# Cell Detection and Classification from Urine Sediment Microscopic Images

Dipam Goswami, Hariom Aggrawal, Vinti Agarwal

Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani, Pilani Campus, India

**Abstract.** This paper proposes deep learning approaches to automate the detection and classification of pus cells (WBCs), red blood cells (RBCs), epithelial cells and others. It also identifies clusters of these cells which indicates serious conditions of urine infection (indicated by the presence of pus cells and its clusters), kidney stones and other diseases. We prepared an annotated dataset of microscopic images of urine sediment particles which is expected to accelerate research in Biomedical Imaging. We trained YOLOv3 and RetinaNet models on this dataset which can be used to detect and classify the cells from urine samples for medical diagnosis.

**Keywords:** Urine Sediment Analysis · Cell Detection · YOLOv3 · RetinaNet

## 1 Introduction

Analysis of urine sediment particles from microscopic images play a vital role in the diagnosis of kidney diseases. The patient's urine is subjected to centrifuging and the sediment(solid) part of the urine is taken in a glass slide and observed under the microscope. The clinical examination of the urine sediment requires skilled technicians who has expertise in identifying the cells from images. The trained technicians then count the number of various cells by visual inspection, which is labor-intensive, time-consuming and dependent on the technician. There has been considerable interest in developing automated systems for detecting and classifying cells from digital images of urine sediment and blood smears.

This paper presents a dataset comprising of more than 500 microscopic images of urine sediment particles. We use this dataset to train YOLOv3 and RetinaNet models in order to detect the cells and classify them as RBCs, pus cells/WBCs, epithelial cells and cluster of pus cells. This classification of cells is vital for the detection of diseases and its diagnosis. This paper is aimed at providing medical support to medical centres across rural areas where there is a shortage of skilled lab technicians to analyze urine samples of patients.

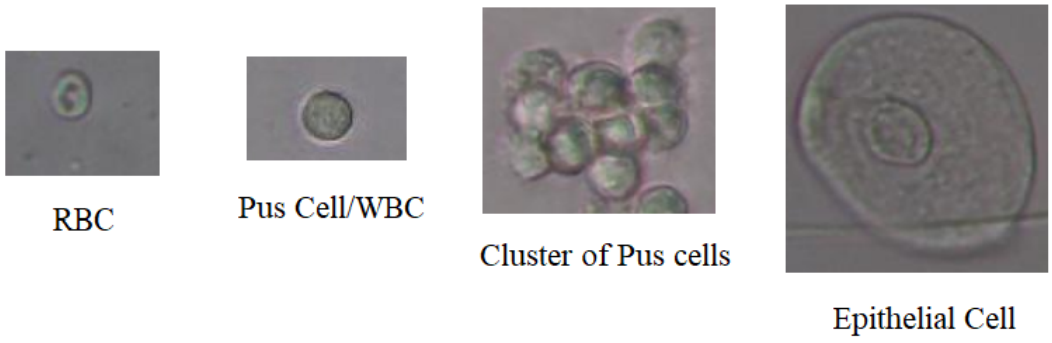


**Fig. 1.** Cells obtained from microscopic images of urine sediment samples

## 2 Related Work

Several efforts have been made by researchers to automate cell classification from digital microscopic images using deep learning approaches. In [1] Liang et al. explored two state-of-the-art object detection methods namely Faster R-CNN [2] and SSD [3] along with their variants for urine particles recognition. Faster R-CNN belongs to the Family of 2 stage object detectors. R-CNN [4] extracts a bunch of regions from the image using selective search[5], and then checks if these boxes contains an object. We first extract these regions, and then apply CNN to each of these regions to extract specific features. Finally, these features are then used to detect objects. R-CNN is slow due to these multiple steps involved in the process.

Next comes Fast R-CNN [6], which passes the entire image to ConvNet generating regions of interest. Also, instead of using different models (as in R-CNN), it uses a single model which extracts features from the regions, classifies them into different classes, and returns the bounding boxes. All these steps happen simultaneously, making it faster than R-CNN. Further, in Faster R-CNN [2], Ren et al. merged a region proposal network (RPN) and Fast R-CNN into a single network. It fixes the problem of selective search by replacing it with Region Proposal Network (RPN). We extract feature maps from the input image using ConvNet and then pass those maps through a RPN which returns object proposals. Finally, these maps are classified and the bounding boxes are predicted.

On the other hand, SSD [3] a single-shot multibox detector for multiple categories, sets a range of default bounding boxes over different aspect ratios and scales for each feature map location, and combines all predictions from multiple feature maps with different resolutions. Thus, SSD is relatively faster and robust to scale variations.

Similar work has been carried out by wang et al. [7], where SSD and YOLOv3 [8] have been explored for leukocyte detection from blood smears. You Only Look Once (YOLO) [9], YOLOv2 [10], YOLOv3 [8] and Single Shot Multibox Detector (SSD) [3] are all one-stage object detector architectures. YOLOv3 is an incremental improvement over YOLO and YOLOv2. YOLOv3 considers object detection as a regression problem from image pixels to spatially separated bounding boxes and associated class probabilities. The approach involves a single deep convolutional neural network (DarkNet 53) that splits the input image into a grid of cells and each cell directly predicts a bounding box and object classification. The result is a large number of candidate bounding boxes that are consolidated into a final prediction by a post-processing step. Zhang et al.[10] proposed a YOLOv3 and image density estimation approach for cell counting from images.

Li et al. [11] trained a RetinaNet [12] model for detecting cellular compenents of urine. RetinaNet, another one-stage detector uses Feature Pyramid Network (FPN) [13] on top of ResNet for constructing a rich multi-scale feature pyramid from the input image. Each level of pyramid is used for detecting objects at a different scale. The backbone architecture is followed by 2 subnetworks for classification and bounding box regressions. RetinaNet introduces Focal Loss [12] to resolve the foreground-background class imbalance problems for Dense Object Detection.

## 3 Dataset Details

The dataset consists of 422 gray-scale images of dimensions (1280,720) which has been separated into training set of 370 samples (including the validation set of 37 samples) and testing set of 52 samples. Initially, we had images of different resolutions mainly of (1280,720) and (800,600). These images are changed to a single resolution of (1280,720) using interpolation techniques, so that we have images of a single dimension which is required for concatenating multiple images into a batch and training them in batches parallely using GPU.

It consists of cells from 4 classes - RBCs, pus cells/WBCs, epithelial cells and cluster of pus cells (c-pus). All the cells in the dataset are annotated with bounding boxes around them using the Microsoft Visual Object Tagging Tool (VoTT). The validation set is taken as 10% of all the training samples. The training and validation set consists of 3183 annotated instances of cells while the testing set has 488 annotated cells.

The RBCs are smaller, rounded and have well-defined sharp boundaries resembling a cup shaped structure. The pus cells are generally rounded and grainy in texture with no well-defined borders. The epithelial cells are irregular in shape, larger in size and generally contains a nucleus. The number of different cells in the data are listed in table 1.

| Cell type | Training Set | Testing Set |
|---|---|---|
| RBCs | 1201 | 150 |
| Pus Cells/WBCs | 947 | 169 |
| Epithelial Cells | 919 | 159 |
| Cluster of Pus Cells | 116 | 10 |
| Total | 3183 | 488 |

**Table 1.** Dataset statistics - Number of different cells in Training and Testing data

## 4 Implementation

This paper focusses on the implementation of YOLOv3 and RetinaNet for classifying the cells from urine sediment images. We extract the sharp frames from videos taken using microscope and then apply the trained deep learning models on the sharp frames to obtain inference about the number and types of various cells present in the sample images.

### 4.1 YOLOv3

We trained the YOLOv3 model using the backbone of darknet-53 consisting of 53 convolutional layers, each followed by batch normalization and Leaky ReLU activation. No form of pooling is used and a convolutional layer with stride 2 is used for downsampling the feature maps. This prevents the loss of low-level features like sharp edges and others which are often attributed to pooling. The prediction is done by using a convolutional layer which uses 1*1 convolutions. We resize the images to shape of (w,h) = (800*448) for training. The input to the model is a batch of images of shape (m, 800, 448, 1) where m is the batch size (taken as 8). The output is a list of bounding boxes with class confidence scores and objectness score.

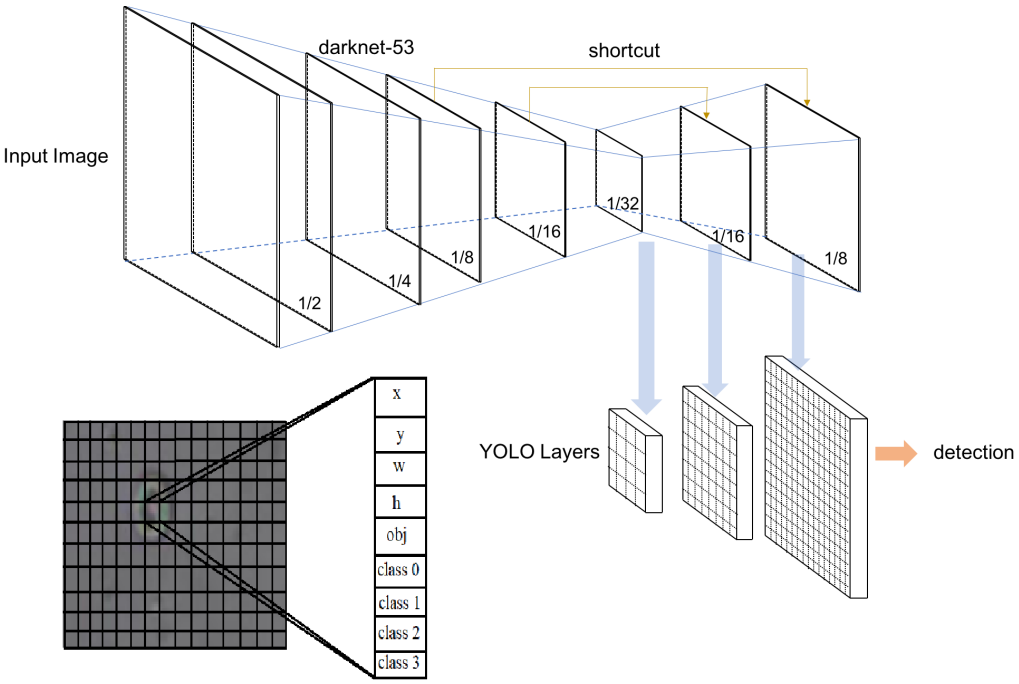


**Fig. 2.** YOLOv3 Architecture and Bounding Box Prediction

The model makes prediction across 3 different scales. The final detection layer receives 3 feature maps of different sizes having strides of 32, 16 and 8. For the input image shape of 800*448, we make detection on scales of 25*14, 50*28, 100*56. For each of these scales, we divide the image into grids of dimensions (25,14) or (50,28) or (100,56). Each grid cell in the image of one scale can predict 3 bounding boxes. So, we have a total of 9 bounding boxes to detect across the 3 different scales. For this, we need to provide 9 anchor shapes to the model for predicting bounding boxes - 3 anchors for each scale. We train the anchor shapes specific to our dataset by K-Means classification which takes the (width,height) dimensions of all the cells in our dataset, predicts the clusters and obtain anchor shape dimensions for each of the 9 cluster of shapes which is then used by the network.

We train the model in 2 set of epochs - the first set of epochs uses the pretrained YOLOv3 weights on ImageNet dataset and train only the final detection layers in our dataset while keeping all the other layers weights frozen. After the detection layer is trained, the second set of epochs train the entire architecture for fine-tuning and reducing the training and validation loss. We used Adam optimizer and a learning rate of 10e-3. We have also used Augmentation (involves resizing the image, flipping image horizontally, distorting the image with change in hue, saturation and value) on the training data to increase data variety and overcome the overfitting issues due to which the model fails to generalize well on unseen images.

### 4.2 RetinaNet

RetinaNet uses feature pyramid networks (FPN) to identify objects at different scales, it uses multi-scale pyramidal hierachy of deep convolutional networks to create feature pyramids. It uses the FPN on top of the ResNet architecture to generate the feature pyramid. Each level of the pyramid is used for detecting objects at a different scale. Since, Pyramids are of different scales, we can use anchors of a single scale in a specific level. RetinaNet uses 9 anchors per level of 3 aspect ratios {1:2, 2:1, 1:1}. For each anchor, there is a length 4 one-hot vector of classification targets and a 4 length vector of box regression targets.

RetinaNet attaches 2 subnetworks to the feature pyramid net, one for classification of anchor boxes and the other for regressing from anchor boxes to ground-truth object boxes. The classification subnet predicts the probability of object presence at each spatial position for each of the anchors and 4 classes. The subnet is a fully convolutional layer having four 3*3 conv layers, each with 256 filters and each followed by ReLU activation, followed by a 3*3 conv layer with 9*4 filters. The box regression subnet is a fully convolutional network to each pyramid level for regressing the offset from each anchor box to a nearby ground-truth object, if one exists. This is similar to the classification subnet except that it gives 4 linear outputs per spatial location per anchor.

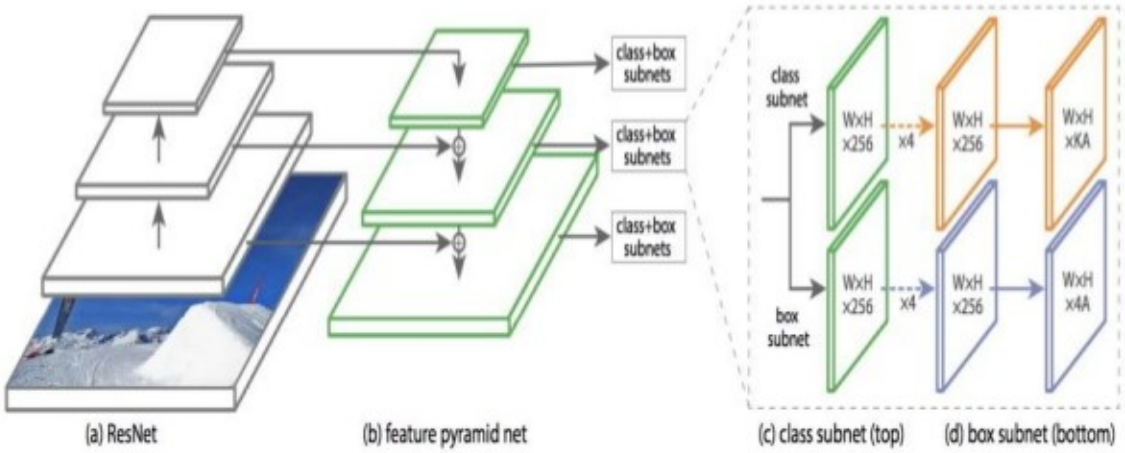


**Fig. 3.** RetinaNet Architecture

It uses the focal loss function [12] which focusses more on training of the hard negative samples and less on the easy samples. It also resolves the issue of class imbalance of number of training examples by multiplying a factor with the loss function. Our dataset has class imbalance issues since the number of training examples for cluster of pus cells is much less compared to the other cell types. We observed that using RetinaNet with focal loss has improved the accuracy of detecting the c-pus class of cells better than YOLOv3.

### 4.3 Model Training and Inference

**Training Settings** We trained 2 models of YOLOv3 - one using only the training data and the other one using augmented training data. We trained 3 variants of RetinaNet models using three different backbone architectures for feature extraction namely ResNet50, ResNet101 and ResNet152. RetinaNet training with augmented data gave almost the same accuracy and has not improved anything and hence, it is not recommended.

**Evaluation Settings** We use a minimum confidence detection value of 25% and Intersection over Union value of 40% with annotated bounding boxes. We denote both the original bounding box and the detected bounding box in the evaluation images for better inference. For individual cells detected on all images, we show the confidence of detection and the IOU value as well.

The average precision of the all the trained models are listed in table 2. On evaluation of YOLOv3 models, we observed that the model trained on augmented data performs better on testing set. The YOLOv3 model without augmentation has overfitted on the training data which reflects in the difference in average precision between training and testing set.

All the RetinaNet models outperform the YOLOv3 Models and gives higher accuracy of 0.783 average precision on the testing set which is pretty good. Among the retinaNet models, the ResNet101 backbone gives the best performance on both the training and testing sets.

| Model | Dataset(Train/Test) | AP-RBC | AP-pus | AP-epithelial | AP-cluster-pus | mAP |
|---|---|---|---|---|---|---|
| YOLOv3 | Training Set | 0.92 | 0.91 | 0.87 | 0.81 | 0.907 |
| | Testing Set | 0.65 | 0.42 | 0.50 | 0.10 | 0.449 |
| YOLOv3 with Augmentation | Training Set | 0.74 | 0.44 | 0.80 | 0.68 | 0.665 |
| | Testing Set | 0.69 | 0.43 | 0.81 | 0.83 | 0.688 |
| RetinaNet with ResNet50 | Training Set | 0.96 | 0.96 | 0.95 | 0.92 | 0.949 |
| | Testing Set | 0.79 | 0.67 | 0.90 | 0.74 | 0.778 |
| RetinaNet with ResNet101 | Training Set | 0.96 | 0.97 | 0.96 | 0.95 | **0.965** |
| | Testing Set | 0.81 | 0.73 | 0.88 | 0.70 | **0.783** |
| RetinaNet with ResNet152 | Training Set | 0.96 | 0.97 | 0.95 | 0.94 | 0.959 |
| | Testing Set | 0.78 | 0.69 | 0.85 | 0.69 | 0.759 |

**Table 2.** Average Precision Values of all the classes of cells and the Mean Average Precision for all the Models

## 5 Conclusion

This paper analyzes the various deep learning models like YOLOv3 and RetinaNet and their performance in detecting and classifying the cells from images of urine sediments. We present the dataset of more than 500 annotated images of urine sediment samples classifying 4 types of cells. We establish through our findings that RetinaNet model trained using ResNet101 is a preferred deep learning model for automated detection and classification of cells.

This work will be continued and there is more scope of research and analysis in the implementation of both YOLOv3 and RetnaNet Models - change in anchor shapes in different scales of YOLOv3, a diiferent evaluation setting, a method for estimating number of cells in cluster of pus cells and others. We are also working on annotating more images taken from a different microscope setting and hence will improve the variety of the dataset. We expect our models to be more robust on adding more data for training.

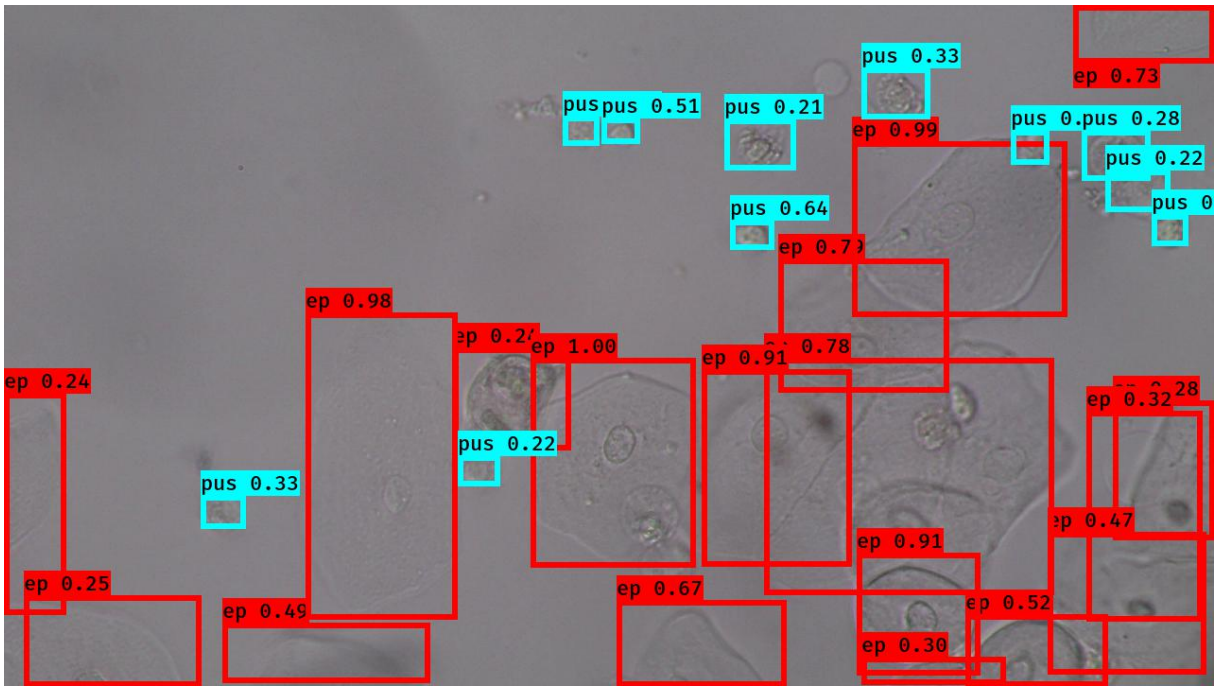


**Fig. 4.** Cell detection with confidence values

## References


1. Yixiong Liang, Rui Kang, Chunyan Lian, and Yuan Mao, "An end-to-end system for automatic urinary particle recognition with convolutional neural network," Journal of medical systems, vol. 42, no. 9, pp. 165, 2018.
2. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. (2015) 91–99
3. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European Conference on Computer Vision, Springer (2016) 21–37
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE confer- ence on computer vision and pattern recognition. (2014) 580–587
5. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeul- ders, A.W.: Selective search for object recognition. International journal of computer vision 104 (2) (2013) 154– 171
6. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448
7. Wang, Q., Bi, S., Sun, M., Wang, Y., Wang, D. and Yang, S., 2019. Deep learning approach to peripheral leukocyte recognition. Plos one, 14(6), p.e0218808.
8. 44. Redmon J, Farhadi A. YOLOv3: An Incremental Improvement. Preprint. Available from: arXiv: 1804.02767v1 Cited 8 Apr 2018.
9. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. Preprint. Available from: arXiv: 1506.02640v5 Cited 9 May 2016.
10. 43. Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. Preprint. Available from: arXiv: 1612.08242v1 Cited 25 Dec 2016.
11. Li, Qiaoliang, Zhigang Yu, Tao Qi, Lei Zheng, Suwen Qi, Zhuoying He, Shiyu Li, and Huimin Guan. "Inspection of visible components in urine based on deep learning." Medical Physics (2020).
12. Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal loss for dense object detection." In Proceedings of the IEEE international conference on computer vision, pp. 2980-2988. 2017.
13. Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. "Feature pyramid networks for object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117-2125. 2017.