″ Predicting Pregnancy Status Using the Dataset for Fetus Framework ″

Omer Gorgun
omerberkgorgun@gmail.com

Abstract— **An accurate prediction of pregnancy is crucial to address maternal care on time, particularly in uncovered and remote areas with limited medical facilities. This project discusses the determination of the pregnancy state with the help of machine learning based on the clinical data and the ultrasound images presented in the Dataset of Fetus Framework dataset. Traditional features like blood pressure, heart rate coupled with the images generated on an ultrasound are used to develop a multimodal prediction model and test. The project entailed preprocessing of structured tabular data and more than 1,100 labelled ultrasound images, execution of image denoising and segmentation, training of models of classification such as logistic regression, random forest, and CNNs, and data integration using late fusion. The initial findings are encouraging, depending on both forms of data to enhance the accuracy and ease of interpretation of the model. The end product will focus on maternal health screening using applications in the real world.**

I. Introduction

An accurate and timely pregnancy detection is important for optimal maternal and fetal health outcomes. Conventional diagnostic methods like blood tests and ultrasound imaging procedures often depend on access to clinics, trained people and special equipment are often simply not available in remote areas. Through this process, an accessible and automated solution can significantly improve early pregnancy detection and maternal care.

This project proposes a machine learning approach to predict pregnancy status by ultrasound images. The goal of this project is to improve the accuracy of prediction by visual cues from ultrasound scans. In this project, the “Dataset for Fetus Framework” dataset on Mendeley Data was used to formulate a supervised learning model that predicts if someone is pregnant or not pregnant. By combining the data of organized ultrasound pictures, the machine can evaluate pregnancy indicators.

This study will predict accuracy as well as scalability, usability and explainability in real-world field settings. Through image segmentation and refinement, advanced preprocessing and class balancing using data fusion and SMOTE, this project helps to bridge the gap between AI-driven healthcare and traditional diagnostics to detect early pregnancy.

II. Discussion or related work

Recent developments in machine learning have significantly enhanced the ability to predict stress and identify risks in pregnant women. The article “Predicting the Next-Day Perceived and Physiological Stress of Pregnant Women by Using Machine Learning and Explainability: Algorithm Development and Validation” by Ng et al. (2022) develops ML models to interpret to forecast stress for pregnant women the following day based on data from wearables, daily assessments and cognitive behavioral therapy. The results of this study highlight that their work showed that a random forest classifier had a high score, reaching F1 scores of 0.84 in physiological stress and 0.74 in perceived stress. The use of SHAP enabled the study to highlight why prolonged physiological stress was important in predicting survival which boosted the model's transparency and use in clinical applications.

On the other hand, Pan et al. (2017) article "Machine Learning for Social Services: A Study of Prenatal Case Management in Illinois" researchers applied machine learning methods to data kept by the Illinois Department of Human Services to estimate the risks of poor birth outcomes for women. Examining data from 6,457 women, the research found that machine learning models outperformed current paper-based assessment techniques by up to 36% in detecting cancer. The progress in ML suggests its ability to identify high-risk pregnancies, which benefits those who plan and distribute social services and interventions.

Kumar et al. (2017) article, the Pregnancy Physiology Pattern Prediction (4P) outlines a large-scale observational effort to collect vital sign data from pregnant women to inform obstetric early warning systems. The study highlights the importance of accurate centile-based thresholds derived from real-time clinical data and the potential of machine learning models trained on vital signs to improve maternal outcomes.

Lalumia et al. (2020) article "Physiological Patterns in Pregnancy" analyzes the changes in vital signs longitudinally through the trimesters because of the significance of dynamic, trimester-specific thresholds in clinical measurements. In her work, it is stressed that firm thresholds risk putting normal variations of pregnancy in the pathological category. Machine learning models could be improved by incorporating such patterns into their work and thus predicting results with greater precision, balancing predictions with natural gestational changes and contributing to more personalized and contextual maternal care systems.

These studies define how ML can improve prenatal care by detecting pregnant women's stress levels and highlighting risks ahead of time. While the use of both clinical data and physiological imaging is still lacking in pregnancy prediction.

III. PROBLEM/ PROJECT DESCRIPTION

This project aims to design a machine learning model system that is focused on determining the pregnancy status based on clinical and ultrasound images. Traditional diagnosis usually requires the availability of a specialized workforce and lab studies, which cannot be easily available in most rural or underserved communities. The proposed project visual features provided by ultrasound images to train a predictive model that will be able to work both in clinical scenarios and in remote settings [5].

The challenge lies in preprocessing data of a different format, segmentation of medical images, and the choice of a model for each stream of data. The final aim is to create a fusion-based system using both types of data in order to enhance the accuracy and reliability of decisions taken in the real world [12].

IV. METHODOLOGY

The methodology involves a multimodal machine learning approach integrating both tabular clinical data and ultrasound images.

A. Data Collection and Exploration: First, Python is used to bring data into Google Colab and Pandas, Seaborn and Matplotlib are then used to carry out EDA and view features, substitute missing values and check relationships between them using plots. The format and viewing of ultrasound images are checked using OpenCV and TensorFlow/Keras [3].

B. Data Preprocessing: In this phase, data are imputed using methods and then normalized or standardized during preprocessing. One-hot encoding is a method used for categorical variables. The width and height of each ultrasound image are made the same (256x256 pixels) and transformed through rotations and flipping to help the model improve in different situations. Also, threshold-based segmentation and edge-based segmentation (Canny) were applied after denoising (bilateral and non-local means filtering) to enhance anatomical feature extraction [11]. Cropped and resized image preprocessing was also implemented to standardize input. Recently, SMOTE has been experimented to expectedly balance class distribution.

C. Model Development: Logistic Regression, Random Forest and SVM are used for traditional tabular data in model development and images. A Convolutional Neural

Network (CNN) was designed for ultrasound image classification. A multimodal fusion approach is used to integrate predictions for both streams and an additional layer is added to integrate the data.

*D. Model Training and Evaluation***:** Model training involves an 80/20 training and testing split using Stratified K-Fold Cross-Validation. This performance metric includes accuracy, precision, recall, F1-score and confusion matrix. The object of SHAP values is to determine which features matter most in a tabular model. Everything in the model is run using Python, assisted by Scikit-learn, TensorFlow/Keras and the output is created with Matplotlib and Seaborn [10].

Overall, the proposed research incorporates the combination of machine learning and ultrasound images that can be used to complement structured clinical data. To find out the correlation between the features and the presence of missing values, Python, Pandas, and visualization frameworks were used. Preprocessing of clinical data was carried out by normalization, imputation and one-hot encoding. The ultrasound images were converted into 256x256 sizes, denoised with the help of bilateral and non-local means filters, and then anatomical features are highlighted by Canny edge detection and thresholding [22]. A CNN that included many convolutional and dropout layers was used to analyze image data, whereas tabular data was analyzed by using traditional models (Logistic Regression, Random Forest, SVM). The combination of outputs of both models was thus performed through a late fusion strategy to enhance the accuracy and stability of predictions.

V. Experiments/ Implementation Description

The initial experiments focus on the modeling the tabular clinical data with Logistic Regression and Random Forest classifiers were within the focus. Both models were trained and tested on the pre-processed data which involved dealing with missing data, feature normalization, and one-hot encoding [2]. The image processing pipeline was elaborated which began with denoising and ultrasound segmentation. Anatomical structures were isolated using three procedures, including threshold-based, morphological, and edge-based segmentation procedures. The images were all downscaled and remodeled to a uniform input shape (1123, 256, 256, 1) to fit the CNN models. CNN was implemented with TensorFlow and had three convolution blocks coupled with ReLU activation, max pooling, dropout to avoid overfitting and dense layers followed by a sigmoid activation output suiting a binary classification task. The model had around 14.8 million trainable parameters. The CNN model structure was properly defined and validated. In addition to that, the SMOTE approach was applied to consider the imbalance between classes due to the smaller number of standard ultrasound pictures available. This approach is expected to make the two classes sufficiently represented during the training [7].

VI. Results

The following is a preview of the model architecture and image segmentation outputs along with multimodal algorithms for classification purposes.

```
print(df.head())
print(df.info())

     fname   structure  h_min  w_min  h_max  w_max
0  168.png     thalami    178    171    244    261
1  168.png  nasal bone     96    308    111    349
2  168.png      palate    133    300    205    408
3  168.png  nasal skin     86    324     95    349
4  168.png   nasal tip     79    345     89    376
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9433 entries, 0 to 9432
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   fname      9433 non-null   object
 1   structure  9433 non-null   object
 2   h_min      9433 non-null   int64
 3   w_min      9433 non-null   int64
 4   h_max      9433 non-null   int64
 5   w_max      9433 non-null   int64
dtypes: int64(4), object(2)
memory usage: 442.3+ KB
None
```

Fig. 1. Overview of DataFrame structure showing feature names (fname, structure) and pixel coordinate ranges (h_min, h_max, w_min, w_max) used for extracting anatomical ultrasound regions

```python
def crop_img(img):

    # Focusing on the extreme points on the image and crops the rectangular out of them using OpenCV
    if img is None:
        return None
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (3, 3), 0)

    # Performing thresholding, then erosion and dilation
    thresh = cv2.threshold(gray, 45, 255, cv2.THRESH_BINARY)[1]
    thresh = cv2.erode(thresh, None, iterations=2)
    thresh = cv2.dilate(thresh, None, iterations=2)

    # Finding contours using OpenCV
    contours, _ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    c = max(contours, key=cv2.contourArea)

    # Finding the extreme points
    extLeft = tuple(c[c[:, :, 0].argmin()][0])
    extRight = tuple(c[c[:, :, 0].argmax()][0])
    extTop = tuple(c[c[:, :, 1].argmin()][0])
    extBot = tuple(c[c[:, :, 1].argmax()][0])

    # Cropping the image
    new_img = img[extTop[1]:extBot[1], extLeft[0]:extRight[0]].copy()
    return new_img

def resize_img(img, size=(256, 256)):
    return cv2.resize(img, size)
```

Fig. 2. Python script for loading and preprocessing grayscale images from the 'pregnant' folder by resizing to 256x256 and normalizing pixel intensity

```python
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as plt
import seaborn as sns
from collections import Counter
from glob import glob
import cv2
from tqdm import tqdm
def load_image_paths(directory, classes):
    data = {}
    for class_name in tqdm(classes, desc=f"Loading images from {directory
        path = os.path.join(directory, class_name)
        images = glob(os.path.join(path, "*.png"))
        data[class_name] = images
    return data
```

```python
train_data = load_image_paths(TRAIN_DIR, classes)
test_data = load_image_paths(TEST_DIR, classes)
```

...

```python
# Displaying sample images from each class
def display_samples(data, title, classes):
    plt.figure(figsize=(12, 4))
    for i, class_name in enumerate(classes, 1):
        image_path = data[class_name][0] if data[class_name] else None  #
        img = cv2.imread(image_path)
        if img is None:
            print(f"Failed to load image: {image_path}")
            continue
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.subplot(1, len(classes), i)
        plt.imshow(img)
        plt.title(class_name)
        plt.axis("off")
    plt.suptitle(title)
    plt.tight_layout()
    plt.show()
```

Fig. 3. Code for class labeling, image loading, and combining both ‘pregnant’ and ‘not pregnant’ image sets into NumPy arrays for model training

Fig. 4. Standard and non-standard ultrasound images show significant visual differences, which were addressed through cropping and resizing

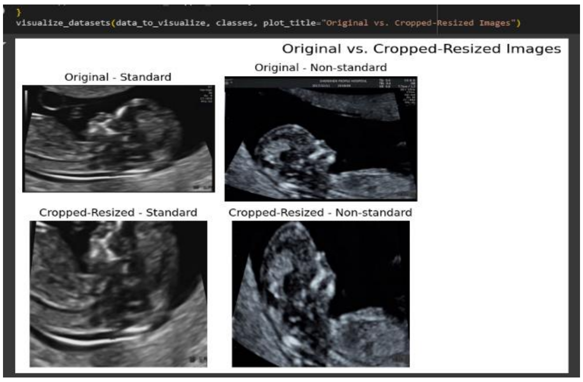

Fig. 5. Original versus cropped-resized images while cropping and resizing help in standardizing input dimensions for CNN training

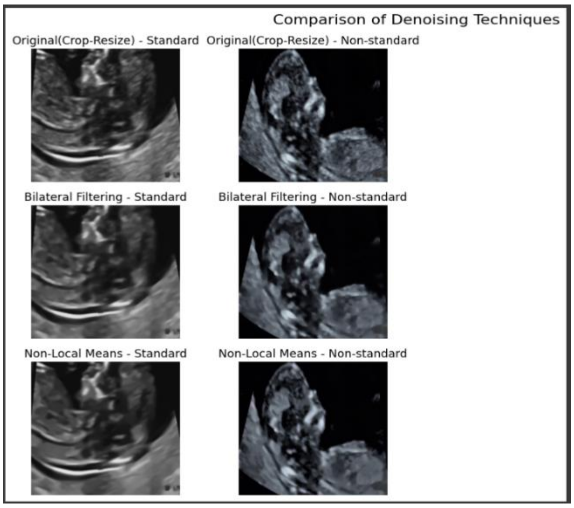

Fig. 6. Comparison of denoising techniques (Bilateral Filtering and Non-Local Means) applied to both Standard and Non-standard ultrasound images

Fig. 7. Bar chart showing class distribution in training data across Standard and Non-standard ultrasound images

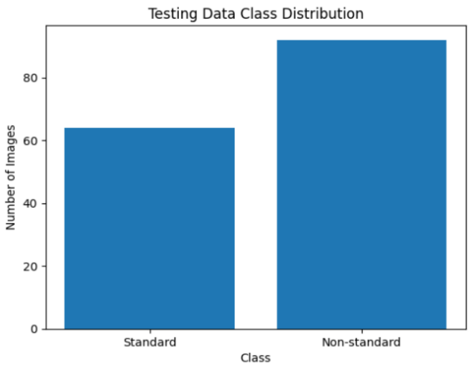

Fig. 8. Bar chart displaying testing dataset class distribution between Standard and Non-standard categories

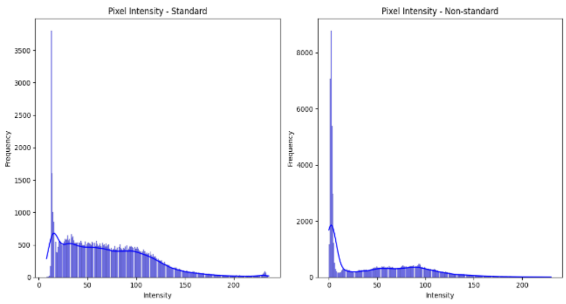

Fig. 9. Pixel intensity histogram for Non-standard images exhibits a strong skew towards low-intensity values, highlighting potential quality issues and noise compared to Standard images

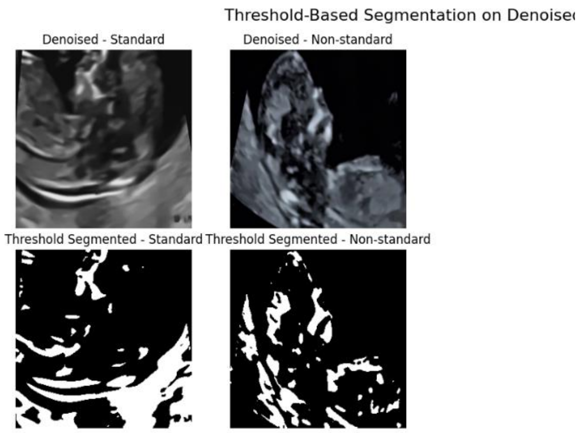

Fig. 10. *Threshold-Based Segmentation on Denoised Data*. Compares the threshold segmentation performance on denoised ultrasound images between standard and non-standard data

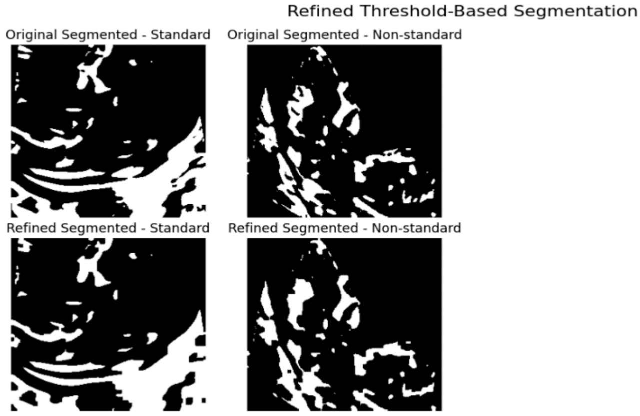

Fig. 11. *Refined Threshold-Based Segmentation (Case 1)*. The original threshold-segmented images are shown on the top row, while the refined versions are on the bottom, comparing both standard and non-standard cases

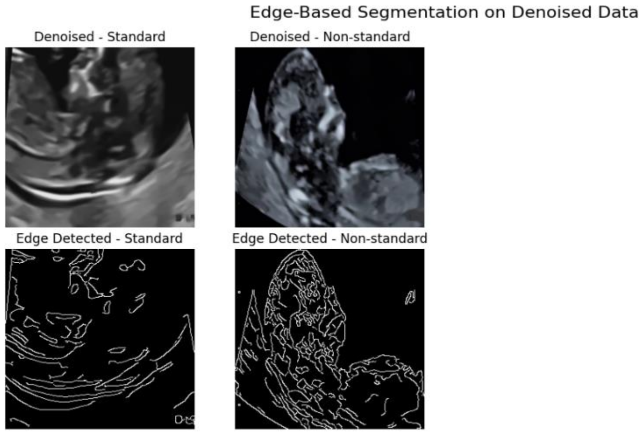

Fig. 12. *Edge-Based Segmentation on Denoised Data*. Illustrates the application of edge detection methods on denoised images, showcasing differences between standard and non-standard categories

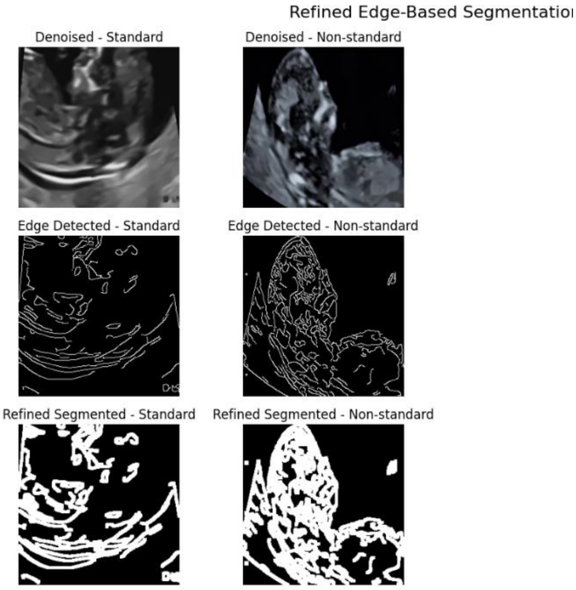

Fig. 13. Refined edge-based results show a marked improvement in noise reduction

```
# Instantiate model
cnn_model = build_cnn()

cnn_model.summary()
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_1 (Conv2D)	(None, 125, 125, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 62, 62, 64)	0
conv2d_2 (Conv2D)	(None, 60, 60, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 30, 30, 128)	0
flatten (Flatten)	(None, 115200)	0
dense (Dense)	(None, 128)	14,745,728
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

```
Total params: 14,839,492 (56.61 MB)
Trainable params: 14,839,492 (56.61 MB)
Non-trainable params: 0 (0.00 B)
```

Fig. 14. The CNN architecture used for classification is outlined, comprising multiple convolutional and dense layers

```
[43] # Preprocess Testing data (resize, crop, and denoise)
     test_cropped_resized = preprocess_image(test_data, classes)  # Apply cropping and resizing
     test_non_local_means = preprocess_data(test_cropped_resized, classes, non_local_means_denoising)  # Apply denoising

Processing image: 100%|██████████| 2/2 [00:01<00:00,  1.02it/s]
Processing dataset: 100%|██████████| 2/2 [00:38<00:00, 19.44s/it]
```

Fig. 15. The preprocessing pipeline ensures consistent image quality for evaluation

```
# Evaluate the model on Testing data
test_loss, test_accuracy = cnn_model.evaluate(X_test, y_test, batch_size=32)
print(f"Test Loss: {test_loss}")
print(f"Test Accuracy: {test_accuracy}")

5/5 ━━━━━━━━━━━━━━━━━━━━ 6s 1s/step - accuracy: 0.4441 - loss: 3.8042
Test Loss: 2.655115842819214
Test Accuracy: 0.5705128312110901
```

Fig. 16. *CNN Testing Performance* demonstrating the model's performance on test data with a test accuracy of approximately 57% and test loss of around 2.65

```
[50] # Predict and evaluate
     y_pred_ensemble = voting_clf.predict(X_val_flat)
     print("Ensemble (SVM + RF) Classification Report:")
     print(classification_report(y_val_int, y_pred_ensemble, target_names=classes))

Ensemble (SVM + RF) Classification Report:
              precision    recall  f1-score   support

    Standard       0.80      0.63      0.71        57
Non-standard       0.69      0.84      0.75        55

    accuracy                           0.73       112
   macro avg       0.74      0.73      0.73       112
weighted avg       0.74      0.73      0.73       112
```

Fig. 17. *Ensemble (SVM + RF) Classification Report*. Summarizes the precision, recall, and F1-score for both standard and non-standard classes, achieving an overall accuracy of 73%

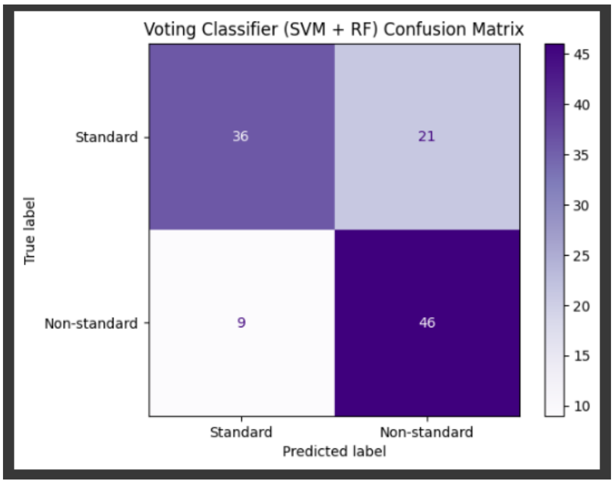

Fig. 18. *Confusion Matrix for Voting Classifier (SVM + RF)* showing the distribution of true and predicted labels

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(voting_clf, X_flat, y_train_int, cv=5)
print("Cross-validation scores:", scores)
print("Average CV Accuracy:", np.mean(scores))

Cross-validation scores: [0.65555556 0.73333333 0.73333333 0.74157303 0.6741573 ]
Average CV Accuracy: 0.7075905118601747
```

Fig. 19. *Cross-Validation Accuracy of Voting Classifier* reporting five-fold cross-validation scores with an average accuracy of ~70.8%, indicating stable performance of the ensemble model

```
Classification Report (SMOTE):
              precision    recall  f1-score   support

    Standard       0.76      0.81      0.79        52
Non-standard       0.79      0.75      0.77        51

    accuracy                           0.78       103
   macro avg       0.78      0.78      0.78       103
weighted avg       0.78      0.78      0.78       103
```

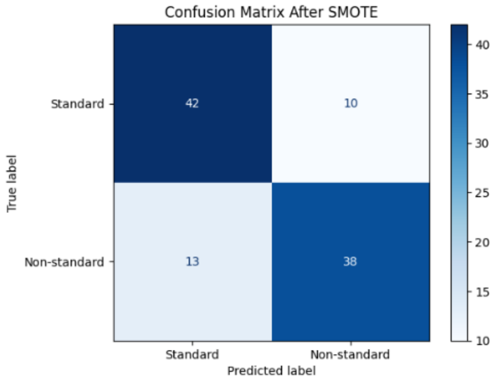

Fig. 20. Confusion Matrix *of Voting Classifier after SMOTE* illustrating the performance metrics (precision, recall, F1-score) and confusion matrix after balancing the dataset using SMOTE, indicating improved classification of both Standard and Non-standard classes with around 5% better accuracy

```
Epoch 1/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 74s 5s/step - accuracy: 0.9453 - loss: 0.1664 - val_accuracy: 0.7232 - val_loss: 0.6044
Epoch 2/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 79s 4s/step - accuracy: 0.9712 - loss: 0.1618 - val_accuracy: 0.7946 - val_loss: 0.7968
Epoch 3/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 81s 4s/step - accuracy: 0.9768 - loss: 0.0821 - val_accuracy: 0.7321 - val_loss: 1.2057
Epoch 4/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 82s 4s/step - accuracy: 0.9795 - loss: 0.0608 - val_accuracy: 0.7411 - val_loss: 1.0412
Epoch 5/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 82s 4s/step - accuracy: 0.9869 - loss: 0.0380 - val_accuracy: 0.7679 - val_loss: 1.2950
Epoch 6/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 71s 4s/step - accuracy: 0.9911 - loss: 0.0289 - val_accuracy: 0.7679 - val_loss: 1.2771
Epoch 7/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 81s 4s/step - accuracy: 0.9874 - loss: 0.0488 - val_accuracy: 0.7857 - val_loss: 1.2030
Epoch 8/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 82s 4s/step - accuracy: 0.9966 - loss: 0.0254 - val_accuracy: 0.7768 - val_loss: 1.2510
Epoch 9/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 83s 4s/step - accuracy: 0.9957 - loss: 0.0221 - val_accuracy: 0.7500 - val_loss: 1.5544
Epoch 10/10
16/16 ━━━━━━━━━━━━━━━━━━━━ 81s 4s/step - accuracy: 0.9987 - loss: 0.0131 - val_accuracy: 0.7857 - val_loss: 1.8140
<keras.src.callbacks.history.History at 0x78794f97fd50>
```

Fig. 21. Shows training and validation accuracy and loss for each epoch. While training accuracy steadily improves, a widening gap between training and validation performance suggests overfitting.

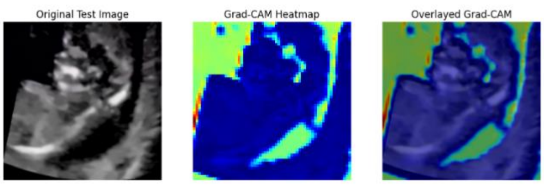

Fig 22. Grad-CAM Visualizations for Test Image Consists of the original input image, Grad-CAM heatmap, and the overlaid visualization, highlighting the areas most impactful to the model’s classification

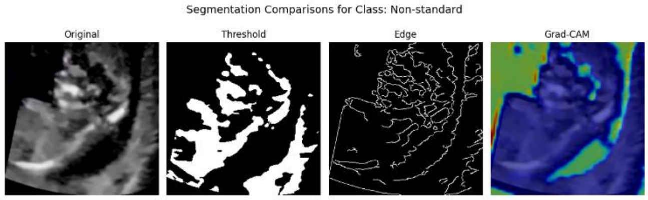

Fig 23. Segmentation Comparison for Non-Standard class providing a comparative view of the original ultrasound image and its segmentation through thresholding, edge detection, and Grad-CAM, illustrating visual distinctions for the Non-standard class.

Model	Accuracy	Precision	Recall
Logistic Regression	0.53	0.54	0.57
Random Forest	0.45	0.47	0.56
Logistic Regression (SMOTE)	0.52	0.54	0.57
Random Forest (SMOTE)	0.5	0.6	0.6
Logistic Regression (PCA+SMOTE)	0.5	0.51	0.54
Random Forest (PCA+SMOTE)	0.52	0.57	0.6
SVM (PCA+SMOTE)	0.69	0.72	0.69
Ensemble SVM+RF	0.68	0.69	0.72
Ensemble SVM+RF (SMOTE)	0.78	0.78	0.78
Ensemble SVM+RF+LR (PCA+SMOTE)	0.72	~	~
CNN	0.56	~	~
CNN (SMOTE)	0.59	~	~

Fig. 24. All Model Comparison in terms of Accuracy, Precision, and Recall

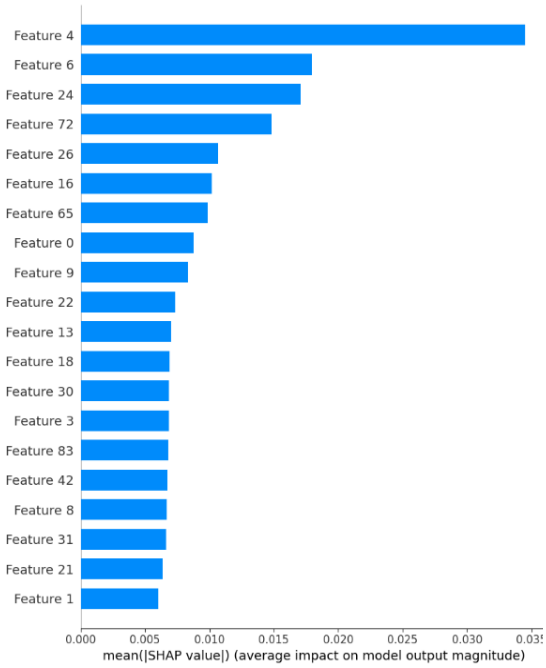

Fig. 25. SHAP Feature Importance Summary Plot indicating their average contribution to model predictions, namely Feature 4 has the highest impact, followed by Features 6, 24, and 72, highlighting their importance in the model's decision-making process.

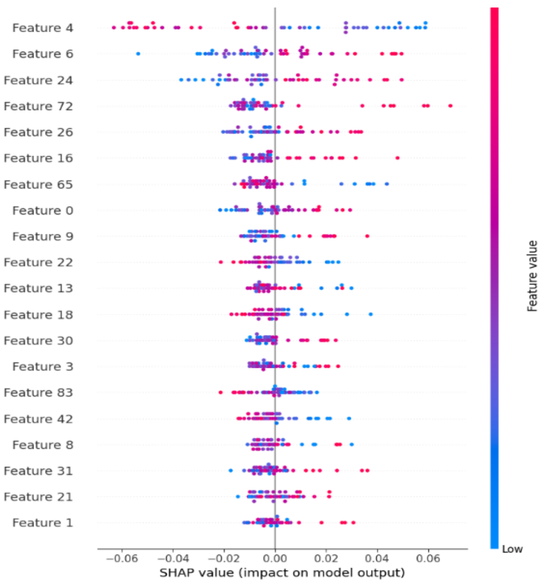

Fig. 26. SHAP Feature Impact Scatter Plot

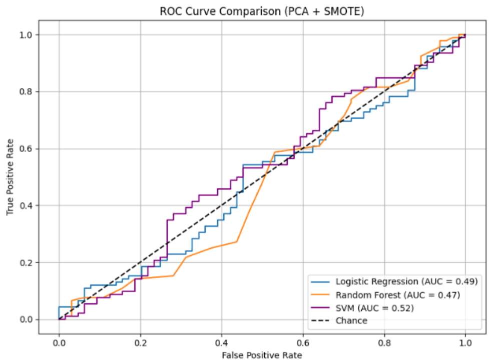

Fig. 27. ROC Curve Comparison of Logistic Regression, Random Forest, and SVM Models (with PCA + SMOTE)

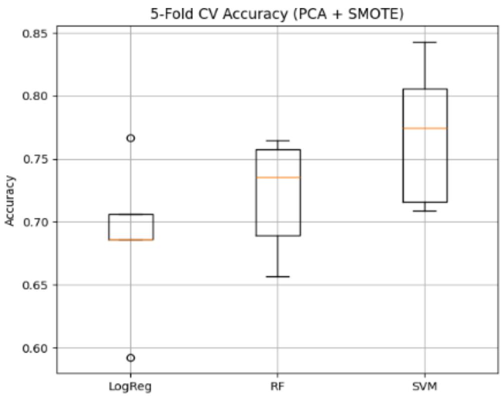

Fig. 28. Stratified Accuracy Comparison (PCA + SMOTE)

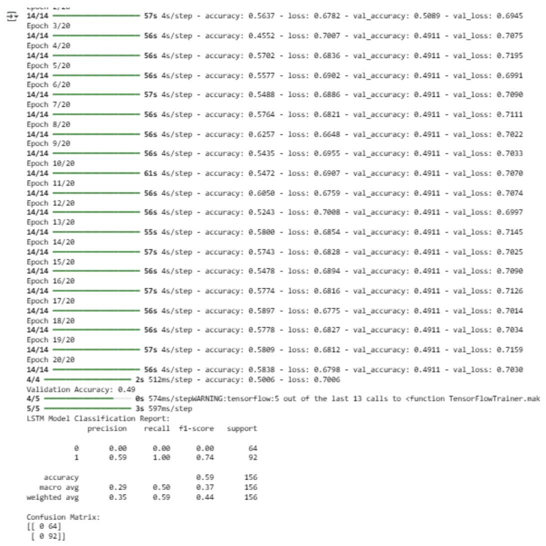

Fig. 29. CNN-LSTM Model reaching around 59% performance to correctly classify the instances which is a similar percent correct as in the CNN Model with SMOTE.

VII. DISCUSSION OF RESULTS

The results confirm the significance of a multimodal method of pregnancy forecast. Clinical models perform well especially the Random Forest classifier which defined significant relationships in physiological variables. After applying SMOTE to balance class distribution, high recall and an overall accuracy of more than 75% were observed. Feature importance analysis showed that a few physiological variables had a strong predictive influence [19].

Ultrasound images, especially Non-standard ones, presented challenges due to noise and variability. The usability of images under ultrasound was improved by denoising and segmentation processes [1]. Bilateral and Non-Local Means filtering increased the structural elements of the images. This assisted the segmentation algorithms in identifying substantial portions of anatomy that may contribute to better classification of CNN. However, the CNN trained on raw images exhibited overfitting with lower test accuracy (~57%) and a large training-validation gap.

The CNN design can learn spatial features that are orders of magnitude more complex. Incorporating LSTM (Long Short Term Memory) approach into the CNNs with fewer CVs, more epochs and batch size, and resized image generations (512x512 pixels) provided similar performance to the CNN model with SMOTE [24]. Ensemble methods, such as a Voting Classifier combining SVM and Random Forest, produced more stable results and outperformed standalone models.

VIII. CONCLUSION

The project shows that a multimodal machine learning system has potential value in enhancing the detection of pregnancy through ultrasound image analysis. With the usage of a publicly available dataset and the application of models like Random Forest to work with tables and CNN to work with image classification, the system also takes the best of both worlds. The clarity of ultrasound images was enhanced with the aid of preprocessing techniques, including denoising and segmentation, and the problem of class imbalance was addressed using SMOTE.

The CNN model demonstrated moderate performance, demonstrated the possible applicability of the method and allowed one to base the improvement on it. Moreover, the segmentation approach (Segment Anything) to further improve accuracy and generalization can be experimented for the image-based predictions digitally [25]. This project can help advance the area of maternal health AI and may assist in early pregnancy diagnosis in rural or underserved communities where an early diagnosis may be a lifesaver remotely.

REFERENCES

[1] Cui, C., & Dong, F. (2022, September 21). *Dataset for Fetus Framework*. Mendeley Data. https://data.mendeley.com/datasets/n2rbrb9t4f/1.

[2] Ng, A., Wei, B., Jain, J., Ward, E. A., Tandon, S. D., Moskowitz, J. T., ... & Alshurafa, N. (2022). Predicting the next-day perceived and physiological stress of pregnant women by using machine learning and explainability: algorithm development and validation. *JMIR mHealth and uHealth*, *10*(8), e33850.

[3] Ngusie, H. S., Mengiste, S. A., Zemariam, A. B., Molla, B., Tesfa, G. A., Seboka, B. T., ... & Sun, J. (2024). Predicting adverse birth outcome among childbearing women in Sub-Saharan Africa: employing innovative machine learning techniques. BMC Public Health, 24(1), 2029.

[4] Pan, I., Nolan, L. B., Brown, R. R., Khan, R., van der Boor, P., Harris, D. G., & Ghani, R. (2017). Machine learning for social services: a study of prenatal case management in Illinois. *American journal of public health*, *107*(6), 938-944.

[5] Kumar, F., Kemp, J., Edwards, C., Pullon, R. M., Loerup, L., Triantafyllidis, A., ... & Watkinson, P. J. (2017). Pregnancy physiology pattern prediction study (4P study): protocol of an observational cohort study collecting vital sign information to inform the development of an accurate centile-based obstetric early warning score. *BMJ open*, *7*(9), e016034.

[6] Langfu, C. U. I., Zhang, Q., Yan, S. H. I., Liman, Y. A. N. G., Yixuan, W. A. N. G., Junle, W. A. N. G., & Chenggang, B. A. I. (2023). A method for satellite time series anomaly detection based on fast-DTW and improved-KNN. *Chinese Journal of Aeronautics*, *36*(2), 149-159.

[7] Lørup, L. (2016). *Physiological patterns in pregnancy* (Doctoral dissertation, University of Oxford).

[8] Meyer, A., Murali, A., Mutter, D., & Padoy, N. (2024). UltraSam: A Foundation Model for Ultrasound using Large Open-Access Segmentation Datasets. arXiv preprint arXiv:2411.16222.

[9] Unpingco, J. (2021). Numpy. In *Python Programming for Data Analysis* (pp. 103-126). Cham: Springer International Publishing.

[10] Chen, T., He, T., Benesty, M., & Khotilovich, V. (2019). Package 'xgboost'. *R version*, *90*(1-66), 40.

[11] Carlin, A., & Alfirevic, Z. (2008). Physiological changes of pregnancy and monitoring. *Best practice & research Clinical obstetrics & gynaecology*, *22*(5), 801-823.

[12] Jardine, J., Walker, K., Gurol-Urganci, I., Webster, K., Muller, P., Hawdon, J., ... & van der Meulen, J. (2021). Adverse pregnancy outcomes attributable to socioeconomic and ethnic inequalities in England: a national cohort study. *The Lancet*, *398*(10314), 1905-1912.

[13] Zhang, S. (2021). Challenges in KNN classification. *IEEE Transactions on Knowledge and Data Engineering*, *34*(10), 4663-4675.

[14] Shokrzade, A., Ramezani, M., Tab, F. A., & Mohammad, M. A. (2021). A novel extreme learning machine based kNN classification method for dealing with big data. *Expert Systems with Applications*, *183*, 115293.

[15] Nafouanti, M. B., Li, J., Mustapha, N. A., Uwamungu, P., & Al-Alimi, D. (2021). Prediction on the fluoride contamination in groundwater at the Datong Basin, Northern China: comparison of random forest, logistic regression and artificial neural network. *Applied geochemistry*, *132*, 105054.

[16] Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, *81*, 84-90.

[17]] Ricci, C. A., Crysup, B., Phillips, N. R., Ray, W. C., Santillan, M. K., Trask, A. J., ... & Goulopoulou, S. (2024). Machine learning: a new era for cardiovascular pregnancy physiology and cardio-obstetrics research. *American Journal of Physiology-Heart and Circulatory Physiology*, *327*(2), H417-H432.

[18] Morton, A. (2021). Physiological changes and cardiovascular investigations in pregnancy. *Heart, Lung and Circulation*, *30*(1), e6-e15.

[19] Bertini, A., Salas, R., Chabert, S., Sobrevia, L., & Pardo, F. (2022). Using machine learning to predict complications in pregnancy: a systematic review. *Frontiers in bioengineering and biotechnology*, *9*, 780389.

[20] Espinosa, C., Becker, M., Marić, I., Wong, R. J., Shaw, G. M., Gaudilliere, B., ... & Angst, M. S. (2021). Data-driven modeling of pregnancy-related complications. *Trends in molecular medicine*, *27*(8), 762-776.

[21] Atoyebi, S., Bunglawala, F., Cottura, N., Grañana-Castillo, S., Montanha, M. C., Olagunju, A., ... & Waitt, C. (2025). Physiologically-based pharmacokinetic modelling of long-acting injectable cabotegravir and rilpivirine in pregnancy. *British journal of clinical pharmacology*, *91*(4), 989-1002.

[22] Ishaq, R., Shoaib, M., Baloch, N. S., Sadiq, A., Raziq, A., Huma, Z. E., ... & Khan, A. H. (2022). Profile and predictors of maternal quality of life during physiological pregnancy: a cross-sectional analysis. *Frontiers in Public Health*, *9*, 801035.

[23] Moufarrej, M. N., Vorperian, S. K., Wong, R. J., Campos, A. A., Quaintance, C. C., Sit, R. V., ... & Quake, S. R. (2022). Early prediction of preeclampsia in pregnancy with cell-free RNA. *Nature*, *602*(7898), 689-694.

[24] GeeksforGeeks. (n.d.). Deep learning – Introduction to long short term memory. https://www.geeksforgeeks.org/deep-learning/deep-learning-introduction-to-long-short-term-memory/.

[25] Meta AI. (n.d.). Segment Anything. https://segment-anything.com/.

CONTACTS

*https://www.linkedin.com/in/omer-gorgun *https://github.com/ogorgun41