MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

FPT UNIVERSITY

Capstone Project Document

Block-Wise Attention-Driven Soft Segmentation for Imbalanced Multi-Label Chest X-Ray Classification

<Group 4>	
Group Members	<Nguyen Huu Duy> <RollNo> <SE183995> <Hoang Khuong Duy> <RollNo> <SE184883> <Tran Khanh Nguyen> <RollNo> <SE183486> <Nguyen Khac Vuong> <RollNo> <SE183769>
Supervisor	Dr. Huynh Cong Viet Ngu
Ext Supervisor	
Capstone Project code	DSP391m

- HoChiMinh, July/2025 -

Table of Contents

Acknowledgement

We would like to thank our supervisor, Mr. Huynh Cong Viet Ngu, for his continuous guidance and feedback throughout this project. We also express our gratitude to FPT University for the opportunity and resources provided. Special thanks to our friends and family for their encouragement and support.

Definition and Acronyms

Acronym	Definition
AI	Artificial Intelligent
DL	Deep Learning
CNN	Convolutional Neural Network
AWB	Attention-Wise Block
LKA	Large Kernel Attention
AUC	Area Under the Curve
BCE	Binary Cross Entropy
CXR	Chest X-Ray

List of tables

List of figures

I. Project Introduction

1. Overview

1.1 Project Information

- **Project Title**: Block-Wise Attention-Driven Soft Segmentation for Imbalanced Multi-Label Chest X-Ray Classification
- **Team**: Group 4
- **Supervisor**: Dr. Huynh Cong Viet Ngu

1.2 Project Overview

This project investigates a novel attention-enhanced CNN architecture for the task of multi-label classification on chest X-ray images. We integrate Attention-Wise Blocks (AWBs) based on Large Kernel Attention (LKA) into DenseNet121 and VGG16, and adopt a two-stage training strategy to address class imbalance.

2. Project Background

Chest X-rays are commonly used in clinical practice but remain challenging to interpret due to multiple co-occurring pathologies and class imbalance in datasets like ChestXray14. Traditional CNNs are limited in capturing long-range spatial dependencies. This motivates the need for more context-aware mechanisms like attention modules.

3. Project Objective

To improve diagnostic accuracy for multi-label thoracic diseases using lightweight attention-enhanced CNNs and a training strategy designed for highly imbalanced medical datasets.

4. Problem Statement

Class imbalance and lack of pixel-level annotations hinder performance in thoracic disease classification. Existing models either require heavy computational resources (e.g., Transformers) or insufficiently address attention granularity and label imbalance.

5. Significance of the Project

Our method achieves state-of-the-art results on the ChestXray14 dataset, especially on rare diseases like Hernia and Nodule. It demonstrates how strategic attention placement and training design can significantly improve diagnostic accuracy in medical imaging.

6. Project Scope & Limitations

- **Scope**: Focus on multi-label thoracic disease classification using ChestXray14; implement and evaluate AWBs in CNN backbones.
- **Limitations**: Evaluation limited to one dataset; pixel-level annotations are not used; clinical deployment not covered.

II. Project Management Plan

1. Team Work

1.1 Team Structure and Roles

- Nguyen Huu Duy – Model development, Evaluation
- Hoang Khuong Duy – Experimental design, Evaluation
- Tran Khanh Nguyen – Model development, Data preprocessing
- Nguyen Khac Vuong – Experimental design, Demo system
- Dr. Huynh Cong Viet Ngu – Supervisor, technical advisor

1.2 Communication Plan

Weekly in-person meetings and asynchronous updates via email and Messenger. Github is used for document sharing and source control.

III. Existing Systems

1. Overview of the Field

Automated chest X-ray (CXR) analysis has become a critical area in medical image processing due to the high demand for scalable diagnostic tools. Chest X-rays are one of the most commonly performed radiological procedures, but their interpretation remains time-consuming and error-prone, requiring specialized radiological expertise. Deep learning, particularly Convolutional Neural Networks (CNNs), has emerged as a powerful tool for automating disease classification from CXR images. Recent advancements also explore the use of attention mechanisms and Transformer-based architectures to further improve model focus, interpretability, and diagnostic accuracy.

2. Historical Context

- Initial efforts in CXR analysis relied heavily on handcrafted features and traditional machine learning classifiers, such as Support Vector Machines and Random Forests. These approaches required significant domain knowledge and often struggled with generalization across datasets.
- The breakthrough came with the advent of CNNs, particularly after the release of large annotated datasets like **ChestXray14** by Wang et al. (2017), which enabled end-to-end training of deep neural networks. **CheXNet** (Rajpurkar et al., 2017) was one of the first high-profile models to reach radiologist-level performance on pneumonia detection, using a DenseNet121 architecture.

3. Key Studies and Theories

Several landmark studies have shaped the current landscape of chest X-ray classification:

- **CheXNet (Rajpurkar et al., 2017)**: DenseNet121-based model that achieved high performance on pneumonia classification using binary cross-entropy loss. However, it struggled with class imbalance and multi-label generalization.

- **CBAM (Woo et al., 2018)**: Introduced the Convolutional Block Attention Module, which enhances CNNs by applying sequential channel and spatial attention. While CBAM improved model focus on relevant areas, it was not specifically optimized for medical imaging challenges such as label imbalance.

- **SynthEnsemble (Ashraf et al., 2023)**: Proposed an ensemble approach combining CNNs and Vision Transformers, showing strong performance but requiring significant computational resources.

- **SwinCheX and MedViT**: Transformer-based architectures that incorporate self-attention for multi-label classification. These models leverage long-range dependencies but have high memory and data requirements, limiting their practicality in low-resource settings.

- **LKA (Guo et al., 2023)**: Large Kernel Attention introduces efficient spatial reasoning into CNNs by using depthwise convolutions with large receptive fields. LKA enables Transformer-like expressiveness with CNN-level efficiency, making it suitable for medical imaging tasks.

4. Technological Advancements

Recent advances in deep learning for medical imaging have emphasized the integration of attention mechanisms and domain-specific architectures:

- **Hybrid Attention-CNNs**: Combining CNN backbones with lightweight attention modules has shown promising results in medical image analysis by improving feature localization and model interpretability.

- **Vision Transformers (ViT)**: Although powerful in natural image classification, ViTs generally underperform on smaller datasets like ChestXray14 without heavy pretraining or data augmentation.

- **Loss Functions for Imbalance**: Traditional models used binary cross-entropy, which is biased toward majority classes. The introduction of **Focal Loss** helps address this by down-weighting well-classified examples and focusing on hard samples—particularly beneficial for rare disease detection.

5. Comparison of Existing Systems

Model/System	Backbone	Method Architecture	Imbalance Handling	Performance (AUC)
CheXNet	DenseNet121	Finetune	BCE Loss	0.841

AG-CNN	Resnet50 DenseNet121	3-Branch Attention based CNN	BCE Loss	0.868 0.871
SynthEnsemble	CNN + ViT	Ensemble Method	BCE Loss With 2 stage training	0.854
LKA + CNN Backbone	DenseNet121 VGG16	Channel + Spatial (LKA)	BCE Loss + Focal Loss With 2 stage training	**0.8498** **0.8818**

Table 1: Comparison of Existing Systems

6. Gaps in the Literature/Technology

Despite these advances, several limitations persist in the existing body of work:

- **Limited Sensitivity to Rare Diseases**: Many models, including CheXNet and CBAM-enhanced CNNs, still perform poorly on underrepresented classes such as Hernia and Nodule due to severe class imbalance.
- **Lack of Integrated Attention + Curriculum Learning**: While attention mechanisms (e.g., CBAM, LKA) help localize features, few models combine them with training curricula (like two-stage pipelines) tailored to medical data imbalance.
- **Transformer Dependence**: Some state-of-the-art systems rely heavily on Vision Transformers, which require large datasets and may not generalize well to medical imaging tasks where annotated data is scarce.

7. Justification for the Project

This project addresses the above gaps by proposing a **lightweight yet powerful CNN architecture enhanced with strategically placed Attention-Wise Blocks (AWBs)** based on the LKA mechanism. Furthermore, we introduce a **two-stage training strategy** that first focuses on abnormal cases and then refines the model with Focal Loss to better detect rare pathologies.

Compared to prior works, our approach offers:

- Improved interpretability via spatial and channel attention
- Enhanced performance on minority classes

- Lower computational cost
- Stronger generalization on a widely-used public benchmark (ChestXray14)

This makes the project both novel and practically valuable for real-world medical AI deployment.

IV. Methodology

1. Research Questions and Objectives

This project is centered around the following research questions:

- **RQ1**: Can lightweight attention modules integrated into CNN backbones improve multi-label thoracic disease classification?
- **RQ2**: How does strategic placement of attention modules affect the model's ability to detect rare and co-occurring pathologies?
- **RQ3**: Can a two-stage training process using Focal Loss improve classification performance in class-imbalanced datasets like ChestXray14?

To answer these, our objectives are:

- To implement and integrate Attention-Wise Blocks (AWBs) using Large Kernel Attention (LKA) into the DenseNet121 and VGG16 backbone.
- To design a training pipeline that separates feature learning and class rebalancing using Binary Cross-Entropy and Focal Loss in sequence.
- To evaluate the model on ChestXray14 using appropriate metrics and benchmarks.

2. Data Collection and Preprocessing

2.1 Data Source

We use the **ChestXray14** dataset, a public benchmark from the NIH Clinical Center consisting of:

- 112,120 frontal-view chest X-ray images
- Image-level annotations for 14 thoracic diseases
- Labels are extracted from radiology reports using NLP algorithms

2.2 Preprocessing Steps

Most existing approaches for ChestXray14 multi-label classification share a common preprocessing pipeline:

- **Image resizing**: All images resized to 224×224 pixels.
- **Normalization**: Normalized using ImageNet statistics.
- **Data Augmentation**: Applied during training to improve generalization:
    - Random rotation (±10°)
    - Horizontal flipping
    - Random cropping and zooming
- **Patient-wise splitting**: Ensured no patient appears in both training and test sets to avoid data leakage.

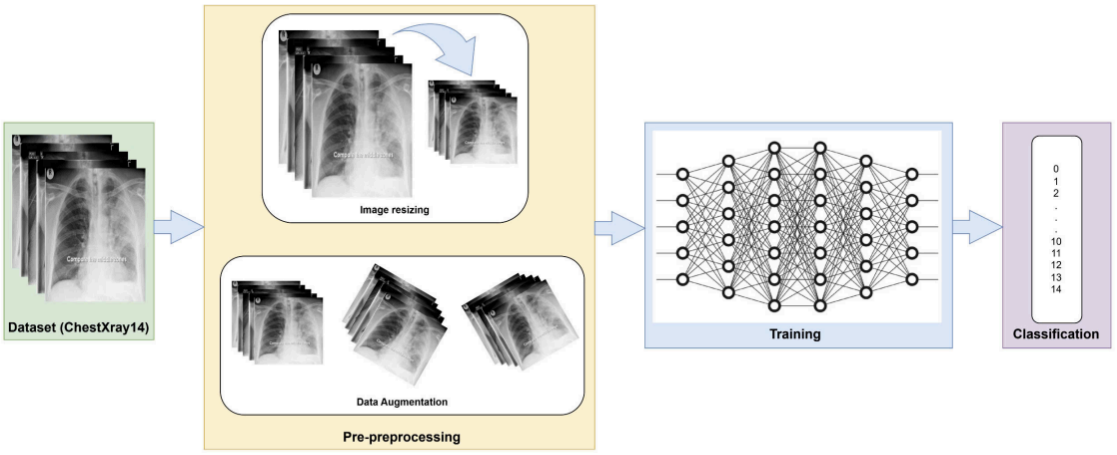


Fig 1: Workflow for ChestXray14 Classification

3. Feature Selection and Engineering

Our model performs **end-to-end feature extraction** via CNN layers enhanced with AWBs.

3.1 Attention-Wise Blocks (AWBs)

- Constructed by embedding Large Kernel Attention (LKA) modules into selected convolutional layers (blocks 3, and 4) of DenseNet121 backbone and convolutional layers (blocks 3, 4, and 5) of VGG16 backbone.
- AWBs apply both **channel and spatial attention** using depthwise and dilated convolutions to create soft attention masks.

- These masks emphasize disease-relevant features and suppress background noise in feature maps.

This hierarchical soft segmentation acts as implicit feature selection, allowing the network to focus on regions of diagnostic importance.

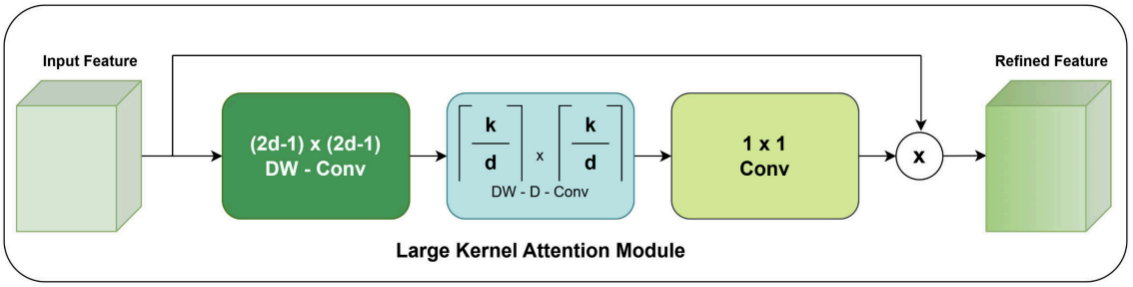


Fig 2: The Overview of LKA

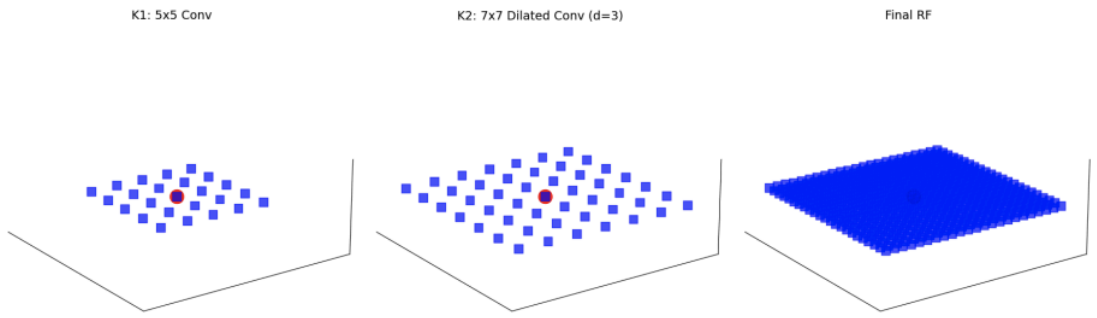


Fig 3: LKA Receptive Field Illustration

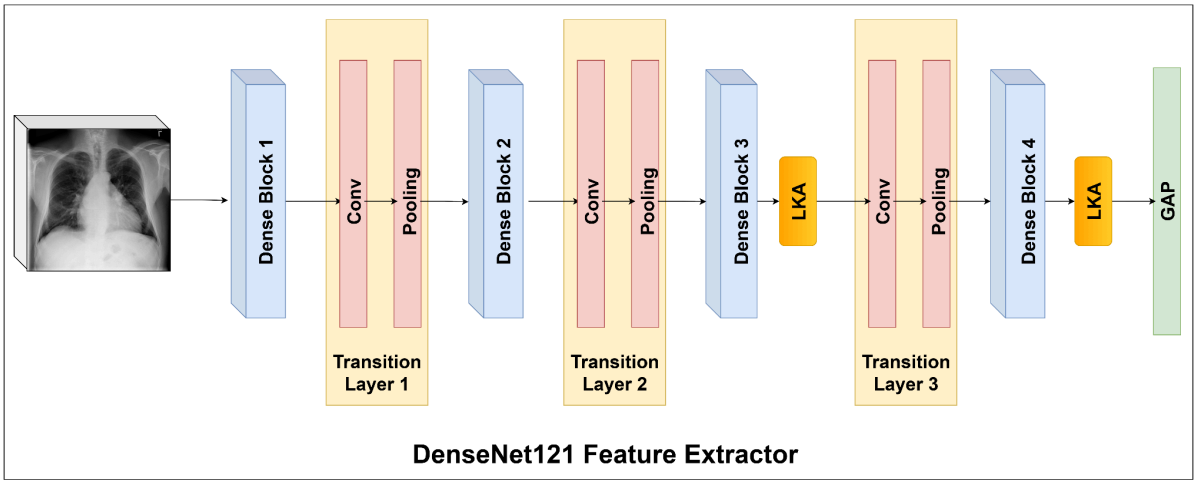


Fig 4: Architecture with AWBs in DenseNet121

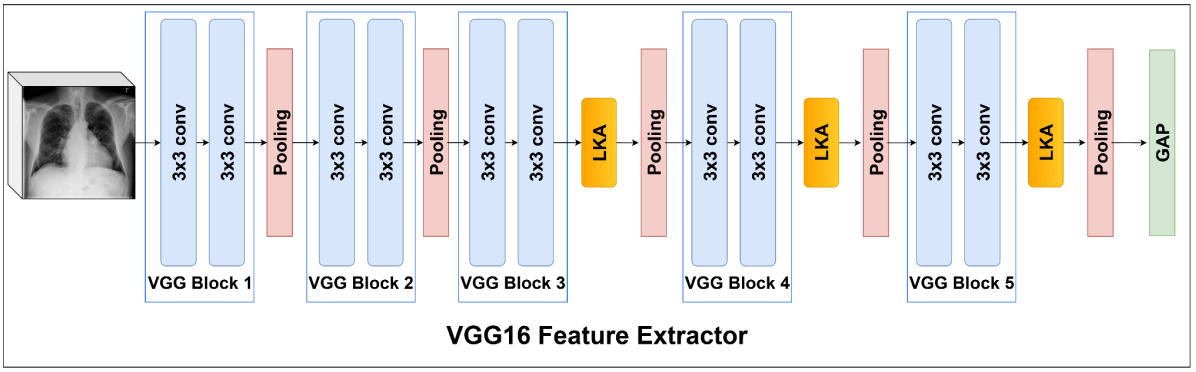


Fig 5: Architecture with AWBs in VGG16

4. Model Training and Validation

4.1 Model Architecture

- **Base Network**: DenseNet121[18], VGG16 [19] with batch normalization
- **Modifications**:
    - Inserted AWBs into blocks 3, and 4 for DenseNet121, blocks 3, 4, and 5 for VGG16
    - Replaced the final classification layer with a sigmoid-activated 14-unit dense layer

4.2 Training Pipeline

We used a **two-stage training process**:

- **Stage 1: Abnormal-Only Training**
    - Training data filtered to exclude “No Finding” images.
    - Binary Cross-Entropy (BCE) loss used to learn features purely from pathological samples.
    - Goal: help the network learn disease-relevant patterns without class imbalance interference.
- **Stage 2: Full Dataset Fine-Tuning**
    - Reintroduced full dataset (including normal cases).
    - Loss switched to **Focal Loss** to mitigate class imbalance by down-weighting easy examples and emphasizing hard-to-classify samples.

4.3 Validation Strategy

- **Split**: 70% training, 10% validation, 20% test (patient-wise)

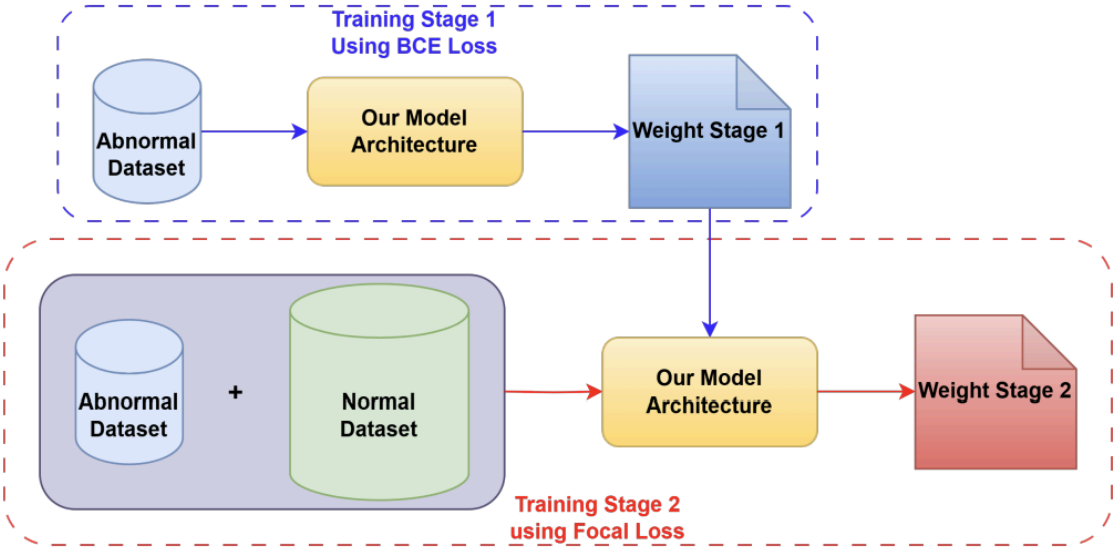


Fig 6: Two-Stage Training Strategy

5. Evaluation Metrics

To comprehensively evaluate performance on multi-label classification with class imbalance, we used the following metrics:

- **Area Under the Curve (AUC)**: Calculated per class and averaged
- **ROC Curve**: To visualize sensitivity/specificity trade-offs
- **Grad-CAM Visualizations**: For interpretability of attention maps

6. Implementation Plan

6.1 Development Tools

- **Language**: Python 3.8
- **Frameworks**: PyTorch, NumPy, OpenCV, Scikit-learn
- **Visualization**: Matplotlib, Seaborn
- **Version Control**: GitHub
- **Platform**: Kaggle (NVIDIA T4 GPU)

6.2 Implementation Steps

1. Load and preprocess ChestXray14 dataset
2. Implement AWB modules using LKA design
3. Modify VGG16 to include AWBs at strategic locations
4. Train model in two phases using BCE and Focal Loss
5. Evaluate and analyze performance
6. Visualize attention using Grad-CAM
7. Compare with baselines (CheXNet, SynthEnsemble)

Code repo: https://github.com/NNNguyenDuyyy/DSP391m_GROUP4

7. Ethical Considerations

- **Data Privacy**: ChestXray14 is a publicly available, de-identified dataset with no personally identifiable information.
- **Bias and Fairness**: Efforts were made to reduce dataset bias by using balanced training and fair evaluation. However, demographic attributes (e.g., age, sex) were not used in the analysis.
- **Clinical Use**: This model is designed for research purposes only. It is not FDA-approved and should not be deployed for real-world diagnosis without further validation.
- **Transparency**: All code and model checkpoints are publicly available to encourage reproducibility and transparency.

V. System Design and Implementation

1. AI Model Integration

AWBs are embedded into CNN at architectural level. Input X-ray → CNN + AWB → Multi-label Output

2. Data Flow and Processing

Image → Preprocess → CNN Backbone → AWBs → Prediction (14 disease labels)

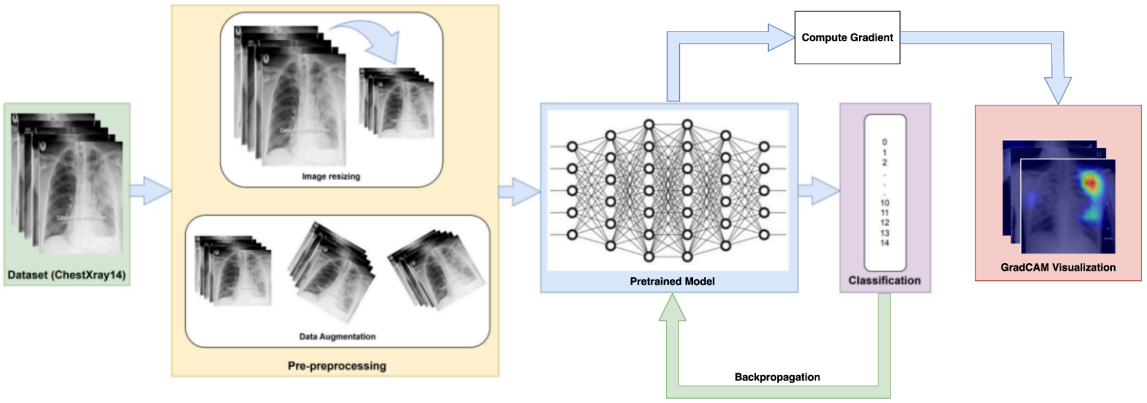


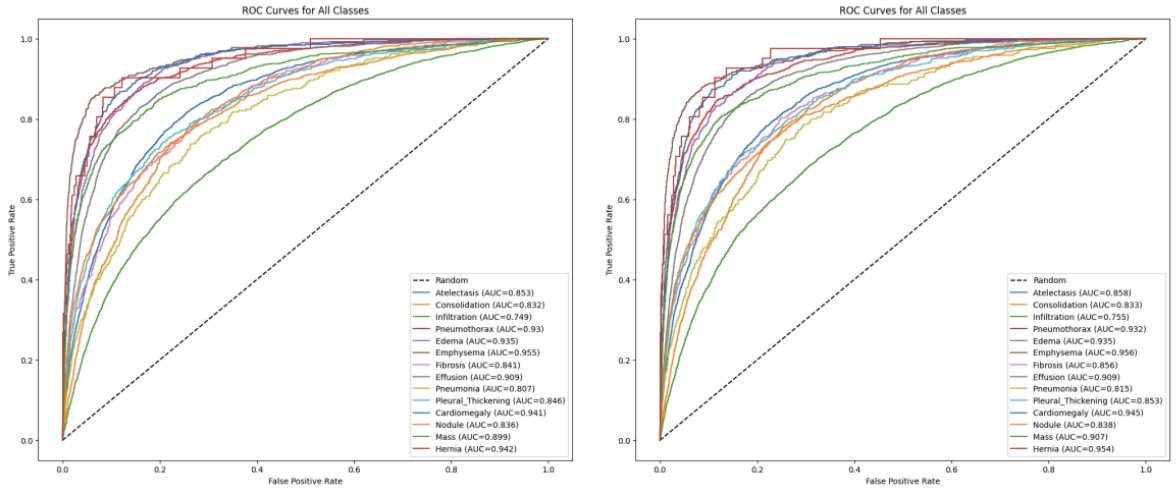
Fig 7: System Workflow

VI. Results and Discussion

1. Results and Analysis

We evaluated our proposed Attention-Wise Block (AWB)-based VGG16 architecture on the **ChestXray14** dataset, which contains over 100,000 frontal-view X-ray images labeled with 14 thoracic disease categories. We adopted the standard patient-wise data split to avoid label leakage and ensure clinically relevant evaluation.

The **main evaluation metric** used was the Area Under the Receiver Operating Characteristic Curve (**AUC**), which is widely accepted in medical image analysis for imbalanced, multi-label classification.



(a) Ours (BCELoss 2 Stage). (b) Ours (BCE + FocalLoss).

Fig 8: ROC AUC

1.1 Overall Performance

Pathology	Wang et al.[1]	Rajpurkar et al.[5]	Ashraf et al.[7]	Guan et al.(1)[16]	Guan et al.(2)[16]	Ours(1)	Ours(2)
Atelectasis	0.716	0.8094	0.83390	0.844	0.853	0.853	**0.858**
Cardiomegaly	0.807	0.9248	0.91954	0.937	0.939	0.941	**0.945**
Effusion	0.784	0.8638	0.88977	0.904	0.903	**0.909**	**0.909**
Infiltration	0.609	0.7345	0.74102	0.753	0.754	0.749	**0.755**
Mass	0.706	0.8676	0.87315	0.893	0.902	0.899	**0.907**
Nodule	0.671	0.7802	0.80611	0.827	0.828	0.836	**0.838**
Pneumonia	0.633	0.7680	0.77648	0.776	0.774	0.807	**0.815**
Pneumothorax	0.806	0.8887	0.90164	0.919	0.921	0.930	**0.932**
Consolidation	0.708	0.7901	0.81575	**0.842**	**0.842**	0.832	0.833
Edema	0.835	0.8878	0.91034	0.919	0.924	**0.935**	**0.935**
Emphysema	0.815	0.9371	0.92946	0.941	0.932	0.955	**0.956**
Fibrosis	0.769	0.8047	0.83347	0.857	**0.864**	0.841	0.856
Pleural Thickening	0.708	0.8062	0.81270	0.836	0.837	0.846	**0.853**
Hernia	0.767	0.9164	0.91723	0.903	0.921	0.942	**0.954**
Mean AUC	0.738	0.841	0.85433	0.868	0.871	0.8768	**0.8818**

Table 2: Our Results Compared to Previous Studies

Ours				
Model variant	**Training stage 1**	**Training stage 2**	**Total**	**GFLOPS**
Densenet121+LKA full	6563.2s	8588.8s	15152s	3.64
Densenet121+LKA 2 block	6375.6s	8239s	14614.6	3.14
Densenet121+LKA after	5611.4s	6957.3s	12568.7s	2.92
VGG16+LKA full	14225.2s	10206.1s	24431.3s	16.70
VGG16+LKA 3 block	8628.6s	5682.1s	14310.7s	15.93
VGG16+LKA after	4851.2s	5492.6s	10343.8s	15.39

Ours		
Model variant	**Trainable params**	**Total params**
Densenet121+lka full	9609870	9609870
Densenet121+lka 2 block	9223054	9223054
Densenet121+lka after	8095630	8096530
VGG16+lka full	15453966	15453966
VGG16+lka 3 block	15418702	15418702
VGG16+lka after	15031886	15031886

Synth Ensemble				
Model	**Trainable Params**	**Total Params**	**FLOPS (GFLOPS)**	**Training time**
CoAtNet	1128320	73904264	14.55	Stage 1: 21045.6s Stage 2: 6811.2s
DenseNet121	1144512	8014720	2.87	Stage 1: 40958.1s
MaxViTV2	1122944	116128376	23.88	Stage 1: 13791.6s
SwinV2	842240	49725140	9.08	Stage 1: 31071.3s
VOLO D2	319872	57923696	14.24	Stage 1: 32398.5s
convnextv2	1647488	198007040	34.4	Stage 1: 35538.9 Stage 2: 11597.6s
Total	6205376	503703236	99.02	~193212.8s

Table 3: Our Parameters and Training Time Compared to Synth Ensemble

Our model **achieved a new state-of-the-art mean AUC of 0.8818**, outperforming both earlier CNN-based baselines (e.g., CheXNet, DenseNet121) and hybrid transformer models.

1.2 Per-Class AUC Improvements

We observed substantial improvements on underrepresented diseases, which are typically difficult to detect in imbalanced datasets:

- **Hernia**: 0.954 (previous best: 0.921)
- **Edema**: 0.935 (previous best: 0.924)
- **Pneumonia**: 0.815 (previous best: 0.776)

These gains demonstrate the ability of our architecture to **enhance feature learning on rare pathologies** via attention-guided modules and training rebalancing.

1.3 Ablation Study Results

Model Variant	One-Stage BCE	One-Stage Focal	Two-Stage
DenseNet121	0.8227	0.8044	0.8402
DenseNet121 + AWBs	0.8340	0.8369	**0.8498**
VGG16	0.8275	0.8094	0.8401
VGG16 + AWBs	0.8238	0.8247	**0.8818**

Table 4: Impact of Training Strategy on Mean AUC

Model	Position	Mean AUC
DenseNet121 + LKA	After	0.8475
DenseNet121 + LKA	Full Block	0.8416
DenseNet121 + LKA	Last 2 Block	**0.8498**
VGG16 + LKA	After	0.8470
VGG16 + LKA	Full Block	0.8666
VGG16 + LKA	Last 3 Block	**0.8818**

Table 5: Impact of AWBs placement on Mean AUC

These results confirm that **combining a two-stage training pipeline with AWB placement in deeper layers of the CNN significantly improves performance**, especially compared to one-stage.

2. Discussion

2.1 The experimental results validate our two key hypotheses:

1. **Integrating attention mechanisms enhances feature discriminability**, especially for subtle and spatially diffuse abnormalities in CXR images.

2. **Addressing class imbalance through a two-stage training strategy**—first training on abnormal samples and later rebalancing with Focal Loss—enables better generalization, particularly for rare classes.

2.2 Why Our Method Performs Better

- **AWBs improve both spatial and channel-wise attention**, which helps the model identify not just what features are important but also where they occur.

- The **hierarchical placement** of AWBs in blocks 3–5 aligns with semantic depth in CNNs, boosting high-level decision-making.

- **Focal Loss** in the second training stage helps the model avoid being overwhelmed by the "No Finding" majority class, which often dominates learning in medical datasets.

2.3 Comparison with Prior Work

- **CheXNet** used DenseNet121 but lacked attention mechanisms and imbalance handling—limiting sensitivity to rarer conditions.

- **SynthEnsemble** introduced hybrid ensembles but did not offer as lightweight or interpretable a model.

- **Transformer-based methods** such as SwinChex and MedViT require more data and computation, whereas our approach achieves competitive (and better) performance using just VGG16 + attention modules.

2.4 Visualization Insights

Grad-CAM visualizations show that AWBs lead to **sharper and more anatomically consistent activations**. Our LKA-inspired attention maps localize disease-relevant areas more precisely, acting as an implicit soft segmentation mechanism.

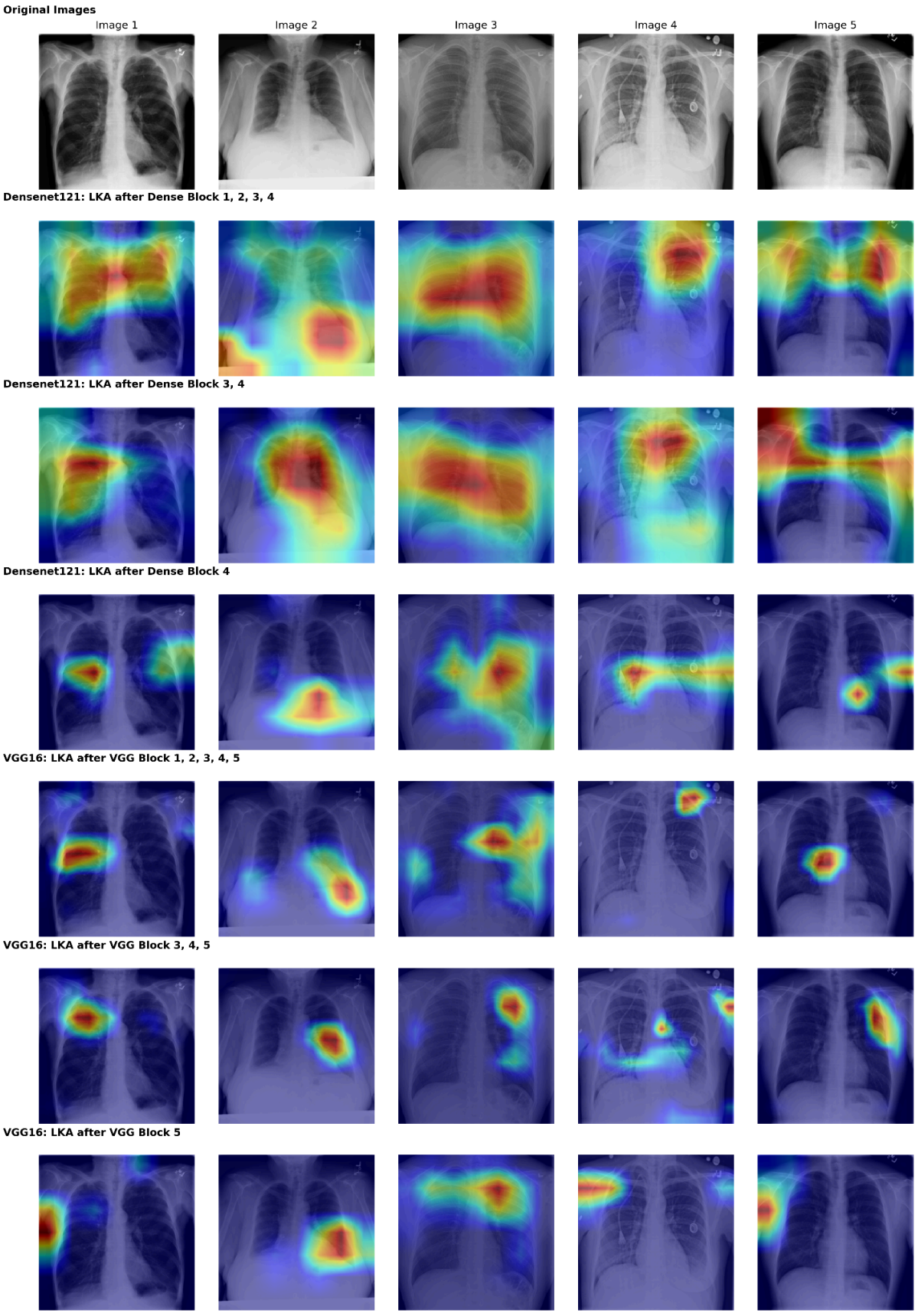


Fig 9: Grad-CAM Visualizations

3. Recommendations

Based on our results, we recommend the following for future AI-based medical imaging systems:

- Combine **architecture-level attention** with **loss-level imbalance handling** for better generalization in multi-label tasks.
- Prioritize **lightweight attention modules** like LKA for resource-constrained environments, e.g., in developing countries or rural clinics.
- Use **two-stage or curriculum-based training strategies** for datasets with extreme label imbalance.

VII. Conclusion

1. Summary of Findings

This project set out to address two significant challenges in chest X-ray diagnosis using deep learning: the difficulty of detecting co-occurring thoracic diseases and the severe class imbalance present in large medical datasets such as ChestXray14.

To tackle these issues, we proposed a novel framework that integrates **Attention-Wise Blocks (AWBs)**—based on the Large Kernel Attention (LKA) mechanism—into a **DenseNet121** and **VGG16 backbone**, combined with a **two-stage training pipeline** that isolates abnormal cases in early training and uses **Focal Loss** for fine-tuning.

Key findings include:

- Our method achieved a **mean AUC of 0.8818**, surpassing state-of-the-art models including CheXNet, SynthEnsemble, and DenseNet121.
- It performed especially well on **rare pathologies** like Hernia (AUC: 0.954), Pneumonia (AUC: 0.815), and Nodule (AUC: 0.838), validating the effectiveness of our approach on underrepresented classes.
- Grad-CAM visualizations confirmed that AWBs enabled the model to focus more precisely on disease-relevant anatomical regions.
- Ablation studies showed that both the **strategic module placement** and **two-phase curriculum training** contributed significantly to overall performance improvements.

These results strongly support the hypothesis that integrating attention mechanisms and tailored training strategies can dramatically enhance performance on complex, real-world medical image classification tasks.

2. Contributions and Reflections on the Project

2.1 Contributions

This capstone project contributes to the field in several meaningful ways:

- **Architectural Contribution**: Introduced AWBs as lightweight, interpretable modules that can be easily plugged into existing CNNs to enhance their spatial reasoning capabilities.

- **Training Strategy**: Designed a novel two-stage training procedure that balances representation learning and class imbalance handling, leading to significantly improved detection of rare conditions.

- **Open Access**: All source code and model weights are released publicly, promoting transparency, reproducibility, and potential clinical collaboration in the future.

- **Evaluation and Insight**: Conducted rigorous experiments, ablation studies, and visualizations to demonstrate the effectiveness and practicality of the proposed approach.

2.2 Reflections

Throughout this project, we learned to approach deep learning not only as a performance optimization problem but also as a **practical, interpretable, and ethical tool** in high-stakes domains like healthcare. We recognized that minor architectural innovations, when paired with thoughtful training design, can have a significant impact—even when using legacy architectures like VGG16.

We also encountered and overcame challenges related to computational constraints, data imbalance, and model generalization. These experiences enriched our understanding of deep learning workflows in applied AI projects and strengthened our collaborative and research skills.

3. Limitations and Future Work

3.1 Limitations

While our project achieved strong results, some limitations remain:

- **Dataset Boundaries**: Evaluation was conducted on ChestXray14 only. The model has not yet been tested on other datasets like CheXpert or MIMIC-CXR, which may vary in resolution, label quality, and demographic composition.

- **Lack of Clinical Validation**: Although Grad-CAM helped improve interpretability, we did not conduct formal clinical validation with radiologists to assess diagnostic trustworthiness.

- **Fixed Backbone**: Our experiments focused on VGG16. It is unclear how AWBs will generalize across modern architectures like EfficientNet or ResNet variants.

- **No Pathology Dependency Modeling**: The model does not explicitly consider inter-label dependencies (e.g., Infiltration and Effusion frequently co-occur).

3.2 Future Work

To build upon this project, we propose the following directions:

- **Cross-Dataset Evaluation**: Test the AWB-enhanced models on other public datasets to evaluate generalization and robustness.

- **Multimodal Learning**: Incorporate radiology reports using vision-language models for richer representations.

- **Graph-based Label Modeling**: Integrate label co-occurrence graphs or multi-task frameworks to improve understanding of pathological dependencies.

- **Clinical Collaboration**: Partner with hospitals to deploy the model in a low-risk diagnostic assistant setting and gather clinician feedback.

- **AutoML Exploration**: Use neural architecture search (NAS) to automate and optimize AWB placement across different CNN backbones.

VIII. References

1. Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., Summers, R.M.: Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2097–2106 (2017)

2. Johnson, A.E., Pollard, T.J., Berkowitz, S.J., Greenbaum, N.R., Lungren, M.P., Deng, C.y., Mark, R.G., Horng, S.: Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. Scientific data 6(1), 317 (2019)

3. Irvin, J., Rajpurkar, P., Ko, M., Yu, Y., Ciurea-Ilcus, S., Chute, C., Marklund, H., Haghgoo, B., Ball, R., Shpanskaya, K., et al.: Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 590–597 (2019)

4. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., Van Der Laak, J.A., Van Ginneken, B., S´anchez, C.I.: A survey on deep learning in medical image analysis. Medical image analysis 42, 60–88 (2017)

5. Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C., Shpanskaya, K., et al.: Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. arXiv preprint arXiv:1711.05225 (2017)

6. Yao, L., Poblenz, E., Dagunts, D., Covington, B., Bernard, D., Lyman, K.: Learning to diagnose from scratch by exploiting dependencies among labels. arXiv preprint arXiv:1710.10501 (2017)

7. Ashraf, S.N., Mamun, M.A., Abdullah, H.M., Alam, M.G.R.: Synthensemble: a fusion of cnn, vision transformer, and hybrid models for multi-label chest x-ray classification. In: 2023 26th International Conference on Computer and Information Technology (ICCIT). pp. 1–6. IEEE (2023)

8. Taslimi, S., Taslimi, S., Fathi, N., Salehi, M., Rohban, M.H.: Swinchex: Multilabel classification on chest x-ray images with transformers. arXiv preprint arXiv:2206.04246 (2022)

9. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)

10. Manzari, O.N., Ahmadabadi, H., Kashiani, H., Shokouhi, S.B., Ayatollahi, A.: Medvit: a robust vision transformer for generalized medical image classification. Computers in biology and medicine 157, 106791 (2023)

11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)

12. Yan, C., Yao, J., Li, R., Xu, Z., Huang, J.: Weakly supervised deep learning for thoracic disease classification and localization on chest x-rays. In: Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics. pp. 103–110 (2018)

13. Yan, Y., Kawahara, J., Hamarneh, G.: Melanoma recognition via visual attention. In: Information Processing in Medical Imaging: 26th International Conference, IPMI 2019, Hong Kong, China, June 2–7, 2019, Proceedings 26. pp. 793–804. Springer (2019)

14. Guo, M.H., Lu, C.Z., Liu, Z.N., Cheng, M.M., Hu, S.M.: Visual attention network. Computational visual media 9(4), 733–752 (2023)

15. Lin, T.Y., Goyal, P., Girshick, R., He, K., Doll´ar, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)

16. Guan, Q., Huang, Y., Zhong, Z., Zheng, Z., Zheng, L., Yang, Y.: Diagnose like a radiologist: Attention guided convolutional neural network for thorax disease classification. arXiv preprint arXiv:1801.09927 (2018)

17. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)

18. Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

20. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Gradcam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)

IX. Appendices

1. LKA in code

```python
class LKA(nn.Module):
    def __init__(self, dim):
        super().__init__()
        self.dw_conv1 = nn.Conv2d(dim, dim, kernel_size=5, padding=2, groups=dim)
        self.dw_conv2 = nn.Conv2d(dim, dim, kernel_size=7, padding=9, dilation=3, groups=dim)
        self.pw_conv = nn.Conv2d(dim, dim, kernel_size=1)

    def forward(self, x):
        u = x
        x = self.dw_conv1(x)
        x = self.dw_conv2(x)
        x = self.pw_conv(x)
        return x * u  # Attention applied element-wise
```

2. How we integrate LKA into CNN backbones in code

2.1 DenseNet121 + LKA Full Block

```python
class DenseNet121_fullLKA(nn.Module):
    def __init__(self, num_classes=1000, dropout=0.5):
        super(DenseNet121_fullLKA, self).__init__()
        densenet = models.densenet121(pretrained=True)
        self.features = densenet.features

        # Apply attention only in deeper blocks to save FLOPs
        self.att_block1 = LKA(256)
        self.att_block2 = LKA(512)
        self.att_block3 = LKA(1024)
        self.att_block4 = LKA(1024)

        self.avgpool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout)
        self.classifier = nn.Linear(1024, num_classes)
```

```python
def forward(self, x):
    x = self.features.conv0(x)
    x = self.features.norm0(x)
    x = self.features.relu0(x)
    x = self.features.pool0(x)

    x = self.features.denseblock1(x)
    x = self.att_block1(x)
    x = self.features.transition1(x)

    x = self.features.denseblock2(x)
    x = self.att_block2(x)
    x = self.features.transition2(x)

    x = self.features.denseblock3(x)
    x = self.att_block3(x)
    x = self.features.transition3(x)

    x = self.features.denseblock4(x)
    x = self.att_block4(x)

    x = self.features.norm5(x)
    x = self.avgpool(x)
    x = torch.flatten(x, 1)
    x = self.dropout(x)
    x = self.classifier(x)
    return x
```

2.2 DenseNet121 + LKA 2 Block

```python
class DenseNet121_fullLKA(nn.Module):
    def __init__(self, num_classes=1000, dropout=0.5):
        super(DenseNet121_fullLKA, self).__init__()
        densenet = models.densenet121(pretrained=True)
        self.features = densenet.features

        # Apply attention only in deeper blocks to save FLOPs
        #self.att_block1 = LKA(256)
        #self.att_block2 = LKA(512)
        self.att_block3 = LKA(1024)
        self.att_block4 = LKA(1024)

        self.avgpool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout)
        self.classifier = nn.Linear(1024, num_classes)
```

```python
def forward(self, x):
    x = self.features.conv0(x)
    x = self.features.norm0(x)
    x = self.features.relu0(x)
    x = self.features.pool0(x)

    x = self.features.denseblock1(x)
    #x = self.att_block1(x)
    x = self.features.transition1(x)

    x = self.features.denseblock2(x)
    #x = self.att_block2(x)
    x = self.features.transition2(x)

    x = self.features.denseblock3(x)
    x = self.att_block3(x)
    x = self.features.transition3(x)

    x = self.features.denseblock4(x)
    x = self.att_block4(x)

    x = self.features.norm5(x)
    x = self.avgpool(x)
    x = torch.flatten(x, 1)
    x = self.dropout(x)
    x = self.classifier(x)
    return x
```

2.3 DenseNet121 + LKA After

```python
class densenet_LKA_after(nn.Module):
    def __init__(self, num_classes, dropout=0.5):
        super(densenet_LKA_after, self).__init__()
        base = models.densenet121(pretrained=True)

        # Full backbone (keep as-is)
        self.features = base.features

        # Add LKA after norm5
        self.lka = LKA(dim=1024)

        # Classification head
        self.pool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout) if dropout else nn.Identity()
        self.classifier = nn.Linear(1024, num_classes)

    def forward(self, x):
        x = self.features(x)      # DenseNet121 full features (includes norm5)
        x = self.lka(x)           # LKA-enhanced final feature map
        x = self.pool(x).flatten(1)
        x = self.dropout(x)
        x = self.classifier(x)
        return x
```

2.4 VGG16 + LKA Full Block

```python
class VGG16_LKA(nn.Module):
    def __init__(self, num_classes, dropout=None):
        super().__init__()
        base = models.vgg16_bn(pretrained=True)

        self.block1 = nn.Sequential(*base.features[0:6])    # 64
        self.block2 = nn.Sequential(*base.features[7:13])   # 128
        self.block3 = nn.Sequential(*base.features[14:23])  # 256
        self.block4 = nn.Sequential(*base.features[24:33])  # 512
        self.block5 = nn.Sequential(*base.features[34:43])  # 512

        # Insert LKA modules after block3, block4, block5
        self.lka1 = LKA(dim=64)
        self.lka2 = LKA(dim=128)
        self.lka3 = LKA(dim=256)
        self.lka4 = LKA(dim=512)
        self.lka5 = LKA(dim=512)

        self.pool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout) if dropout else nn.Identity()
        self.classifier = nn.Linear(512, num_classes)
```

```python
def forward(self, x):
    x = self.block1(x)
    x = self.lka1(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block2(x)
    x = self.lka2(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block3(x)
    x = self.lka3(x)  # Attention-enhanced features
    x = F.max_pool2d(x, 2, 2)

    x = self.block4(x)
    x = self.lka4(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block5(x)
    x = self.lka5(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.pool(x).flatten(1)
    x = self.dropout(x)
    x = self.classifier(x)
    return x
```

2.5 VGG16 + LKA 2 Block

```python
class VGG16_LKA(nn.Module):
    def __init__(self, num_classes, dropout=None):
        super().__init__()
        base = models.vgg16_bn(pretrained=True)

        self.block1 = nn.Sequential(*base.features[0:6])    # 64
        self.block2 = nn.Sequential(*base.features[7:13])   # 128
        self.block3 = nn.Sequential(*base.features[14:23])  # 256
        self.block4 = nn.Sequential(*base.features[24:33])  # 512
        self.block5 = nn.Sequential(*base.features[34:43])  # 512

        # Insert LKA modules after block3, block4, block5
        self.lka3 = LKA(dim=256)
        self.lka4 = LKA(dim=512)
        self.lka5 = LKA(dim=512)

        self.pool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout) if dropout else nn.Identity()
        self.classifier = nn.Linear(512, num_classes)
```

```python
def forward(self, x):
    x = self.block1(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block2(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block3(x)
    x = self.lka3(x)  # Attention-enhanced features
    x = F.max_pool2d(x, 2, 2)

    x = self.block4(x)
    x = self.lka4(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.block5(x)
    x = self.lka5(x)
    x = F.max_pool2d(x, 2, 2)

    x = self.pool(x).flatten(1)
    x = self.dropout(x)
    x = self.classifier(x)
    return x
```

2.6 VGG16 + LKA After

```python
class VGG_LKA_after(nn.Module):
    def __init__(self, num_classes, dropout=0.5):
        super(VGG_LKA_after, self).__init__()
        base = models.vgg16_bn(pretrained=True)

        # Full backbone (keep as-is)
        self.features = base.features

        # Add LKA after norm5
        self.lka = LKA(dim=512)

        # Classification head
        self.pool = nn.AdaptiveAvgPool2d((1, 1))
        self.dropout = nn.Dropout(dropout) if dropout else nn.Identity()
        self.classifier = nn.Linear(512, num_classes)

    def forward(self, x):
        x = self.features(x)      # DenseNet121 full features (includes norm5)
        x = self.lka(x)           # LKA-enhanced final feature map
        x = self.pool(x).flatten(1)
        x = self.dropout(x)
        x = self.classifier(x)
        return x
```

- THE END -