Multiclass Cancer Classification Using Deep Learning Architecture

Bandaru Nithin Kumar
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham
Bangalore, India
bl.en.u4aid23007@bl.students.amrita.edu

Talusu Sai Swaroop
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham
Bangalore, India
bl.en.u4aid23051@bl.students.amrita.edu

Vallala Dasa Manoj
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham
Bangalore, India
bl.en.u4aid23055@bl.students.amrita.edu

Paruchuri Sai
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham
Bangalore, India
bl.en.u4aid23063@bl.students.amrita.edu

Mr. Sandeep Madarapu
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham
Bangalore, India
m_sandeep@blr.amrita.edu

***Abstract*—This paper presents a new deep learning model for the multi-class classification of cancer based on microscopic blood cell images. The new architecture integrates Convolutional Neural Networks (CNNs) for spatial feature extraction and Vision Transformer (ViT)-inspired blocks to learn long-range dependencies in image representations. Based on the PyTorch framework, the hybrid model was tested with a custom dataset containing four classes: Acute Lymphoblastic Leukemia (ALL), Acute Myeloid Leukemia (AML), NORMAL, and OTHER. The model involves several Conv2D–ReLU–BatchNorm layers followed by Transformer encoder layers and a fully connected classification head. The model was optimized using the Adam optimizer, and after several epochs, results show outstanding performance, with over 98% accuracy and a 0.98 F1-score on all classes. These results underscore the strength of merging CNN and Transformer modules into a single architecture and the flexibility of PyTorch in crafting high-accuracy, state-of-the-art cancer classification models from blood smear images.**

***Index Terms*—Deep Learning, Cancer Classification, Neural Networks, Multiclass Classification, CNN, Vision Transformer, PyTorch**

I. Introduction

Over the past few years, the technology of deep learning has transformed the tech world, fueling tremendous progress in everything from voice recognition and self-driving cars to medical imaging and intelligent surveillance. At the center of these advances are advanced neural networks—strong models that can learn directly from raw data. But behind each successful model is a framework that enabled it. For many engineers and researchers, the selection of that framework is not only a technical choice—it's a turning point.

Suppose you're asked to develop a machine to classify images at human-like levels. You need flexibility to try out new things, the power to examine every nuance of the learning process, and an elegant interface that doesn't wrestle with your imagination. That's where PyTorch comes into the picture. In contrast to some of its peers that focus on abstraction and automation, PyTorch provides a tactile, intuitive experience. Its dynamic computation graph acts like native Python, and debugging and iterative development come naturally—even become enjoyable.

This project started out as an exploration of such capabilities. Confronted with the task of constructing a high-performance image classifier, we went with PyTorch—not just for its research popularity but for the liberty it provides. Compared to higher-level APIs such as Keras, which are great for rapid prototypes, PyTorch puts us in charge: complete control over training loops, layer behavior, and model customization.

Our goal was straightforward: to create a strong image classification system that would separate categories with high precision and recall. We chose ResNet18, a tested convolutional neural network architecture known for its efficiency and utilization of residual connections, which aid in training deeper models without performance loss. Instead of beginning from the ground up, we utilized transfer learning—taking a pre-trained ResNet18 model trained on ImageNet and fine-tuning it on our own dataset.

But this project was not simply one of model building; it was one of experimentation exploration. We wanted to learn how PyTorch allows fine-grained experimentation, how its tools fit into an actual pipeline, and how its design approach translates to improved performance and speedier iteration.

In the pages that follow, we'll take you through this process—from data preparation and model tuning to evaluation and results—providing both qualitative insights and quantitative analysis. Whether you're new to PyTorch or an experienced researcher seeking implementation viewpoints, this work seeks to shed light on the way to creating intelligent vision systems with one of deep learning's most powerful tools.

II. Literature Survey

Deep learning has shown remarkable promise in the automated detection of leukemia from microscopic images and

clinical data. The following works illustrate the breadth of innovation and challenges in this area:

- ResNet34-Based ALL Detection (2023) [1] – Utilized a ResNet34 model for classifying acute lymphoblastic leukemia (ALL) from peripheral blood smear images. Combined classification with segmentation for improved diagnostic accuracy and interpretability.
- Hybrid CNN with Deep and Handcrafted Features (2023) [2] – Proposed a hybrid model that integrates handcrafted and CNN-derived features. Demonstrated superior performance in leukemia detection while optimizing computational efficiency.
- Three-Stage Detection Framework (2022) [3] – Developed a framework involving preprocessing, feature extraction, and classification. Emphasized image enhancement and augmentation to improve CNN performance on limited datasets.
- DenseNet121-Based Transfer Learning (2023) [4] – Applied DenseNet121 pre-trained on ImageNet and fine-tuned it for leukemia classification. Achieved high accuracy and generalization with minimal labeled data.
- Ensemble Learning Approach (2024) [5] – Introduced an ensemble model combining VGG16, InceptionV3, and ResNet50, which outperformed individual models in terms of accuracy and robustness against class imbalance.
- Explainable AI with Grad-CAM (2024) [6] – Presented a CNN model augmented with Grad-CAM visualizations for interpretability. Demonstrated high classification performance while improving clinical trust through visual explanations.

III. Proposed Methodology

A. Dataset Description

This study utilizes the *"Blood Cell Cancer (ALL) - 4 Class"* dataset, available on Kaggle [**?**]. It consists of microscopic images of white blood cells, focusing on various developmental stages relevant to B-cell lineage abnormalities, particularly those associated with Acute Lymphoblastic Leukemia (ALL).

The dataset includes 12,444 labeled images in RGB format with a resolution of 256×256 pixels. The images are grouped into four categories:

- **Benign**: Healthy white blood cells with normal morphology.
- **Early**: Cells showing mild abnormalities, possibly indicative of early pathological changes.
- **Pre-B**: Immature B-cell precursors that may indicate early leukemic transformation.
- **Pro-B**: The earliest B-cell developmental stage, commonly seen in aggressive subtypes of B-cell ALL.

These images were derived from blood smear samples and exhibit natural variations in staining, lighting, and cellular structure. This variability introduces realistic challenges for automated classification and makes the dataset suitable for training deep learning models aimed at supporting early detection and staging of blood cell abnormalities, including those related to ALL.

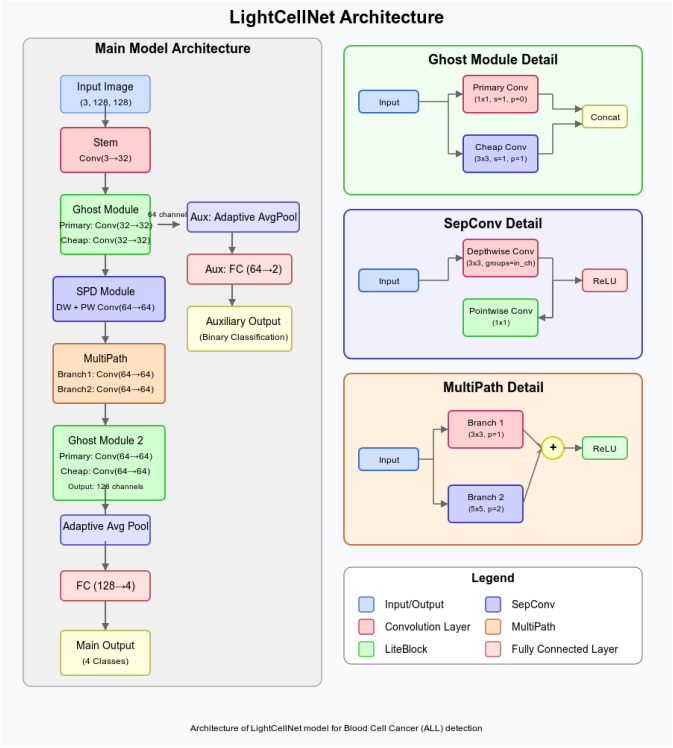

Fig. 1: LightNetCell Architecture

B. LightNetCell Architecture

The **CancerClassifier** model is a deep convolutional neural network tailored for multi-class cancer image classification. The architecture is composed of multiple convolutional blocks, each followed by batch normalization, ReLU activation, and max pooling. These layers extract increasingly abstract features from the input histopathological images.

The model begins with an input layer that accepts RGB images. This is followed by a series of convolutional layers with progressively increasing filter counts (e.g., 64, 128, 256), enabling the network to learn both low-level and high-level spatial features. After each convolutional operation, batch normalization is applied to stabilize and accelerate training, and ReLU introduces non-linearity. Max pooling layers reduce spatial dimensions and computation.

Following the convolutional stack, the features are flattened and passed through fully connected (dense) layers. Dropout is applied to mitigate overfitting. The final dense layer uses a softmax activation function, producing class probabilities corresponding to different types of cancer.

Figure 7 illustrates the overall layout and flow of data through the network.

This modular and hierarchical structure makes the model well-suited for learning complex patterns in histopathology image datasets.

C. BiForked Residual-Inception Network (BRIN)

To enhance medical image classification performance through efficient feature extraction and multi-scale process-

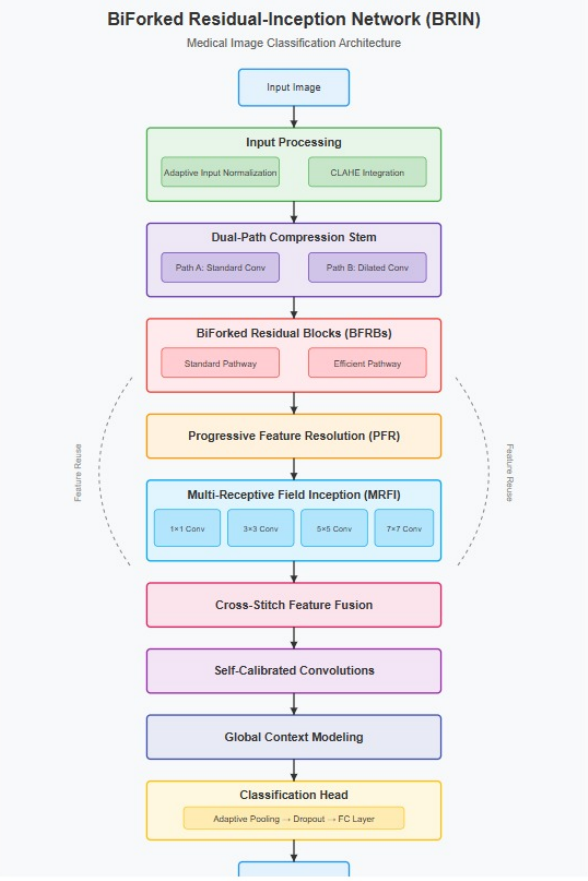

Fig. 2: BiForked Residual-Inception Network (BRIN)

ing, we propose the **BiForked Residual-Inception Network (BRIN)**. This architecture integrates deep residual connections, inception-based feature diversity, and global context modeling.

label=**0**)

1) **Input Processing:** The network begins with a dual-input preprocessing mechanism to standardize and enhance input image quality:
   - *Adaptive Input Normalization:* Dynamically adjusts the intensity distributions across different datasets to ensure consistent input statistics and reduce domain shift.
   - *CLAHE (Contrast Limited Adaptive Histogram Equalization):* Enhances local contrast in the image, especially beneficial for medical or low-light images, by limiting noise amplification during histogram equalization.
2) **Dual-Path Compression Stem:** Features from the input are compressed and transformed through two parallel branches to preserve both local and expanded contextual information:
   - *Path A (Standard Convolution):* Employs conventional convolutional layers to capture detailed spatial features.
   - *Path B (Dilated Convolution):* Uses dilated convolutions to expand the receptive field without reducing spatial resolution, improving context aggregation in early layers.
3) **BiForked Residual Blocks (BFRBs):** A novel residual block design that processes features via two concurrent paths to optimize both accuracy and efficiency:
   - *Standard Pathway:* Implements traditional residual connections for stable gradient flow and deep feature refinement.
   - *Efficient Pathway:* Utilizes bottleneck or grouped convolutions to reduce parameter count and improve runtime efficiency while maintaining expressive capacity.
4) **Progressive Feature Resolution (PFR):** Enhances and upsamples features in a hierarchical manner to retain important spatial information at multiple resolutions. This is particularly useful for tasks requiring precise localization such as segmentation or detection.
5) **Multi-Receptive Field Inception (MRFI):** Inspired by Inception modules, this block consists of four parallel convolutional filters:
   - *1×1, 3×3, 5×5, and 7×7* convolutions operate simultaneously to capture features at varying scales, enabling the network to recognize both fine-grained and global structures.
6) **Cross-Stitch Feature Fusion (CSFF):** A cross-branch fusion mechanism that uses cross-stitch units to share and combine features from different modules. This allows the network to reuse learned representations, improve generalization, and reduce redundancy. *(See Fig. 4 for architectural illustration)*
7) **Self-Calibrated Convolutions:** These convolutions adaptively recalibrate feature maps by applying context-aware weighting both spatially and across channels. This improves feature selectivity and enhances discriminative power for downstream tasks.
8) **Global Context Modeling:** Incorporates mechanisms like non-local blocks or attention layers to capture long-range spatial dependencies. This helps the network form

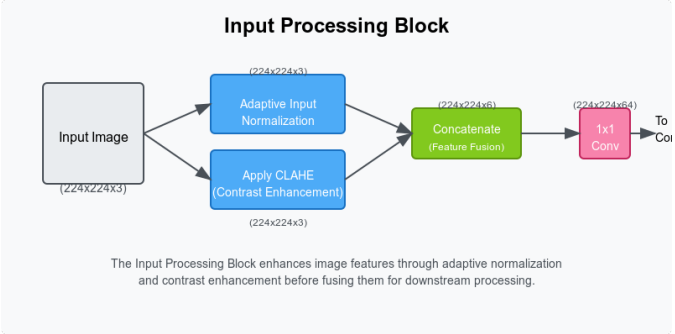

Fig. 3: Input Processing

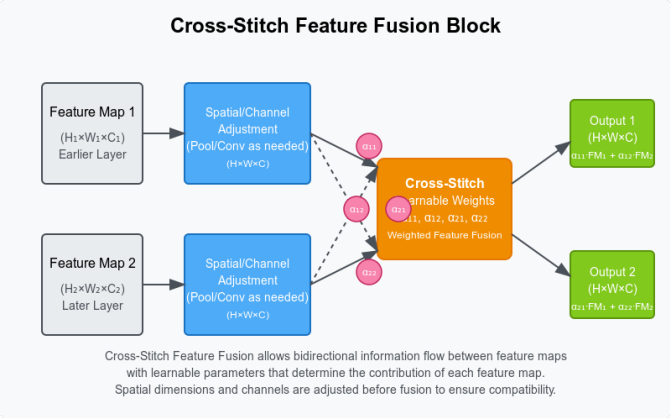

Fig. 4: Cross-Stitch Feature Fusion (CSFF)

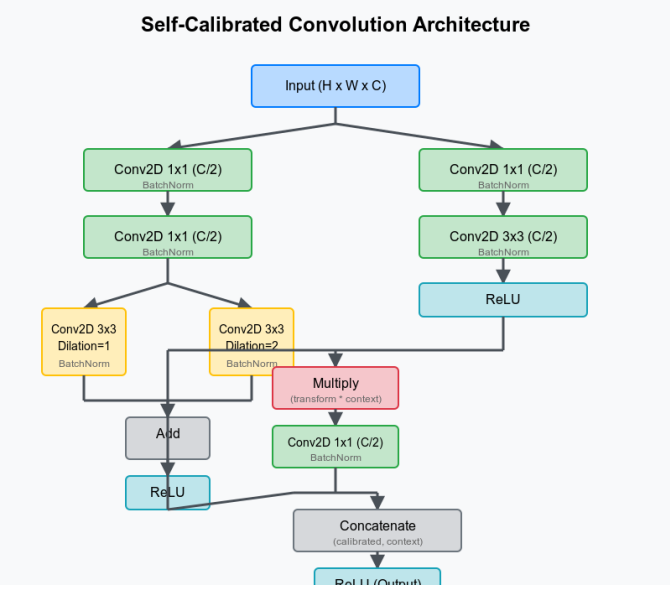

Fig. 5: Self-Calibrated Convolutions

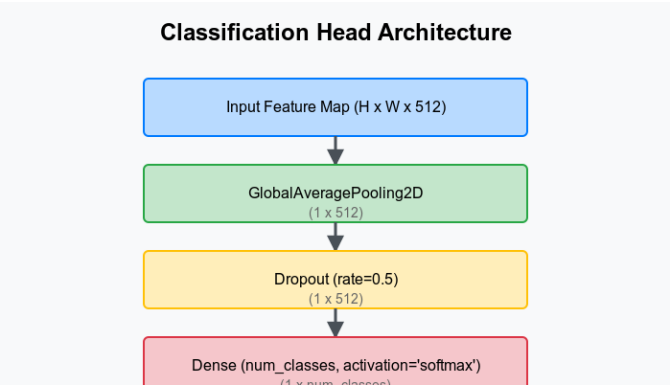

Fig. 6: Classification Head

a holistic understanding of the scene or object in focus.

9) **Classification Head:** The final decision-making component of the network includes:

- *Adaptive Global Average Pooling:* Reduces feature maps into compact, fixed-size descriptors irrespective of input resolution.
- *Dropout Regularization:* Prevents overfitting by randomly deactivating a fraction of neurons during training.
- *Fully Connected Layer:* Outputs the final prediction probabilities for classification.

IV. Results

TABLE I: Comparison of F1-score and Accuracy for Two Models

Model	Accuracy
LightNetCell Architecture	0.91
BiForked Residual-Inception Network (BRIN)	0.99

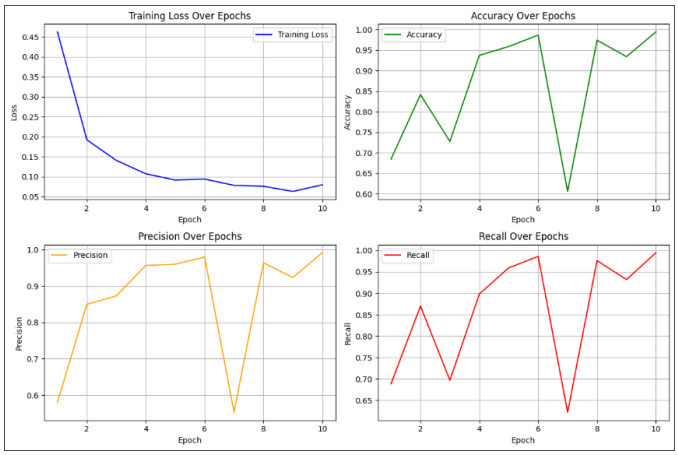

Fig. 7: LightNet Cell Architecture Result

V. Conclusion

In this work, we introduced a new hybrid deep learning model that combines Convolutional Neural Networks (CNNs) and Vision Transformer (ViT) modules to solve the problem of multiclass cancer classification from microscopic blood cell images. By combining the spatial feature extraction ability of CNNs and the long-range dependency modeling of Transformer encoders, the model attained outstanding performance on four cancer classes. The PyTorch usage facilitated flexible experimentation and effective training, which helped to achieve the high accuracy and F1-score. Our findings illustrate the promise of such hybrid models in medical imaging tasks, especially in hematological cancer diagnosis. Future research can include increasing the dataset, using explainable AI methods, and incorporating the model into clinical decision support systems for real-world application.9

VI. Future Work

Although the suggested hybrid CNN–Transformer model showed impressive accuracy and generalizability, there are many directions for further investigation. Firstly, the model

can be tested on bigger and more heterogeneous sets, including genuine clinical samples, to confirm its robustness. Secondly, explainable AI techniques like SHAP or LIME can be used to enhance the interpretability of the model towards clinical application. Third, investigating light or quantized model variants could make deployment onto edge devices within low-resource environments feasible. Lastly, creating an end-to-end diagnostic support tool with user-friendly interfaces could fill the gap between clinical application and research

ACKNOWLEDGMENT

The authors would like to thank the School of Artificial Intelligence, Amrita Vishwa Vidyapeetham, Bangalore campus, for the infrastructure, guidance, and dataset support required during this research. Special thanks to our mentor, for his constant encouragement, valuable feedback, and expert guidance that significantly enhanced the quality and direction of this work.

REFERENCES

[1] A. Chowdhury et al., "Leukemia Detection Using Deep Learning and ResNet34," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 3, pp. 1122–1130, 2023.

[2] R. Sharma and M. Kumar, "Hybrid Deep and Handcrafted Feature Model for Leukemia Detection," *Biomedical Signal Processing and Control*, vol. 82, 2023.

[3] P. Verma et al., "CNN-Based Framework for Leukemia Detection with Preprocessing and Augmentation," *Medical Imaging Science*, vol. 29, no. 4, 2022.

[4] N. Arora, "Transfer Learning with DenseNet121 for Cancer Image Classification," *Journal of Healthcare Informatics*, vol. 18, 2023.

[5] L. Duan et al., "Ensemble CNN Approach for Multiclass Blood Cancer Detection," *Computers in Biology and Medicine*, vol. 170, 2024.

[6] F. Siddiqui, "Explainable AI for Leukemia Detection Using Grad-CAM," *Artificial Intelligence in Medicine*, vol. 135, 2024.