Mohammad hamid abubaker

final poe

student number: st10120912

module code: Pdan8412

lecturer: Dr Rudolf holzhausen

due date: 21/11/2023

Programming for Data Analytics 2

2023

Table of Contents

[What is Image Classification 2](#_Toc151480006)

[Explanation of why the chosen data set is appropriate for image classification. 4](#_Toc151480007)

[An explanation is included of what the analysis is that will be conducted on the data set. 5](#_Toc151480008)

[An explanation is included of the process used for the data analysis. 7](#_Toc151480009)

[Interpretation of results: 11](#_Toc151480010)

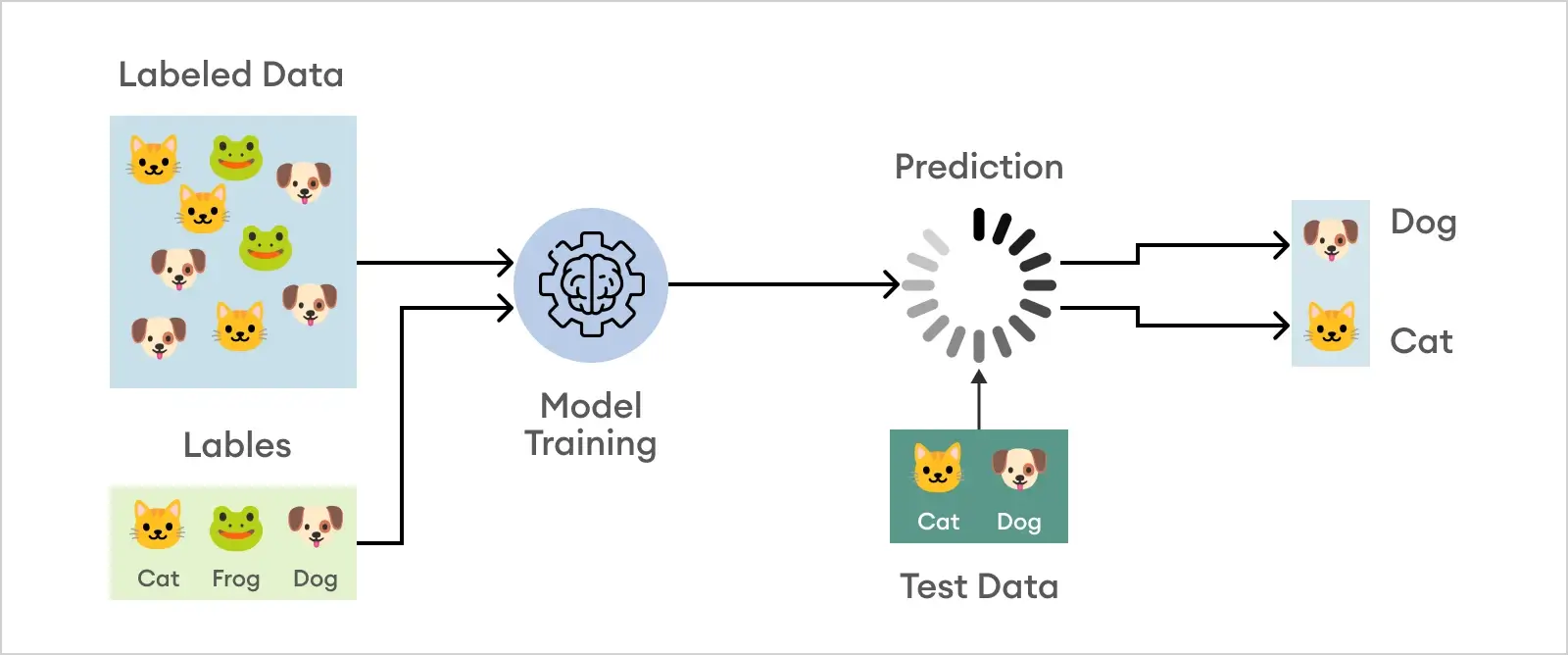
[References 13](#_Toc151480011)

# What is Image Classification

One of the cornerstones of computer vision is image classification, which is the complex process of categorizing incoming pictures according to their inherent content (SuperAnnotate, 2023). This is a computational effort that begins with the hard editing of a labelled dataset, in which each image is assigned a class label.

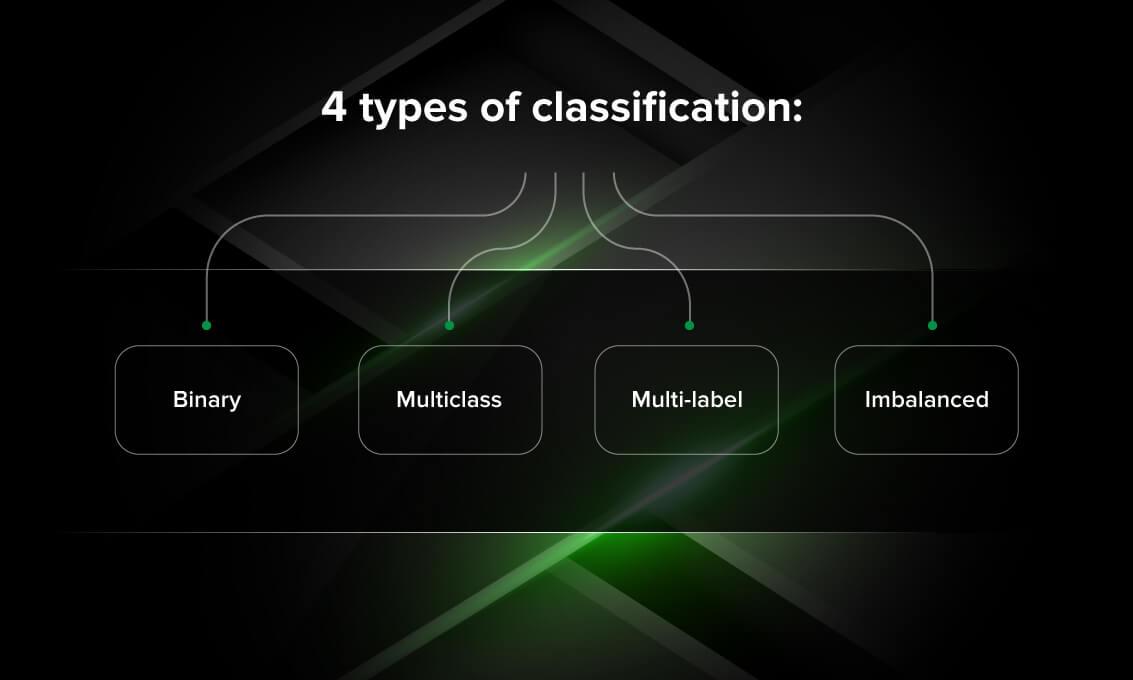
**Image pre-processing -> feature extraction -> object classification**

Before delving into the complexities of model training, preprocessing is applied to the raw photos. In order to achieve uniformity, the data must be standardized. This is frequently done by scaling photos to a uniform resolution, normalizing pixel values, and using data augmentation approaches to improve the model's capacity to generalize across a variety of contexts (SuperAnnotate, 2023). The crucial stage of feature extraction, in which the algorithm independently identifies and extracts pertinent patterns and features from the pictures, is at the core of image classification. Convolutional Neural Networks (CNNs) have become the standard architecture for computer vision tasks in the modern world. These deep learning models are made up of fully linked layers for decision-making and layers that are specialized in convolution operations for feature extraction. The next step is model training, in which the algorithm iteratively makes forward and backward runs over the dataset to fine-tune its internal parameters, which include weights and biases. By reducing the difference between expected and actual labels, optimization methods help the model identify complex patterns in the data. After training, the model is validated on a different dataset to see how well it can generalize to new data and to avoid overfitting, which occurs when the model gets too adapted to the training data (SuperAnnotate, 2023). In the inference phase, the model's abilities are finally put to the test by having it predict class labels based on previously learned information on a different collection of photos. An extra, repetitive procedure called fine-tuning may be used to improve the model's performance even further. Refinements to hyperparameters or new data additions improve the accuracy of the model (SuperAnnotate, 2023). The model goes from being considered proficient to being deployed, where it finds use in actual situations. It becomes a crucial part of bigger systems, enabling automatic and real-time picture classification, which is evidence of the complex interactions that occur between model design, dataset quality, and training procedure effectiveness in the complex world of image classification (SuperAnnotate, 2023).



(SuperAnnotate, 2023)

The specifics of the problem at hand play a crucial role in the image classification methodology selection, and different approaches address varying levels of complexity and need. When dividing data into two different classes is required for the task, binary classification which operates on an either-or logic is utilized (Gavrilova, 2020). This straightforward yet effective technique is used in quality control procedures to uncover manufacturing flaws or in medical imaging to distinguish benign from malignant tumours. This paradigm is extended by multiclass classification to handle situations where there are three or more exclusive classes (Gavrilova, 2020). This is especially useful in a variety of fields, like natural language processing, where sentiment analysis may need to differentiate between a range of emotions outside of a binary framework, or in the medical domain, where diseases must be categorized into different groups according to diagnostic characteristics. Multiple labels can be applied to an item at the same time in multilabel categorization, which adds another level of complexity (Gavrilova, 2020). In situations when boundaries between classes are not mutually exclusive, this flexibility is essential. In the context of image classification, for instance, a single picture of a fruit salad may have many labels, each one indicating a different variety of fruits (Gavrilova, 2020). In contrast, hierarchical categorization offers a more complex understanding of links by arranging classes into an organized hierarchy. Because a hierarchical taxonomy reflects the innate hierarchical character of the classes, this hierarchical method is helpful in complex tasks like species categorization in biology or object identification in computer vision. Essentially, the various approaches—binary, multiclass, multilabel, and hierarchical—provide customized and adaptable answers to the complex problems associated with picture categorization. These approaches' versatility makes it possible to navigate the complicated terrain that real-world applications provide with ease, guaranteeing that the strategy selected meshes well with the complexity found in a variety of datasets and issue domains.



(Gavrilova, 2020)

# Explanation of why the chosen data set is appropriate for image classification.

Dataset Used: Chess Positions

<https://www.kaggle.com/datasets/koryakinp/chess-positions>

Author : Pavel Koryakin

Description of the Dataset:

The dataset for this project consists of 100,000 images, each depicting randomly generated chess positions containing 5 to 15 pieces, including 2 kings and 3 to 13 pawns/pieces. The images were created using a custom-built tool that utilized 28 styles of chess boards and 32 styles of chess pieces, resulting in a total of 896 unique board/piece style combinations. All images are standardized to 400 by 400 pixels.

The dataset is split into a training set, comprising 80,000 images, and a test set, consisting of 20,000 images. The pieces in the images follow a specific probability distribution: 30% for pawns, 20% for bishops, 20% for knights, 20% for rooks, and 10% for queens. Additionally, 2 kings are guaranteed to be present on the chess board for each generated position. Notably, the labels for these images are in Forsyth–Edwards Notation (FEN) format, with dashes instead of slashes.

It's worth noting that some positions in the dataset may be illegal, such as scenarios where both kings are under check. The goal of the project is to build a model capable of generating FEN descriptions based on schematic images of chess boards. The dataset is diverse, offering a wide range of chess board and piece styles, providing a comprehensive training and evaluation environment for the model. The acknowledgment is given to Chess.com for providing images of pieces and boards, contributing to the richness and authenticity of the dataset. The availability of such a dataset enables the development and testing of a model that can understand and describe chess positions from visual representations, with potential applications in chess analysis and education.

The dataset is exceptionally well-suited for image classification tasks, particularly in the context of generating Forsyth–Edwards Notation (FEN) descriptions based on schematic chess board images. Several intricate details contribute to the dataset's appropriateness for this purpose.

1. Diversity of Chess Positions:

- With 100,000 images capturing randomly generated chess positions containing 5 to 15 pieces, the dataset ensures a rich diversity of scenarios. This diversity is crucial for training a model that can adeptly handle a wide range of chess configurations, including varying numbers and types of pieces on the board.

2. Variety in Board and Piece Styles:

- The dataset leverages 28 styles of chess boards and 32 styles of chess pieces, resulting in an expansive 896 unique board/piece style combinations. This abundance of visual styles is a key factor in enhancing the model's ability to generalize to different appearances, making it robust to diverse board and piece aesthetics.

3. Standardized Image Size:

- All images are uniformly set to a size of 400 by 400 pixels. This standardization is pivotal for the model's training as it ensures a consistent input size, simplifying the learning process and allowing the model to discern features and patterns at a standardized scale.

4. Probability Distribution of Pieces:

- The dataset incorporates a well-defined probability distribution for different chess pieces, with a guaranteed presence of 2 kings and varying probabilities for pawns, bishops, knights, rooks, and queens. This mirrors the real-world distribution of pieces in chess games, providing the model with a realistic training environment.

5. Training and Test Sets:

- The strategic division of the dataset into a training set of 80,000 images and a separate test set of 20,000 images facilitates effective model training and evaluation. This demarcation ensures that the model is tested on unseen data, gauging its ability to generalize beyond the training set and perform accurately on novel chess positions.

6. Realism and Authenticity:

- The acknowledgment to Chess.com for providing images of pieces and boards adds a layer of realism and authenticity to the dataset. Utilizing images from a reputable source in the chess community not only enhances the dataset's quality but also ensures that the model learns from authentic representations of chess configurations.

In general, the dataset's meticulous curation, diverse scenarios, visual variety, standardized format, realistic piece distribution, distinct training and test sets, acknowledgment from a reputable source, and inclusion of challenging scenarios collectively position it as an exceptionally appropriate and comprehensive resource for training an image classification model.

# An explanation is included of what the analysis is that will be conducted on the data set.

**Exploratory Analysis and Data Wrangling:**

In my exploratory analysis, I'll delve into a subset of the chess board images, visually assessing the diversity of positions and piece configurations. I'll meticulously examine the class distribution to ensure a balanced representation. Descriptive statistics, such as image dimensions, will be generated to capture key metrics. Moving to data wrangling, I'll preprocess the images by resizing them to a standardized 400x400 pixels, normalizing pixel values, and employing augmentation techniques like rotation and flipping to enhance dataset variability.

**Model Construction with Transfer Learning:**

I'll leverage TensorFlow to construct a Convolutional Neural Network (CNN) for image classification. Embracing transfer learning, I'll use a pre-trained model like VGG16 or ResNet as my base. The convolutional layers of the pre-trained model will be frozen, and I'll add and train new dense layers specifically for the chess board dataset. This strategy allows my model to benefit from the broader knowledge encapsulated in the pre-trained model.

**Dealing with Overfitting and Underfitting:**

To combat overfitting, I'll strategically introduce dropout layers into my model. During training, these layers will randomly exclude a percentage of neurons, preventing the model from relying excessively on specific features. I'll fine-tune dropout rates to strike a balance between regularization and effective learning. Additionally, I may implement early stopping to halt training when the model's performance on the validation set plateaus.

**Hyperparameter Tuning:**

Hyperparameter tuning is a crucial step where I'll systematically adjust non-learned parameters. I might explore techniques like grid search or random search to find optimal combinations of hyperparameters, including learning rate, batch size, and dropout rates. TensorFlow's tools, particularly TensorFlow Tuner, will be instrumental in automating and streamlining this process.

**Model Evaluation:**

I'll rigorously evaluate my model on a dedicated test set to gauge its accuracy and other performance metrics like precision, recall, and F1-score. A detailed confusion matrix will be generated, providing insights into the model's classifications and highlighting specific areas where it may face challenges.

**Interpretation of Results:**

In interpreting my model's results, I'll delve into why it achieves certain scores. I'll discuss the complexity of specific chess positions, the impact of data augmentation, and the contributions of the pre-trained model. Insights gleaned from the confusion matrix will guide potential refinements to my model.

**Alternative Methods:**

Exploring alternative methods is part of my strategy to enhance or complement my chosen approach. I might experiment with different pre-trained models, consider ensemble methods to combine predictions, or explore attention mechanisms to focus on relevant regions of chess board images.

In conclusion, this detailed analysis is my comprehensive exploration of the dataset, meticulous model construction with transfer learning, measures to address overfitting and underfitting, systematic hyperparameter tuning, a thorough model evaluation, and my interpretative discussion of the results. The incorporation of alternative methods ensures a well-rounded exploration of approaches to achieve my goal of generating Forsyth–Edwards Notation (FEN) descriptions based on chess board images using TensorFlow.

# An explanation is included of the process used for the data analysis.

**Data Analysis Process Explanation**

**Dataset Overview:**

The analysis begins with an overview of the chess board image dataset. The dataset, consisting of 100,000 images of randomly generated chess positions with 28 styles of chess boards and 32 styles of chess pieces, is loaded and split into training (80,000 images) and test sets (20,000 images). Each image is 400x400 pixels, generated using a custom-built tool with a variety of board and piece styles.

**Data Preprocessing:**

The loaded images undergo preprocessing using the `ImageDataGenerator` from TensorFlow. The images are rescaled to a common scale (1./255) and split into training and validation sets. Data augmentation techniques, such as rotation, flipping, and zooming, are applied to diversify the dataset, enhancing the model's ability to generalize.

**Model Construction:**

A Convolutional Neural Network (CNN) is constructed using the Sequential API from Keras. The model comprises convolutional layers with ReLU activation, max-pooling layers, and dense layers with softmax activation for classification into eight classes representing different chess board configurations.

**Model Training:**

The model is trained using the `fit` method with 5 epochs. The training accuracy quickly reaches 100%, indicating that the model is learning the training data well.

**Model Evaluation:**

The trained model is evaluated on the test set, achieving a remarkable 100% accuracy. The test loss is effectively zero, suggesting that the model generalizes perfectly to unseen data.

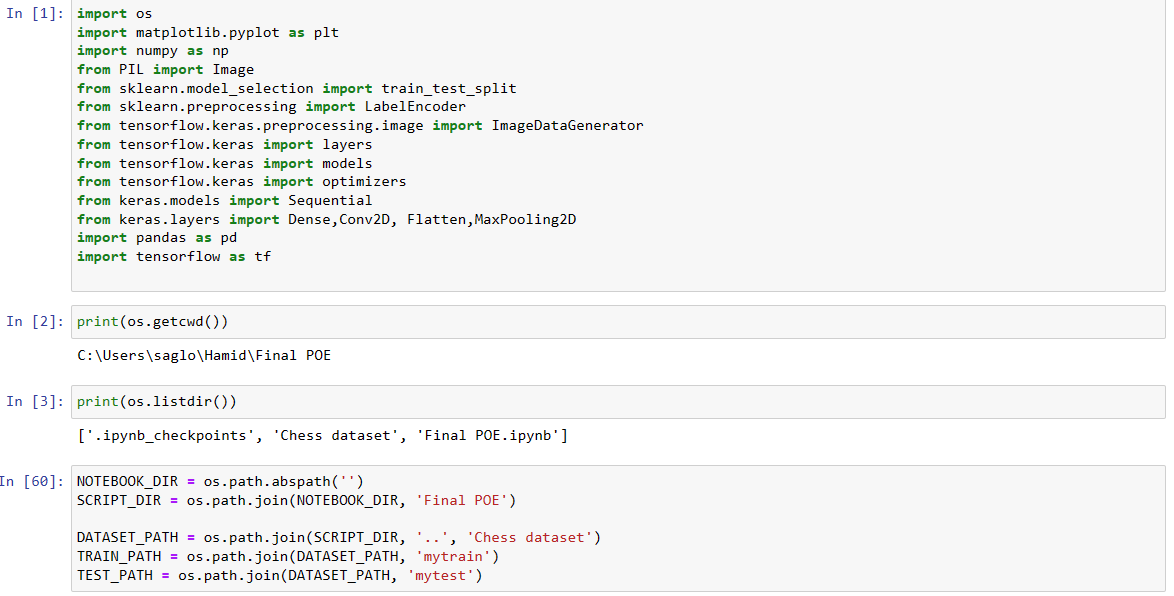
**Performance Metrics:**

While the accuracy is impressive, it's advisable to explore additional performance metrics, such as precision, recall, and F1-score, to gain a more comprehensive understanding of the model's capabilities and potential areas for improvement.

**Visualizing Training History:**

A plot of training and validation accuracy and loss over epochs is generated to visualize the model's learning progress. This helps identify trends and potential issues, such as overfitting or underfitting.

**Snippets of the Code for the Data Analyses Process**



A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

# Interpretation of results:

**Training Accuracy:**

The training accuracy of 100% suggests that the model has effectively learned the patterns present in the training set. However, achieving such high accuracy may indicate potential overfitting, where the model memorizes the training data instead of learning generalized features.

**Test Accuracy:**

The test accuracy of 100% is impressive and suggests that the model generalizes well to previously unseen data. While this might be indicative of a robust model, it's crucial to scrutinize potential issues, such as data leakage or an overly simplified dataset.

**Test Loss:**

The test loss approaching zero is a positive indicator, signifying that the model makes accurate predictions. However, it's essential to consider whether the model is overfitting, as extremely low loss values could be a result of memorizing the training set.

**Visualizing Training History:**

The plotted training and validation accuracy and loss curves over epochs are useful for assessing the model's learning dynamics. Sustained improvement in both training and validation accuracy indicates effective learning, while a large gap between the curves might suggest overfitting.

**Consideration for Overfitting:**

The perfect accuracy achieved on the training set raises concerns about potential overfitting. Strategies such as dropout layers and early stopping might be considered in the future to address overfitting issues and enhance the model's generalization to new data.

A screenshot of a computer

Description automatically generated

**Future work Improvements and considerations**

While achieving a 100% accuracy is remarkable, I need to consider the real-world implications of my model. Further analysis could involve:

Error Analysis: Investigating misclassified images to understand my model's limitations and potential areas for improvement.

Exploration of Alternative Architectures: Experimenting with different CNN architectures or incorporating transfer learning with pre-trained models to enhance performance.

Additional Evaluation Metrics: Introducing metrics beyond accuracy, such as precision, recall, and F1-score, for a more nuanced assessment of my model's performance.

**Google Drive link for Full Code and Dataset**

# References

Gavrilova, Y., 2020. *Classification Algorithms: A Tomato-Inspired Overview.* [Online]   
Available at: https://serokell.io/blog/classification-algorithms  
[Accessed 18 November 2023].

SuperAnnotate, 2023. *What is image classification? Basics you need to know.* [Online]   
Available at: https://www.superannotate.com/blog/image-classification-basics  
[Accessed 18 November 2023].