One Shot Learning on Perceptual Features for WBC segmentation

Atharva Dubey, BITS PILANI

19 June 2020

Abstract

Segmenting and identifying different bodies from a microscopic snapshot of blood sample is of extreme intererest and has multiple real life application, for example to to quickly get results of blood test and various statistics associated with it. It is of extreme importance to segment and count the cells correctly in an enivronment polluted my noise or foriegn particles. This article proposes a one shot learning approach which makes a decision based on comparision of feature vectors rather than following the traditional classification paradigm.

1 Introduction

With the use of Convolutional Neural Networks(Le cunn et al.)multiple modelshave achieved state of the art results in multiple areas, be it NLP, image classification or segmentation and even on generative models. The biggest caveat of such models are that they require a considerable amount of labelled data which is not feasible in many areas, an example of this being segmenting out WBC from a microscopic snapshot of a blood sample. Traditional classification approaches indentifying the images by generating a probability distribution which is inefficient for a small amount of data because it leads to overfitting. Secondly, addition of an extra class requires retraining the whole model again which is time consuming. One shot learning circumvents these problems, by requiring only one training example per class, facial recognition being a good example of this. Instead of approaching a traditional method of classification, the classification is done on a similarity score by generating a feature vector and calculating a similarity score by using a standard similarity for example cosine similarity, pairwise-distance etc.. [1]

[1] Code, Dataset, Results and model can be found at `https://www.github.com/AD2605/WBC_segmentation`

2 Materials and Methods

CNNs have proved to be a powerful feature extractor and can extract high level features only with a few convolutional layers. Gatys et al[1] showed that the first few layers extract the content, shape and other high level features and the deeper layers of the model extract the features,texture, style and common pattern.

I have used a VGG16 to extract the content and texture features. The first four layers extract the content and shape, and the last four layers, i.e. 16 to 23 is used to construct the grammanian matrix. Let $\emptyset$ denote the pretrained network and C denote the content features and T denote the textural features,

$C = \emptyset(\text{relu2\_2})$

And the textural features of only the last layer have been used –

$$T = \emptyset(relu4\_3)$$

The gram matrix of style representation is given by –

$$G = (\sum\sum \emptyset(I)_{h,w,c}\phi(I)_{h,w,c})/C_j H_j w_j$$

Here - h, w, c denote the height, weight and channels, G denotes the gram matrix and inner and outer sigma run through the width and height respectively and I represents the image.

Once the content and style features are obtained, each is further reduced to a feature to a feature vector of size 2048 and concatenated. This feature vector is passed to a simple ANN to produce a final vector of size 2048.

The network can be summed up by the following set of equations,

Gram Matrix feature –

$$T = \emptyset(relu4\_3)$$

$$G = (\sum\sum \emptyset(I)_{h,w,c}\phi(I)_{h,w,c})/C_j H_j w_j$$

$$G = \text{relu}(G)$$

$$G = \text{maxpool2D}(G)$$

$$G = \text{relu}(G)$$

$$G = \text{maxpool2D}(G)$$

$$G = \text{flatten}(G)$$

$$G = GA^T + b \ldots . \text{ Linear Layer}$$

And generation of Content Vector –

$$C = \emptyset(\text{relu2\_2})$$
$$C = \beta(\ n) + \sum_0^n \omega\,(n) \star x(n) \ldots\ 128 \text{ channels to } 64 \text{ channels}$$
$$C = \beta(\ n) + \sum_0^n \omega\,(n) \star x(n) \ldots\ 64 \text{ channels to } 3 \text{ channels}$$

Final Layers –

$$x = \text{fully\_connected}(F)$$

The loss function used is a contrastive loss (Hadsell et al) function to compare the two feature vectors of the two images, -

$$\mathcal{L} = \frac{1}{2}(1-Y)E_d^2 + Y^{\star}\frac{1}{2}\{\max(0,\ (m - E_d))\}^2$$

Where $\mathcal{L}$ is the contrastive loss Y is a label, it is 1 if image pairs are of different classes or 0 if it is of the same class. E_d denotes the Euclidian distance between the two feature vectors x_1 and $x_{2.}$ This loss is propagated backward.

During inference, a representative sample of the WBC is compared is compared with a generated feature vector. If the vectors are similar, the sub image is labelled as a WBC.

3 Data Preperation and Training Paradigm

Siamese networks take image pairs as input. From the given sample images, just WBCs and not WBCs like RBCs dust particles and dark spots were extracted and 50 image pairs of each label were made. The label is given the value 0 the image pair are of the same class and the value 1 if image pair belongs to different class. During training, epochs were set to 100, optimizer was set to Adam with a learning rate of 1e-5. Training was done on a RTX 2070 with cuda 10.2, 440.31 drivers and a compute capability of 8. The model with the least training loss was saved. The model took 3.5Gb during training and 1.67Gb during inference

4 Results

During inference, the image on which WBCs had be segmented was travelled with a 50x50 kernel and this 50x50 sub-image served as an input to the model, and the final feature vector of the sub image was compared with a feature vector from a known WBC sample. If the absolute difference between the two feature vector(l1 loss) is less than a specified value, the sub image patch is classified as a WBC and a bounding box drawn around it.

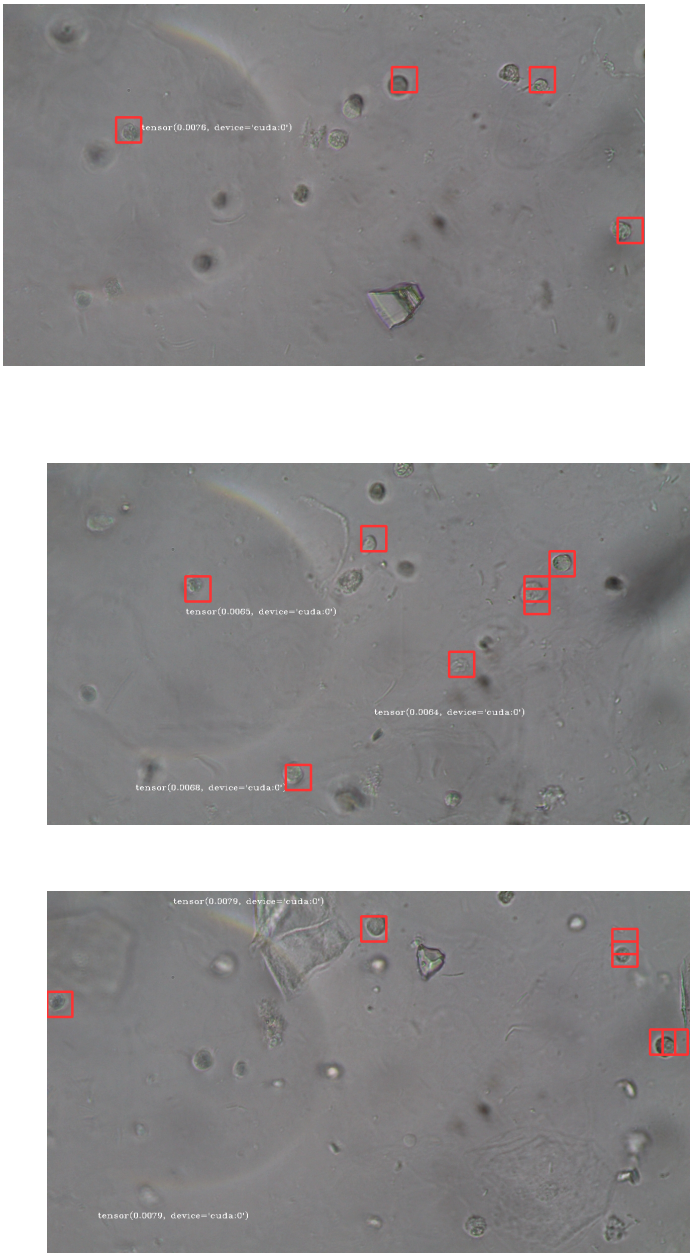

Figure 1: Segmented WBC Images

5 Conclusion

The method discussed takes into account the shape of the WBCs i.e. the content feature vector and the grainy texture of the WBC i.e the texture feature vector. The model learns on the aforementioned vectors and creates a final feature vector, to be compared with a known WBC sample. The model not fine-tuned and performs on raw images without any image preprocessing.