Prediction confidence for DNN classification of medical images

Maxime LANGLET - *Université Libre de Bruxelles (ULB)*

5 mai 2021

***Abstract* - A solution for predicting cancer in the prostate is proposed in this essay. Some neural network models were built to classify prostate cancer in medical images. It is necessary to do some data-augmentation giving the limited size of our database. It is shown that combining models which takes different input images yields the best test accuracy. Whilst it is a good start to creating software capable of predicting prostate cancer, the models could benefit from a few improvements.**

Keywords : Deep Learning, Prostate cancer classification, ProstateX Challenge

1 Introduction

This report summarizes my attempt at the ProstateX Grand Challenge. This challenge is the live continuation of the offline PROSTATEx Challenge ("SPIE-AAPM-NCI Prostate MR Classification Challenge") that was held in association with the 2017 SPIE Medical Imaging Symposium. The task of this challenge is to predict the clinical significance of prostate lesions found in MRI Data. In particular, a comparison between multiple models taking as input different images is provided testing this conclusion : **"The addition of an ADC map to T2-weighted images can improve the diagnostic performance of MR imaging in prostate cancer detection"**.

Moreover, four distinct models with the same neural network architecture are studied ; the first taking ADC images of size 40 by 40 (ADC 40) and the same but taking 60 by 60 as input (ADC 60), one taking T2-weighted images (T2) and lastly taking both, ADC and T2-weighted images as input (Combined). The implementation can be found at here.

1.1 Data

The data provided with this challenge is a collection of MRI studies. In this particular report, only Apparent diffusion coefficient (ADC) and T2-weighted images were used, although more were available in the data set. These images came in a DICOM encoding. Each patient having one study with a series of DICOM images. Extra information about the database and the capture methods for each series of images are given here [1].

For each patients, several folders can be found, each comprising several instances. All the information concerning a particular patient and images is found in the *ProstateX-Images-Train.csv* table. Some of the columns composing this table will help us extract the correct images for a given patient. These columns are the following :

- ProxID – ProstateX patient identifier
- Name – Series Description
- fid – Finding ID
- ijk – image col,row,slice coordinate of finding
- DCMSerNum – The DICOM Series Number

For example, to get the ADC image of Patient ProstateX-0123 go to patient ProstateX-0123 in the csv file and find the series with ADC in it. In this case it is 'ep2d_diff_tra_DYNDIST_ADC'. It has SeriesNumber 8. The DICOM images in that series form the ADC image for this challenge. Image slice k at coordinate i,j contains a finding fid.

1.2 Findings

Documented in the *ProstateX-Findings.csv* table are the findings, this table having, in our case, these useful columns :

- ProxID – ProstateX patient identifier
- fid – Finding ID
- ClinSig – Identifier available in training set that identifies whether this is a clinically significant finding.

As a note, for a finding (i.e. lesion) to be Clinically significant (Clinsig), the biopsy GleasonScore was 7 or higher. Findings with a PIRADS score 2 were not biopsied and are not considered clinically significant. The occurrence of clinically significant cancer in PIRADS 2 lesions were less than 5%.

The fid is the finding ID in both the ProstateX-Findings.csv file and the ProstateX-Images.csv file. The Findings spreadsheet has one row per lesion (if a case has only one lesion, then the only fid for that case will be "1" , alternately if a case has two lesions, then there will be an fid of "1" and an fid of "2" for that case).

Note that the Gleason Score is the grading system used to determine the aggressiveness of prostate cancer. This grading system can be used to choose appropriate treatment options. PI-RADS (Prostate Imaging–Reporting and Data System) is a structured reporting scheme for multiparametric prostate MRI in the evaluation of suspected prostate cancer in treatment naive prostate glands.

2 Data Extraction and Assumptions

2.1 Data Extraction

Following the steps given in the preceding example, we now have steps to follow for extracting the correct images constituting our data set. Similarly, the Clinical Significance of each picture can be loaded which will compose our feature set. In this report, our models will be given the exact image slice of the finding given by the ijk indices. Hence, these indices must be kept in memory. Later on, we will also crop the images using these indices.

Feeding the model with the entire series of images was considered, but every patient didn't have the same number of images per series. Giving a fixed number of images for each patient could have been tested, naturally including the slice with the finding to be classified. This topic is further discussed in section 4.3.

2.2 Assumptions

As pointed before, only ADC and T2-weighted images were used in this report. More precisely, only transverse cross section images and ADC images were used since in practice, only these images are absolutely necessary in the diagnostic of a lesion. On the other hand, coronal cross section images are mostly used for better locate a lesion found using the transverse section. Using these images, we will try and verify the conclusion given by this paper [8] which said that **"The addition of an ADC map to T2-weighted images can improve the diagnostic performance of MR imaging in prostate cancer detection"**.

Another assumption made on the data was that the indices ijk, in particular k, ranged from 1 to the number of slices. Since no details were covering this issue in the challenge documentation and our coefficients ijk extracted were always superior to 0, it felt natural to range them from 1 to the number of slices.

Indices i and j represent pixels, misinterpreting them by one will not hold much weight since a large enough window moved one pixel to the right or left will still show the finding we are trying to isolate.

One uncertainty encountered was, when extracting the images using the indices ijk, some coefficients k were too large for our number of slices per series. Since the difference between k and the number of slices sometimes exceeded 6, I treated this issue as an error in our database and decided to exclude this finding completely from the training, validation, or testing data. Although this anomaly didn't present itself much, 5 findings for T2-weighted images and only one for ADC images were excluded.

In the section introducing our models, when we will give as input a superposition of ADC and T2-weighted images, of course, the findings being excluded for one were also excluded for the other.

3 Preprocessing and Data Augmentation

Firstly, after collecting every image, they were converted into 8-bit unsigned integers.

Afterward, using the indices ijk, only the slice k was kept and this slice cropped around the pixel at coordinates (i, j). The screenshots provided by the database inspired the shape of the crop. Based on this information, the cut of ADC images will result in an image of shape 40 by 40 in the first instance, then images of size 60 by 60 for our second model. Figure 1 shows an example of the transformation.

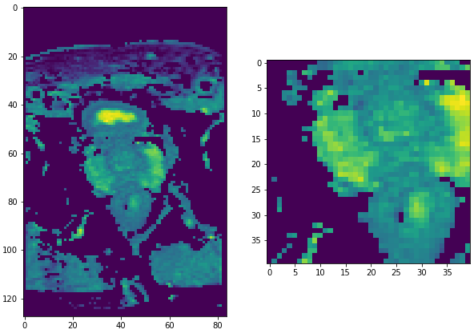


FIGURE 1 – Example of ADC image and the same image after crop (40,40)

T2-weighted images were cropped in a 160 by 160 image also inspired by the size of the given screenshots, then resized into 80 by 80 picture to facilitate the analysis. Figure 2 shows the resulting image after cropping.

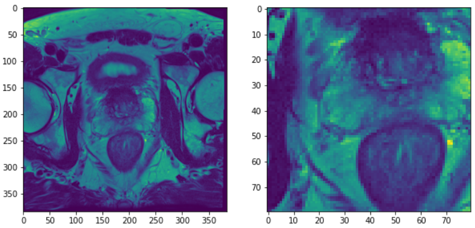


FIGURE 2 – Example of T2-weighted image and the same image after crop and resize (80,80)

When a model uses both types of images as input, the ADC images will be up-scaled to fit the size of the T2 images (i.e. resized to a 80 by 80 image). It is paramount to note that utilizing the screenshots provided as a training set was not an option since some didn't represent the correct finding.

After randomizing the feature set and the data set equally, our database was divided into 60% training set, 20% validation and testing set. It is important to note that, since we shuffled the data-set, the testing, validation, and training set will hold different proportions of true or false findings. The following table summarizes the proportions of the sets :

	Training	Validation	Testing
ADC model	189	70	70
T2-weighted model	184	70	70
T2-ADC model	184	70	70

TABLE 1 – Summary of the size of the different sets for each model

The difference seen between the testing sets of each model is the consequence of the elimination of some images as pointed in the *Hypotheses* section.
Given the limited number of images in our data set, data augmentation will be required on training data. Furthermore, our database presented a balancing problem. Images with a clinical significance as "False" were far more present than those that are "True". The final count is 253 "False" Clinsig for 330 images, which represents more than 75% of the data set. To solve this problem, on the images with a "True" Clinsig, the window after the crop was shifted slightly in different directions creating quite the same number of images for both classes. Figure 3 and 4 illustrates the resulting images after this shift on the ADC and T2 respectively.

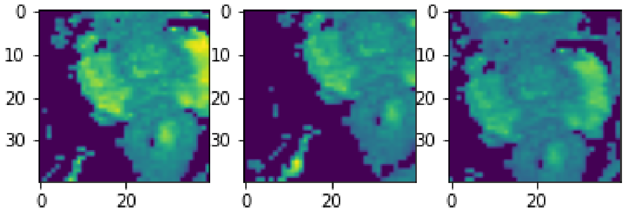


FIGURE 3 – Illustration on the shift countering the biaising problem for ADC

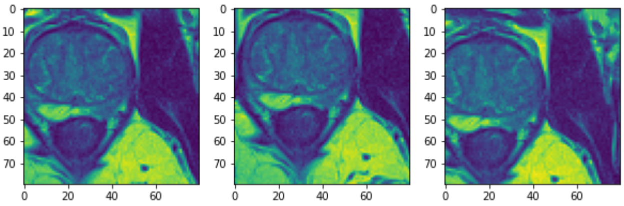


FIGURE 4 – Illustration on the shift countering the biaising problem for T2

Now that the images are all of size 80 by 80 and we have a balanced set of training images, the same data augmentation can be applied. Since a comparison between multiple models is made, It was established as interesting to apply the same transformations on T2 and ADC images.
Here's the transformations applied on the data-set :

- Histogram Equalization
- Brightness dimming
- Rotations
- Gaussian filter

The purpose of the equalization was to introduce different contrasts on the images of the training set.

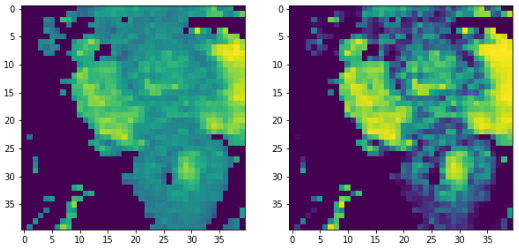


FIGURE 5 – Histogram equalization for ADC image

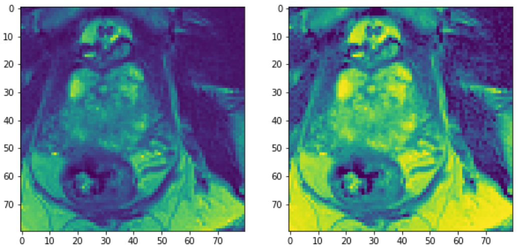


FIGURE 6 – Histogram equalization for T2-weighted image

From one medical image to another, pixels' values might vary, so reducing brightness was deemed relevant when doing data augmentation. Note that the images don't show any substantial differences, the color bar distinguishes the real difference between images. We see that we have a difference in scale between both images.

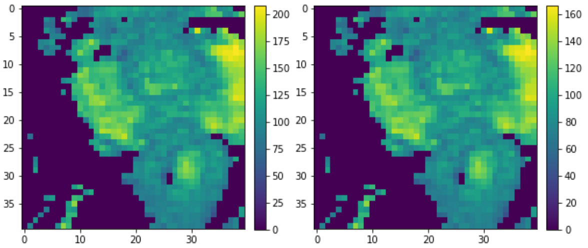


FIGURE 7 – Brightness dimming for ADC image (look at color-bar)

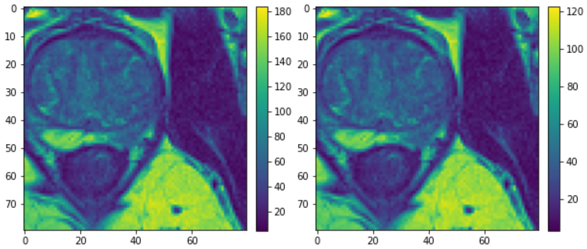


FIGURE 8 – Brightness dimming for T2-weighted image (look at color-bar)

Slight rotations were also introduced since they might occur naturally with image acquisition. The following angles were considered as acceptable, bigger rotations could cause our model to train on data that would never appear.

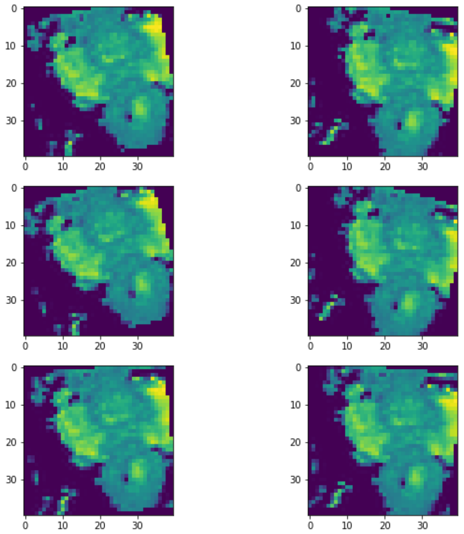


Figure 9 – Rotations on ADC image (10°,-10°,15°,-15°,5°,-5°) respectively

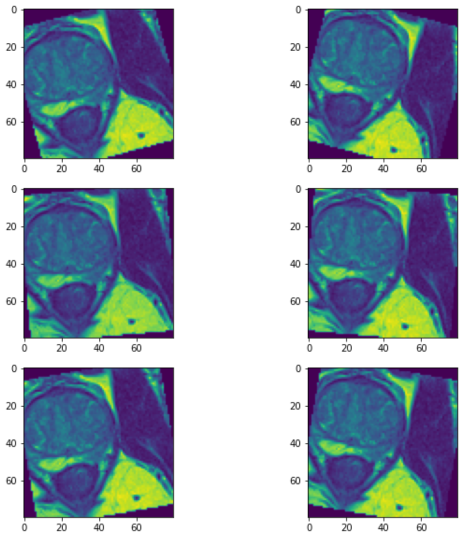


Figure 10 – Rotations on T2-weighted image (15°,-15°,5°,-5°,10°,-10°) respectively

Finally, a Gaussian filter is used on the images. This paper [7] speaks about the gains from introducing a Gaussian filter on the data, it is why it was thought this transformation could help the learning of our model.

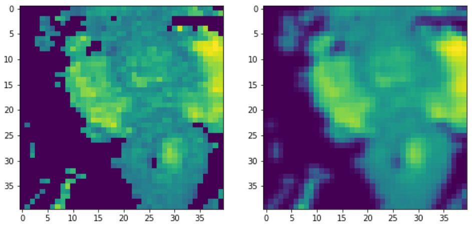


Figure 11 – Gaussian filter on ADC image ($\sigma = 0.7$)

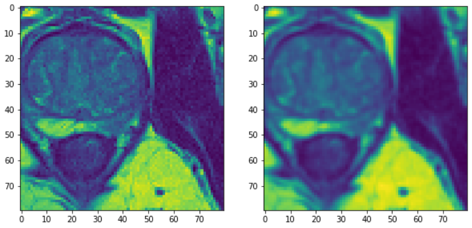


Figure 12 – Gaussian filter on T2-weighted image ($\sigma = 0.7$)

Naturally, other techniques for data-augmentation exists but were not implemented for various reasons. For example, a vertical flip of the image was not implemented since organs might not be symmetric. Adding Gaussian noise to an image, for example, was also not implemented because, as talked the preceding linked article, the validation accuracy doesn't benefit as much as a Gaussian filter. Before entering the model, tensors created from the images were normalized, giving better overall results. This observation was not tested thoroughly but could be investigated.

4 Model Implementation

4.1 Network architecture

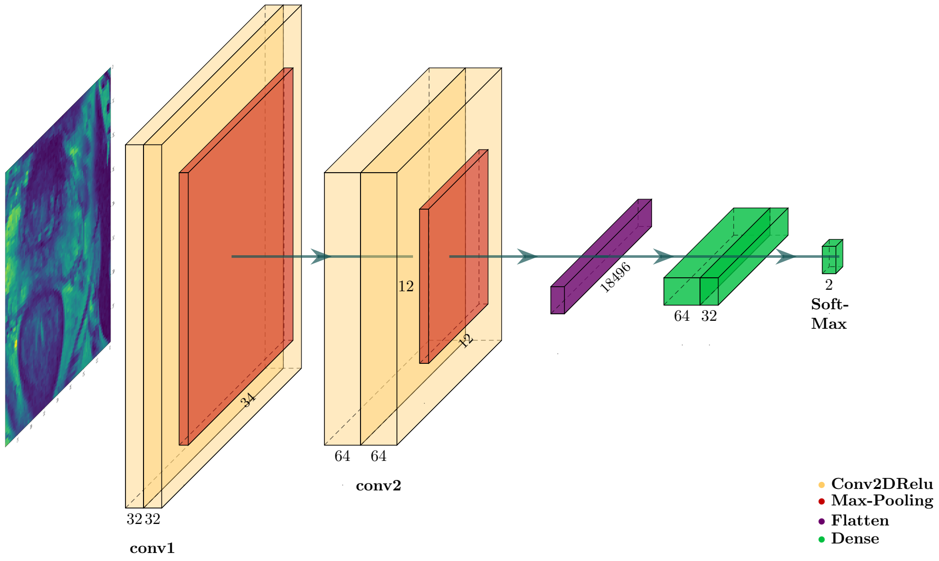


FIGURE 13 – Global model architecture

Figure 13 outlines the global architecture of the different models. The distinction between one and another is rudimentarily due to the input shape which changes between one another.

The architecture involves stacking convolutional layers with small 3×3 filters followed by a drop-out layer with small dropout rate. At the end of a block, a max-pooling is added. Together, these layers form a block, and these blocks can be repeated where the number of filters increases from 32 to 64. We then flatten the output and add a couple of dense layers. Then, a final dense layer with two outputs gives us our completed model.

All convolutional layers use Rectified Linear Unit (ReLU) as activation. The last dense layer uses a soft-max as activation, giving us an output of our two classes (True and False). This activation was taken as, in general, it learns better with those features.

4.2 Training results

Figure 15 shows the history of the different models. Some of graphs may differ purposefully to show different hypotheses tested. Firstly, the models are doing some overfitting despite the many dropout layers added. Some models history might let us believe that training on more epochs would slowly increase the accuracy but it doesn't. As shown in sub-figure (c), the validation accuracy is slightly decreasing. Further more, since we have perturbations on validation accuracy, the weights giving the best accuracy are saved and reused after training for testing. It is noticeable that the sub-figure (d) present a validation accuracy lower than others. This is due to a balanced validation set rather than a randomly drawn one. Since the initial database is unbalanced, a randomly drawn set will most probably be too. So, a first conclusion that can drawn from these results is that the models will perform better at predicting "False" Clinsig.

Note that our models are divergent. All models during training and testing presented large loss numbers. The general behavior of this loss function was, at first, the loss decreased as expected when the model starts training. When the training accuracy begins to stabilize, that's when the loss explodes, more or less. Let's see an example, every model presents the same behavior, then it's not necessary to show every loss graph since there are similar.

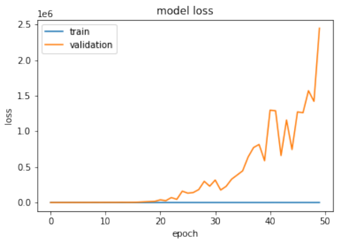
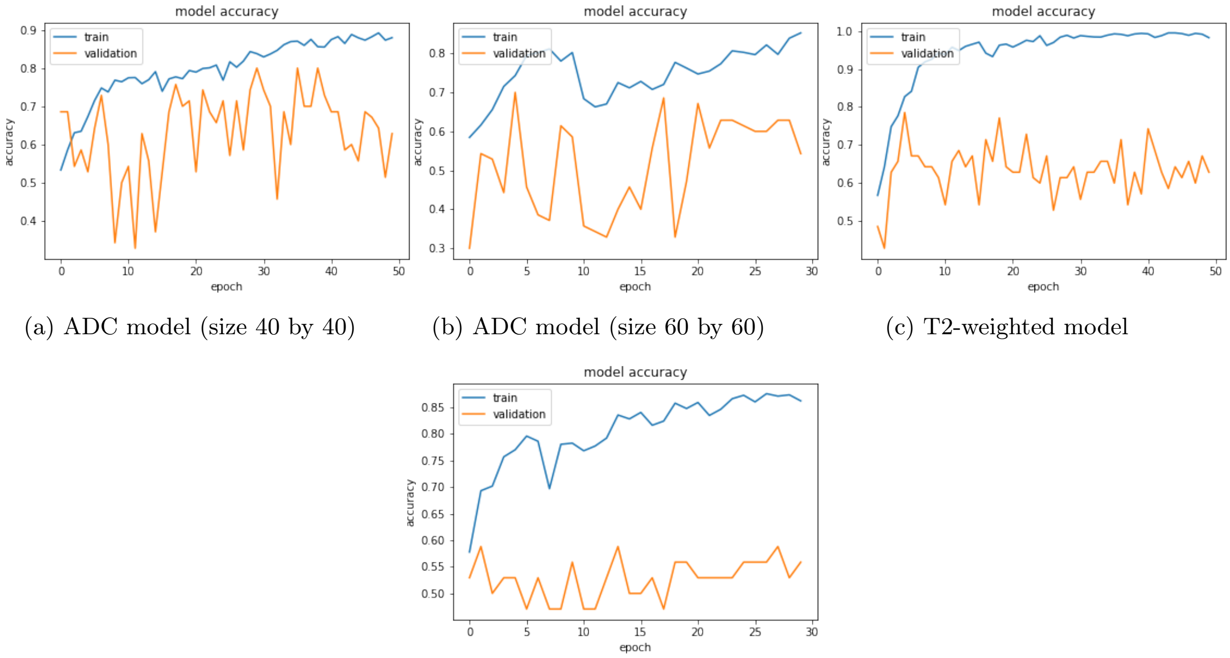


FIGURE 14 – Loss graph of T2-weighted images

Lowering the learning rate from the default 0.001 to 0.0001 only somewhat altered this effect. No obvious explanation was found concerning this divergence.



(a) ADC model (size 40 by 40)

(b) ADC model (size 60 by 60)

(c) T2-weighted model

(d) Model combined

FIGURE 15 – Training history of each model

4.3 Analysis and Possible improvements

To evaluate the models on the testing data, we'll use the following classification metrics :

- Misclassification rate : $ER = \frac{F_N+F_P}{N} = 1 - A$ where A is the Accuracy
- Balanced Error rate : $BER = \frac{1}{2}(\frac{F_P}{N_N} + \frac{F_N}{N_P})$
- Sensitivity : $SE = \frac{T_P}{N_P}$
- Specificity : $SP = \frac{T_N}{N_N}$

where T_N is True Negative, F_N False Negative, T_P is True Positive, F_P is False Positive, $N_N = T_N + F_P$ and $N_P = T_P + F_N$. Let's present the confusion matrices for each result over testing data.

		Prediction		
		Negative	Positive	Total
True diagnosis	Negative	41	17	58
	Positive	7	5	12
	Total	48	22	70

TABLE 2 – Diffusion matrix of ADC model (40) on testing data

		Prediction		
		Negative	Positive	Total
True diagnosis	Negative	28	26	54
	Positive	9	7	16
	Total	37	33	70

TABLE 3 – Diffusion matrix of ADC model (60) on testing data

		Prediction		
		Negative	Positive	Total
True diagnosis	Negative	39	15	54
	Positive	9	7	16
	Total	48	22	70

TABLE 4 – Diffusion matrix of T2-weighted model on testing data

		Prediction		
		Negative	Positive	Total
True diagnosis	Negative	40	19	59
	Positive	7	4	11
	Total	47	23	70

TABLE 5 – Diffusion matrix of combined model on testing data

	ER	BER	SE	SP
ADC 40	0.343	0.438	0.417	0.707
ADC 60	0.500	0.522	0.438	0.518
T2-weighted	0.343	0.420	0.438	0.661
Combined	0.371	0.479	0.364	0.678

Table 6 – Metrics summary

So it seems that our simple neural network architecture have reached it's limitations. These metrics highlights that our models are more capable of predicting a "False" Clinsig, with the exception of the 60 by 60 ADC images. Since the original database consists of more than 75% negative Clinsigs, this might be a consequence of this disparity. The data-augmentation done on the data may cause an over-fitting on "True" Clinsig since we multiply the same image with slight modifications into our training set. So it could cause the model to recognize these particular images better, to the detriment of other "True" findings. Looking more precisely at the BER metric, if we only balanced our testing set, we would have other results, supposing that we do not retrain our models. But a balanced initial database would impact the training and, therefore, the performance of the model. Hence, one possible improvement could be to collect more images with a better balance.

The prediction confidence we can expect from these models is limited at assessing false Clinsigs. It does a worst job of recognizing true lesions, confirmed by the SP metric having a much higher overall than the SE one. In a real-world test, these models could serve as an additional layer of confirmation to a false Clinsig, rather than the other way around.

After consulting with Dr. Constantin Moschopoulos, radiologist (with an expertise in prostate cancer), some key takeaways from the predictions on testing data are the following. When the model correctly classifies a "True" Clinsig, it seems that the lesions are located in the peripheral zone (PZ) of the prostate, which is usually the case in real life. Approximately 70 to 75% of prostate cancer are located in the PZ zone and are easier to detect in practice. On the other hand, the analysis is harder when the prediction is incorrect. Different cases arise, one of them being that the lesion stretches over the entire prostate, which might make the model unable to see contrast between lesion and prostate, which causes a false prediction. Another aspect, in the PIRADS criteria, a lesion should appear "black enough" to be considered as positive, and possibly some of the prediction might inadequately see a lesion as dark enough.

Taking in this information, some pre-processing might help us limit some of these inadequate results, for example, specify a certain threshold over the darker regions which would help with the degree of darkness.

One other instance tested, since the training was finalized for each individual models, could be to combine the different models into one bigger merged model. Since the performance of the ADC_60 model are underwhelming, this model will be excluded from this algorithm. The decision algorithm is quite simple, since 3 models are predicting, the maximum argument of the sum of the predictions will give our predicted class. In other words, two models predicting a same class will determine our global prediction. **Disclaimer :** this idea came after completing the training of the three different models. Remember that a random shuffle on the data was used before separating the different sets. Providing a new set of images will most probably pollute the results since there's 1% chance that an image was only present in the 3 testing sets. Still, to estimate the accuracy after merging theses models, a statistical approach can help us. The following equation will help us achieve this feat.

$$P\Big(\frac{A_1 + A_2 + A_3}{3} > \frac{1}{2}\Big) = A_1A_2A_3 + (1 - A_1)A_2A_3 + \\ A_1(1 - A_2)A_3 + A_1A_2(1 - A_3)$$

where A are the different accuracies. We find that our estimated final accuracy for the merge models is 0.71.

To confirm this result, we will still try our merge model with randomly drawn pictures, but keep in mind, these following results won't represent how the models really perform. Individually, our models gives these results : 0.8143, 0.8571, 0.7857. Now the merge model, on the same input images yields 0.8857 as accuracy. Applying the formula from above, we get a statistically estimated merged model accuracy around 0.91, which is tending to the observations. It can concluded that, with new images, our merge model would predict correctly input image at a 70% rate, which is definitely better.

Furthermore,in this report weight initialization was overlooked. The current standard approach for the initialization of the weights of neural network layers and nodes that use the ReLU activation function is the “he” initialization. Adding this could bring some improvements.

Also, a custom approach could have been taken. For each different inputs, the better way would have been to try and create the better performing model and then comparing them. It is likely that some models perform better for certain types of images.

Another way of improving our model could be to give the entire series of images as input. This resolution would potentially give the model more freedom in predicting the correct classes. If this proposition is too computationally costly, a less costly approach could have been to feed the complete images rather than a crop and resize them.

5 Conclusions

In conclusion, the results didn't initially show that a model taking as input ADC and T2-weighted images improve the predictions. But on the contrary, taking multiple models with these images as input does improve the overall performance. So, from a different perspective, we indeed verified our initial assumption of improving the diagnostic using both types of images.

Since our data-set is quite limited, which is relatively common in the medical field, data augmentation is required. It manifested improvements in the training of our models. For now, the models created will not be used in a professional environment since the results aren't quite reaching professional standards. Comparing it to the best results posted for this challenge on Grand-Challenge leaderboard, it is far from the best performing implementation, which is around 95% accuracy. But, this project gave the perfect introduction to the foundations of machine learning and neural networks. We learned the importance and wide range of use in a real-world application, specifically in medical imaging.

Acknowledgments

I would like to thank Olivier Debeir and Adrien Foucart for their advice and patience. I would like to thank also Louan Mol and Eliott Van Steirteghem for their help. Lastly, I would like to thank Dr. Constantin Moschopoulos for his help, advice and medical expertise.

References

[1] Tracy Nolan Pam Angelus. *SPIE-AAPM-NCI PROSTATEx Challenges.* URL : `https://wiki.cancerimagingarchive.net/display/Public/SPIE-AAPM-NCI+PROSTATEx+Challenges#23691656d4622c5ad5884bdb876d6d441994da38`.

[2] Jason Brownlee. *How to Classify Photos of Dogs and Cats (with 97% accuracy).* URL : `https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/`.

[3] Jason Brownlee. *Weight Initialization for Deep Learning Neural Networks.* URL : `https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/`.

[4] Samuel G. Armato Lubomir Hadjiyski Karen Drukker. *ProstateX.* URL : `https://prostatex.grand-challenge.org`.

[5] *Gleason Score and Grade Group.* URL : `https://www.pcf.org/about-prostate-cancer/diagnosis-staging-prostate-cancer/gleason-score-isup-grade/`.

[6] HarisIqbal88. *PlotNeuralNet.* URL : `https://github.com/HarisIqbal88/PlotNeuralNet?fbclid=IwAR0EEF21cYS2Why_bvr-0uCG8yam6_LcxEeJauDvdXUJ-scUpMhTm110J_4`.

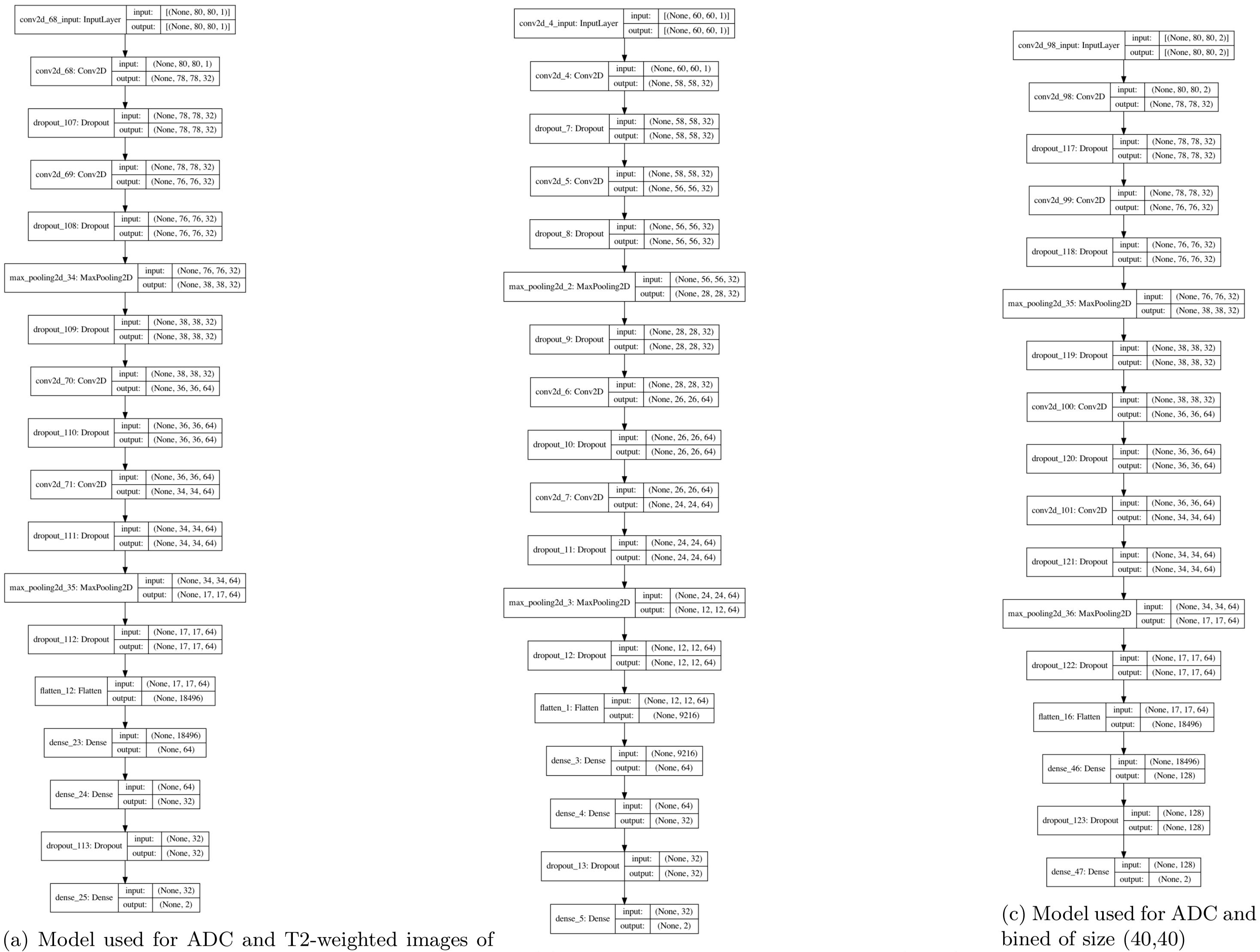
[7] Zeshan Hussain et al. « Differential Data Augmentation Techniques for Medical Imaging Classification Tasks ». In : *AMIA ... Annual Symposium proceedings. AMIA Symposium* 2017 (2017), p. 979-984. ISSN : 1942-597X. URL : `https://europepmc.org/articles/PMC5977656`.

[8] Hyun Kyung Lim et al. « Prostate Cancer : Apparent Diffusion Coefficient Map with T2-weighted Images for Detection—A Multireader Study ». In : *Radiology* 250.1 (2009). PMID : 19017927, p. 145-151. DOI : `10.1148/radiol.2501080207`. eprint : `https://doi.org/10.1148/radiol.2501080207`. URL : `https://doi.org/10.1148/radiol.2501080207`.

[9] *Prostate Imaging-Reporting and Data System (PI-RADS).* URL : `https://radiopaedia.org/articles/prostate-imaging-reporting-and-data-system-pi-rads-1`.

[10] *Tensorflow weight initialization.* URL : `https://stackoverflow.com/questions/43489697/tensorflow-weight-initialization`.

6 Appendix


conv2d_68_input: InputLayer
input: [(None, 80, 80, 1)]
output: [(None, 80, 80, 1)]
conv2d_68: Conv2D
input: (None, 80, 80, 1)
output: (None, 78, 78, 32)
dropout_107: Dropout
input: (None, 78, 78, 32)
output: (None, 78, 78, 32)
conv2d_69: Conv2D
input: (None, 78, 78, 32)
output: (None, 76, 76, 32)
dropout_108: Dropout
input: (None, 76, 76, 32)
output: (None, 76, 76, 32)
max_pooling2d_34: MaxPooling2D
input: (None, 76, 76, 32)
output: (None, 38, 38, 32)
dropout_109: Dropout
input: (None, 38, 38, 32)
output: (None, 38, 38, 32)
conv2d_70: Conv2D
input: (None, 38, 38, 32)
output: (None, 36, 36, 64)
dropout_110: Dropout
input: (None, 36, 36, 64)
output: (None, 36, 36, 64)
conv2d_71: Conv2D
input: (None, 36, 36, 64)
output: (None, 34, 34, 64)
dropout_111: Dropout
input: (None, 34, 34, 64)
output: (None, 34, 34, 64)
max_pooling2d_35: MaxPooling2D
input: (None, 34, 34, 64)
output: (None, 17, 17, 64)
dropout_112: Dropout
input: (None, 17, 17, 64)
output: (None, 17, 17, 64)
flatten_12: Flatten
input: (None, 17, 17, 64)
output: (None, 18496)
dense_23: Dense
input: (None, 18496)
output: (None, 64)
dense_24: Dense
input: (None, 64)
output: (None, 32)
dropout_113: Dropout
input: (None, 32)
output: (None, 32)
dense_25: Dense
input: (None, 32)
output: (None, 2)
conv2d_4_input: InputLayer
input: [(None, 60, 60, 1)]
output: [(None, 60, 60, 1)]
conv2d_4: Conv2D
input: (None, 60, 60, 1)
output: (None, 58, 58, 32)
dropout_7: Dropout
input: (None, 58, 58, 32)
output: (None, 58, 58, 32)
conv2d_5: Conv2D
input: (None, 58, 58, 32)
output: (None, 56, 56, 32)
dropout_8: Dropout
input: (None, 56, 56, 32)
output: (None, 56, 56, 32)
max_pooling2d_2: MaxPooling2D
input: (None, 56, 56, 32)
output: (None, 28, 28, 32)
dropout_9: Dropout
input: (None, 28, 28, 32)
output: (None, 28, 28, 32)
conv2d_6: Conv2D
input: (None, 28, 28, 32)
output: (None, 26, 26, 64)
dropout_10: Dropout
input: (None, 26, 26, 64)
output: (None, 26, 26, 64)
conv2d_7: Conv2D
input: (None, 26, 26, 64)
output: (None, 24, 24, 64)
dropout_11: Dropout
input: (None, 24, 24, 64)
output: (None, 24, 24, 64)
max_pooling2d_3: MaxPooling2D
input: (None, 24, 24, 64)
output: (None, 12, 12, 64)
dropout_12: Dropout
input: (None, 12, 12, 64)
output: (None, 12, 12, 64)
flatten_1: Flatten
input: (None, 12, 12, 64)
output: (None, 9216)
dense_3: Dense
input: (None, 9216)
output: (None, 64)
dense_4: Dense
input: (None, 64)
output: (None, 32)
dropout_13: Dropout
input: (None, 32)
output: (None, 32)
dense_5: Dense
input: (None, 32)
output: (None, 2)
conv2d_98_input: InputLayer
input: [(None, 80, 80, 2)]
output: [(None, 80, 80, 2)]
conv2d_98: Conv2D
input: (None, 80, 80, 2)
output: (None, 78, 78, 32)
dropout_117: Dropout
input: (None, 78, 78, 32)
output: (None, 78, 78, 32)
conv2d_99: Conv2D
input: (None, 78, 78, 32)
output: (None, 76, 76, 32)
dropout_118: Dropout
input: (None, 76, 76, 32)
output: (None, 76, 76, 32)
max_pooling2d_35: MaxPooling2D
input: (None, 76, 76, 32)
output: (None, 38, 38, 32)
dropout_119: Dropout
input: (None, 38, 38, 32)
output: (None, 38, 38, 32)
conv2d_100: Conv2D
input: (None, 38, 38, 32)
output: (None, 36, 36, 64)
dropout_120: Dropout
input: (None, 36, 36, 64)
output: (None, 36, 36, 64)
conv2d_101: Conv2D
input: (None, 36, 36, 64)
output: (None, 34, 34, 64)
dropout_121: Dropout
input: (None, 34, 34, 64)
output: (None, 34, 34, 64)
max_pooling2d_36: MaxPooling2D
input: (None, 34, 34, 64)
output: (None, 17, 17, 64)
dropout_122: Dropout
input: (None, 17, 17, 64)
output: (None, 17, 17, 64)
flatten_16: Flatten
input: (None, 17, 17, 64)
output: (None, 18496)
dense_46: Dense
input: (None, 18496)
output: (None, 128)
dropout_123: Dropout
input: (None, 128)
output: (None, 128)
dense_47: Dense
input: (None, 128)
output: (None, 2)

(a) Model used for ADC and T2-weighted images of size (40,40)

(b) Model used for ADC images of size (60,60)

(c) Model used for ADC and T2-weighted images combined of size (40,40)

FIGURE 16