Fracture classification using Neural Networks

Murat Bilici - bilicmur@students.zhaw.ch

Marko Simic - simicma1@students.zhaw.ch

Deep Learning (CAS machine intelligence, 2024)

Background and Motivation

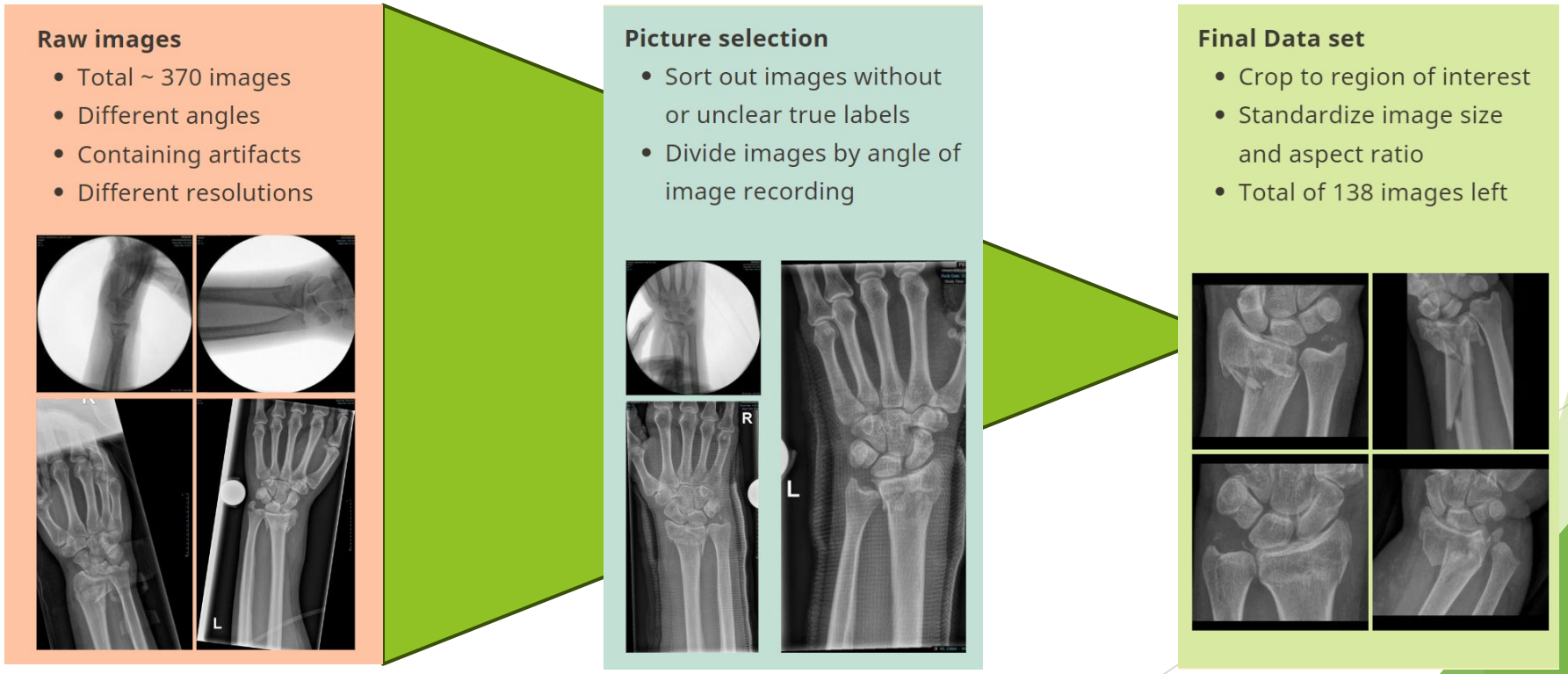
- Background:
  - Accurate classification of wrist fractures is paramount in guiding treatment decisions, predicting outcomes, and preventing complications.
  - Due to variations in interpretation and the complexity of cases, often multiple doctors and evaluations of X-ray images are required for the correct diagnosis.
  - Misdiagnosis or delayed diagnosis can lead to prolonged pain and decreased functionality that results in increased healthcare cost.
- Motivation to automate
  - Accuracy: Consistent and precise classification reducing risk of misdiagnosis
  - Efficiency: Rapid analysis process reduces the need of multiple manual evaluations
  - Accessibility: Automated classification algorithm can be installed and shared in various healthcare institutes.

The Data set

- Raw data set of ~ 370 grey scale x-ray images in three categories
- True labels (categorization) available created by three scientific experts
- No standardization of image acquisition and different attributes such as:
  - Pixel size and aspect ratio, picture orientation, shooting angle, resolution
- Artifacts not belonging to wrist such as:
  - Infusion tubings, fibers from textiles, artificial image labelings
- Class imbalance. Strong over- and underrepresentation of classes.

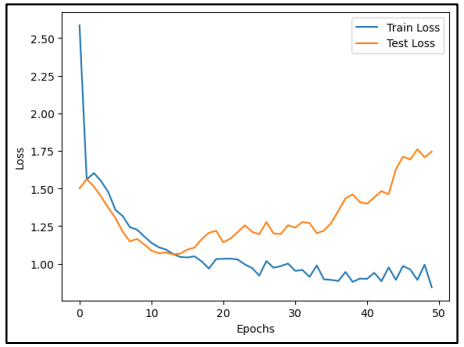
Data set preparation

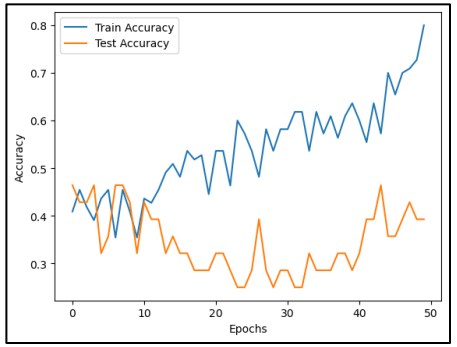
- Significant data set reduction due neccessity of sorting out of pictures
  - Focus on one shooting angle (top).
  - Total of 138 working pictures left (from ~ 370)

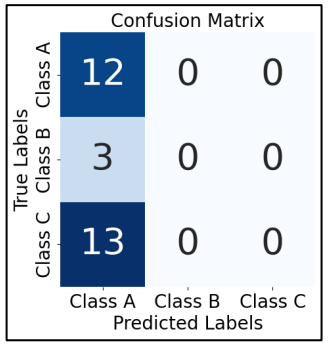


The initial situation

- Problems encountered are:
  - Overfitting, poor accuracy on test set and one sided missclassifications







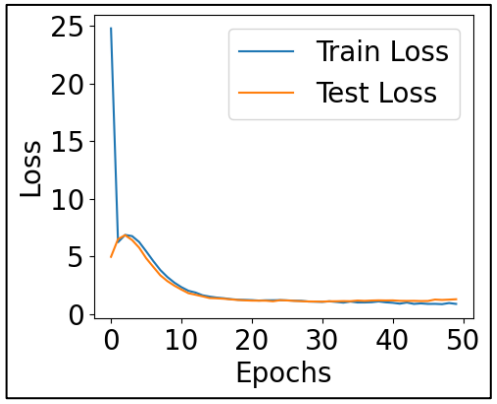
  - Performance and oom issues when handling complex models
    - GPU calculates significantly faster but limited with VRAM (10GB with rtx3080)
    - CPU draws advantage from more RAM (64GB) but way too slow

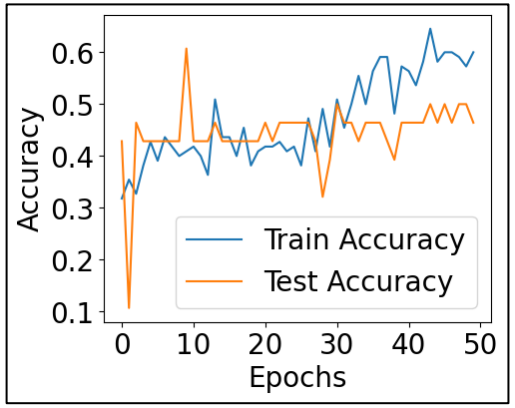
Searching for the «best» model

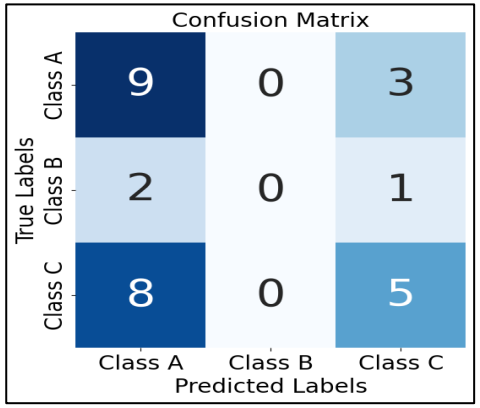
- Initialy familiarizing by manual step-by-step model design
  - Adding more, different combinations of layers
  - Combinations of hyperparameters their respective attributes
  - Changing optimizers
- Usage of cross validation to test multiple hyperparameter combinations
  - Evaluation of 320 models (GPU based) and further manual fine tuning

```
hyperparameters = {
    "kernel_size": [(2, 2), (3, 3)],
    "dropout": [0.5],
    "l2_regularization": [0.001, 0.1],
    "dense_layers": [300,64],
    "batch_normalization": [True, False],
    "optimizer_lr": [('sgd', 0.001), ('sgd', 0.1), ('adam', 0.001), ('adam', 0.1)]
}
```

Comparison to basic random forest model







Metric	cNN model
Accuracy	0.464
Precision	0.461
Recall	0.500
F1 score	0.459
Accuracy CI	0.315 – 0.685

```
batch_size = 32
epochs= 50
optimizer= "adam"
loss= "categorical_crossentropy"
model = Sequential([
    Input(shape=(image_height, image_width, 1)),
    Conv2D(32, (3, 3), activation='relu'),
    Conv2D(32, (3, 3), activation='relu'),
    Conv2D(64, (3, 3), activation='relu'),
    Conv2D(64, (3, 3), activation='relu'),
    Flatten(),
    Dense(64, activation='relu', kernel_regularizer=l2(0.01)),
    Dropout(0.75),
    Dense(3, activation='softmax')
])
```

Comparison to basic random forest model

- How does the model compete agains a random forest?

Metric	Ranom forest	cNN model	Evaluation cNN
Accuracy	0.286	0.464	Higher overall accuracy
Precision	0.256	0.461	Fewer false positive predictions
Recall	0.286	0.500	Capturing of more true positive
F1 score	0.269	0.459	Better balance between TP and FP
Accuracy CI	0.279 - 0.289	0.315 – 0.685	Higher uncertainty due to wider range

Further attempts

Method	Ranom forest	Outcome
Data augmentation	Increasing data set and variation	No siginificant difference / improvement
SGD optimizer	Alternative to adam	No siginificant difference / improvement
VGG16 & ResNet	PreTrained networks	Tweaking required to load greyscale images Both showed no significant difference / improvement
EarlyStopping	Prevent Overfitting	At cost of training depth
Batch size 16	Small data set	Increase in calculation time No siginificant difference / improvement

Conclusions

- The selected model does not perform very accurate and robust on the data set.
- It outperforms a random forest model somewhat indicating that the model was able to learn features along the way.
- The original raw data set was already limited with ~ 370 images
- The picture pre seletion reduced the available data further to 138 images.
- The pixel reduction to a standard 224x224 input pixel size reduced the resolution and potentially the ability of the NN to detect the features of interest

Conclusions

- The original raw data present some challenges:
  - Inhomogeneity of picture characteristics such as size, orientation, angle, resolution


  - Disturbing artifacts such as infusion tubings, bandages


Conclusions

- Differences between the classes / categories mostly subtle and features often don’t stand out from the «background» (normal bone features)
- No reference pictures of non-fractures available.
- Disagreement in classification even between experts (what is «right»?)