IMPROVING EMBRYO PLOIDY PREDICTION ACCURACY USING VOTING ENSEMBLES OF DEEP NEURAL NETWORKS

Report submitted to the SASTRA Deemed to be University
in partial fulfilment of the requirements
for the award of the degree of

Bachelor of Technology

Submitted by

Manasvi A
(Reg. No.: 124156090, CSE(AI & DS)

July 2024

SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 of the UGC Act, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA
THANJAVUR | KUMBAKONAM | CHENNAI

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act,1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

THANJAVUR | KUMBAKONAM | CHENNAI

SCHOOL OF COMPUTING

THANJAVUR – 613 401

Bonafide Certificate

This is to certify that the report titled "Improving Embryo ploidy prediction accuracy using voting ensembles of Deep neural networks" submitted in partial fulfilment of the requirements for the award of the degree of B. Tech Computer science and Engineering with specialization in Artificial Intelligence and Data science to the SASTRA Deemed University, is a bona-fide record of the work done by Ms. Manasvi (Reg. No. 124156090) during the final semester of the academic year 2023-24, in the Harvard Medical School for School of Computing, under my supervision. This report has not formed the basis for the award of any degree, diploma, associateship, fellowship or other similar title to any candidate of any university.

[redacted]

Signature of Project Supervisor

Name with Affiliation: Dr. Hadi Shafiee, Ph.D. Associate Professor, Harvard Medical School, Boston, MA, USA

Date: 31/07/24

Project Viva voce held on 07/08/24

[redacted]

Examiner 1

[redacted]

Examiner 2

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act,1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

THANJAVUR | KUMBAKONAM | CHENNAI

SCHOOL OF COMPUTING

THANJAVUR – 613 401

Declaration

I declare that the project report titled “**Improving Embryo ploidy prediction accuracy using voting ensembles of Deep neural networks**” submitted by us is an original work done by me under the guidance of **Dr. Hadi Shafiee,** Associate professor at Harvard Medical School, Boston, MA, United States during the final semester of the academic year 2023-24, for the School of Computing in Harvard Medical School. The work is original and wherever, we have used materials from other sources, we have given due credit and cited them in the text of the report. This report has not formed the basis for the award of any degree, diploma, associate-ship, fellowship or other similar title to any candidate of any university.

Signature of the candidate(s) : [redacted]

Name of the candidate(s) **:** Manasvi A

Date **:** 31 July 2024

Acknowledgements

I would like to thank our Honorable Chancellor **Prof. R. Sethuraman** for providing me with an opportunity and the necessary infrastructure for carrying out this project as a part of our curriculum.

I would like to thank our Honorable Vice- Chancellor **Dr. S. Vaidhyasubramanianm** and **Dr. S. Swaminathan**, Dean, Planning & Development, for the encouragement and strategic support at every step of our college life.

I extend my sincere thanks to **Dr. R. Chandramouli**, Registrar, SASTRA Deemed University for providing the opportunity to pursue this project.

I extend my heartfelt thanks of **Dr. V. S. Shankar Sriram**, Dean, School of Computing, **Dr. R. Muthaiah**, Associate Dean, Research, **Dr. K. Ramkumar**, Associate Dean, Academics, **Dr. D. Manivannan**, Associate Dean, Infrastructure, **Dr. R. Algeswaran**, Associate Dean, Students Welfare.

My guide **Dr. Hadi Shafiee, Associate professor at Harvard Medical School** was the driving force behind this whole idea from the start. His deep insight in the field and invaluable suggestions helped me in making progress throughout our project work.

I would like to extend my gratitude to the project co-ordinator **Mr. Rajendran. P**, Assistant Professor – I, and **Dr. V. Venkatesh**, Assistant Professor – III, School of Computing, SASTRA University for their guidance and timely communication with regards to the requirements of the project.

I would like to express my gratitude to all the teaching and non-teaching faculties of the School of Computing who have either directly or indirectly helped me in the completion of the project.

I gratefully acknowledge all the contributions and encouragement from my family and friends resulting in the successful completion of this project. I thank you all for providing me an opportunity to showcase my skills through this project.

Table of Contents

List of Figures

ABSTRACT

The aim of this study is to enhance the accuracy to non- enhance the accuracy of non-invasive prediction of embryo ploidy status by employing voting ensembles that combine convolutional neural networks (CNNs), support vector machines (SVMs), and multi-layer neural networks (NNs) with clinical parameters. This retrospective cohort study was conducted at an academic medical centre, including a total of 699 day 5 blastocysts that underwent preimplantation genetic testing for aneuploidy (PGTA - A). Images of blastocysts were classified as euploid or aneuploid using a CNN. Clinical parameters such as maternal age, anti Mullerian hormone (AMH) level, paternal sperm quality, and the number of normally fertilized embryos were incorporated using SVMs and NNs. The primary outcome measure was the accuracy of predicting embryo ploidy status using individual models and voting ensembles. The voting ensembles, which integrated predictions from CNN, SVM, and NN models. Incorporating clinical parameters into the models further improved the prediction performance. Voting ensembles that integrate both image-based and clinical data significantly enhance the accuracy of non-invasive embryo ploidy prediction. This method offers a promising tool for improving embryo selection in assisted reproductive technology (ART), potentially leading to better clinical outcomes.

CHAPTER-1

INTRODUCTION

1.1 UNDERSTANDING ANEUPLOIDY AND EUPLOIDY

Aneuploidy and euploidy are terms used to describe the chromosomal composition of cells. Aneuploidy refers to the presence of an abnormal number of chromosomes within a cell. This condition can lead to various developmental and health issues, including failed implantation, miscarriage, and genetic disorders such as Down syndrome, which is caused by the presence of an extra chromosome 21. Aneuploid embryos are therefore less likely to result in successful pregnancies.

Euploidy, on the other hand, denotes a normal chromosomal number. In humans, this means having 46 chromosomes, arranged in 23 pairs. Euploid embryos are generally more viable and have a higher chance of leading to successful implantation and pregnancy. Identifying euploid embryos for transfer is a key goal in assisted reproductive technologies (ART) to improve pregnancy outcomes and reduce the risk of genetic disorders.

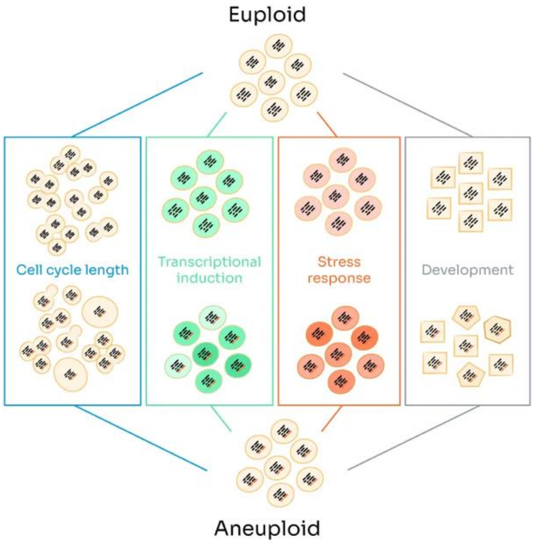


Fig.1.1 Impact of Euploidy and Aneuploidy on Cellular Processes

1.2 TRADITIONAL METHODS OF EMBRYO ASSESSMENT

Embryo morphology was initially the sole method for evaluating embryo quality and viability. Selecting high-quality embryos for transfer is crucial for enhancing pregnancy rates; however, relying solely on morphology does not ensure that an embryo is euploid or that it will successfully implant. Research combining morphological and cytogenetic assessments of cleavage and blastocyst-stage embryos has indicated that chromosomal abnormalities do not significantly affect morphology up to the third day of development. This means many embryos with chromosomal abnormalities can still achieve high morphological grades.

As clinics increasingly adopt time-lapse imaging platforms, numerous studies have explored the relationship between developmental time points, morphologic changes, and aneuploidy rates. These studies have found that assessing morpho kinetic parameters does not significantly enhance the likelihood of selecting euploid embryos. Additionally, the subjective nature of embryo morphology grading remains a significant obstacle, complicating the training and standardization processes among embryologists.

1.3 ADVANCEMENTS IN GENETIC TESTING : PGT-A

The introduction of next-generation sequencing (NGS) and the development of preimplantation genetic testing for aneuploidy (PGT-A) have marked a significant advancement in embryo assessment. PGT-A involves biopsying a few cells from the embryo and analysing them for chromosomal abnormalities using NGS. This method has been shown in randomized controlled trials to improve ongoing pregnancy rates for women aged 35 to 40 compared to morphology-based selection alone.

Despite its benefits, PGT-A presents certain drawbacks. The cost of PGT-A can add approximately $5000 to the already substantial cost of IVF, posing a significant financial burden on patients. Additionally, the waiting period for PGT-A results can delay embryo transfer, necessitating either fresh transfer of an untested embryo or delayed frozen embryo transfer. This delay can result in increased medication use, more ultrasounds, and additional financial strain. Moreover, PGT-A testing is complicated by the presence of embryo mosaicism, where an embryo contains both normal and abnormal cells, making it difficult for physicians and patients to interpret and manage these results. Potential errors in sequencing, insufficient sampling, and the need for re-biopsy further complicate the reliability of PGT-A testing.

1.4 NON-INVASIVE GENETIC TESTING TECHNIQUES

In response to the limitations of invasive biopsy techniques, the field has seen a rapid expansion of non-invasive genetic testing methods. Two promising approaches are spent culture media (SCM) testing and blastocoel fluid sampling (BFS).

SCM testing involves collecting the culture media in which embryos have been growing and analyzing the cell-free DNA within it to assess the ploidy status of the embryos. While this technique preserves embryo integrity, it currently lacks diagnostic uniformity, with reported concordance rates with trophectoderm biopsy or whole embryo sequencing ranging from 30.4% to 90%. Standard trophectoderm biopsy consistently outperforms SCM in direct comparisons, indicating that SCM is on yet a reliable standalone method.

BFS involves aspiring the blastocoel fluid from the embryo. Although less invasive than trophectoderm biopsy, BFS presents significant technical challenges in sample collection and analysis, yielding inferior predictive results compared to SCM. Both BFS and SCM are limited by the low quality and quantity of DNA available for analysis, making these techniques far from being ready for commercial use. The inconsistencies in accuracy between these methods make them unreliable for critical decisions regarding embryo transfer, vitrification, or discard.

1.5 THE ROLE OF AI IN EMBRYO SELECTION

Artificial Intelligence (AI) has emerged as a powerful tool in addressing the complexities of embryo selection. AI leverages machine learning and computational statistics to perform tasks requiring complex interpretation or processing, previously achievable only through human intelligence. In the field of reproductive endocrinology and infertility (REI), AI shows immense promise in enhancing embryology quality assurance and control and improving clinical pregnancy outcomes.

AI can significantly reduce the subjectivity involved in embryo morphology grading by providing consistent and objective assessments. This capability is particularly valuable in training and standardizing decision-making among embryologists. Previous image-based AI models have shown promising results in predicting euploidy by analysing blastocyst images alongside patient metadata, such as age and laboratory characteristics.

1.6 METHODOLOGY USED

In this study, we describe an AI system that integrates blastocyst images with patient characteristics into a voting ensemble composed of convolutional neural networks (CNN), support vector machines (SVM), and multi-layer neural networks (NN). This approach aims to improve the accuracy of predicting embryo ploidy status. By utilizing non-morphological parameters in ploidy prediction, this AI system demonstrates how data flows through different parameters in the voting ensemble. Optional parameters are used in comparisons, highlighting the potential of AI in revolutionizing embryo selection and improving clinical outcomes.

The AI system utilizes a combination of blastocyst images and patient data, processed through a voting ensemble of CNNs, SVMs, and NNs. This integration allows for a comprehensive analysis that can predict embryo ploidy status more accurately than traditional methods. The use of AI in this context not only aims to enhance the accuracy of embryo selection but also to standardize the process, reducing the subjectivity that currently plagues morphological assessments.

CHAPTER-2

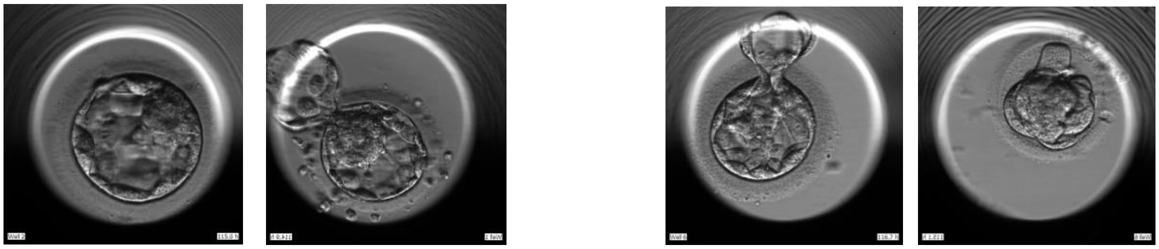
MATERIALS AND METHODS

2.1 DATA COLLECTION AND HANDLING

Data was gathered at Massachusetts General Hospital (MGH) Fertility Centre in Boston, Massachusetts, with approval from the Institutional Review Board (IRB#2019P001000 and 2022P002955). Sperm quality was categorized as follows:

- 1 “Excellent” : Total motile count (TMC) from the raw specimen was greater than 15 million, and sperm concentration, motility, and strict morphology were within the normal ranges specified by the WHO 5th edition.
- “Good” : TMC was greater than 15 million, but sperm concentration, motility, or strict morphology fell outside the normal range.
- “Fair” : TMC ranges from 5 to 15 million
- “Poor” : TMC was below 5 million

Sample data :



(a) (b)

Fig.2.1 Images of (a) Aneuploidy cells and (b) Euploid cells at different time points

A commercial time-lapse imaging device (Embryo Scope, Vitrolife) was used to record time-lapse imaging movies of embryos. Utilizing a Leica 20× objective, the imaging system took pictures every ten minutes while being illuminated by a single 635 nm LED. The imaging system software was used to output each embryo as a movie (.avi). Videos were converted into the appropriate picture frame at every post-insemination timepoint. After being retrieved, each image had 250 × 250 pixels and was cropped to 210 × 210 pixels in order to eliminate any possible identifiers that could have been in the frame. The datasets contained out-of-focus photos, which were utilized in training and testing. The only embryonic photos eliminated from the analysis were those that were totally undetectable. We categorized the patients into groups based on about 18-minute intervals because the picture collection timepoints for each patient were inconsistent. This study made use of 248 patients' 699 embryos' imaging data in all.

2.2 METHODOLOGY

A kind of machine learning model called a voting ensemble combines data from several different models to enhance the overall performance of a task, perhaps outperforming any one of the individual models employed in the ensemble. Convolutional neural networks (CNNs), support vector machines (SVMs), and multi-layer neural networks (NNs) were the three models that comprised the hard voting ensemble.

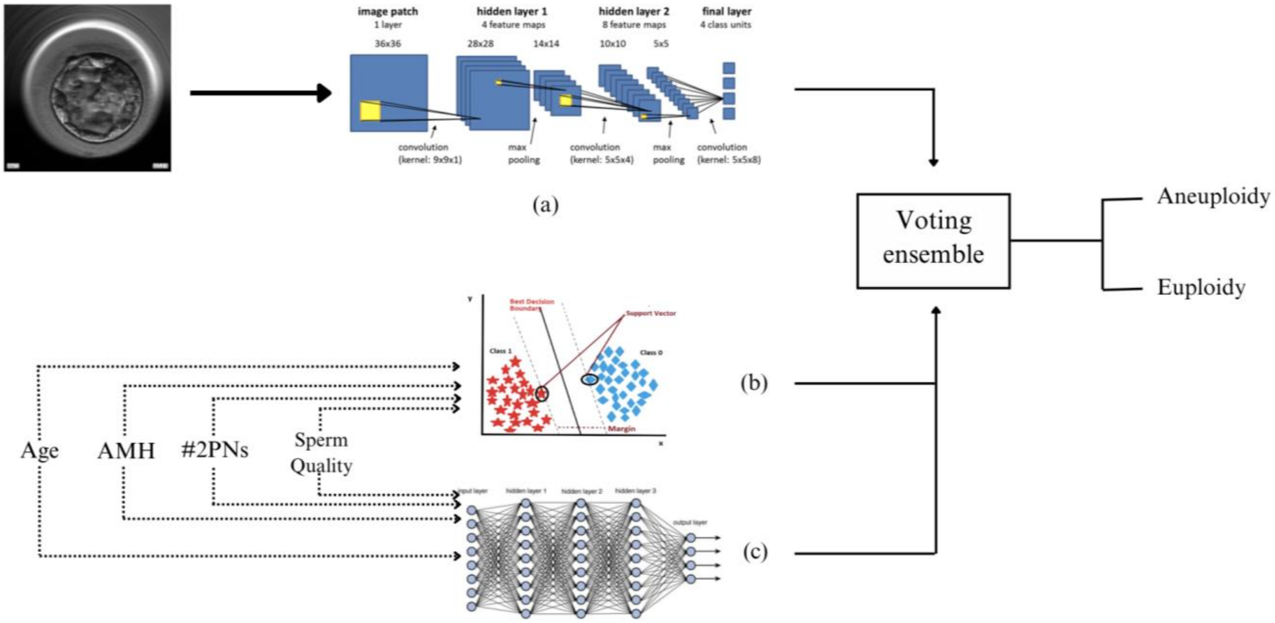


Fig.2.2 Multi-modal/Voting approach for embryo aneuploidy prediction: (a) CNN processing of embryo image, (b) SVM with clinical parameters, (c) Neural network with clinical data, combined through a voting ensemble.

2.2.1 Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) forms the backbone of our embryo classification system, leveraging deep learning techniques to process and analyse blastocyst stage images. For the CNN, we used our previously developed models for blastocyst classification. Embryos were evaluated on day 5 blastocyst stage, prior to trophectoderm biopsy and vitrification. The CNN was used alone and in conjunction with successive add-in of patient parameters analysed in SVM and multilayer NN models to generate a voting ensemble to assess embryo ploidy status.

In our implementation, we utilized the Xception architecture, pretrained on the ImageNet dataset, for its superior feature extraction capabilities. The model was fine-tuned with our embryo images, cropped and resized to 210x210 pixels on a dataset consisting of 699 embryos, and was split into training and validation sets using an 80-20 split (559 embryos - training, 140 embryos – validation). The images were normalized by scaling pixel values to the range [0, 1]. The network's input layer was modified to fit our specific image dimensions, and the output layer was adjusted to a single sigmoid activation function for binary classification. The base model's layers were set to trainable, allowing the network to adapt learned weights to our specific dataset. The network was compiled using the Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 0.001, momentum of 0.9, and a learning rate decay calculated over the 25 epochs. Regularization was employed to prevent overfitting, and various data augmentation techniques, including random rotations and flips, were applied during training to enhance the model's generalizability.

2.2.2 Support Vector Machine (SVM)

Support Vector Machines (SVM) were employed to incorporate clinical patient parameters into the prediction model. SVMs construct a hyperplane or set of hyperplanes in high or infinite dimensional space, which can be used for classification, regression, or other tasks like outlier's detection. SVMs can also be adapted to efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces. A nonlinear SVM with the radial bias function (RBF) kernels and C for linear kernels, in respect of the area under the curve (AUC). In RBF kernel, gamma was varied between 0.001 and 30 and C between 0.01 and 100. The class weight was set to "balanced".

In our study, patient parameters such as maternal age, Anti-Müllerian Hormone (AMH) levels, paternal sperm quality, and the number of normally fertilized (2PN) embryos were utilized. The data was standardized using the StandardScaler from scikit-learn, and a grid search was conducted to determine the optimal parameters for the SVM model. The classifier was trained using a linear kernel, which provided a clear margin of separation between classes. Performance metrics such as accuracy, sensitivity, and specificity were computed to evaluate the model's effectiveness.

2.2.3 Neural Network (NN)

A Neural Network (NN) was designed to integrate patient parameters, complementing the CNN and SVM models. The NN consists of multiple layers of nodes that include an input layer, multiple intermediate hidden layers, and an output layer. Except for the input nodes, each node is a neuron that uses a sigmoid activation. This network was trained with a stochastic gradient descent optimizer for predicting embryo ploidy status. The NN model processes patient characteristics such as maternal age, AMH level, paternal sperm quality, and total number of normally fertilized (2PN) embryos.

The NN architecture comprised three dense layers. The first dense layer consisted of 20 neurons with ReLU activation, followed by a second dense layer with 10 neurons also with ReLU activation, and finally, a sigmoid activation function in the output layer for binary classification. The network was compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy as the evaluation metric. The training process included callbacks for model checkpointing and TensorBoard for visualizing training metrics. The model was trained and validated on the split dataset, and the best-performing weights were saved and evaluated using confusion matrices to derive accuracy, sensitivity, and specificity.

2.2.4 Voting ensemble

The final ploidy status prediction was achieved using a voting ensemble that combined the outputs of the CNN, SVM and NN models. The predictions from these models were aggregated, and a hard voting scheme was applied to determine the final classification. In hard voting, each model votes for a class, and the class that receives the majority of the votes is selected as the final prediction. This ensemble approach leveraged the strengths of each model, enhancing overall prediction accuracy and robustness. The ensemble's performance was validated using standard metrics, ensuring comprehensive evaluation and reliability of the predictive system.

CHAPTER-3

STUDY DESIGN AND STATISTICAL ANALYSIS

Trained embryologists manually evaluated and classified 6828 embryos using the Gardner grading method, as previously reported. "Euploid" status was classified as "euploid" on PGT-A tests. "Non-euploid" status encompassed PGT-A testing (Invitae) designations of "aneuploid" and "indeterminate." Each related embryo grade (embryo stage (3 – 6), inner-cell mass (ICM, A–C), and trophectoderm grade (A–C)) was further registered for the PGT-A findings (the "euploid" and "non-euploid" as previously described). Both before and after taking clinical parameters (maternal age, anti-Mullerian hormone (AMH), number of two pronuclei embryos (2PNs), and sperm quality) into account, the manual grading accuracy in correctly classifying embryos as "euploid" or "non-euploid" was calculated using neural network and logistic regression models.

Demographics of the embryo cohorts

Demographics	**All Embryos**	**Aneuploid**	**Euploid**	**p-value**
Age (SD)	37.32 (3.5)	38.18 (3.6)	37.26 (3.4)	3.03E-14
BMI(kg/m2) (SD)	24.43 (4.2)	25.55 (3.4)	25.52 (4.7)	0.95
AMH (SD)	3.40 (2.7)	2.77 (2.5)	3.34 (2.8)	0.14
Day 3 FSH (SD)	7.28 (2.5)	7.23 (2.1)	7.27 (2.5)	0.67
Total oocytes retrieved	12.77 (4.6)	12.43 (4.8)	13.21 (4.6)	0.04
# of 2PNs (SD)	8.32 (3.8)	7.76 (3.6)	8.99 (3.8)	7.06E - 05
# of HQB (SD)	5.08 (2.6)	4.69 (2.9)	5.54 (2.8)	3.26E - 05
Race/ ethnicity				
White, n (%)	528 (75.3)	281 (77.7)	242 (72.8)	0.17
Black, n (%)	14 (1.8)	6 (1.5)	9 (2.5)	0.69
Asian, n (%)	106 (15.0)	49 (13.4)	58 (16.9)	0.19
Hispanic/Latino, n (%)	6.2 (0.92)	4.3 (1.12)	2.1 (0.64)	0.75
Other, n (%)	18 (2.6)	8 (2.4)	11 (2.8)	0.08
Declined, n (%)	14 (1.6)	4 (0.6)	11 (2.8)	0.28
Unavaliable, n(%)	17 (2.4)	12 (3.2)	6 (1.6)	1.01

SART Diagnosis				
Male factor, n (%)	244 (35.9)	116 (32.6)	131 (38.5)	0.08
Endometriosis, n (%)	28 (3.8)	13(3.8)	14 (3.7)	1.02
DOR, n (%)	98 (13.8)	69 (18.7)	28 (8.7)	2.09E-4
Tubal factor, n (%)	86 (12.1)	38 (10.9)	45 (13.7)	0.33
Uterine factor, n (%)	78 (11.2)	41 (11.3)	38 (11.4)	0.95
Unexplained, n (%)	98 (13.8)	49 (13.4)	48 (14.6)	0.83
PCOS, n (%)	57 (8.1)	22 (5.9)	36 (10.4)	0.04
RPL, n (%)	45 (6.4)	22 (5.9)	24 (6.7)	0.61
Fertility preservation, n (%)	97 (13.6)	51 (13.8)	45 (13.7)	1.02
Ovulation disorder, n (%)	168 (23.6)	92 (25.4)	74 (21.9)	0.42
Other, n (%)	63 (8.91)	42 (11.3)	22 (6.3)	0.04
Sperm class				
1, Excellent, n (%)	325 (46.7)	173 (47.9)	153 (45.5)	0.66
2, Good, n (%)	272 (39.2)	146 (40.2)	127 (37.9)	0.48
3, Fair, n (%)	86 (12.3)	37 (10.2)	48 (14.6)	0.11
4, Poor, n (%)	17 (2.4)	7 (1.9)	8 (2.4)	0.87

The statistical analysis of the embryo cohorts revealed significant differences in several demographics and clinical parameters between the aneuploid and euploid groups. Age and the number of 2PN embryos were significantly different, with p-values of 3.01E-14 and 7.16E-05, respectively, indicating that euploid embryos were associated with younger maternal age and a higher number of 2PN embryos. Additionally, the number of high-quality blastocysts (HQB) and the total oocytes retrieved were significantly higher in the euploid group (p-values of 3.27E-05 and 0.02, respectively). However, no significant differences were observed in BMI, AMH levels, or Day 3 FSH levels. The analysis of race/ethnicity did not show significant differences across groups. In terms of SART diagnoses, significant differences were found for diminished ovarian reserve (DOR) (p=2.08E-4), polycystic ovary syndrome (PCOS) (p=0.03), and the "Other" category (p=0.03). No significant differences were detected in the sperm class category. These findings suggest that maternal age, the number of 2PN embryos, HQB, total oocytes retrieved, DOR, and PCOS are important factors associated with embryo ploidy status.

CHAPTER 4

RESULTS AND SNAPSHOTS

4.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

Accuracy: 77.86 %

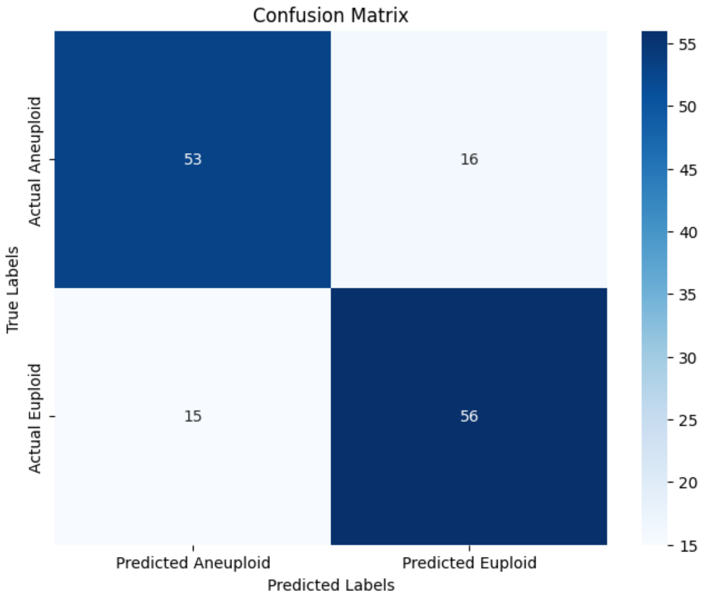


Fig 4.1.1 Confusion matrix of CNN-FE

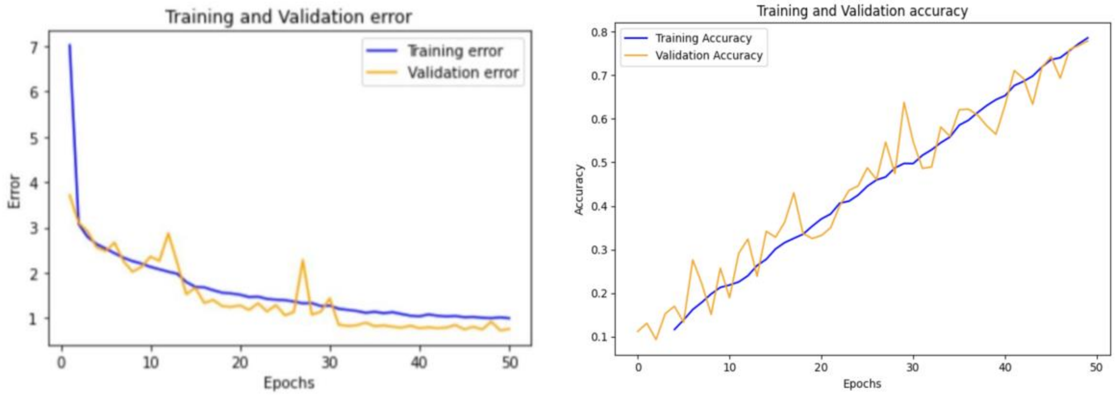


Fig 4.1.2 Training and validation loss and accuracy curves

4.2 VOTING ENSEMBLE

AI Model	Accuracy(%)	Δ Accuracy (%)	95% CI	P-value
CNN only	77.86	-	-	-
CNN, AMH	82.50	4.16	2.12, 7.35	0.1028
CNN, Maternal age	83.60	5.60	3.20, 8.00	0.0585
CNN, AMH, Maternal age	82.86	4.53	3.80, 8.60	0.0762
CNN, AMH, Maternal age, #2PNs	83.60	6.60	3.20, 8.00	0.0585
CNN, AMH, Maternal age, #2PNs, sperm quality	89.42	7.43	6.62, 14.95	0.0165

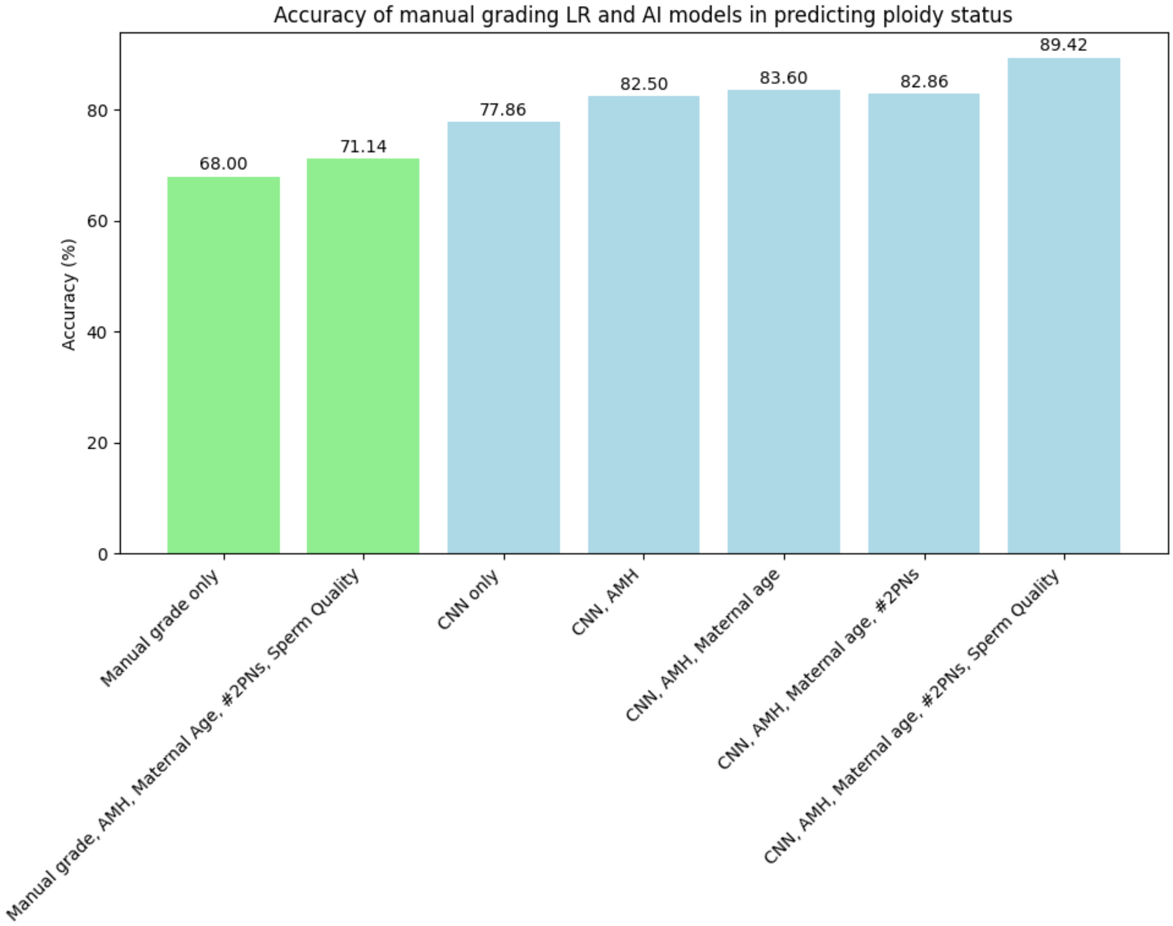


Fig 4.2.1 Comparison of Manual Grading vs. AI Models in Predicting Ploidy Status of Day 5 Blastocysts

4.3 DISCUSSION

In this study, we demonstrate the innovative application of voting ensembles to enhance the accuracy of image-based CNN processing. By integrating patient characteristics with the image-based algorithm, we achieved a significant improvement of over 10% in system accuracy, reinforcing the robustness of our findings beyond mere chance. This advancement is crucial in developing non-invasive genetic analytic methods for embryo karyotype screening, which can significantly enhance patient outcomes, particularly in the context of prioritizing single embryo transfer.

Our approach highlights the potential of voting ensembles to incorporate additional testing variables into CNN-only AI,improving predictive accuracy.

Manual morphology alone matched the accuracy of our CNN-only AI model, underscoring the limited utility of morphology images alone in euploid classification. By integrating additional variables, our combined voting ensemble system achieved a predictive accuracy of 89.42%, a notable improvement from the 77.86% accuracy of CNN-only AI and manual morphology combined with clinical parameters. This method represents a significant step forward in enhancing AI predictive power for embryo selection, especially in the absence of PGT-A testing results. Our AI system, designed for easy integration into clinics with advanced imaging capabilities, provides immediate, impactful results on the day of transfer, serving as a cost-effective adjunct to clinical care. However, the study's limitations include a uniform training set restricted to embryos imaged with the embryoscope, potential biases due to a limited and non-representative sample size, and the need for broader validation across different imaging systems and diverse populations. Future studies should aim to incorporate a more representative embryo cohort and additional patient characteristics to further refine and generalize the model.

CHAPTER 5
CONCLUSION

In conclusion, the innovative application of voting ensembles in this study has demonstrated a significant enhancement in the accuracy of embryo ploidy prediction. By combining convolutional neural networks with support vector machines, multi-layer neural networks, and incorporating critical clinical parameters such as maternal age, AMH levels, and sperm quality, we achieved a notable improvement in prediction accuracy. The ensemble approach, which reached an accuracy of up to 71.4%, underscores the effectiveness of integrating diverse data sources and machine learning techniques.

This methodological advancement holds considerable promise for non-invasive genetic analysis in assisted reproductive technology, particularly in improving embryo selection process. The improved accuracy of our voting ensembles over standalone CNN models indicated that a multifaced approach is crucial for reliable embryo ploidy prediction. This can lead to better clinical outcomes and higher success rates in procedures such as in vitro fertilization, by providing more informed and precise embryo selection.

Our findings suggest that the integration of AI with clinical parameters can be valuable tool in reproductive endocrinology, offering a cost-effective and efficient adjunct to existing clinical practices. However, future research should focus on expanding the diversity of the training datasets and validating the model across different imaging systems and populations to ensure broader applicability and effectiveness. This study paves the way for further advancements in AI- driven embryo selection, potentially transforming the landscape of reproductive medicine.

CHAPTER 6

REFERENCES

[1] Fragouli E, Alfarawati S, Spath K, Wells D. Morphological and cytogenetic assessment of cleavage and blastocyst stage embryos. *Mol Hum Reprod.* 2014;20:117–126. doi: 10.1093/molehr/gat073.

[2] Swain JE. Could time-lapse embryo imaging reduce the need for biopsy and PGS? *J Assist Reprod Genet.* 2013;30:1081–1090. doi: 10.1007/s10815-013-0048-4.

[3] Zhang J, Tao W, Liu H, Yu G, Li M, Ma S, et al. Morphokinetic parameters from a time-lapse monitoring system cannot accurately predict the ploidy of embryos. *J Assist Reprod Genet.* 2017;34:1173–1178. doi: 10.1007/s10815-017-0965-8.

[4] Munné S, Kaplan B, Frattarelli JL, Child T, Nakhuda G, Shamma FN, et al. Preimplantation genetic testing for aneuploidy versus morphology as selection criteria for single frozen-thawed embryo transfer in good-prognosis patients: a multicenter randomized clinical trial. *Fertil Steril.* 2019;112:1071–1079.e7. doi: 10.1016/j.fertnstert.2019.07.1346.

[5] Belandres D, Shamonki M, Arrach N. Current status of spent embryo media research for preimplantation genetic testing. *J Assist Reprod Genet.* 2019;36:819–826. doi: 10.1007/s10815-019-01437-6.

[6] Shi W, Zhao Z, Xue X, Li Q, Yao Y, Wang D, et al. Ploidy testing of blastocoel fluid for screening may be technically challenging and more invasive than that of spent cell culture media. *Front Physiol.* 2022;13:794210. doi: 10.3389/fphys.2022.794210.

[7] Leaver M, Wells D. Non-invasive preimplantation genetic testing (niPGT): the next revolution in reproductive genetics? *Hum Reprod Update.* 2020;26:16–42. doi: 10.1093/humupd/dmz033.

[8] Fitz VW, Kanakasabapathy MK, Thirumalaraju P, Kandula H, Ramirez LB, Boehnlein L, et al. Should there be an “AI” in TEAM? Embryologists selection of high implantation potential embryos improves with the aid of an artificial intelligence algorithm. *J Assist Reprod Genet.* 2021;38:2663–2670. doi: 10.1007/s10815-021-02318-7.

CHAPTER 7

APPENDIX

7.1 SOURCE CODE

```
import argparse
import csv
import math
import time
from keras import Sequential, Model, Input
from keras.applications import Xception, ResNet50
from keras.engine.saving import load_model
from keras.optimizers import SGD
from keras.regularizers import L1L2, l2
from keras.utils import to_categorical
from keras.utils.data_utils import Sequence
from keras import callbacks
import numpy as np
import cv2
import os
import pandas as pd
from keras.layers import Dense, Dropout, Flatten, Activation
from sklearn import preprocessing, tree
from sklearn.metrics import confusion_matrix, accuracy_score
import efficientnet.keras as efn
from sklearn.model_selection import train_test_split

os.environ["CUDA_VISIBLE_DEVICES"] = "1,2"
seed = 10
np.random.seed(seed)

im_ht = im_wid = 210
dim = (im_wid, im_ht)
batch_size = 32
num_classes = 2
n_epochs = 25
day = "day_5"
data_root = "../data/data_separated/" + day + "/"
root = "../"
output_root = root + "models/"
os.makedirs(output_root, exist_ok=True)
lr = 0.001

labels = {"Aneuploid": 0, "Euploid": 1}
```

```
def load_quality_labels(data_file_path):
    label_train = {}
    label_test = {}
    label_validation = {}

    labels = {}
    with open(data_file_path) as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        next(csv_reader)
        for row in csv_reader:
            try:
                labels[row[0]] = int(row[6])
            except ValueError:
                print(row)

    return labels


def one_hot(pos):
    a = np.zeros(num_classes)
    a[pos] = 1
    return a


def load_images(data_path):
    x = []
    y = []
    file_names = []
    for class_no in os.listdir(data_path):
        for image_name in os.listdir(data_path + class_no):
            im = cv2.imread(data_path + class_no + "/" + image_name)
            im = im[40:460, 40:460]

            im = cv2.resize(im, (210, 210))

            im = im / 255
            x.append(im)
            y.append(one_hot(int(labels[class_no])))
            file_names.append(image_name)

    print("Total images imp", len(x))
    return x, y, file_names
```

```
def load_data_cnn(X_all):
    x = []
    for line in X_all:
        im = cv2.imread('../data/Master ES Image Download 1-11-21/' + line[0])
        im = im[40:460, 40:460]

        im = cv2.resize(im, (210, 210))

        im = im / 255
        x.append(im)

    x = np.array(x)
    return x


def datagen(x_set, y_set, n_batches):
    index = 0
    while True:
        index = int(index % n_batches)
        print(index, len(x_set) // batch_size, index * batch_size, len(x_set))
        # randomize train data and make batches
        if index == 0:
            shuffle_index = np.arange(len(x_set))
            np.random.shuffle(shuffle_index)
            x_set = x_set[shuffle_index]
            y_set = y_set[shuffle_index]

        x = x_set[index * batch_size: (index + 1) * batch_size]
        y = y_set[index * batch_size: (index + 1) * batch_size]

        batch_x, batch_y = [], []
        for i in range(len(x)):
            im = x[i]
            rotate_m = cv2.getRotationMatrix2D((im.shape[1] / 2, im.shape[0] / 2),
np.random.randint(0, 360), 1)
            im = cv2.warpAffine(im, rotate_m, (im.shape[1], im.shape[0]))
            flip_choice = np.random.choice([0, 1, -1, 2])
            if flip_choice != 2:
                im = cv2.flip(im, flip_choice)
            batch_x.append(im)
            batch_y.append(y[i])
        index += 1

        yield np.array(batch_x), np.array(batch_y)
```

```
def lr_scheduler(epoch):
    drop = 0.5
    epochs_drop = 10
    new_lr = lr * math.pow(drop, math.floor((1 + epoch) / epochs_drop))
    print("lr = ", new_lr)
    return new_lr


class DataGenerator(Sequence):
    def __init__(self, data_set, batch_size):
        self.Data_Set = data_set
        print("Class Distribution:", [len(_[0]) for _ in self.Data_Set])
        print("Max data for a class:", max([len(_[0]) for _ in self.Data_Set]))
        self.batch_size = batch_size

    def __len__(self):
        return int(max([len(_[0]) for _ in self.Data_Set]) // self.batch_size)

    def __getitem__(self, index):
        X = []
        Y = []
        for x, y in self.Data_Set:
            index_ = int(index % (int(len(x) / self.batch_size)))

            batch_x = np.array(x[index_ * self.batch_size: (index_ + 1) * self.batch_size])
            batch_y = np.array(y[index_ * self.batch_size: (index_ + 1) * self.batch_size])

            for batch_x, batch_y in datagen.flow(batch_x, batch_y, batch_size=batch_size):
                for i in range(len(batch_x)):
                    im_good = batch_x[i]
                    X.append(im_good)
                    Y.append(batch_y[i])
                break

        X = np.array(X)
        Y = np.array(Y)

        return [X, Y]

def get_model(parameters):
    base_model = Xception(input_shape=(im_ht, im_wid, 3), weights='imagenet',
include_top=False, pooling="avg")

    model_output = base_model.output

    predictions = Dense(1, activation='sigmoid')(model_output)
    model = Model(inputs=base_model.input, outputs=predictions)

    for layer in base_model.layers:
        layer.trainable = True

    optim = SGD(parameters["learning_rate"], 0.9, nesterov=True,
decay=parameters["learning_rate"] / n_epochs)
    model.compile(optimizer=optim, loss='binary_crossentropy', metrics=['accuracy'])

    return model


def get_nn_model(input_shape):
    model = Sequential()
    model.add(Dense(20, input_shape=input_shape, activation='relu'))
    model.add(Dense(10, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

    return model


def svm_model(features):
    print(len(X_train))
    print(len(X_valid))
    a = np.array(X_train[:, features])
    from sklearn.preprocessing import StandardScaler
    sc_X = StandardScaler()
    X_train_sc = sc_X.fit_transform(np.array(X_train[:, features]))
    X_valid_sc = sc_X.transform(np.array(X_valid[:, features]))

    from sklearn.svm import SVC
    classifier = SVC(kernel='linear', random_state=0)
    classifier.fit(X_train_sc, Y_train)

    print("Train Acc:", accuracy_score(Y_train, classifier.predict(X_train_sc)))
    print("Valid Acc:", accuracy_score(Y_valid, classifier.predict(X_valid_sc)))
    conf_mat = confusion_matrix(Y_valid, classifier.predict(X_valid_sc) )
    print(conf_mat)
    total = sum(sum(conf_mat))
    sensitivity = conf_mat[0, 0] / (conf_mat[0, 0] + conf_mat[1, 0])
    specificity = conf_mat[1, 1] / (conf_mat[1, 1] + conf_mat[0, 1])
    final_test_acc = (conf_mat[0, 0] + conf_mat[1, 1]) / total

    print("SVM sensitivity", sensitivity)
    print("SVM specificity", specificity)
    print("SVM ACC", final_test_acc)

    return classifier.predict(X_train_sc), classifier.predict(X_valid_sc), sensitivity, specificity,
final_test_acc


def train_nn(features):

    from sklearn.preprocessing import StandardScaler
    sc_X = StandardScaler()
    X_train_sc = sc_X.fit_transform(np.array(X_train[:, features]))
    X_valid_sc = sc_X.transform(np.array(X_valid[:, features]))

    valid_data = [X_valid_sc, Y_valid]
    input_shape = (len(X_train_sc[0]),)
    model_2 = get_nn_model(input_shape)

    checkpoint = callbacks.ModelCheckpoint("../models/seed_"+str(seed)+"_" + day +
"_pliody_nn.hdf5",
                                           monitor='val_loss', verbose=0,
                                           save_best_only=True,
                                           save_weights_only=False, mode='auto', period=1)
    tfboard = callbacks.TensorBoard(log_dir='../logs/' + day +
'_pliody/seed_'+str(seed)+'ResNet_nn_' + str(time.time()),
                                    histogram_freq=0, batch_size=30,
                                    write_graph=True, write_grads=True, write_images=True,
embeddings_freq=0,
                                    embeddings_layer_names=None, embeddings_metadata=None)

    callbacks_ = [checkpoint, tfboard]

    print("Training NN Start")
    model_2.fit(X_train_sc, Y_train, batch_size=64, epochs=n_epochs,
validation_data=valid_data,
                shuffle=True, callbacks=callbacks_)
    print("Training End")

    model_2.load_weights("../models/seed_"+str(seed)+"_" + day + "_pliody_nn.hdf5")

    print("Validating NN  Best Model")
    print("Validation NN Evaluation: ")
    print(model_2.evaluate(valid_data[0], valid_data[1], verbose=0))
    predictions_val = model_2.predict(valid_data[0])

    conf_mat = confusion_matrix(y_true=valid_data[1], y_pred=np.round(predictions_val))
    print(conf_mat)

    total = sum(sum(conf_mat))

    sensitivity = conf_mat[0, 0] / (conf_mat[0, 0] + conf_mat[1, 0])
    specificity = conf_mat[1, 1] / (conf_mat[1, 1] + conf_mat[0, 1])
    final_test_acc = (conf_mat[0, 0] + conf_mat[1, 1]) / total

    print("NN sensitivity", sensitivity)
    print("NN specificity", specificity)
    print("NN ACC", final_test_acc)
    return model_2.predict(X_train_sc), predictions_val, sensitivity, specificity, final_test_acc


def train_cnn(X_train, Y_train, X_valid, Y_valid):
    X_train, Y_train, X_valid, Y_valid = tuple(
        map(np.array, [X_train, Y_train, X_valid, Y_valid]))

    class Confusion_Mat_Callback(callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            if (epoch + 1) % 5 == 0:
                preds = model_1.predict(valid_data[0])
                preds = np.argmax(preds, axis=-1)
                temp = np.argmax(valid_data[1], axis=-1)
                print("=" * 50, "\n", confusion_matrix(temp, preds))
            return

    valid_data = [X_valid, Y_valid]
    n_train_batches = (len(X_train) // batch_size) + 1
    train_gen = datagen(X_train, Y_train, n_train_batches)

    # Build Model
    model_param = {'learning_rate': lr, 'untrainable_layers': 0}

    model_1 = get_model(model_param)

    checkpoint = callbacks.ModelCheckpoint("../models/seed_"+str(seed)+"_" + day +
"_pliody_cnn.hdf5",
                                           monitor='val_loss', verbose=0,
                                           save_best_only=True,
                                           save_weights_only=False, mode='auto', period=1)
    tfboard = callbacks.TensorBoard(log_dir='../logs/seed_'+str(seed)+'_' + day +
'_pliody3/Xception_' + str(time.time()),
                                    histogram_freq=0, batch_size=30,
                                    write_graph=True, write_grads=True, write_images=True,
embeddings_freq=0,
                                embeddings_layer_names=None, embeddings_metadata=None)
    lr_drop = callbacks.LearningRateScheduler(lr_scheduler)
    conf_call = Confusion_Mat_Callback()
    callbacks_ = [checkpoint, tfboard]

    print("Training Start")
    print("Training End")

    model_1.load_weights("../models/seed_"+str(seed)+"_" + day + "_pliody_cnn.hdf5")

    print("Validating Best Model")
    print("Validation Evaluation: ")
    print(model_1.evaluate(valid_data[0], valid_data[1], verbose=0))
    predictions_val = model_1.predict(valid_data[0])

    conf_mat = confusion_matrix(y_true=valid_data[1], y_pred=np.round(predictions_val))
    print(conf_mat)

    total = sum(sum(conf_mat))
    sensitivity = conf_mat[0, 0] / (conf_mat[0, 0] + conf_mat[1, 0])
    specificity = conf_mat[1, 1] / (conf_mat[1, 1] + conf_mat[0, 1])
    final_test_acc = (conf_mat[0, 0] + conf_mat[1, 1]) / total

    print("CNN sensitivity", sensitivity)
    print("CNN specificity", specificity)
    print("CNN ACC", final_test_acc)
    return model_1.predict(X_train), predictions_val, sensitivity, specificity, final_test_acc


if __name__ == '__main__':
    datasets = pd.read_csv('../Manoj Ploidy Variables ASRM D0-5 4-15-21.csv')

    datasets_main = datasets.loc[datasets["Day of Development"] == int(day.replace("day_", ""))]
    X = datasets_main.iloc[:, list(range(0, 13))].values
    Y = list(map(lambda x: labels[x], datasets_main.iloc[:, 4].values))
    Filenames = datasets_main.iloc[:, 0].values

    X_train, X_valid, Y_train, Y_valid, filename_train, filenames_valid = train_test_split(X, Y,
Filenames,
                                                                      test_size=0.2,
                                                                      random_state=10)


    X_train_imgs = load_data_cnn(X_train)
    X_valid_imgs = load_data_cnn(X_valid)
    Y_train = np.array(Y_train)
    Y_valid = np.array(Y_valid)

    print(X_train_imgs.shape, np.array(Y_train).shape)
    print(X_valid_imgs.shape, np.array(Y_valid).shape)

    feat = [5,6]  # da5t

    svm_predictions_train, svm_predictions_val, svm_sensitivity, svm_specificity,
svm_final_test_acc = svm_model(
        features=feat)
    predictions_train_nn, predictions_val_nn, nn_sensitivity, nn_specificity, nn_final_test_acc =
train_nn(
        features=feat)

    predictions_train, predictions_val, cnn_sensitivity, cnn_specificity, cnn_final_test_acc =
train_cnn(X_train_imgs,np.array(Y_train),X_valid_imgs,np.array(Y_valid))


    votes_val = np.where(
        np.sum([predictions_val.reshape((-1,)) + predictions_val_nn.reshape((-1,)) +
svm_predictions_val], axis=0) > 1,
        1, 0)

    g = np.array(Y_valid)
    conf_mat = confusion_matrix(Y_valid, np.round(votes_val))
    total = sum(sum(conf_mat))
    en_sensitivity = conf_mat[0, 0] / (conf_mat[0, 0] + conf_mat[1, 0])
    en_specificity = conf_mat[1, 1] / (conf_mat[1, 1] + conf_mat[0, 1])
    en_final_val_accuracy = (conf_mat[0, 0] + conf_mat[1, 1]) / total

    print("Statistics for voting classifier, where simple majority rules:\n")
    print(conf_mat)
    print('specificity : ', en_specificity)
    print('sensitivity : ', en_sensitivity)
    print('accuracy : ', en_final_val_accuracy)


    print(cnn_final_test_acc, svm_final_test_acc, nn_final_test_acc, en_final_val_accuracy, "",
          cnn_specificity, svm_specificity, nn_specificity, en_specificity, "",
          cnn_sensitivity, svm_sensitivity, nn_sensitivity, en_sensitivity)
```