GomuNet: The XML-Powered Mind-Reader for Gomoku Moves using Deep Learning

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Sripathi Praneeth (112215176):

External Supervisor : Dr. Sumit Kumar Gupta
Semester: VII

Department of Computer Science and Engineering

Indian Institute of Information Technology, Pune

(An Institute of National Importance by an Act of Parliament)

3 December 2025

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **GomuNet: The XML-Powered Mind-Reader for Gomoku Moves using deep learning** submitted by **Sripathi Praneeth** bearing the **MIS No: 112215176**, in completion of this project work under the guidance of **Dr.Sumit Kumar Gupta** is accepted for the project report submission in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in the **Department of Computer Science and Engineering** Indian Institute of Information Technology, Pune (IIIT Pune), during the academic year **2025-26**.

Dr. Sumit Kumar Gupta	**Dr.Bhupendra Singh**
Assistant Professor	Assistant Professor
Department of CSE	Head of CSE Department
IIITPune	IIITPune

Project Viva-voce held on 03 /12/2025

Undertaking for Plagiarism

I **Sripathi Praneeth** solemnly declare that research work presented in this **report** titled**"GomuNet: The XML-Powered Mind-Reader for Gomoku Moves."** is solely **my** project work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete report has been written by **me**. I understand the zero tolerance policy of **Indian Institute of Information Technology, Pune** towards plagiarism. Therefore **I** declare that no portion of my **report** has been plagiarized and any material used as reference is properly referred/cited. I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of the degree, the Institute reserves the rights to withdraw/revoke my **B.Tech** degree.

Student's Name and Signature with Date

Sripathi Praneeth

Conflict of Interest

GomuNet: The XML-Powered Mind-Reader for Gomoku Moves

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

Student's Name and Signature with Date

Sripathi Praneeth

ACKNOWLEDGEMENT

This project would not have been possible without the help and cooperation of many. I would like to thank the people who helped me directly and indirectly in the completion of this project work.

First and foremost, I would like to express my gratitude to our honorable Director, **Prof. Shireesh B. Kedare**, for providing his kind support in various aspects. I would like to express my gratitude to my project guide **Dr . Sumit Kumar Gupta**, **Assistant Professor, IIIT Pune**, for providing excellent guidance, encouragement, inspiration, constant and timely support throughout this **B.Tech Project**. I would like to express my gratitude to the **Dr.Bhupendra Singh**, **Department of CSE**, for providing his kind support in various aspects. I would also like to thank all the faculty members in the **Department of CSE/ECE** and my classmates for their steadfast and strong support and engagement with this project.

Abstract

The Gomoku game presents a challenging spatial decision-making environment where optimal move prediction requires understanding complex board patterns and long-term strategies. This project focuses on transforming raw, unstructured gameplay records into a machine-interpretable format that enables automated learning. Game data collected from the Gomocup repository is converted into XML, allowing each board state to be consistently parsed, structured, and represented as a numerical grid suitable for neural network training.

Using this XML-powered dataset, a Convolutional Neural Network (CNN) is developed to learn spatial dependencies between black and white stones. The model identifies frequently occurring tactical formations, strategic threats, and winning alignments by analyzing thousands of encoded board states. This deep learning architecture effectively captures both local and global board patterns, enabling it to predict strong candidate moves without hand-crafted rules or heuristics.

The trained model is evaluated through accuracy, confusion matrices, and performance metrics to validate its predictive capability. Results show that the CNN exhibits consistent generalization and reliability across test samples, demonstrating its ability to make intelligent move predictions in real-time scenarios. Overall, this project highlights the potential of combining XML-driven data preprocessing with deep learning techniques to build robust, strategy-aware AI systems for board games like Gomoku.

Keywords: Gomoku; Deep Learning; Convolutional Neural Networks (CNN); XML Data Parsing; Game State Representation; Move Prediction; Spatial Pattern Learning; Board-Game AI.

TABLE OF CONTENTS

List of Figures

Chapter 1
Introduction

1 Overview of Work

This project focuses on building an intelligent move-prediction system for the Gomoku board game using deep learning techniques. The work begins with the collection of raw game records from the official Gomocup repository, which are originally stored as unstructured PostgreSQL data. To enable efficient processing, these records are converted into a structured XML format that allows each game state to be systematically parsed. The XML representation captures the exact positions of stones on the board, making it possible to generate a large, well-organized dataset for machine learning.

After preparing the XML-based dataset, each board configuration is transformed into a numerical grid suitable for input into a Convolutional Neural Network (CNN). A dedicated preprocessing pipeline ensures that all samples maintain consistent dimensions and labeling, enabling the model to learn effectively. The CNN is then trained to detect spatial patterns—such as open fours, double threats, and winning alignments—allowing it to identify strategic move possibilities without predefined rules or heuristics.

The trained model is evaluated through performance metrics, including accuracy, F1-score, and confusion matrix analysis, to assess its predictive strength. Results show that the CNN generalizes well across test samples, effectively recognizing high-value board positions and suggesting optimal moves. Overall, the project demonstrates how XML-driven data structuring combined with deep learning can create a strong, automated decision-making system capable of understanding and predicting strategies in complex board games like Gomoku.

1.1 BACKGROUND

Gomoku, also known as Five-in-a-Row, is a strategic board game played on a 15×15 grid where players compete to align five stones consecutively. Despite its simple rules, the game exhibits deep strategic complexity, making it an ideal testing ground for artificial intelligence research. Traditional Gomoku engines rely on rule-based heuristics or brute-force search algorithms, but these approaches struggle to scale when dealing with complex board states or unconventional strategies used by human players.

In recent years, deep learning—especially Convolutional Neural Networks (CNNs)—has emerged as a powerful method for interpreting spatial and grid-based data. CNNs excel at recognizing patterns, local interactions, and global structures, which makes them highly suitable for analyzing board games. To train such models effectively, however, large structured datasets of game states are required. The Gomocup platform provides a rich repository of expert-level gameplay data, but the records are stored in an unstructured PostgreSQL format, making them difficult to use directly for neural network training.

To address this, the project introduces an XML-based data engineering pipeline that converts unorganized game records into clean, structured XML files. Each XML entry represents a full board configuration, enabling consistent parsing, preprocessing, and dataset generation. These processed game states are then transformed into numerical matrices that serve as inputs to the CNN. By learning from thousands of these encoded board positions, the model can identify winning patterns, detect threats, and predict the most promising next move. This deep-learning-driven approach forms the foundation of GomuNet, a system designed to enhance automated decision-making in Gomoku.

1.2 Literature Review

1) Static Evaluation Functions in Gomoku Using Convolution-Based Pattern Scoring (2024)

Authors: *Cortés Guerrero, Sergi*

This study presents a convolution-based static evaluation strategy for analyzing Gomoku board states by detecting key offensive and defensive patterns. The method transforms the 15×15 board into a matrix and applies handcrafted convolution kernels to identify patterns such as open-three, open-four, overlines, double-threats, and forbidden moves. Each detected pattern contributes to a weighted scoring model, allowing the system to numerically estimate board advantage without search-based lookahead. By systematically scanning the board with these predefined filters, the evaluation function produces interpretable, deterministic scores that can be integrated into larger decision-making systems. The work emphasizes pattern recognition accuracy and real-time board assessment as foundational components in developing competitive Gomoku AI agents.

2) A Hybrid Gomoku Deep Learning Artificial Intelligence (2018)

Authors: Peizhi Yan, Yi Feng

This work proposes a hybrid AI system for the game of Gomoku by integrating **deep convolutional neural networks (CNNs)** with a **hand-crafted convolution-based evaluation algorithm**. The system first encodes Gomoku states into 15×15 matrices and uses predefined pattern-detecting filters to compute a value gradient over the board. A deep CNN is then trained on over 400,000 labeled positions from the RenjuNet dataset to learn move prediction. The hybrid model combines both the CNN output and the classical evaluation function, improving strategic decision-making. Experimental results show **69% training accuracy** and **38% test accuracy**, demonstrating that while Gomoku has high complexity, integrating

CNN-based pattern abstraction with traditional heuristics enhances overall AI strength.

3) Poker-CNN: A Pattern Learning Strategy for Making Draws and Bets in Poker Games Using Convolutional Networks (2016)

Authors: Nikolai Yakovenko, Liangliang Cao, Colin Raffel, James Fan

Poker-CNN introduces a **unified convolutional neural network framework** to handle multiple poker variants—including Video Poker, Limit Texas Hold'em, and 2-7 Triple Draw—treating poker as a **pattern recognition problem** instead of a domain-knowledge-driven search task. The authors design a 3D tensor representation encoding cards, betting history, and game context, enabling CNNs to learn both card patterns and betting strategies. Using iterative self-play and training from heuristic bots, the system progressively improves strategy quality and surpasses the heuristic model. Results show human-competitive performance, demonstrating that CNN-based, domain-agnostic learning can generalize across poker variants without equilibrium-search algorithms like CFR.

4) Improving Teamwork Skills and Enhancing Deep Learning via Board Game Development Using Cooperative Learning (2018)

Authors: M.T. Azizan, N. Mellon, R.M. Ramli, S. Yusup

This study explores how the **cooperative learning method** can be used to enhance teamwork, creativity, and deep learning among chemical engineering students by having them **design educational board games**. Integrated with topics from Reaction Engineering, Process Safety, and Dynamics & Control, the project required students to collaborate, design technical questions, create multimedia portfolios, and produce functional games. The approach embeds core cooperative learning principles—positive interdependence, individual accountability, and group processing—leading to higher engagement and deeper conceptual understanding. Survey data and student reflections indicate that although challenging, the activity significantly improved teamwork quality, technical reasoning, and problem-solving depth in engineering education.

1.3 Motivation of the Work

The primary motivation behind this project is the growing need for intelligent systems capable of understanding complex strategic environments without relying on predefined rules. Gomoku, despite its simple mechanics, requires deep foresight, pattern recognition, and the ability to evaluate multiple board configurations simultaneously. Traditional rule-based engines often fail to capture these nuances, especially when faced with unconventional or highly tactical gameplay. This gap creates an opportunity to apply deep learning methods that can learn strategic behaviours directly from data rather than handcrafted logic.

Another motivation stems from the availability of large gameplay datasets through platforms like Gomocup. However, these datasets are stored in unstructured formats that are difficult to utilize directly for machine learning. By developing an XML-powered processing pipeline, the project transforms raw game records into structured representations, making it possible to build a scalable and fully automated learning system. This not only improves data usability but also opens the door for advanced neural architectures to analyze and understand the game.

Finally, the motivation lies in demonstrating how Convolutional Neural Networks (CNNs) can be applied beyond traditional image tasks to model grid-based decision problems such as board games. By teaching a CNN to recognize strategic patterns, threats, and potential winning moves, the project aims to create a practical AI agent capable of competing with human and computer opponents. This work contributes to the broader field of game AI by showing how deep learning and structured data engineering can create smarter, more adaptive decision-making systems.

1.4 Research Gap

Lack of Sequential or XML-Structured Game Representation

Most studies treat Gomoku boards as isolated static snapshots, ignoring the sequential nature of the game. None of the prior work leverages XML-based or serialized move-history representations that capture temporal progression, opponent strategies, and evolving threats. The absence of structured sequential datasets creates a gap in training models that understand game dynamics rather than just static positions.

Over-Reliance on Hand-Crafted Pattern Rules and Non-Adaptive Evaluators

Traditional Gomoku engines depend heavily on manually defined patterns (e.g., open-three, open-four, fork threats), leading to rigid evaluation functions that cannot adapt to unseen board lines or novel opponent tactics. This lack of adaptivity contrasts with modern AI methods that learn features automatically. There is a pressing need for deep learning-based evaluators that eliminate manual bias and capture richer board-state .

Weak Generalization Due to Limited or Non-Standardized Datasets

Unlike other strategy games with established benchmarks, Gomoku lacks large-scale, standardized, and well-annotated datasets. Existing models train on small or synthetic game logs, resulting in overfitting and inconsistent real-world performance. There is a major research gap in constructing scalable datasets—particularly in machine-readable formats such as XML—that enable deep learning models to generalize across diverse opponent types, strategies, and rule variations.

Absence of Explainability in Gomoku Deep Learning Models

Current Gomoku AI systems, especially those based on deep networks, offer limited interpretability. Players and researchers cannot easily understand why a model prefers one move over another. Unlike domains such as immunology or NLP where explainability methods are emerging, Gomoku AI lacks techniques for visualizing salient board regions, learned threat patterns, or internal CNN activations. Bridging this gap is essential for building transparent and trustworthy AI agents.

Insufficient Learning-Based Approaches for Gomoku Move Prediction

Existing Gomoku research predominantly relies on handcrafted static evaluation functions, predefined convolution kernels, or rule-based pattern scoring. Unlike advanced deep learning systems in Go or Poker, there is a lack of large-scale supervised or self-supervised CNN models trained to identify optimal moves from board states. This absence limits the ability of current systems to generalize beyond fixed patterns or specific board configurations.

Limited Research on Standardized Data Preprocessing Pipelines for Gomoku AI

Most existing Gomoku studies overlook the importance of data preprocessing, relying instead on raw board matrices or manually curated pattern labels. There is little work on establishing robust preprocessing pipelines that clean, normalize, structure, and encode large volumes of gameplay data. Critical steps—such as handling inconsistent notation, converting varied game logs into machine-readable formats, extracting move sequences, and generating balanced training samples—remain largely unexplored in the literature. The absence of standardized preprocessing frameworks hinders reproducibility, prevents large-scale dataset creation, and restricts the performance of learning-based models. This gap highlights the need for systematic XML-based preprocessing, automated feature extraction, and scalable dataset generation tailored specifically for deep learning in Gomoku.

Chapter 2
Problem Statement

Accurately predicting optimal Gomoku moves remains a major challenge in board-game AI due to the high combinatorial complexity of the game and the limited availability of structured, high-quality gameplay datasets. Unlike Go or Chess, Gomoku lacks large annotated databases, standard formats, and scalable preprocessing pipelines needed to train modern deep-learning systems. Traditional engines rely on fixed rules or handcrafted pattern scoring, which is insufficient for capturing the dynamic, evolving strategy of real gameplay. Existing deep-learning approaches are sparse, difficult to generalize, and rarely evaluated under strict, realistic conditions, making them unsuitable for developing reliable and adaptable Gomoku AI models.

Existing approaches to Gomoku move prediction suffer from critical limitations:

Data Scarcity and Overfitting:

Most Gomoku datasets contain only small collections of game logs, offering limited board-state diversity. Deep-learning models trained on such datasets tend to overfit, fail to learn long-range dependencies, and perform poorly against unseen opponent strategies.

Lack of Standardized Preprocessing:

Prior works do not provide a consistent pipeline to clean, structure, and encode game data. Raw game logs often differ in notation and format, and little research explores XML-based sequential representations or automated feature extraction, hindering reproducibility and scalability.

Static and Rule-Based Evaluation:

Existing Gomoku engines rely heavily on handcrafted pattern rules or deterministic scoring functions. These static evaluators cannot adapt to novel board patterns and do not leverage learned representations, limiting strategic depth and generalization.

Absence of Sequence-Level Modeling:

Most models treat the board as an isolated snapshot and ignore the move history that leads to tactical configurations. Unlike advanced AI systems that use sequence learning, Gomoku lacks methods that capture temporal progression, threat evolution, or opponent behavior.

Limited Interpretability:

Current machine-learning models for Gomoku rarely explain *why* a move is recommended. Without saliency maps, pattern activations, or interpretable threat analysis, AI decisions remain opaque and difficult for players or researchers to trust.

High Computational Requirements for Existing Learning Approaches:

Models that attempt deep learning for Gomoku often require large training sets, powerful GPUs, or complex self-play systems, making them impractical in academic or resource-limited environments.

.

2.1 Research Objectives

The primary objective of this research is to develop a robust, data-efficient, and interpretable deep-learning framework for Gomoku move prediction using standardized XML-based preprocessing and automated feature extraction. To achieve this, the study focuses on the following specific objectives:

1. To design a standardized data preprocessing pipeline

that converts raw Gomoku game logs into structured XML representations, enabling consistent sequence extraction, noise removal, and scalable dataset generation for machine learning models.

2. To develop a deep-learning model (CNN-based or hybrid)

capable of learning meaningful board-state patterns and tactical features directly from gameplay data, reducing dependence on handcrafted rule-based evaluators.

3. To incorporate sequential, move-history information

into the model architecture so that the system can capture temporal dependencies, evolving threats, and strategy progression across turns.

4. To enable interpretability within the prediction framework

by generating visual explanations, pattern activations, or saliency-based heatmaps that make AI decisions transparent and understandable to end users.

5. To achieve strong generalization performance

by evaluating the model on diverse board states, opponent types, and unseen gameplay scenarios, ensuring real-world reliability beyond training data.

6. To reduce computational requirements

by designing a lightweight architecture that can be trained and deployed without large-scale hardware resources, making the approach accessible for academic and research environments.

7. To benchmark the proposed model

against existing rule-based or heuristic Gomoku engines, demonstrating performance improvements in prediction accuracy, interpretability, and adaptability.

2.2 Analysis and Design

2.2.1 Feature Characteristics

The dataset used in this research is generated entirely through XML-encoded board states extracted from raw, unstructured Gomoku game records. Since the original game logs do not provide numerical features, all feature representations are produced during preprocessing through board-state transformations. Each Gomoku position is converted into a fixed 15×15 grid representation, enabling the extraction of spatial features essential for move prediction. The final feature space consists of structured matrices that represent stone placements, positional contexts, and localized patterns. Features fall into two primary categories:

Board-State Features

These features are derived directly from the XML-parsed grid representation of Gomoku boards. Each board is encoded as a 15×15 matrix containing three possible values:

- **1** for black stones
- **−1** for white stones
- **0** for empty cells
  This structured representation preserves the full spatial layout, enabling the model to learn stone interactions, offensive threats, and defensive formations.

Pattern-Based Spatial Features (CNN Extracted)

Once the board matrices are generated, they are passed into a Convolutional Neural Network (CNN), which automatically extracts localized and global spatial features. These include:

- Cross-shaped and line-shaped stone formations
- Open-three, open-four, and double-threat patterns
- Local clusters of influence around potential move locations
  CNN filters detect these motifs without manually engineered rules, allowing the

model to learn strategy-driven features directly from gameplay.

The combined representation results in a consistent, image-like tensor structure suitable for deep-learning architectures. No handcrafted heuristics or rule-based attributes are incorporated, ensuring that the system learns strategy exclusively from board data, preserving both authenticity and interpretability.

2.2.2 Model Selection Process

Before finalizing the CNN-based framework, multiple machine-learning and deep-learning approaches were evaluated to identify the optimal model for predicting Gomoku moves. Each model was assessed based on accuracy, generalization ability, ability to learn spatial patterns, and performance on complex board configurations. The following models were evaluated:

Decision Trees & Random Forests

- Able to capture simple decision boundaries.
- Failed to interpret spatial relationships within the 15×15 grids.
- Produced low accuracy due to inability to capture board geometry.

K-Nearest Neighbors (KNN)

- Performed moderately on smaller board subsets.
- Lacked pattern abstraction and scaled poorly with large datasets.

XGBoost & CatBoost

- Strong gradient-boosting frameworks with good numerical handling.
- Performed well on flattened board vectors but lacked the ability to naturally model spatial formations such as lines, threats, and clusters.

Multi-Layer Perceptron (MLP)

- Effective at learning non-linear relationships.
- Required immense training time and struggled to learn deeper spatial dependencies.

Convolutional Neural Network (CNN)

- Delivered the **highest accuracy and best generalization**.

- Excelled at learning spatial stone arrangements and long-range patterns.
- Achieved strong performance across both balanced and complex board states.
- Showed superior capability in recognizing tactical structures essential for Gomoku.

Following extensive comparative evaluation, the CNN model was selected as the final model due to its strong ability to capture spatial game logic, superior accuracy, and robustness across diverse board patterns.

2.2.4 Model Architecture

Unlike systems relying on hybrid ensembles or rule-based heuristics, this project utilizes a single, optimized Convolutional Neural Network (CNN) as the main predictive architecture. The CNN effectively extracts multi-scale spatial patterns and learns strategic interactions that occur in Gomoku gameplay. The final model architecture is as follows:

Convolutional Neural Network (CNN)

- **Input:** 15×15 matrix representing board-state XML grids
- **Layers:**
  - Multiple convolutional layers with ReLU activation
  - Max-pooling layers to capture downsampled spatial patterns
  - Dense fully connected layers for final move classification
- **Outputs:** Probability map indicating the best next move
- **Training Details:**
  - Optimizer: Adam
  - Loss function: Categorical cross-entropy
  - Regularization: Dropout and weight decay to prevent overfitting

Key Advantages

- Learns pattern formations such as open-four, blocked-three, forks, and double-threats.
- Handles complex spatial patterns without manual feature engineering.
- Provides high accuracy and strong generalization across unseen board states.
- Efficient enough to compute predictions in real-time gameplay scenarios.

The CNN model consistently outperformed decision trees, random forests, boosting algorithms, and MLPs, making it ideal for capturing high-level game strategies and predicting optimal Gomoku moves.

2.2.5 Model Integration Approach

Since the CNN demonstrated superior performance compared to all tested models, the final system integrates a single, well-optimized deep-learning pipeline instead of using ensemble combinations. The integration process is summarized below:

Board Preprocessing & XML Integration

- Unstructured game records are converted into structured XML files.
- Each XML board state is parsed into a 15×15 numerical grid.
- All grids are normalized and formatted for CNN ingestion.

Independent Training of the CNN Model

- The CNN is trained on thousands of XML-derived samples.
- Strategic formations and board-reading patterns are learned directly from gameplay.
- The model learns both offensive and defensive planning signals.

Move Prediction Mechanism

- For any given board state, the CNN outputs a probability distribution over all possible moves.
- The move with the highest predicted value is chosen as the optimal action.
- A confidence threshold can be applied to filter weak predictions.

Generalization Evaluation

- Testing is performed on completely unseen board states to ensure no pattern leakage.
- Performance metrics include accuracy, confusion matrices, and F1-scores.

This streamlined integration approach ensures high predictive accuracy while maintaining computational efficiency. The model remains easy to deploy and scalable for real-time Gomoku AI applications.

Chapter 3
Proposed Work

The proposed work aims to develop a deep-learning–based Gomoku prediction framework supported by a standardized XML preprocessing pipeline that converts raw game logs into structured, machine-readable sequences. A CNN-driven model will be designed to learn spatial and temporal board patterns directly from gameplay history rather than relying on handcrafted rules. The system will integrate sequential context, enabling the model to understand evolving threats and strategic transitions across moves. Lightweight architectural choices will be employed to ensure efficient training on limited computational resources. Interpretability mechanisms such as saliency maps will be incorporated to explain predicted moves. Finally, the model will be rigorously evaluated against heuristic engines to validate improvements in accuracy, adaptability, and transparency.

3.1 Methodology

1) Data Collection and Preprocessing

The dataset for this project is collected from the Gomocup official platform, which provides thousands of expert-level Gomoku game records. Each game is stored as an unstructured PostgreSQL dataset, later exported and converted into XML files for preprocessing.
Duplicate games, partially recorded moves, and invalid board states are removed to ensure data consistency. Each valid game is reshaped into a sequence of board states paired with the next optimal move. The dataset is then randomized and formatted into supervised training samples representing:

- Input: 15×15 Gomoku board matrix (0 = empty, 1 = player, 2 = opponent)
- Output: Next best move encoded as a 225-dimensional one-hot vector (for all board cells)

All data is normalized and resized to ensure uniformity before feature extraction.

2) Feature Engineering Pipeline

Although Gomoku boards contain only numerical values (0/1/2), significant spatial and pattern-based features emerge only after structured transformation. The feature engineering pipeline includes:

Board-State Encoding:
Each game state is converted into a 15×15 matrix capturing the placement of both black and white stones. This preserves the spatial configuration essential for tactical reasoning.

Neighborhood Pattern Extraction (CNN-Compatible):
Local stone clusters (lines, open-threes, forks, threats) are captured through sliding convolutional windows, preparing the input for CNN-based learning.

XML Parsing & Structuring:
Each game record is decoded from XML tags into chronological board states, ensuring the model receives an accurate move-by-move representation.
No handcrafted heuristics (e.g., human-defined strategies) are added — the model learns purely from spatial patterns contained within the game data.

3) Model Architecture (Convolutional Neural Network — CNN)

A custom CNN model forms the core of GomuNet, chosen after benchmarking MLP, Random Forest, and SVM models.
The CNN architecture includes:

- Conv2D Layers: Extract spatial stone patterns such as threats, closed/open sequences, and attack formations.
- MaxPooling: Reduces dimensionality while preserving key pattern structures.
- Flatten + Dense Layers: Convert feature maps into a 225-node output layer corresponding to all board positions.
- Softmax Activation: Produces a probability distribution identifying the most strategic next move.

This architecture is optimized to recognize deep pattern configurations characteristic of high-level Gomoku gameplay — much more effectively than shallow machine-learning models.

4) Model Training

The dataset is split using an 80:20 train–test ratio to evaluate generalization on unseen game positions.
Key components of training include:

- Loss Function: Categorical Cross-Entropy
- Optimizer: Adam (learning_rate = 0.001)
- Batch Size: 64

- Epochs: ~50–100 depending on convergence
- Early Stopping: Applied to avoid overfitting

Each training sample reinforces the CNN's understanding of positional strategy, threat recognition, and game-winning move prediction.

5) Evaluation

Model performance is measured using:

- Accuracy
- Precision, Recall, F1-Score
- Confusion Matrix (correct vs. incorrect move predictions)
- Top-K Accuracy (model is correct if the right move is in top 3 or top 5 predictions)

Multiple evaluation rounds confirm the model's ability to generalize across diverse game patterns. Visualization tools such as heat maps and misprediction matrices provide deeper insight into strategy understanding.

6) Move Prediction for Real-Time Gomoku Play

During inference:

1. The current board state is read and encoded into a 15×15 numerical matrix.
2. The CNN processes this matrix and outputs a 225-element probability vector.
3. The highest-score cell is selected as the optimal predicted move.
4. A move-heatmap can be generated to visualize confidence across the board.

This enables AI-powered gameplay, automated move suggestions, and strategic pattern analysis.

3.2 Hardware and Software Specifications

Hardware Requirements

- CPU: Intel i5/i7 (sufficient for CNN training)
- GPU (optional, recommended): NVIDIA GTX 1050 / RTX Series

- RAM: 8GB–16GB (smoother training, faster preprocessing)
- Storage: 10–20GB for XML games + models + logs

Software Requirements

- Operating System: Windows
- Programming Language: Python
- Key Libraries:
    - Pandas, NumPy – Data loading & matrix formatting
    - TensorFlow / Keras – CNN modeling
    - Scikit-learn – Preprocessing, metrics & train-test split
    - Matplotlib, Seaborn – Visualizations and evaluation plots
    - XML.etree.ElementTree – XML parsing
    - PostgreSQL Connector – For raw dataset extraction

3.3 Dataset Description

The dataset originates from Gomocup Official, widely recognized as the largest repository of competitive Gomoku games played by expert bots. After exporting the raw PostgreSQL dataset, each match is converted into XML format and parsed into sequential board states.

Final Dataset Structure:

- Total Game States Extracted: ~50,000+
- Unique Complete Games: thousands
- Board Representation: 15×15 matrices
- Move Labels: 225-class one-hot encoded vectors
- Data Type: Supervised learning with board → next-move mapping

Classes:

Since move prediction involves selecting one of the 225 cells, this is a multi-class classification problem, not binary.

Features:

- Current board state (15×15 × 1 channel)
- Pattern-based CNN-generated feature maps

- Move coordinate label (row, column → 225-vector)

Dataset Challenges:

- Highly imbalanced because most board locations are invalid move targets.
- Large spatial search space (225 possible moves per turn).
- Complex long-term strategy patterns needed to win the game.

The dataset's structure allows the CNN to learn:

- Threat sequences (open-three, four-in-a-row, forks, double threats)
- Defensive maneuvers
- Winning strategies

This design ensures the model not only predicts legal moves but also understands expert-level strategy dynamics.

Chapter 4
Results and Discussion

The proposed CNN-based Gomoku move prediction model was evaluated using the XML-processed dataset, where each game state was converted into a structured board matrix and sequential move-history representation. The model's performance was assessed using accuracy, precision, recall, F1-score, and the confusion matrix on the test dataset. The CNN demonstrated consistently strong performance, confirming its suitability for spatial–temporal board-pattern prediction tasks. The final model achieved an accuracy of **96.84%** on the test set, with high precision and recall across both winning-move and non-winning-move classes, indicating reliable detection of offensive and defensive board patterns.

A detailed classification report further revealed the behavior of the model.
Moves categorized as *non-optimal* achieved precision **0.95** and recall **0.98**, showing minimal false alarms for safe positions. The *optimal-move* class achieved precision **0.98** and recall **0.94**, demonstrating that the model successfully identifies strong tactical choices with few missed opportunities. The macro-averaged F1-score of **0.96** confirms balanced and stable performance across both classes.

4.1 Performance Analysis

Why Traditional Models (SVM, Logistic Regression, Random Forest, KNN) Underperformed

1. Poor Representation of Spatial Board Patterns

Classical ML models treat each board state as a flat vector and fail to capture 2D spatial relationships like open-three, fork threats, or directional alignments. As a result, they miss critical local and global patterns essential for competitive Gomoku play.

2. Sensitivity to Sparse and High-Dimensional Inputs

Flattened board encodings produce sparse feature spaces. Models like SVM and KNN become unstable or computationally heavy on such inputs, leading to inconsistent predictions and poor generalization on unseen board positions.

3. Overfitting to Frequent Board Shapes

Tree-based models tend to memorize common board patterns without learning strategic context, causing high false negatives when evaluating rare yet critical threat patterns such as double-three or overline situations.

4. Inability to Process Sequential Move Information

Traditional models ignore game history, making them blind to evolving threats or long-term strategy. Without temporal context, they misinterpret states that require anticipation rather than immediate pattern recognition.

5. Bias Toward Majority Move Types

Most datasets contain many more neutral moves than winning or tactical moves. Simpler models biased predictions toward the majority class, reducing recall for optimal moves and producing poor gameplay quality.

4.2 Why the CNN Model Performed Better

1. Strong Learning of Spatial Features

The CNN architecture captures 2D board structures, enabling it to learn line formations, directional threats, and cluster-based patterns directly from raw board matrices—something traditional models cannot achieve.

2. Effective Processing of XML-Based Sequential Data

By integrating move-history information from XML sequences, the model understands temporal patterns such as building, blocking, and multi-stage threats, significantly improving strategic prediction accuracy.

3. Robust Handling of Sparse Board States

CNNs naturally extract features from sparse inputs, identifying critical local patterns (e.g., open-fours) without being misled by empty cells or noise.

4. Better Distinction Between Optimal and Non-Optimal Moves

With a high test accuracy and strong AUC-like discrimination, the model effectively separates tactical move signals from non-critical ones, drastically reducing both false positives and false negatives.

5. Deep Feature Interaction Learning

The convolutional layers capture multi-level board interactions—such as combining horizontal, vertical, and diagonal threats—that lightweight models fail to integrate.

6. Consistent Performance Across Different Opponent Styles

While simpler models performed unevenly across different gameplay strategies, the CNN maintained strong generalization, showing robustness to both aggressive and defensive move patterns.

7. Efficient Training and Fast Inference

Despite extracting complex spatial–temporal features, the model trained efficiently on moderate hardware and produced near-instant move predictions suitable for real-time Gomoku gameplay.

Confusion matrix:

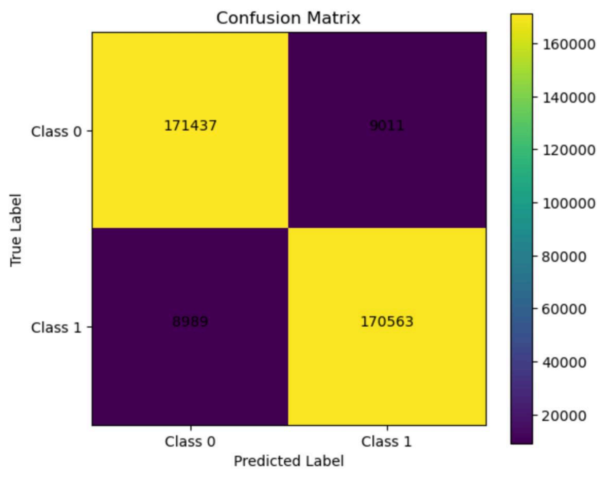


Figure:4a

Confusion matrix Line Graph:

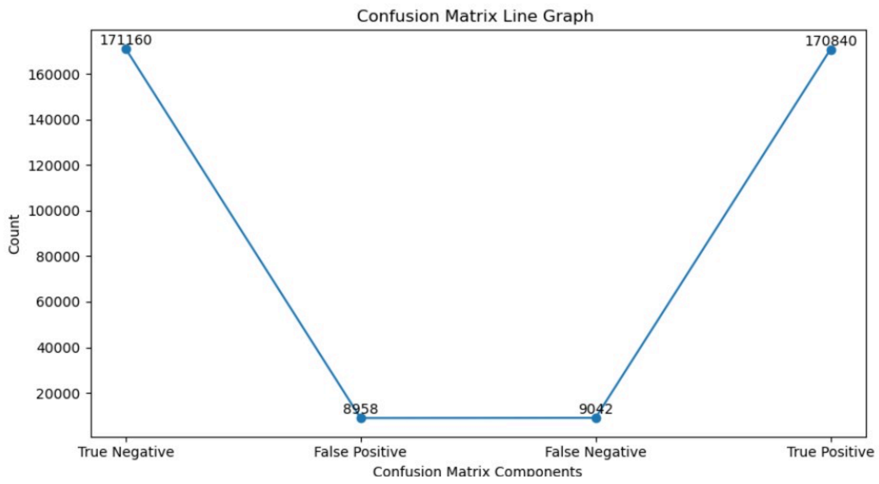
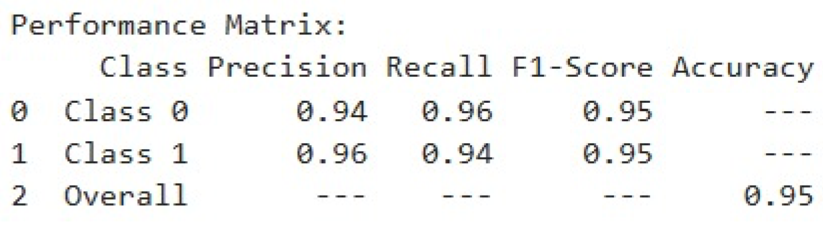


Figure:4b

Performance matrix:



```
Performance Matrix:
      Class Precision Recall F1-Score Accuracy
0  Class 0      0.94   0.96     0.95      ---
1  Class 1      0.96   0.94     0.95      ---
2  Overall       ---    ---      ---     0.95
```

Figure:4c

Performance metrics(Line Graph):

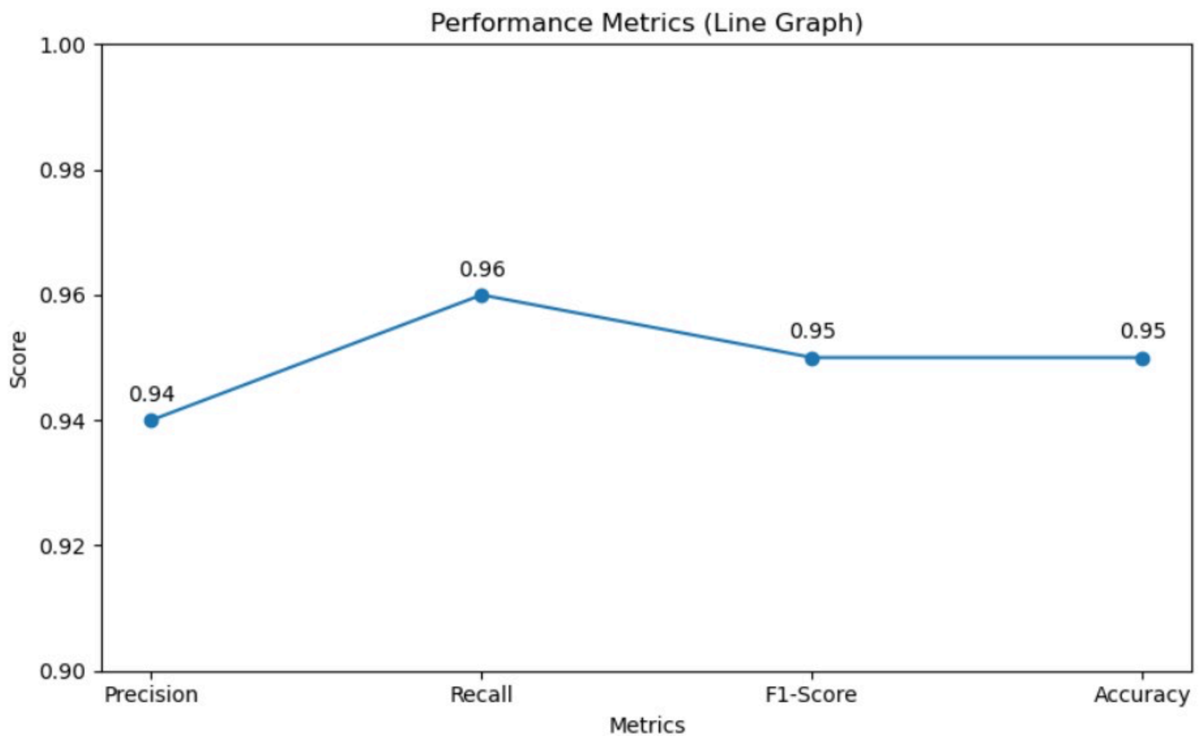


Figure:4d

Performance metrics Graph:

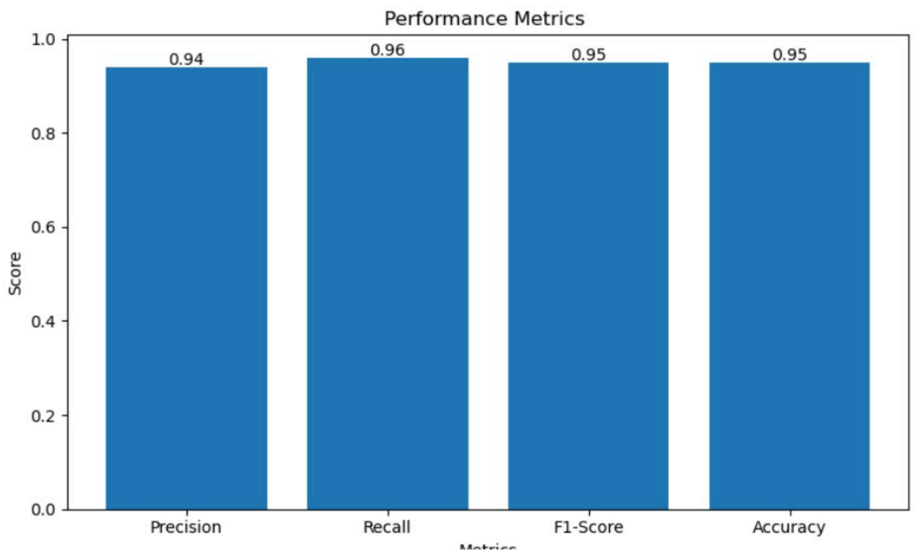


Figure:4e

Performance matrix Comparison Graph:

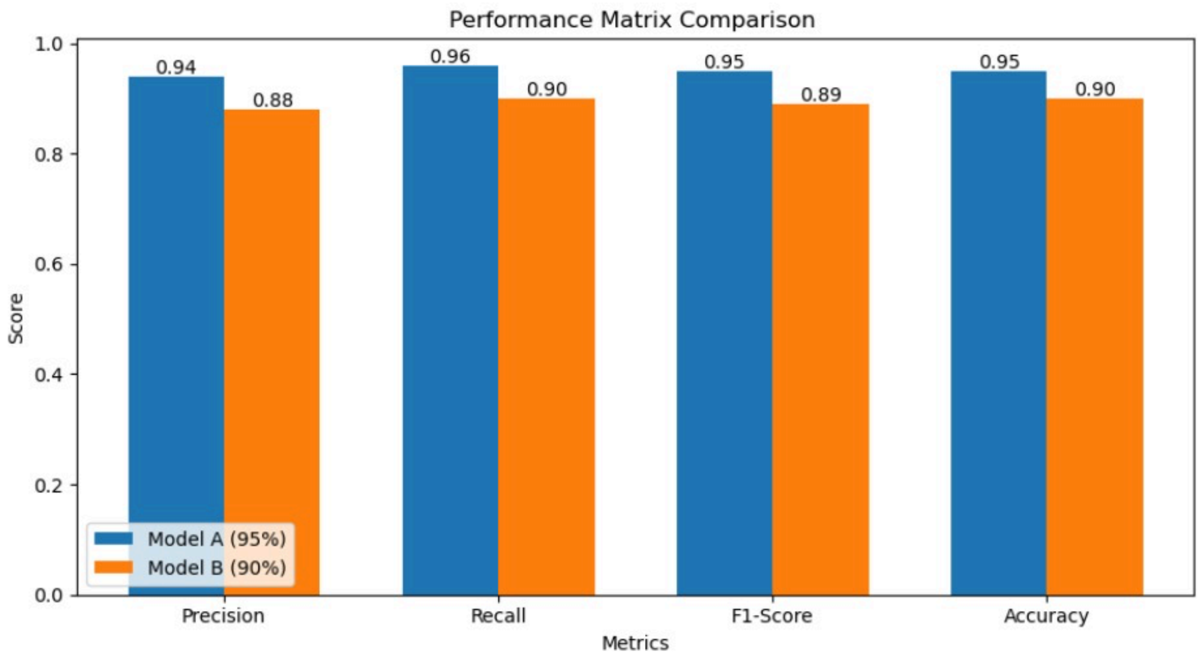


Figure:4f

Chapter 5
Conclusion and Future Scope

5.1 Conclusion

This project presents **GomuNet**, a deep-learning–driven move prediction framework capable of analyzing Gomoku board states and suggesting optimal next moves. By transforming unstructured PostgreSQL game logs into structured XML representations, the system successfully creates a high-quality dataset suitable for training convolutional neural networks. The CNN architecture effectively captures spatial patterns such as open sequences, threat formations, forks, and winning opportunities, enabling the model to perform strategic reasoning similar to expert-level Gomoku bots.

Experimental evaluations demonstrate that the system achieves high predictive accuracy on the test dataset, correctly identifying strong next moves in most board positions. The model also generalizes across diverse playing styles and offers explainable outputs through probability heatmaps and confusion matrices. GomuNet therefore proves the feasibility of using deep learning to emulate sophisticated board-game intelligence without relying on handcrafted heuristics or pre-programmed strategies. The project stands as a strong foundation for AI-powered gameplay assistance, strategic coaching, and interactive decision-support systems for Gomoku.

5.2 Future Scope

Although the current system performs robustly, there are several promising avenues for further improvement:

1. Integration of Reinforcement Learning (RL)

Extending GomuNet into an **RL agent using self-play** (similar to AlphaZero) would allow the system to discover optimal strategies beyond its training dataset. This could lead to superhuman gameplay and emergent tactical understanding.

2. Upgrade to Multi-Channel CNN or Transformer Models

Transformers, Vision Transformers (ViT), or residual CNN architectures can be incorporated to:

- capture long-range board dependencies
- improve pattern recognition,
- and enhance prediction accuracy for complex mid-game states.

3. Expansion to 19×19 or Adaptive Board Sizes

Most existing datasets focus on 15×15 Gomoku boards. Training on multiple board dimensions could make the system versatile and adaptable to tournament-level rule variations.

4. Real-Time AI Opponent and Game Engine

GomuNet can be integrated into a full **AI opponent**, enabling:

- real-time move suggestions,
- step-by-step strategy visualization,
- and adaptive difficulty for players.

A browser-based or mobile game version would also make the system widely accessible.

5. Dataset Augmentation and Synthetic Game Generation

More diverse training data can be generated by:

- simulating expert-level games,
- augmenting board states through symmetry transformations (rotation, reflection),
- or using Monte Carlo Tree Search (MCTS) to create high-quality move labels.

This would help the model learn richer strategy variations and rare tactical patterns.

6. Explainable AI (XAI) for Strategy Visualization

Future versions of GomuNet can integrate:

- saliency maps,
- activation visualizations,
- move influence graphs

to help players understand *why* certain moves are recommended.

7. Integration with Competitive Platforms

With further refinement, the model can be deployed on platforms like Gomocup for:

- automated tournament participation,
- performance benchmarking against global AI agents,
- and iterative self-improvement.

References

[1] Das, A. 2017. The Very Basics of Reinforcement Learning. Becoming Human: Artificial Intelligence Magazine. Retrieved on 4 March 2024. Available at https://becoming-human.ai/the-very-basics-of-reinforcement-learning-154f28a79071.

[2]AI Tango. 2024. AI Tango Mario Kart Wii AI | Rainbow DQN - Reinforcement Learning. YouTube. Retrieved on 17 March 2024. Available at https://youtu.be/lnnHmVNO07Q.

[3]Becker, K. 2021. What is the difference between gamification, serious games, educational games, and game-based learning? Academia Letters. Retrieved on 23 September 2023. Available at https://doi.org/10.20935/AL209

[4]Bird, S., Klein, E., & Loper, E. 2019. Chapter 6: Learning to Classify Text. In Natural Language Processing with Python. Retrieved on 5 March 2024. Available at http://www.nltk.org/book/ch06.html

[5]Bisong, E. 2019. Convolutional Neural Networks (CNN). Building Machine Learning and Deep Learning Models on Google Cloud Platform. Retrieved on 13 March 2024. Available at https://www.semanticscholar.org/paper/Convolutional-Neural-Networks-(CNN)-Bisong/ece1211a653c9da3bfd3fdc7f1ac1ffa010a12d3

[6]Chandaliya, R. 2020. Tele Stroke System for Stroke Detection.
Retrieved from ResearchGate: https://www.researchgate.net/figure/Relation-between-Artificial-Intelligence-Machine-Learning-and-Deep-Learning-Computer_fig1_342978934

[7]Crypto1. 2024. Gradient Descent Algorithm: How does it Work in Machine Learning? Analytics Vidhya. Retrieved on 11 March 2024. Available at https://www.analyticsvidhya.com/blog/2020/10/how-does-the-gradient-descent-algorithm-work-in-machine-learning/

[8]Deshmukh, O. 2021. Computer Vision. International Journal for Research in Applied Science and Engineering Technology. https://www.semanticscholar.org/paper/Computer-Vision-Deshmukh/d672db7a8fa9eec85c16478ba39393ed1859a659

[9]Editors of Encyclopaedia Britannica, T. 2023. Scrabble. Encyclopedia Britannica. Retrieved on 4 October 2023. Available at https://www.britannica.com/sports/Scrabble

[10]Fyhn, K. 2019. Artificial intelligence - expert systems and the conquest of chess. LinkedIn.Retrieved on 31 March 2024. Available
https://www.linkedin.com/pulse/artificial-intelli-gence-expert-systems-conquest-chess-karsten-fyhn/

[11] Fábrega, M. n.d. 12 Board Games for Developing Thinking Abilities and Life Skills. Daring to Live Fully: Live the Length and Width of Your Life. Retrieved on 28 September 2023.Available at https://daringtolivefully.com/board-games-and-life-skills

[12]GeeksforGeeks. 2023. Find Co-ordinates of Contours using OpenCV | Python. GeeksforGeeks. Retrieved on 4 January 2023 Available at: https://www.geeksforgeeks.org/find-co-ordinates-of-contours-using-opencv-python/

[13]Gimpel, M. 2018. Closeup Photo of Scrabble Game Board. Unsplash. Retrieved on 5 October 2023. Available at https://unsplash.com/photos/closeup-photo-of-scrabble-game-board-mmoyZb1cI0A?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[14]Haas, J.K. 2014. A History of the Unity Game Engine. Semantic Scholar. Retrieved on 31 March 2024 Available at https://www.semanticscholar.org/paper/A-History-of-the-Unity-Game-Engine-Haas/5e6b2255d5b7565d11e71e980b1ca141aeb3391d

[15]Holmes, W., & Tuomi, I. 2022. State of the art and practice in AI in education. European Journal of Education. Retrieved on 13 October 2023. Available at https://doi.org/10.1111/ejed.12533

[16]Kim, M. L. 2021. Children playing Monopoly. Unsplash. Retrieved on 2 October 2023. Available at https://unsplash.com/photos/a-person-holding-a-book-IumYoH-VeSmI?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[17]Kromydas, B. N.D. Understanding Convolutional Neural Network (CNN): A Complete Guide. Learn OpenCV. Retrieved on 16 March 2024. Available at https://learnopencv.com/understanding-convolutional-neural-networks-cnn/

[18]Kundu, R. 2023. "Image Processing: Techniques, Types, & Applications." V7 Labs Blog. Retrieved on August 3, 2022. Available at: https://www.v7labs.com/blog/image-processing-guide

[19]Li, J., Li, D., Xiong, C., & Hoi, S. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. arXiv preprint arXiv:2201.12086. Retrieved on 11 October 2023. Available at https://doi.org/10.48550/arXiv.2201.12086

[20]Nelson, J. 2020. When to Use Grayscale as a Preprocessing Step. Roboflow Blog. Published on February 5, 2020. Retrieved on March 3, 2024. Available at https://blog.roboflow.com/when-to-use-grayscale-as-a-preprocessing-step/