Proceedings of 10th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2021 Month: October Volume: 10

Uncertainty Estimation in Detecting Knee Abnormalities on MRI using Bayesian Deep Learning

Pankaj Dhakal [a], Sashidhar Ram Joshi [b]

[a, b] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, Tribhuvan University, Nepal*
Corresponding Email: [a] 074mscsk007.pankaj@pcampus.edu.np, [b] srjoshi@ioe.edu.np

Abstract
CNN is mostly used to detect knee injuries because of its high accuracy which can extract features automatically but has a millions of parameters and is often subjected to overconfident results. This research work introduces distribution in weights in the last three convolutional operations which acts as an ensemble of networks, and uncertainty is calculated from the mean and variance from the prediction of the multiple networks using variational inference. We have used multipath Bayesian CNN to extract features from MRI of the knee from three different planes; axial, coronal and sagittal to detect multi-label abnormalities. The Bayesian CNN model achieved the result comparable to the pretrained model and slightly better than Montecarlo methods. When both Bien et al. Sagittal plane and Stajduhar et al. training datasets are combined to detect ACL label of the sagittal plane, the AUC score increased to 0.917 for the Bayesian CNN model, where the state of the art for the similar case is 0.911. The uncertain images from training data of Sagittal plane are removed and tested on Stajduhar et al. dataset using Alexnet pretrained model to increase the AUC of the test datasets.

Keywords
CNN, MRI, Bayesian CNN, Montecarlo, AUC, Uncertainty

1. Introduction

Deep Learning has revolutionized computer vision tasks like classification, recognition, etc. It has also entered the field of medical images rapidly. It has also reduced the workload of physicians, radiologists by assisting them in classification and detection tasks. Although deep learning has been a solution for different medical applications, it does not quantify risk or uncertainties. It cannot predict how a model is confident about each prediction. Some images are difficult to diagnose even for a trained model. When predicting abnormalities on sensitive areas, we should not always rely on accuracy but should also look for confidence value. We have to look not only on prediction but also on uncertainty. Relying on prediction alone can be dangerous. If somehow a model can predict how certain it is about each prediction, then it can help physicians and radiologists greatly. For an uncertain prediction, physicians and radiologists can consult experts in related fields or take different measures to diagnose an injury or a disease. One such uncertainty estimation is by introducing probabilistic modeling of deep neural networks. In such cases, weights in neural networks are assigned a probability distribution. By introducing distribution in weights, networks become intractable, and using a variational distribution, such distribution in weights can be learned. Distribution in weights provides a measure of uncertainty. The probabilistic distribution provides confidence bound in the output which is necessary for data analysis and decision making. Point estimates only provide if someone has diseased/injured or not but it is often subjected to overconfident results and not sure if prediction makes sense or is random. It does not tell whether to change the model or create more diverse data or should be more careful during the analysis of results. In an automated system where physicians rely on models, uncertainty can provide more information to physicians if the test data is out of distribution.
For a binary image classifier, if the model predicts 0 then it is abnormal and if the model predicts 1 then it is normal. For a difficult image, the model may predict 0.5. In such a case, variance of declaring normal or abnormal would be high. Such variability can be captured by aleatoric uncertainty. Aleatoric uncertainty measures the noise inherent in the

observations. It is present during the data collection from sensors which is present in all datasets. Increasing the dataset does not reduce aleatoric uncertainty.
If different models are used, then there may be different probability and variability to declare difficult images as normal or abnormal. Such variability can be captured by epistemic uncertainty. Epistemic uncertainty is caused by the model itself. Increasing data reduces epistemic uncertainty.

2. Literature Review

2.1 Deep Learning on Knee MRI

There are various ways in using deep learning on MRI images. Due to the multidimensional and multiplanar properties of the MRI image, there are different architectures. Some of the architecture is single path architecture where the image is taken from anyone plane and some of the architecture uses multipath architecture for more accurate prediction. Bien et al.[1] developed CNN which predicts injuries from three different planes. They performed nine CNN operations for three outcomes. They achieved good performance comparable to general physicians and radiologist. Liu et al.[2] segmented ACL from MR images using CNN and applied another CNN to detect structural abnormalities in the ligament. Roblot et al.[3] applied fast-region CNN and faster-region CNN to detect the position of the horn, tear, and orientation on 1123 MR images of the knee.

2.2 Uncertainty Estimation in Deep Learning

Shridhar et. al. [4] introduced uncertainty estimation for Bayesian CNN with variational inference. They have also estimated aleatoric and epistemic uncertainty by normalizing the output at the final layer, introduced distribution in weights in convolutional layers and calculated intractable posterior weights by Bayes by backprop. They tested their results on different datasets such as MNIST, CIFAR-10, CIFAR-100. They achieved the performance equivalent to frequentist in identical architecture. Blundell et. al.[5] introduced distribution on weights using Bayes by backprop. They used an expected lower bound on likelihood function to regularize the weight by minimizing variational free energy. They also calculated uncertainty to improve generalization in nonlinear regression problems. They performed the model on MNIST dataset and obtained results equivalent to that of dropout. Kendall et al.[6], in the paper, have breakdown the uncertainties into two parts using moment-based uncertainty decomposition; aleatoric and epistemic uncertainty. Aleatoric uncertainty captures noise that is inherent to the observation and epistemic uncertainty captures the uncertainty in the model itself. They evaluated the model with pixel-wise depth regression and semantic segmentation. To approximate the posterior there are methods like Bayesian inference and Montecarlo sampling. Gast et. al.[7] proposed a model where distribution is kept only at input and activation but not on the weight. It made a model simple and agnostic to network architecture and optimization processes. Not including distribution in weights results overconfident prediction for input that is not well represented in the training data. Montecarlo sampling is another technique to estimate uncertainty. It is like a multiple network that learns in training and dropouts at the test time. However, it cannot represent data uncertainty due to sensor noise. To represent data uncertainty Kendall et al. [6] added variance to output. It is then trained by maximum likelihood loss on data using Montecarlo sampling, both model and data uncertainty can be calculated. It is unsure about the network architecture because of the addition of variance. Gal and Ghahramani et al. [8] proposed a method of Montecarlo dropout that uses dropout at every weight layer at a test time. MC dropout averages output over many samples at the test time. In this way, the author calculated mean and predictive uncertainty. One advantage is that it can be applied to trained models.

2.3 Related Works

Deodata et. al.[9] used a Bayesian Neural Network and used uncertainty estimation to perform prediction and analysis on a dataset. After uncertainties estimation, they discarded highly uncertain predictions and identified unfamiliar patterns in the data which is classified to outliers which can be both corrupted observation or data belonging to different domains. The author also applied a Bayesian approach to biomedical imaging and they identified noise in labels as well as uncertainty. Kwon et al.[10] have proposed Bayesian Neural Network which breaks down variance into aleatoric and epistemic uncertainty. They have implemented a method that yields correct implementation of each certainty. Uncertainty calculated gives additional insights than a pointwise prediction. They have applied a variational

inference technique and used to classify and segment ischemic stroke lesion segmentation with uncertainty in segmentation image as well. There are various methods to calculate uncertainty estimation as proposed by the above methods. Some of the paper calculates uncertainty without dealing with complexity. Gal et al.[11] calculates uncertainty using Montecarlo dropout which does not provide true uncertainty because it does not model predictive probabilities. Some of the papers only calculated epistemic uncertainty i.e. model uncertainty but uncertainty can be from both model and data uncertainty.
This research work calculates uncertainty using variational approximation in the multi-label output. Dataset consists of output from three planes which are concatenated after the convolutional operation. Prior distribution in weight is introduced which is calculated using variational approximation by minimizing KL divergence loss because of the intractability of true posterior. KL divergence term, which is intractable as well, is sampled using Montecarlo approximations. The parameters are updated using a backpropagation algorithm. During testing, all the distribution in weights is sampled to get the mean and variance of the predicted output. The resulting mean and variance are then used to measure uncertainty in the output. Variance can be decomposed into aleatoric and epistemic uncertainty. A detailed explanation is given in the methodology section.

3. Bayesian Convolutional Neural Network

In Bayesian CNN, we do not deal with the point estimate of the weights but filters with distribution in the weights. BCNN is a Bayesian Convolutional Neural Network layer that consists of two convolutional operations; one for mean and other for a variance. Using a distribution in weight makes weight intractable. So, weight is approximated using variational approximation and the predictive probability is calculated using samples taken from the weights. By taking a number of samples, the variance is calculated which is used to calculate uncertainty in the model.

Posterior probability on weights is calculated using Bayes rule.

$$p(x) = \frac{p(x|\theta)}{p(x)} \tag{1}$$

where,

$$p(x) = \int p(x,\theta)p(\theta)d\theta$$

x is an input data and θ is the parameter that consists of weights and biases. $p(\theta)$ is the prior probability which is in the form of $N(\mu,\sigma^2)$. $p(x)$ is intractable i.e. it cannot be solved. So variational inference is used to approximate the functional form. Posterior function $p(w|D)$ is approximated with another distribution $q(w|D)$ with some variational parameters. KL divergence is used as an optimizer. If θ is the parameter to be optimized then

$$\theta_{opt} = argKL[q_\theta(w)||p(w|x,y)] \tag{2}$$

where,

$$KL[q_\theta(w)||p(x,y)] = \int q_\theta(w)log\frac{q_\theta(w)p(x,y)}{p(w)p(w)}dw$$

$$\theta_{opt} = argmin_\theta \int q_\theta(w)log\frac{q_\theta(w)}{p(w)}dw$$

$$-\int q_\theta(w)logp(w)dw$$

Where the first part is KL divergence between q and prior p(w) which acts as a regularization. The second part is the Expectation of the log-likelihood of variational distribution which is used to fit the data.

$$\theta_{opt} = argmin_\theta KL[q_\theta(w)||p(w)] - E_q\theta(w)log(x,y|w) \tag{3}$$

KL divergence is again intractable i.e. it cannot be solved. A variational method is used to solve the intractable part. Weight w is sampled from the variational distribution $q_\theta(w|D)$ because it is easier to sample from the variational distribution than from the true distribution p(w—D). After sampling, the cost function is obtained which can be optimized during training phase given by

$$F(D,\theta) \cong \sum logq_\theta(D) - logp(w^i) - logp(D|w_i) \tag{4}$$

Where n is the number of samples taken.

To account for class imbalance problems, low occurrence labels are multiplied by higher weights and higher occurrence is multiplied by small weights in the loss function. Weights are determined by the number of occurrences of a particular label divided by the total number of exams. To calculate derivatives of a parameter i.e. distribution (Kingma et al. 2015) proposed local reparameterization trick which learns μ, σ for any weight. Let ε be a sample of standard Gaussian distribution, it is multiplied with standard deviation and added with mean.

$$\theta = (\mu, \sigma^2)$$

$$\varepsilon \sim N(0,1)$$

$$f(\varepsilon) = w = \mu + \sigma.\varepsilon$$

Weight consists of these parameters: value μ, σ that is required. Mean is learned as in frequentist approach which is also maximum a posterior probability of variational posterior distribution . Variance is learned from the second convolutional operation. It means how much value weight (w) deviates from the mean. To ensure variance never reaches zero, the softplus activation function is used.

3.1 Sampling the Variational Distribution

During testing for classification, predictive distribution is given by $p(y_{test}|x_{test})$, where x_{test} is a test data and y_{test} is a predicted class.

$$p(y_{test}|x_{test}) = \int p(x_{test}, w) q_\theta(w) dw \tag{5}$$

To compute $p(y_{test}|x_{test})$, Montecarlo approximation is used. Predictions of multiple weight samples are taken from variational parameter $q_\theta(w)$.

$$\mu = E_q[p(x_{test})] \approx \frac{1}{N_m} \sum_{n}^{N_{mc}} p(y_{test}|x_{test}, w_n) \tag{6}$$

$$w_n \sim q_\theta(w)$$

$$\sigma = var_q p(x_{test}) = E_q[(y - E[y])]^2 \tag{7}$$

Variance can be decomposed into aleatoric and epistemic uncertainty. Aleatoric uncertainty provides information about the noise that is inherent to the observation. Epistemic uncertainty provides uncertainty in the model.

3.2 Uncertainty Estimation

Variance can be decomposed into two uncertainty model by the equation below

$$var_q p(y_{test}|x_{test}) = \frac{1}{T}\sum_{t=1}^{T} diag(\sigma_t)^2 + \frac{1}{T}\sum_{t=1}^{T} (\hat{\mu}-\mu)(\hat{\mu}-\mu)^T$$

$$Aleatoric = \frac{1}{T}\sum_{t=1}^{T} diag(\sigma_t)^2 \tag{8}$$

$$Epistemic = \frac{1}{T}\sum_{t=1}^{T} (\hat{\mu}-\mu)(\hat{\mu}-\mu)^2 \tag{9}$$

In this way, Mean, Aleatoric, and Epistemic uncertainty is estimated. Mean value gives the point estimate which is compared with the frequentist approach for comparison along with the benefit of aleatoric and epistemic uncertainty for the extra information to out of the distribution test dataset.

3.3 Weight Initialization

Neural Network has millions of parameters that is why initializing weights is challenging. If suitable weights are used during initialization, the Neural network converges otherwise it does not converge at all. One good practice for weight initialization is to keep random weights in suitable range. Setting weights close to zero without making it small is considered common practice. If n is the number of inputs given to a neuron, a good practice is to keep the weights in the range of [-y,y] where y =sqrt(n). For Normal distribution, initialization of weight is 0 for mean and sqrt(n) for a variance.

4. Montecarlo Dropout

Montecarlo Dropout is the dropout performed at a test time. Dropout is used as a regularization technique used during training, but if used during testing, it acts as an ensemble of networks. The ensemble of different networks help us to find the mean and variance of the prediction which can be used as an inference technique by estimating the uncertainty of the prediction. The dropout layer used before every layer in the network is also considered as a Bayesian approximation. When a small dropout is used, it eliminates Montecarlo sampling, and when a large dropout is used, it requires more iterations to converge. Dropout enabled at a test time generates different output at every forward pass. From the first and second moments of those forward passes, uncertainty can be estimated.

5. Methodology

5.1 Model Architecture

Model Architecture consists of the architecture which consists of 5 convolutional operations as shown in the Figure- **??**. The first two layers are tuned to weights from the Alexnet model trained in Imagenet datasets. For the remaining last three layers gaussian initialization is used for the Bayesian CNN model and random initialization for the Montecarlo dropout method. s*256*256 size of an input image is reshaped to s*3*224*224 after the preprocessing, where s is a series of an image in exams and 3 is the number of channels. Then it is passed to the architecture. In the Bayesian CNN model, the convolutional operation is performed twice for both mean and variance. In this way, distribution in the weights is introduced. The reparameterization trick is introduced which combines mean and variance with Gaussian noise so that gradients can be defined in the forward pass. The weights of both mean and variance are updated using the backpropagation algorithm. Instead of sampling weights from combined mean and variance, Gaussian noise is sampled during the forward pass. Dropout of 0.2 is kept at the last three convolutional layer to reduce the complexity of the model in the Bayesian CNN model. The average pooling layer is inserted after the last convolutional operation and the maximum values of each series of exams are taken. It is done in all three planes and the output results are concatenated with each other which becomes 768-dimensional vectors after flattening. It is then mapped to the output layer. In the Montecarlo method, there is a single convolution operation, unlike the Bayesian CNN model. The weights are updated using backpropagation and Adam optimizer. The loss function used is binary cross-entropy. After the training is completed, dropout is introduced at the test time, which generates multiple forward outputs. The first and second moments of those outputs can be used to estimate uncertainty in the model. Because of multi-label output, this architecture is performed on each label to estimate both the mean and variance of the output prediction.

To calculate uncertainty in a model, distribution is added to the weights of a model architecture. Prior distribution p_{theta} is added which is in the form of $N(\mu,\sigma^2)$ and posterior distribution on parameter $p(\theta|x)$ is calculated using Bayes rule as discussed in section 3.

$$b = A_i * \mu + \varepsilon_j . \sqrt{A_i^2 * (\alpha_i . \mu_i^2)} \tag{10}$$

The reparameterization trick is to sample from parameter-free distribution and then transform ε with a deterministic function b so that gradients can be defined. The distribution function b consists of two convolutional operations. In the first convolution operation, it is treated as a frequentist convnets and optimized using backpropagation. In the second convolutional operation, the variance is learned. The gradients are calculated using a backpropagation algorithm which are same for both mean and standard deviation. Instead of weights, sampling is taken from b for computational acceleration. The mean value μ is learned from the first convolutional operation and in the second convolutional operation α is learned as mean μ is already learnt in first convolutional operation. The mean value is added to the outcome of the second convolution operation multiplied by ε to obtain deterministic function b. Bayes rule is used to calculate posterior which is intractable. The variational inference technique is used to approximate posterior as shown in by the given equation.

$$F(D,\theta) \cong \sum logq_{\theta}(D) - logp(w^i) - logp(D|w_i) \tag{11}$$

The first two-term in the equation is data-independent and is calculated in each layer. It also acts as a regularizer in the network. The last term is the data-dependent term and is evaluated at the end of the forward pass. The last term is also called maximum likelihood estimation which is same as conventional loss function in deep learning. In our case, the binary cross-entropy loss function is used for its estimation. The weighted binary cross-entropy is used to tackle the class imbalance problem. The value computed from the first two-term is in the range of 106, so the last term is multiplied by 106 for the numerical stability of the network. Now that posterior probability is approximated to obtain the variational

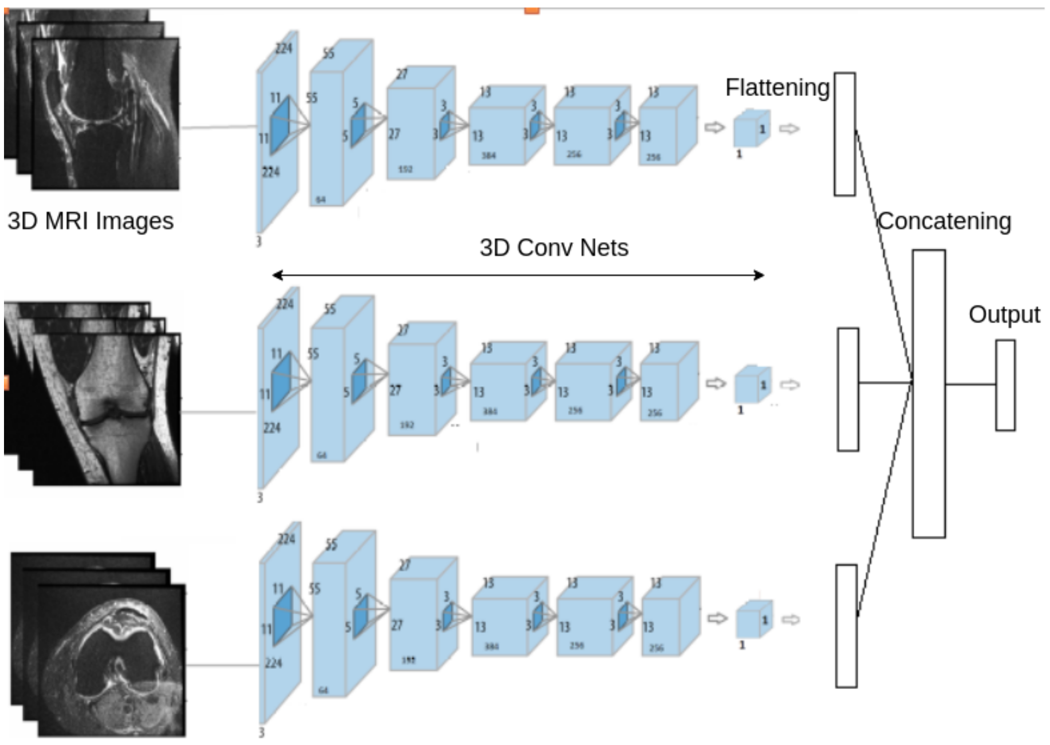

Figure 1: Model Architecture

parameters, it is then used to predict never seen data i.e. test dataset. The test data is passed through the model which generates different output each time when test data is passed. Montecarlo approximation is used where multiple weight samples are taken and the average value is calculated. In this way, mean and variance are calculated from multiple samples and then it is decomposed to aleatoric and epistemic uncertainty as given by 8 and 9.

5.2 Datasets

Dataset [1] consists of 1,250 knee MRI exams from three different planes; Saggital, Coronal, and Axial planes. Dataset is split into training, validation and test sets in ratio 80:10:10 respectively. Publicly available dataset from Štajduhar et al.[12] which consists of 917 sagittal PD-weighted exams from a Siemens Avanto 1.5-T scanner at Clinical Hospital Centre Rijeka, Croatia. There are three labels of ACL disease; non-injured (690 exams), partially injured (172 exams), and completely ruptured (55 exams). Dataset is split upon 60:20:20 ratio into train, validation and test set then the model is optimized using external training and validation set to discriminate between non-injured and injured ACL.

6. Results

Table 1: Performance metrics of various models

	AUC	Aleatoric	Epistemic
Bayesian CNN (Abnormal)	0.87	0.096	0.005
Montecarlo Dropout (Abnormal)	0.85	0.173	0.00088
Pretrained Alexnet (Abnormal)	0.8		
Bayesian CNN (ACL)	0.86	0.12	0.0049
Montecarlo Dropout (ACL)	0.84	0.18	0.00055
Pretrained Alexnet (ACL)	0.8		
Bayesian CNN (Meniscus)	0.75	0.168	0.0036
Montecarlo Dropout (Meniscus)	0.73	0.19	0.00097
Pretrained Alexnet (Meniscus)	0.76		

We can observe that the AUC score of the Bayesian CNN and pretrained model are similar, and the AUC score of the Montecarlo dropout is slightly less than that of Bayesian CNN. The Aleatoric uncertainty is nearly similar in both Bayesian CNN and Montecarlo dropout method as it provides the information about the data uncertainty, whereas epistemic uncertainty is higher for the Bayesian CNN which indicates Montecarlo dropout has a higher model uncertainty than a Bayesian CNN model.

Table 2: Performance metrics of various models of Sagittal plane of ACL label

	AUC (without additional training)	AUC (with additional training)
Bayesian CNN	0.877	0.917
Montecarlo dropout	0.85	0.903
Bien et al. [2]	0.824	0.911
Stajduhar et al. [2,5]		0.89
Alexnet model after rejecting uncertain images	0.891	0.923

External dataset is formed on Stajduhar et al. dataset [12]. Dataset consists of 917 sagittal MRI images. It is split in 60:20:20 ratio into train, valid and test dataset. For validation, the model is trained on the Stanford sagittal plane and ACL label and tested on the Stajduhar test dataset with no additional training. The model achieved 0.87 for the Bayesian model and 0.85 for the Montecarlo model. Furthermore, Stajduhar training datasets are combined with the Bien et al. training datasets to detect the ACL injury. The model achieved the AUC score of 0.917 for the Bayesian CNN model. The results obtained are shown in the table above. Aleatoric uncertainty is in the range of 10^{-1}, whereas epistemic uncertainty is in the range of $10^{-}3$ for both the labels. Given that the AUC score for both the models is high, their uncertainty is also similar and in the same range. As the test AUC increases in the model, epistemic uncertainty decreases [4] which can be observed in the results. The uncertain images from training the ACL labels in the Bayesian model on sagittal plane of Bien et al. dataset without additional training are removed and retrained using Alexnet pretrained model and the AUC score obtained is 0.89 on the test dataset of Stajduhar test datasets, whereas the model trained by Bien et al. has the AUC score of 0.824, and when trained with an additional training dataset of Stajduhar et al. dataset, the AUC score obtained is 0.923, whereas the model trained by Bien et al. is 0.911.

7. Conclusion

Hence, probabilistic modeling of weights is performed on multi-label 3D knee MRI data. For multi-label problems, the model is trained separately for each label. The obtained result is compared with pretrained models that are trained on 14 million images of ImageNet datasets. Although the pretrained model converges faster, it has an overfitting problem, and it cannot incorporate uncertainty. The probabilistic modeling of weights acts as a regularization technique with the benefit of uncertainty estimation. Uncertainty is estimated from the mean and variance obtained from the sampling of weights of variational posterior which is used as extra information during the diagnosis of the patients. Both Bayesian and Montecarlo methods can be used as an inference technique to estimate the uncertainty. The AUC of abnormal, ACL, and meniscus for Bayesian model is found to be 0.87,0.86, and 0.75 respectively. The obtained result is comparable to the pretrained Alexnet model and is slightly better than a Montecarlo method. For external validation, the dataset is trained on a sagittal plane on Bien et al. dataset for the ACL label and is tested on Stajduhar test dataset without further training. The obtained AUC score for the Bayesian and Montecarlo Dropout method is 0.87 and 0.85 respectively. After adding both Bien et al. dataset and Stajduhar et al. dataset, the AUC score obtained is 0.917 for the Bayesian model and 0.903 for the Montecarlo dropout method, where the state of the art for the similar case is 0.911 as obtained by Bien et al. [2]. The uncertain images i.e. noisy images obtained from the Bayesian model is removed from the training and then trained again using pretrained weights from Imagenet improved the AUC score of the test dataset of Stajduhar et al. dataset to 0.891 without additional training and 0.923 with the additional training. The uncertainty is breakdown into two part; aleatoric and epistemic uncertainty. Aleatoric uncertainty provides information about data uncertainty, whereas epistemic uncertainty provides information about model uncertainty. From the result, we can see that aleatoric uncertainty is in the same range for all the labels as it is trained on same datasets, whereas epistemic uncertainty decreases when the test AUC increases as obtained in the result section. Also, Montecarlo methods has a higher model uncertainty than a Bayesian model that can be observed from the AUC score of the labels in both Bien et al. and Stajduher et al. datasets.

8. Future Works

To reduce the number of parameters, we can experiment on different hyperparameters like depth of the filters, strides, etc. that may result similar performance to that of obtained result but with fewer training time. In the future, uncertainty can be introduced in the localization of the injuries as well which can provide more information about the injuries.

References

[1] Nicholas Bien, Pranav Rajpurkar, Robyn L Ball, Jeremy Irvin, Allison Park, Erik Jones, Michael Bereket, Bhavik N Patel, Kristen W Yeom, Katie Shpanskaya, et al. Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of mrnet. *PLoS medicine*, 15(11):e1002699, 2018.

[2] Fang Liu, Bochen Guan, Zhaoye Zhou, Alexey Samsonov, Humberto Rosas, Kevin Lian, Ruchi Sharma, Andrew Kanarek, John Kim, Ali Guermazi, et al. Fully automated diagnosis of anterior cruciate ligament tears on knee mr images by using deep learning. *Radiology: Artificial Intelligence*, 1(3):180091, 2019.

[3] V Roblot, Y Giret, M Bou Antoun, C Morillot, X Chassin, A Cotten, J Zerbib, and L Fournier. Artificial intelligence to diagnose meniscus tears on mri. *Diagnostic and interventional imaging*, 100(4):243–249, 2019.

[4] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. Uncertainty estimations by softplus normalization in bayesian convolutional neural networks with variational inference. *arXiv preprint arXiv:1806.05978*, 2018.

[5] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

[6] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.

[7] Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3369–3378, 2018.

[8] Y Gal. *Uncertainty in deep learning. University of Cambridge*. PhD thesis, PhD Thesis. 2016. URL: http://mlg. eng. cam. ac. uk/yarin/thesis/thesis. pdf . . . , 2016.

[9] Giacomo Deodato, Christopher Ball, and Xian Zhang. Bayesian neural networks for cellular image classification and uncertainty analysis. *bioRxiv*, page 824862, 2020.

[10] Yongchan Kwon, Joong-Ho Won, Beom Joon Kim, and Myunghee Cho Paik. Uncertainty quantification using bayesian neural networks in classification: Application to biomedical image segmentation. *Computational Statistics & Data Analysis*, 142:106816, 2020.

[11] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[12] Ivan Štajduhar, Mihaela Mamula, Damir Miletić, and Goezde Uenal. Semi-automated detection of anterior cruciate ligament injury from mri. *Computer methods and programs in biomedicine*, 140:151–164, 2017.