EE 451 - SRE 24/04/2018

Multiple Instance Learning in Histology Images

Guide: Amit Sethi *Name: Rudrajit Das - 140020012*

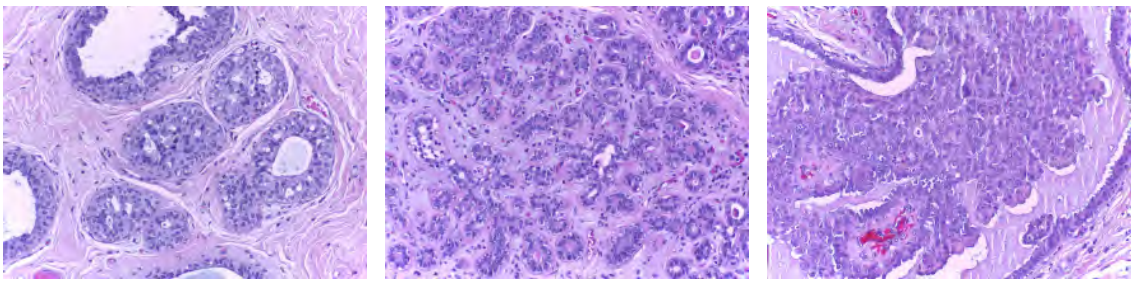
1 Abstract

Today, deep learning is a powerful tool in assisting specialists in analyzing medical images. Specifically for histology images, work has been done for their classification into benign (non-cancerous) or malignant (cancerous) categories. Most of these techniques rely on manual annotation done at the patch level, which significantly improves results. However, this is extremely time consuming and we would like to have this done automatically without the need of any manual intervention. Another practical problem is the lack of large amounts of usable data, on which the performance of most deep learning systems depend. Multiple Instance Learning (MIL) has been proposed to solve this problem of automatic annotation. MIL is an unsupervised learning (some might say semi-supervised) problem which is what makes it particularly challenging. MIL has been studied extensively for natural images. Even in the medical imaging community, MIL has been explored for quite a few applications, but not so much for histology images which are significantly different from natural images. The aim of this SRE is to explore and try out MIL for classification of histology images on 2 relatively small sized data sets. I referred to a few papers for potential approaches, tried out 3 techniques which I myself came up with and got decent results on one of the data sets.

2 Multiple Instance Learning Problem Statement

Consider a bag of N objects with their corresponding labels $(x_i, y_i), 1 \leq i \leq N$. Each object x_i consists of multiple unlabelled instances $x_{i,j}$ with their corresponding labels $y_{i,j}, 1 \leq j \leq M$. The object x_i will have label 1 if there is at least one instance $x_{i,j}$ with label 1, else it will have label 0. For the problem of cancer detection in histology images (objects), the entire image is divided into patches (instances), each of which may either be benign (non-cancerous) or malignant (cancerous) and we do not know which ones are what. We are only given the labels of the entire image, i.e. benign(0) or malignant(1).

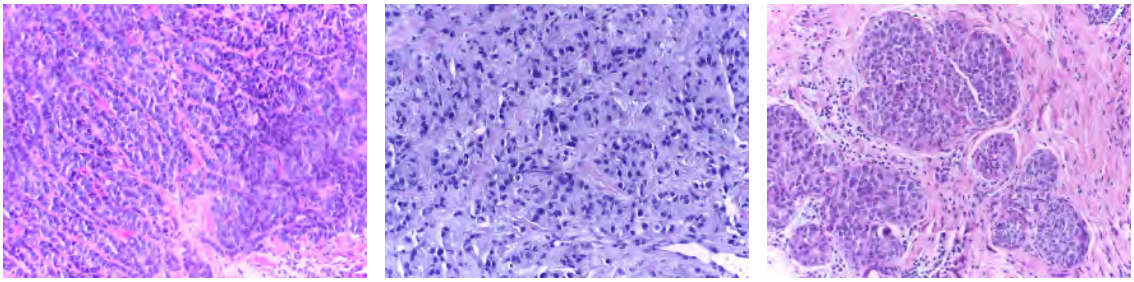
3 Sample Whole Slide Images (WSIs)



(a) 1 (b) 2 (c) 3

Figure 1: Sample Benign WSIs

The rules of thumb for classifying an image as malignant is to look out for large sized nuclei with conspicuous nucleoli or even scattered nuclei. If however, the nuclei are small sized and are arranged around a gland, then most likely the image is benign.



(a) 1 (b) 2 (c) 3

Figure 2: Sample Malignant WSIs

4 Probabilistic Interpretation of MIL

Let $p(y_{i,j} = 1|x_{i,j}) = f(x_{i,j}; \theta)$ where θ is the set of parameters of our model. The function $f(.; \theta)$ could be as simple as logistic regression classifier or as complex as a deep neural network.
Then,

$$p(y_i = 0|x_i) = \prod_j p(y_{i,j} = 0|x_{i,j}) = \prod_j (1 - f(x_{i,j}; \theta)) \tag{1}$$

$$p(y_i = 1|x_i) = 1 - p(y_i = 0|x_i) = 1 - \prod_j (1 - f(x_{i,j}; \theta)) \tag{2}$$

Thus the negative log likelihood function is:

$$\begin{aligned} -log(L(\theta, x_1, x_2, ..., x_N)) &= -\sum_i y_i log(p(y_i = 1|x_i)) + (1 - y_i) log(p(y_i = 0|x_i)) \\ &= -\sum_i y_i log(1 - \prod_j (1 - f(x_{i,j}; \theta))) + (1 - y_i) log(\prod_j (1 - f(x_{i,j}; \theta))) \end{aligned} \tag{3}$$

The above objective function is more commonly known in literature as Noisy-OR. This is not the usual cross entropy loss at the instance level (although it is derived from the cross-entropy loss at the object level). There are several other smooth objective functions to mimic the "max" function such as Generalized mean (GM) or Log Sum Exponent (LSE) but apparently Noisy-OR works better than the others.

5 Primary Reference Papers

I referred in parts to several papers (given in references) but the following are the main ones:

5.1 Bayesian MIL

My main reference paper [1] uses the sigmoid function (logistic regression classifier), i.e. $f(\boldsymbol{x}, \boldsymbol{w}) = \sigma(\boldsymbol{w}^T \boldsymbol{x})$. This paper presents a pure Bayesian approach wherein a Gaussian prior is imposed on the weights $\boldsymbol{w}$ - the weights are asssumed to be uncorrelated and therefore the covariance matrix A is assumed to **diagonal** (and unknown of course). The problem is thereby formulated as :

$$A^* = \arg\max_A p(D|A) = \arg\max_A \int_{\boldsymbol{w}} p(D|\boldsymbol{w}) p(\boldsymbol{w}|A) \boldsymbol{dw}. \tag{4}$$

Note that $p(D|\boldsymbol{w})$ is modelled using the sigmoid function whereas $p(\boldsymbol{w}|A)$ is modelled using a Gaussian with 0 mean and covariance matrix A. A 2 step iterative solution is proposed to solve this problem - iteratively updating w and A.

In the first step, the MAP estimate of $\boldsymbol{w}$ ($\boldsymbol{w_{MAP}}$) is obtained using a fixed A by optimizing the negative log likelihood objective function given in section 4 and a regularization penalty on w imposed in the form of $\|A\boldsymbol{w}\|^2$. **Newton-Raphson** method is used for the optimization (gradient and Hessian are tractable). The convergence criteria for this step is basically the norm of the gradient falling below a (very well chosen) threshold.
In the second step, using the $\boldsymbol{w_{MAP}}$ obtained in the previous step, A is updated. The integral mentioned above (4) is intractable in practice and hence is approximated via a Taylor series expansion centered about $\boldsymbol{w_{MAP}}$. This is followed by a series of manipulations and approximations from which an approximate update rule for A is proposed.
The paper presents a convergence criteria for the overall algorithm, but as I found out, it doesn't work very well as mentioned in section 6.1.

5.2 MIL using CNNs

The main problem in using CNNs for MIL with a sigmoid activation in the last layer is that the optimization becomes hard in the sense that the gradient expressions are ugly and especially become messy when back-propagated.
In [2], the authors propose an ingenious solution to this problem. They essentially construct a CNN (say with weights W) calling the output of the second last layer (they use ReLU) as h for an input x. The output of the last layer (MIL layer) is modelled using an exponential distribution as $p(y = 1|x) = 1 - exp(-h)$.
Using this construction, the negative log likelihood(L) for MIL reduces to :

$$L = -\sum_i y_i log(1 - exp(-\lambda \sum_j h_{i,j})) + (1 - y_i)(-\lambda \sum_j h_{i,j}) \tag{5}$$

The following equations provide the back-prop. recursion for the last layer:

$$dL/dW = \sum_{i,j} (dL/dh_{i,j}).(dh_{i,j}/dW) \tag{6}$$

$$dL/dh_{i,j} = \lambda \text{ if } y_i = 0 \tag{7}$$

$$dL/dh_{i,j} = -\lambda exp(-\sum_j h_{i,j})/(1 - exp(-\sum_j h_{i,j})) \text{if } y_i = 1 \tag{8}$$

The authors test this method on CIFAR10, CIFAR100 and ILSVRC2015 datasets. They obtain appreciable improvements over RESNET for CIFAR10 and CIFAR100 and marginal improvement for ILSVRC2015. All experiments were performed on natural images and so it would be interesting to see how this method does for medical images.

5.3 MIL using Expectation Maximization (EM) Algorithm

In the previous 2 papers, the labels of the instances are not estimated per se and thus we had to work with a different (and harder!) objective function. Another approach for MIL is to try to actually determine the label of each instance using an algorithm like EM and try to fit a Maximum Likelihood model with the normal cross-entropy loss function using these predicted labels, as done in [3]. However, this paper was already tried by the group and I came to know about this mid-way. Apparently, it did not yield good results. However, I liked their idea and have used a similar idea in my experiments.
The labels of the patches (in this paper the labels indicate whether the patch is discriminative or not) are considered as hidden variables for the entire image. In the M-step, they consider the joint distribution $\prod_i P(X_i, H_i)$ where $X_i = < x_{i,1}, y_i >, < x_{i,2}, y_i >, ..., < x_{i,M}, y_i >$ is the entire image and its label (y_i) with $x_{i,j}$ being the j^{th} patch of X_i and similarly $H_i = h_{i,1}, h_{i,2}, ..., h_{i,M}$ is the set of labels

for the M patches of X_i. They show somehow (found this to be a bit sketchy!) that maximizing this is equivalent to maximizing the following likelihood $\prod_{j \in D_i} p(y_i|x_{i,j})$ where D_i is the set of discriminative patches for x_i. A CNN is used to model this likelihood term. This is the M-step.
In the E-step, they set $h_{i,j} = 1$ only if $p(h_{i,j}|x_{i,j})$ is above a certain threshold (again not very convincing!). They estimate $p(h_{i,j}|x_{i,j})$ by applying Gaussian smoothing on $p(y_i|x_{i,j})$ which I interpreted as applying a Markovian structure on the hidden labels. Their problem formulation appears to be similar to an image segmentation problem using Gaussian Mixture Models(GMMs) and Markov Random Fields(MRFs) but on patch level rather than pixel level.

5.4 Other objective functions for MIL

As mentioned earlier, the Noisy-OR objective function works better than all other conventional objective functions which have been tried for MIL. However, the authors in [4] present a different objective function altogether and claim that their objective function works better than Noisy-OR for classification of microscopy images.
They define the following score (calling it Adaptive Noisy-AND pooling function) for an image:

$$P_i = g_i(p_{i,j}) = \frac{\sigma(a(\overline{p}_i - b_i)) - \sigma(-ab_i)}{\sigma(a(1 - b_i)) - \sigma(-ab_i)} \text{ where } \overline{p}_i = \sum_j p_{i,j}/|j| \tag{9}$$

Here, $p_{i,j}$ are the scores for individual patches and can be thought of as the probabilities for the individual patches. Based on these patch based scores, a global score for the entire image is computed using the equation defined above. Also, a is a fixed parameter and b_i's are a set of parameters meant to be learned during training itself.
Although I did not read this paper in great depth, I think their objective function can be tried out for histology images too.

6 Implementation approaches, details and results

The ideal approach would be to write code on Tensorflow or Pytorch using the Noisy-OR objective function being applied over an appropriately constructed CNN architecture. However, I wanted to see whether we can do away with all this hassle by just extracting some "good" features from a separate CNN and use these features to train the Bayesian MIL (section 5.1) system separately. In the following subsections, I briefly mention all the approaches that I thought of on my own, tried and the results obtained.

6.1 Bayesian MIL using features obtained from an auto-encoder

My first idea was to run the Bayesian MIL algorithm with a lower dimensional representation of the images obtained using an auto-encoder. I tried 2 auto-encoder architectures : one being the familiar convolutional auto-encoder [5] architecture and another being a U-Net [6] architecture. I made some changes to the U-Net architecture mainly replacing the binary cross-entropy loss function with mean squared error loss function and decreasing the number of channels across all layers due to memory constraints on my PC. The features extracted were those of the lowest layer of size $8 \times 8 \times 256$ reduced to just a vector of length 256 using max pooling/average pooling, with the latter giving better results.
The first step of the proposed algorithm in [1] works very well, i.e. finding the MAP estimate of $\boldsymbol{w}$ using a fixed A (set to I, i.e. the identity matrix initially). But the second step, i.e. updating A using the w_{MAP} obtained in the previous step, doesn't work as well! Thus, I stopped at the MAP estimation of w using $A = I$.
I coded this algorithm separately in Python as the Newton-Raphson algorithm isn't available in Keras. Also, I tried to optimize the MIL cost function with all gradient (only) based optimization routines

available in Keras, but none of them worked well as the cost function value didn't change much. Using the U-Net architecture, I got **90**% (comparable to the results obtained in [7]) and **77**% test accuracy on **Bisque**[7] (cross-validated) and **Bach** (not cross-validated) data sets respectively. Using the normal convolutional auto-encoder, got **75**% test accuracy on **Bach** (not cross-validated) data set. Finally, I visualized the reconstructed images but they weren't particularly good!

6.2 Bayesian MIL with segmented images obtained from U-Net

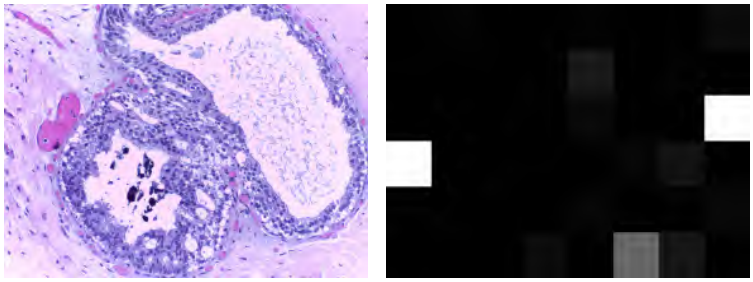
Another approach which I tried is to train a U-Net to segment the images into nuclei/non-nuclei regions and use a sub-sampled version of this segmented map as the input (features) to the Bayesian MIL algorithm. The segmented map was accurately (very nearly) obtained for a subset of the training images using just intensity differences between nuclei and non-nuclei regions and these were used to train the U-Net. Test accuracy obtained was 69% on Bach data set.
I was hoping that the U-Net would learn to segment the nuclei based on their shape and other relevant features but unfortunately due to the poorly chosen training set (for the U-Net) and probably lack of data augmentation, the U-Net also learned to segment images based on just intensity differences. Thus, the U-Net failed to segment many images properly, which in turn led to poor test accuracy.

6.3 Bayesian MIL in conjunction with CNNs

My final idea was to iteratively train a CNN (for classification) with the Bayesian MIL algorithm, each iteration consisting of 2 steps. This was inspired by the EM based approach in section 5.3.
In the the first step, we train a CNN with the labels obtained from the previous iteration and store the features of the second last layer of this trained CNN for every patch. This constitutes the M-step.
In the the second step, use the stored features from the previous step as the input for the Bayesian MIL algorithm and obtain the new labels for the next iteration. This is the E-step. Repeat till convergence.
I tried this approach but ran into several problems. Firstly, the pre-designed network which I was using was over-fitting severely. Also, I realized much later that I wasn't handling the order in which the images are picked up by $flow\_from\_directory()$ properly, which probably led to even worse results. Plan to rectify this and see how it works.

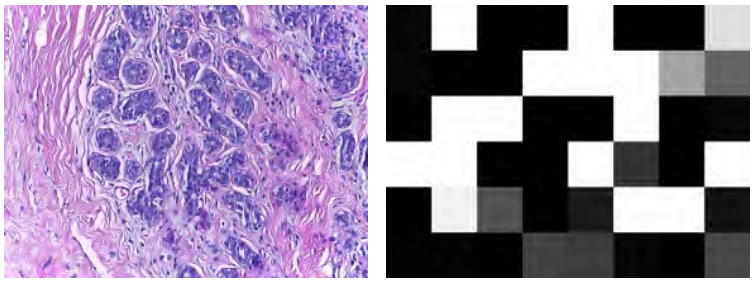
7 Visualization of high probability patches

A heat map helps to visualize the predicted regions of high probability (of being malignant) in an image. The patches which have a lighter intensity correspond to high probability, whereas those having darker intensity correspond to low probability.
Figure 3 shows the heat map for a correctly classified benign image. As expected, there are very few (2 to be precise) white patches (corresponding to high probability) in the heat map. However, the left side white patch seems to be incorrect as there are hardly any nuclei there.
Figure 4 shows the heat map for an incorrectly classified benign image. There are many white patches in the heat map due to which it is incorrectly classified. Some of these white patches seem legitimate but some are again wrong in this as well.
Figure 5 shows the heat map for a correctly classified malignant image. As expected, there are several white patches in the heat map due to which it is correctly classified. But again some of the white patches seem wrong.
Figure 6 shows the heat map for an incorrectly classified malignant image. This is actually a tough one as only a small portion of the image consists of nuclei (which is what can only contribute to high probability of being malignant). Unfortunately, most of the nuclei portion is missed by our classifier, as can be seen from the heat map.



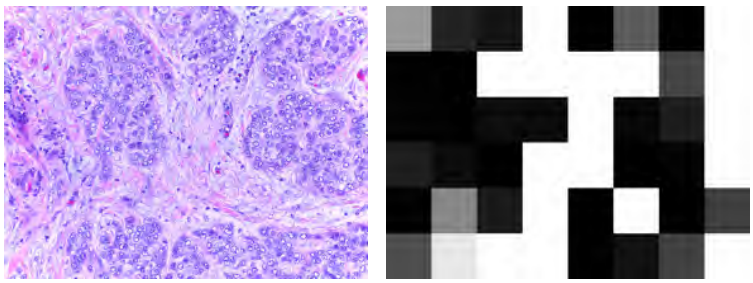
(a) Correctly classified Benign image (b) Heat map

Figure 3: Heat map for a correctly classified Benign image



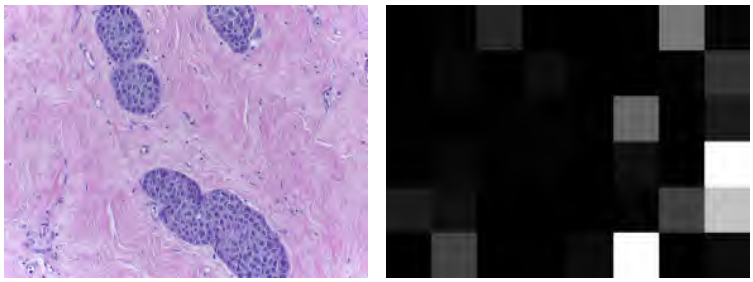
(a) Incorrectly classified Benign image (b) Heat map

Figure 4: Heat map for an incorrectly classified Benign image



(a) Correctly classified Malignant image (b) Heat map

Figure 5: Heat map for a correctly classified Malignant image



(a) Incorrectly classified Malignant image (b) Heat map

Figure 6: Heat map for an incorrectly classified Malignant image

8 Conclusions

- The MIL objective function could not be optimized with only gradient based algorithms. Instead, the Newton-Raphson method proposed in [1] (involving the Hessian in addition to the gradient) works well.
- Observed that the MIL objective function is highly non-convex. Also, the Newton-Raphson method is susceptible to high condition number of the Hessian (ill-conditioned) in many cases, leading to NaN values.
- Need CNNs that are more perceptive to/have the ability to distinguish between different sized nuclei which is critical to the classification of benign vs. malignant.
- Also something like Graph CNNs could be potentially useful because of their ability to learn to incorporate structure - for our problem, scattered or unscattered nuclei.
- Need to take care of over-fitting which is very common for the MIL problem.

9 Further Work

- Found out a very crude implementation of the Newton-Raphson algorithm for Tensorflow.
- Build on the existing code to create a CNN system with the MIL objective function being optimized by the Newton-Raphson algorithm.
- Look out for CNNs with the properties mentioned in section 8.
- Possibly explore and try out other objective functions like the one suggested in [4].

References

[1] Raykar, Vikas C., et al. "Bayesian multiple instance learning: automatic feature selection and inductive transfer." Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[2] Sun, Miao, et al. "Multiple instance learning convolutional neural networks for object recognition." Pattern Recognition (ICPR), 2016 23rd International Conference on. IEEE, 2016.

[3] Hou, Le, et al. "Efficient multiple instance convolutional neural networks for gigapixel resolution image classification." arXiv preprint (2015).

[4] Kraus, O. Z., L. J. Ba, and B. Frey. "Classifying and Segmenting Microscopy Images Using Convolutional Multiple Instance Learning. arXiv preprint." arXiv preprint arXiv:1511.05286 (2015).

[5] https://blog.keras.io/building-autoencoders-in-keras.html

[6] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[7] Tomczak, Jakub M., Maximilian Ilse, and Max Welling. "Deep Learning with Permutation-invariant Operator for Multi-instance Histopathology Classification." arXiv preprint arXiv:1712.00310 (2017).

[8] Xu, Yan, et al. "Deep learning of feature representation with multiple instance learning for medical image analysis." Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. IEEE, 2014.