Probabilistic Graphical Models and their Applications - Final Report

TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context

Keyne Oei, 2580667
Lotfy Abdel Khaliq, 7013592
Amr Amer, 2572216

February 2021

1 Introduction & Problem Description

This report addresses the problem of semantic segmentation of photographs. The technique is detailed from "TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout and Context" by Shotton et al. [6]. The model utilise Texton technique which jointly search for relation between the pattern and the spatial layout of the image. The texton then used in unary classification and feature selection using shared boosting. Shotton et al. [6] also incorporate the contribution of color, location and edge potential into the conditional random field (CRF) model which makes a small improvement on segmentation accuracy.

In our experiment, we uses VOC2010 database [3]. There is 20 object classes followed by 11321 images and 1928 ground truth segmented image with different resolution each image. To encounter this, we resize the image to 300 x 500 for the sake of implementation simplicity.

In summary, our contribution goes as follows:

1. We generate the color, location, texture and edge potential
2. We train each potential (Color, Location, Texture, Edge) separately with its respective technique described in the paper
3. We use modified Joint Boost classifier to decide if the area is segmented w.r.t to the given class

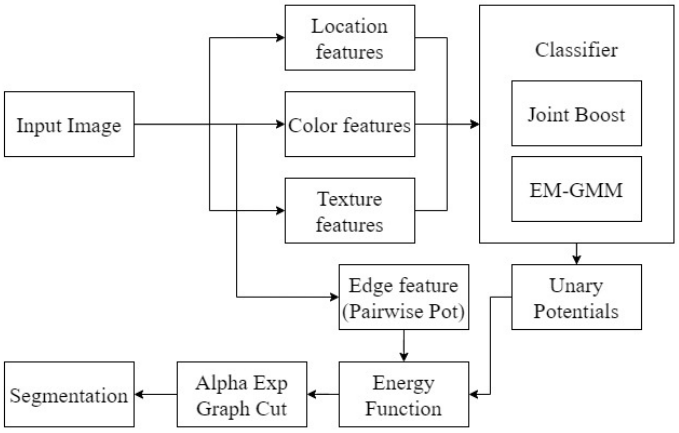


Figure 1: Our pipeline for the segmentation process.

2 Approach & Implementation

2.1 Conditional Random Field

The conditional random field (CRF) model is learned and used to predict the segmented area. The CRF can be described as follow:

$$log\, P(c|x,\theta) = \sum_i \psi_i(c_i, x; \theta_\psi) + \pi(c_i, x_i; \theta_\pi) + \lambda(c_i, x_i; \lambda + \sum_{i,j} \phi(c_i, c_j, g_{ij}; \theta_\phi) \tag{1}$$

where c is class labels and x is the image. To learn the potentials, we follow what Shotton et al. [6] suggest with the piecewise training technique. Thus, the parameter update is described independently with their respective learning technique. These learned potentials will be combined when performing the inference. For easier description, please see Fig 1 for our segmentation pipeline visualisation. To segment the area, we use alpha expansion graph cut algorithm for the optimal labeling. The detail of the process will be described as follow.

2.2 Unary Potential

2.2.1 Color Potential π

`feature-color/getColorPotential.m`
`feature-color/colorEM.m`

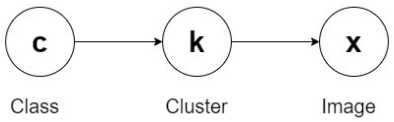


Figure 2: Graphical Model of the Color Potential

The unary color potential capture the distribution of color in an image. To represent the distribution, we uses the Gaussian Mixture Models (GMMs) in CIELab color space. The conditional probability can be defined such as $P(x|c) = \sum_k P(x|k)P(k|c)$ where $P(x|k) = \mathcal{N}(x|\mu_k, \Sigma_k)$. Please note that x represent the image, c represent the class and k represent the cluster. The Graphical Model of color potential can be seen in Fig 2. The color potential soft assignment where $P(k|x_i) \propto P(x_i|k)$ can be assigned such as

$$\pi(c_i, x; \theta_\pi) = log \sum_k \theta_\pi(c_i, k) P(k|x_i) \tag{2}$$

$$\theta_\pi(c_i, k) = (\frac{\sum_i [c_i = c_i*] P(k|x_i) + \alpha_\lambda}{\sum_i P(k|x_i) + \alpha_\lambda})^{w_\lambda} \tag{3}$$

To learn the color potential, first, we use K-means to learn the color clusters. Then, we use expectation minimization (EM) algorithm to get the GMMs μ_k and σ_k. We update the color potential and its parameter θ_π to the image and mask of the $c*$. The mask is obtained from the boost of the texture layout potential in each iteration. The color potential is only learned on test images and executed after the second iteration of performing inference.

2.2.2 Location Potential λ

`feature-location/getAndTrainLocationPotential.m`
`feature-location/getLocationProb.m`

The unary location potential capture the exact likelihood location in the image. The potential is generated in forms of images with a fixed dimension. To fit the fixed dimension, the input image is sampled uniformly. The potential can be computed such as

$$\lambda_i(c_i, i; \theta_\lambda) = log \theta_\lambda(c_i, \hat{i}); \tag{4}$$

$$\theta_\lambda(c, \hat{i}) = (\frac{N_{c,\hat{i}} + \alpha_\lambda}{N_{\hat{i}} + \alpha_\lambda})^{w_\lambda} \tag{5}$$

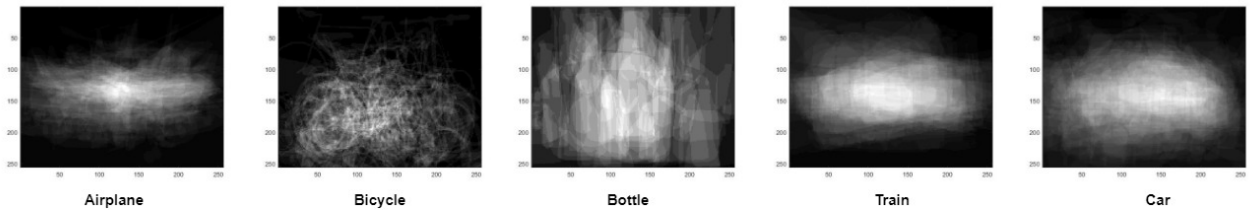


Figure 3: Our generated samples of learned location potentials.

To learn the location potential parameters, we simply maximize the likelihood of the normalized/sampled training images. We also given the option to raise the result with the fixed power of w_λ. The location parameter is updated by the fraction of the number of pixels of class c at a certain location $\hat{i}$ and the total number of pixel at certain location $\hat{i}$.

2.2.3 Texture Layout Potential ψ

```
feature-texton/applyFilterBank.m
feature-texton/getClassMap.m
feature-texton/fitCmap.m
feature-texton/getTextonPotential.m
feature-texton/trainTextonPotential.m
feature-texton/classifyTextonPotential.m
```

The unary texture layout potential defined such as $\psi_i(c_i, x; \theta_\psi) = log\, P(c_i|x, i)$ where $P(c_i|x, i)$ is the normalized distribution by boosted classifier. It is based by a set of features which called as *texture-layout filters*. Simplified, we can summarize the process of learning the texture layout potential as 3 steps such as follows

1. Textonization

To analyse and characterize the texture in image, Shotton et al. [6] suggest the use of texton [5]. These texton vocabulary become our representation of the image texture in a small set of response vectors. To get the texton map, we start by convolving the image CIELab color space with the 17D filter bank. The 17D filter bank consist of Gaussian filter at scales $\kappa, 2\kappa, 4\kappa$, x and y derivatives of Gaussian filter at scales $\kappa, 2\kappa$, Laplacian filter at scales $\kappa, 2\kappa, 4\kappa, 8\kappa$. After the convolution, we employ K-means algorithm to cluster the data. Each image will then be assigned to the nearest cluster center using the knn-search which generate the final texton map representing the texture of the class.

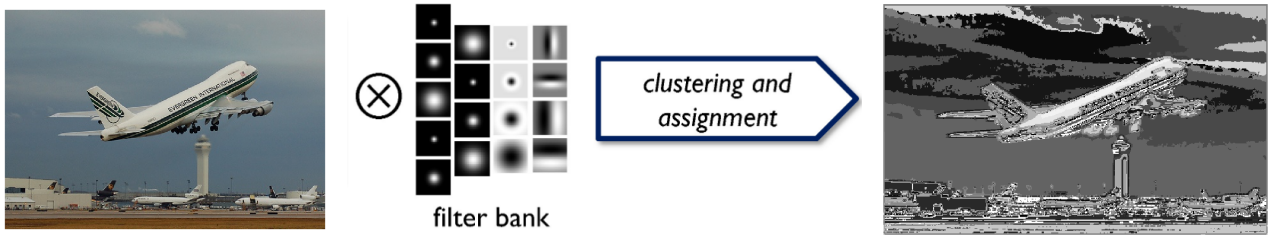


Figure 4: The input image is convolved with the 17D filter bank and then merged with filter response of all the training images. Then K-means is used to cluster the filter responses, then using KNN, each pixel is assigned to the nearest texton cluster center.

2. Texture Layout Filter

After getting the texton map, the texture layout filter can be generated. The Filter is represented by the pair of a region and the corresponding texton. The region is a box shaped which position are generated randomly inside the image. The value inside this box will contribute to the prediction of the segmentation. These generated random regions will be applied to a separated K channel texton map. See Fig 5 for the visualisation.

And for each of these texton map channel, an integral image will be computed forming the $v(i) = \hat{T}_{BR} - \hat{T}_{BL} - \hat{T}_{TR} + \hat{T}_{TL}$ where $\hat{T}$ is an integral image. The detail this integral image implementation is explained more by Viola et al. [7]. Integral image can be computed by the sum of the pixels of the image to the above

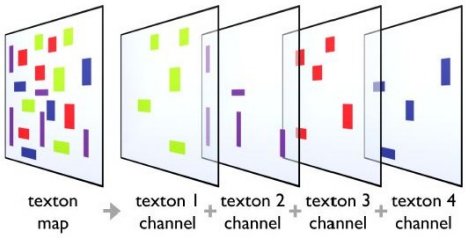


Figure 5: Texton map channels. [6]

and the left. Simplified, the integral image can be computed by having cumulative summation in the image.

3. Joint Boost

To train the texture layout potential, Joint Boost algorithm is used which iteratively will select different texture layout filter and combine them as classifier. The texture layout filters generated before is considered as the 'weak' learners and the summation of these weak learners is the 'strong' classifier. These can be defined such as $P(c|x,i) \propto exp\, H(c,i) = exp\,(\sum_m h_i^m(c))$ where h is our weak learners and H is our strong learner. The strong learner H here can be defined as the probability distribution over class using softmax transformation. The weak learner h can be defined from feature response such as

$$h_i(c) = \begin{cases} a[v_{r,t}(i) > \theta] + b & if\, c \in C \\ k^c & otherwise \end{cases} \tag{6}$$

From here, we can form an error function J_{wse} with Mean Squared Error (MSE).

$$J_{wse} = \sum_c \sum_i w_i^c (z_i^c - h_i^m(c))^2 \tag{7}$$

Optimizing these we can get the new value $w_i^c = w_i^c\, exp(-z_i^c\, h_i^m(c))$. The process of this optimization ensure that the weight will converge with increasing iterations. Shotton et al. [6] also give us a closed form solution of getting the $a, b, and\, k^c$. They are defined as follow.

$$b = \frac{\sum_c \sum_i w_i^c z_i^c [v(i,r,t) \le \theta]}{\sum_c \sum_i w_i^c [v(i,r,t) \le \theta]} \tag{8}$$

$$a + b = \frac{\sum_c \sum_i w_i^c z_i^c [v(i,r,t) \le \theta]}{\sum_c \sum_i w_i^c [v(i,r,t) \le \theta]} \tag{9}$$

$$k^c = \frac{\sum_i w_i^c z_i^c}{\sum_i w_i^c} \tag{10}$$

2.3 Pairwise Potential

`feature-texton/compute_pairwise.m`

After defining the unary potential terms for (location, color,& texture Potentials) that's promote a single node or pixel probability of being a class label(i.e. aeroplane,person) or background label.We also need a way to incorporate pairwise dependencies between nodes or neighbours pixels using a pairwise potential potential.

According to Shotton et al. [6] the pairwise potential is denoted by an edge pairwise potential as following:

$$\sum_{(i,j)\in\mathcal{E}} \overbrace{\phi\,(c_i, c_j, \mathbf{g}_{ij}(\mathbf{x}); \boldsymbol{\theta}_\phi)}^{\text{edge}} \tag{11}$$

Where i and j are different index pixels locations of the image corresponding to nodes in the conditional random field graph, and we sum on each potential ϕ between neighbours pixels for the 4 edges of the 4 surrounding pixels or local grid $\mathcal{E}$.

The $\phi\,(c_i, c_j, \mathbf{g}_{ij}(\mathbf{x}); \boldsymbol{\theta}_\phi)$ is a contrast Potts model that prompt the probabilities of neighbours pixels corresponds to the same class label by taking a weighted difference between the pixels color intensities such as

$$\phi\,(c_i, c_j, \mathbf{g}_{ij}(\mathbf{x}); \boldsymbol{\theta}_\phi) = -\boldsymbol{\theta}_\phi^T \mathbf{g}_{ij}(\mathbf{x})\,[c_i \neq c_j] \tag{12}$$

Where $\mathbf{g}_{i,j}$ is an edge feature that measures colors between neighbor pixels $\mathbf{x}_i$ and $\mathbf{x}_j$ as follow

$$\mathbf{g}_{ij} = \begin{bmatrix} \exp\left(-\beta \left\|x_i - x_j\right\|^2\right) \\ 1 \end{bmatrix} \tag{13}$$

The weight β is depending on the contrast of each image by taking an average sum over the difference between the pixels colors as follow

$$\left(2\left\langle \left\|x_i - x_j\right\|^2 \right\rangle\right)^{-1} \tag{14}$$

2.4 Graph Cut Energy Minimization

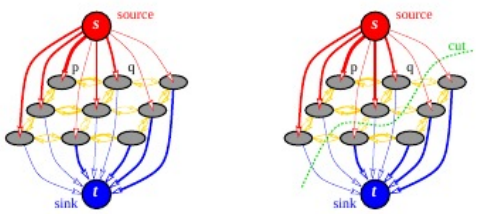
`feature-texton/alphaExpansion.m`

After defining the CRF model and identifying each potential function as both unary and pairwise terms.Now we do the label inference by finding the optimal class labels corresponding to each pixel or graph node, where $\mathbf{c}^\star$ is the optimal label that maximize the conditional probability mentioned in (1) or correspondingly we minimize the energy function such as

$$E(X) = \sum_i \psi_i\left(X_i\right) + \sum_{i,j \in N} \psi_{ij}\left(X_i, X_j\right) \tag{15}$$

and probability density is $P(X) = \frac{1}{Z}\exp(-E(X))$.We found the corresponding optimal label by minimizing the energy using a combinatorial graph cut algorithm called alpha-expansion.
Alpha-expansion algorithm can finds a good energy approximation in efficient polynomial time by iteratively running max-flow algorithm on the graph represented by our input images pixels as graph's nodes. We implemented on a grid CRF graph as $\mathbf{N}$ in equation (15) refer to the only 4 neighbours pixels of our image where a better results can be achieved if we consider a dense CRF model.

The problem that 2D graph can be very large according to number of nodes or equivalently the image pixels,which makes energy minimizing an issue and that's the reason behind using approximation minimization algorithms such as alpha expansion by Yuri Boykov and M-P Jolly. [2]



(a) A graph $\mathcal{G}$ (b) A cut on $\mathcal{G}$

Figure 6: (a)Example of 2D graph grid with 3 x 3 image with two labels source(background: label 0) and sink (class: label 1),(b)A graph cut spliting pixels in two disjoint sets or classes

The idea behind graph cut algorithm is simply illustrated in Fig 6.We only consider the case of two labels for simplifying the idea, where a cut on a graph with two terminals (source, and sink) or two class labels (0,1) partitioning the nodes or the image pixels in two disjoint sets each corresponds to different class labels.Since edge weights are Incorporated the energy potential in the pairwise term so a minimum cost cut will correspond to a labeling with minimum value of the energy for more details please refer to Yuri Boykov and Vladimir Kolmogorov. [1]

For implementing alpha expansion algorithm we used a Matlab API Wrapper by Andrew Delong, Which is actually based on Yuri Boykov and Vladimir Kolmogorov C-Plus code that implements the maxflow algorithm described by Yuri Boykov and Vladimir Kolmogorov. [1]

3 Result

Boosting

In Fig 7, we illustrate 4 different increasing number of boosting iteration for aeroplane class segmentation. With only 10 iterations, the parameter seems to only learn the most likely position of the airplane which is in the middle position. With more iterations, the texture layout filter successfully adding more region to the

(a) (b) (c) (d)

Figure 7: Increasing Boost Iteration. (a) Num Boost: 10. (b) Num Boost: 50. (c) Num Boost: 100. (d) Num Boost: 750.

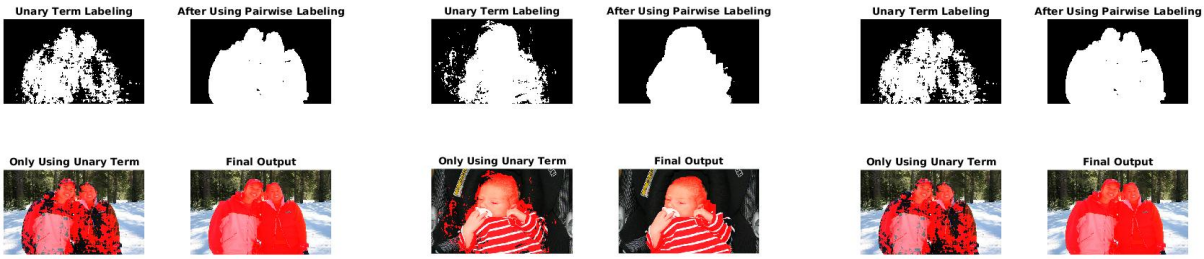


Figure 8: Person segmentation results. *Left Section of Image*: Unary Term Labeling. *Right Section of Image*: After Pairwise Labeling and the output with the image.

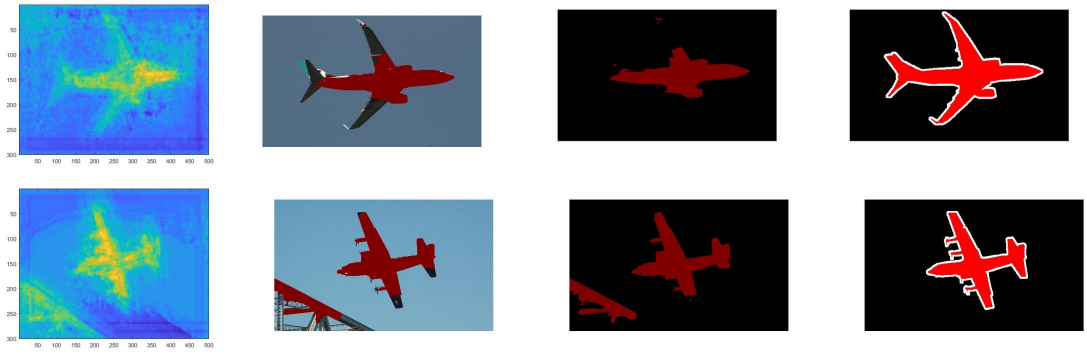


Figure 9: Airplane Segmentation Map. The average accuracy is 81% on our test set. Number of K-means cluster is set to 100 and Joint Boost iteration is set to 750.

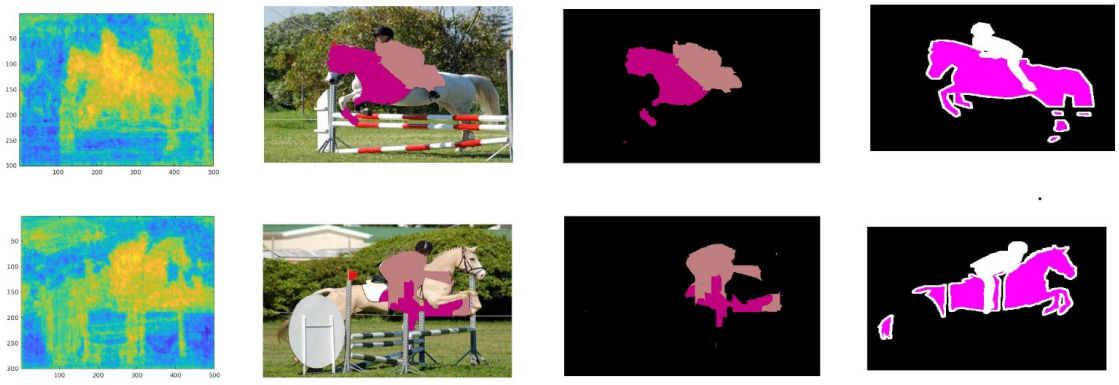
segmentation area. By iteration 750, it gives an accurate segmentation on the shape of the aeroplane.

Single Class Segmentation

In Fig 8 and Fig 9, we illustrate several result images for single class segmentation that trains only with single class training set. The performance of the pairwise potential addition and alpha expansion graph-cut algorithm can be seen specifically in Fig 8 where it connects the unary potentials and segments the correct region class. For single class segmentation, we get a high accuracy. Shown in Fig 9, the average accuracy is 81% on the aeroplane test set.

Mixed Class Segmentation

In Fig 10, we illustrate result images for mixed class segmentation. In this experiment, we used 100 images for Person and Horse classes using 100 clusters for Kmeans and 700 num boost for each class. We can observe that the results are not as good compared to training on only one class because it needed more training images for the two classes combined as well as more num boost. Also the performance can be greatly improved using a dense CRF pairwise potential instead of the Grid CRF approach [4].



(a) (b) (c) (d)

Figure 10: Multiclass Segmentation Map. (a) Confidence Map. (b) and (c) is the Labeled Map. (d) Ground Truth Image.

4 Conclusion & Further Improvements

With our time spent on this project, we have read several papers that point out Shotton et al. [6] limitation and also possible improvements to further increase the accuracy. Some of them are listed below.

1. **Fully Connected Dense-CRF.** Philipp Krahenbuhl [4] proposed another approach for segmenting the image using a fully connected dense CRF graph, while Shotton et al [6] used a grid CRF comparing only 4 neighbour pixels.A fully connected Dense CRF let us compare the current node or image pixel with all others graph nodes which incorporates more information at the pairwise potential function.For energy minimization they used a mean field approximation with KL-divergence at each iteration to approximate the final probability density function, and a Bilateral-Gaussian pair potential which smooth the result using a gaussian kernel and a bilateral part of the function comparing pixels colors and spatial locations or indices.

2. **CRF-RNN.** Zheng et al. [8] uses mean field approximate inference for the CRD with Gaussian pairwise potentials as RNN. For unary energy, the model obtain the data using the Convolution Neural Network (CNN) which predicts the labels for pixel. Different from Shotton et al [6] implementation that highlight the use of texture filter layout, Zheng et al. highlight the use of multiple mean-field iteration to estimate the unary value. The method successfully achieve the accuracy of 75.7% in VOC2010 dataset.

A limitation that we are experiencing ourselves is the time constraints to train such large data. Since we use matlab, the process is computed in CPU which constraints the capability to process big data effectively in short time. The implementation can be further improved by the GPU library provided by matlab, however, the time constraints of 1 month set us back in this regard.

Our experiment gives accurate segmentation for single class but it struggle to achieve the same accuracy for the mixed class segmentation. We acknowledge this struggle because of small amount for the training images used in our experiment. For better accuracy, we propose the implementation using GPU which will cut down hours of training. The use of neural network is also recommended for better accuracy and implementation since it can handle the non-linear weights better representing each feature, thus, giving better confidence map.

References

[1] Yuri Boykov and Vladimir Kolmogorov. "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision". In: *IEEE transactions on pattern analysis and machine intelligence* 26.9 (2004), pp. 1124–1137.

[2] Yuri Y Boykov and M-P Jolly. "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images". In: *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*. Vol. 1. IEEE. 2001, pp. 105–112.

[3] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results*. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.

[4] Philipp Krähenbühl and Vladlen Koltun. "Efficient inference in fully connected crfs with gaussian edge potentials". In: *arXiv preprint arXiv:1210.5644* (2012).

[5] Jitendra Malik et al. "Contour and Texture Analysis for Image Segmentation". In: *International Journal of Computer Vision* 43 (June 2001), pp. 7–27. DOI: 10.1023/A:1011174803800.

[6] Jamie Shotton et al. "TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context". In: *Int. Journal of Computer Vision (IJCV)* (Jan. 2009). URL: https://www.microsoft.com/en-us/research/publication/textonboost-for-image-understanding-multi-class-object-recognition-and-segmentation-by-jointly-modeling-texture-layout-and-context/.

[7] P. Viola and M. Jones. "Rapid object detection using a boosted cascade of simple features". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 1. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.

[8] Shuai Zheng et al. "Conditional Random Fields as Recurrent Neural Networks". In: *International Conference on Computer Vision (ICCV)*. 2015.