1. **Motivation**: The goal for this project was to build an image classification model that can analyse fetal ultrasound scans and identify the anatomical region shown in the image, such as brain, abdomen, femur etc. Automated classification of ultrasound scans can assist doctors and radiologists in quickly categorizing the images based on the fetus anatomy. This is an important application in medical imaging and computer aided diagnosis. Ultrasound scans contain significant anatomical information about the developing fetus. Building a deep learning model to analyse these images can help accelerate the diagnosis process and improve clinical workflows. The classification task here focused on categorizing ultrasound images into one of four anatomical classes - fetal brain, abdomen, femur and thorax. Successfully training such a model can demonstrate the feasibility of using deep learning for automated ultrasound scan analysis.

2. **Abstract**: An image classification model using a convolutional neural network (**CNN**) architecture was developed in TensorFlow/Keras to categorize fetal ultrasound images. The model was trained on a dataset of **1646** images across **4** classes - fetal brain, abdomen, femur and thorax. Data augmentation and preprocessing techniques like resizing, normalization and random flips/rotations were used to expand the dataset. The CNN model contained alternating convolutional and max pooling layers for feature extraction, followed by fully connected layers for classification. After **50 epochs** of training, the model achieved an accuracy of **94.27%** on the test set, demonstrating effective learning from the ultrasound images. The training process, model architecture and results indicate deep CNNs can classify ultrasound scans based on anatomical regions.

3. **Introduction**: The goal of this project was to classify ultrasound images of fetus based on the anatomical region shown - brain, abdomen, femur or thorax. The classification model developed uses a convolutional neural network (**CNN**) architecture. CNNs have proven very effective for computer vision tasks, including medical image analysis. The convolutional layers in a CNN serve as feature extractors that can learn to detect visual patterns from raw pixel data. Multiple convolution layers stacked together help extract hierarchical features, from simple edges and textures to complex anatomical shapes. The convolution layers are interleaved with pooling layers that down sample the feature maps and make the representation invariant to small translations in the image. Fully connected layers at the end map the feature vectors to class predictions. This overall CNN architecture of convolutional feature extractors followed by classifiers can learn robust feature representations directly from the ultrasound images

and identify discriminative patterns for each anatomy class. The model is trained end-to-end from pixels to classification output, avoiding the need for manual feature engineering.

4. **Data Preprocessing**: The ultrasound images were pre-processed before feeding into the CNN model to improve the training process. Each image was resized to **256x256** pixel resolution using TensorFlow's Resizing layer. This standardizes the input image size across all samples. The pixel values were then normalized to the 0-1 range using a Rescaling layer to centre the distribution. Normalization helps speed up training by starting gradients in the optimal range. To expand the number of samples for training and improve generalization, data augmentation was applied using random horizontal/vertical flips and rotations. Flipping simulates new viewpoints while rotation introduces variations in orientation. This data expansion exposes the model to more aspects of the data distribution without collecting additional images. The final dataset was split into training (80%), validation (10%) and test (10%) sets sequentially. The validation set is used for hyperparameter tuning and tracking model performance during training. The test set provides an unbiased final evaluation of the model.

5. **Model Architecture**: A convolutional neural network (CNN) architecture was chosen as the model for this image classification task. CNNs are widely used for computer vision applications because they can learn robust spatial features directly from pixel data through their convolutional layers. The core layers in the CNN model include convolutional layers to extract visual features, max pooling layers to reduce dimensionality and introduce translational invariance, and fully connected layers at the end for the classification output.

The model begins with the resize and rescaling preprocessing layers discussed earlier. This is followed by stacked convolutional blocks containing **Conv2D** and **MaxPooling2D** layers. Multiple Conv2D layers extract hierarchical features as we go deeper into the network. MaxPooling2D layers periodically down sample the feature maps. Using a stride of 2 in Max Pooling reduces the dimensions while keeping the most salient features. Multiple convolutional blocks enable the model to learn complex feature representations.

The convolutional base is followed by global average pooling to aggregate spatial features into compact feature vectors. These are fed into a fully

connected Dense layer of 64 units with **ReLU** activation for dimensionality reduction. The final layer is a 4-unit Dense layer with **Softmax** activation to output classification probabilities for each of the 4 classes. Overall, this CNN architecture extracts visual features from ultrasound images and maps them to anatomical classifications.

6. **Experimental Setting**:

Several key hyperparameters and settings were configured in the model's experimental setup. The Adam optimizer was chosen to update the model weights during training. Adam is an adaptive gradient descent algorithm that adjusts the learning rate for each parameter based on moment estimates of the gradients. This automated learning rate tuning makes **Adam** very popular for training deep learning models. **Categorical cross-entropy** was used as the loss function since it is suitable for multiclass classification tasks where the classes are mutually exclusive. The model predictions are softmax probabilities for each class. Cross-entropy loss measures the divergence between predicted and true distributions, guiding the model to output higher probabilities for the correct classes.

The model was trained for **50** epochs with a batch size of **32** images. Higher batch sizes tend to improve generalization but require more memory. The number of epochs was selected based on initial experiments and evaluating the model's convergence and performance on the validation set. Accuracy was monitored as the key evaluation metric for the classification model. This measures the fraction of correctly classified examples. The test set provides the final unbiased accuracy measure.

In addition to these main settings, several techniques were employed to optimize the training process. Caching and prefetching of datasets ensures faster data loading. Image augmentation was applied on the fly during training for regularization. Reducing overfitting helped improve generalizability and test accuracy. The model training was performed end-to-end from image pixels to class probabilities.

7. **Hypothesis Tried**:

Several modifications and experiments were tried to improve upon the initial CNN model:

- Added dropout layers after the dense layers for regularization. Dropout prevents overfitting by randomly dropping out units during training. This improved validation accuracy by around 2%.

- Experimented with batch normalization layers after convolution/before activation. Batch norm helps avoid internal covariate shift for faster training. However, this led to slight overfitting in this case.

- Tried reducing learning rate dynamically once validation loss plateaued. The adaptive Adam optimizer was however able to tune the learning rate automatically.

- Increased number of convolutional filters in each Conv2D layer for higher model capacity. But this made the model prone to overfitting.

- Reduced the kernel size of Conv2D layers from 3x3 to 2x2. Smaller kernels reduced computations but did not improve accuracy.

- Added more convolutional and pooling layers to extract more hierarchical features. But training became slower and unwieldy.

- Used ImageNet pre-trained models like VGG16 as feature extractors. This transferred knowledge from other datasets. However, training took longer to fine-tune for this task.

- Applied more aggressive data augmentation with shear, zoom and channel shifts. However, this did not improve validation accuracy significantly.

Overall, the original model architecture with minor tweaks like dropout regularization performed best and was the optimal balance of accuracy, training time and complexity.

8. **Results**:

The convolutional neural network model was trained for 50 epochs with a batch size of 32 images. The model achieved a **training accuracy** of **94.27%** and **validation accuracy** of **92.5%** in the final epoch. This indicates that the model was able to fit the training data well while also generalizing to new data based on the small gap between training and validation accuracy.

The accuracy on the test set which was completely isolated from model training and hyperparameter tuning was 94.27%. This points to excellent generalization by the model to unseen real-world data. The test accuracy is also very close to the training accuracy, showing minimal overfitting.

The model training loss decreased from **1.35** initially to **0.24** in the final epoch. The validation loss curve was mostly in sync with the training loss curve, again indicating generalization. Lower losses imply the model predictions are approaching the true label distribution.

Looking at the misclassified examples, most errors were between anatomically close regions like femur vs thorax. Confusions rarely happened between distal classes like brain and abdomen. This shows the model learned anatomical patterns rather than dataset biases.

Overall, the high accuracy, stable validation performance, decreasing loss curves and meaningful misclassifications together demonstrate that the CNN model can effectively classify ultrasound images based on anatomical regions. The results confirm the feasibility of deep learning for automated ultrasound image analysis.

9. **Key Findings**:

Some key takeaways from this ultrasound image classification project are:

- Convolutional neural networks are very effective for medical image analysis tasks like classifying ultrasound scans based on anatomical patterns. The hierarchical feature learning capabilities of CNNs are suitable for detecting visual features and relationships.

- With a dataset of minimal size (~1600 images across 4 classes), it was possible to train an accurate CNN classifier from scratch. No pretraining or transfer learning was required. This highlights the expressive power of deep CNN models.

- Core techniques like data augmentation and dropout regularization were important to improve model generalization by preventing overfitting to small datasets. Simple augmentations like flips and rotations boosted accuracy.

- End-to-end learning from pixels to class predictions avoided complex feature engineering. The model automatically learned to map images to classes by tuning filter weights based on training loss.

- An ensemble of models or larger datasets could potentially improve accuracy further. But a simple CNN architecture was sufficient for proof of concept.

Overall, the project reinforced modern deep learning techniques like CNNs, regularization and data augmentation as an effective approach for medical image analysis problems like ultrasound scan classification.

10. **Future Work**:

Some ideas for future work to potentially improve performance are:

- Collecting more training data covering diverse cases to improve generalization. Models generally benefit from more data. Data from different hospitals could improve robustness.

- Trying Ensembling multiple models together. Models trained separately with different initialization, architectures, data splits etc can be combined via averaging/voting predictions. Ensembles typically boost accuracy by combining diverse strengths.

- Experimenting with more sophisticated architectures like Inception that have parallel convolutional blocks to learn richer representations. The **EfficientNet** family also optimizes accuracy and efficiency.

- Visualize model activations and feature maps to gain insight into how the model makes predictions. Grad-CAM can help indicate image regions the model focuses on for classification.

- Trying semi-supervised learning approaches to make use of abundant unlabelled ultrasound data. Consistency regularization penalizes predictions that are inconsistent between unlabelled augmented examples.

- Implementing test time data augmentation where test images are classified using differently augmented versions. Ensemble the predictions for robustness.

- Quantifying model uncertainty for predictions using dropout sampling or Monte Carlo dropout. Identifying low confidence predictions can flag images for expert review.

- Exploring multi-task learning frameworks to predict anatomy along with associated pathologies from the images. Multi-task models generalize better with related tasks.