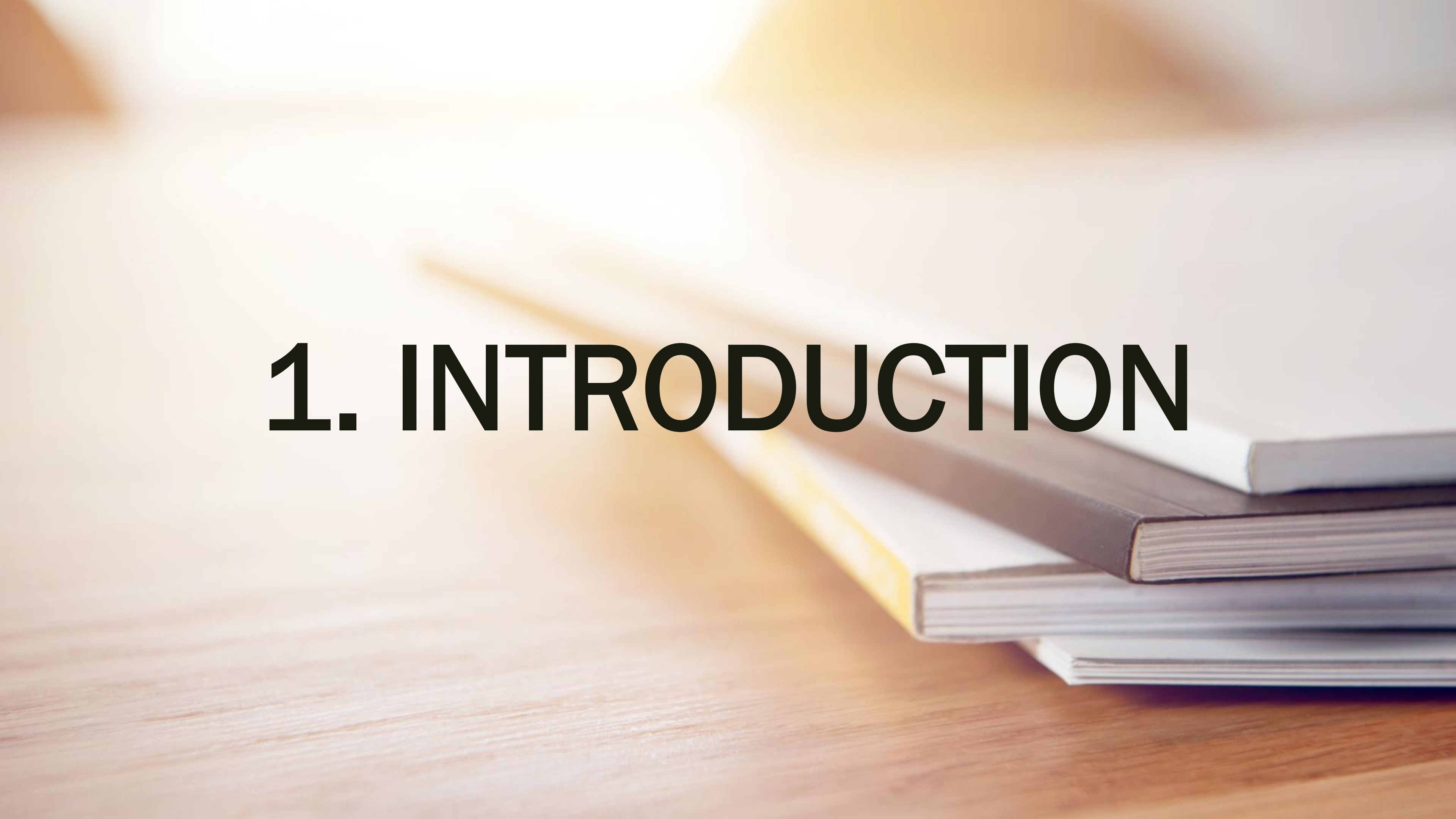
EN.520.612 MLSP Project

Cervical Spine Fracture Detection with Deep Learning

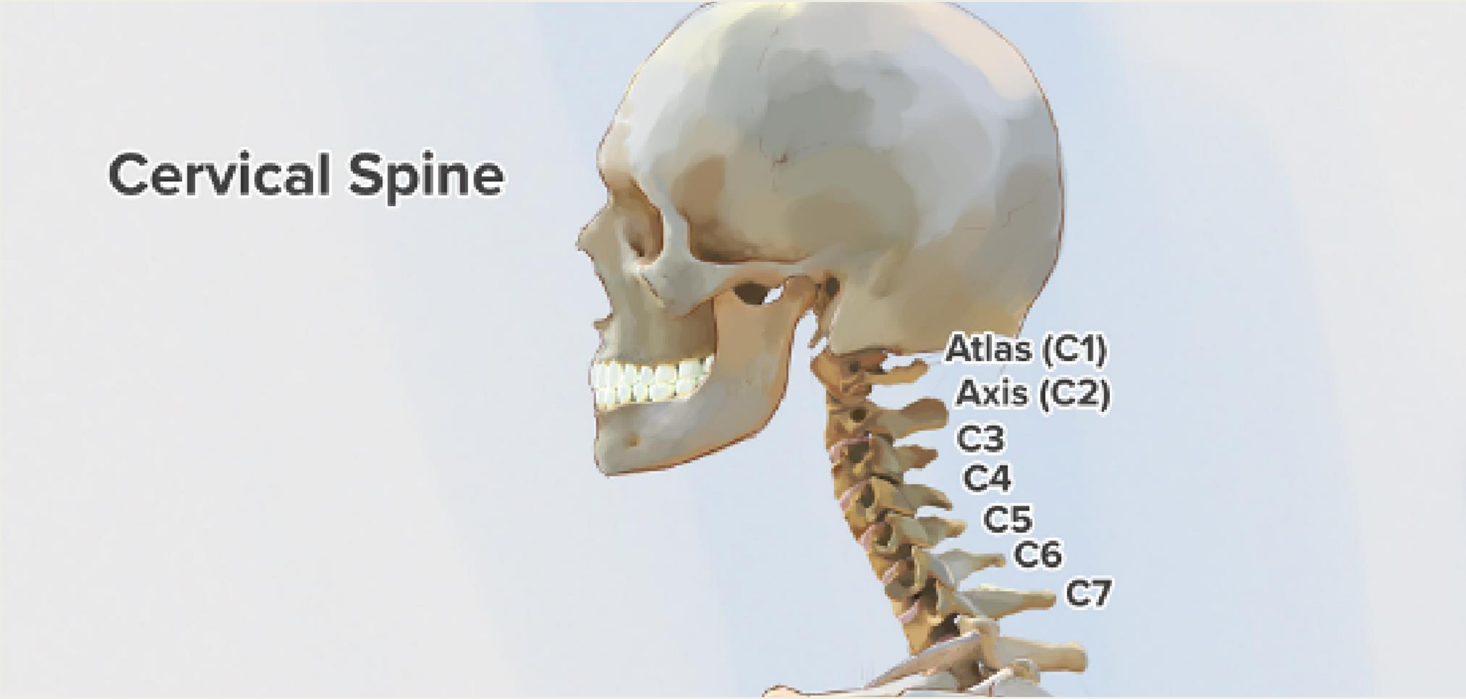
Team Members: Ruxiao Duan
Qihua Gong
Weichen Qi
Tingying Lu
Yifan Zhou

Contents



1. INTRODUCTION

1.1. Background



Seven vertebrae of a human cervical spine

1.2. Objectives

Main task:

- Identify fractures on human cervical spine with deep convolutional neural networks
  - on the slice-level
  - on the vertebra-level (C1-C7)
  - on the patient-level

Minor tasks:

- Detect the exact fracture position on a slice
- Extract the identified fractures and perform 3D visualization

1.3. Applications

- Early detection of cervical spine fractures
- Auto-diagnosis of relevant diseases
- Auto-extraction and visualization of cervical spine fractures

2. DATASET

2. Dataset

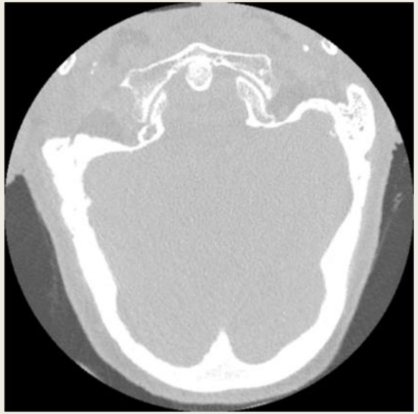
- Computed tomography (CT) scans collected from 12 sites globally by
  - Radiological Society of North America (RSNA)
  - American Society of Neuroradiology (ASNR)
  - American Society of Spine Radiology (ASSR)

- 2,019 CT studies (patients) available for use
  - 200-600 slices per patient
  - 711,601 CT images in total

2.1. CT Images (2,019 Patients)

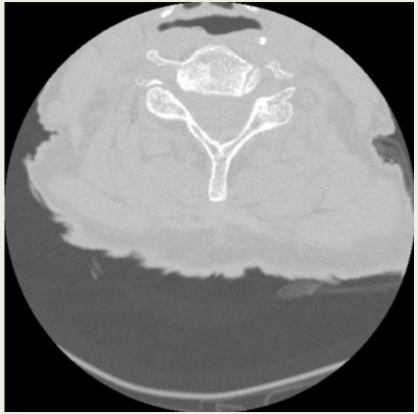
- CT scans:

Patient id: 17625



Slice 50

Slice 100

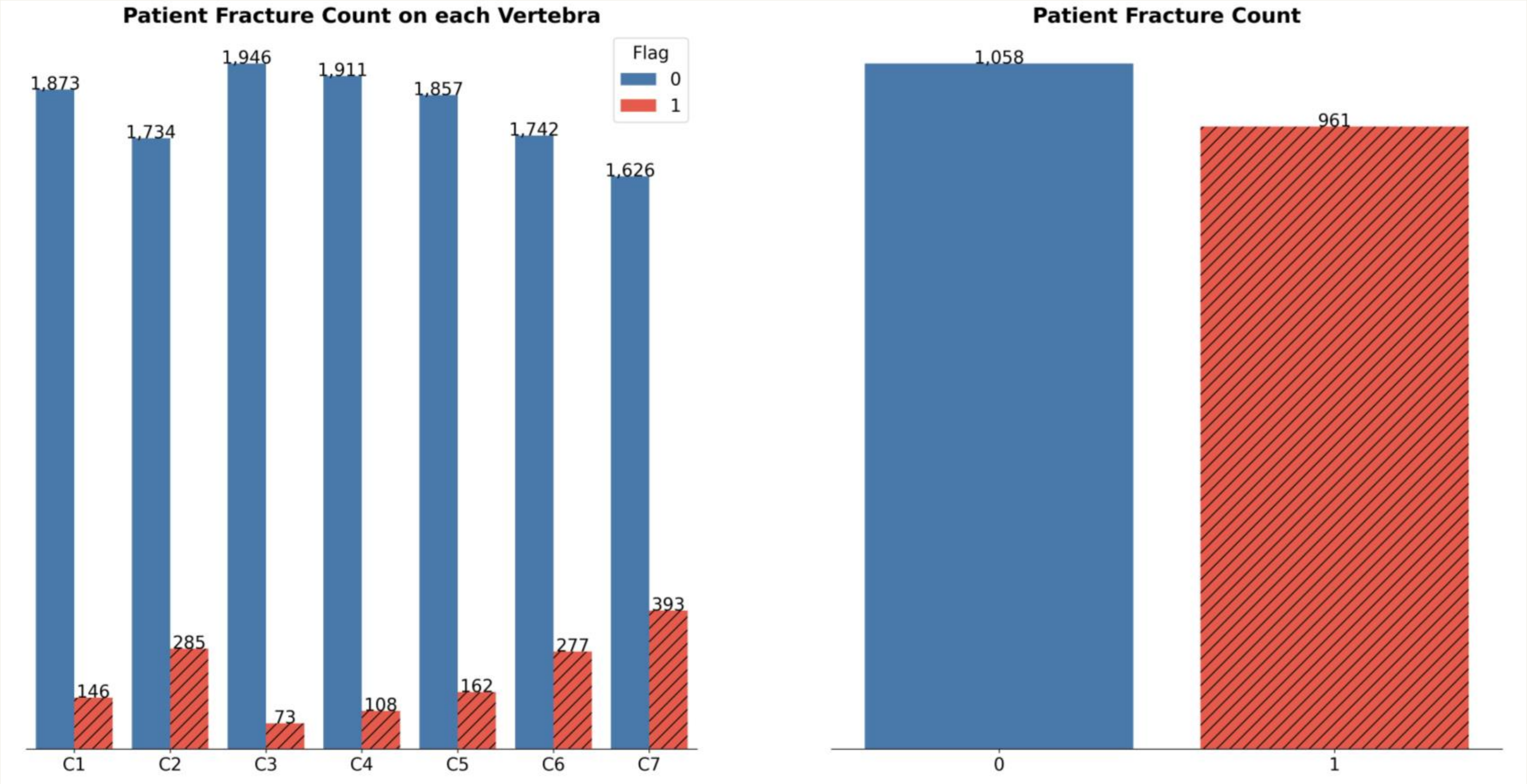


Slice 150

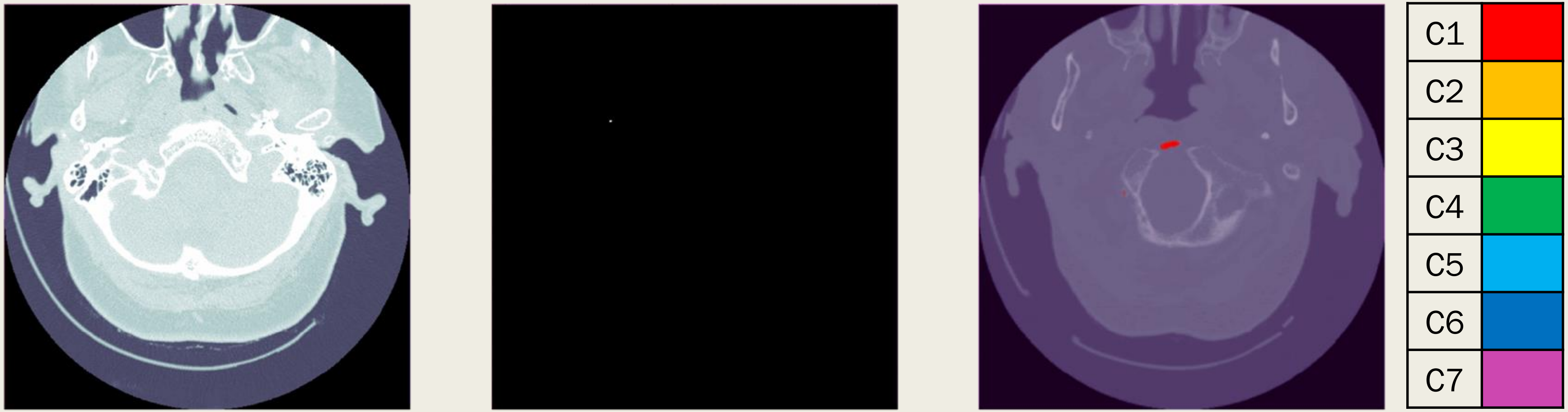
Slice 200

- Image metadata:
  - Image position (x, y, z coordinates)
  - Slice thickness

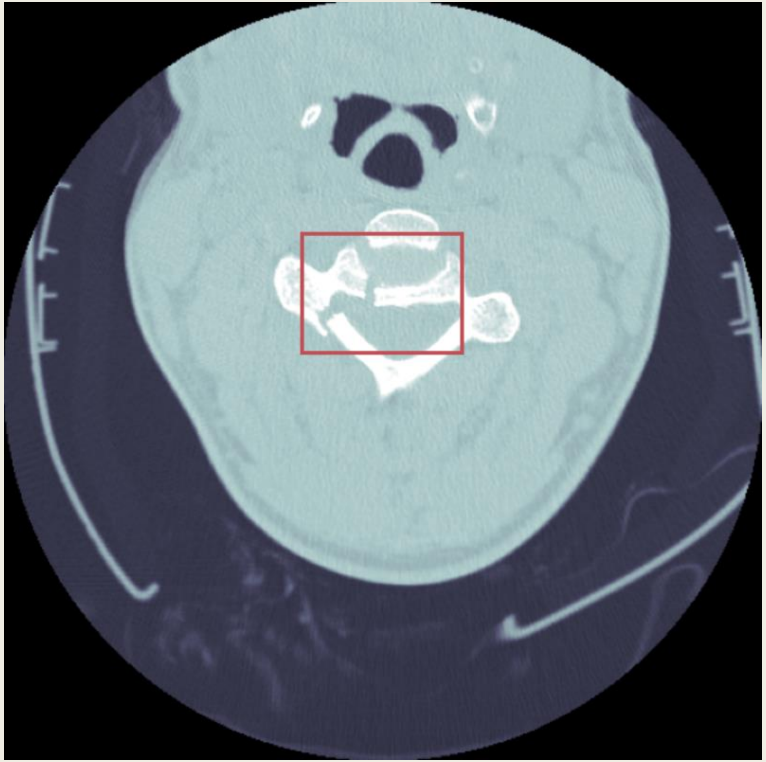
2.2. Fracture Labels (2,019 Patients)

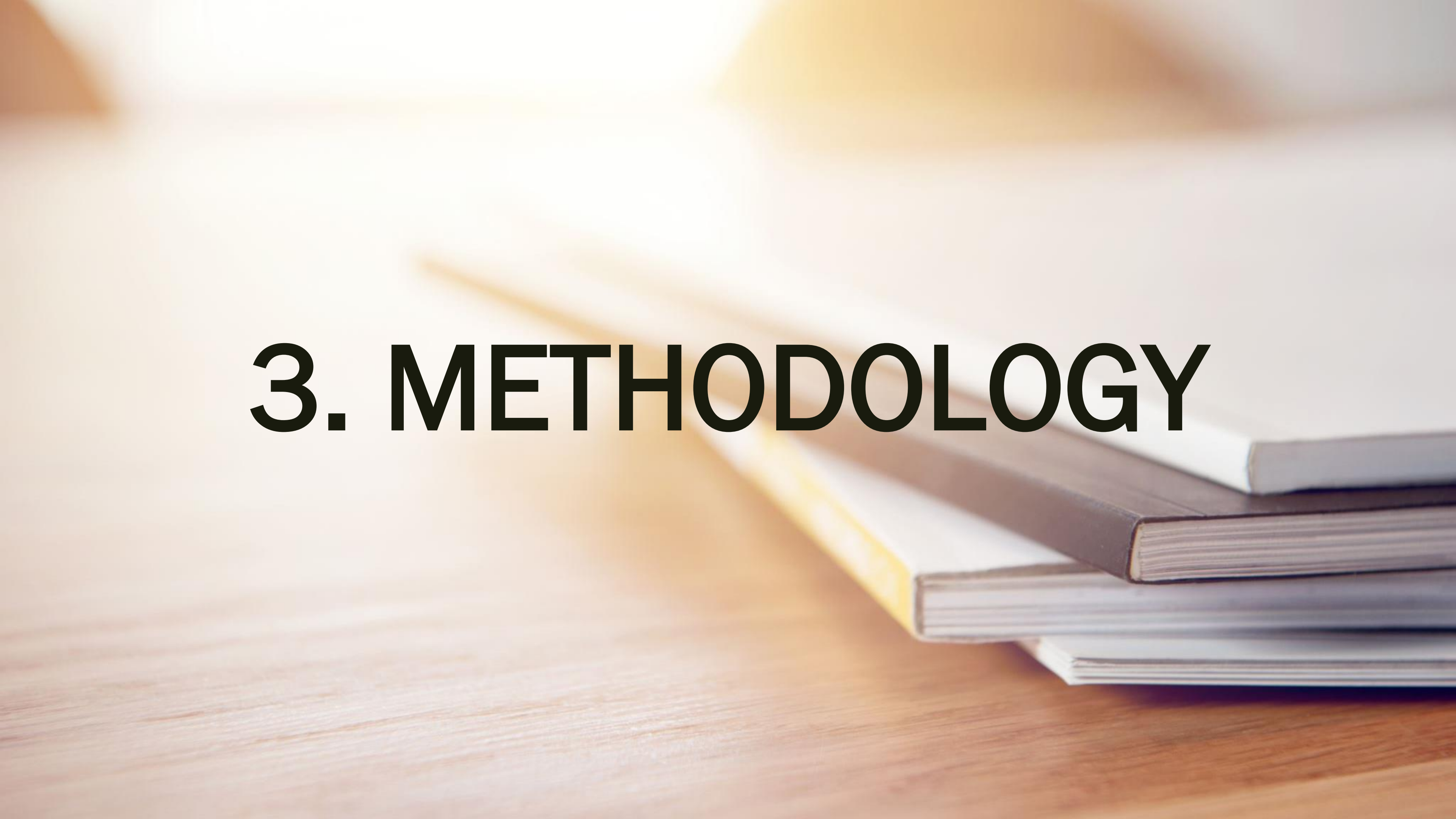


2.3. Pixel-level Vertebrae Labels (87 Patients)



2.4. Fracture Locations (239 Patients)





3. METHODOLOGY

3.1. Evaluation Metrics

- Vertebrae/fracture detection:

  Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

  Precision = $\frac{TP}{TP+FP}$

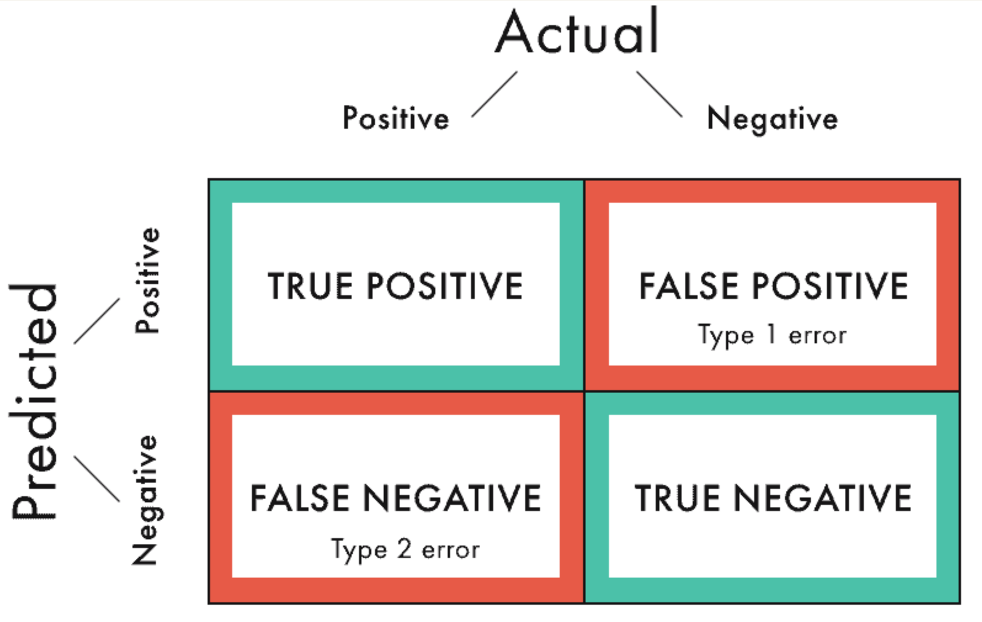
  Recall = $\frac{TP}{TP+FN}$

  F1-score = $2 * \frac{Precision*Recall}{Precision+Recall}$

  AUC: area under ROC curve

- Fracture localization:

  IoU: intersection over union

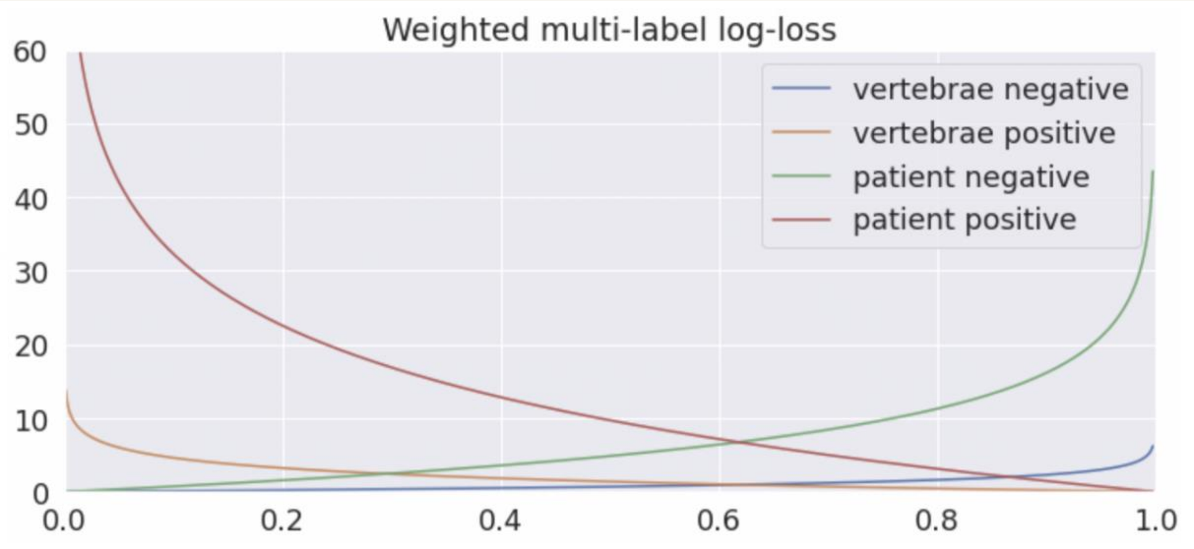


Confusion Matrix

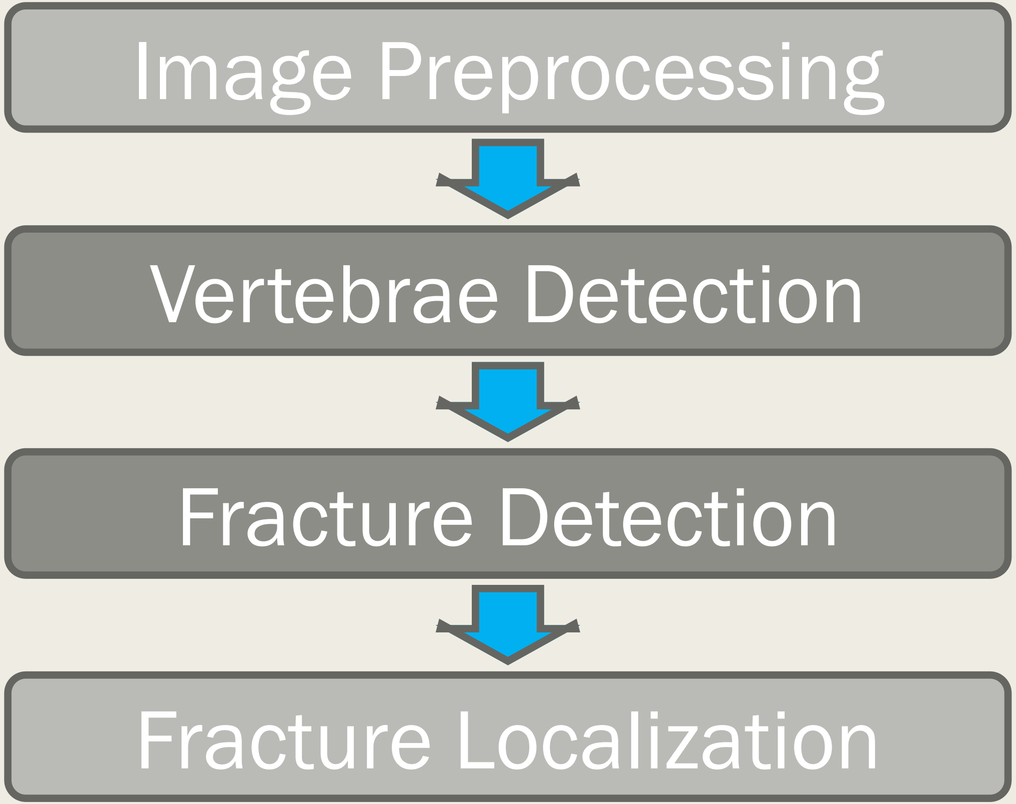
3.2. Loss Function

- Weighted binary cross-entropy

$$L = -w(y\log(p) + (1-y)\log(1-p)) \qquad w = \begin{cases} 1, & \text{if vertebra negative} \\ 2, & \text{if vertebra positive} \\ 7, & \text{if patient negative} \\ 14, & \text{if patient positive} \end{cases}$$



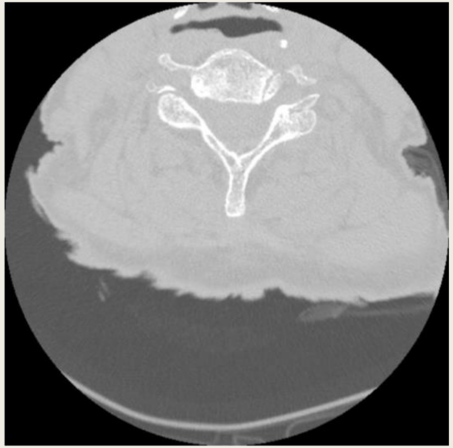
3.3. Project Pipeline



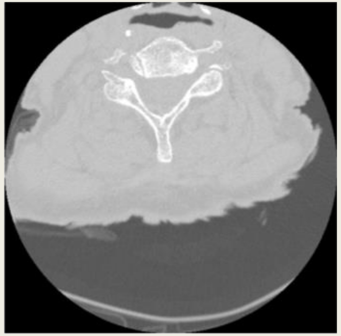
3.3.1. Image Preprocessing

- Resizing: resolution 512×512
- Rescaling: pixel value 0-255
- Data augmentation:
  - random rotation
  - random horizontal flip
  - random brightness, contrast, saturation and hue
- Noise reduction:
  - background detection by thresholding
  - image segmentation by K-means clustering

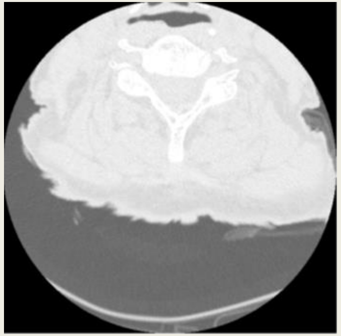
3.3.1. Image Preprocessing: Data Augmentation



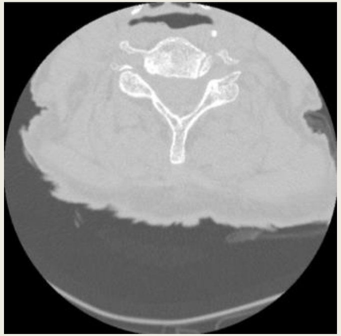
Original Image



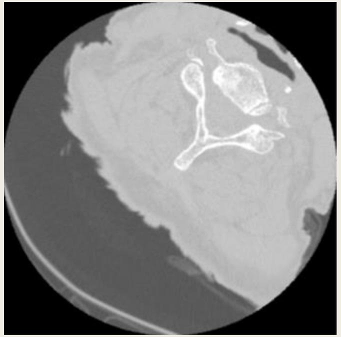
Horizontal Flip



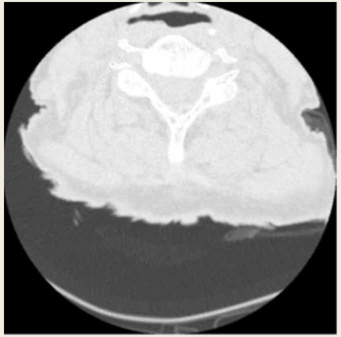
Brightness



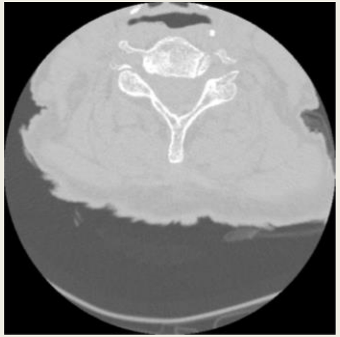
Saturation



Rotation

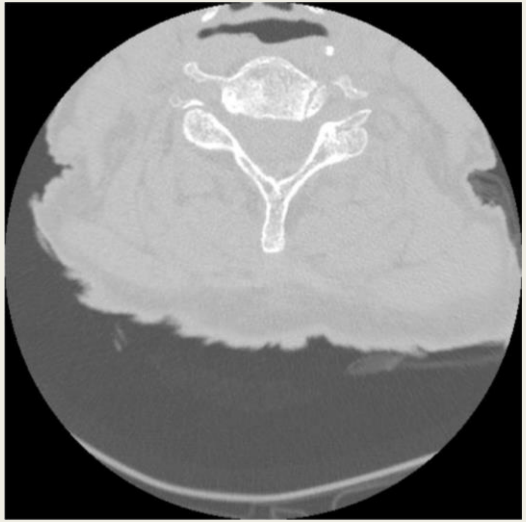


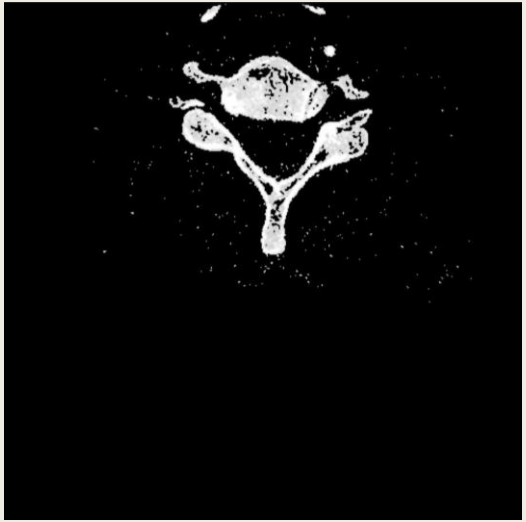
Contrast

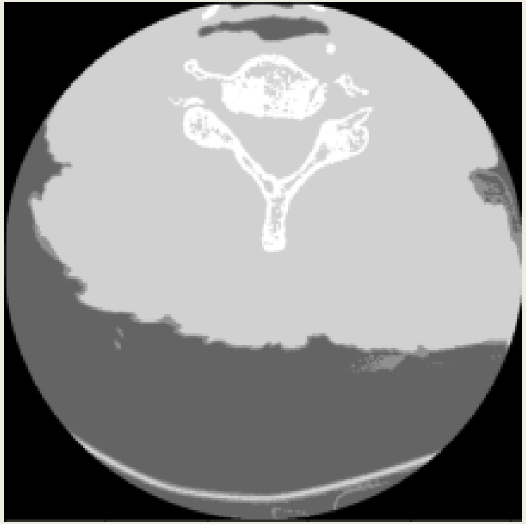


Hue

3.3.1. Image Preprocessing: Noise Reduction

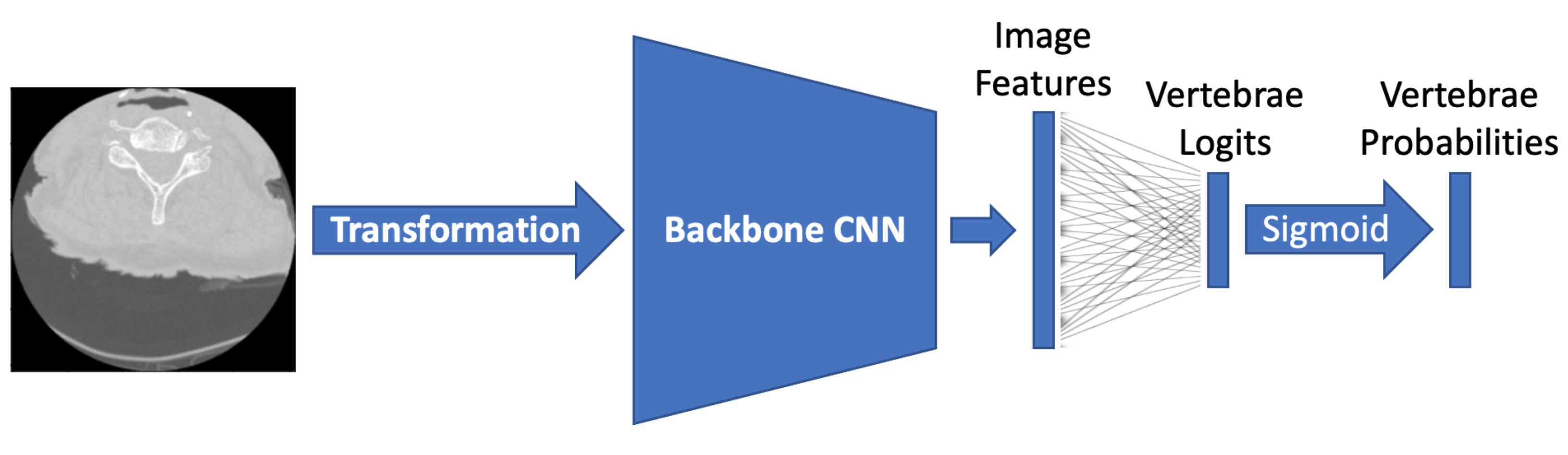

Original Image


Thresholding


K-Means Segmentation

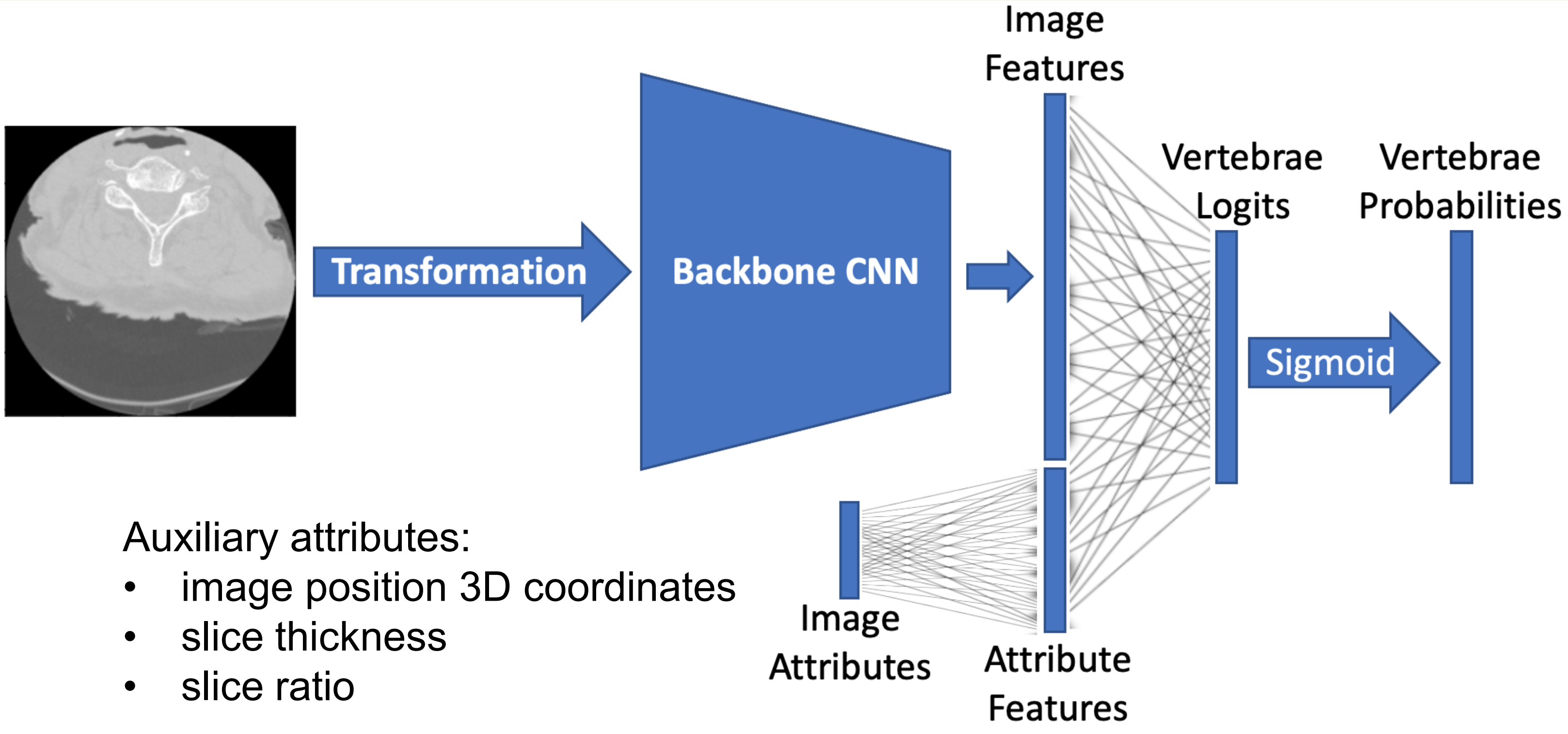
3.3.2. Vertebrae Detection: Type 1 Model

- Image input only



3.3.2. Vertebrae Detection: Type 2 Model

- Image and metadata inputs [multimodal learning]

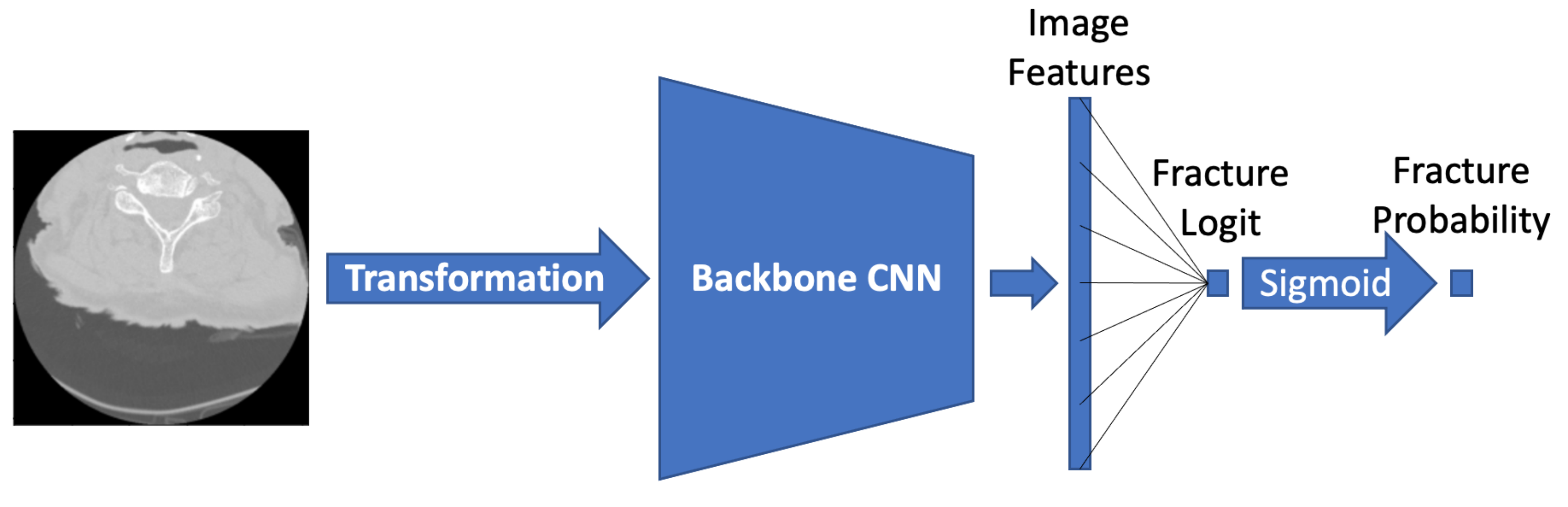


Auxiliary attributes:

- image position 3D coordinates
- slice thickness
- slice ratio

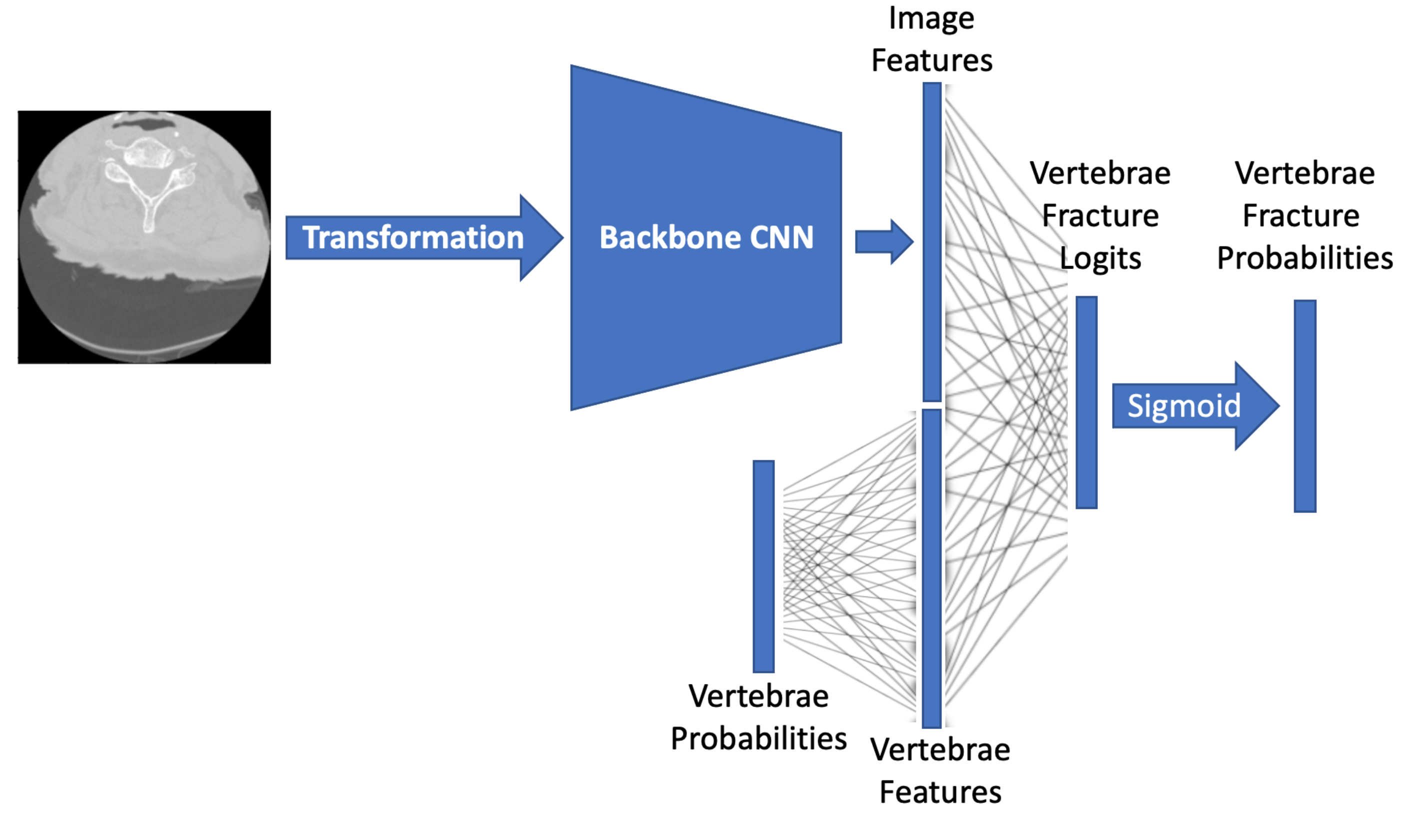
3.3.3. Fracture Detection: Type 1 Model

- Single probability output



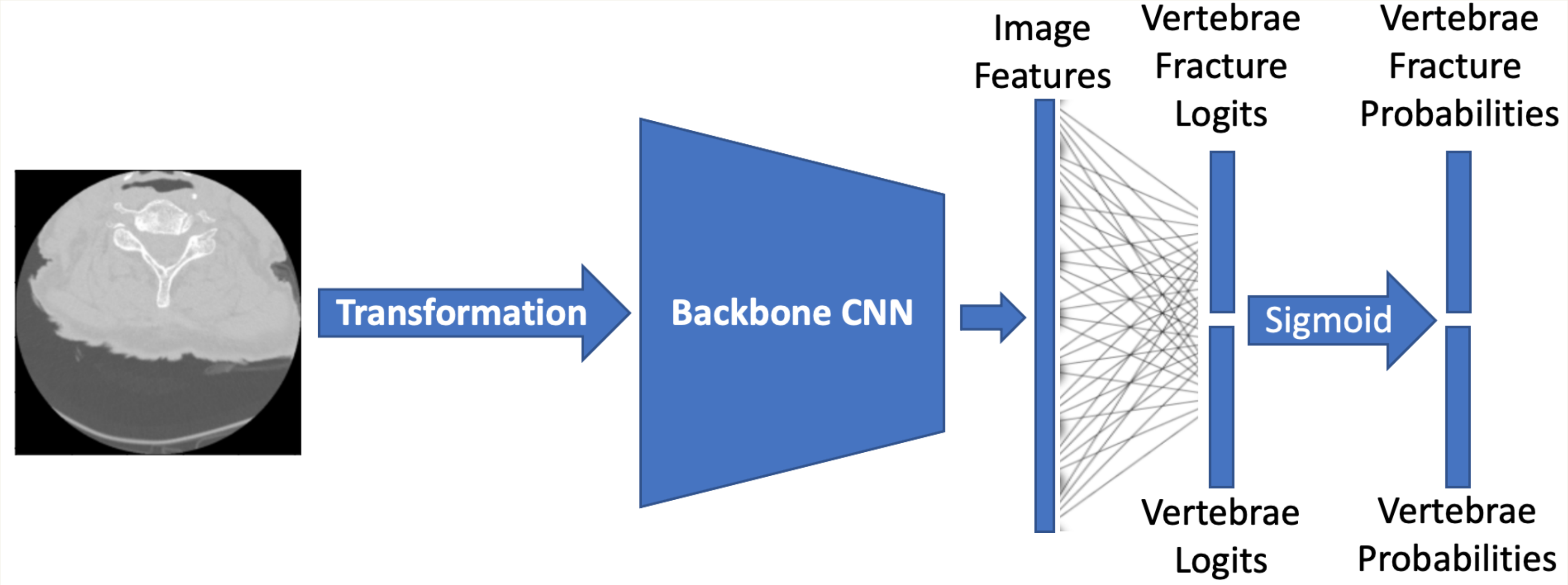
3.3.3. Fracture Detection: Type 2 Model

- Vertebrae probabilities input [multimodal learning]



3.3.3. Fracture Detection: Type 3 Model

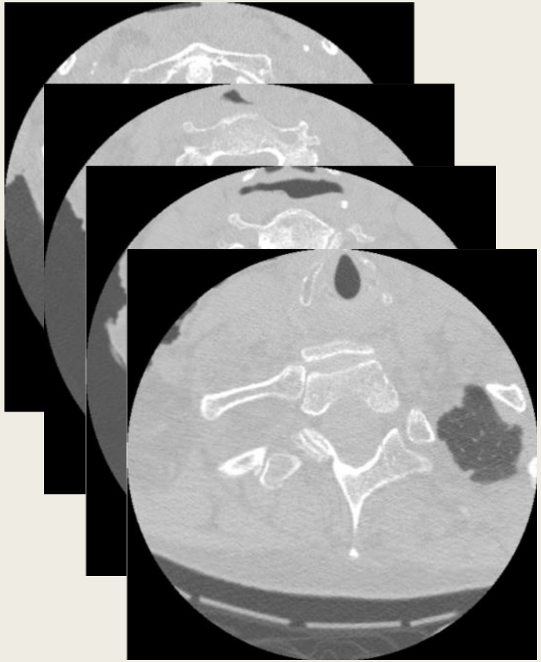
- Vertebrae probabilities output [multi-task learning]



- Combined loss function:
  - $L = \alpha L_{frac} + L_{vert}$ (α: relative weighting)

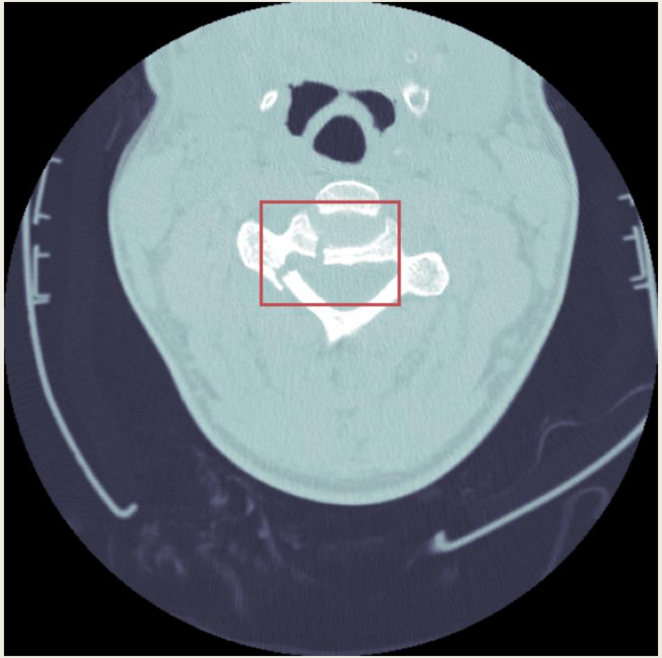
3.3.3. Fracture Prediction

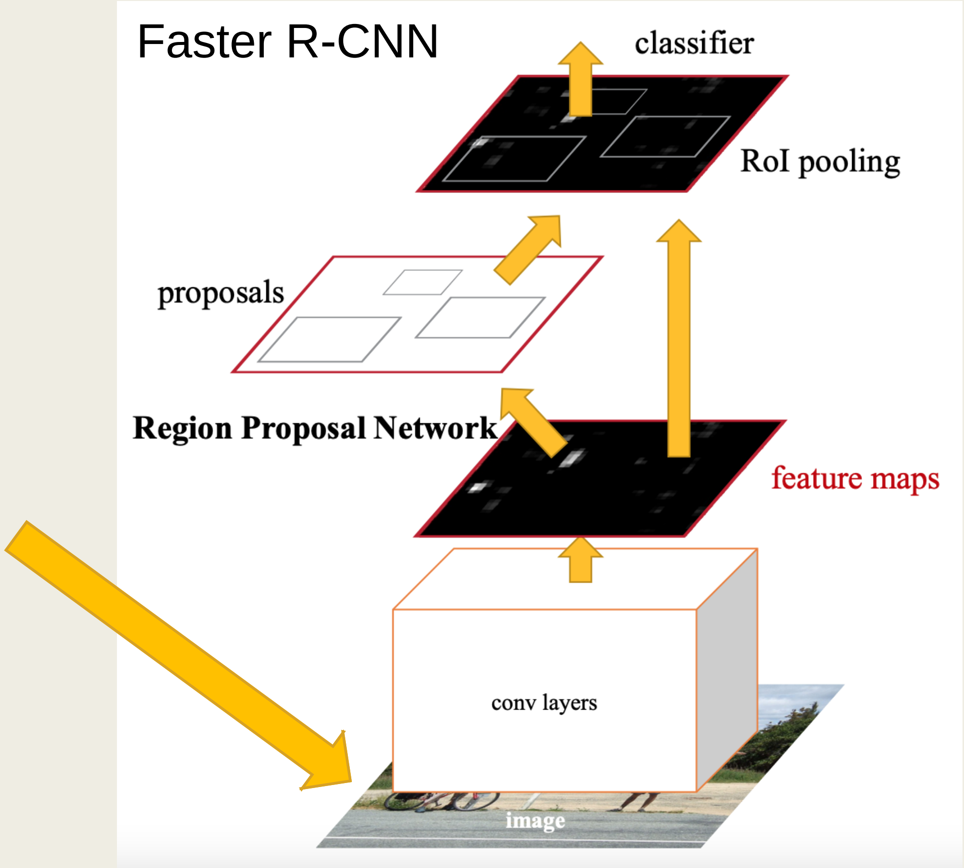
- Slice-level fracture prediction
  - Type 1 model:
    - p_i is the model output
  - Type 2 & 3 model:
    - $p_i = \frac{\sum_{j=1}^{7} f_{ij} v_{ij}}{\sum_{j=1}^{7} v_{ij}}$
- Vertebra-level fracture prediction
  - Type 1 model:
    - $p_j = \frac{\sum_{i=1}^{n} f_i v_{ij}}{\sum_{i=1}^{n} v_{ij}}$
  - Type 2 & 3 model:
    - $p_j = \frac{\sum_{i=1}^{n} f_{ij} v_{ij}}{\sum_{i=1}^{n} v_{ij}}$
- Patient-level fracture prediction
  - $p = 1 - \prod_{j=1}^{7} (1 - p_j)$



i: slice index
j: vertebra index
v: vertebra probability
f: slice fracture probability
p: predicted fracture probability/score

3.3.4. Fracture Localization





4. EXPERIMENTS

4.1. Data Preprocessing

- Data augmentation:
  - Negligible improvement
  - Not that beneficial in single-epoch training

- Noise reduction:
  - Ineffective
  - Model performance degrades significantly
  - More sophisticated techniques needed

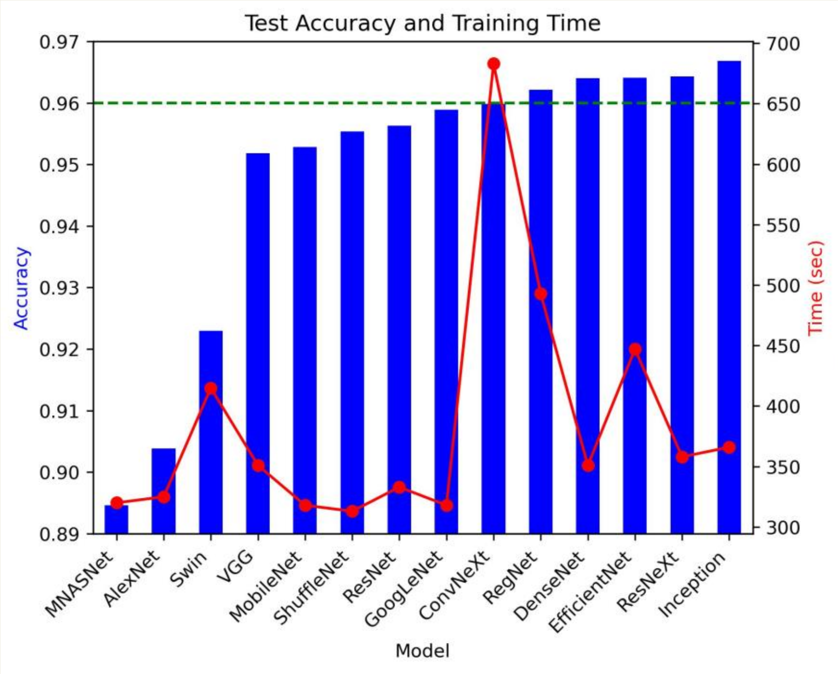
4.2. Vertebrae Detection: Backbone Model Selection

Selected backbone CNN models:

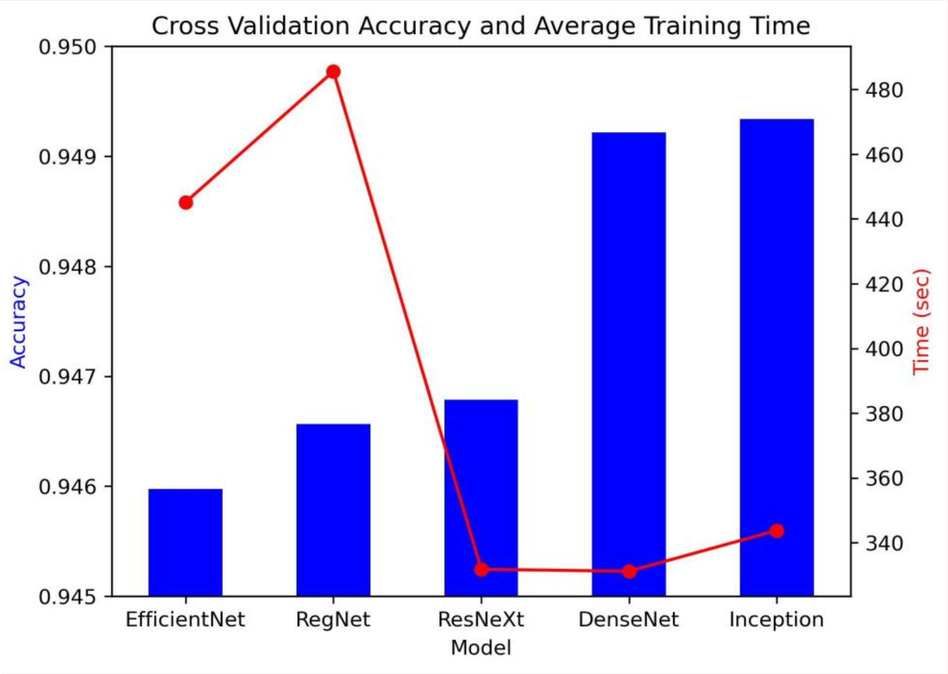
- AlexNet
- VGG16 (with batch normalization)
- ResNet50
- ResNeXt50 (32 × 4D)
- GoogLeNet
- InceptionV3
- DenseNet121
- EfficientNetV2 (small)
- MobileNetV3 (large)
- ConvNeXt (small)
- MNASNet (with a depth multiplier of 1.3)
- ShuffleNetV2 (with 2.0 × output channels)
- Swin Transformer (small)
- RegNetY-8GF

4.2. Vertebrae Detection: Backbone Model Selection

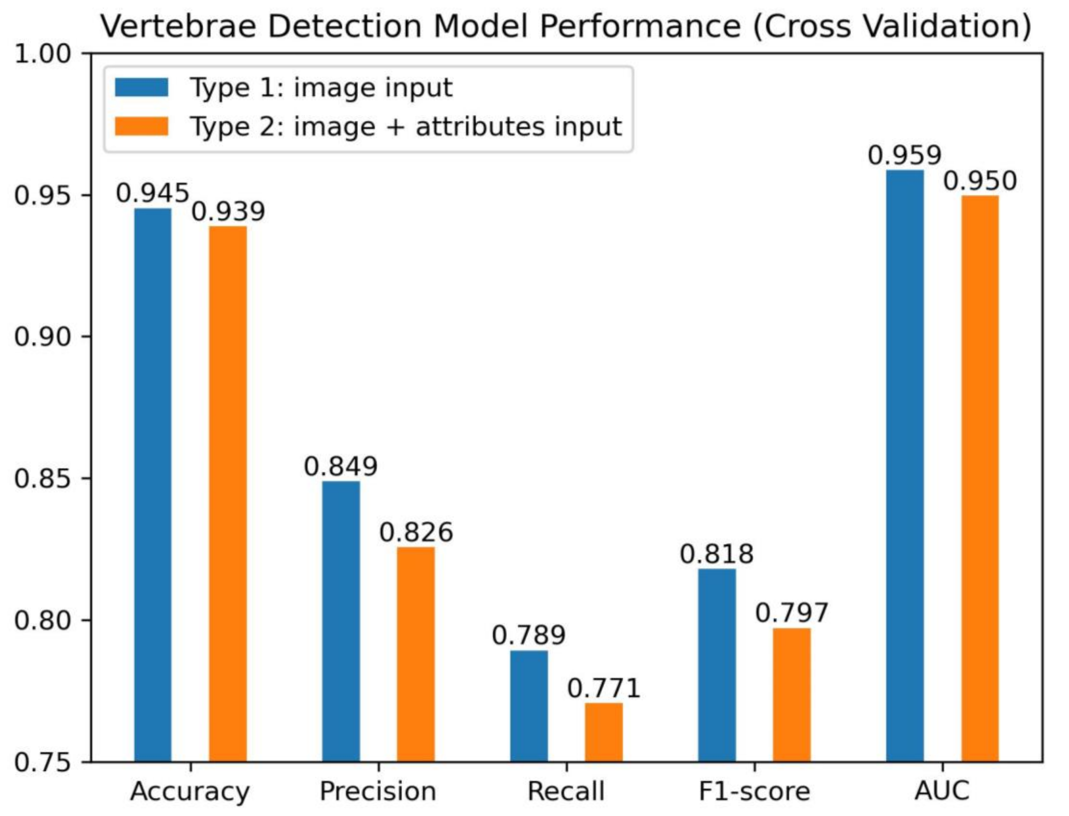
- Single validation test (train-test ratio 8:2)



- 5-fold cross validation



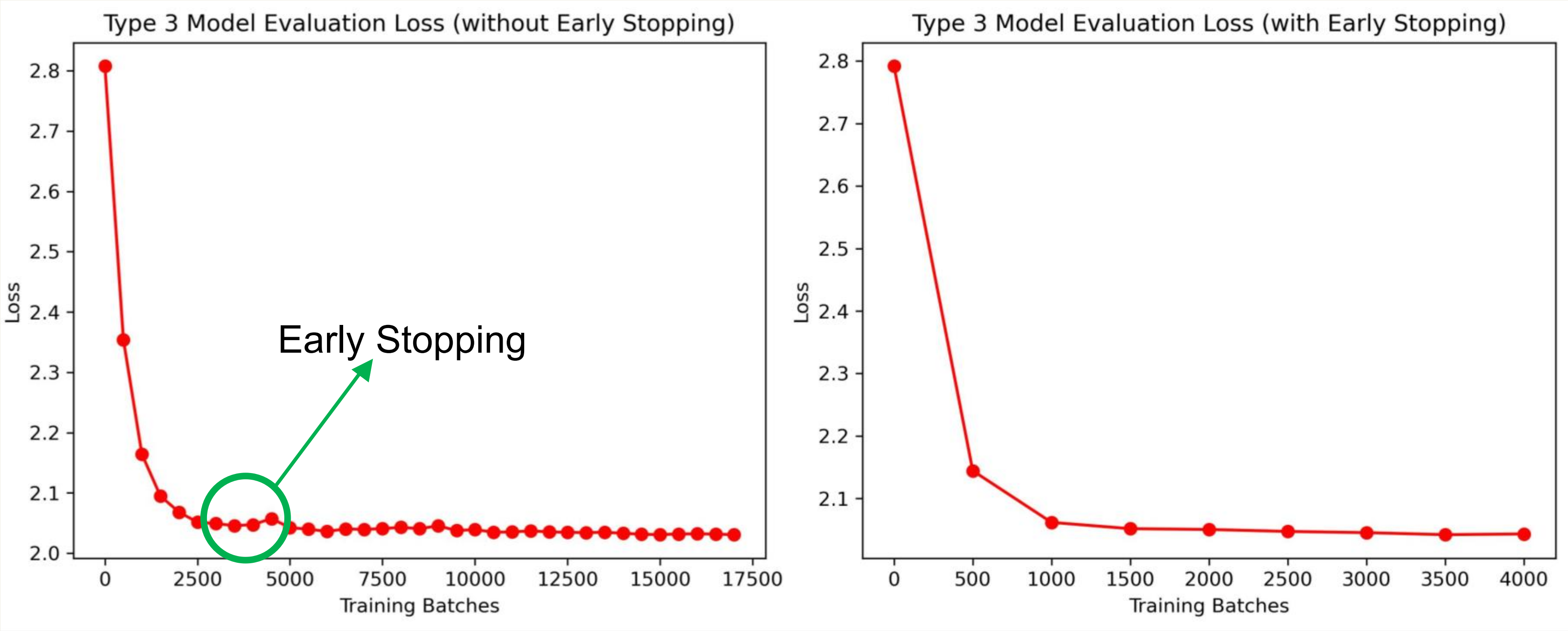
4.2. Vertebrae Detection: Type 1/2 Model Comparison



- InceptionV3 backbone
- 5-fold cross validation

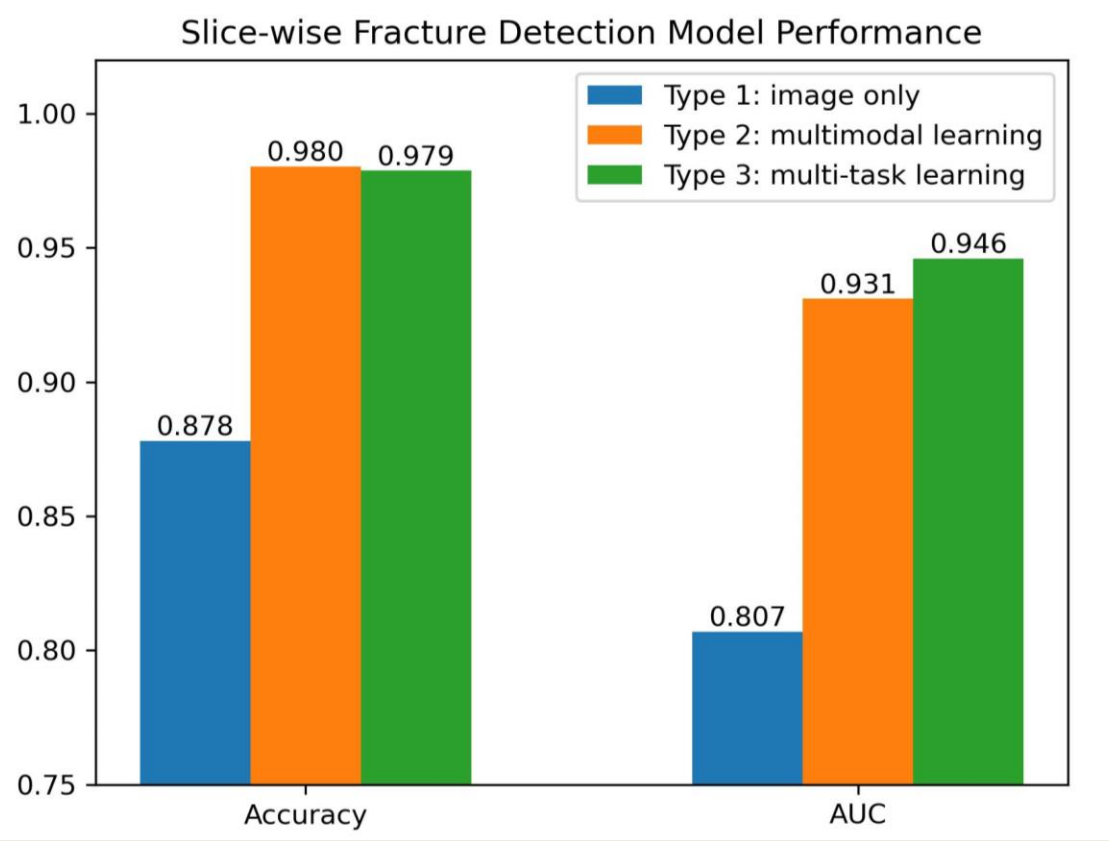
- Performance: Type 1 > Type 2
- Auxiliary attributes (slice ratio, image position, etc.) did not further bring useful information for vertebrae detection.

4.3. Fracture Detection: Evaluation Loss



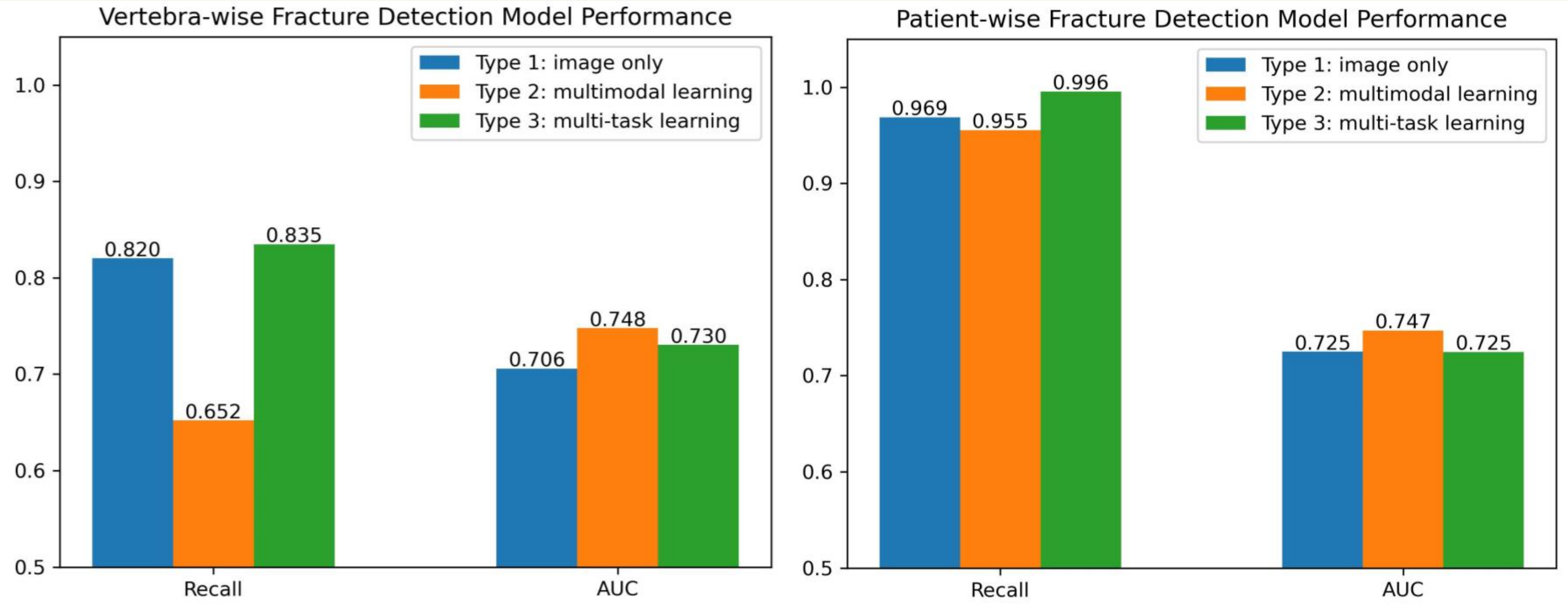
❖ Early stopping demonstrated a positive effect on fracture detection model training.

4.3. Fracture Detection (Slice-wise)



- Performance:
  Type 2 ≈ Type 3 >> Type 1
- Type 2 & 3 models take multiple vertebrae into consideration for slice fracture detection.

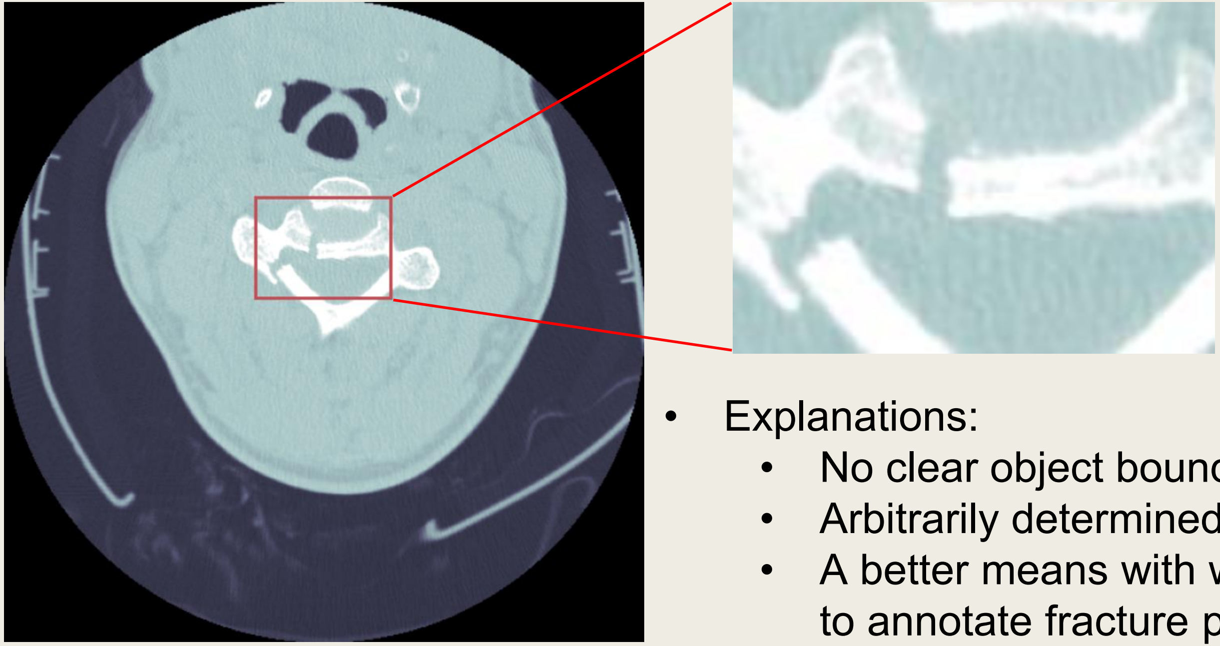
4.3. Fracture Detection (Vertebra/Patient-wise)



- Decision threshold was determined by highest f1-score.
- Models showed a lower recall for vertebra-level fracture detection.
- 99.6% patients with fracture can be successfully detected using Type 3 model.

4.4. Fracture Localization

- Faster R-CNN with ResNet-50-FPN backbone
- Weighted boxes fusion applied to merge multiple matches
- Failed to locate fractures: validation IoU 0.100



- Explanations:
  - No clear object boundary for a fracture
  - Arbitrarily determined bounding boxes
  - A better means with well-defined criterion to annotate fracture position is needed

4.5. Implementation Details

- PyTorch for deep learning model construction
- GPU for model training acceleration
- Adam used as optimizer
- Single-epoch model training
- One-cycle learning rate policy adopted for learning rate scheduling
  - Maximum learning rate 1e-3 for vertebrae detection
  - Maximum learning rate 1e-4 for fracture detection
  - Percentage of learning rate increase set to 0.3
- Early stopping applied for fracture detection model training
  - Loss decrease threshold 0.01
  - Patience 5 × 500 batches
- Relative weighting $\alpha = 2$ in combined loss function
- Group K-fold cross validation employed to avoid information leakage
- Parametric ReLU chosen as the activation layer of DNN
- Automatic mixed precision training leveraged to speed up model training and save memory

5. DISCUSSION

5.1. Difficulties and Limitations

- Large-scale dataset (350 GiB):
  - Slow data preprocessing
  - Hyperparameter tuning too costly
  - Cannot train models for multiple epochs

- Insufficient GPU memory:
  - Backbone models restricted to lightweight architectures
  - Limited batch size and slow training process
  - Inefficient to train a 3D CNN

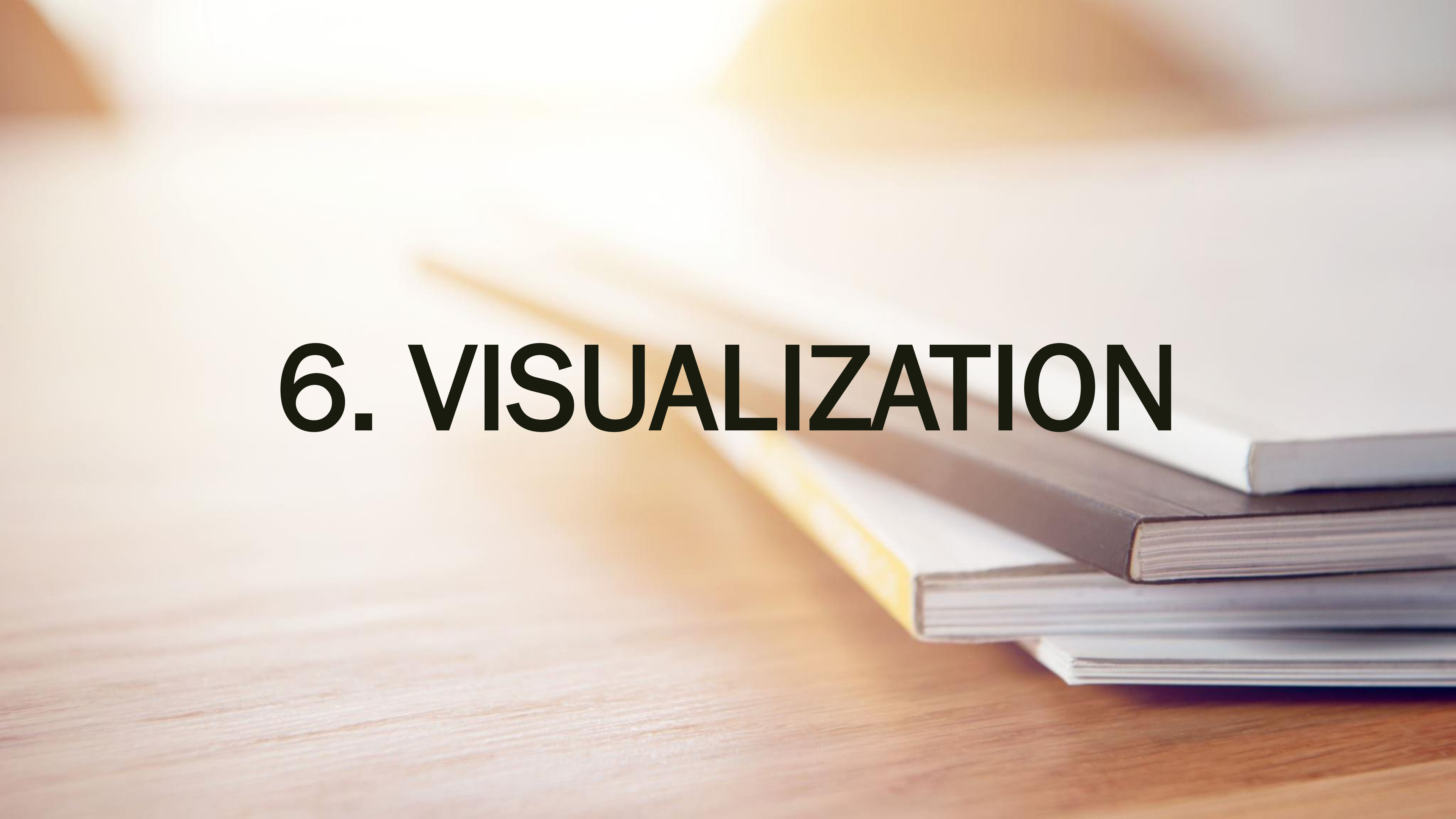
5.2. Result Summary

- Vertebrae detection:
  - accuracy 0.945, AUC 0.959
- Fracture detection:
  - slice-level: accuracy 0.980, AUC 0.946
  - vertebra-level: recall 0.835, AUC 0.748
  - patient-level: recall 0.996, AUC 0.747
- Fracture localization:
  - IoU 0.100 (failed)

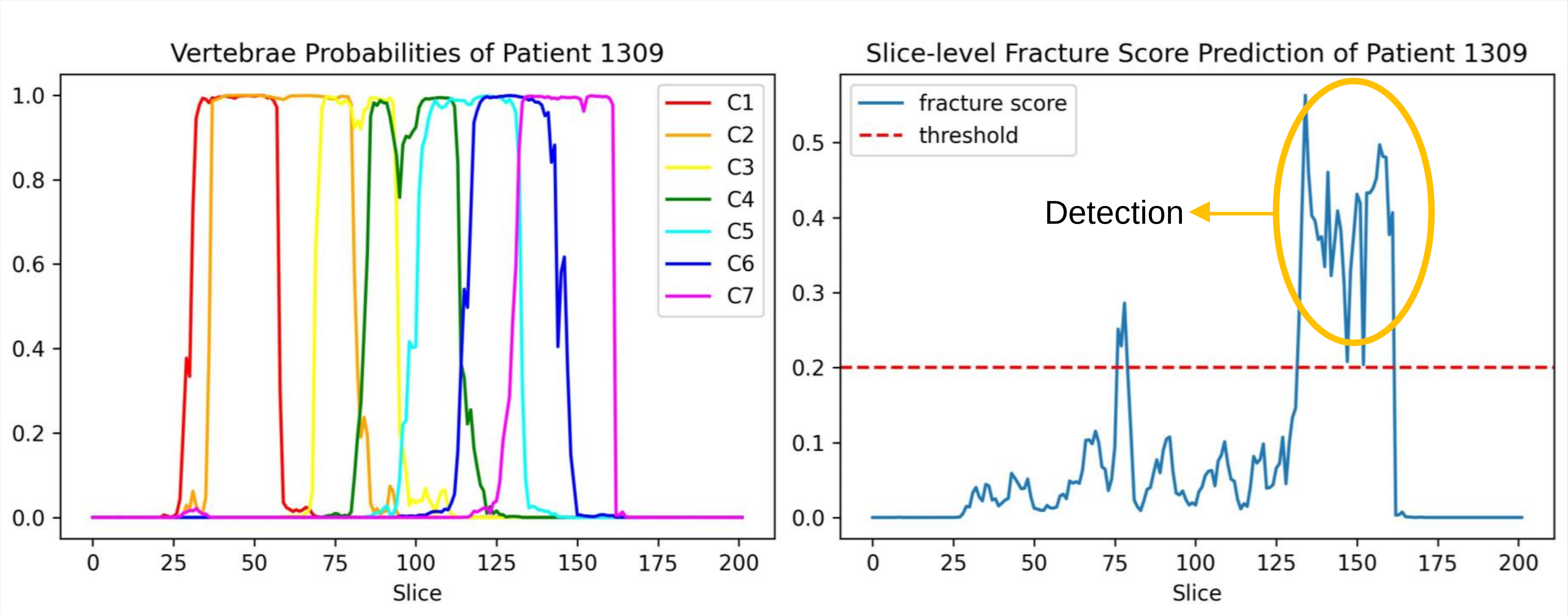
5.3. Future Work

- Image segmentation with advanced models
- Multi-epoch model training for fracture detection
- Hyperparameter tuning
- Model architecture optimization
- 3D CNN for fracture detection
- Fracture localization model design

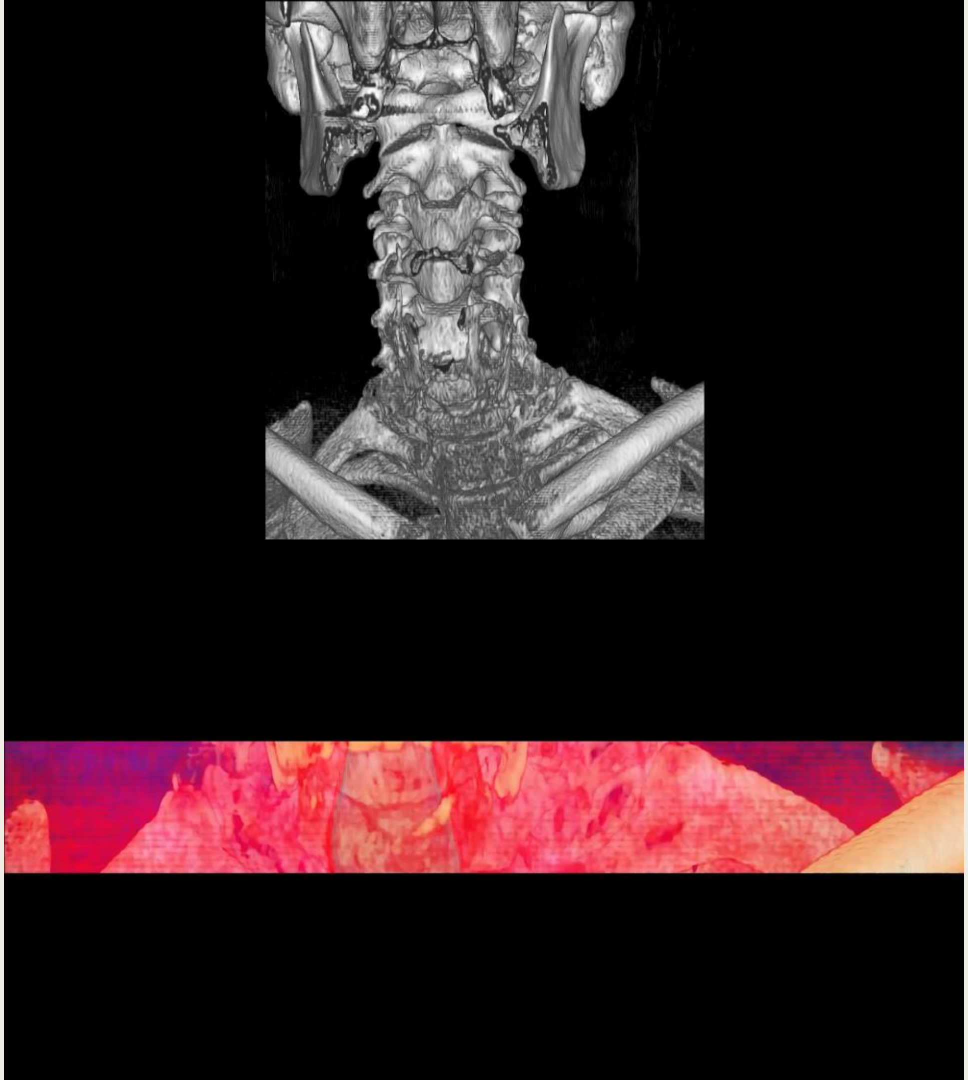
6. VISUALIZATION

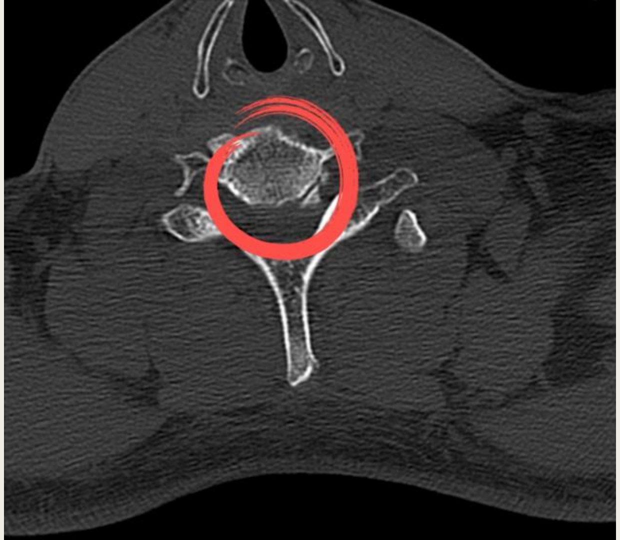


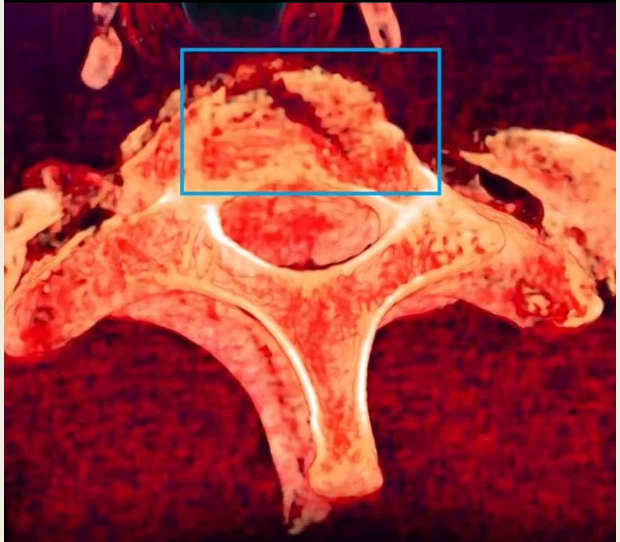
6. Visualization

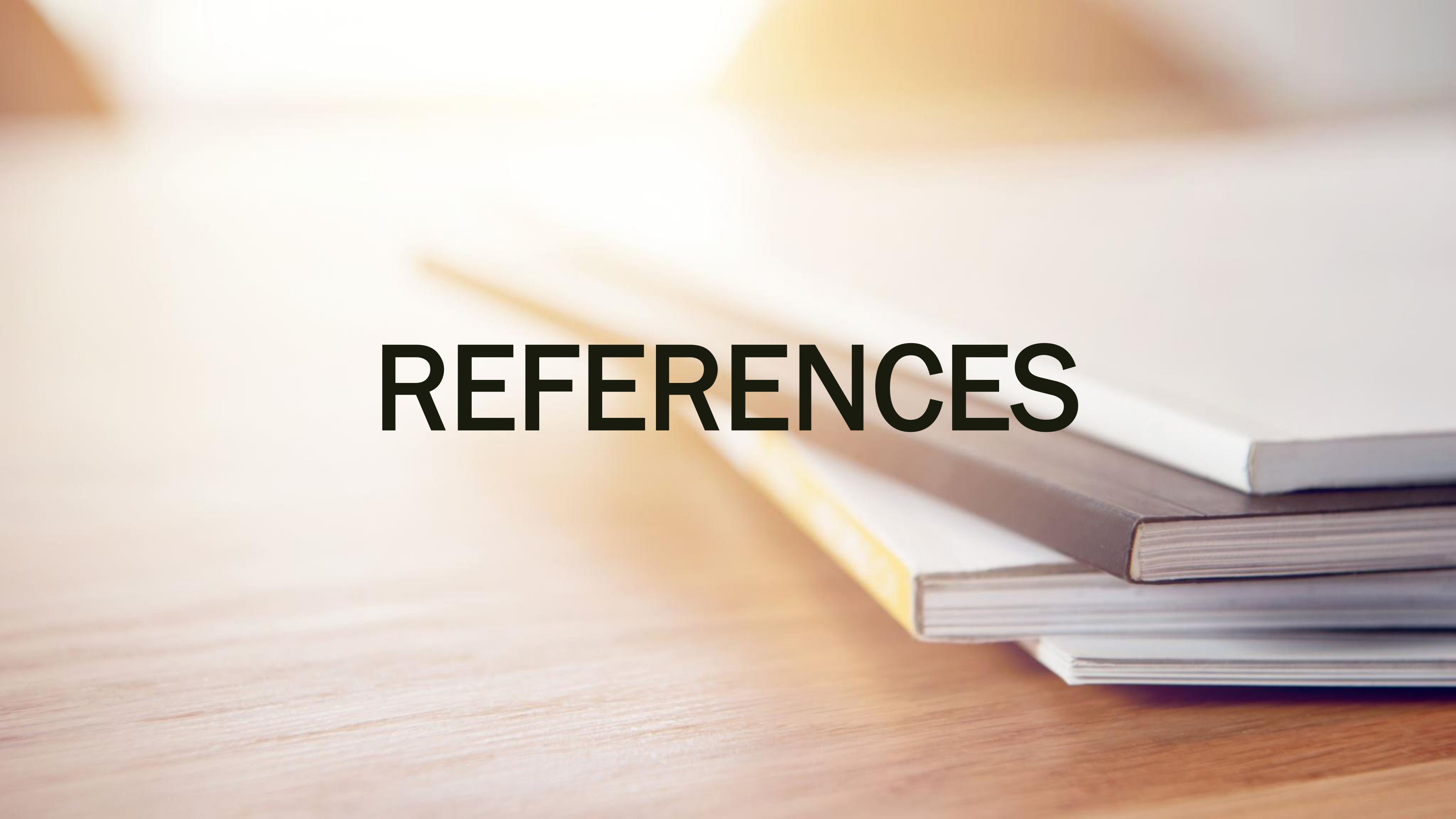


- Fracture approximately located at vertebrae C6 and C7








REFERENCES

References

[1] D. Shen, G. Wu, and H.-I. Suk, “Deep learning in medical image analysis,” Annu. Rev. Biomed. Eng., vol. 19, no. 1, pp. 221–248, 2017.

[2] G. Litjens et al., “A survey on deep learning in medical image analysis,” Medical Image Analysis, vol. 42, pp. 60–88, 2017.

[3] J. Cho, K. Lee, E. Shin, G. Choy, and S. Do, “How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?,” arXiv [cs.LG], 2015.

[4] Y. Li et al., “A comprehensive review for MRF and CRF approaches in pathology image analysis,” arXiv [cs.CV], 2020.

[5] A. N. Basavanhally et al., “Computerized image-based detection and grading of lymphocytic infiltration in HER2+ breast cancer histopathology,” IEEE Trans. Biomed. Eng., vol. 57, no. 3, pp. 642–653, 2010.

[6] J. Bioucas-Dias, F. Condessa, and J. Kovacevic, Alternating direction optimization for image segmentation using hidden Markov measure field models In: Gurcan MN, Madabhushi A, editors. IS&T/SPIE electronic imaging. International Society for Optics and Photonics. 2014.

[7] A. Van Engelen K.Á, “Three-dimensional carotid ultrasound plaque texture predicts vascular events,” Stroke, vol. 45, no. 9, pp. 2695–2701, 2014.

[8] H. C. Achterberg et al., “Hippocampal shape is predictive for the development of dementia in a normal, elderly population: Hippocampal Shape is Predictive for Dementia,” Hum. Brain Mapp., vol. 35, no. 5, pp. 2359–2371, 2014.

[9] M. de Bruijne, “Machine learning approaches in medical image analysis: From detection to diagnosis,” Med. Image Anal., vol. 33, pp. 94–97, 2016.

[10] J. Zhuang, J. Cai, R. Wang, J. Zhang, and W.-S. Zheng, “Deep kNN for Medical Image Classification,” in Medical Image Computing and Computer Assisted Intervention – MICCAI 2020, Cham: Springer International Publishing, 2020, pp. 127–136.

References

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, 2017.
[12] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv [cs.CV], 2014.
[13] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” arXiv [cs.CV], 2015.
[14] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional Neural Networks,” arXiv [cs.LG], 2019.
[15] S. Ruder, “An overview of gradient descent optimization algorithms,” arXiv [cs.LG], 2016.
[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv [cs.LG], 2014.
[17] F. Zhuang et al., “A comprehensive survey on transfer learning,” arXiv [cs.LG], 2019.
[18] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, “Multimodal machine learning: A survey and taxonomy,” arXiv [cs.LG], 2017.
[19] S. Ruder, “An overview of multi-task learning in deep neural networks,” arXiv [cs.LG], 2017.
[20] L. N. Smith and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates,” arXiv [cs.LG], 2017.
[21] “RSNA 2022 cervical spine fracture detection,” Kaggle.com. [Online]. Available: https://www.kaggle.com/competitions/rsna-2022-cervical-spine-fracture-detection/data. [Accessed: 03-Dec-2022].
[22] P. Micikevicius et al., “Mixed Precision Training,” arXiv [cs.AI], 2017.
[23] R. Solovyev, W. Wang, and T. Gabruseva, “Weighted boxes fusion: Ensembling boxes from different object detection models,” Image Vis. Comput., vol. 107, no. 104117, p. 104117, 2021.

Q & A


THANK YOU