Classification of Immunofluorescence (IIF)Staining patterns on Human Ethilial-2 (HEp-2) cells for Autoimmune diseases

AUTHORS

Megha Manoj (002043038)
Charul (002535330)

Contributions Per Member

- ◊ Charul & Megha: Literature review on image classification, Exploratory Data Analysis, Trained an Ensemble model using CNN, ViT as the first layer and performed fine-tuning for optimal performance, Created Presentation and Project Report.

- ◊ Megha: Constructed a custom preprocessing function for image stretching and normalization, Created a custom dataset to accommodate data augmentation via rotation and random brightness, Built a Convolutional Neural Network model(CNN) and CNN model with residual layers (ResCNN) evaluated performance using validation and test data, Conducted fine-tuning via addition of early stopping, Created demo.py.

- ◊ Charul: Performed preprocessing, including normalization, resizing, and preparing the dataset for ViT, Added rotation-based data augmentation, Implemented and trained a Vision Transformer (ViT), Created evaluation utilities such as accuracy/loss plots and a confusion matrix, Researched ViT performance to justify its effectiveness for HEp-2 cell classification.

ABSTRACT

Autoimmunity refers to a medical condition in which the antibodies present within the human body attack healthy cells. According to statistics, about 10% of the global population has been diagnosed with one or more autoimmune diseases. Although research and diagnostic techniques have been developed to identify these diseases, diagnosing individuals with a specific autoimmune disease has proven challenging due to the overlapping symptoms among these medical conditions. The application of machine learning for automatic identification of stain patterns in the Indirect Immunofluorescence Antinuclear Antibody test (IIF-ANA) on Human epithelial cells (Hep2 cells)

is a popular method used in the automatic diagnosis of many autoimmune diseases using the ICPR 2012 image dataset. While several state-of-the-art models have been implemented to classify the patterns, the class imbalance present within the dataset has acted as a major factor in underperformance of the models. Through this paper, we propose a stack ensemble model and conduct a comparison on its performance with the individual models used in it as well as a modified convolutional neural network with residual blocks to improve generalization and reduce overfitting of the model.

I. INTRODUCTION

The application of machine learning in combination with diagnostic techniques has improved the efficiency of early detection of autoimmune diseases via computer vision. One such diagnostic technique is pattern recognition of stains from Indirect Immunofluorescence Antinuclear Antibody test (IIF-ANA) on Human epithelial cells (Hep2 cells) to narrow down the number of Autoimmune diseases one may have.

The IIF-ANA test uses Hep-2 cells, obtained from human laryngeal (voice box) epidermoid carcinoma, which is used as a substrate for autoantibodies present within the patient serum to bind to. A process using fluorescent-stained antibodies then results in staining cellular components of the Hep2 cells which indicate possible autoimmune diseases that the patient may have.

The examination of large sets of cells for determining the presence of such patterns is a time-consuming process and susceptible to chances of human error due to the presence of similar features between various stain patterns. This led to the research and development of several pattern recognition techniques involving implementation of image classification and segmentation models as well as varying preprocessing techniques for automatic identification of stain patterns. However,

due to lack of availability of data pertaining to certain rarely occurring stain patterns which correlate to rare autoimmune diseases, a class imbalance is present within the available dataset which has resulted in poor generalization when tested on unseen samples.

Contributions

This work makes the following contributions:

- ◊ Implementation of a CNN architecture with 9 convolutional layers and batch normalization for HEp-2 classification
- ◊ Development of a Residual CNN (ResCNN) with skip connections to improve generalization and reduce overfitting
- ◊ Adaptation of Vision Transformer (ViT) for medical image classification, leveraging self-attention for global pattern recognition
- ◊ Development of a stack ensemble combining CNN and ViT with a linear meta-layer
- ◊ Custom preprocessing pipeline with image stretching using 1st and 99th quantile intensities
- ◊ Data augmentation strategy using rotation (18° multiples) and random brightness adjustment
- ◊ Comprehensive evaluation achieving 95% accuracy with balanced precision-recall metrics

II. PRIOR WORK

The classification of HEp-2 cell images has been an active research area. Early approaches relied heavily on hand-crafted features based on expert knowledge. Gao and others (2014) proposed a deep CNN framework that extracted features directly from raw pixels, they showed that data augmentation through 18° rotation intervals significantly improved performance [9]. The paper by

Cheng-Chia Huang et. Al [3] introduces a modified version of the CNN model for reduction of overfitting and increase the model's ability to generalize unseen data by addition of residual blocks and a global average pooling layer. The model showcased remarkable improvement with a root mean squared error (RMSE) of 134 in comparison to primitive models that had RMSE of 294.51. The ideal approach to prevent overfitting would be to train on ResNet models. However, the application of ResNet is computationally expensive and although the training process can be optimized via the use of pretrained weights, the upscaling small images to size 224x244 would result in reduce the model's ability to generalize while training. [4]

The use of an ensemble model would, in cases of overfitting, allow combining individual models to learn the discriminative features, giving a high accuracy rate as done by P. H. Kasan et. Al using Inception V3 and Xception architectures via transfer learning to achieve an accuracy of 95.07%. The approach had the drawback of increased computational expense.[5]

Vision Transformers (ViT), introduced by Dosovitskiy and others (2020) [6], brought the transformer architecture from natural language processing to computer vision. ViT divides images into patches, treats them as sequences, and applies self-attention mechanisms. This approach enables modeling of long-range dependencies and global spatial relationships, making it suitable for capturing global staining patterns in HEp-2 cells.

III. METHODOLOGY

A. Overview

Our proposed methodology involved training four models using the steps as depicted in Figure 1: a convolutional neural network (CNN), modified CNN with residual blocks (ResCNN), a Vision

Transformer (ViT), and a stack ensemble model comprising ViT and CNN as the first layer with a linear meta-layer.

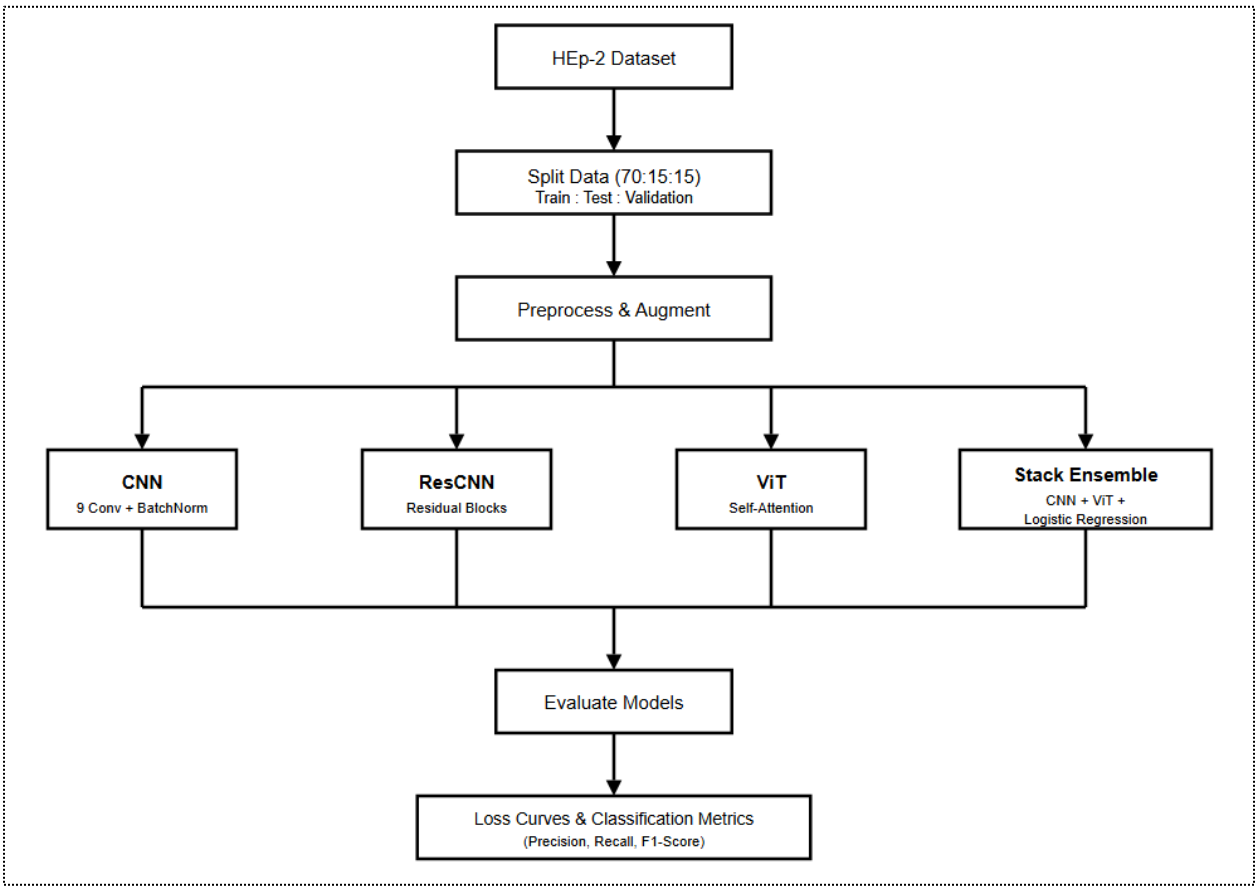


Figure 1. Overview of the proposed methodology pipeline for HEp-2 cell classification.

B. Preprocessing

Preprocessing for all models involved the same steps. Each dataset was converted into PyTorch tensor format. Image enhancement was applied by image stretching using intensities at the 1st and 99th quantile in order to normalize and standardize the images. The train dataset was further augmented to account for the data imbalance via image rotation at 18° in multiples of 1 to 5 (18°, 32°, 54°, 72° and 90°) and randomized adjustment of brightness to introduce variation in the augmentation.

C. Convolutional Neural Network

The convolutional neural network (Figure 2) consists of 9 convolutional layers with each layer followed by a Batch Normalization layer for prevention of overfitting. The first Sequential block visible comprises a convolutional layer followed by a batch normalization layer with a 3x3 kernel, padding of value 2 and a stride of 1. The next 4 layers are blocks consisting of 2 convolutional layers of kernel size 3x3 and a stride of 1. After the first convolutional and batch normalization layer in the Block, a ReLU layer is present for activation. Additionally, before the output layer. A global average pooling layer is added to reduce the number of features involved in learning the training data.

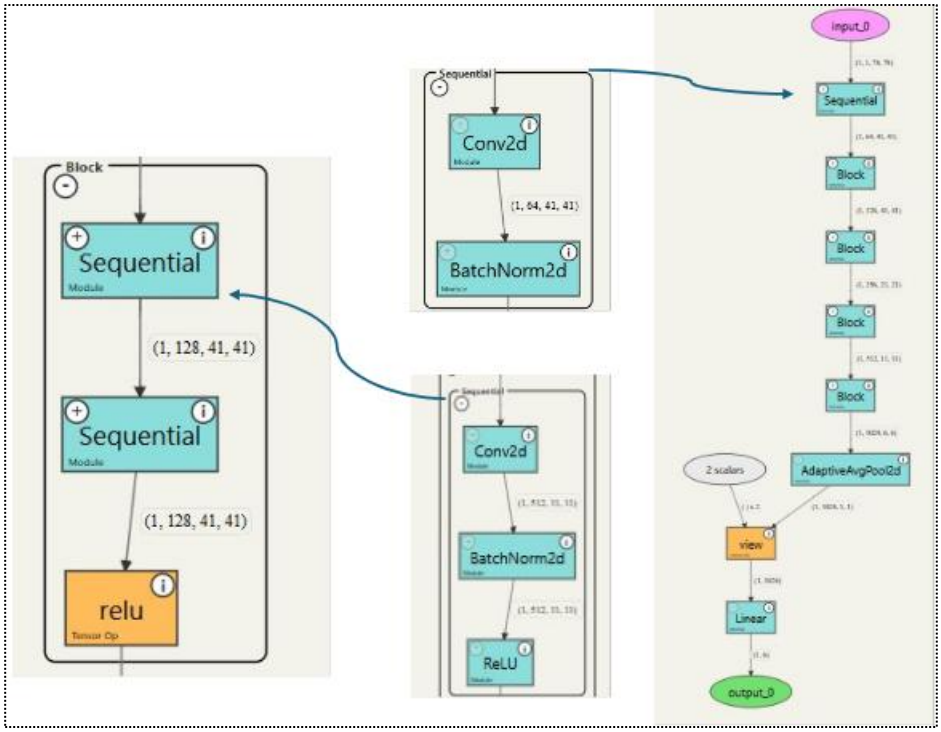


Figure 2. Architecture of the convolutional neural network

D. Residual CNN (ResCNN)

The ResCNN (Figure 3) followed a similar structure to CNN but incorporated residual blocks with skip connections. The architecture consisted of 9 convolutional layers with batch normalization. The 4 residual blocks each contained 2 convolutional layers (3×3 kernel, stride 1) with ReLU activation. Skip connections allowed gradients to flow directly through the network, addressing vanishing gradient problems and improving generalization. Furthermore, like the CNN model, a global average pooling layer has been added as well.

E. Vision Transformer (ViT)

The ViT architecture (Figure 4) processed images by dividing the input into fixed-size patches, linearly embedding each patch, adding positional embeddings to retain spatial information, processing through transformer encoder layers with multi-head self-attention, and using a learnable classification token for final prediction. This architecture captured global spatial relationships through self-attention mechanisms.

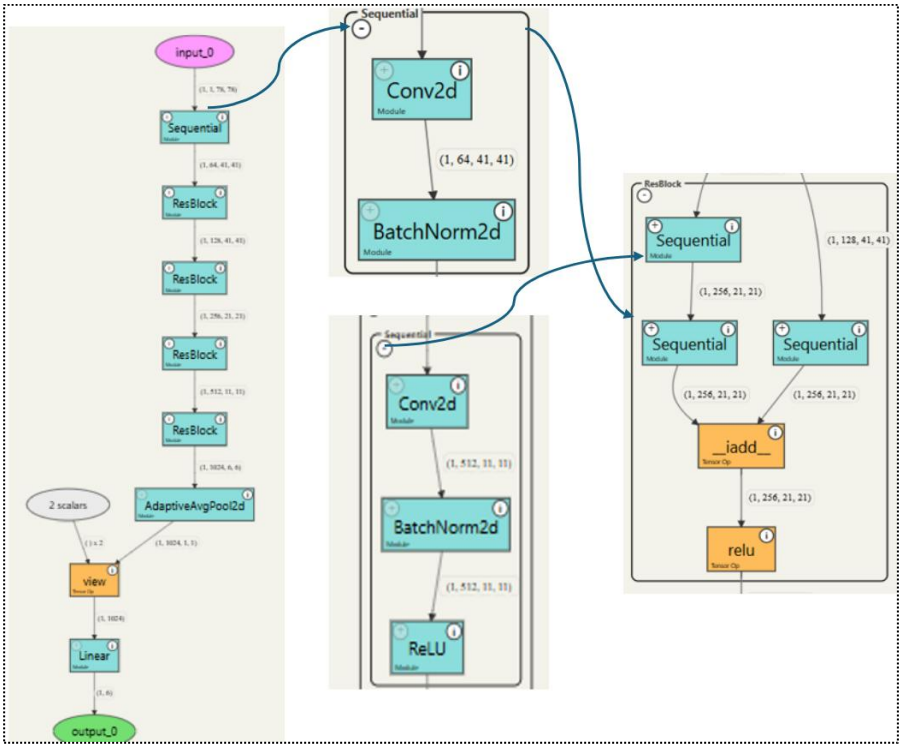


Figure 3. Architecture of the residual CNN

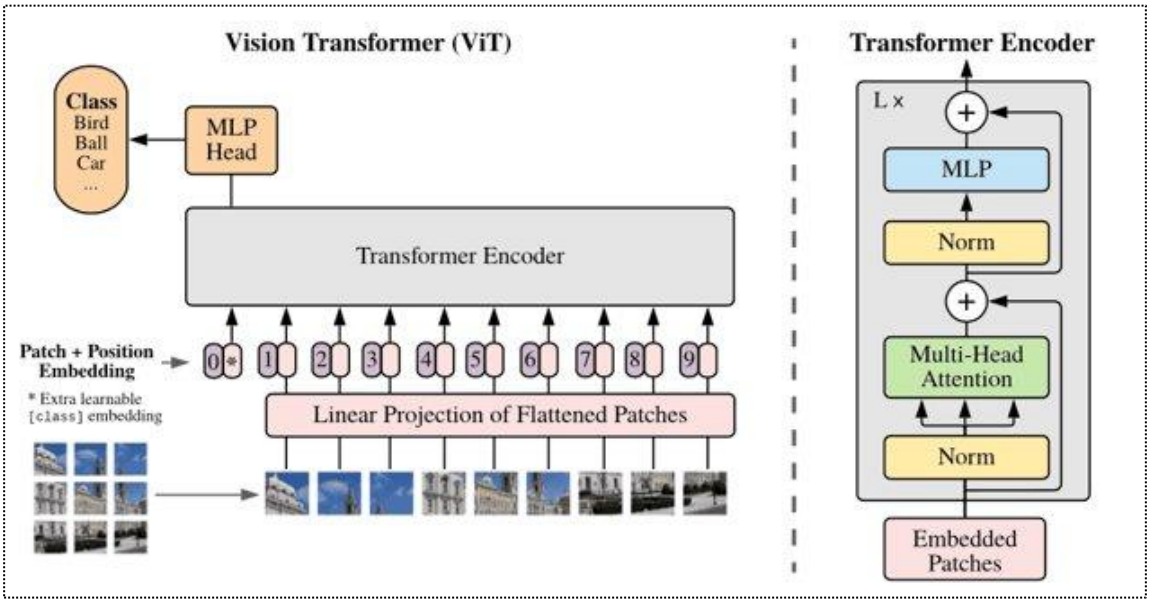


Figure 4. Vision Transformer Architecture

F. Stack Ensemble Model

The ensemble combined CNN and ViT (Figure 5) through a stacking approach. Input images (1×78×78) were passed to both CNN and ViT simultaneously. Each model outputs a 6-dimensional probability vector. Outputs were concatenated into a 12-dimensional feature vector. A linear layer mapped concatenated features to final 6-class predictions. This approach reduced overfitting by leveraging multiple models and processed errors of individual models to achieve better performance.

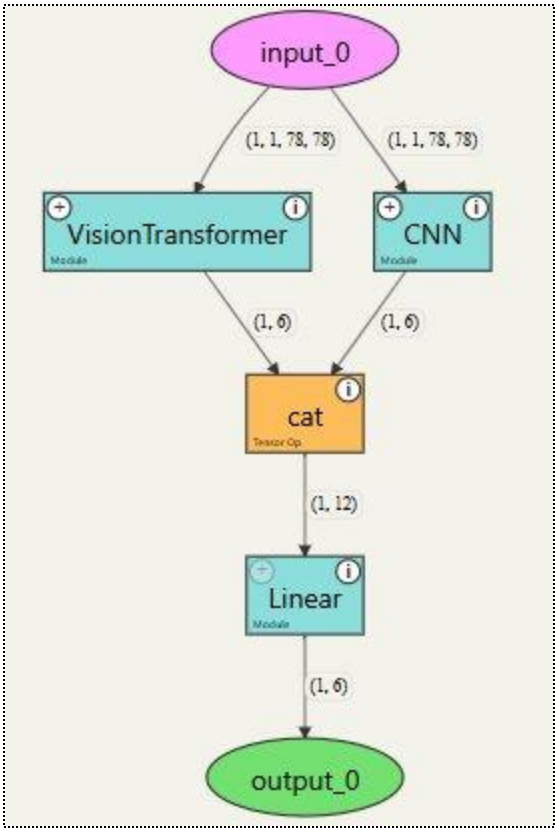


Figure 5. Stack ensemble architecture combining Vision Transformer and CNN outputs through concatenation and a linear meta-layer.

G. Training Configuration

Cross-entropy loss was used as the loss function due to its ability to reduce overfitting by penalizing incorrect classification. Optimization was performed using stochastic gradient descent with momentum and weight decay adjusted per model performance. Early stopping based on validation loss was employed for regularization with a patience of 2 epochs before terminating the train loop. Finally, the train dataset was passed as input to the training loop to train each model.

IV. DATASET

The dataset comprised grayscale images of individual HEp-2 cells depicting six stained patterns from the ICPR 2012 dataset, available via Hugging Face (Genius-Society/HEp2) [1]. The six classes were Centromere, Golgi, Homogeneous, Nuclear Membrane (NuMem), Nucleolar, and

Speckled (Table 1, Figure 6 and Figure 7). The dataset of size 13,593 was divided into train, test, and validation splits in the ratio 70:15:15 such that after preprocessing the dataset, the train dataset consisted of 57102 samples while the test dataset contained 2039 samples, and the validation dataset contained 2040 samples.

Table 1. Dataset size statistics showing class distribution and sample counts.

Class	**Total Samples**	**Test Samples**	**Proportion**
Centromere	2,693	404	19.8%
Golgi	760	114	5.6%
Homogeneous	2,527	379	18.6%
NuMem	2,067	310	15.2%
Nucleolar	2,633	395	19.4%
Speckled	2,913	437	21.4%
Total	13,593	2,039	100%

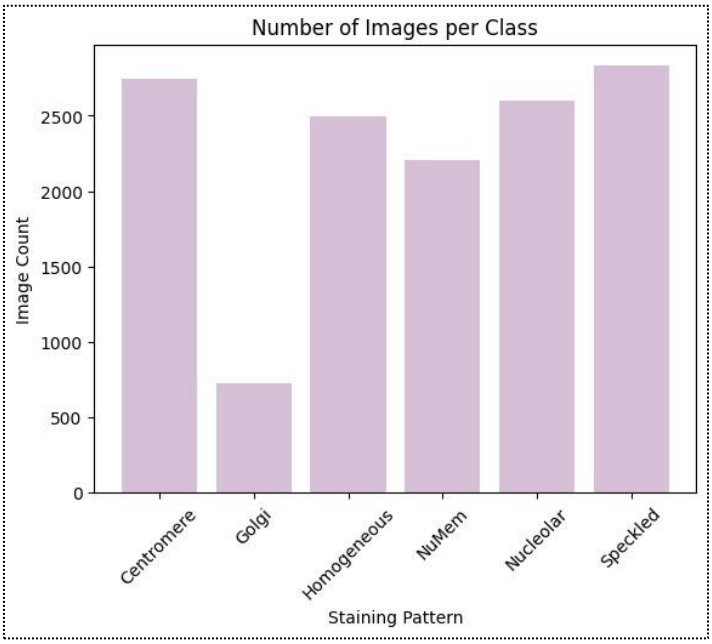


Figure 6. Distribution of images across the six staining pattern classes in the HEp-2 dataset

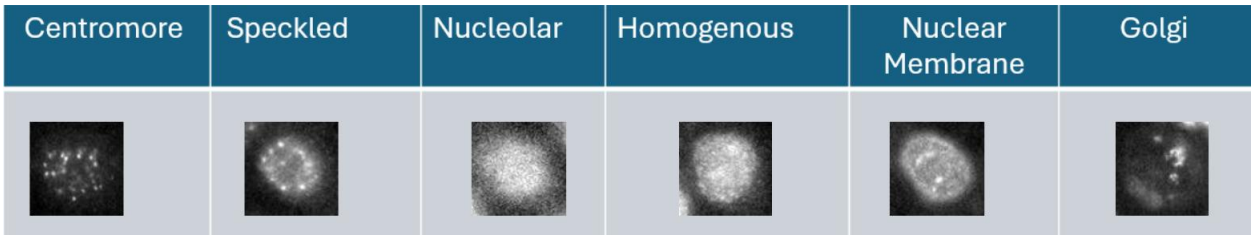


Figure 7. Representative HEp-2 cell images for each staining pattern class

V. RESULTS

The CNN model was trained for a total of 73 epochs which early stopping implemented to observe a difference of 0.001 in validation loss, learning rate of 0.0001 for the stochastic gradient descent (SGD) optimizer with weight decay of 0.001 and slow momentum of 0.4. The ResCNN model was configured in a similar manner with a varied weight decay of 0.005 to prevent the loss curves from diverging and creating a maximized difference between losses. Using these configurations both models achieved an accuracy of 89% as stated in Table 3. When the performance of the 2 models is compared using their respective classification reports as depicted in Table 4 and Table 5, it can be observed that the performance of is better than the ResCNN model. The F1-score of class "Speckled" is better for the CNN model than ResCNN by a 0.1 difference which is further evident upon comparing the recall score for “Speckled” as it is lower for the ResCNN model. Looking at the loss curves of the models (Figure 8), both models were able to generalize the data well and terminate their training right before overfitting occurred.

The ViT model was trained for 20 epochs with delta for early stopping and learning rate at 0.0001. The momentum and weight decay of SGD optimizer were set to 0.4 and 0.001 respectively. Using this configuration, the ViT model achieved an accuracy of 94% as stated in Table 3, representing a 5% improvement over the CNN and ResCNN models. When examining the classification report in Table 6, the ViT model showed improvements across most classes like the Speckled class

improved from an F1-score of 0.82 (CNN) to 0.91 (ViT), and Golgi improved from 0.83 (CNN) to 0.93 (ViT). The model achieved particularly strong performance on Centromere (F1: 0.96) and NuMem (F1: 0.96). However, Golgi recall remained relatively lower at 0.89, likely due to the limited training samples for this class.

The Ensemble model was also trained for 20 epochs but with patience for early stopping configured to 1 epoch. The Stack Ensemble achieved the highest accuracy of 95% among all models. Comparing the classification reports in Table 6 and Table 7, the ensemble showed improved balance between precision and recall scores across classes. For instance, Speckled precision improved significantly from 0.89 (ViT) to 0.95 (Ensemble), while maintaining comparable recall. The NuMem class achieved perfectly balanced metrics with precision and recall both at 0.97. The Centromere class achieved the highest precision of 0.98 among all classes and models.

Looking at the loss curves (Figure 9), both ViT and Ensemble models demonstrated good convergence with validation loss closely tracking training loss, indicating effective generalization. The ViT model showed a steeper initial decline in loss, stabilizing around epoch 10, while the Ensemble model converged more rapidly within approximately 8 epochs due to the pre-trained CNN and ViT components providing strong initial feature representations.

Table 3. Model performance summary.

Model	**Accuracy**	**Macro Precision**	**Macro Recall**	**Macro F1**
CNN	89%	0.89	0.88	0.88
ResCNN	89%	0.89	0.88	0.89
ViT	94%	0.95	0.94	0.94
Stack Ensemble	95%	0.95	0.95	0.95

Table 4. CNN detailed classification results.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Centromere	0.94	0.90	0.92	404
Golgi	0.88	0.80	0.83	114
Homogeneous	0.88	0.87	0.87	379
NuMem	0.92	0.95	0.94	310
Nucleolar	0.90	0.95	0.92	395
Speckled	0.82	0.82	0.82	437
Macro Avg	0.89	0.88	0.88	2039
Weighted Avg	0.89	0.89	0.89	2039

Table 5. ResCNN detailed classification results.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Centromere	0.93	0.91	0.92	404
Golgi	0.89	0.84	0.86	114
Homogeneous	0.85	0.86	0.85	379
NuMem	0.94	0.94	0.94	310
Nucleolar	0.90	0.97	0.93	395
Speckled	0.82	0.79	0.81	437
Macro Avg	0.89	0.88	0.89	2039
Weighted Avg	0.88	0.89	0.88	2039

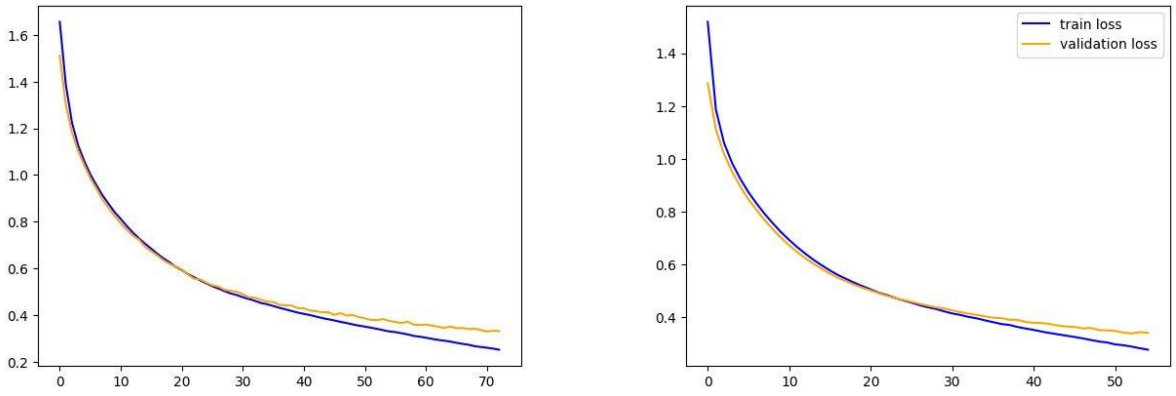
Table 6. Vision Transformer detailed classification results.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Centromere	0.96	0.97	0.96	404

Golgi	0.97	0.89	0.93	114
Homogeneous	0.96	0.93	0.95	379
NuMem	0.96	0.96	0.96	310
Nucleolar	0.93	0.94	0.94	395
Speckled	0.89	0.92	0.91	437
Macro Avg	0.95	0.94	0.94	2039
Weighted Avg	0.94	0.94	0.94	2039

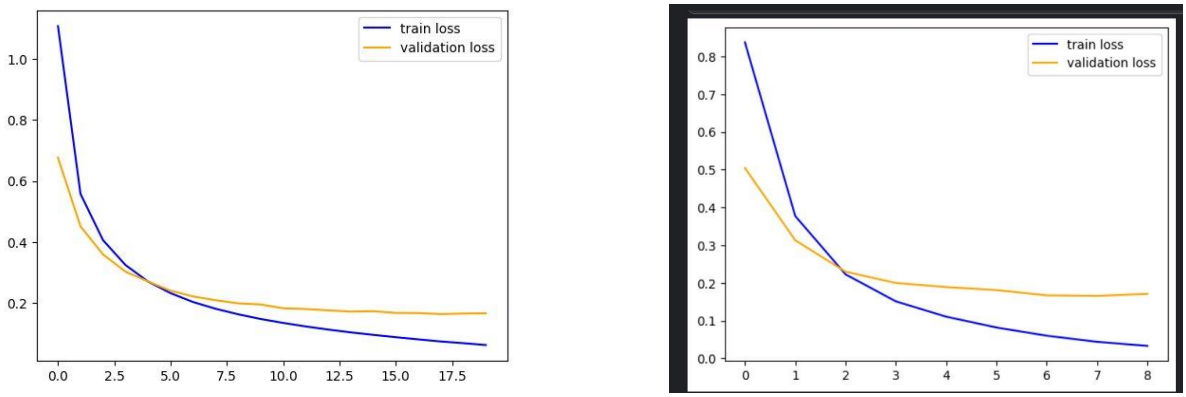
Table 7. Stack Ensemble detailed classification results.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Centromere	0.98	0.96	0.97	404
Golgi	0.92	0.90	0.91	114
Homogeneous	0.96	0.97	0.96	379
NuMem	0.97	0.97	0.97	310
Nucleolar	0.93	0.96	0.95	395
Speckled	0.95	0.93	0.94	437
Macro Avg	0.95	0.95	0.95	2039
Weighted Avg	0.95	0.95	0.95	2039



(A) (B)

Figure 8. Training and validation loss curves for (A) CNN and (B) ResCNN



(A) (B)

Figure 9. Training and validation loss curves for (A) ViT and (B) Stack Ensemble

VI. DISCUSSION

We trained four different models in an attempt to improve generalization and reduce overfitting by applying data augmentation techniques.

A. Scientific Achievements and Findings

CNNs and ViT offered complementary strengths in this classification task. CNNs excelled at capturing local texture and edge features inherent to staining patterns, such as the granularity differences between Speckled subtypes or the discrete dots in Centromere patterns. ViT captured global spatial relationships across the cell, understanding how features were distributed throughout the entire image. The CNN and ResCNN both achieved 89% accuracy, while ViT achieved 94% and the ensemble reached 95%, demonstrating the value of combining architectural approaches. The ensemble approach improved the balance between precision and recall scores across classes. The stack ensemble showed more consistent precision and recall scores compared to individual models. For example, NuMem achieved 0.97 for both precision and recall in the ensemble, compared to 0.92/0.95 in CNN and 0.96/0.96 in ViT. Speckled improved from 0.82/0.82

(precision/recall) in CNN to 0.95/0.93 in the ensemble. This balance is clinically important where both false positives and false negatives have diagnostic consequences.

B. Challenges Faced

The dataset exhibited significant class imbalance, with Golgi (114 test samples, 5.6%) having nearly four times fewer samples than Speckled (437 samples, 21.4%). Limited data availability for Golgi corresponded with rare autoimmune diseases. This imbalance affected model training and required careful metric selection beyond simple accuracy. Certain patterns shared visual characteristics, making discrimination challenging. Fine Speckled patterns could resemble Homogeneous staining at certain intensities, contributing to misclassifications between these classes.

C. Strengths and Shortcomings

The primary strengths of this work included high accuracy achieved across models (89% for CNN/ResCNN, 94% for ViT, 95% for ensemble), robust performance across all six classes with F1-scores ranging from 0.81 to 0.97, and balanced precision-recall metrics in the ensemble important for clinical use. The data augmentation strategy effectively addressed class imbalance during training, and early stopping prevented overfitting while maintaining model performance.

Shortcomings included lower recall for the Golgi class (0.80 in CNN, 0.84 in ResCNN, 0.89 in ViT, 0.90 in ensemble) due to limited training samples and subtle visual patterns. Speckled also showed lower performance in CNN/ResCNN models (F1 of 0.82 and 0.81 respectively). The model required GPU for efficient inference, limiting deployment options, and current augmentation may not fully address extreme class imbalance.

D. Test Set Examples

Centromere patterns with discrete dots were consistently identified correctly across all models (90-97% recall). NuMem patterns showing clear membrane boundaries achieved strong recall across models (94-97%). Nucleolar patterns achieved particularly high recall in ResCNN (97%) and ensemble (96%). Conversely, Golgi patterns with subtle, localized staining showed lower recall across all models (80-90%), often confused with cytoplasmic artifacts. Speckled patterns showed the most improvement from CNN (82% recall) to ensemble (93% recall).

E. Future Directions

In order to counter the class imbalance property of the dataset, applying Synthetic Minority Oversampling Technique (SMOTE) to generate synthetic samples for underrepresented classes like Golgi could further improve classification performance. Additionally, it is possible to provide more specific diagnosis of autoimmune diseases that a patient may possess by combining HEp-2 classification results with analysis from other relevant autoimmune tests and electronic health records to increase the effectiency of the overall diagnostic procedure.

ACKNOWLEDGEMENTS

We thank Professor Amir Tahmasebi Maraghoosh for his guidance and valuable feedback throughout this project. We also thank the teaching assistants, Niranjan Sathish and Craig Roberts, for their support and assistance. Finally, we acknowledge the creators of the HEp-2 dataset for their efforts in gathering this important information in the form of a labelled image dataset and making their data publicly available.

REFERENCES

[1] https://huggingface.co/datasets/Genius-Society/HEp2

[2] Danieli MG, Brunetto S, Gammeri L, Palmeri D, Claudi I, Shoenfeld Y, Gangemi S. Machine learning application in autoimmune diseases: State of art and future prospectives. Autoimmun Rev. 2024 Feb;23(2):103496. doi: 10.1016/j.autrev.2023.103496. Epub 2023 Dec 9. PMID: 38081493.

[3] Huang, CC., Chang, CC. & Chang, CM. Evaluating convolutional neural networks using residual blocks and global average pooling techniques for predicting sediment concentration. *Sci Rep* **15**, 35278 (2025). https://doi.org/10.1038/s41598-025-19161-w

[4] Rukundo, Olivier. 2023. "Effects of Image Size on Deep Learning" Electronics 12, no. 4: 985. https://doi.org/10.3390/electronics12040985

[5] P. H. Kasani, S. H. Kasani, H. W. Kim, K. H. Cho, J. -W. Jang and C. -H. Yun, "HEp-2 Cell Classification Using an Ensemble of Convolutional Neural Networks," 2021 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, Republic of, 2021, pp. 196-200, doi: 10.1109/ICTC52510.2021.9621075. keywords: {Solid modeling;Adaptation models;Transfer learning;Computer architecture;Feature extraction;Convolutional neural networks;Task analysis;Computer-aided diagnosis;Convolutional neural network;Deep learning;HEp-2 cell classification;Transfer learning}

[6] Dosovitskiy, Alexey, et al. “An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale.” ArXiv:2010.11929 [Cs], 22 Oct. 2020, arxiv.org/abs/2010.11929.

[7]https://medium.com/data-science/stochastic-gradient-descent-with-momentum-a84097641a5d

[8] https://towardsdatascience.com/weight-decay-and-its-peculiar-effects-66e0aee3e7b8/

[9] Gao Z, Zhang J, Zhou L, Wang L. 2014. HEp-2 cell image classification with convolutional neural networks. In: 2014 1st Workshop on Pattern Recognition Techniques for Indirect Immunofluorescence Images; Stockholm, Sweden. IEEE. p. 24-28.

[10] https://www.bates.edu/biology/files/2010/06/How-to-Write-Guide-v10-2014.pdf

APPENDIX

Instructions to run demo.py

1. View the dataset at https://huggingface.co/datasets/Genius-Society/HEp2 to understand the staining patterns.
2. Navigate to the DEMO folder and open the notebook demo.py
3. Install all required packages and libraries using the command pip install requirements.txt
4. Download the model weights from the provided links below and replace the variables listed in cell number one with corresponding paths of the model weights correspond to their location on the device. If not, replace them with the correct directory path.
    a. ViT: https://www.kaggle.com/models/case143/vit-best/
    b. CNN: https://www.kaggle.com/models/case143/cnn/
    c. ResCNN: https://www.kaggle.com/models/case143/rescnn/
    d. Ensemble: https://www.kaggle.com/models/case143/ensemble-fai/
5. Run all cells before the heading "Pre-defined inputs" for initializing required functions, models, and obtaining an array label_samples containing 6 subsets. Here, the array index i in label_samples corresponds to the encoded label ID of a class and stores all records pertaining to that particular class as a dataset.
6. To test data using predefined inputs, which contain 5 images belonging to each class, run the cell under the heading 'Pre-defined inputs'.
7. To test using user-defined inputs, create a dictionary using samples from label_samples to create a new test dataset in the format {"image": [], "label": []}. To view images to be

included in the input, call function view_image(). Finally the function defined_inputs(data) to create a data loader for the new test dataset.

8. Run cells under the heading 'Test Data' to test the performance of all models on the test data. To test performance on specific models, alter the models parameter of the function classify()