Chess Board Classification

Vidya Preetha Anandamurali, Oindrilla Chatterjee, Leo Kosta, Sha Lai

Outline

- Motivation
- Problem Statement
- Background
- Model 1: SVM
- Model 2: MLP
- Model 3: CNN
- Results
- Model Interpretation
- Conclusions
- Future Work

Motivation

- Chess is important!
  - Teaches us about risk, strategy, and consequences
  - Analogous to many real-life situations
  - Cultural significance
  - Lots of people play
- Chess is a living game
- We want to use machine learning techniques to learn more about chess

Problem Statement

- Given only the orientation of pieces on the board, can we predict the outcome of a given chess game?
- Binary classification problem (for simplicity, ignore stalemates)
- Dataset: Kaggle chess dataset of more than 20,000 games

- **Despite implementing/tuning various models, we had only limited success**

Background

- Each board represented as a 64x13 matrix
  - Board: 8x8 = 64 tiles
  - States: 13
    (Empty, W/B King, Queen, Rook, Bishop, Knight, Pawn)
- Generate all the boards in a game
  - Play moves 2 at a time (always white to play)
  - Skip first N boards
  - Label is the game outcome
- Final dataset: 601,253 labeled boards (balanced)

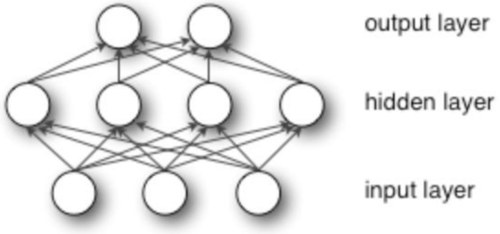
8
7
6
5
4
3
2
1
A
B
C
D
E
F
G
H

Model 1: Support Vector Machine (SVM)

- Vectorize each piece of data in the dataset
- Randomly sample a subset of the data to perform SVM
- Repeat the previous step multiple times
- Preprocesses the data with PCA techniques and repeat the above steps

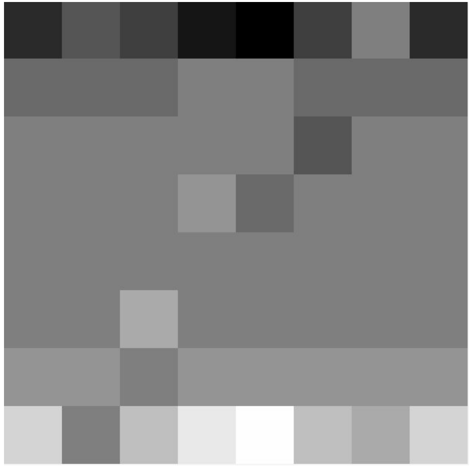
Model 2: Multilayer Perceptron (MLP)

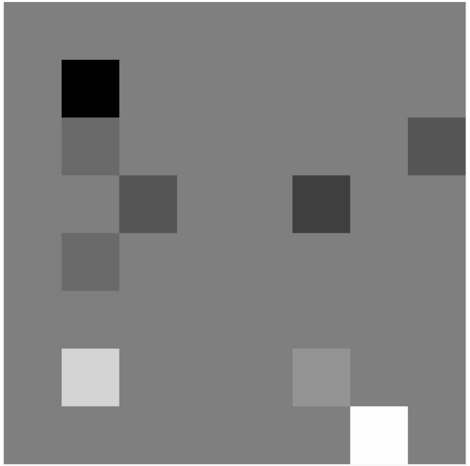
- Treat the boards as images.
- Since a single hidden layer is enough make MLPs a universal estimator, we use only one hidden layer in this model.



Model 3: Convolutional Neural Network (CNN)

- Assumption: we can treat the board like an image
  - Most valuable pieces map to higher intensity white/black
  - Blank is gray
- Use convolutions to identify features on a subset of the board



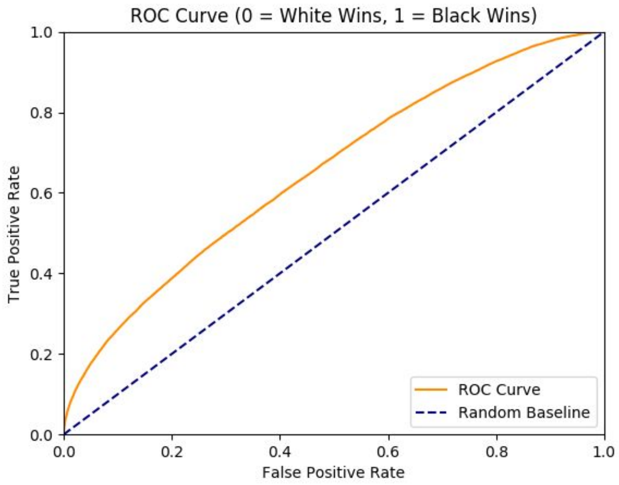


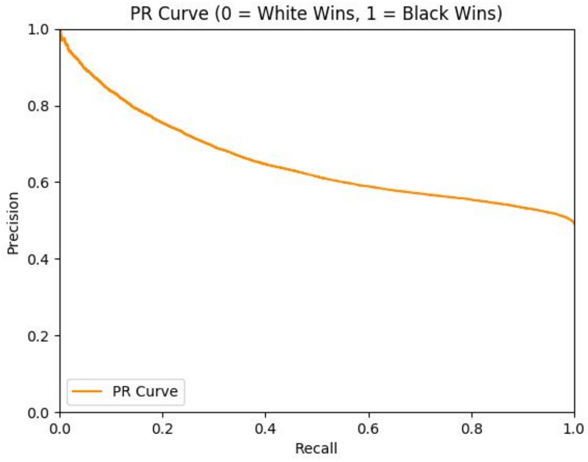
Layer	Size
Input	64
Conv1	32 5x5 Filters
MaxPool1	2x2
Conv2	64 3x3 Filters
MaxPool2	2x2
Dense	256
Output	2 (or 3)

Results

- Hyperparameter tuning done on all models
- Best model picked by test accuracy
- Graphics shown for best model

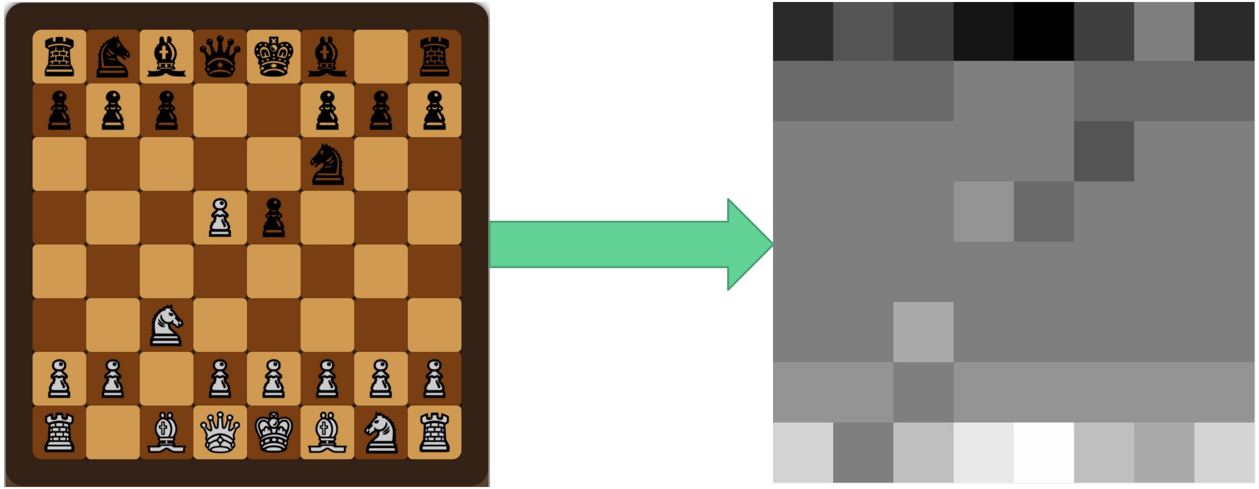
Model	Training Accuracy	Test Accuracy	ROC AUC
SVM	0.816	0.515	0.7201
MLP	0.655	0.480	0.600
CNN	0.627	0.598	0.598





Model Interpretation

- How should we think about chess?
    - If CNN is the best model, maybe we should think about it like an image
- What do our models tell us about superior position?
    - What kind of trends do we see when both sides have the same or similar material?
- Which openings are dominant?



Conclusions

- Classification of chess boards is a difficult problem
  - Situation changes rapidly - each move is important
- CNN seems to be the best model, but it still does not perform very well
- A model that captures context may do better
  - Context: previous moves or games

Future Work

- Improve model performance using ensembling techniques
  - Goal: 60% accuracy on the test set
- Naive model: who has more pieces?
- Extend classifier into a chess AI
  - Enumerate all possible moves in a turn
  - Find the move which most probably leads to victory
  - Play the move
- Try approaches on Go