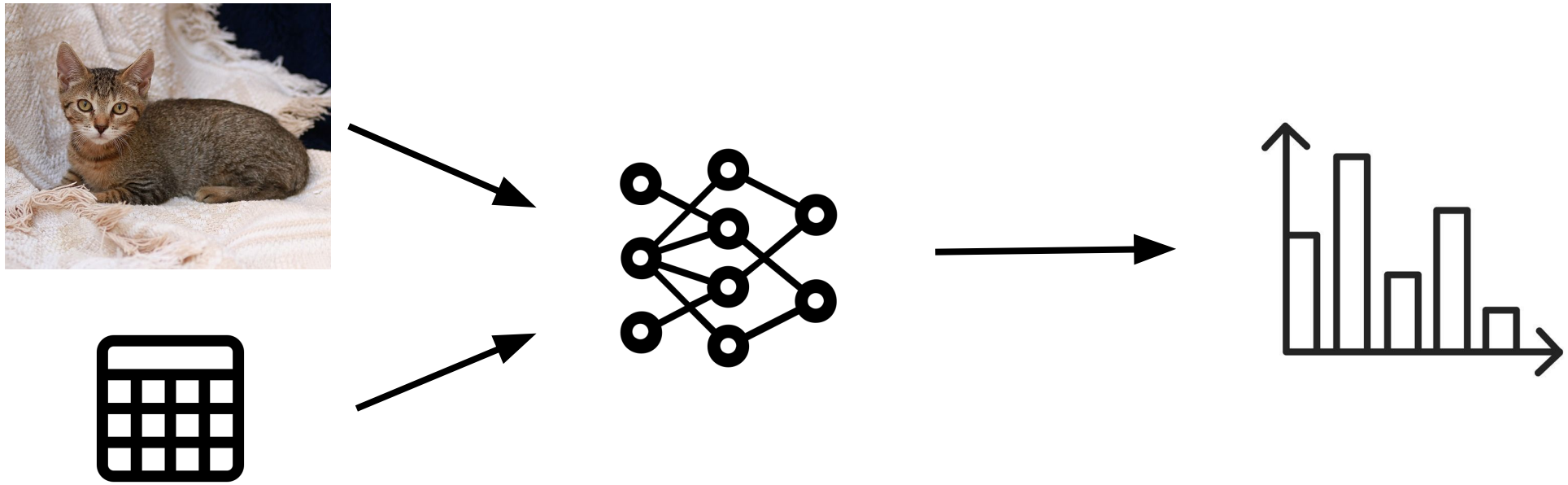
Mike Krähenbühl, Tatiana Eliseeva, Andrea Staub

Probabilistic Deep Learning

24.02.2025

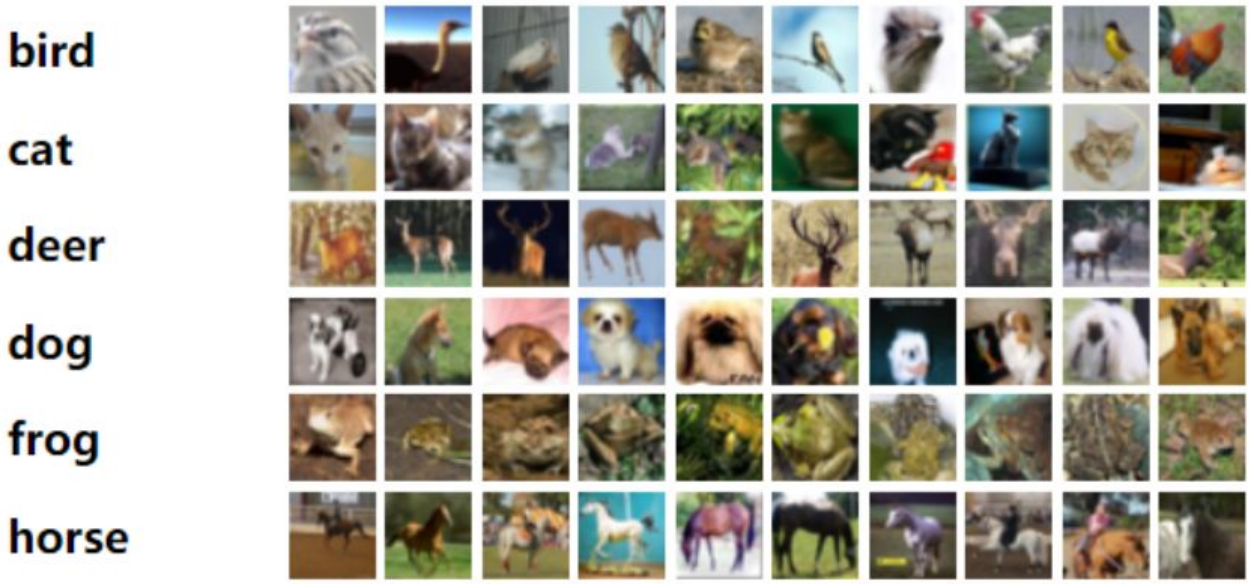
Goal

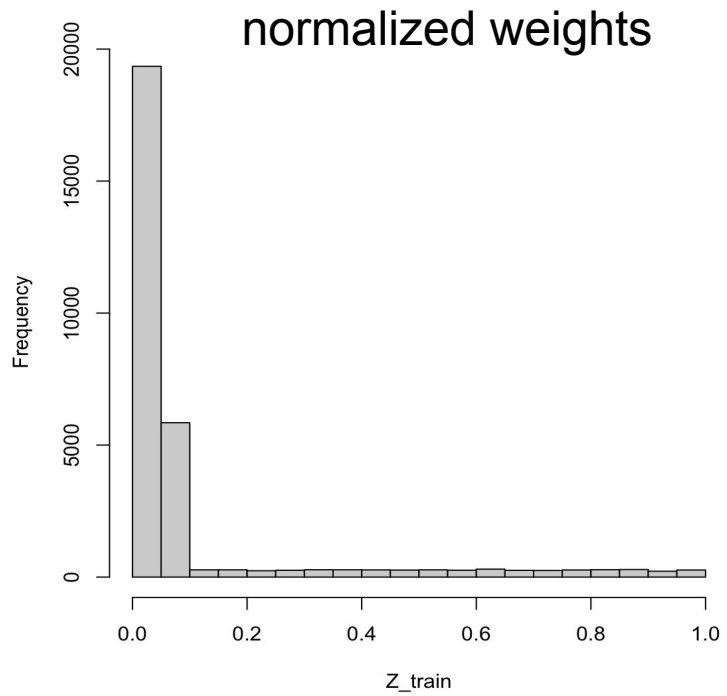
- Predicting the probability distribution of animal categories based on **image** data and **tabular** data using different NN architectures.



Data

- **Image data:** CIFAR-10 animal images, 32x32 colour images in six classes, with 6’000 images per class.
- **Tabular data:** Randomly generated weight from uniform distribution with reasonable lower and upper bounds for each animal type.

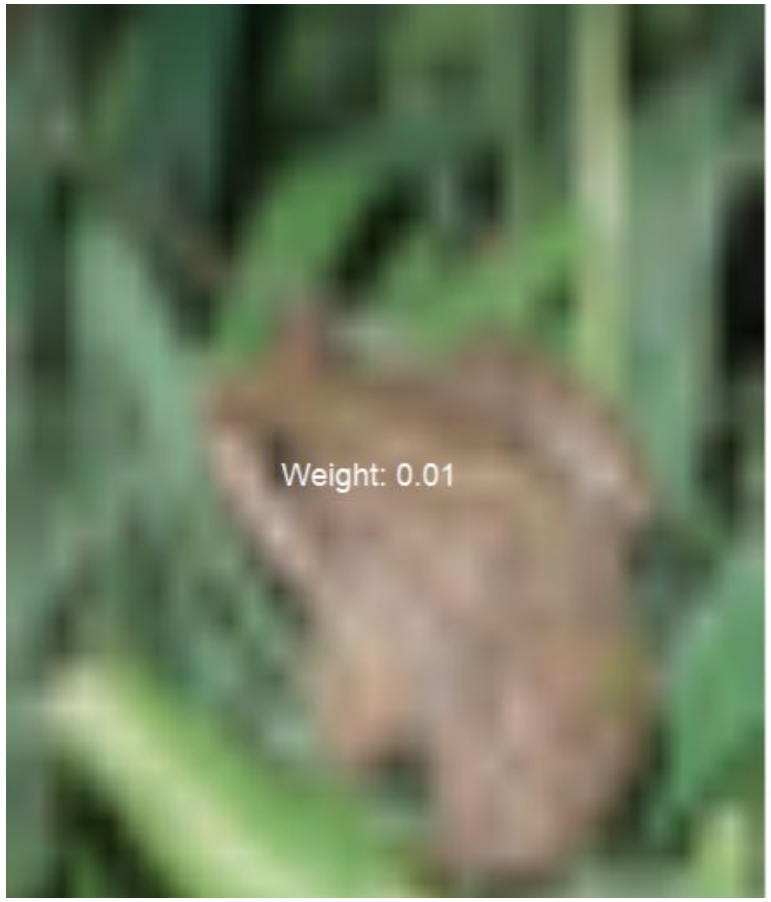


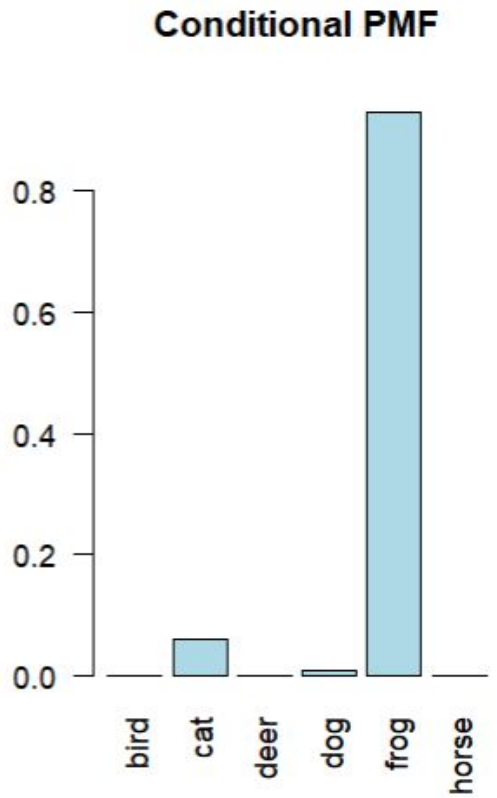


Estimated Conditional Probability Distribution for an Observation

Model:

CNN (image) + Tabular (weight)



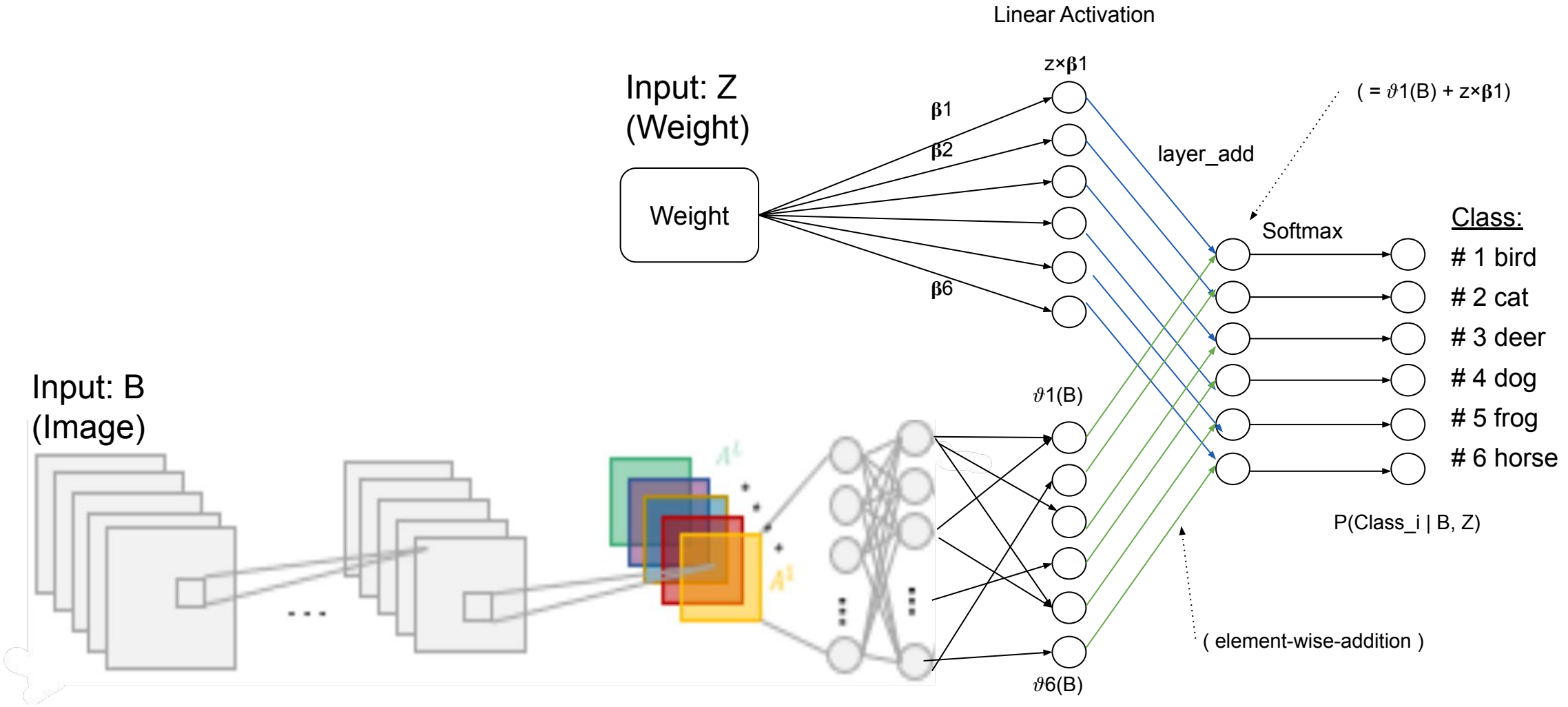


Models to compare

Model Name	Architecture Brief Summary*	Inputs
Model 1 - CNN	“Classic” Image CNN	Images
Model 2 - 1LfcNN-b	Single-Layer fcNN. Contains weights only, softmax activation function without bias	Weights
Model 3 - 1LfcNN	Single-Layer fcNN. Contains weights only, softmax activation function with bias	Weights
Model 4 - Deep-fcNN	Deep fcNN. Contains weights only, ReLu kern with bias	Weights
Model 5 - CNN + 1LfcNN	CNN + Single-Layer fcNN	Images and weights
Model 6 - CNN + Deep-fcNN	CNN + Deep fcNN	Images and weights

*Always softmax activation at output nodes

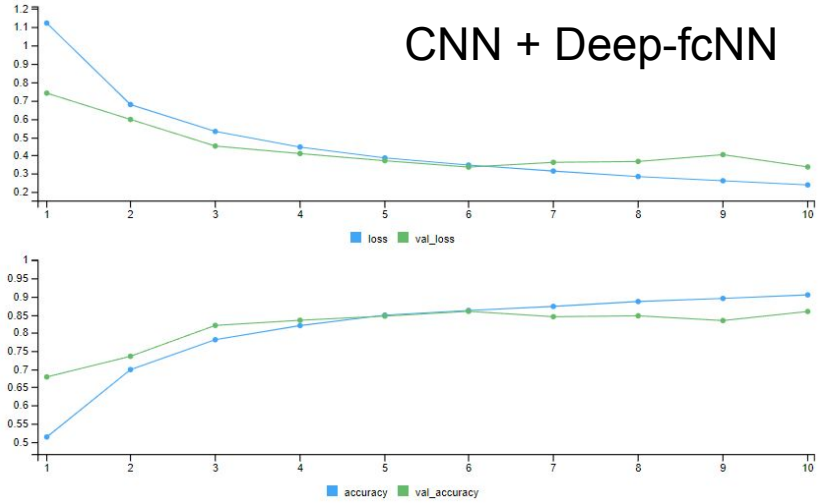
Model 5 Architecture (CNN + linearNN)



Training Workflow

1. Imput Normalisation and One-hot encoding for classes
2. Model compilation with NLL (loss = "loss_categorical_crossentropy", optimizer = optimizer_adam(), metrics = c("accuracy"))
3. Initialize Early Stopping parameters (callback_early_stopping(monitor = "val_loss", patience = 3, restore_best_weights = TRUE))
4. Model training: epochs = 100, batch_size = 64, validation_split = 0.2, callbacks = list(early_stopping)

CNN + Deep-fcNN

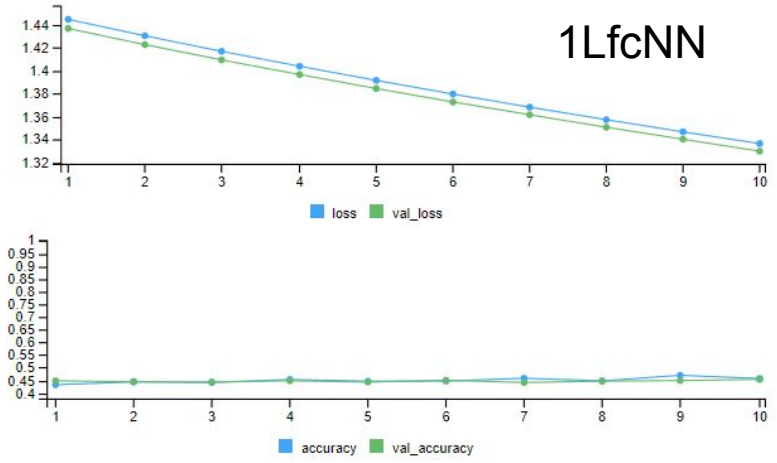


Multiple model initialisation did not make any significant difference for the model accuracy. As an example - validation loss about two digits after comma. It shows that the model is robust for the chosen data.

Deviations on “classical settings”, sometimes quite big, like using only half of the nodes, did not brought any significant results.

“Thumb Rule” for our models and our problems: 10 Epochs for the run is enough.

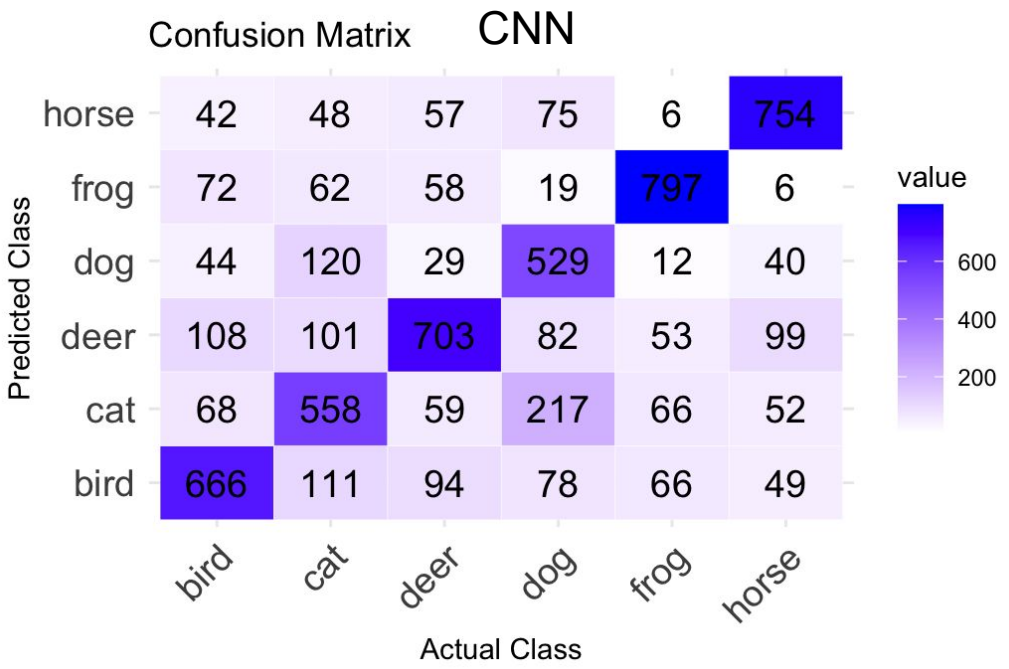
1LfcNN

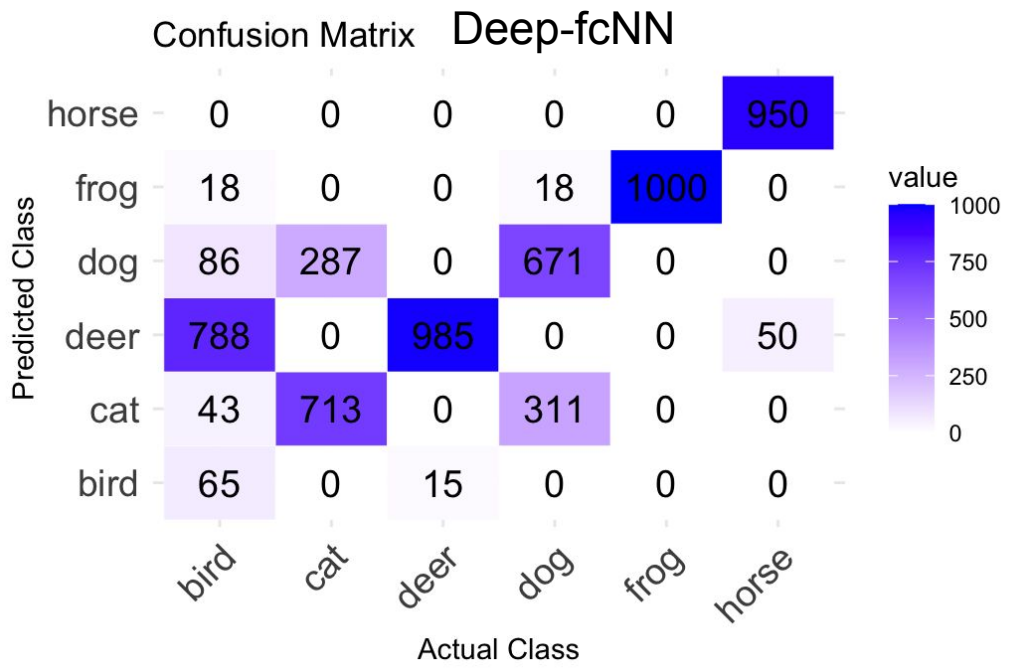


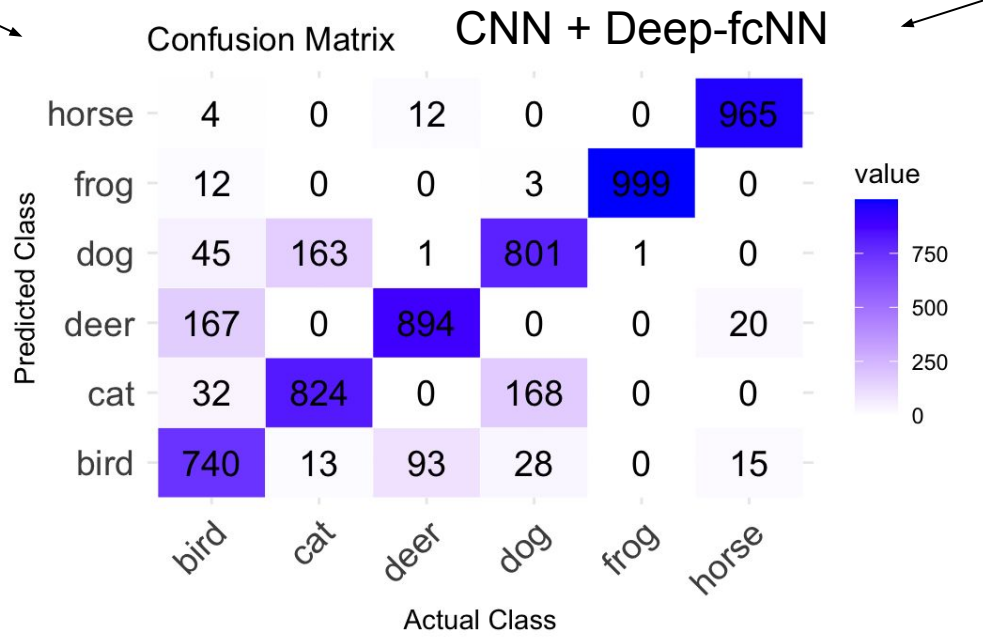
Test Results

Model	Accuracy	Loss
CNN	0.674	0.902
1LfcNN-b	0.167	1.766
1LfcNN	0.446	1.451
Deep-fcNN	0.757	0.537
CNN + 1LfcNN	0.713	0.792
CNN + Deep-fcNN	0.871	0.344

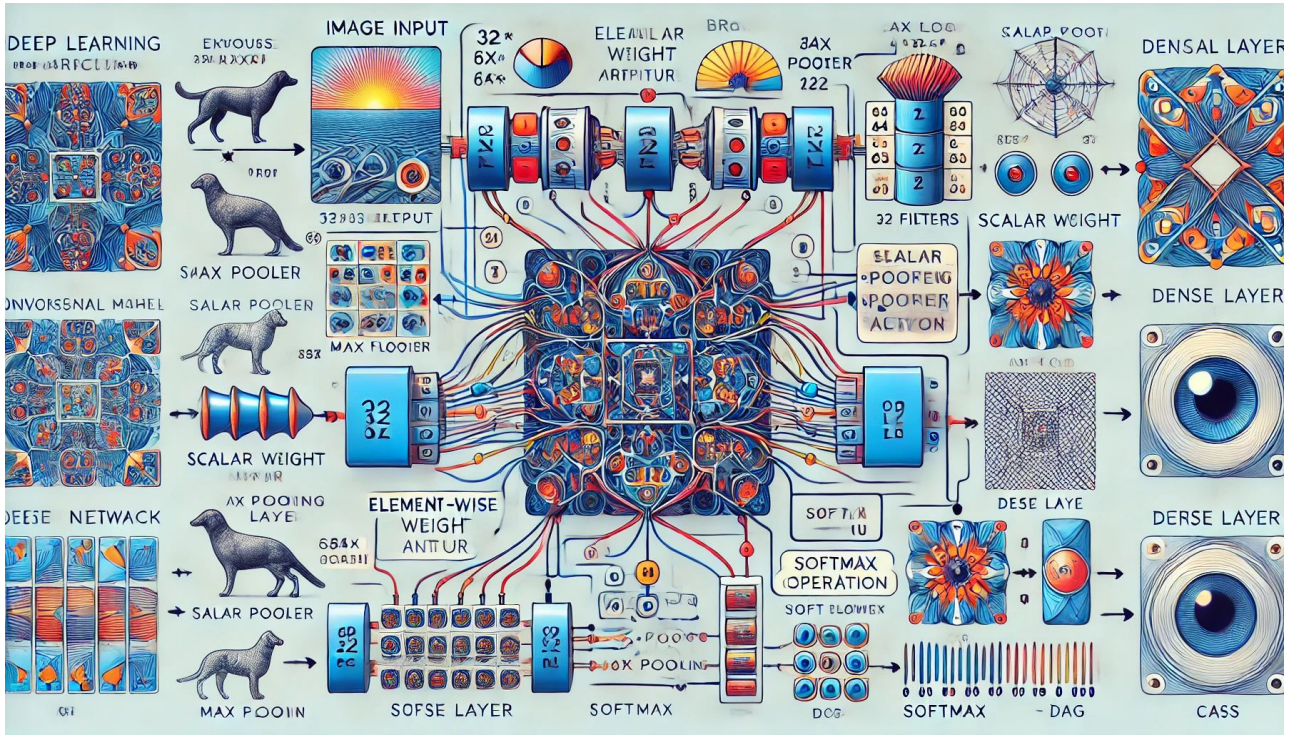
- Single-Layer NN has too low accuracy
- Deep NN allow to capture the features needed for proper categorisation
- Already one layers of fcNN that allows to process tabular data, can significantly increase the accuracy
- Best results shows combination of CNN and Deep fcNN


Confusion Matrix
CNN
horse 42 48 57 75 6 754
frog 72 62 58 19 797 6
dog 44 120 29 529 12 40
deer 108 101 703 82 53 99
cat 68 558 59 217 66 52
bird 666 111 94 78 66 49
Predicted Class
bird cat deer dog frog horse
Actual Class
value
600
400
200


Confusion Matrix
Deep-fcNN
horse 0 0 0 0 0 950
frog 18 0 0 18 1000 0
dog 86 287 0 671 0 0
deer 788 0 985 0 0 50
cat 43 713 0 311 0 0
bird 65 0 15 0 0 0
Predicted Class
bird cat deer dog frog horse
Actual Class
value
1000
750
500
250
0


Confusion Matrix
CNN + Deep-fcNN
horse 4 0 12 0 0 965
frog 12 0 0 3 999 0
dog 45 163 1 801 1 0
deer 167 0 894 0 0 20
cat 32 824 0 168 0 0
bird 740 13 93 28 0 15
Predicted Class
bird cat deer dog frog horse
Actual Class
value
750
500
250
0

Conclusions



- Combination of different types of input data by training of DeepNN tools allows to increase the recognition efficiency for the low-weight solutions
- For data without significant preprocessing a NN hast to have sufficient capacity to store the features