IMPROVING DEEP ENSEMBLE BAYESIAN INFERENCE: A POLLING APPROACH

*Landon Clark**, *Nathan Russell*†

*University of Kentucky, landon.clark@uky.edu †University of Kentucky, nathan.russell@uky.edu.

ABSTRACT

Diabetic retinopathy (DR) can lead to vision loss and in severe cases blindness. DR is one of the numerous consequences of diabetes and represents the most common cause of blindness for middle aged adults in the United States. DR can only be diagnosed by an eye care specialist. However, machine-learning methods are effective at providing DR detection. The application of machine-learning to DR identification is a popular area of research. This is true to the extent competitions have utilized retina image data sets and DR detection as a metric for model effectiveness. One of the most successful methods for Bayesian inference has been Deep Ensemble Bayesian inference. While this approach is simple and effective, the methods for combining the ensemble network outputs is counterproductive to the goals of Bayesian inference, typically hiding model uncertainty even when present. We present a polling based method that takes into account the predictions of all of our ensemble networks to make classification predictions for each label. Our results show that we retain effective network uncertainty estimates while also improving network prediction accuracy. Our code can be found at [1].

Index Terms— Diabetic Retinopathy, Deep Ensemble, Deep Learning, Bayesian Inference, Regularization

1. INTRODUCTION

The NeurIPS 2021 challenge on Approximate Inference in Bayesian Deep Learning aimed to compare various deep learning models against a Hamiltonian Monte Carlo (HMC) method. Submissions for the competition began July 15 2021 and submissions were closed on November 26 2021. The competition had two tracks, specifically the *light track* and the *extended track*. In both tracks, contestants had to utilize a diabetic retinopathy image data set for DR detection. The competition is broader than just DR detection. However, a large and publicly available set of retina images is available for analysis by the contestants. The competition used this data set as one of the basic benchmarks for testing model accuracy. There are other competitions that are dedicated to DR detection such as the *Kaggle DR Detection Challenge*.

1.1. Applications

Only licensed and certified eye specialists can give a DR diagnosis. However, early detection using quick imaging would be useful. Mobile devices and apps have provided great insight for consumers and health care professionals. Virtual visits have become a norm in the COVID era. However, consumer grade medical devices still serve patients well between doctor visits. Mobile devices can monitor diabetic glucose levels and even detect abnormal heart beat patterns. According to the NIH, more than 60 percent of physicians utilized data from mobile devices in 2014 [2]. This data ranged from surveys and general communication using apps to Bluetooth enabled wearable devices. Symptoms of diabetes range from numb feelings to changes in the retina. According to the CDC, 88 million adults, which translates to more than 1 in 3 US adults, have prediabetes and 84 percent are unaware of the condition [3]. Deep learning models may allow consumers to upload retina images captured from cell phones to potentially diagnose medical issues like diabetes.

Insurance companies may not be able to provide an official diagnosis for medical condition, but they can make assumptions. In fact they routinely make assumptions about undiagnosed conditions. Health insurance companies often offer incentives to members that get regular checkups or use wearable activity tracking devices. This data allows the insurance companies to apply models to detect medical conditions that have yet to be diagnosed. Members of health plans are usually categorized into groups. These groups are often associated with large employers, unions, or industries. Accurately detecting undiagnosed medical conditions of individuals can help provide risk estimates for the groups. Health insurance companies develop bids based on these estimates to compete for new client groups. It is common for people to suffer the effects of diabetes before an official diagnosis is given. It is often the case that advanced conditions create symptoms that can not be ignored due to severity. Health insurance companies can offer incentives for members to upload retina images on regular intervals. Employers could even negotiate lower rates to include contractual agreements for this data. Large insurance corporations with deep pockets, coupled with an incentive to keep members health, could develop sophisticated models that are used in early detection, perhaps well in-advance of severe symptoms.

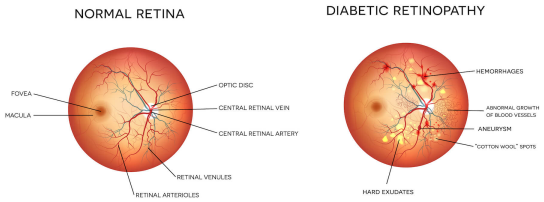

Fig. 1. Normal Vs DR Retina

1.2. Diagnosis of Diabetic Retinopathy

Diabetic Retinopathy (DR) is typically a preventable condition that impacts vision. Blood vessels toward the back of the eye, near the pupil, are damaged resulting in symptoms that include blurriness, dark areas of vision, and difficulty perceiving colors. In severe cases DR can cause blindness. The condition is chronic and is usually a lifelong condition. An accurate diagnosis will require imaging, but advanced cases are usually obvious. Laser based treatments and surgeries are possible, but the effectiveness is limited.

The damaged blood vessels near the retina are usually the result of poorly controlled blood sugar levels. Figure 1 demonstrates the difference between normal blood vessels near the retina and blood vessels associated with DR. A physician will look for hemorrhages, regions known as *cotton wool* spots, and hard exudates in the eye to diagnose DR.

1.3. Deep Learning Approach

A physician will take careful observations when diagnosing DR. Literature research implies proliferative DR is easy to identify, but mild non-proliferative cases are easy to misdiagnose [4]. Deep learning models are more than capable of such obvious detections. Given a large data set of retina images, properly labelled whether DR has been diagnosed for each image, the data set can be divided into two categories. The first category is the training set and the second category is the test set. The training set is used to train the deep learning model. After the model has been trained, the model is applied to the test set to determine accuracy of the model. Larger data sets often yield better results at the cost of time training the models.

The publicly available DR data set used in the competitions are a collection of 224×224 sized images. They are each labelled using one of five categories. The categories and associated image counts are as follows:

1. **No DR** - 1805 images categorized as no DR diagnosis.
2. **Mild** - 370 images categorized as mild DR.
3. **Moderate** - 999 images categorized as moderate DR.
4. **Proliferate** - 193 images categorized as proliferate DR.
5. **Severe** - 295 images categorized as severe DR.

Our experimental attempts using the image sets for our own models involved predicting the probability of any of these classes, trained on a one-hot-encoded vector corresponding to the DR severity.

2. LITERATURE REVIEW

Our project required significant literature review to come up to speed on the topic of diabetic retinopathy. Most of the general information about DR is publicly available on websites maintained by various universities and government agencies. Most of that information is uniform across varon ious reliable sources. However, modeling DR using deep learning methods varied significantly across publications. Hamiltonian Monte Carlo (HMC) models were frequently used as the benchmark for comparison. Many of the publications concluded that HMC models were the best performing. Our approach

2.1. General Detection Reliability Beyond Color

There were initial concerns about mispredicted labels due to variations in DR presentations. However, a publication dedicated to deep learning DR detection provided some ease here. The research explains that training models using general image sets are stable since the natural features of DR are consistent between different people [5]. With respect to training models, this is an important and fortunate fact. Labeled images can lead a person to believe the signature of DR is red eyes due to blood vessel damage. However, deep learning models should be able to identify much more than simply color hues.

2.2. Reliability of Deep Ensemble Methods

One of the shared properties between classical and Bayesian statistics is the power of large data sets. As the number of observations grow, the variations caused by certain parameters become more detectable. Deep Ensemble methods are essentially the aggregated results of other models. The weaknesses of one model are offset by the strengths of another model. When strengths overlap, the result tends to be a strong observable instance. Models often share the same weaknesses. Significant weaknesses in models are usually the result of a computational hurdle that is hard to overcome, estimate, or even identify regardless of model. As a result, significant weaknesses can be uniform across models and perhaps not independently offset certain strengths.

There are several peer-reviewed publications that provide solid footing for the reliability of ensemble methods. Seemingly unrelated theories regarding bias-variance are cited to support the effectiveness of ensemble methods [6]. We found

an exhaustive state-of-the-art style publication that cites over 30 popular peer-reviewed papers outlining various theoretical supports for the efficiency of ensemble methods [7].

2.3. DR Specific Deep Ensemble Research

Deep learning models have the ability to identify combinations of attributes not obvious to humans. However, relying on such models without a benchmark can fall short with respect to confidence. As mentioned earlier, proliferative DR instances are usually not hard to identify. We were able to find at least 15 peer-reviewed papers that have analyzed popular and publicly available DR image sets. One such paper from the journal of *Informatics in Medicine Unlocked* created subcategories that are more specific than the 5 general DR categories our test image set uses [4]. This paper differentiated non-proliferative spots as microaneurysms(MA) and Haemorrhages (HM). Images falling into the MA category have red dots that are the result of weakened blood vessel walls. Images falling into the HM category have larger red dots and with an irregular margin. The referenced paper also differentiated between hard and soft exudates. Our goal was to see if images our models failed to detect could be explained by the refined categories. The referenced paper did not utilize deep ensemble but provided refined results and image examples for strong benchmark models.

An *IEEE Access* paper authored by several authors provided some good insights and benchmarks for estimating accuracy [8]. This paper was a valuable resource since it provided benchmarks for popular models, a general algorithm for analysis, and formulas for estimating sensitivity and accuracy.

3. METHODS

Traditionally the deep ensemble method is not considered a Bayesian inference technique, however this problem is remedied by means of MC Dropout. [9] proves that MC Dropout is a sufficiently accurate form of approximate Bayesian inference. Instead of using dropout as a form of regularization alone, we use it at inference time to sample over our estimated posterior $\pi(\theta|D)$ where θ is our model parameters and D is the training data. The frequentist approach to deep networks would be satisfied using dropout to find an optimal posterior, then performing inference on this set of network parameters. The Bayesian approach considers our posterior estimation to be part of a complex distribution which we would like to sample over. This is simply accomplished by continuing to use dropout on our trained models during inference time. Now that we have a concept of the posterior distribution of each of our models, we need to consider what happens when we combine the network outputs of our ensemble. The traditional approach to combining network outputs is to simply take the mean of our model predictions for each class. Our ensemble posterior is then $\pi_E(\theta_M|D) = \frac{1}{L}\sum_l^L \pi_l(\theta_l|D)$, where L is the total count of networks in our ensemble. The general idea of this approach is to mitigate inaccurate network predictions by combining them with more accurate predictions. This practice introduces two problems for use in Bayesian inference. For one, we would ideally weight more accurate networks higher then less accurate networks. We also see that the uncertainty of each network would roughly be scaled by $\frac{1}{L}$. While decreasing uncertainty may seem like a good thing, we end up losing uncertainty estimates for cases when one network is chosen above others (this case naturally implying we are uncertain of our predictions). To remedy these issues, we propose a polling based approach – where the individual network outputs are combined with a matrix $B \in \mathbb{R}^{C,L\times C}$, where C is the number of classes in our dataset. This matrix is applied after the ensemble has been trained and optimized with respect to the outputs of models, x_E (we use an x here to denote an input into our polling matrix, although it is comprised of network outputs). Before we begin discussing the techniques we used to train the polling matrix B, we will first introduce a couple of useful decompositions used throughout the remainder of the paper:

$$B = \begin{bmatrix} O_1 & O_2 & \cdots & O_L \end{bmatrix}$$

$$x_E = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{bmatrix}$$

where O_l represents the linear dependencies of predicting each class based on the model l outputs, and $x_l \in \mathbb{R}^{C,1}$ is the model l output. The following sections will explain our training and regularization techniques.

3.1. Training

To train our matrix, we used the PyTorch library and several interesting features therein. The most important feature by far is the automatic differentiation feature – made convenient with operator overloading [10]. The training was similar to that of a neural network - we performed forward passes given $X_E = \{x_E^{(i)}|i = 1,\ldots,N\}$ for N data points, and back-propagated our losses. With automatic differentiation, we were able to define a forward pass simply as $\hat{y} = Bx_E$. A naive approach might leave the gradient calculations on for the networks, which will lead to complex and poorly defined prediction behavior as our loss is back-propagated through the polling matrix into the individual models. To avoid this, we simply switch the PyTorch context to no gradient computations when predicting x_E with the instruction:

```
with torch.no_grad():
```

To compute the loss of our combined prediction, we use categorical cross-entropy, the same loss function used to calculate

loss for the individual networks. This gives us the equation

$$\mathcal{L}(B) = -\frac{1}{N}\sum_{i}^{N}\sum_{c}^{C}\log softmax(Bx_E, c)\cdot y_c^{(i)}$$

$$softmax(x, i) = \frac{e^{x_i}}{\sum_c^C e^{x_c}}$$

where y is the true label for a given input image. In reality the loss function is relative to y, but since our training set and model predictions are fixed, we view this more as a feature extraction problem than a purely supervised learning problem. As our optimizer, we use AdamW, which is the traditional Adam algorithm with weight decay enabled. This helps to keep our model sparse (we used the same optimizer for the underlying networks as well). The weight decay factor was set to $\lambda = 0.01$ and the learning rate was $\alpha = 0.0005$ for 40 epochs.

3.2. Regularization

One important note to make is that during training we do not actually use Bx_E as our polling output. One downside to categorical cross-entropy is that overly confident incorrect predictions can result in massive losses. Originally we saw losses anywhere from 10^{20} to 10^{30}. To reduce the confidence of our predictions, we applied the tanh function to Bx_E, thus making our output

$$\hat{y} = tanh(Bx_E)$$

which reduced our losses to somewhere typically between 8 and 3. However, at inference time the tanh function causes severe underestimations of prediction confidences. It will also cause the values in B to grow very large as our inputs to the softmax term in our loss function will be bounded between -1 and 1, causing consistent losses even when we are most confident in the correct label. To mitigate these issues we add two terms to our loss function. The first is a standard L2-regularization term. The second is a rather unique idea – we subtract the absolute values of the determinants of our O_l submatrices from the loss function. Our loss function now looks like:

$$\mathcal{L}(B) = -CCE(B) + \eta L2(B) - \rho(abs(|O_1|) + \cdots + abs(|O_L|))$$

The L2-regularization is straightforward – we would like to keep individual values in B from being too large, causing over-confident predictions when we remove the tanh function at inference time. However, the determinant loss may not be as straightforward. Let us consider what each of our O_l submatrices represent. A single class output $\hat{y_c} = \sum_l^L O_{l,c}x_l$ where $O_{l,c}$ represents the c^th row of O_l. Each row of O_l then represents the confidence in class c being the correct label given the output of model l. Of course then the ideal O_l would be something similar to the identity matrix, or a matrix with positive diagonals and negative entries elsewhere. However, this is where the feature extraction view is important. While training, it is almost guaranteed that any systematic model mispredictions will affect the corresponding O_l, resulting in positive values not along the diagonals. By analyzing each O_l we can determine what each model uncertainty looks like for each class, but we have to consider how this will affect our predictions. For instance, given that we are looking to find our confidence in class i when class j predictions weigh heavily (weight notably greater than zero) into this prediction – we will likely hardly ever have high confidences in our class i predictions. As we are sampling along our models' posteriors, we want to take advantage of the fact that our class i and class j confidences will vary accordingly, so we lose some information about our uncertainty if we consider these classes heavily interdependent then high confidences (again, sampled along our posterior distribution so not static for a given input) will average out between the two classes. Our solution then is to maximize our determinant without completely removing the interdependence between classes. Thus, not only do we keep ρ small - we also normalize the rows of O_l before taking the determinant. Thus, we keep this term from taking over our loss function and becoming the primary optimization objective.

4. RESULTS

For our specific problem $C = 5$ and $L = 5$. Our 5 models were all Resnet18 CNN's, initialized with the following initializers:

1. Xavier Uniform
2. Xavier Normal
3. Kaiming Uniform
4. Kaiming Normal
5. Orthogonal

Figure 2 shows the loss before each back-propagation. Given the relatively small parameter count of these networks, there is not much surprising about their lack of convergence during training, especially when only updated about 320 times. The initial learning rate began at $\alpha = 0.001$ and was updated as $\alpha = \alpha \cdot 0.9$ every 5 epochs. The dropout rate was set to 0.25 for both training and inference.

With respect to our polling matrices, we ended up computing two separate versions - one highly regularized and one with no regularization then a low regularization level on a second training sequence. The highly regularized model was trained with $\eta = 0.001$ and $\rho = 0.05$ - our other model was regularized with values of $\eta = 0.0001$ and $\rho = 0.01$. The regularized training results can be seen in figure 3. One interesting note is that with the heavy regularization, O_5 is the only model that begins with a positive determinant value,

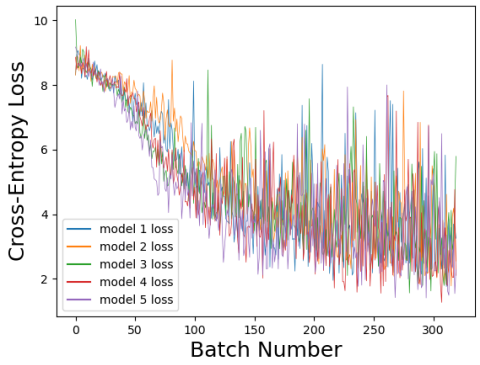

Fig. 2. Loss of our ensemble models during training.

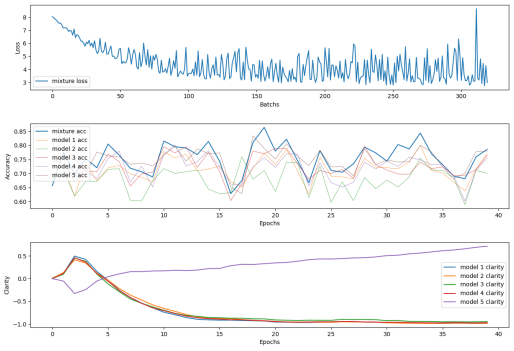

Fig. 3. Reg Mixed Train

with it then flipping signs as training goes on. For the non-regulated training, all of the determinants started out positive and became negative over time. As seen in figure 5, model 5 is the most accurate individual network - so perhaps we can view the over-regularized polling matrix as picking a favorite network that it tends to believe more than other networks.

Figure 4 illustrates the difference in uncertainty predictions between the different forms of combining ensemble networks. Inference was performed on 200 images, with 150 samples predicted for each each image. The figure displays predictions for confidence in selecting the correct class - where 40 out of the 200 images belong to each class. The following table summarizes the results from this testing:

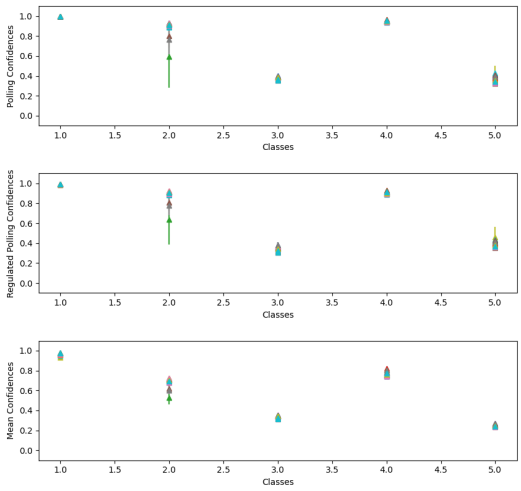

Fig. 4. Bayesian inference results. Long bars represent variances and triangles represent prediction means.

Combin. and Class	Minimum Mean	Maximum Var
Poll, No DR	0.992	$2.2{\cdot}10^{-9}$
Poll, Mild DR	0.589	0.292
Poll, Mod. DR	0.352	$3.1{\cdot}10^{-3}$
Poll, Severe DR	0.937	$8.2{\cdot}10^{-7}$
Poll, Prolif. DR	0.309	0.091
Poll Reg., No DR	0.987	$4.1{\cdot}10^{-8}$
Poll Reg., Mild DR	0.612	0.201
Poll Reg., Mod. DR	0.319	$8.9{\cdot}10^{-2}$
Poll Reg., Severe DR	0.922	$6.5{\cdot}10^{-6}$
Poll Reg., Prolif. DR	0.332	0.122
Mean, No DR	0.914	$7.7{\cdot}10^{-7}$
Mean, Mild DR	0.509	0.081
Mean, Mod. DR	0.341	$2.6{\cdot}10^{-6}$
Mean, Severe DR	0.889	$2.5{\cdot}10^{-8}$
Mean, Prolif. DR	0.224	$4.4{\cdot}10^{-4}$

As we can see from the previous table, poor predictions result in higher uncertainty for the polling method, while the mean combination method has relatively high certainty even when the predictions are wrong. As figure 5 shows, the minimally-regularized polling method beats the prediction accuracy over 100 epochs of 80 (16 of each class) images per epoch by roughly 0.7% (77.3% vs. 76.6%).

Our final value for B^T is the following matrix (transposed for convenience):

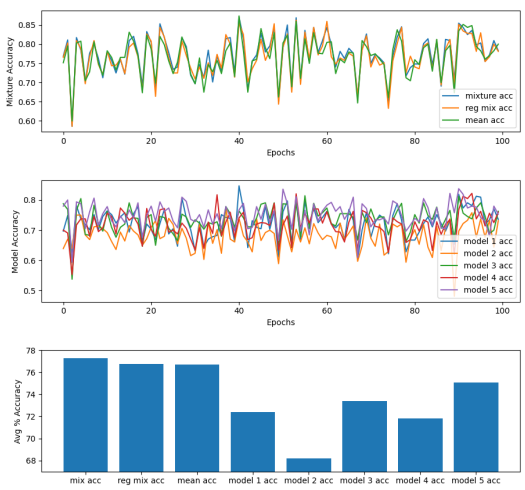

Fig. 5. Comparison of prediction accuracy from epoch to epoch and total accuracy over all epochs.

$$\begin{bmatrix}
0.9659 & -0.7911 & -0.9025 & -0.2986 & -0.8327 \\
-1.0747 & 0.7529 & -0.6991 & -0.6035 & -0.3834 \\
-1.1228 & -0.7584 & 0.7663 & -0.5564 & 0.2205 \\
-0.9983 & -0.3947 & -0.5062 & 0.6544 & -0.1551 \\
-1.1184 & -0.3298 & 0.1754 & -0.4990 & 0.6056 \\
1.0092 & -0.8142 & -0.8750 & -0.4266 & -0.8410 \\
-1.0535 & 0.6043 & -0.5731 & -0.5064 & -0.1610 \\
-1.0991 & -0.7496 & 0.8458 & -0.6061 & -0.0820 \\
-0.9913 & -0.5299 & -0.6225 & 0.7320 & -0.2644 \\
-1.1168 & -0.5843 & 0.4446 & -0.5300 & 0.6342 \\
1.2326 & -1.0014 & -1.0378 & -0.9649 & -0.9391 \\
-0.9252 & 0.7265 & -0.8168 & -0.4781 & -0.4598 \\
-1.1079 & -0.7719 & 0.6858 & -0.4313 & 0.2862 \\
-0.7285 & -0.4559 & -0.7423 & 0.8446 & -0.3821 \\
-1.0622 & -0.4421 & 0.0768 & -0.2149 & 0.6058 \\
1.2487 & -0.9274 & -0.9896 & -0.9119 & -0.7768 \\
-0.9532 & 0.7116 & -0.6898 & -0.5350 & -0.4108 \\
-1.0981 & -0.8448 & 0.8612 & -0.6900 & 0.0520 \\
-0.7733 & -0.4862 & -0.4534 & 0.7045 & -0.1721 \\
-1.1001 & -0.5845 & 0.3511 & -0.5651 & 0.7345 \\
1.2004 & -0.9147 & -1.0642 & -0.9373 & -0.8288 \\
-1.0696 & 0.8820 & -0.9005 & -0.6420 & -0.7243 \\
-1.1390 & -0.8972 & 0.7186 & -0.4057 & 0.2848 \\
-0.9144 & -0.5613 & -0.7888 & 0.9165 & -0.4315 \\
-1.1225 & -0.4885 & 0.0185 & -0.3120 & 0.7596
\end{bmatrix}$$

5. CONCLUSION AND FUTURE WORKS

As our results suggest, our polling method is not only an adequate method for overall accuracy, but it retains model uncertainty much better than the mean combination approach. While it performs well in these areas, another region of use is the analysis of Bayesian networks. By analyzing the structure of our polling matrix, we can decipher whether or not our network(s) have some interdependencies between class predictions. This is as simple as computing the determinant of one of our row-normalizedO_l sub-matrices - the closer to 1 the less dependent our predictions are.

While our method proved successful, there is certainly room for improvement. the standard tanh function likely damps our confidences too much. An improved approach might be to initially use the tanh function while B has not been trained at all, and later remove it once we are more accurately predicting. A clear next step would be to add another simple neural network to act as our polling matrix, allowing for nonlinear relationships between our predicted confidences. However, our matrix provides a well-defined framework for relating our predictions and determining how the networks in our ensemble are related to one another.

6. DIVISION OF LABOR

Various parts of the project were handled by different team members. This section contains a general overview of the contributions of each team member and combined efforts.

6.1. Project Contributions

The project proposal and general direction were both developed together. Our decision to pursue deep ensemble methods was motivated by the availability of benchmarks and development libraries. The literature review was a joint effort where each team member found relevant papers and shared their content with each other. After a few relevant publications were shared, we as a group sifted through the various components of each publication. In some instances the research was more advanced or required more ground-level development than time permitted. In other cases, we felt publications were in the same thread as our own ideas.

6.2. Landon Clark

Landon performed the work associated with generating the final output of our models. This included separating training images from test images. Landon also generated the output plots to provide visual interpretations for our analysis.

6.3. Nathan Russell

Nathan compiled the various components of the analysis into the report. The report was generated using Latex and so a general template was used. Each team member gathered results and individual components of the report and Nathan compiled them together.

7. ACKNOWLEDGEMENTS

We would like to thank Dr. Samson Cheung for his help and guidance during our class. Bayesian methods continually prove to be complex and hard to understand - but Dr. Cheung has made things understandable nonetheless. His patience and availability have been largely beneficial to our personal success.

8. REFERENCES

[1] N. Russell L. Clark, "Bayesian ensemble polling," *https://github.com/landonclark97/BayesianEnsemblePolling*.

[2] C. Ventola, "Mobile devices and apps for health care professionals: uses and benefits," *P & T*, vol. 39, pp. 356–364, May 2014.

[3] "What is diabetes?," November 16 2021, https://www.cdc.gov/diabetes/basics/diabetes.html. Retrieved December 11 2021.

[4] M. Abulkhair W. Alyoubi, W. Shalash, "Diabetic retinopathy detection through deep learning techniques: A review," *Informatics in Medicine Unlocked*, vol. 20, 2020.

[5] D. Spodarets B. Tymchenko, P. Marchenko, "Deep learning approach to diabetic retinopathy detection," *arXiv*, Mar. 2020.

[6] D. Wolpert, "On bias plus variance," *Neural Computation*, vol. 9, pp. 1211–1243, 1997.

[7] M. Tanveer P. Suganthan M. Ganaie, M. Hu, "Ensemble deep learning: A review," *arXiv*, April 2021.

[8] S.Shah A.Khan S.Shamshirband Z.Rehman I.Khan W.Jadoon S.Qummar, F.Khan, "A deep learning ensemble approach for diabetic retinopathy detection," *IEEE Access*, vol. 7, pp. 150530 – 150539, 2019.

[9] Z. Ghahramani G. Yarin, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," *arXiv*, Oct. 2015.

[10] D. Paszke et all., "Pytorch: An imperative style, high-performance deep learning library," *arXiv*, Dec. 2019.