Multiple Anatomical Structure Recognition in Fetal Ultrasound Images

14016036

***Abstract*—Ultrasound imaging is heavily dependent on operator skill - sonographers are trained to recognise key anatomical features when navigating around the body. This is not too dissimilar to how convolutional neural networks (CNNs) can be trained for image classification. Development of a robust and accurate CNN may assist in identification of difficult bodily anatomy. In this report we train and optimise a CNN in multiclass and one vs rest configurations to classify fetal ultrasound images with 87.81±1.96% accuracy.**

I. Introduction

Ultrasound (US) imaging has become a routine procedure in both early [1] and late [2] pregnancy to assess healthy fetal development. Although it does not directly affect perinatal morbidity [2] it does act as an effective screening tool for chromosomal abnormalities [3]. Unfortunately, any useful US screening metrics are heavily dependent on sonographer skill [4] - particularly worrying as the recent trend in trisomy 21 detection has moved to identifying more challenging markers, (e.g. the presence/absence of the nasal bone or tricuspid regurgitation [5]).

Deep learning has the potential to assist sonographers with identification of fetal anatomy [6], and has even achieved similar accuracy to said clinical experts [7]. Within deep learning, convolutional neural networks are a powerful tool for automated image classification. They are sparsely connected; features are evaluated on the scale of the convolution kernel rather than the whole image, allowing for memory and time efficient processing of images with variable input size [8]. CNNs have been successfully used to classify a variety of anatomical features in US images, at the multi-organ scale (e.g. within the abdomen [9]), single-organ scale (e.g. the thyroid [10]) or even microscale deposits (e.g. plaque composition in the carotid artery [11]).

In this report we create and optimise a relatively straightforward CNN able to classify fetal US images of different anatomical regions, evaluating both a multiclass and one vs rest variant.

II. Methods

A. Splitting the Data

Labeled fetal ultrasound images were provided for 266 subjects, and contained four different class labels: the head, heart, abdomen and other.

Each subject had a different number of frames and a unique class distribution. To prevent our CNN erroneously learning to classify based on intra-subject similarities rather than the actual data in each frame we split our training, validation and testing datasets based on subject, such that the model would be evaluated using frames from previously unseen subjects. The training, validation and testing data were split along a 70:20:10 ratio. By randomising which subjects were contained in each split we were able to create five different dataset permutations for use in cross validation [12]. The distribution of class labels in each dataset was not equal, as shown in Figure 1.

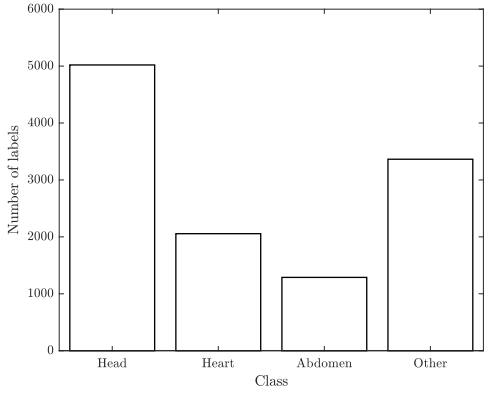


Figure 1: Distribution of class labels for the 266 subject.

B. Data Augmentation

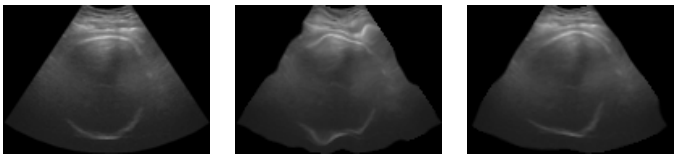
Training on unbalanced data can lead to misclassification problems if not addressed [13]. As such we augmented the training data using an elastic deformation algorithm [14] such that each class now contains $7,500$ frames. This method uses two parameters to control the degree of deformation, α, which randomly permutes the pixels and σ which applies a Gaussian convolution for smoothing. We found that setting $\alpha = 100$ and $\sigma = 8$ produced the most natural deformation (see Figure 2), this ratio has been used elsewhere for data augmentation of medical images [14]. We did not augment the validation and testing datasets as we wanted any frames used for evaluations to be completely physical.

C. Convolutional Neural Network Architecture

We started by creating a stripped down *base* CNN containing only one convolutional max-pooling pair followed by a relatively small fully connected layer (see Figure 3 and Table I for the complete architecture.) This basic CNN could act as a baseline from which we could evaluate any more complicated models. Our methodology for improving the

Table I: A simple *base* CNN from which we can build more complex models.

Layer	Name	Input shape	Output shape	Parameters	Kernel num.	Kernel size	Stride	Padding	Activation
1	Convol.	(92,128)	(45,63)	320	32	3	1	Same	Relu
2	Max-Pool	(45,63)	(45,63)	0	-	3	2	Same	-
6	Flatten	(45,63)	(90720)	0	-	-	-	-	-
8	Dense	(90720)	(32)	2903072	-	-	-	-	-
9	Softmax	(128)	(4)	132	-	-	-	-	-



(a) An undeformed ultrasound image of a fetal head.

(b) Elastic deformation with $\alpha = 34$ and $\sigma = 4$

(c) Elastic deformation with $\alpha = 100$ and $\sigma = 8$

Figure 2: Elastic deformation of an example frame. Using the parameters provided by Simard *et al.* produces very irregular deformation, resulting in the unnatural bending of developing bone and soft tissue. Using values of $\alpha = 100$ and $\sigma = 8$ produces a more natural deformation.

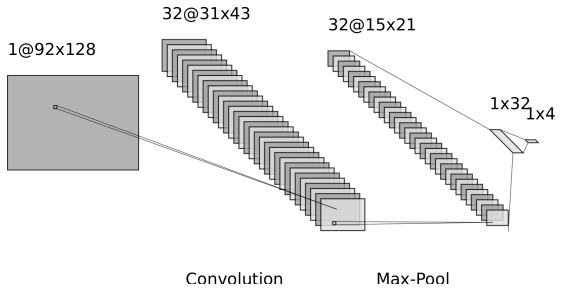


Figure 3: Architecture of the simple *base* CNN used as a baseline for evaluating more complex models.

underlying structure of the *base* model and optimising any hyperparameters is described in in the Experiments section.

We made two variants of our optimised model - a multiclass CNN with a softmax activation function and a one vs rest CNN with a sigmoid activation function. The one vs rest CNN has been trained to recognise each class separately, such that when evaluating a frame it iterates through the four different models and classifies based on the highest score.

D. Metrics to Monitor

To evaluate the efficacy of our proposed classifiers, testing accuracy and F1 score was calculated. F1 score was calculated to supplement accuracy as information on the rate of false positive and false negative misclassifications is particularly important in medical imaging. Five-fold cross validation was used to assess the generalisability of our CNN. To test for a statistically significant difference between the models we calculated p-values using a two-tailed t-test.

We also monitored the growth in validation accuracy during each training epoch to look for overfitting.

E. Visualisation

It can be difficult and unituitive to understand what occurs during deep learning [15] and so we implemented a few methods to aid in both understanding our CNN and visualising the high dimensional data used.

Confusion matrices were created to help identify any patterns in the misclassification of frames. Using this information we generated feature maps of a misclassifed example image. We also used t-SNE dimensionality reduction to help visualise the high dimensionality data used in this report [16].

III. Experiments

A. Hyperparameter Optimisation

Hyperparameter optimisation is computationally expensive [17] and as such we searched the literature to identify any hyperparameter that we could define outright. We decided to use maximum pooling as it outperforms other subsampling operations [18] and Relu activation functions due to the faster training time [19]. We also use the Adam algorithm for optimisation as it is much faster than stochastic gradient decent [20]. Inspired by LeNet-5 [21] and AlexNet [19] we use two convolution-pooling pairs and end with two fully connected layers.

To optimise our CNN we modify four different parameters: the convolutional layer kernel size and stride length, the size of the final fully connected layers and the degree of dropout. We use a grid search algorithm to iterate over the various hyperparameter permutations, train a model for each, and then evaluate. Due to time constraints no cross validation was performed during this optimisation.

B. Model Evaluation

Having optimised our CNN we trained it for 10 epochs against the *base* CNN using the additional datasets for cross validation. We compare the evolution in validation accuracy over each epoch and compare the final models accuracy and F1 score using a t-test. We also plot confusion matrices to evaluate the most problematic labels and qualitatively evaluate an example of a misclassified image.

We also train the proposed CNN using the augmented and non-augmented data to observe the effects of data augmentation.

Finally we compare multiclass and one vs rest variants of our CNN. Unfortunately due to time constraints we could not run cross validation of the one vs rest model.

Table II: The proposed *final* CNN as determined primarily through hyperparameter optimisation.

Layer	Name	Input shape	Output shape	Parameters	Kernel num.	Kernel size	Stride	Padding	Activation
1	Convol.	(92,128)	(31,43)	320	32	3	3	Same	Relu
2	Max-Pool	(31,43)	(15,21)	0	-	3	2	Same	-
3	Convol.	(15,21)	(15,21)	18496	64	3	1	Same	Relu
4	Max-Pool	(15,21)	(7,10)	0	-	3	2	Same	-
5	Dropout	(7,10)	(7,10)	0	-	-	-	-	-
6	Flatten	(7,10)	(4480)	0	-	-	-	-	-
7	Dense	(4480)	(128)	573568	-	-	-	-	-
8	Dense	(128)	(128)	16512	-	-	-	-	-
9	Softmax	(128)	(4)	516	-	-	-	-	-

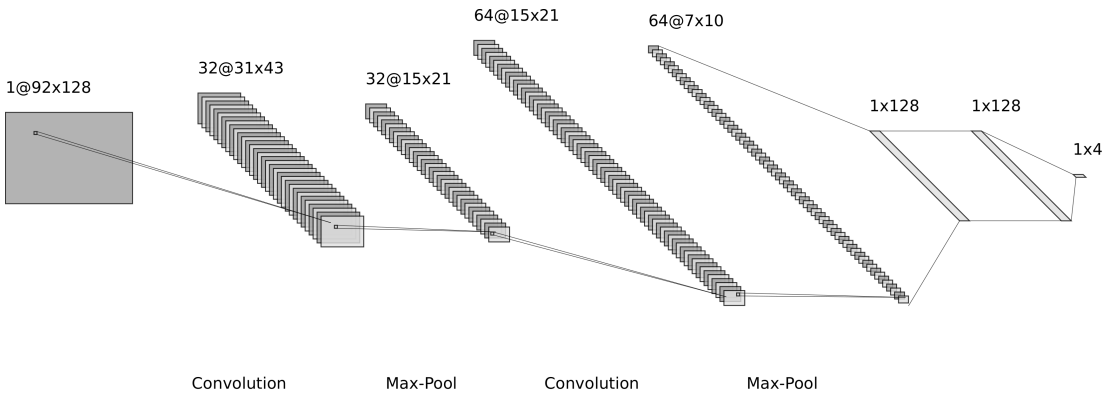


Figure 4: Architecture of our optimised *final* CNN.

IV. Results

A. Hyperparameter Optimisation

Table III: Testing accuracies (%) for different hyperparameter permutations without dropout. K=conv. layer kernel size, S=conv. layer stride and FC=fully connected layer size.

	K3S1	K3S3	K7S1	K7S3
FC32	83.45	81.39	84.16	82.10
FC64	84.50	82.73	87.06	83.95
FC128	85.17	86.64	85.84	87.14

Table IV: Testing accuracies (%) for different hyperparameter permutations with dropout=0.25. K=conv. layer kernel size, S=conv. layer stride and FC=fully connected layer size.

	K3S1	K3S3	K7S1	K7S3
FC32	83.91	82.02	86.30	83.36
FC64	86.93	86.13	85.88	84.83
FC128	86.64	85.88	87.94	87.52

Table V: Testing accuracies (%) for different hyperparameter permutations with dropout=0.50. K=conv. layer kernel size, S=conv. layer stride and FC=fully connected layer size

	K3S1	K3S3	K7S1	K7S3
FC32	83.07	84.12	84.62	87.82
FC64	85.00	86.85	87.65	87.39
FC128	86.30	88.07	86.85	87.90

As previously mentioned we use a grid search algorithm to evaluate all 36 possible permutations of the four key hyperparameters. The results are shown in Tables III, IV and V.

We find the highest accuracies occur when using a dropout of 0.5, which has a significant effect on limiting overfitting. Our overall highest test accuracy is found using a convolutional kernel size of 3, stride of 3, dense layers with 128 neurons and a dropout of 0.5. We use a learning rate of 0.001 and a batch size of 24. For a visualisation of this proposed *final* CNN and the full architecture see Table II and Figure 4.

B. Base vs Final CNN

As shown Figure 5 the *final* CNN reaches a higher overall validation accuracy much faster over the 10 epochs as compared to the the *base* CNN. The *final* CNN also has a statistically significant higher testing accuracy of $87.81 \pm 1.96\%$ against $81.25 \pm 3.70\%$ for the simpler *base* CNN ($p<0.05$).

C. Impact of Data Augmentation

We note that our proposed model converges to a higher validation accuracy after 10 epochs when using the augmented data (see Figure 5). The testing accuracy of the model trained using augmented data is significantly greater ($p<0.05$) at $87.81 \pm 1.96\%$ as compared to the same model trained on non-augmented data ($85.37 \pm 1.02\%$).

D. Multiclass v One vs Rest

The one vs rest CNN was substantially more computationally expensive; we did not have time to train it on the cross validation data and as such we cannot determine statistical

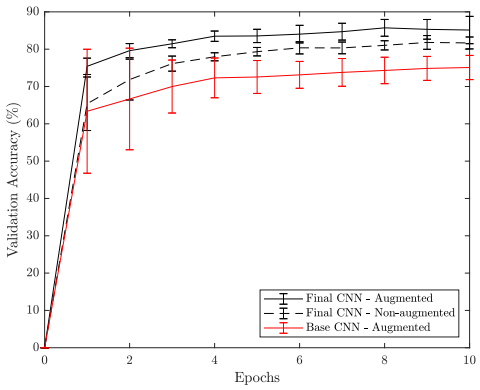


Figure 5: Change in validation accuracy per epoch for the *final CNN* trained with and without augmented data and the *base* CNN with augmented data.

significance. We have provided the code to do so however. For the single model we do train we find a slightly lower testing accuracy from the one vs rest model (86.5%). F1 scores were also very similar, with the multiclass scoring 82.34±4.23% and the one vs rest getting 83.2%.

E. Misclassification

To identify which labels were most commonly misclassified we calculated the confusion matrices for both the multiclass and one vs rest models (both matrices were similar, and so only the multiclass model is shown in Figure 6).

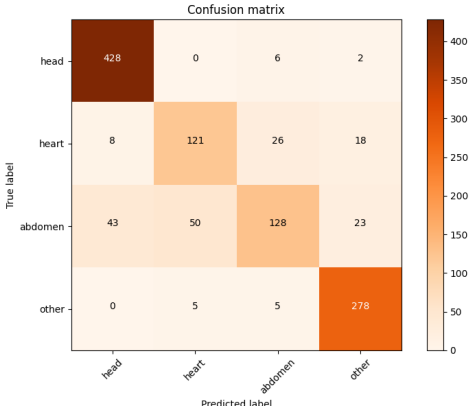
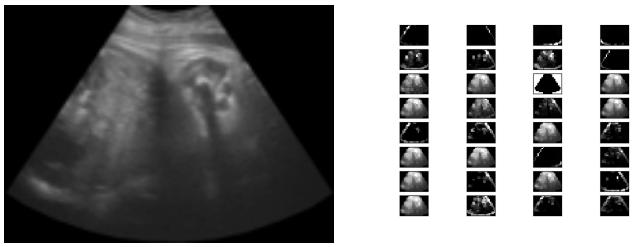


Figure 6: Confusion matrix for the multiclass *final* CNN trained on augmented data.

Class 1 (heart) and class 2 (abdomen) were the most incorrectly classified classes. In Figure 7 we show an example of a frame from class 1 that was classified as class 2, and the resulting feature map of the image.

A similar pattern can also be observed using t-SNE to visualise our data (see Figure 8). The heart and abdomen are close together and may easily be misclassified. The abdomen class is grouped into two main clusters which may make it particularly hard to classify. It is unsurprising that these two classes also have the least amount of data (Figure 1).



(a) Fetal heart US image. (b) Feature map of the image.

Figure 7: An example of a misclassified image - this frame of the fetal heart was classified as the abdomen. Feature maps are generated for the *final* multiclass CNN.

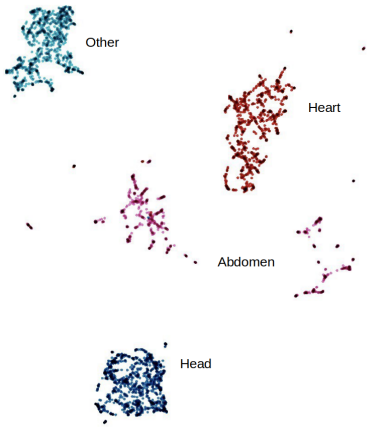


Figure 8: Using t-SNE to visualise high dimensionality data.

V. Conclusion

Both our multiclass and one vs rest *final* CNN achieved testing accuracies significantly higher than the *base* CNN model. However, the one vs rest CNN has a few significant drawbacks - it is much more computationally expensive to train meaning we were unable to finish training cross validation models. Most importantly, even though we have generated a balanced dataset, the binary nature of one vs rest classification means that there are far more negative labels than positive [22] (for example, when training to recognise class 0, the negative frames are classes 1,2 and 3 combined). One vs rest classifiers are most useful in situations where a frame could potentially belong to multiple classes (e.g. a frame of the heart can belong to the heart class, the organ class and the thorax class), as softmax cannot be used in these situations.

Possible improvements to our methodology include using a random search algorithm with cross validation to speed up hyperparameter optimisation [23]. A more aggressive data augmentation strategy including left-right flips and removal of random chunks of image data may make the CNN more robust and further reduce overfitting.

REFERENCES

[1] M. Whitworth, L. Bricker, and C. Mullan, "Ultrasound for fetal assessment in early pregnancy," *Cochrane database of systematic reviews*, no. 7, 2015.

[2] L. Bricker and J. P. Neilson, "Routine ultrasound in late pregnancy (after 24 weeks' gestation)," *Cochrane database of systematic reviews*, no. 2, 2007.

[3] K. H. Nicolaides, G. Azar, D. Byrne, C. Mansur, and K. Marks, "Fetal nuchal translucency: ultrasound screening for chromosomal defects in first trimester of pregnancy." *Bmj*, vol. 304, no. 6831, pp. 867–869, 1992.

[4] J. Siemer, N. Egger, N. Hart, B. Meurer, A. Müller, O. Dathe, T. Goecke, and R. Schild, "Fetal weight estimation by ultrasound: comparison of 11 different formulae and examiners with differing skill levels," *Ultraschall in der Medizin-European Journal of Ultrasound*, vol. 29, no. 02, pp. 159–164, 2008.

[5] K. Nicolaides, K. Spencer, K. Avgidou, S. Faiola, and O. Falcon, "Multicenter study of first-trimester screening for trisomy 21 in 75 821 pregnancies: results and estimation of the potential impact of individual risk-orientated two-stage first-trimester screening," *Ultrasound in Obstetrics and Gynecology: The Official Journal of the International Society of Ultrasound in Obstetrics and Gynecology*, vol. 25, no. 3, pp. 221–226, 2005.

[6] Q. Huang, F. Zhang, and X. Li, "Machine learning in ultrasound computer-aided diagnostic systems: a survey," *BioMed research international*, vol. 2018, 2018.

[7] B. Rahmatullah, A. T. Papageorghiou, and J. A. Noble, "Image analysis using machine learning: Anatomical landmarks detection in fetal ultrasound images," in *2012 IEEE 36th Annual Computer Software and Applications Conference*. IEEE, 2012, pp. 354–355.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[9] P. M. Cheng and H. S. Malhi, "Transfer learning with convolutional neural networks for classification of abdominal ultrasound images," *Journal of digital imaging*, vol. 30, no. 2, pp. 234–243, 2017.

[10] J. Chi, E. Walia, P. Babyn, J. Wang, G. Groot, and M. Eramian, "Thyroid nodule classification in ultrasound images by fine-tuning deep convolutional neural network," *Journal of digital imaging*, vol. 30, no. 4, pp. 477–486, 2017.

[11] K. Lekadir, A. Galimzianova, A. Betriu, M. del Mar Vila, L. Igual, D. L. Rubin, E. Fernández, P. Radeva, and S. Napel, "A convolutional neural network for automatic characterization of plaque composition in carotid ultrasound," *IEEE journal of biomedical and health informatics*, vol. 21, no. 1, pp. 48–55, 2016.

[12] C. Schaffer, "Selecting a classification method by cross-validation," *Machine Learning*, vol. 13, no. 1, pp. 135–143, 1993.

[13] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[14] P. Y. Simard, D. Steinkraus, J. C. Platt *et al.*, "Best practices for convolutional neural networks applied to visual document analysis."

[15] D. Castelvecchi, "Can we open the black box of ai?" *Nature News*, vol. 538, no. 7623, p. 20, 2016.

[16] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[17] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in neural information processing systems*, 2011, pp. 2546–2554.

[18] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International conference on artificial neural networks*. Springer, 2010, pp. 92–101.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[22] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[23] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol. 13, no. Feb, pp. 281–305, 2012.