CS361 Machine Learning Course Project
Background Removal using ML

Team Amber
Abhishek Kumar (220101002)
Adarsh Gupta (220101003)
Vasudha Meena (220101108)

Problem Statement & Introduction

The goal of this course project is to experiment with Background Removal techniques that use Machine Learning algorithms (i.e. without Neural Networks) and give a comparative performance with deep learning-based algorithms.
Note: This document is majorly focused on the motivation behind why we did what we did and how our learnings impacted our choices. For a more detailed discussion of the strategies, you can refer to the attached link at the end. (A short note on the debugging hell that GIMP was is also included).
Yes, the maximum limit is 2 pages, but we tried a lot and there was no way this could have been included in just 2 pages.

Why is this problem hard for ML-based algorithms?

The main idea behind common binary image segmentation (or image segmentation in general) algorithms that have good performance is a block called a Convolutional Block. The convolutional block allows the model to capture short and long-range dependencies. This also allows it to capture tiny details like edges, textures, and corners, which allows it to easily reconstruct the image and not lose too many details.
We don't have that sort of luxury in ML town. ML-based algorithms can at best learn to approximate a function or learn a separating boundary between data points. Thus we need some way of incorporating those advanced features and had to model the problem of Binary Image Segmentation into a problem of pixel-level Classification into foreground or background to make it tractable for ML algorithms.

General overview of ML strategies we used

The problem of Background Removal was modeled as pixel-level classification into foreground or background. For this classification task, we first tried using on-the-fly model training as a form of unsupervised learning, with some supervision provided by user-defined object bounding boxes. We then tried using a supervised learning approach followed by a hierarchical supervised learning approach. Finally to compare the performance we trained a U-Net model on the same dataset and also performed analysis using a large image-to-image model. Now we will discuss each strategy in detail along with the motivation behind every strategy.

Strategy 1: User-guided on-the-fly model training

The main idea behind using this model is that a model with less expressive power (say Logistic Regression) to reduce its loss as much as possible would just learn to predict the majority of the positive class in the on-the-fly created dataset as positive and would be unable to learn to predict the less prominent pixels, marked by the user as false positives in the training dataset, to belong to the positive class. This is fueled by the fact that the pixel-wise features used in this strategy were kept simple to reinforce this more. This hypothesis is evident in our experimentation, the simple Logistic Regression model learned that anything colored red needs to be predicted in the positive class and it did not have enough motive or expressive power to learn to predict the orange part and the black part (the black part is false positive to a human eye) as belonging to the positive class.

This strategy does work, in very rare cases, when there is a clear separation between the actual foreground and background in the user-supplied bounding boxes. This, very ideal case, is not generally true, and thus this model, even though intuitively sound, is much more suited as an intuitive idea rather than an actual application for the task at hand.

This experiment however gives us some possible ways in which we can improve towards this task. The first inference is that we need to restrict our domain to a narrow domain and model this as a supervised learning task, and the second inference is that we need to extract more meaningful features rather than just local patterns and histogram gradients (as done here). These two things are handled for the next 2 proposed ML-based strategies, and a detailed discussion on how we incorporated these strategies is provided in Sidesnote 1 and Sidenote 2.

Strategy 2: SingleScale Supervised Random Forest

The SingleScale and Hierarchical MultiScale strategies both have the same starting point. We need to extract features for each pixel. Which features? It was described in Sidenote 2 (see extended report). SingleScale strategy only extracts features from the dataset at the original scale. In the MultiScale strategy, the features are extracted for every pixel, in every image, at every scale.

In the SingleScale Supervised Random Forest Strategy, we simply trained a Random Forest model on the original scale dataset to predict, pixel-wise, whether the pixel belongs to the foreground or the background. This strategy is pretty sound and does perform well on a narrow domain dataset, as is our case, however, it is not scalable. As the number of estimators are bounded there is an upper bound on the amount of the knowledge that the Random Forest model can store in its trees. To get around this bottleneck and to make this strategy a bit more generalized (I say generalized in the sense that it can accommodate many more samples of the same domain, not generalized across domains), we proposed to use a Hierarchical strategy, where we first downsample the image, predict each pixel in the downsampled image as foreground or background, and use this prediction at lower resolutions to guide the predictions at the higher resolutions. This is precisely the motivation behind the final ML strategy: Hierarchical MultiScale Supervised Random Forests.

Strategy 3: Hierarchical MultiScale Supervised Random Forests (HMS-SRF)

The final ML-based strategy that we proposed for the background removal, aka binary image segmentation problem, is called Hierarchical MultiScale Supervised Random Forests, aka HMS-SRF. The complete 5-phase strategy has been explained in the figure below. The motivation and advantages of using this strategy have been covered in the last section. The 3 step inference technique is also clear from the training methodology. Does using this hierarchical strategy work? YES! In the final trained RandomForestClassifier3 model, we analyzed feature importance, and fifty_pred and twenty_five pred were found to be the 2 most important features!

Some conclusions from comparative analysis

In the transition from traditional supervised learning to deep learning approaches for background removal we got a few points like
1. The traditional algorithms rely on manually engineered features that may inadvertently focus on irrelevant image properties (position, isolated color values), deep learning models autonomously discover hierarchical feature representations directly relevant to the segmentation task.
2. Non-DL approaches often process pixels in isolation or with limited neighborhood context, whereas CNN-based architectures like U-Net inherently incorporate spatial relationships through expanding receptive fields and multi-scale feature analysis.

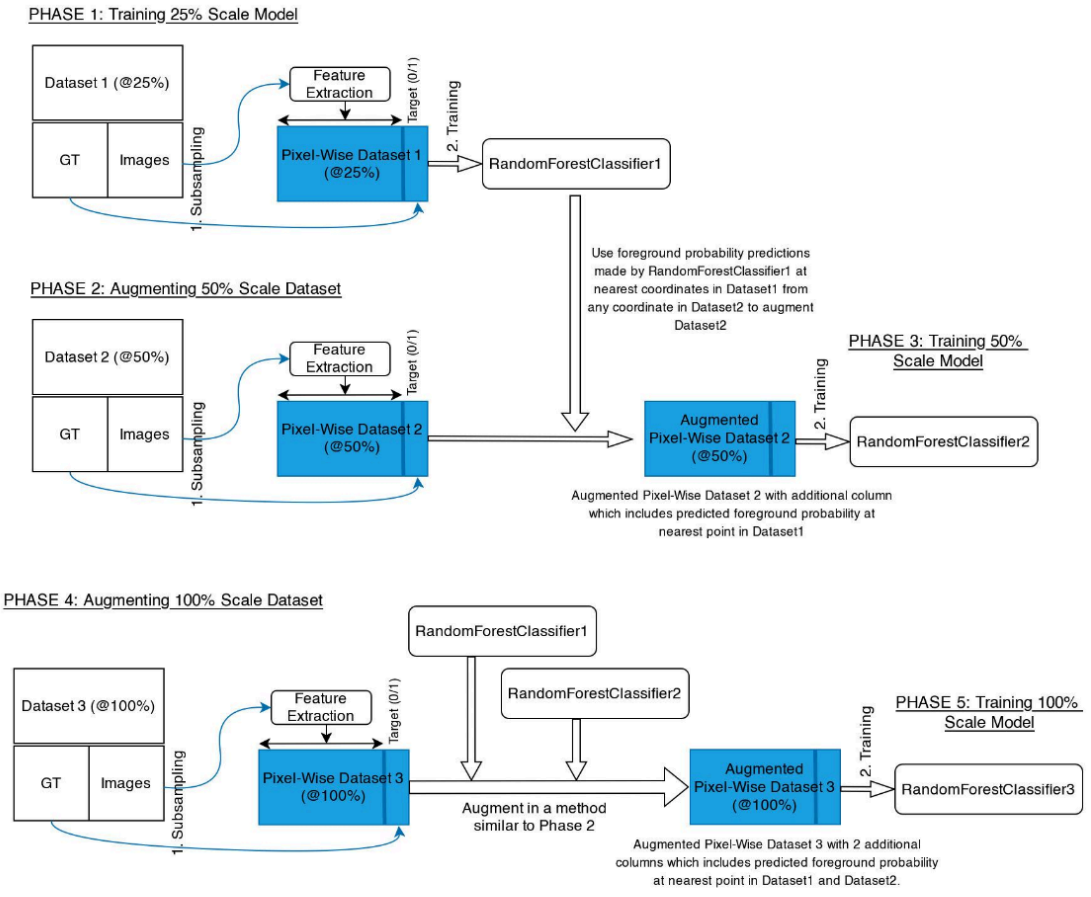
3. The CNN-based model can learn complex, multi-scale features directly from data which overcomes the inherent limitations of traditional approaches where feature extraction quality directly constraints segmentation performance
4. CNN-based algorithm: U-Net outperformed our ML strategies, but not by a huge margin for our narrow domain task.
5. We also experimented with a large-scale image-to-image model: DALL-E but we noticed that it was creating details that were not there in the original image.

Sidenote: Deployment in GIMP as a Plugin

The created script was also deployed in GIMP3.0.2 as a plugin. We have provided the final plugin script in the Kaggle Dataset but setting it up to run is a hassle, if you do proceed with trying to run and you inadvertently run into import errors because GIMP dies when it tries to import NumPy, you can use the following Links for debugging. (Link 1, 2). Debugging this was a process that took days and over 3 computers with different OS.

Links

Strategy 1: GitHub Link: https://github.com/CoolSunflower/LR-Background-Removal Deployment: Link
Strategy 2 & 3: The complete code has been uploaded as Kaggle Dataset (LFS restriction on GitHub) Link: https://www.kaggle.com/datasets/weirdanalyst/cs361-background-removal-with-ml-team-amber
Strategy 4: Kaggle Notebook: https://www.kaggle.com/code/abhishekumar1819/cs361
A more detailed report: Team Amber CS361 Report



Strategy 3: Hierarchical MultiScale Supervised Random Forests (HMS-SRF)