# Gastrointestinal disease detection using ResNet on Hyper-kvasir dataset


Manthan Patel
Btech Computer Science
(Ahmedabad University)
manthan.p@ahduni.edu.in

Meet Patel
Btech Computer Science
(Ahmedabad University)
meet.p6@ahduni.edu.in

Dev Patel
Btech Computer Science
(Ahmedabad University)
dev.p1@ahduni.edu.in

Sagar Bajaj
Btech Computer Science
(Ahmedabad University)
sagar.b2@ahduni.edu.in



***Abstract*— Gastrointestinal disease detection is an important task in medical image analysis, and the Hyper-Kvasir dataset is a widely-used benchmark dataset for this task. ResNet (Residual Network) is a deep learning architecture that has achieved state-of-the-art performance on many image classification tasks. This report presents an implementation of ResNet50, a deep learning model, for the detection of gastrointestinal diseases using the hyper-kvasir dataset. The dataset contains 8,000 high-resolution images of the gastrointestinal tract, labeled with 23 different classes, and we have selected six classes for this project. We performed data preprocessing, including resizing images and data augmentation, followed by fine-tuning the pre-trained ResNet50 model and training the model using the Adam optimizer. The model's performance was evaluated on the testing set, and the results showed an accuracy of around 92% after five epochs. Overall, the implementation of ResNet50 on the hyper-kvasir dataset shows promising results in detecting gastrointestinal diseases, and further improvements can be made by optimizing the hyperparameters and experimenting with other deep-learning architectures.**



***Keywords*— *Gastrointestinal diseases, Hyper-kvasir, deep learning, Residual Network (ResNet), fine-tuning.***


## I. Introduction

A collection of illnesses known as gastrointestinal (GI) diseases include those that affect the esophagus, stomach, small and large intestines, liver, gallbladder, and pancreas. Effective treatment and management of many disorders depend on an early and precise diagnosis. Deep learning-based methodologies have recently demonstrated promising outcomes in the identification and categorization of a variety of diseases. The deep learning model ResNet has been applied to a variety of computer vision tasks, including the processing of medical images. The Hyper-kvasir dataset may be used to train a ResNet model, which can then be used to correctly classify GI illnesses from endoscopic pictures. Better patient outcomes may result from increasing the effectiveness and precision of diagnosis and treatment. Deep neural networks suffer from the problem of vanishing gradients making it more difficult to train. As the network depth increases, the gradient multiplication during the backpropagation results in extremely small gradients. This results in stagnated or degraded performance by the network. ResNet is a deep convolution neural network architecture that proposed a residual learning framework for solving the degradation problem. The ResNet-50 architecture comprises 48 convolutional layers, along with 1 layer each of max pooling and average pooling.

## II. Literature Survey

Deeper neural network stack multiple layers which inculcate the problem of vanishing gradients during the process of backpropagation. When the gradients backpropagate using the chain rule, the repetitive multiplication of weights leads to extremely small weights resulting in stagnation or degradation of the performance of the neural network. Residual networks put forward the framework of skip connections through which the problem of degradation can be mitigated. The skip connection helps the deeper layer to learn an identity function that outputs the same result as the shallow layer and hence stacking multiple layers in between the connection does not affect the output. The original ResNet architecture that was proposed was ResNet-34 comprising 34 layers. The ResNet-50 was based on the original architecture with 50 layers that implemented the bottleneck design. The bottleneck block uses 1*1 convolutions which helps in reducing the number of parameters and matrix multiplication. This enables faster training associated with each layer. The ResNet-50 uses a stack of three layers and is divided into 4 main segments. The convolutional layer has 64 kernels of size 7 by 7 and a stride of 2, followed by a max pooling layer with a stride of 2. The internal architecture comprises 48 layers with residual connections. Each convolution layer is followed by a batch normalization layer and a ReLU(rectified linear unit) activation function. After the convolutional and max pooling layers, the resulting output is fed into a fully connected layer with 1024 neurons that uses the SoftMax activation function, and is convolved with an average pooling layer.

## III. Implementation

The dataset used is the Hyper-kvasir dataset. The hyper-kvasir dataset contains 8,000 high-resolution images of the gastrointestinal tract, with each image labeled according to the type of disease present. There are two types of images. First is anatomical landmarks which represent a normal condition and are used for positioning during endoscopic examination. Other is pathological findings, which represent abnormality in the gastrointestinal tract. There are a total of 23 different classes in the dataset. We have selected only 6 classes with more images: Z-line, Pylorus, Esophagitis,

Cecum, Polyps, and Ulcerative-colitis. Three of them are anatomical landmarks, and three are pathological findings.
Disease prediction using the hyper-kvasir dataset involves the use of convolutional neural networks (CNN) to predict different gastrointestinal diseases. In this project, we will use the ResNet50 architecture, a deep-learning model that has proven effective in image classification tasks. To implement this project, we follow the following steps:
Data preprocessing: This step involves loading the hyper-kvasir dataset, resizing the images to a standard size, and dividing the dataset into training and testing sets. First, we split the dataset into training, validation, and test sets with 60% train set, 20% validation set, and 20% test set using the "split_folders" library. We resized all the images to the dimension (224,224) because the ResNet50 model requires images to be in that dimensions. To prevent overfitting and increase the size of the dataset, we utilized data augmentation techniques such as random rotation, flipping, and zooming. These image augmentation operations were implemented using the Image Data Generator module, which is available in the Keras library. We performed the following data augmentation:

- shear_range = 0.2
- zoom_range = 0.2
- horizontal_flip = True
- batch_size = 32 (for training and validation) and 1 (for testing)

Model architecture: We will use the ResNet50 architecture, which is a pre-trained model available in the Keras library. This architecture is composed of 50 layers and has skip connections, which help avoid the vanishing gradient problem and improve the model's accuracy.
Fine-tuning: In this step, we fine-tune the ResNet50 architecture by retraining the last few layers of the model. We freeze the initial layers of the model and only train the final layers to adapt the model to our specific classification task. Using the Keras library, we used the pre-trained ResNet50 model trained on the ImageNet dataset. We used the GlobalAveragePooling2D() method, which takes a 4D tensor as input and returns a 2D tensor with the same number of channels (depth) as the input tensor, but with spatial dimensions collapsed to a single dimension. To the pre-trained ResNet50 model, we appended a fully connected layer with 1024 neurons that employs the ReLU activation function. Additionally, we defined an output layer consisting of 6 classes with the SoftMax activation function.
Training: We train the model using the training set, which involves feeding the preprocessed images to the ResNet50 architecture and updating the model's weights using backpropagation. We used categorical cross-entropy as the loss function and the Adam optimizer to update the model's weights.
Model evaluation: We evaluated the trained model using the testing set, which involves feeding the preprocessed images to the model and predicting the disease type for each image.

## IV. Results

The model was trained for 5 epochs, and its results are as follows:

- In the first epoch, the model had a loss of 0.8407 and an accuracy of 0.7649.
- In the second epoch, the model's loss decreased significantly to 0.2498, and its accuracy improved to 0.9098.
- In the third epoch, the model's loss decreased slightly to 0.2183, and its accuracy remained high at 0.9093.
- In the fourth epoch, the model's loss continued to decrease to 0.1959, and its accuracy improved further to 0.9210.
- In the fifth and final epoch, the model's loss increased slightly to 0.2107, but its accuracy remained unchanged at 0.9210.

## V. Conclusions

The model improved after the first epoch, as the loss decreased and accuracy increased significantly. However, the improvement was not consistent, as the loss and accuracy fluctuated slightly in subsequent epochs. Overall, the model seems to have performed reasonably well, with an accuracy of around 92%.

## References


[1] S. (n.d.). *GitHub - simula/hyper-kvasir: GitHub repository for the Hyper-Kvasir dataset.* GitHub. https://github.com/simula/hyper-kvasir

[2] Borgli, H., Thambawita, V., Smedsrud, P. H., Hicks, S. A., Kwon, G., Eskeland, S. L., Randel, K. R., Pogorelov, K., Lux, M., Nguyen, D. K., Johansen, D., Griwodz, C., Stensland, H. K., Garcia-Ceja, E., Schmidt, P., Hammer, H. L., Riegler, M., Halvorsen, P., & De Lange, T. (2020). HyperKvasir, a comprehensive multi-class image and video dataset for gastrointestinal endoscopy. *Scientific Data*, *7*(1). https://doi.org/10.1038/s41597-020-00622-y

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *ArXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1512.03385

[4] Dwivedi, P. (2021, December 7). *Understanding and Coding a ResNet in Keras - Towards Data Science*. Medium. https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33

[5] Bengio, Y., Simard, P. Y., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166. https://doi.org/10.1109/72.279181