Ovarian Cancer detection using Machine Learning

Nahid Hasan, Shahriar Parvez Shamim, Ridwan Ahmed Arman, Nissan Bin Sharif

1. Introduction:

Ovarian cancer stands as a significant global health challenge, characterized by its high mortality rates and often delayed diagnosis. Despite advancements in medical technology and treatment strategies, effectively addressing ovarian cancer remains a daunting task in the field of oncology. Therefore, there is a critical need for precise and timely detection methods to enhance patient outcomes and increase survival rates.

Machine learning, a branch of artificial intelligence, has emerged as a promising solution in healthcare, offering the potential to revolutionize disease diagnosis, prognosis, and therapeutic approaches. By leveraging extensive datasets and sophisticated algorithms, machine learning techniques can uncover subtle patterns and relationships within complex biological data, paving the way for the development of predictive models for disease detection.

Global health statistics underscore the severity of ovarian cancer, ranking it among the leading causes of cancer-related deaths in women. This highlights the urgency of devising effective screening and diagnostic tools. Current diagnostic methods, such as imaging studies and biomarker assays, often exhibit limitations in sensitivity and specificity, resulting in delayed or missed diagnoses.

The primary aim of this study is to harness the capabilities of machine learning to enhance the early detection of ovarian cancer. By analyzing diverse datasets encompassing clinical, imaging, and molecular data, our objective is to create robust predictive models capable of identifying individuals at elevated risk of ovarian cancer. These models will integrate various features, including patient demographics, genetic markers, imaging findings, and biomarker profiles, to maximize accuracy and dependability.

Through the utilization of machine learning algorithms such as logistic regression, support vector machines, random forests, and deep learning, our endeavor is to develop predictive models that can accurately stratify patients based on their likelihood of developing ovarian cancer. By leveraging advanced computational techniques and comprehensive datasets, our goal is to equip healthcare providers with valuable tools for early detection and targeted interventions.

2. Project Methodology:

The project methodology involves collecting and preprocessing ovarian cancer image datasets, followed by constructing a Convolutional Neural Network (CNN) model using the Keras Sequential API. This model consists of convolutional layers with ReLU activation, max-pooling layers, and fully connected layers with a sigmoid activation function for binary classification. Subsequently, the model is trained on the preprocessed training data and validated on a separate testing dataset to ensure generalizability. Evaluation metrics such as accuracy, loss, precision, recall, and F1-score are used to assess the model's performance, with visualization techniques aiding in the analysis. The model with the highest accuracy and optimal performance is then selected as the best fit for ovarian cancer detection. Overall, this methodology aims to utilize machine learning techniques to develop a robust model for the early detection of ovarian cancer, with the potential to improve patient outcomes and healthcare management.

2.1 Data Collection Procedure:

For the data collection procedure, the IEEE Dataport's STRAMPN-histopathological dataset for ovarian cancer was utilized [1]. The dataset was accessed and retrieved from the IEEE Dataport repository, which provides a comprehensive collection of datasets for various research purposes. Specifically curated for histopathological analysis of ovarian cancer, the STRAMPN dataset contains a diverse range of high-resolution histopathological images representative of different ovarian cancer subtypes and stages. Upon accessing the dataset, the images were systematically extracted, ensuring the inclusion of relevant metadata and annotations for accurate labeling

and analysis. Overall, the data collection procedure adhered to rigorous standards to procure a reliable and representative dataset for subsequent analysis and model development in the study of ovarian cancer detection.

2.2.Data Validation Procedure:

The data validation procedure involved thorough checks to ensure the reliability of the ovarian cancer dataset. This included verifying data completeness, consistency, and accuracy, as well as validating against established standards. Any identified discrepancies were addressed through data cleaning and correction. Additionally, the dataset's authenticity and compliance with ethical guidelines were confirmed, ensuring its suitability for analysis. Overall, these steps ensured the dataset's reliability for accurate ovarian cancer detection analysis.

2.3.Data Preprocessing and Normalization:

In the data preprocessing and normalization stage, the images from the ovarian cancer dataset were loaded and resized to a uniform size of 128x128 pixels. The images were converted to grayscale to reduce computational complexity and standardize the input format. Each image was then normalized by dividing the pixel values by 255, ensuring that they ranged between 0 and 1. This normalization step is crucial for stabilizing the training process and improving the convergence of the neural network model. Additionally, the labels associated with each image were encoded using Label Encoder to convert categorical labels into numerical values, facilitating model training. Finally, the images and labels were converted into numpy arrays and further processed to prepare them for training the convolutional neural network (CNN) model. This preprocessing and normalization pipeline ensures that the input data is properly formatted and scaled, optimizing the performance of the machine learning model for ovarian cancer detection.

2.4.Feature Extraction:

We implemented some algorithms and techniques which helped our model learn and also helped extracted relevant features from the input images during the training process. As the model learned from the dataset, the convolutional layers detected various patterns and features at different levels of abstraction, such as edges, textures, and shapes, while the max-pooling layers down-sampled the feature maps, retaining the most important information. The subsequent dense layers then flattened the extracted features into a vector format suitable for classification. By leveraging the hierarchical feature extraction capabilities of CNNs, the model could effectively discern distinguishing characteristics indicative of ovarian cancer within the input images, ultimately aiding in accurate disease classification.

2.5.Classification Algorithms:

In this project, a Convolutional Neural Network (CNN) model was employed for the classification task of ovarian cancer detection. CNNs are a class of deep neural networks specifically designed for processing structured grid-like data, such as images. The architecture of the CNN consisted of multiple convolutional layers, each followed by a max-pooling layer for down-sampling. This stack of convolutional and pooling layers allowed the model to automatically learn relevant features from the input images, capturing patterns at different levels of abstraction. Subsequently, the extracted features were flattened and passed through fully connected dense layers, enabling the model to perform classification based on the learned representations. The final layer of the CNN utilized a sigmoid activation function to output the probability of ovarian cancer presence. The model was trained using the Adam optimizer and binary cross-entropy loss function.

2.6.Data Analysis Techniques:

Data analysis techniques involved in this project include evaluating the performance of the trained Convolutional Neural Network (CNN) model for ovarian cancer detection. This assessment encompasses analyzing various metrics such as accuracy, loss, precision, recall, and F1-score, which provide insights into the model's predictive capability and generalization ability. Additionally, a confusion matrix is generated to visualize the classification results, depicting the true positive, true negative, false positive, and false negative predictions. By employing these data analysis techniques, researchers can effectively gauge the CNN model's performance in distinguishing between ovarian cancer and non-cancer cases based on histopathological images.

$$\text{accuracy} = \frac{\text{number of currect prediction}}{\text{number of total prediction}}$$

$$\text{F1 score} = 2 \times \frac{precision \times recall}{precision + recall}$$

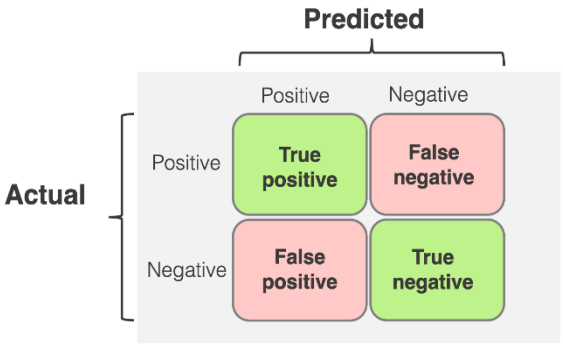

Figure 1 : confusion matrix

2.7. Block Diagram and Workflow Diagram of Proposed Model:

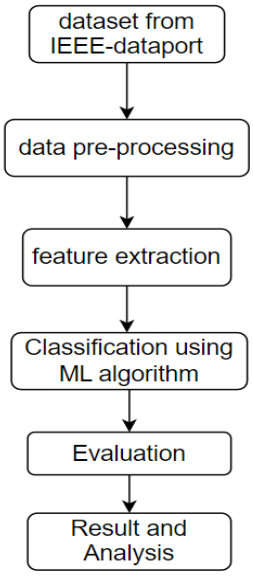

Figure 2 : Workflow of Proposed Model

2.7. Experimental setup and Implementation:

The model is implemented on Google Colab notebooks, enabling users to write and execute Python code directly in a web browser.

3. Results and Discussion

The convolutional neural network (CNN) model and Support vector machine were used in the classification of ovarian cancer cells. The model was trained on 90% of the available data, while the remaining 10% was reserved for testing purposes.

Convolutional neural network and support vector machines (SVM) are two different algorithms that were trained and evaluated. For the models' evaluation, the accuracy and F1- score were employed. Following are each algorithm's accuracy and F1-score:

Model	Accuracy	F1 Score
Convolutional Neural Network (CNN)	0.89	0.89
Support Vector Machine (SVM)	0.87	0.87

Convolutional Neural Network, the best-fit model for predicting ovarian cancer in this dataset among the two algorithms, had the highest accuracy and F1-score.

3.1 Confusion matrix analysis

Convolutional Neural Network (CNN)

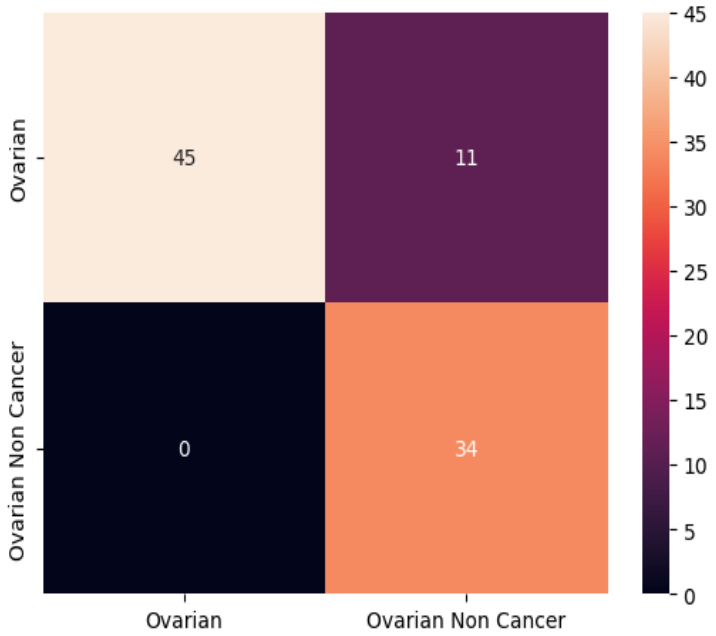

Figure 3 : Confusion matrix analysis of CNN model

Support Vector Machine (SVM)

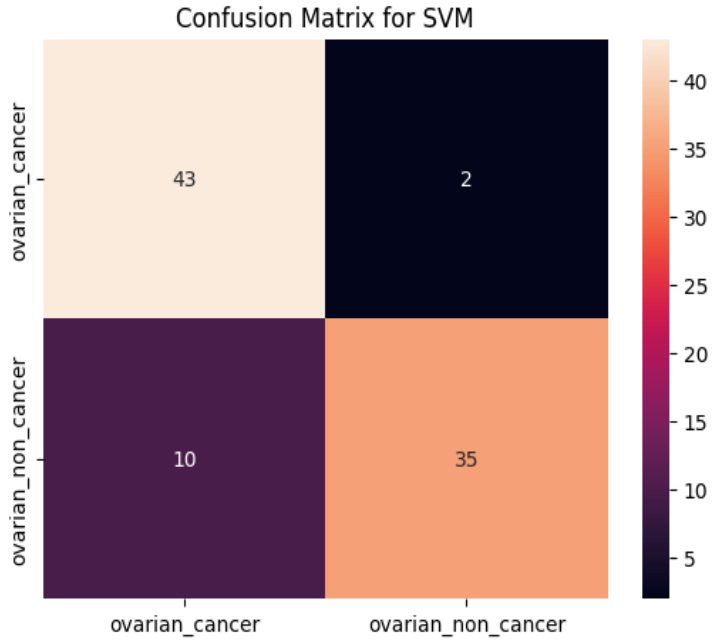

Figure 4 : Confusion matrix analysis of SVM model

3.2 Results validation by Graphical Representation

Accuracy graph:

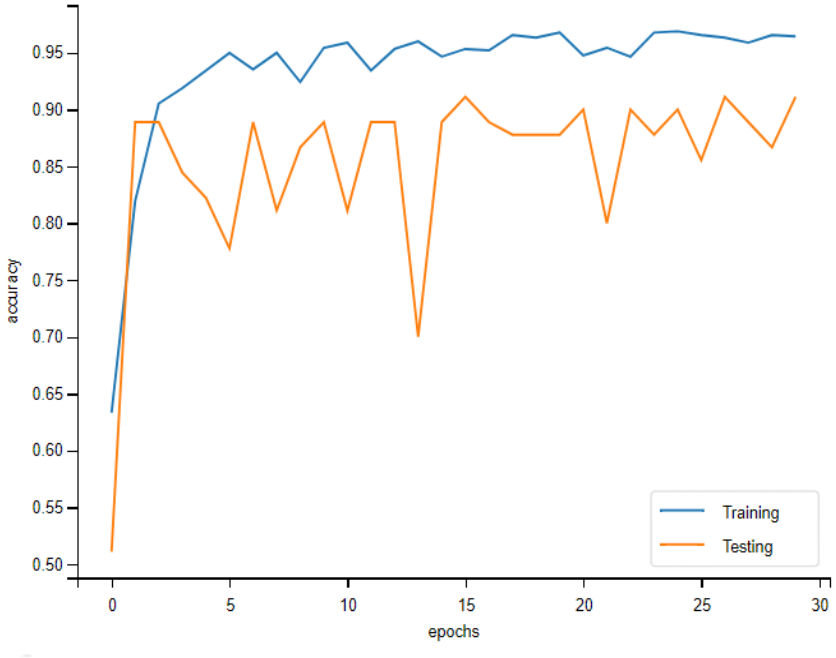

Figure 5 : Accuracy graph of training and testing

Loss graph:

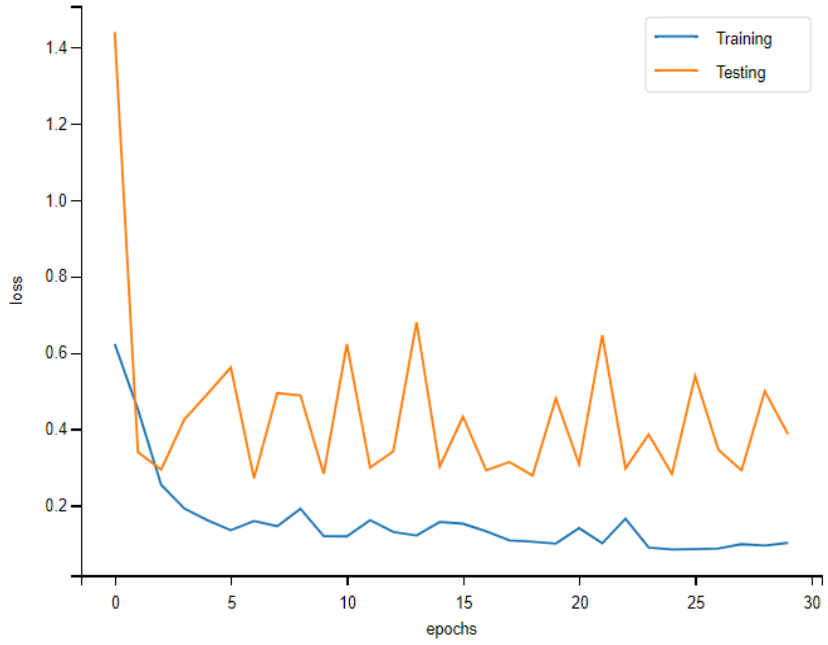

Figure 6 : Loss graph of training and testing

4. Conclusion and Future Recommendations

In conclusion, the CNN and SVM models demonstrated strong performance in classifying ovarian cancer cells, with the CNN achieving an 84% accuracy and the SVM reaching 87% accuracy and F1 score. To further improve their effectiveness, future work could concentrate on diversifying the dataset, employing advanced techniques like transfer learning, and incorporating domain knowledge. These enhancements could enhance the models' robustness and pave the way for their practical use in real-world healthcare scenarios, facilitating early detection and diagnosis of ovarian cancer.

Appendixes

```
import os #file and directory operations.
import numpy as np #handling arrays and mathematical operations.
import cv2 as cv #image loading, preprocessing, and feature extraction.
from sklearn.preprocessing import LabelEncoder #encoding categorical labels as numerical values
from keras.layers import Conv2D, MaxPool2D, Flatten, Dense #image classification
from keras.models import Sequential #build a linear stack of layers
import matplotlib.pyplot as plt #visualizing training/validation metrics and results.
from keras.applications.vgg16 import VGG16 #image classification tasks
import seaborn as sns #creating more attractive and informative statistical graphics.
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score #confusion_matrix,
classification_report, and accuracy_score

#Data preprocessing
def input_data(folder_path, output_data):    #importing image data into the output_data list
  for dirs in os.listdir(folder_path):  #starting loop for path
    class_name = dirs #subdirectory names represent class labels for the images.
    new_path = os.path.join(folder_path, class_name) #constructs a new path by joining folder_path and
class_name.
    for img in os.listdir(new_path):  #loop that iterates through the contents of the subdirectory (class-specific
directory) represented by new_path
      img_arr = cv.imread(os.path.join(new_path, img), cv.IMREAD_GRAYSCALE) #using OpenCV (cv) and
converts it to grayscale using the cv.IMREAD_GRAYSCALE flag
      resize = cv.resize(img_arr, (128,128)) #converting size of 128x128 pixels
      output_data.append([resize, class_name]) #containing the resized image (resize) and the corresponding
class name (class_name).
  return output_data

train_data = "dataset/train"
test_data = "dataset/test"

# Load the data into numpy arrays
train_data = input_data("dataset/train", [])
test_data = input_data("dataset/test", [])

train_images = []              #separating the image and labels from the train_data list
train_labels = []
for features, labels in train_data:
  train_images.append(features)
  train_labels.append(labels)

label_enc = LabelEncoder()          # encoding the labels
train_labels = label_enc.fit_transform(train_labels)
```

```
test_labels = label_enc.transform(test_labels)

train_images = np.array(train_images)    #converting the images and labels into numpy array
train_labels = np.array(train_labels)
test_images = np.array(test_images)
test_labels = np.array(test_labels)

train_images = train_images/255    # normalizing the image pixels
test_images = test_images/255

train_images = np.expand_dims(train_images, axis=3)      # adding a dimension on the images
test_images = np.expand_dims(test_images, axis=3)

# Print the shape and contents of the numpy arrays
print("Shape of train_images:", train_images.shape)
print("Contents of train_images:", train_images)
print("Shape of test_images:", test_images.shape)
print("Contents of test_images:", test_images)

import matplotlib.pyplot as plt

plt.figure(figsize=(15,10))
for i in range(25):
  plt.subplot(5, 5, i+1)
  plt.imshow(test_images[i], cmap='gray')
  plt.title(f"{label_enc.inverse_transform([test_labels[0]])}")
  plt.axis("off")

model1 = Sequential()

model1.add(Conv2D(32, (3, 3), input_shape=(128,128,1), activation="leaky_relu"))
model1.add(MaxPool2D(2,2))
model1.add(Conv2D(64, (3, 3), activation="leaky_relu"))
model1.add(MaxPool2D(2,2))
model1.add(Conv2D(128, (3, 3), activation="leaky_relu"))
model1.add(MaxPool2D(2,2))
model1.add(Conv2D(256, (3, 3), activation="leaky_relu"))
model1.add(MaxPool2D(2,2))
model1.add(Flatten())
model1.add(Dense(256, activation="relu"))
model1.add(Dense(1, activation="sigmoid"))
model1.summary()

model1.compile(optimizer="adam", loss="binary_crossentropy", metrics = ['accuracy'])

history1 = model1.fit(train_images, train_labels, validation_data=(test_images, test_labels), epochs=30)

plt.plot(history1.history["accuracy"])
plt.plot(history1.history["val_accuracy"])
plt.xlabel("epochs")
plt.ylabel("accuracy")
plt.legend(["Training","Testing"])
plt.show()

plt.plot(history1.history["loss"])
plt.plot(history1.history["val_loss"])
```

```
plt.xlabel("epochs")
plt.ylabel("loss")
plt.legend(["Training","Testing"])
plt.show()

y_pred1 = model1.predict(test_images)

y_pred1 = np.where(y_pred1>0.6,1,0)

y_pred1 = y_pred1.reshape(1,-1)[0]

print(classification_report(y_pred1, test_labels))

sns.heatmap(confusion_matrix(y_pred1, test_labels), fmt='g', annot=True , xticklabels=["Ovarian", "Ovarian Non Cancer"], yticklabels=["Ovarian", "Ovarian Non Cancer"])

from sklearn.svm import SVC

# SVM model
svm_model = SVC(kernel='linear')
svm_model.fit(train_images.reshape(train_images.shape[0], -1), train_labels)

# Predictions
svm_pred = svm_model.predict(test_images.reshape(test_images.shape[0], -1))

print(classification_report(test_labels, svm_pred))
sns.heatmap(confusion_matrix(test_labels, svm_pred), fmt='g', annot=True, xticklabels=label_enc.classes_, yticklabels=label_enc.classes_)
plt.title("Confusion Matrix for SVM")
plt.show()
```

Reference:

[1]. Saravanan R., Sivakumari S., Manimegalai D. "STRAMPN: Histopathological Images for Ovarian Cancer Prediction," IEEE DataPort, 2021. [Online]. Available: https://ieee-dataport.org/documents/strampn-histopathological-images-ovarian-cancer-prediction