Deep Learning Approaches for Detecting Absence of Lung Sliding in Ultrasound Videos

1st Roman Dzhulai
Technical University of Košice
Košice, Slovakia
roman.dzhulai@student.tuke.sk

2nd Ivan Tkachenko
Technical University of Košice
Košice, Slovakia
ivan.tkachenko@student.tuke.sk

3rd Nikita Pohorilyi
Technical University of Košice
Košice, Slovakia
nikita.pohorilyi@student.tuke.sk

4th Dmytro Varich
Technical University of Košice
Košice, Slovakia
dmytro.varich@student.tuke.sk

5th Dmytro Marchuk
Technical University of Košice
Košice, Slovakia
dmytro.marchuk@student.tuke.sk

Mentor: Maroš Hliboký
Technical University of Košice
Košice, Slovakia
maros.hliboky@tuke.sk

***Abstract*—Accurately diagnosing pneumothorax, a potentially life-threatening condition characterized by the accumulation of air in the pleural space, is crucial for effective patient care. The identification of lung sliding in ultrasound videos is a key component of pneumothorax assessment, as it indicates the presence of normal pleural movement during respiration. However, manual interpretation of these videos is challenging and time-consuming, requiring expertise and careful observation. To address this, we propose a novel hybrid deep learning architectures that integrate optical flow analysis with Convolutional Neural Networks (CNNs) for automated lung sliding classification. Through extensive experimentation, leveraging both state-of-the-art pre-trained models like RAFT and custom-designed convolutional and recurrent architectures, we achieve robust performance, paving the way for more efficient and accurate pneumothorax diagnosis.**

***Index Terms*—Pneumothorax Diagnosis, Lung Sliding, Deep Learning, Optical Flow, CNN, 3D Convolution, Recurrent Neural Network**

I. Introduction

Pneumothorax, characterized by the presence of air in the pleural cavity, presents a significant clinical challenge due to its potential to impair respiratory function and lead to life-threatening complications. [1] The assessment of pneumothorax often involves the evaluation of lung sliding, a physiological phenomenon where the visceral and parietal pleura glide smoothly over each other during respiration, indicating the absence of air in the pleural space. [2] However, accurate identification of lung sliding in ultrasound videos requires expertise and may be subject to interobserver variability.

In this context, the development of automated systems for lung sliding classification holds promise for improving the efficiency and reliability of pneumothorax diagnosis. Our proposed deep learning architecture combines optical flow analysis, which captures the temporal dynamics of lung motion, with CNNs to extract meaningful features for classification.

The remainder of this paper is organized as follows: Section II offers an extensive review of previous work related to pneumothorax diagnosis and deep learning-based image analysis. In Section III, we delve into the intricacies of our proposed hybrid architectures, elucidating our approach to model design, and training strategies. Additionally, Section IV provides insights into our dataset and its preprocessing steps. Section V presents the results of our experiments along with a comprehensive performance evaluation. Following this, Section VI delves into discussions and insights drawn from the experimental outcomes. Finally, in Section VII, we summarize our findings and present avenues for future research endeavors.

II. Related Work

In recent years, the intersection of artificial intelligence (AI) and medical imaging has significantly advanced diagnostic capabilities, particularly in the field of lung ultrasound (LUS). Several studies have made notable contributions to this domain, focusing on various aspects of LUS interpretation and automation.

Kolárik et al. [3] emphasized the importance of accurate diagnosis and monitoring of lung diseases, particularly in light of the COVID-19 pandemic. Their work centered on detecting the absence of lung sliding in ultrasound videos using 3D Convolutional Neural Networks (CNNs), offering a safer and more accessible alternative to ionizing radiation-based techniques.

Jaščur et al. [4] underscored the potential of AI-enabled automation in enhancing LUS interpretation, specifically in identifying lung sliding artifacts crucial for diagnosing pneumothorax. Their study highlighted the scarcity of automated approaches in this domain and proposed a DL solution with promising sensitivity and specificity for eventual clinical validation.

In the context of post-surgery complications monitoring, Kulhare et al. [5] explored DL's application in LUS interpretation, developing methods for detecting various lung abnormalities, including pneumothorax, using CNNs and single-shot detectors. Their approach demonstrated high sensitivity and specificity in identifying pneumothorax, addressing a critical aspect of post-thoracic surgery care.

Mehanian et al. [6] contributed to automated lung sliding assessment by developing multiple classification methods, including M-mode and LSTM-based approaches. Their study, focusing on the detection of lung sliding absence, demonstrated competitive sensitivity and specificity rates, leveraging a dataset comprising LUS recordings from porcine models.

Summers et al. [7] and Lindsey et al. [8] provided valuable datasets and algorithms for pneumothorax detection in LUS, utilizing M-mode and B-mode images in their analyses. Their studies showcased high sensitivity and specificity, highlighting the potential of DL-based approaches in augmenting LUS interpretation for clinical applications.

Collectively, these studies underscore the growing interest and progress in employing AI and DL techniques to automate LUS interpretation, addressing critical diagnostic challenges and enhancing patient care in respiratory medicine.

III. Methodology

In our model development journey, we explored various methods to enhance the classification accuracy. Initially, we experimented with InceptionV3 [9] and 2D convolution [10] for its simplicity. However, this approach did not yield the desired results. Subsequently, we shifted focus to 3D convolution [11] and recurrent neural networks (RNNs) [12] to leverage optical flow dynamics. While we initially tried the RAFT (Recurrent All-Pairs Field Transforms) [13] pretrained model, its resource-intensive nature and limited efficacy led us to develop our own optical flow architectures. Recognizing the challenges posed by a small dataset, we took proactive steps based on insights from literature [14], such as partitioning manually selected areas below the pleura into 32x32 images to clean the dataset and augment its volume. However, to streamline the process, ultimately, we decided to implement augmentation techniques on the full-sized dataset.

Our final model includes the following methods:

- Utilization of three-dimensional convolution on three-dimensional data, such as video sequences, with consideration given to the third dimension (time) [3].
- Residual Units, which are building blocks of deep neural networks with skip connections [15]. These units learn to compute residual functions with respect to the input of the layer, rather than learning new functions from scratch.
- Using data augmentation techniques, which help overcome class imbalance and overfitting by applying various transformations to existing data [16]. These transformations involve several stages:
  1) First, the images are resized to a specified size using bilinear interpolation.
  2) Then, with a probability of 0.5 for each, the following transformations are applied: random horizontal flipping, random changes in brightness, contrast, and saturation of the image, random rotation of the image by a specified angle, as well as random cropping and resizing of the image.
  3) Finally, the images are normalized using the calculated means and standard deviations across the training dataset.

  These transformations help increase the diversity of the data, address class imbalance, and overfitting issues, leading to more effective model training.
- We also utilized the Adam optimizer for training our models. Adam automatically adjusts the learning rate for each parameter, which helps to achieve faster and more stable convergence [17].
- Furthermore, to stabilize the learning process, improve the generalization of the model, and adapt it to different conditions, we applied a learning rate scheduler. The scheduler allowed us to dynamically restrict the learning rate, helping to avoid "overshooting" optimal values and contributing to fine-tuning the model more effectively in later stages of training [18].
- To stabilize the learning process, we employed the gradient clipping method, which prevents too large changes in the model weights caused by gradient explosion. This is especially important in our case, as our model performed worse due to large gradient differences [19].
- We applied regularization with weight decay to reduce the growth of the weights and avoid overfitting the model. Using this method, we addressed the issue of excessive optimization of the training cost, which, in turn, would lead to an increase in the model's generalization ability [20].
- Additionally, we applied batch normalization [21] and dropout [22] to further improve the generalization and robustness of our model. Batch normalization helps in reducing internal covariate shift and accelerates the training process, while dropout helps in preventing overfitting by randomly dropping out units during training, thus forcing the model to learn more robust features.
- In addressing unbalanced data with difficult-to-classify objects, such as the absence of lung sliding in our case, which represents the minority class, we employed the Focal Loss function to increase their weights. This directly impacts the quality of the model's classification in uncertain cases [23].

For the most critical metric, we prioritize recall, as it is vital for detecting the absence of lung sliding. However, we also aimed to strike a balance with the specificity metric.

IV. Dataset

Our dataset comprises 171 ultrasound videos capturing lung examinations from various patients, recorded using different equipment. These videos contain between 20 and 900 frames in grayscale format, with a color depth ranging from 0 to 255. The frames are stored in PNG format, and their resolutions vary from 240 by 360 to 530 by 450 pixels. On average, each frame occupies between 35 and 80 kilobytes of storage space.

The dataset includes two target classes: "Lung sliding absent" indicates the absence of characteristic lung movement during breathing, which may pose a health risk to the patient,

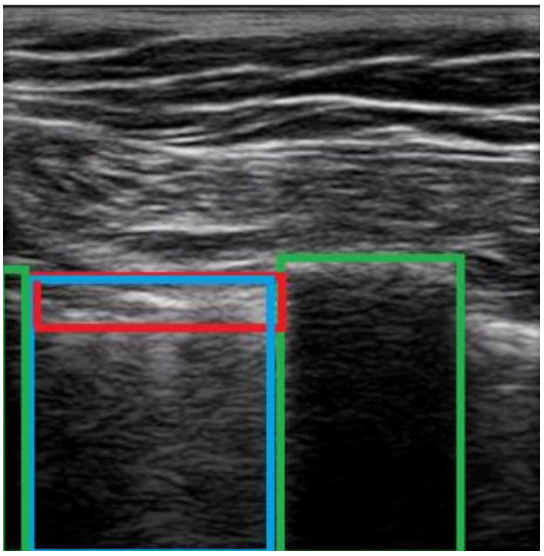


Fig. 1: Frame from USG video with marked areas of the lung

while "Lung sliding present" indicates the presence of such movement, typical for normal lung function during breathing. These classes are denoted as `ls_a` and `ls_p`, respectively. Class `ls_a` contains 32 videos, while class `ls_p` contains 139 videos. An example of a frame with marked areas of the lung is shown in Figure 1.

Before commencing model training, we conducted data preparation and preprocessing. Initially, we divided each video into several segments, cropping the frames of the lung ultrasound videos to 32 by 32 pixels while preserving the main image characteristics to focus on the essential information in each frame. This was done to obtain more data for training and get the essential features. However, after experimentation and further consideration, we decided to use the original full-size frames for classification, as this enables full automation for ease of use.

Further data preprocessing included the following steps:

- **Class Labels Transformation:** The class labels `ls_p` and `ls_a` were converted into numerical values 0 and 1, respectively. This allows us to use class labels as input data for the model.
- **Data Splitting:** The data were divided into training, validation, and test sets using a patient-wise split, with ratios of 60% for training, and 20% each for validation and testing. The validation set was exclusively utilized for experiments and fine-tuning hyperparameters, and later combined with the training set for augmentation purposes.
- **Class Balancing:** A class balancing strategy was applied to make the ratios of classes in the sets the same by dropping some examples of class 0 from the sets.

To address dataset imbalance for our recurrent convolutional model, we tried a solution of pre-dividing the dataset into two, each with its class. At each epoch, we selected from the larger dataset the same number of new images as the number of the smaller set. This balanced mini-batch training ensured unbiased learning across all classes. [24]

V. Experiments and Their Results

In this section, we will describe experimentation process. How we tested various models with different combinations of hyperparameters to choose the best one.

A. Recurrent Convolutional Model

We opted for Gated Recurrent Units (GRUs) [25] as the recurrent component for several reasons: they are computationally efficient compared to Long Short-Term Memory (LSTM) [26] networks, have fewer parameters which aids in training with limited data, offer a simpler architecture enhancing interpretability, perform well on shorter sequences, and train faster.

The final structure of our recurrent model is illustrated in Figure 2. It operates as follows: a batch of 30 frames (adjustable) is processed in parallel through the convolutional component. The output of the convolutional model is then flattened to a two-dimensional tensor, which is further reduced in size by a linear layer. Subsequently, this sequence is fed into the GRU component, where the convolutional output of 30 frames acts as a sequence. The last iteration of the GRU produces the input for the final linear layer, which converts it into logits.

An important consideration is the mandatory normalization before inputting into the GRU, as this layer expects normalized data at the input. Failure to do so will hinder training, resulting in consistent output regardless of input variation.

Following experiments, we obtained the results depicted in Figures 3, 4, 5, and 6.

Despite our efforts to mitigate overfitting, we encountered challenges in achieving stable results with the recurrent model.

B. 3D Convolutional Model With RAFT

This model utilized RAFT to extract feature maps of movement throughout the videos. Subsequently, these feature maps were processed by our custom model consisting of two 3D convolutional layers and a fully connected part responsible for classification. The structure of the model is depicted in Figure 8.

During training, we employed 32x32 videos extracted from the main dataset. To optimize the feature maps obtained from RAFT, we experimented with the `flow_step` hyperparameter, which determines the time difference between consecutive frames. The results of these experiments are illustrated in Figures 9, 10, 11, and 12. As our methodology prioritizes recall, we focused on this metric for evaluation.

Additionally, we investigated the impact of the `gamma` hyperparameter of the focal loss function on model performance. The results of this analysis, focusing on recall, are presented in Figure 13.

Despite extensive experimentation, satisfactory results were not achieved on the entire dataset (with 20% reserved for the test set) when utilizing feature maps from the RAFT pretrained

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
RCM2d_Small                              [1, 1]                    --
├─Sequential: 1-1                        [30, 4096]                --
│    └─Conv2d: 2-1                       [30, 2, 256, 256]         100
│    └─ReLU: 2-2                         [30, 2, 256, 256]         --
│    └─BatchNorm2d: 2-3                  [30, 2, 256, 256]         4
│    └─Resblock: 2-4                     [30, 2, 256, 256]         --
│    │    └─Conv2d: 3-1                  [30, 2, 256, 256]         38
│    │    └─ReLU: 3-2                    [30, 2, 256, 256]         --
│    │    └─Conv2d: 3-3                  [30, 2, 256, 256]         38
│    │    └─ReLU: 3-4                    [30, 2, 256, 256]         --
│    │    └─BatchNorm2d: 3-5             [30, 2, 256, 256]         4
│    └─MaxPool2d: 2-5                    [30, 2, 64, 64]           --
│    └─Conv2d: 2-6                       [30, 4, 64, 64]           76
│    └─ReLU: 2-7                         [30, 4, 64, 64]           --
│    └─BatchNorm2d: 2-8                  [30, 4, 64, 64]           8
│    └─Resblock: 2-9                     [30, 4, 64, 64]           --
│    │    └─Conv2d: 3-6                  [30, 4, 64, 64]           148
│    │    └─ReLU: 3-7                    [30, 4, 64, 64]           --
│    │    └─Conv2d: 3-8                  [30, 4, 64, 64]           148
│    │    └─ReLU: 3-9                    [30, 4, 64, 64]           --
│    │    └─BatchNorm2d: 3-10            [30, 4, 64, 64]           8
│    └─MaxPool2d: 2-10                   [30, 4, 32, 32]           --
│    └─Conv2d: 2-11                      [30, 8, 32, 32]           296
│    └─ReLU: 2-12                        [30, 8, 32, 32]           --
│    └─BatchNorm2d: 2-13                 [30, 8, 32, 32]           16
│    └─Resblock: 2-14                    [30, 8, 32, 32]           --
│    │    └─Conv2d: 3-11                 [30, 8, 32, 32]           584
│    │    └─ReLU: 3-12                   [30, 8, 32, 32]           --
│    │    └─Conv2d: 3-13                 [30, 8, 32, 32]           584
│    │    └─ReLU: 3-14                   [30, 8, 32, 32]           --
│    │    └─BatchNorm2d: 3-15            [30, 8, 32, 32]           16
│    └─MaxPool2d: 2-15                   [30, 8, 16, 16]           --
│    └─Conv2d: 2-16                      [30, 16, 16, 16]          1,168
│    └─ReLU: 2-17                        [30, 16, 16, 16]          --
│    └─BatchNorm2d: 2-18                 [30, 16, 16, 16]          32
│    └─Flatten: 2-19                     [30, 4096]                --
├─Sequential: 1-2                        [30, 512]                 --
│    └─Dropout: 2-20                     [30, 4096]                --
│    └─Linear: 2-21                      [30, 512]                 2,097,664
│    └─ReLU: 2-22                        [30, 512]                 --
│    └─BatchNorm1d: 2-23                 [30, 512]                 1,024
├─GRU: 1-3                               [1, 30, 1024]             4,724,736
├─Sequential: 1-4                        [1, 1]                    --
│    └─Dropout: 2-24                     [1, 1024]                 --
│    └─Linear: 2-25                      [1, 1]                    1,025
==========================================================================================
Total params: 6,827,717
```

Fig. 2: Recurrent model structure

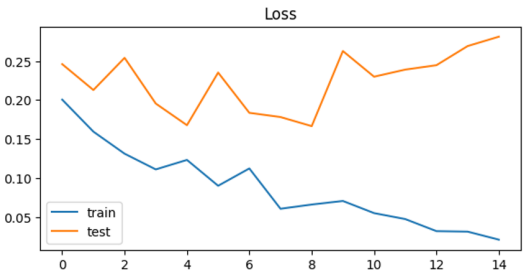


Fig. 3: Recurrent model loss

model. The resulting confusion matrix is depicted in Figure 14, and the corresponding metrics are summarized in Table I.

TABLE I: Metrics of the model with optical flow

Loss	Accuracy	Precision	Recall	F1	Specificity
0.00156	0.3387	0.1864	0.7756	0.3006	0.2407

C. 3D Convolutional Model With Residual Blocks

Our next model comprised one 3D convolutional layer, three residual layers, and fully connected layers for classification, utilized on the full resized videos. The structure of the model is depicted in Figure 15.

Initially, we investigated the impact of batch size on model performance. Larger batch sizes tended to yield better results, as suggested by prior research. Therefore, we adopted the practice of using the largest feasible batch size for our final model iterations.

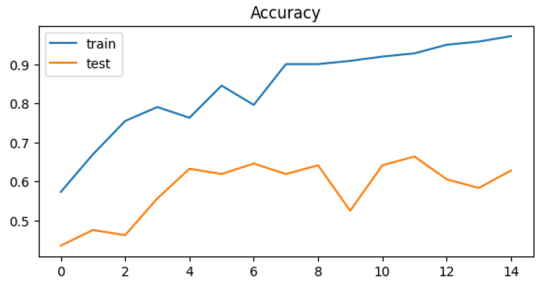


Fig. 4: Recurrent model accuracy

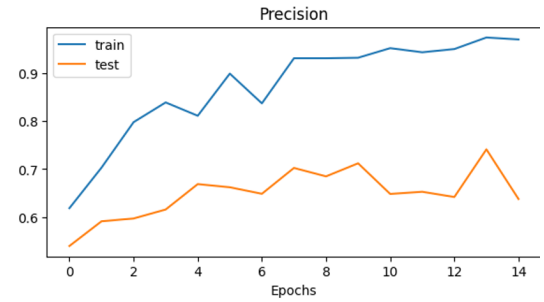


Fig. 5: Recurrent model precision

Another hyperparameter under scrutiny was dropout probability. Four experiments were conducted, each trained for 10 epochs. The comparison of results is presented in Table II, with the model achieving the best recall and specificity with a dropout probability of 60%. A graphical comparison of the recall for dropout probabilities of 0.5 and 0.6 is shown in Figure 16.

TABLE II: Comparison of Results Using Different Dropout Probabilities

Dropout Probability	0.4	0.5	0.6	0.7
Loss	0.006750	0.007058	0.008135	0.008664
Accuracy	0.8258	0.7197	0.7197	0.2197
Precision	0.5714	0.3125	0.3800	0.1953
Recall	0.3200	0.4000	0.7600	1.0000
F1	0.4103	0.3509	0.5067	0.3268
Specificity	0.9439	0.7944	0.7103	0.0374

We also experimented with gradient clipping to prevent ”exploding gradients”. Results of training with different gradient clipping settings are listed in Table III. The test with higher gradient clipping values exhibited worse results, with the best performance achieved when the gradient clipping was set to 0.4.

Additionally, we explored the effect of varying numbers of frames in training. Graphical data from tests with 60 frames indicated fast overtraining, as shown in Figure 17. Consequently, we chose 34 frames for further research, as they exhibited promising results with a potentially delayed onset of overtraining, as depicted in Figure 18.

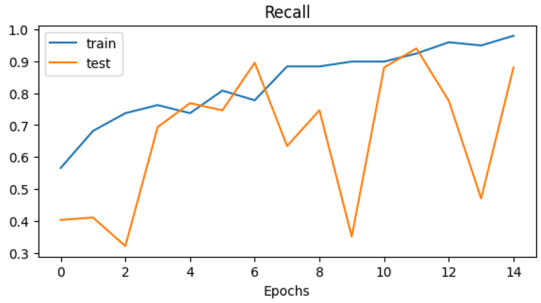


Fig. 6: Recurrent model recall

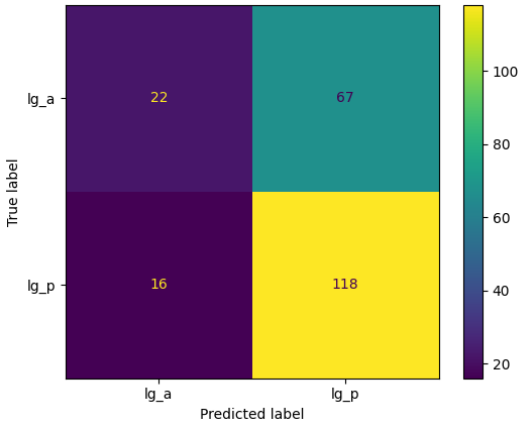
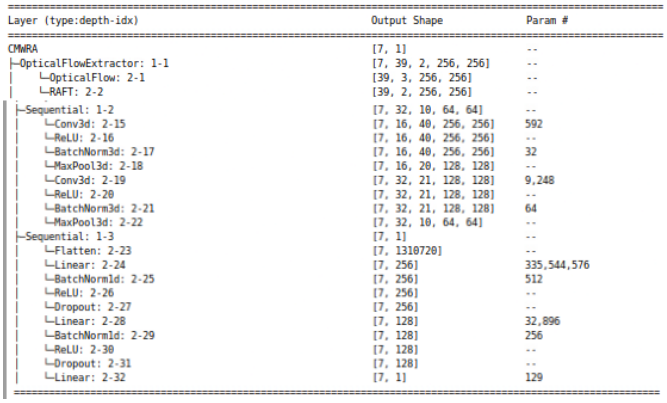


Fig. 7: Test confusion matrix



Layer (type:depth-idx)	Output Shape	Param #
CMWRA	[7, 1]	--
├─OpticalFlowExtractor: 1-1	[7, 39, 2, 256, 256]	--
│ └─OpticalFlow: 2-1	[39, 3, 256, 256]	--
│ └─RAFT: 2-2	[39, 2, 256, 256]	--
├─Sequential: 1-2	[7, 32, 10, 64, 64]	--
│ └─Conv3d: 2-15	[7, 16, 40, 256, 256]	592
│ └─ReLU: 2-16	[7, 16, 40, 256, 256]	--
│ └─BatchNorm3d: 2-17	[7, 16, 40, 256, 256]	32
│ └─MaxPool3d: 2-18	[7, 16, 20, 128, 128]	--
│ └─Conv3d: 2-19	[7, 32, 21, 128, 128]	9,248
│ └─ReLU: 2-20	[7, 32, 21, 128, 128]	--
│ └─BatchNorm3d: 2-21	[7, 32, 21, 128, 128]	64
│ └─MaxPool3d: 2-22	[7, 32, 10, 64, 64]	--
├─Sequential: 1-3	[7, 1]	--
│ └─Flatten: 2-23	[7, 1310720]	--
│ └─Linear: 2-24	[7, 256]	335,544,576
│ └─BatchNorm1d: 2-25	[7, 256]	512
│ └─ReLU: 2-26	[7, 256]	--
│ └─Dropout: 2-27	[7, 256]	--
│ └─Linear: 2-28	[7, 128]	32,896
│ └─BatchNorm1d: 2-29	[7, 128]	256
│ └─ReLU: 2-30	[7, 128]	--
│ └─Dropout: 2-31	[7, 128]	--
│ └─Linear: 2-32	[7, 1]	129

Fig. 8: Structure of the Convolutional Model With RAFT (CMWRA)

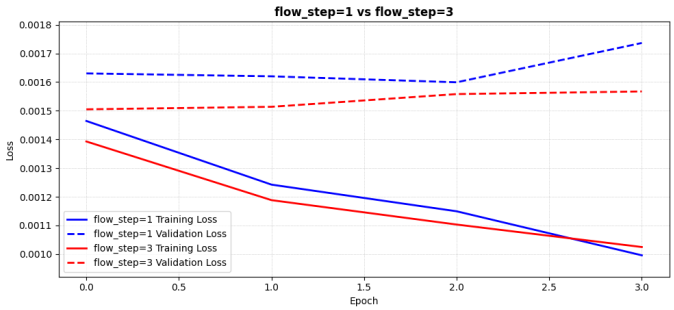


Fig. 9: Loss: flow step 1 and flow step 3

D. Final results

On the final test, we used a bigger train dataset, so final results may vary from experimental results. A final test of our model was conducted using hyperparameters listed in Table IV.

On the Figures 19, 20, 21 you can see history of training loss, accuracy and recall of the model respectively. The confusion matrix is shown on the Figure 22. Final testing loss and metrics are listed in the Table V.

Our models demonstrate a high specificity of classification (80%) and high recall metric(80%), which is the most important metric in our case.

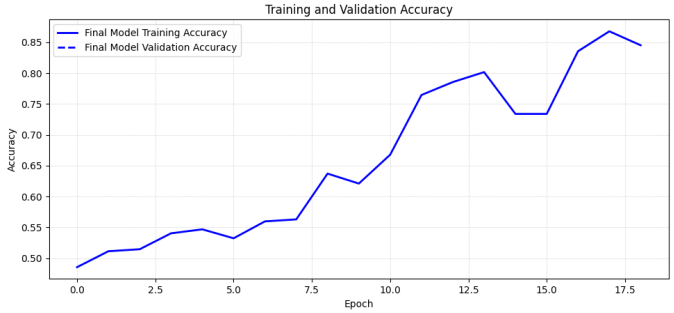


Fig. 20: History of training accuracy of the model

TABLE III: Comparison of Results Using Different Gradient Clipping Values

Gradient Clipping	0.1	0.4	0.5	0.8
Loss	0.008136	0.007489	0.008513	0.007721
Accuracy	0.7197	0.7576	0.7121	0.7197
Precision	0.3800	0.4386	0.3617	0.3636
Recall	0.7600	1.0000	0.6800	0.6400
F1	0.5067	0.6098	0.4722	0.4638
Specificity	0.7103	0.7009	0.7196	0.7383

TABLE IV: Final hyperparameters

Number of frames	45
Learning rate	0.001
Weight decay	0.001
Shed epochs	16
Gradient clipping	0.4
Batch size	20
Dropout probability	0.65
Gamma	1
Alpha	0.8

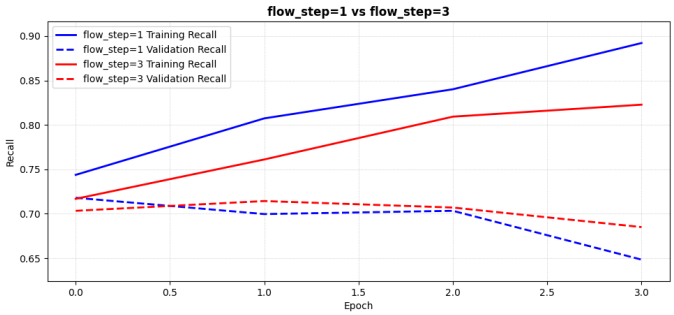


Fig. 10: Recall: flow step 1 and flow step 3

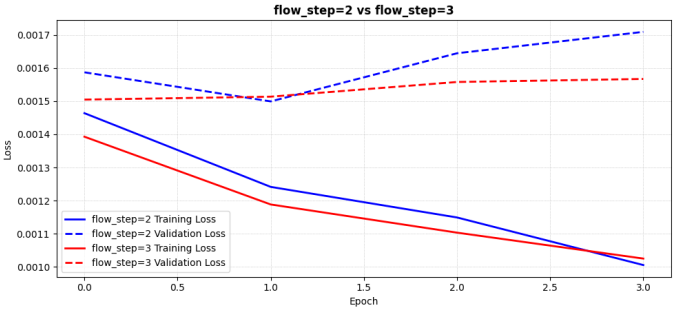


Fig. 11: Loss: flow step 2 and flow step 3

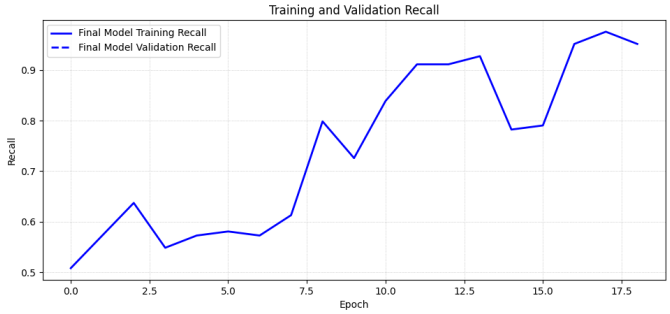


Fig. 21: History of training recall of the model

TABLE V: Final loss and metrics

Loss	0.00454302
Accuracy	0.6700
Precision	0.3556
Recall	0.8000
F1	0.4923
Specificity	0.6375

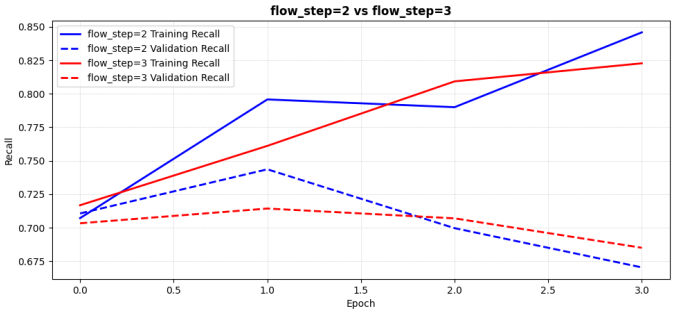


Fig. 12: Recall: flow step 2 and flow step 3

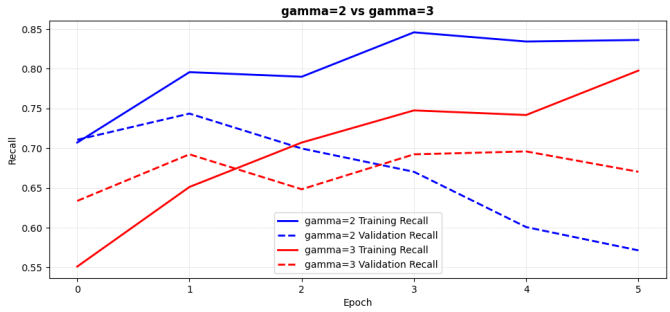


Fig. 13: Recall: gamma 2 and gamma 3

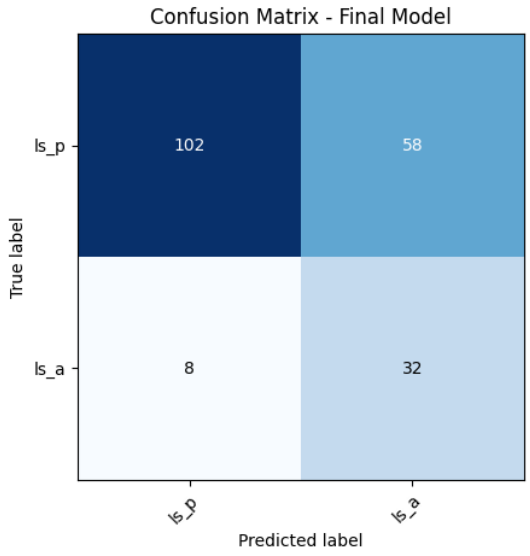


Fig. 22: Confusion matrix of the model at the final test

VI. Discussion

In our exploration of various deep learning models for pneumothorax diagnosis, we encountered challenges that influenced our journey toward creating effective classification systems for identifying lung sliding patterns.

Resource Intensity: While leveraging pretrained models like RAFT for optical flow analysis appeared promising, we were mindful of their resource-intensive nature. Balancing computational demands with model efficacy was crucial, especially given our limited resources.

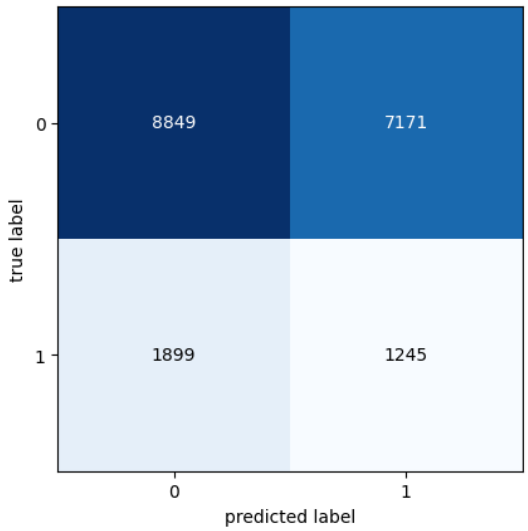


Fig. 14: Confusion matrix of the model with optical flow

Dataset Size: The relatively small size of our dataset presented a challenge. Recognizing that deep learning models benefit from large datasets, we explored strategies to effectively augment our dataset to improve model performance and generalization.

Data Distribution: Addressing class imbalance was a priority. Balanced representation across classes was essential to ensure our models learned to classify both lung sliding present and absent cases accurately.

Overfitting: Despite employing techniques like data augmentation and regularization, overfitting remained a concern, particularly with our recurrent convolutional model.

Fine-Tuning Hyperparameters: Hyperparameter tuning demanded meticulous attention. We experimented with different configurations to optimize parameters such as dropout probability, gradient clipping, and input sequence length to maximize model performance.

We propose several methods that could improve our results or advance the field in the future:

Expansion of Training Data: Expanding our dataset could significantly benefit model performance. Acquiring more diverse ultrasound videos capturing various patient conditions and recording setups would expose the model to a broader range of scenarios, enhancing its ability to generalize and make accurate predictions across different patient demographics and imaging conditions.

Tuning Hyperparameters for Precision: Continual fine-tuning of hyperparameters could yield more precise results. Systematically exploring different configurations and optimizing parameters such as learning rate, weight decay, and dropout probability could unlock additional performance gains, ensuring the model operates at its optimal capacity in terms of accuracy and efficiency.

Investigating Region-Specific Identification Techniques: Exploring techniques for identifying specific regions of interest within ultrasound videos holds potential for refining our model's predictions. Extracting insights from targeted regions could enhance interpretability, improve diagnostic accuracy, and provide clinicians with valuable context for more informed decision-making and better patient outcomes.

Layer (type:depth-idx)	Output Shape	Param #
CMWRU	[7, 1]	--
├─Conv3d: 1-1	[7, 8, 40, 256, 256]	600
├─BatchNorm3d: 1-2	[7, 8, 40, 256, 256]	16
├─ReLU: 1-3	[7, 8, 40, 256, 256]	--
├─MaxPool3d: 1-4	[7, 8, 40, 128, 128]	--
├─Sequential: 1-5	[7, 16, 40, 128, 128]	--
│ └─BasicBlock3D: 2-1	[7, 16, 40, 128, 128]	--
│ │ └─Conv3d: 3-1	[7, 16, 40, 128, 128]	3,456
│ │ └─BatchNorm3d: 3-2	[7, 16, 40, 128, 128]	32
│ │ └─ReLU: 3-3	[7, 16, 40, 128, 128]	--
│ │ └─Conv3d: 3-4	[7, 16, 40, 128, 128]	6,912
│ │ └─BatchNorm3d: 3-5	[7, 16, 40, 128, 128]	32
│ │ └─Dropout3d: 3-6	[7, 16, 40, 128, 128]	--
│ │ └─Sequential: 3-7	[7, 16, 40, 128, 128]	160
│ │ └─ReLU: 3-8	[7, 16, 40, 128, 128]	--
│ └─BasicBlock3D: 2-2	[7, 16, 40, 128, 128]	--
│ │ └─Conv3d: 3-9	[7, 16, 40, 128, 128]	6,912
│ │ └─BatchNorm3d: 3-10	[7, 16, 40, 128, 128]	32
│ │ └─ReLU: 3-11	[7, 16, 40, 128, 128]	--
│ │ └─Conv3d: 3-12	[7, 16, 40, 128, 128]	6,912
│ │ └─BatchNorm3d: 3-13	[7, 16, 40, 128, 128]	32
│ │ └─Dropout3d: 3-14	[7, 16, 40, 128, 128]	--
│ │ └─ReLU: 3-15	[7, 16, 40, 128, 128]	--
├─Sequential: 1-6	[7, 32, 20, 64, 64]	--
│ └─BasicBlock3D: 2-3	[7, 32, 20, 64, 64]	--
│ │ └─Conv3d: 3-16	[7, 32, 20, 64, 64]	13,824
│ │ └─BatchNorm3d: 3-17	[7, 32, 20, 64, 64]	64
│ │ └─ReLU: 3-18	[7, 32, 20, 64, 64]	--
│ │ └─Conv3d: 3-19	[7, 32, 20, 64, 64]	27,648
│ │ └─BatchNorm3d: 3-20	[7, 32, 20, 64, 64]	64
│ │ └─Dropout3d: 3-21	[7, 32, 20, 64, 64]	--
│ │ └─Sequential: 3-22	[7, 32, 20, 64, 64]	576
│ │ └─ReLU: 3-23	[7, 32, 20, 64, 64]	--
│ └─BasicBlock3D: 2-4	[7, 32, 20, 64, 64]	--
│ │ └─Conv3d: 3-24	[7, 32, 20, 64, 64]	27,648
│ │ └─BatchNorm3d: 3-25	[7, 32, 20, 64, 64]	64
│ │ └─ReLU: 3-26	[7, 32, 20, 64, 64]	--
│ │ └─Conv3d: 3-27	[7, 32, 20, 64, 64]	27,648
│ │ └─BatchNorm3d: 3-28	[7, 32, 20, 64, 64]	64
│ │ └─Dropout3d: 3-29	[7, 32, 20, 64, 64]	--
│ │ └─ReLU: 3-30	[7, 32, 20, 64, 64]	--
├─Sequential: 1-7	[7, 64, 10, 32, 32]	--
│ └─BasicBlock3D: 2-5	[7, 64, 10, 32, 32]	--
│ │ └─Conv3d: 3-31	[7, 64, 10, 32, 32]	55,296
│ │ └─BatchNorm3d: 3-32	[7, 64, 10, 32, 32]	128
│ │ └─ReLU: 3-33	[7, 64, 10, 32, 32]	--
│ │ └─Conv3d: 3-34	[7, 64, 10, 32, 32]	110,592
│ │ └─BatchNorm3d: 3-35	[7, 64, 10, 32, 32]	128
│ │ └─Dropout3d: 3-36	[7, 64, 10, 32, 32]	--
│ │ └─Sequential: 3-37	[7, 64, 10, 32, 32]	2,176
│ │ └─ReLU: 3-38	[7, 64, 10, 32, 32]	--
│ └─BasicBlock3D: 2-6	[7, 64, 10, 32, 32]	--
│ │ └─Conv3d: 3-39	[7, 64, 10, 32, 32]	110,592
│ │ └─BatchNorm3d: 3-40	[7, 64, 10, 32, 32]	128
│ │ └─ReLU: 3-41	[7, 64, 10, 32, 32]	--
│ │ └─Conv3d: 3-42	[7, 64, 10, 32, 32]	110,592
│ │ └─BatchNorm3d: 3-43	[7, 64, 10, 32, 32]	128
│ │ └─Dropout3d: 3-44	[7, 64, 10, 32, 32]	--
│ │ └─ReLU: 3-45	[7, 64, 10, 32, 32]	--
├─Sequential: 1-8	[7, 1]	--
│ └─Linear: 2-7	[7, 64]	41,943,104
│ └─BatchNorm1d: 2-8	[7, 64]	128
│ └─ReLU: 2-9	[7, 64]	--
│ └─Dropout: 2-10	[7, 64]	--
│ └─Linear: 2-11	[7, 32]	2,080
│ └─BatchNorm1d: 2-12	[7, 32]	64
│ └─ReLU: 2-13	[7, 32]	--
│ └─Dropout: 2-14	[7, 32]	--

Fig. 15: Structure of the 3D Convolutional Model With Residual Blocks (CMWRU)

VII. Conclusion

Our research represents a step forward in automated lung sliding detection on USG using deep learning.

Our experiments with different model variations, including recurrent convolutional models, 3D convolutional models with RAFT, and residual blocks, provided valuable insights into

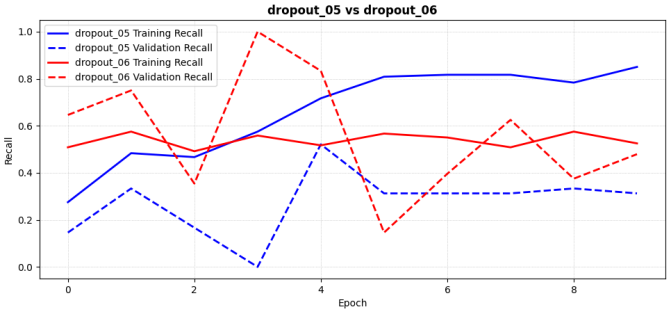


Fig. 16: Graphical Comparison of Recall for Different Dropout Probabilities

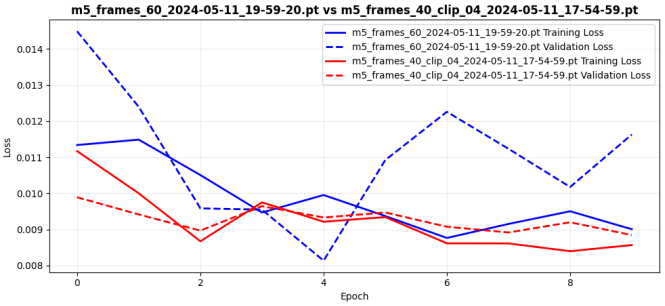


Fig. 17: Graphical comparison of loss of model with different number of frames

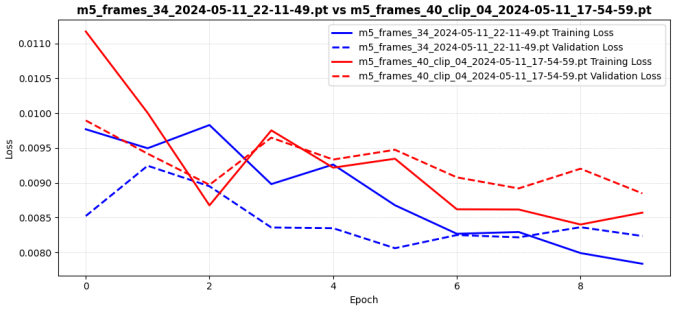


Fig. 18: Graphical comparison of recall of model with different number of frames

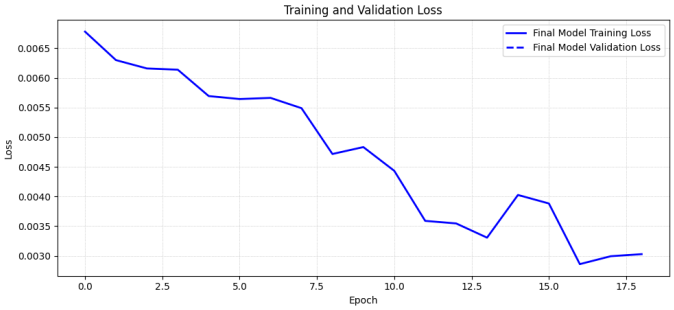


Fig. 19: History of training loss of the model

the strengths and limitations of each approach. While we encountered hurdles such as overfitting in recurrent models and suboptimal performance with RAFT, each iteration informed our understanding and guided our refinement process.

We meticulously addressed challenges such as dataset imbalance, overfitting, and model stability. Techniques like data augmentation, class balancing, regularization, and optimization strategies like Adam optimizer and learning rate scheduling played pivotal roles in enhancing the robustness and efficiency of our models.

The research gave us a lot of experience in using different technologies when working with CNN, with unbalanced dataset, video data, and medical nuances.

We believe that our effort could lead to even more accurate and reliable automatic detection systems for pneumothorax and other chest diseases.

ACKNOWLEDGMENT

This research was supported by the Technical University of Košice.

REFERENCES

[1] J. P. Kanne and M. D. M. Rother, "Pneumothorax: Imaging diagnosis and etiology," *Seminars in Roentgenology*, vol. 58, no. 4, pp. 440–453, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0037198X23000494

[2] M. Strnad, S. Zadel, G. PROSEN, and Z. KLEMENC-KETIS, "Identification of lung sliding: a basic ultrasound technique with a steep learning curve," *Signa vitae: journal for intesive care and emergency medicine*, vol. 8, no. 1, pp. 31–35, 2013.

[3] M. Kolarik, M. Sarnovsky, and J. Paralic, "Detecting the absence of lung sliding in ultrasound videos using 3d convolutional neural networks," *Acta Polytechnica Hungarica*, vol. 20, pp. 47–60, 01 2023.

[4] M. Jaščur, M. Bundzel, M. Malík, A. Dzian, N. Ferenčík, and F. Babič, "Detecting the absence of lung sliding in lung ultrasounds using deep learning," *Applied Sciences*, vol. 11, no. 15, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/15/6976

[5] S. Kulhare, X. Zheng, C. Mehanian, C. Gregory, M. Zhu, K. Gregory, H. Xie, J. McAndrew Jones, and B. Wilson, "Ultrasound-based detection of lung abnormalities using single shot detection convolutional neural networks," in *Simulation, Image Processing, and Ultrasound Systems for Assisted Diagnosis and Navigation - International Workshops, POCUS 2018, BIVPCS 2018, CuRIOUS 2018, and CPM 2018, Held in Conjunction with MICCAI 2018, Proceedings*, ser. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), S. Aylward, A. Simpson, L. Maier-Hein, A. Martel, M. Chabanas, J. Tavares, I. Reinertsen, Z. Taylor, Y. Xiao, K. Farahani, D. Stoyanov, S. Li, and H. Rivaz, Eds. Springer-Verlag, 2018, pp. 65–73, publisher Copyright: © Springer Nature Switzerland AG 2018.; International Workshop on Point-of-Care Ultrasound, POCUS 2018, the International Workshop on Bio-Imaging and Visualization for Patient-Customized Simulations, BIVPCS 2017, the International Workshop on Correction of Brainshift with Intra-Operative Ultrasound, CuRIOUS 2018, and the International Workshop on Computational Precision Medicine, CPM 2018, held in conjunction with the 21st International Conference on Medical Imaging and Computer-Assisted Intervention, MICCAI 2018 ; Conference date: 16-09-2018 Through 20-09-2018.

[6] C. Mehanian, S. Kulhare, R. Millin, X. Zheng, C. Gregory, M. Zhu, H. Xie, J. Jones, J. Lazar, A. Halse, T. Graham, M. Stone, K. Gregory, and B. Wilson, "Deep learning-based pneumothorax detection in ultrasound videos," in *Smart Ultrasound Imaging and Perinatal, Preterm and Paediatric Image Analysis - 1st International Workshop, SUSI 2019, and 4th International Workshop, PIPPI 2019, Held in Conjunction with MICCAI 2019, Proceedings*, ser. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Q. Wang, A. Gomez, J. Hutter, A. Gomez, V. Zimmer, J. Hutter, E. Robinson, D. Christiaens, A. Melbourne,

K. McLeod, O. Zettinig, R. Licandro, and E. Turk, Eds. Springer, 2019, pp. 74–82, publisher Copyright: © 2019, Springer Nature Switzerland AG.; 1st International Workshop on Smart Ultrasound Imaging, SUSI 2019, and the 4th International Workshop on Preterm, Perinatal and Paediatric Image Analysis, PIPPI 2019, held in conjunction with the 22nd International Conference on Medical Imaging and Computer-Assisted Intervention, MICCAI 2019 ; Conference date: 17-10-2019 Through 17-10-2019.

[7] C. Summers, R. S. Todd, G. A. Vercruysse, and F. A. Moore, "Acute respiratory failure," *Perioperative Medicine*, pp. 576–586, 2022, epub 2021 Mar 5.

[8] T. Lindsey, R. Lee, R. Grisell, S. Vega, and S. Veazey, "Automated pneumothorax diagnosis using deep neural networks," in *Iberoamerican congress on pattern recognition*. Springer, 2018, pp. 723–731.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[10] N. Aloysius and M. Geetha, "A review on deep convolutional neural networks," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017, pp. 0588–0592.

[11] R. D. Singh, A. Mittal, and R. K. Bhatia, "3d convolutional neural network for object recognition: a review," *Multimedia Tools and Applications*, vol. 78, no. 12, pp. 15 951–15 995, Jun 2019. [Online]. Available: https://doi.org/10.1007/s11042-018-6912-6

[12] L. R. Medsker, L. Jain *et al.*, "Recurrent neural networks," *Design and Applications*, vol. 5, no. 64-67, p. 2, 2001.

[13] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow (extended abstract)," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Z.-H. Zhou, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2021, pp. 4839–4843, sister Conferences Best Papers. [Online]. Available: https://doi.org/10.24963/ijcai.2021/662

[14] M. Kolárik, M. Sarnovský, and J. Parali, "Detecting the absence of lung sliding in ultrasound videos using 3d convolutional neural networks," *Acta Polytechnica Hungarica*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258575015

[15] L. Qu, C. Wu, and L. Zou, "3d dense separated convolution module for volumetric medical image analysis," *Applied Sciences*, vol. 10, p. 485, 01 2020.

[16] C. Shorten and T. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, 07 2019.

[17] I. W. Mustika, H. N. Adi, and F. Najib, "Comparison of keras optimizers for earthquake signal classification based on deep neural networks," in *2021 4th International Conference on Information and Communications Technology (ICOIACT)*, 2021, pp. 304–308.

[18] J. Konar, P. Khandelwal, and R. Tripathi, "Comparison of various learning rate scheduling techniques on convolutional neural network," in *2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS)*, 2020, pp. 1–5.

[19] C. Ma, X. Kong, and B. Huang, "Image classification based on layered gradient clipping under differential privacy," *IEEE Access*, vol. 11, pp. 20 150–20 158, 2023.

[20] A. Krogh and J. Hertz, "A simple weight decay can improve generalization," *Advances in neural information processing systems*, vol. 4, 1991.

[21] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," *Advances in neural information processing systems*, vol. 31, 2018.

[22] P. Baldi and P. J. Sadowski, "Understanding dropout," *Advances in neural information processing systems*, vol. 26, 2013.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.

[24] R. Shimizu, K. Asako, H. Ojima, S. Morinaga, M. Hamada, and T. Kuroda, "Balanced mini-batch training for imbalanced image data classification with neural network," in *2018 First International Conference on Artificial Intelligence for Industries (AI4I)*. IEEE, 2018, pp. 27–30.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[26] A. Graves and A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.