A Review of Methods for Image Segmentation to Identify Trees

Honours Literature Review

Michael Scott
Computer Science
University of Cape Town
Western Cape South Africa
michael.ross.scott@gmail.com

Abstract

Although image segmentation using neural networks (namely Fully Convolutional Neural Networks, U-Nets and Atrous Networks) achieves accurate results. The limitations of processing, training time and accurate ground truths remain an obstacle. In light of these limitations, this literature review explores large feature extractors (Hough Transforms and templating), rule-based approaches (Decision Trees and Support Vector Machines) and the aforementioned neural networks in order to uncover a mitigation strategy for the limitations listed. From the analysis, it is clearly shown that A U-Net or Atrous Neural Network should be used to perform image segmentation due to their accuracy. Rule-based systems and feature extractors are also shown capable of improving ground truth image segmentation. Lastly, to reduce training times, feature extractors will be used as a pre-processing step before the image is processed in the neural network.

CCS CONCEPTS

• Machine Learning

• Image Processing

• Computer Vision

KEYWORDS

Hough Transformation, Template Matching, FCNN, U-net, Atrous Neural Network, Support Vector Machine, Decision Trees

1 INTRODUCTION

Image segmentation is a method of processing an image in order to classify various objects that lie within that image. The groundwork of the field was first laid in the 1960s by Lawrence Robert's who developed an algorithm to partition digital images based on their edges [1]. Today there are many methods, as well as architecture classes, that can be used in order to partition images into various objects.

Images are typically transformed into image maps using various methods of processing. These maps are then analysed by architecture in order to perform segmentation. Methods for processing those images are typically broken up into small feature extractors (pixel-wise analysis), "edge-based approaches" and large feature extractors (both analysing a neighbourhood of pixels) [1].

The aim of this paper is to analyse the literature of select large feature extractors for image processing and select neural network frameworks (as well as viable alternatives) for image segmentation. This will be done by first critiquing those deep learning architectures, critiquing their alternatives then examining feature extraction for image processing. Finally, these methodologies will be compared and contrasted, and suitable conclusions will be drawn.

1.1 Architecture Classes

The architectures investigated are as follows:

1) Neural Networks - these are architectures which use various connected layers that perform image operations to an image matrix. The results of which are passed to the next layer. Operation weights are adjusted by the error produced in classification (using annotated data) [2-11]. In this review, Fully Convolutional Neural Networks, U-nets and Atrous Networks will be examined.

2) Rule-based systems – these are architectures that classify objects given a system of rules. Typically, these rules are built using training data [12-18]. The architectures investigated are as follows: Support Vector Machines, Decision Trees.

1.2 Large Feature Extractors

The select approaches investigated are namely:

1) Hough transformations - this is a mathematical method that detects shapes of an image by transforming the image space and accumulating votes in that transformed image space. When those votes reach a certain threshold then the shape has been found. [19-22]

2) Templating - this is a method of identifying objects within images using a blueprint of a globally similar object [23, 24]

This analysis seeks to inform Aerobotics of improved method(s) of image segmentation for drone images containing trees.

2 BACKGROUND

Aerobotics (at the time of writing) provide an analytics service to facilitate smart crop management. This service is only available to tree farmers.

The company use manned drones that fly over the tree farm in order to take high-resolution aerial images of customers farms. Pixel-wise information is extracted from those images (RGB, Near Infrared, chlorophyll absorption and height) and a standard convolutional neural network is used in order to classify objects into tree and tree boundaries. Further analysis is then done on those tree objects in order to make recommendations based on their perceptual health. This is information is then made available to the customer.

3 NEURAL NETWORKS

There are a wide variety of neural networks that perform image segmentation. The highest performing of these seem to be fully connected deep learning networks [7-12].

A standard convolutional neural network (CNN) is one of the simplest deep learning networks and is the basis for the architecture explored in this literature review. The following CNNs will be examined: Fully Convolutional Neural Networks, U-nets and Atrous Networks. Each network review will take the form of a brief overview, examination of their characteristics and a brief look at an improved implementation.

3.1 Fully Convolutional Neural Network

3.1.1 Overview

A Fully Convolutional Neural Network (FCNN) is a CNN without the Fully Connected Layer or one with the fully connected layer nested between convolutional layers [3, 5].

The fully connected layers (at the end of a typical CNN) are instead replaced with convolutional layers. The main crux of this is that the locations in layers further down the convolutions are spatially related. By replacing the fully connected layers with convolutional layers, an FCNN can learn to capture spatial relationships and effectively ‘map’ it to an output image (of similar proportions to the input image) [3].

3.1.2 Characteristics

Traditional FCNNs offer good accuracy in image segmentation. Unfortunately, they suffer from some critical drawbacks.

1) The first of which is a loss of global pixel information caused by the repeated pooling and convolutional operations. This causes “fuzzy object” segmentation in the resultant segmented image [6].

2) The second critical drawback is that they perform poorly when objects in images are irregularly large or small as compared to the “receptive fields” in a classical FCNN [4, 5].

3.1.3 Improved Implementation

Solutions to the second drawback are often solved by making use of fully connected layers [4, 5]. Solutions to the first drawback, however, need more complex strategies; below is one such solution:

Multi-scale: This technique tackles both problems mentioned. It does this by using a “multi-scale” FCNN technique in order to do so. Each level of this FCNN is used to predict image “features”. Every successive result obtained from those levels are upsampled and concatenated with the original image (which has undergone a convolution and pooling operation) as input for the next level (called scales).
The first scale of this network analyses the image using a “full image view” which it achieves by making use of 2 fully connected layers, convolutions and pooling. This allows the image to predict features at a low resolution. Subsequent scales analyse finer details of the image. The output image map is then passed to the layer below (Figure 2). As a result, output images are of a higher resolution with the added benefit of solving the second drawback [5].

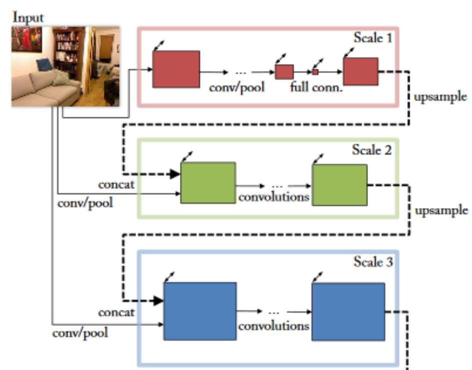

Figure 1: Multi-scale Deep Learning Structure

3.2 U-Net

3.2.1 Overview

A U-net is a U shaped architecture that uses upsampling and concatenation operations in order to capture global pixel relations effectively, much like the Multi-scale FCNN. The first part refers to the left side of the U, whilst the second refers to the right side of the U (see Figure 3).

1) The first part of the U-shaped architecture follows the form of a typical FCNN that specifically makes use of “max pooling, ReLu and convolutions”.

2) The second part of the U-shaped architecture makes use of the aforementioned upsampling and concatenation. Upsampling is performed on feature maps outputted by layers that are part of (2). Concatenation and cropping operations are then performed on the outputted feature maps from (1). The resultant image then undergoes two convolutions followed by a ReLu [7].

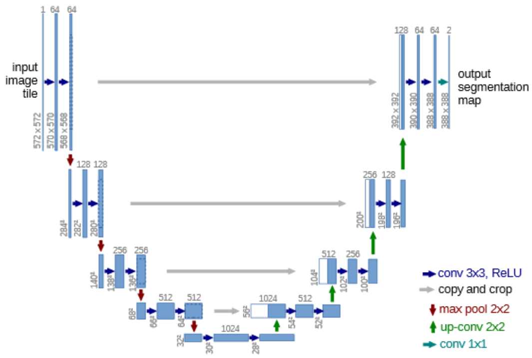

Figure 2: U-net Structure

3.2.2 Characteristics

U-nets are typically faster to train than other FCNN architectures. It is conjectured that the reason for this is the multiple upsampling operations. This causes a two-tuple effect.

1) Improves the rate at which the network learns using more effective mechanisms of passing on pixel information [7, 8]

2) Higher levels in the network become much more representative of local pixel information [8].

3.2.3 Improved Implementation

Cutting edge advances have been made in the last two years in order to obtain better performance from U-nets. Here is a different approach to implementing a U-net in order to improve image segmentation even further:

SUNet: This approach uses stacked U-nets to achieve better accuracy. Every U-net layer only has two levels of depth associated with it. U-net layers perform convolutions to examine lower-level features. Deconvolutions and concatenation from an output a few layers up are used in order to append that information to higher resolution maps (Figure 4). Unlike traditional U-Nets, cropping operations are eliminated in order to preserve the size of images [9].

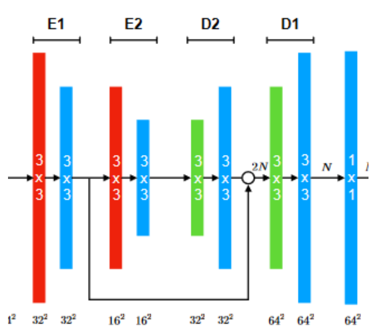

Figure 3: SUNet Layer Structure

Blue and red layers are convolutional strides. Green layers are deconvolutional strides. E1 and E2 are operations to examine pixel information at a lower level. D2 combines features from the lower levels with spatial and resolution properties of the higher level using concatenation (white circle).

A series of convolutional, residual and U-net blocks are then used to form this SUNet. It uses dilation operations (drop pooling stride and dilate subsequent convolutional filters by 2) in order to preserve spatial awareness from U-net to U-net [9].

3.3 Atrous

3.3.1 Overview

Atrous networks that are used for image segmentation typically take the form of Deep Connected Neural Networks (DCNN) that have been implemented to be fully convolutional. These DCNNs use - what is called - an Atrous algorithm for convolutional filters. The goal is to accurately segment objects in images with many features and at a high resolution [10, 11]. This is done by padding the convolutional filter matrices with 0s in order to capture spatial information. In essence, these matrices appear to be a “filter with holes” (Figure 4) [11].

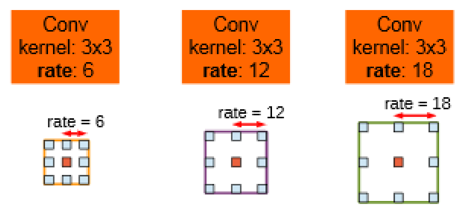

Figure 4: Atrous Filter

3.3.2 Characteristics

Atrous networks are normally faster to train because their operations are not as computationally expensive as other architectures [10]. They also allow users to choose the resolution of the resultant image segments [11].

The disadvantages of these networks are that they can sometimes produce fuzzy segmentations. This is because they have a trade-off between spatial accuracy and segmentation classification [11].

3.3.3 Improved Implementations

There are numerous methods that Atrous networks typically implore to solve this problem. One such method is a post-processing step using a Conditional Random Field (CRF) to “smooth” some of the fuzzy segmentation maps. This decreases the image’s resolution [10]. There are, however, more effective mechanisms which can be built into an Atrous network. One of these mechanisms is examined below.

Atrous Pyramid: This approach uses what is called pyramid pooling with Atrous convolutions. This is where simultaneous Atrous convolutions are performed on one feature map. Other operations are then applied according to the implementation of the particular Atrous. Atrous Pyramid pooling is used in conjunction with a multi-grid approach. The result is a better spatial accuracy in the resultant segmentation [11].

4 RULE-BASED SYSTEMS

There are a wide variety of rule-based systems which use various systems of rules to segment image objects using pre-processed image maps.

The analysis of these systems will include a brief overview of the method, typical applications in crop segmentation and the advantages that the system possesses.

4.1 Support Vector Machines

4.1.1 Overview

Support Vector Machines (SVM) are typically used to classify points of data depending on where they lie in reference to a “hyperplane”. The system uses this plane in order to make decisions on which classes input data will be classified [12, 13].

The plane is determined using training data where class classification is known. The SVM uses this data in order to determine the best equation for separation between classes to achieve maximum accuracy. After the hyperplane is determined all training data can be discarded and the SVM can now be used to perform classification tasks [14].

4.1.2 Applications

Applications examined in this review focused on SVMs in the crop image segmentation space. Namely to discriminate between weeds and crops and identify different families of trees in the Savannah [13-15].

Crop segmentation was performed using a pre-processing step of thresholding. This is used in order to eliminate all non-vegetation from the images. Additional processing to identify crop lines (Hough Transformation) or masking is used to try separate crops and weeds from background noise. The hyperplane equation is built using pixel data from weed or crops and the plane of separation that gives maximum accuracy is then arrived upon using the information derived from the data. Given a new image and the same segmentation processing, weeds and plants can now be classified. Subsequently, pixels in that image can be segmented by the aforementioned classification process [13, 14].

Trees in the Savannah were classified in a similar way with pre-processing steps of region growing and BDRF applied to the images to identify trees from aerial images. Region growing classified trees pixels into the same ‘crown’ if the height matched. BDRF was used in order to get rid of the impact of the direction of light on these images. This was then processed by using a stacked SVM in order to determine how best to separate classes of trees in those images [15].

4.1.3 Characteristics

Common properties of SVMs are as follows:

1) After the training is done, classification using SVMs is non-resource intensive.
2) The separation between classes does not need to be linear
3) SVMs memory allocations are seen as relatively efficient [14]

4.2 Decision Trees

4.2.1 Overview

Decision Trees (DT) are typically used in classification tasks in order to classify points of data using a system of rules. The rules are determined using knowledge of the features of the application. The order that the rules are applied is determined by using training data in order to accurately classify known data. The rule that misclassifies the data the least will be chosen. The resultant tree is then typically pruned using a variety of algorithms, this is to prevent overfitting [16, 17].

4.2.2 Applications

Applications involving DTs, in this review, focused on crop image segmentation. Namely mapping arid regions and identifying plants based on “greenness” in poor lighting conditions [16, 18].

Pre-processing is normally performed in order to eliminate noise and prepare the data to be effectively analysed.

In arid region segmentation, this took the form of pre-processing using a select software and masking to isolate those shrubs. PCA was used to select the most appropriate colour representation and SAVI was used to eliminate background grass. “Layer features” were then used to classify the shrubs, the DT selected the most appropriate ordering in order to determine how to classify the segmented plants. This was determined through training [16].

In plant image segmentation this took the form of transforming the image space into HSV. The threshold value used to isolate the plants was then determined using a decision tree to select the best range. This was determined using training data [18].

4.2.3 Characteristics

Common properties of DTs are as follows:

1) Overfitting is common when training DTs, this is mitigated by various algorithms
2) Good at identifying important variables that can be used to classify data out of a set of possible variables
3) Trial and error involved in choosing the best possible DT [16, 17, 18]

5 LARGE FEATURE EXTRACTORS

As examined in the previous section, image segmentation is generally performed post image map creation. These maps mainly represent data which you want to analyse with a given architecture. The effect of this is:

1) Architecture can be trained more accurately in order to segment images
2) Architectures only need to process the relevant pixels in the images [13-18]

The process of pixel neighbourhood segmentation typically follows 1 or more pre-processing steps in order to isolate key information [13, 14, 21]. However, sometimes these strategies can be plainly applied to the source image depending on the application.

This section will attempt to analyse two methods of extracting large features (i.e. extracting features from a pixel neighbourhood). This analysis will include a structure similar to section 3; wherein a brief overview of each technique will be given, characteristics of the technique will be examined, and a better approach will be investigated.

5.1 Hough Transformations

5.1.1 Overview

Hough transformations are used in order to detect lines, curves or geometric shapes in a given image [19].

A Hough transformation effectively reduces the global problem of finding shapes in an image to a simple problem of finding peaks in the Hough parameter space [20].

This is done using a "Point to Line" transformation [21] effectively mapping a single pixel to multiple hypothesises of curves/lines. The hypothesis of which pixel maps to what shape is arrived upon using "votes" from the Hough space. Stronger hypothesises (ones with the most votes) are then chosen [19-21].

The process of voting stems from the PTL (point to line transform) where a single point becomes a 'parameterised' line/curve in the Hough space. The intersection of these lines/curves in the Hough space increase the number of votes for the presence of that line/curve in the image space (see Figure 5) [22].

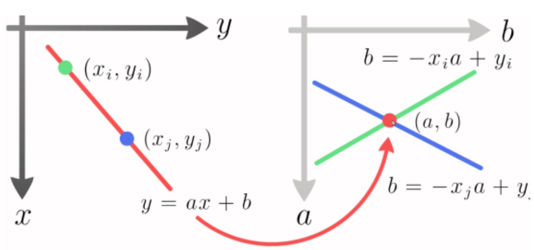

Figure 5: Basic Illustration of Hough Transform

On the left, there are many (x, y) coordinates that typically make up a line in the real image space. On the right those (x, y) coordinates are transformed into lines, these lines will inevitably intersect. The intersection of these lines at a single point say (a, b) will then correspond to the hypothesis of a line in the real image space.

There are many versions of Hough Transformations. The two main versions most useful to this project would be straight-line detection (to detect crop lines) and circle/ellipsis detection (to detect trees). Both of them use different parametrisations of the image space.

Straight-line detection uses the following parameters in order to check lines [19].

$$f(x,y) = p(\theta) = x\cos(\theta) + y\sin(\theta) \quad (1)$$

Whereas circle/ellipsis detection use these parameters [20]

$$f(x,y) = (x - a^2) + (y - b^2) - r^2 = 0 \quad (2)$$

5.1.2 Characteristics

Common properties of Hough transforms are that they have robust mechanisms to test shapes (non-edge pixels can be included in the voting process) and they are insensitive to image 'noise' [20].

However, they can be computationally expensive and use a lot of memory to perform their operations. Not to mention older implementations have not been able to find non-geometric shapes efficiently [19].

5.1.3 Improved Implementations

In order to achieve critical speedup, pre-processing is typically done in order to eliminate data not crucial for calculations and increase accuracy [21]. However, pre-processing eliminate context that may be important in determining shapes. Therefore, a Random Hough Transform is proposed as a mechanism of speeding up computation and maintaining image integrity. Random Hough Transforms select points at random in order to detect lines, this method is less robust in detection but uses less computation resources [19].

In order to match non-geometric shapes, Generalized Hough Transforms (GHT) can be used. One particular method of GHTs that is useful to this project is called a template matching Hough Transform. It uses a reference shape image to create an "R" table in order to perform detection in the actual image. The parameters include a reference point and the angle between those points. The R table is then used in the voting process to check those pixels in the image [19, 22].

5.2 Template Matching

5.1.1 Overview

Template matching is a strategy to detect shapes or patterns in a given image. This is done by comparing a reference shape to a particular target in the image. The most widely used method of comparison uses a normalized cross product. This operation was picked because it minimizes the effect of brightness on similarity and scales the correlation obtained to a value in the range of (-1, 1). Normally a threshold is picked for similarity that determines the segmentation [23].

5.1.2 Characteristics

Template matching as an approach is relatively simple to implement and provides serviceable results if the implementation is done correctly [23].

However, simple template matching cannot consider image objects that are rotated. Another key flaw is that comparison operations are expensive to perform on the whole image [23].

5.1.3 Improved Implementations

In order to obtain better accuracy when an image is rotated, a set of image patterns need to be computed in order to make better comparisons. After pre-processing is performed in order to isolate key features, a process of pyramid searching is done. Pyramid searching is performed in order to determine whether the target part of the image is in a similar orientation to one of the reference images. If it is, then the relevant reference image and target image are compared. Computation can be sped up by just comparing the edges of these orientation variant images [23].

Another technique involves rotating a light matrix around a reference image in order to simulate different angles of the sun. This method is performed when looking at trees. It effectively simulates template rotation by simulating real-world conditions [24].

6 DISCUSSION

The purpose of this section is to review the information captured in the previous three sections and discuss its implications in the context of the Aerobotics project. The order of discussion will follow a discourse on the type of characteristics the implemented system needs to capture, along with some of the challenges of this project; this will be discussed in the overview. Thereafter, each major section will be discussed with reference to the information laid out in the aforementioned overview.

6.1 Overview

6.1.1 Critical Characteristics

The system that is implemented to segment trees and tree boundaries needs to exhibit the following characteristics:

1) Accurate classification: the system needs to be accurate in its predictions.
2) Accurate segmentation: the system needs to be accurate in its labelling

6.1.2 Limitations

The project has the following limitations that could severely affect the final system. Therefore, methods of mitigation need to be explored. The limitations are as follows:

1) 'Ground truth' segmented images provided by Aerobotics (these are images that our system will use for training) have fuzzy border segmentation. This means that the system will be less accurate in segmenting borders of objects.
2) Training deep learning architecture schemes is often times computationally expensive. This can lead to long training times.

6.2 Neural Networks

The various neural networks examined in this paper will be the main component of the system used to segment images. Therefore, these architectures will be critiqued based on the critical characteristics needed for the project's system.

All these architectures classify image objects relatively accurately. This is due to the fact that they are based on a system that exhibits good performance on that task, namely CNNs [2].

However, not all of these architectures segment image objects accurately. This is due to the fact that some operations performed by the neural network, decrease the resolution of the image and lead to 'fuzzy image segmentation' [5]. U-nets and (some) Atrous Networks avoid this pitfall by performing operations to preserve the image's resolution [7-9, 11]. Whilst FCNNs can be altered to perform better [4, 5].

This means that both critical characteristics are addressed, but there are still limitations that these architectures cannot address

6.3 Rule-based Systems

The rule-based systems examined in this paper provided good segmentation results in their various applications [12-18]. The purpose of examining these systems was to investigate the feasibility of using them to improve the 'ground truth' images used for training. Specifically, to train the various neural network architectures that were examined. If the rule-based systems could achieve better accuracy in defining borders, it could lead to a more accurate neural network. This would be a good mitigation technique for the first limitation listed.

6.4 Pixel Segmentation

Pixel segmentation could be used as a pre-processing step for images examined by the neural network architectures. It is hypothesized that processing these image maps could lead the neural network layers to learn image features more quickly, decreasing the time it takes to train those neural networks. This technique is mostly covered in the Rule-based section, where it is needed in order for the systems to perform segmentation. However, neural networks often learn these features via back-propagation [2], therefore this pre-processing is not normally investigated in the literature.

Large feature extractors, in particular, isolate critical features in the image [19-24]. This means that they could be useful in order to process images into image maps that have important elements, such as trees and crop lines, highlighted. As examined above Hough transformations could be useful for identifying trees and tree rows; whilst templating could be useful for identifying trees.

This pre-processing step could reduce training times and slightly improve the segmentation results of the neural networks implemented. It could also be used in conjunction with rule-based approaches in order to obtain more accurate segmentation [12-18].

7 CONCLUSIONS

Having discussed the implications of the techniques and systems investigated in the literature the following conclusions can be drawn.

Firstly, Atrous and U-Net neural networks would be the most suitable architectures to use for segmenting images. This is due to their segmentation accuracy as compared to typical FCNNs.

However, the drawbacks of neural networks (namely training times) are still present. These will be resolved by performing various image segmentation techniques on images. The most impactful of these would be the large feature extractors, due to the fact that they highlight the most useful information in images (namely the objects that need to be segmented).

Lastly, 'Fuzzy' ground truth images, images that neural networks analyse in order to train, can be improved by using rule-based systems and feature extractors. This is hypothesized to achieve more accurate segmentation.

8 REFERENCES

[1] Yu-Jin Zhang. 2006. An Overview of Image and Video Segmentation in the Last 40 Years. Advances in Image and Video Segmentation (2006), 1–16. DOI: http://dx.doi.org/10.4018/978-1-59140-753-9.ch001

[2] Rikiya Yamashita, Mizuho Nishio, and Richard Kinh Gian Do. 2018. Convolutional neural networks: an overview and application in radiology. Insights into Imaging 9, 4 (August 2018), 611–629. DOI: http://dx.doi.org/https://doi.org/10.1007/s13244-018-0639-9

[3] Jonathon Long, Evan Shelhammer, Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. The IEEE Conference on Computer Vision and Pattern Recognition (CPVR) (2015), 3431-3440. DOI: https://doi.org/10.1109/cvpr.2015.7298965

[4] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. 2015. Learning Deconvolution Network for Semantic Segmentation. 2015 IEEE International Conference on Computer Vision (ICCV) (2015), 1520–1528. DOI: http://dx.doi.org/10.1109/iccv.2015.178

[5] David Eigen and Rob Fergus. 2015. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. 2015 IEEE International Conference on Computer Vision (ICCV) (2015), 2650–2658. DOI: http://dx.doi.org/10.1109/iccv.2015.304

[6] Yanming Guo, Yu Liu, Theodorous Georgiou, and Michael S. Lew. 2018. A review of semantic segmentation using deep neural networks. International Journal of multimedia retrieval 7, 2 (2018), 87–93. DOI:http://dx.doi.org/10.1007/s13735-017-0141-z

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer Assisted Intervention (November 2015), 234–241. DOI: https://doi.org/10.1007/978-3-319-24574-4_28

[8] Zhengxin Zhang, Qingjie Liu, and Yunhong Wang. 2018. Road Extraction by Deep Residual U-Net. IEEE Geoscience and Remote Sensing Letters 15, 5 (2018), 749–753. DOI: http://dx.doi.org/10.1109/lgrs.2018.2802944

[9] Sohil Shah, Pallabi Ghosh, Larry S. Davis and Tom Goldstein. 2018. Stacked U-Nets: a no-frills approach to natural image segmentation. arXiv: 1804.10343. Retrieved from https://arxiv.org/abs/1804.10343

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, Alan Yuille. 2014. Semantic image segmentation with deep convolutional nets and fully connected CRFs. arXiv: 1412.7062. Retrieved from https://arxiv.org/abs/1412.7062

[11] Liang-Chieh Chen, George Papandreou, Florian Schroff and Hartwig Adam. 2017. Rethinking atrous convolution for semantic image segmentation. arXiv: 1706.05587. Retrieved from https://arxiv.org/abs/1706.05587

[12] Trevor Hastie, Jerome Friedman, and Robert Tisbshirani. 2017. The Elements of statistical learning: data mining, inference, and prediction 2nd ed., New York: Springer.

[13] J.m. Guerrero, G. Pajares, M. Montalvo, J. Romeo, and M. Guijarro. 2012. Support Vector Machines for crop/weeds identification in maize fields. Expert Systems with Applications 39, 12 (2012), 11149–11155. DOI: http://dx.doi.org/10.1016/j.eswa.2012.03.040

[14] Alberto Tellaeche, Gonzalo Pajares, Xavier P. Burgos-Artizzu, and Angela Ribeiro. 2011. A computer vision approach for weeds identification through Support Vector Machines. Applied Soft Computing 11, 1 (2011), 908–915. DOI: http://dx.doi.org/10.1016/j.asoc.2010.01.011

[15] Matthew Colgan, Claire Baldeck, Jean-Baptiste Féret, and Gregory Asner. 2012. Mapping Savanna Tree Species at Ecosystem Scales Using Support Vector Machine Classification and BRDF Correction on Airborne Hyperspectral and LiDAR Data. Remote Sensing 4, 11 (2012), 3462–3480. DOI: http://dx.doi.org/10.3390/rs4113462

[16] Andrea S. Laliberte, Ed L. Fredrickson, and Albert Rango. 2007. Combining Decision Trees with Hierarchical Object-oriented Image Analysis for Mapping Arid Rangelands. *Photogrammetric Engineering & Remote Sensing* 73, 2 (January 2007), 197–207. DOI: http://dx.doi.org/10.14358/pers.73.2.197

[17] José M. Peña-Barragán, Moffatt K. Ngugi, Richard E. Plant, and Johan Six. 2011. Object-based crop identification using multiple vegetation indices, textural features and crop phenology. *Remote Sensing of Environment* 115, 6 (2011), 1301–1316. DOI: http://dx.doi.org/10.1016/j.rse.2011.01.009

[18] Yumiko Suzuki, Hiroshi Okamoto, and Takashi Kataoka. 2008. Image Segmentation between Crop and Weed using Hyperspectral Imaging for Weed Detection in Soybean Field. Environment Control in Biology 46, 3 (2008), 163–173. DOI: http://dx.doi.org/10.2525/ecb.46.163

[19] Olga Barinova, Victor Lempitsky, and Pushmeet Kohli. 2010. On detection of multiple object instances using hough transforms. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2010). DOI:http://dx.doi.org/10.1109/cvpr.2010.5539905

[20] Priyanka Mukhopadhyay and Bidyut B. Chaudhuri. 2015. A survey of Hough Transform. Pattern Recognition 48, 3 (2015), 993–1010. DOI: http://dx.doi.org/10.1016/j.patcog.2014.08.027

[21] Qi-Chuan Tian, Quan Pan, Yong-Mei Cheng, and Quan-Xue Gao. 2005. Fast algorithm and application of Hough transform in iris segmentation. Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826) 7 (January 2005), 3977–3980. DOI: http://dx.doi.org/10.1109/icmlc.2004.1384533

[22] Markus Ulrich, Carsten Steger, and Albert Baumgartner. 2003. Real-time object recognition using a modified generalized Hough transform. Pattern Recognition 36, 11 (2003), 2557–2570. DOI: http://dx.doi.org/10.1016/s0031-3203(03)00169-9

[23] Nazil Perveen, Darshan Kumar, and Ishan Bhardwaj. 2013. An overview of template matching methodologies and its applications. International Journal of Research in Computer and Communication Technology 2, 10 (October 2013), 988–995.

[24] Mats Erikson and Kenneth Olofsson. 2005. Comparison of three individual tree crown detection methods. Machine Vision and Applications 16, 4 (2005), 258–265. DOI: http://dx.doi.org/10.1007/s00138-005-0180-y