GrabCut

Interactive Foreground Extraction using Iterated Graph Cuts

Anirudh Kaushik (2020111), Aditya Harikrish (2020111009),
Pratyay Suvarnapathaki (2020111016), Yash Mehan (2020111020)

International Institute of Information Technology, Hyderabad

---

1. Introduction

1.1. Image Segmentation - An Overview

Image segmentation is a key component in computer vision, and it is the process of dividing an image into multiple segments to simplify and/or change the representation of an image. This process can be used to help identify objects and boundaries in an image. Image segmentation involves identifying and separating objects or parts of an image into distinct regions.

The process of image segmentation can be done in a variety of ways. One approach is to use edge detection algorithms, which work by looking for abrupt changes in intensity across an image. This allows for the identification of objects and their boundaries. Another approach is to use clustering algorithms, which group similar pixels into clusters. This can be used to identify objects in an image and separate them from the background. Finally, image segmentation can also be done using deep learning algorithms. Deep learning algorithms are able to identify complex patterns in an image and can accurately segment objects. These algorithms are more accurate than traditional methods and can be used to identify objects and their boundaries.

1.2. Problem Statement and Objective

The objective of the current project is to implement the GrabCut algorithm, as proposed by Rother et al. [1]. GrabCut has become one of the most popular image segmentation algorithms. It is a semi-supervised algorithm and uses a combination of graph-cut optimization and user input to segment an image. The algorithm works by first creating a boundary box around the object of interest. This boundary box is then refined by taking into account user input. The user input is used to mark which pixels belong to the object and which pixels should be excluded from the segmentation.

We implement and present the following:

1. Define constraints based on the bounding box specified by the user.
2. Run the iterative GrabCut procedure based additionally on user interaction ('brushes' to specify foreground/background elements interactively).
3. Showcase results obtained upon running the procedure, also depicting variations in results upon tweaking the GrabCut hyperparameters.
4. Explore applications of GrabCut in medical image segmentation and experiment with saliency maps as an automated alternative to user interaction.

2. Problem Formulation

2.1. Definitions

1. **Hard Labelling:** A hard label is a label that won't change in the process. This is represented as gc.BG and gc.FG in the code and corresponds to the for sure background and foreground pixels marked by the user brushstrokes in the interactive portion.
2. **Gaussian mixture model:** A gaussian mixture model is a probabilistic model that assumes all data points are generalised from a mixture of a finite number of gaussian distributions with unknown parameters.
3. **EM expectation:** It is an algorithm we use for fitting the Gaussian Mixtures.
4. **Alpha channel:** corresponds to the opacity value of each pixel in the image.
5. **Trimap:** is a rough segmentation that divides an image into Background, Foreground and a mixture of both called the transition region (unknown region, could be BG or FG). It is represented by $\{T_B, T_U, T_F\}$
6. **Histogram of a grayscale image:** the histograms represent the frequency of occurence of each pixel in the image, it is a frequency vs intensity plot.

2.2. Previous Approaches for Interactive Segmentation

At the time of publication of our reference paper [1], the following segmentation techniques were considered state of the art. We briefly examine these to set the broad context required to appreciate what GrabCut achieves.

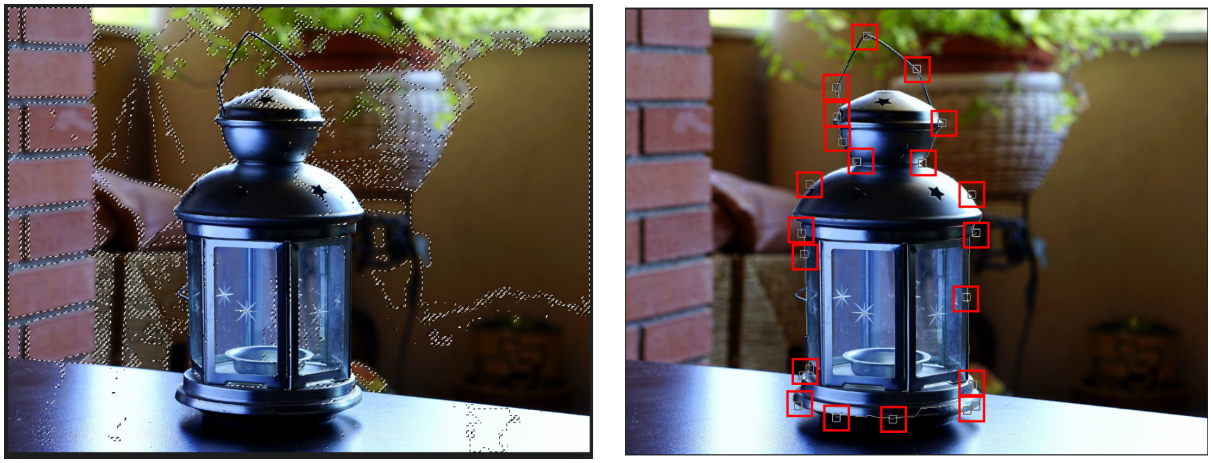

Figure 1. A comparison of the Magic Wand and MagneticLasso tools in popular image editing software PhotoPea. Seed points manually annotated for the latter, for the sake of visibility.

1. The **Magic Wand** approach is straightforward and intuitive, with the user specifying a certain point, after which a region of connected pixels such that all the selected pixels fall within some adjustable 'tolerance' of the specified point's colour is segmented. Its main limitation is that the tolerance aspect makes results vary unpredictably.
2. The **Intelligent Scissors** or **Magnetic Lasso** tool involves the user sketching a primitive boundary around the object to be segmented. Subsequently, the algorithm treats this user input as seed points, and computes a minimum-cost 'optimal boundary'. The downside in this case is the extent of user participation required to obtain the desired results.

3. **Bayesian Matting** models colour distributions probabilistically to achieve full alpha mattes. The user specifies a trimap $T = T_B, T_U, T_F$ in which background and foreground regions TB and TF are marked, and alpha values are computed over the remaining region TU. A considerable degree of user interaction is required to construct an internal and an external path.

4. Lastly, **GraphCut**, as pioneered by Boykov et al [2] is a powerful optimisation technique that can be used in the same trimap setting as Baye's Matting to achieve robust segmentation. **Since GrabCut is an extension of this technique, we explore it in detail below**

2.3. The GraphCut Algorithm

Take an image and represent it as the vector of N pixels $\vec{z} = (z_1, z_2, ..., z_N)$. Let the segmentation be represented by the vector of opacity values $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_N)$. Here, if the constraints are $0 \leq \alpha_i \leq 1$, then it is referred to as a "soft" segmentation, while $\alpha_i \in \{0, 1\}$ is referred to as a hard segmentation. In hard segmentation, $\alpha_i = 0$ and $\alpha_i = 1$ indicate that z_i is in the background and the foreground respectively. We define $\theta = \{h(z; \alpha), \alpha = 0, 1\}$ as the set of 2 histograms, one each for the grey-level distributions of foreground and background pixels. The histograms are parameterized by α where a value of 0 corresponds to the histogram for the background pixels and a value of 1 corresponds to the histogram of the foreground pixels. Normalized histograms ($\int_z h(z; \alpha) = 1$) are assembled from the trimap using the labelled points $\{T_B, T_F\}$.

Now, segmentation via energy minimisation involves defining an energy function $\mathbb{E}$ such that it's minima corresponds to a good segmentation.

The requirements herein are:

1. Must be guided by both, the background and foreground histograms.
2. Opacity must be "coherent", i.e, objects must appear solid (less transparent).

The energy function is defined as follows.

$$\mathbb{E}(\vec{\alpha}, \vec{\theta}, \vec{z}) = \mathbb{U}(\vec{\alpha}, \vec{\theta}, \vec{z}) + \mathbb{V}(\vec{\alpha}, \vec{z})$$

In the above equation, $\mathbb{U}$ evaluates fit of $\vec{\alpha}$ (opacity distribution) to data $\vec{z}$ given histogram $\vec{\theta}$.

$$\mathbb{U}(\vec{\alpha}, \vec{\theta}, \vec{F}) = \sum_n -log h(z_n, \alpha_n)$$

Here, $\theta_n = h(z_n, \alpha_n)$ where the subscript n denotes a specific pixel. $\mathbb{V}$ is referred to as the smoothness term.

$$\mathbb{V}(\vec{\alpha}, \vec{z}) = \gamma \sum_{(m,n) \in \mathbb{C}} \text{dis}(m, n)^{-1} [\alpha_n \neq \alpha_m] e^{-\beta * (z_m - z_n)^2}$$

Here, $[\phi]$ denotes the indicator function of the predicate ϕ, $\mathbb{C}$ is the set of pair of neighbouring pixels, and dis is the Euclidean distance function of neighbouring pixels.

This energy encourages coherence in regions of similar gray level distribution. β is a constant and was chosen to be

$$\beta = (2 < (z_m - z_n)^2 >)^{-1}$$

$< ., . >$ denotes expectation over an image sample.
γ is a hyperparameter whose best value was found to be 50 as per the paper.

$$\vec{\hat{\alpha}} = arg \min_{\vec{\alpha}} E(\vec{\alpha}, \vec{\theta})$$

The output of the above equation will give us the value of α for such that $E(\vec{\alpha}, \vec{\theta})$ will be minimum.
yo

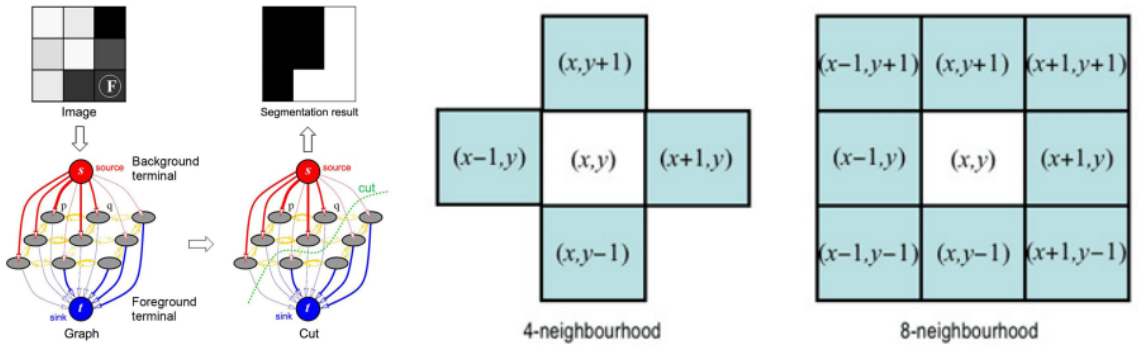

Figure 2. Left - Representing an image as a graph ; Right - Pixel Neighbourhoods

3. GrabCut

3.1. Extension of GraphCut to GrabCut

The above equations are for monochrome images. In GrabCut we do segmentation via energy minimisation for a colored image. The main difference in the approach involves using GMM (Gaussian Mixture Models) to model the pixel distributions instead of a gray value histogram.

Another important feature of this algorithm that set it apart from other segmentation algorithms at the time was that it minimised the effort a user had to put in. To obtain a good segmentation, a user had to provide only the T_B component of the trimap corresponding to the area in the image that was surely in the background, which was provided in the form of drawing a bounding box around the object.

3.2. Color Data Modelling

We now redefine z_n as a pixel in RGB space and the corresponding image as a vector $\vec{z} = (z_1, ..., z_N)$ in the RGB space.

We construct 2 GMMs, one for Background pixels and one for foreground pixels, which are full-covariance gaussian mixtures with K components. Here K is a hyperparameter and is taken to be 5.

$$\vec{k} = k_1, ..., k_N$$
$$k_n \in 1...K$$

As per the above equation, for each pixel a unique GMM component is assigned. One component is for BG and one is for FG corresponding to $\alpha = 0, 1$ respectively.

Now Energy Function becomes

$$E(\vec{\alpha}, \vec{k}, \vec{\theta}, \vec{z}) = U(\vec{\alpha}, \vec{k}, \vec{\theta}, \vec{z}) + V(\vec{\alpha}, \vec{})$$
$$U(\vec{\alpha}, \vec{k}, \vec{\theta}, \vec{z}) = \sum_n D(\alpha_n, k_n, \vec{\theta}, z_n)$$

Here

$$D(\alpha_n, k_n, \vec{\theta}, z_n) = -logp(z_n|\alpha_n, k_n, \vec{\theta}, z_n) - log\pi(\alpha_n, k_n)$$

Where, $p(.) \rightarrow$ gaussian probability distribution $\pi(.) \rightarrow$ mixture weight coefficients

$$D(\alpha_n, k_n, \vec{\theta}, z_n) = -log\pi(\alpha_n, k_n) + \frac{1}{2}[z_n - \mu(\alpha_n, k_n)]^T \Sigma(\alpha_n, k_n)^- 1[z_n - \mu(\alpha_n, k_n)]$$

3.3. Model Params

$$\vec{\theta} = \{\pi(\alpha, k), \mu(\alpha, k), \sum(\alpha, k), \alpha = 0, 1, k = 1...K\}$$

These correspond to the mixture weights, mean for the K Gaussian mixture models and Covariances for the K GMMs for background ($\alpha = 0$) and foreground ($\alpha = 1$)

$$V(\vec{\alpha}, \vec{z}) = \gamma \sum_{(m,n) \in C} [\alpha_m \neq \alpha_n] e^{-\beta ||z_m - z_n||_2^2}$$

3.4. Proof Of Convergence

We now prove that the above equations converge resulting in a minima for the energy function.

At each step we minimize the total energy E w.r.t $\vec{k}$, $\vec{\theta}$, $\vec{\alpha}$ and thus E decreases monotonically. Thus the algorithm is guaranteed to converge at least to a local minimum of E. It is straightforward to detect when E ceases to decrease significantly, and to terminate iteration automatically.

3.5. Psuedocode

3.5.1. Initialisation

- User initialises trimap T by supplying only T_B. The foreground is set to $T_F = \emptyset; T_U = \bar{T}_B$, complement of the background.
- Initialise $\alpha_n = 0$ for $n \in T_B$ and $\alpha_n = 1$ for $n \in T_U$.
- Background and foreground GMMs initialised from sets $\alpha_n = 0$ and $\alpha_n = 1$ respectively.

3.5.2. Iterative minimisation

1. Assign GMM components to pixels:for each n in T_U,

$$k_n := \arg\min_{k_n} D_n(\alpha_n, k_n, \theta, z_n).$$

2. Learn GMM parameters from data z:

$$\underline{\theta} := \arg\min_{\underline{\theta}} U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z})$$

3. Estimate segmentation: use min cut to solve:

$$\min_{\{\alpha_n : n \in T_U\}} \min_{\mathrm{k}} \mathrm{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}).$$

4. Repeat from step 1 , until convergence.
5. Apply border matting (section 4 - not implemented)

3.5.3. User editing

- Edit: fix some pixels either to $\alpha_n = 0$ (background brush) or $\alpha_n = 1$ (foreground brush); update trimap T accordingly. Perform step 3 above, just once.
- Refine operation: [optional] perform entire iterative minimisation algorithm.

4. Results and Analysis

4.1. Basic Procedure

Figure 3 showcases the overall GrabCut workflow for three sets of images. The initial bounding box specification and its corresponding output has been depicted, followed by user interaction (foreground/background marking) and the final GrabCut output.

The default values of the hyperparameters for the GrabCut function were chosen to be:

- $\gamma = 50$
- $k = 5$
- Number of iterations $= 1$

Thus, we demonstrate how GrabCut results are improved based on user interaction.

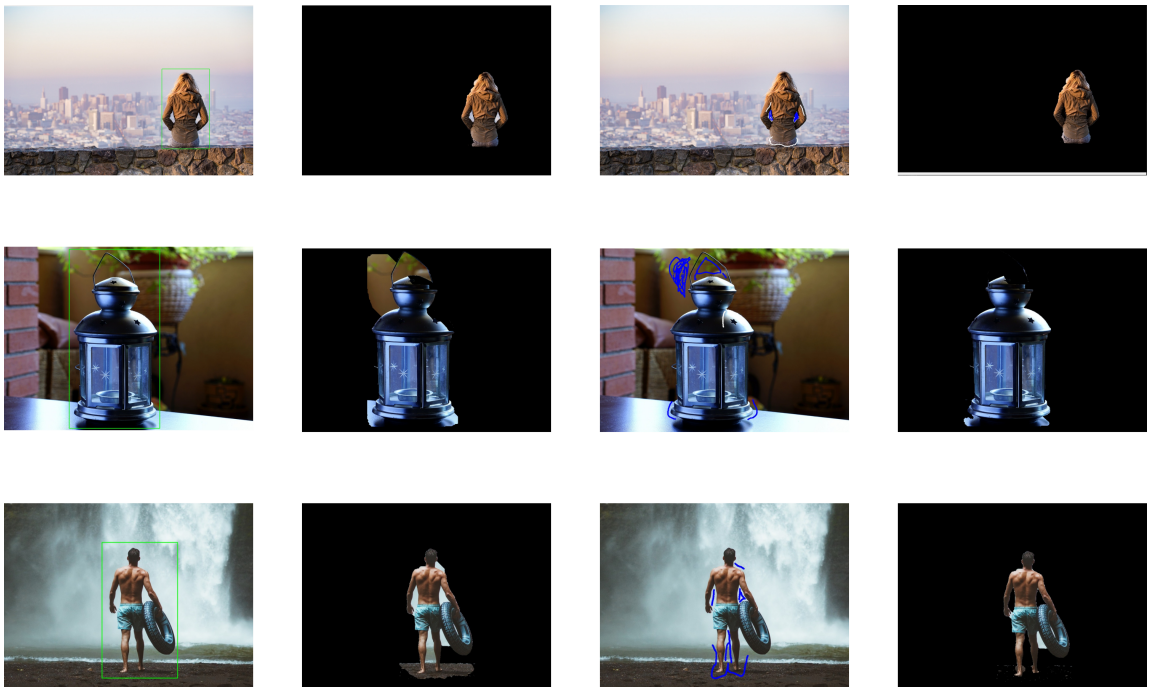

Figure 3. Results at different stages in the GrabCut pipeline:
(i)Bounding box - - - - (ii)Intermediate output - - - - (iii)User interaction - - - - (iv)Final output

4.2. Variation in Bounding Box Margins

Figure 4 showcases the differences in performance depending on how tight the margins of the bounding box are. The results indicate that the quality of segmentation is heavily dependent on this aspect.

4.3. Variation in Gamma and Number of GMM Components

Figure 5 indicates that results are imporved on increasing the gama value. On the other hand, segmentation quality does not visibly improve upon increasing the number of GMM components, i.e. the k value.

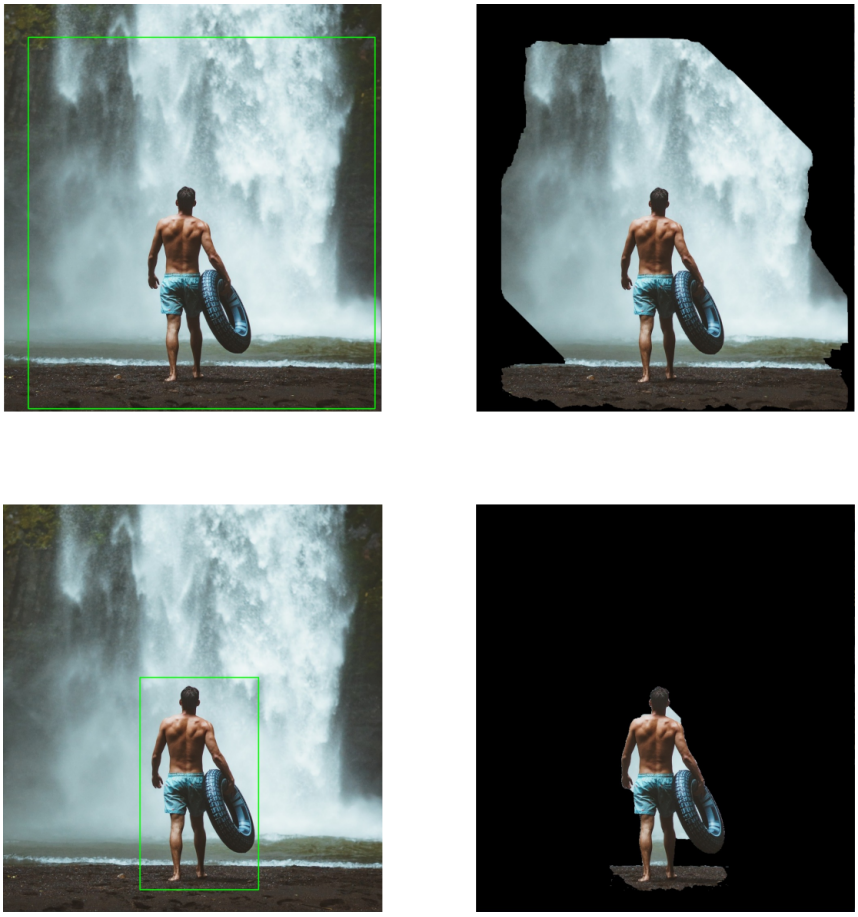

Figure 4. Intermediate outputs for loose and tight bounding boxes

Figure 5. Intermediate outputs for $\gamma = 5, 30, 100$ (top row) and $k = 1, 3, 5$ (bottom row)

4.4. Variation in the Number of Iterations

As indicated by figure 6, the performance visibly increases on increasing the number of iterations from 1 to 3, but a very marginal improvement results upon going from 3 to 5 iterations. This suggests that tweaking all the other aforementioned parameters is more impactful towards

improving segmentation quality than the sheer number of iterations.

Figure 6. Intermediate outputs for iters = 1, 3, 5 for the waterfall and lantern images

5. Applications and Experimentation

5.1. Application on medical Images

We tested the grabcut algorithm for another type of images, namely medical images. The model was tested on one frame of a video sequence of doppler effect being produced in nerves by passing ultrasonic waves. The colors correspond to the velocities of the particles. The doppler was used to

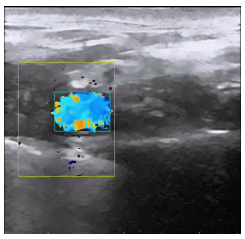

Figure 7. Original doppler

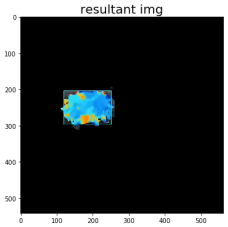

Figure 8. Segmented output

mark the region of interest which was pulsating and GrabCut provided a good segmentation of the ROI. Similarly, grabcut can be used to provide segmentations for tumor in brain MRIs where a bounding box must be drawn in the *suspicious region* and GrabCut could provide a segmentation for better analysis.

5.2. Improved Segmentation via Saliency Map Obtained from Teacher Network

GrabCut is a widely used segmentation algorithm in terms of the segmentation obtained while requiring minimum interaction from a user. An ideal segmentation algorithm would be one where no user input would be required. GrabCut requires the user to draw a bounding box and outputs a segmented image. The user can then improve the quality of the segmentation by marking areas where the algorithm provided a poor segmentation with brush strokes corresponding to definite foreground and definite background. The idea is to use a pre-trained classifier network (teacher network) to obtain a saliency map of an image and use that as the mask input to the GrabCut algorithm. The expectation is that the top 10% of the pixels of the saliency map will almost certainly lie in the region corresponding to the object and thus we should be able to provide this

as *sure background* information to the algorithm. For our experiment we use a vgg13 pre-trained on the CIFAR10 dataset to obtain the saliency map. The saliency map is obtained as

$$S_c(I) = w_c^T I + b_c$$

Where S_c is the classification score for the c^{th} class. We obtain the saliency as the derivative of the classification score with respect to the input

$$w = \frac{\delta S_c}{\delta I}$$

Here, w is the saliency map obtained.

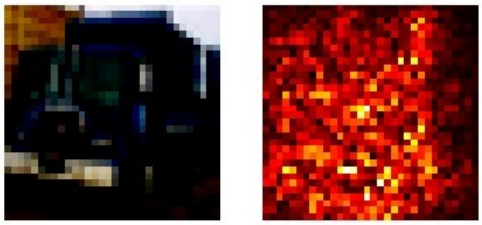

Figure 9. Image and Saliency map

5.2.1. Results with Saliency Map

We provided the Cifar10 image and the corresponding saliency map as the input to the algorithm. The bounding box was chosen to be the entire image so that user interactivity would not be required. The results obtained are shown in Figure 11. From the above image, we can see, the pixels not part of the saliency map were considered background and removed, thus we managed to perform automated segmentation with the same GrabCut algorithm. The results weren't as good since we used a very low resolution image with improperly defined objects. This was due to the nature of the CIFAR10 dataset, if we obtained saliency maps of models trained on other datasets we could obtain better segmentations.

5.2.2. Improvements with Adversarially Trained Teacher Models

We study the connection between linearized robustness and alignment in[3]. Thus, it can be concluded that adversarially trained teacher models which might be used to provide even better segmentations.

5.2.3. Limitations

The limitations are that saliency maps cannot be obtained for every image since we would need a pretrained model to "classify" that image which might not always be possible. The advantages

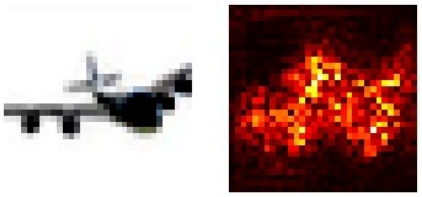

Figure 10. Image and saliency map

Figure 11. Segmentation with saliency map

are that a trained model can segment entirely new images never seen as well for which we could obtain segmentation. The main motivation of the task was to test whether saliency maps could be used for segmentation.

6. Individual Contributions

- Anirudh - Core algorithm implementation, medical image and saliency map applications
- Pratyay - User interactivity, documentation (slides and report)
- Aditya - Analysis, interactivity, ablation testing
- Yash - Integrating the 'frontend' (interactivity component) with the 'backend' (core algo), assisted with core algorithm implementation

References

[1] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. 23, 3 (August 2004), 309–314. https://doi.org/10.1145/1015706.1015720

[2] Y. Y. Boykov and M. . -P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images," Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, 2001, pp. 105-112 vol.1, doi: 10.1109/ICCV.2001.937505.

[3] Etmann, C., Lunz, S., Maass, P. amp; Schoenlieb, C.. (2019). On the Connection Between Adversarial Robustness and Saliency Map Interpretability. Proceedings of the 36th International Conference on Machine Learning, in Proceedings of Machine Learning Research 97:1823-1832 Available from https://proceedings.mlr.press/v97/etmann19a.html.