

DESCRIPTION OF WORK

for

BLG 506E

COMPUTER VISION

COURSE PROJECT

**Generating FEN Descriptions of Chess Boards by
Using Two Important Pre-trained CNN
Architectures**

CANSU YANIK

30.04.2021

Table of Contents

1	EXECUTIVE SUMMARY	3
2	INTRODUCTION.....	3
3	PROJECT DESCRIPTION	4
3.1	Goals of Project.....	8
3.2	Impact of Solution	8
3.3	SOTA.....	8
3.3.1	Novel contributions	10
3.4	Risk Assessment.....	11
4	PROJECT SCOPE	12
4.1	Work Breakdown Structure (WBS)	12
4.2	Work Packages.....	12
4.3	Out of Scope	14
5	ASSUMPTIONS.....	16
6	MILESTONES and DELIVERABLES	16
6.1	Deliverables and Milestone Tables	16
6.2	Project Schedule (Gantt Chart).....	17
7	References.....	17

1 EXECUTIVE SUMMARY

In the article titled "Going deeper with convolutions" published in 2014, it was emphasized that the GoogleNet model is much better than AlexNet [1]. In this project, it is aimed to show with a real dataset that GoogleNet is a better model than AlexNet. The pre-trained models which are AlexNet and GoogleNet will be used and finetuning methods will be applied to these models. The models would be expected to generate the FEN description of an instant board taken from a chess game. It is also aimed to show how this situation changes with the using different hyper parameters (e.g. learning rate, data augmentation, regularization etc.). Algorithms will be analyzed and compared in terms of the performance metrics, speed and confusion. The strengths and weaknesses of these two pre-trained architectures will be discussed. In addition, if there will be enough time, it is planned to add a 1x1 bottleneck layer to the AlexNet architecture in order to reduce the number of parameters. Thus, how its performance changes will be examined.

2 INTRODUCTION

Thanks to the increasing interest in artificial intelligence (AI), machine learning emerged and then gained huge popularity. It has been seen that the machines have to learn from data after a certain stage, because it has become a very important concept for all developments in the technological fields. Thus, computer scientists/researchers have developed machine learning algorithms that can learn from data, make decisions and predict by using given data. Computer vision is one of the most commonly working areas of the machine learning. Computer vision is basically trying to make the machine do what the human brain can do with regard to vision. Its main task is to recognize objects (to be able to recognize and group objects). The aim is to make sense of the content of digital images. In other words, it is the ability to extract an object, text or a model from the image by the machine. Computer vision uses the methods of creating, processing, analyzing and making sense of the digital image in order to produce information numerically or symbolically on the image.

Computer Vision is currently used in many different areas. For example, in the medical field, diagnosing diseases from images, object detection, classification, autonomous vehicles, object tracking, face recognition, gaming and controls and so on. In short, researches have been done and models have been created to do things that are doing using human intelligence, as well, by machines. For example, games played by people using their intelligence are also tried to be played by machines using computer vision and artificial intelligence approaches. The strongest examples of this opinion will be the game of Go [2] and Chess [3] played by artificial intelligence. A successful artificial intelligence model named AlphaZero [4] was created in 2017 for chess, which is a very popular intelligence game these days. Thus, artificial intelligence and computer vision techniques were used for this game, which was played for 1420 years.

With the increasing interest in the field of computer vision, the work done in this field has increased. People have aimed to build models with better performance. The ImageNet [5] project has encouraged such studies. The ImageNet project is a large visual database designed

for use in visual object recognition software research. Between 2010 and 2017, competitions (ILSVRC) were organized in the fields of object detection and classification and it was aimed to find the best algorithms. The winner of 2012, the AlexNet [6] model was an important architecture that raised interest in CNN architectures at that time. But the number of parameters and processing cost was high. GoogleNet, the winner of 2014, stated that they offered a deeper and more computationally low-cost algorithm compared to AlexNet [1].

Inspired by the increasing popularity of the game of chess, its extensive history, my interest in this game, and machine learning/artificial intelligence studies in the field of chess, I wanted to use a dataset and give a work in this field. In the paper "Going deeper with convolutions" published in 2014, it was emphasized that the GoogleNet model is much better than AlexNet [1]. Therefore, in this project, it is intended to show with a real dataset that GoogleNet is a better model than AlexNet. It is also aimed to show how this situation changes with different hyper parameters (e.g. learning rate, data augmentation, regularization etc.). Thus, two different pre-trained models will be used in this project. These are AlexNet and GoogleNet. I will measure the accuracies (train, validation), parameter sizes and the prediction times of the algorithms by using different hyper parameters. By analyzing all outputs and observations, I will comment about performances of algorithms and evaluate them in terms of the efficiency.

In this project, it is planned to apply finetuning, hyper parameter optimization, learning rate selection, regularization methods.

In the rest of the document, there is given more detailed project explanations with the goals of project, impact solutions and the methods used in the experiments. In the next section, scope of the project is included with the WBS, work packages and some information about project. After this section, assumptions are given. Finally, a reference list is found after the deliverables and project schedule section.

3 PROJECT DESCRIPTION

The dataset used in the project belongs to the Chess Positions dataset put by Pavel Koryakin on the Kaggle website [7]. The whole dataset includes 100000 images of a randomly generated chess positions of 5-15 pieces. For this project, it is planned to use less amount of images that will be randomly selected for train, validation and test datasets. The features of the dataset are as follows [7].

- All images are 400 by 400 pixels.
- Training set: 80000 images
- Test set: 20000 images
- Pieces were generated with the following probability distribution:
 - 30% for Pawn
 - 20% for Bishop
 - 20% for Knight
 - 20% for Rook
 - 0% for Queen

- 2 Kings are guaranteed to be on the board.
- Labels are in a filename in Forsyth–Edwards Notation format, but with dashes instead of slashes.

An example image format in the dataset is given in figure 1.

1B1K4-6k1-2b2N2-4R3-7B-2pr4-2R1P3-n4n1n.jpeg

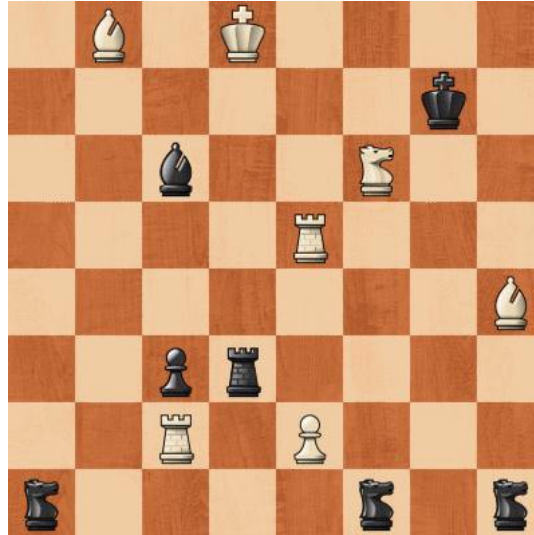


Figure 1. A sample of dataset and its label

In this project two pre-trained models which are AlexNet and GoogleNet will be used, and transfer learning will be used. The following methods will be applied to the models.

- Repurposing a pre-trained model
- Train some layers and leave the others frozen.

Since the lower layers extract general features (problem-independent) from the image and higher layers extract specific features (problem-dependent) from the image, I decided to freeze lower layers and train higher layers of the models. A frozen layer does not change during training. Usually, if there is a small dataset and a large number of parameters, more layers are left frozen to avoid overfitting. But since the dataset is large here and overfitting is not an issue, I can improve my models by training more layers. The same method will be used for both models. The first convolution and max pooling layers will be frozen and the other layers will be trained. Because, as I said, the first layers are used to extract general features. Also, the original classifiers will be removed, then a new classifier that fits this project purpose will be added.



Layer (type)	Output Shape	Param #	
Conv2d-1	[-1, 64, 55, 55]	23,296	 Freeze these layers
ReLU-2	[-1, 64, 55, 55]	0	
MaxPool2d-3	[-1, 64, 27, 27]	0	
Conv2d-4	[-1, 192, 27, 27]	307,392	
ReLU-5	[-1, 192, 27, 27]	0	
MaxPool2d-6	[-1, 192, 13, 13]	0	
Conv2d-7	[-1, 384, 13, 13]	663,936	
ReLU-8	[-1, 384, 13, 13]	0	
Conv2d-9	[-1, 256, 13, 13]	884,992	
ReLU-10	[-1, 256, 13, 13]	0	
Conv2d-11	[-1, 256, 13, 13]	590,080	
ReLU-12	[-1, 256, 13, 13]	0	
MaxPool2d-13	[-1, 256, 6, 6]	0	
AdaptiveAvgPool2d-14	[-1, 256, 6, 6]	0	 Train these layers
Dropout-15	[-1, 9216]	0	
Linear-16	[-1, 4096]	37,752,832	
ReLU-17	[-1, 4096]	0	
Dropout-18	[-1, 4096]	0	
Linear-19	[-1, 4096]	16,781,312	
ReLU-20	[-1, 4096]	0	
Linear-21	[-1, 1000]	4,097,000	
Total params: 61,100,840			
Trainable params: 61,100,840			
Non-trainable params: 0			
Input size (MB): 0.57			
Forward/backward pass size (MB): 8.38			
Params size (MB): 233.08			
Estimated Total Size (MB): 242.03			

Figure 2. Finetuning AlexNet Model

Layer (type)	Output Shape	Param #	
Conv2d-1	[-1, 64, 112, 112]	9,408	
BatchNorm2d-2	[-1, 64, 112, 112]	128	
BasicConv2d-3	[-1, 64, 112, 112]	0	
MaxPool2d-4	[-1, 64, 56, 56]	0	
Conv2d-5	[-1, 64, 56, 56]	4,096	
BatchNorm2d-6	[-1, 64, 56, 56]	128	
BasicConv2d-7	[-1, 64, 56, 56]	0	
Conv2d-8	[-1, 192, 56, 56]	110,592	
BatchNorm2d-9	[-1, 192, 56, 56]	384	
BasicConv2d-10	[-1, 192, 56, 56]	0	
MaxPool2d-11	[-1, 192, 28, 28]	0	
Conv2d-12	[-1, 64, 28, 28]	12,288	
BatchNorm2d-13	[-1, 64, 28, 28]	128	
BasicConv2d-14	[-1, 64, 28, 28]	0	
Conv2d-15	[-1, 96, 28, 28]	18,432	
BatchNorm2d-16	[-1, 96, 28, 28]	192	
BasicConv2d-17	[-1, 96, 28, 28]	0	
Conv2d-18	[-1, 128, 28, 28]	110,592	
BatchNorm2d-19	[-1, 128, 28, 28]	256	
BasicConv2d-20	[-1, 128, 28, 28]	0	
Conv2d-21	[-1, 16, 28, 28]	3,072	
BatchNorm2d-22	[-1, 16, 28, 28]	32	
BasicConv2d-23	[-1, 16, 28, 28]	0	
Conv2d-24	[-1, 32, 28, 28]	4,608	
BatchNorm2d-25	[-1, 32, 28, 28]	64	
BasicConv2d-26	[-1, 32, 28, 28]	0	
MaxPool2d-27	[-1, 192, 28, 28]	0	
Conv2d-28	[-1, 32, 28, 28]	6,144	
BatchNorm2d-29	[-1, 32, 28, 28]	64	
BasicConv2d-30	[-1, 32, 28, 28]	0	
Inception-31	[-1, 256, 28, 28]	0	
Conv2d-32	[-1, 128, 28, 28]	32,768	
BatchNorm2d-33	[-1, 128, 28, 28]	256	
BasicConv2d-34	[-1, 128, 28, 28]	0	
Conv2d-35	[-1, 128, 28, 28]	32,768	
BatchNorm2d-36	[-1, 128, 28, 28]	256	
BasicConv2d-37	[-1, 128, 28, 28]	0	
Conv2d-38	[-1, 192, 28, 28]	221,184	
BatchNorm2d-39	[-1, 192, 28, 28]	384	
BasicConv2d-40	[-1, 192, 28, 28]	0	
... More Layers			
Inception-173	[-1, 832, 7, 7]	0	
Conv2d-174	[-1, 384, 7, 7]	319,488	
BatchNorm2d-175	[-1, 384, 7, 7]	768	
BasicConv2d-176	[-1, 384, 7, 7]	0	
Conv2d-177	[-1, 192, 7, 7]	159,744	
BatchNorm2d-178	[-1, 192, 7, 7]	384	
BasicConv2d-179	[-1, 192, 7, 7]	0	
Conv2d-180	[-1, 384, 7, 7]	663,552	
BatchNorm2d-181	[-1, 384, 7, 7]	768	
BasicConv2d-182	[-1, 384, 7, 7]	0	
Conv2d-183	[-1, 48, 7, 7]	39,936	
BatchNorm2d-184	[-1, 48, 7, 7]	96	
BasicConv2d-185	[-1, 48, 7, 7]	0	
Conv2d-186	[-1, 128, 7, 7]	55,296	
BatchNorm2d-187	[-1, 128, 7, 7]	256	
BasicConv2d-188	[-1, 128, 7, 7]	0	
MaxPool2d-189	[-1, 832, 7, 7]	0	
Conv2d-190	[-1, 128, 7, 7]	106,496	
BatchNorm2d-191	[-1, 128, 7, 7]	256	
BasicConv2d-192	[-1, 128, 7, 7]	0	
Inception-193	[-1, 1024, 7, 7]	0	
AdaptiveAvgPool2d-194	[-1, 1024, 1, 1]	0	
Dropout-195	[-1, 1024]	0	
Linear-196	[-1, 1000]	1,025,000	
=====			
Total params: 6,624,904			
Trainable params: 6,624,904			
Non-trainable params: 0			

Input size (MB): 0.57			
Forward/backward pass size (MB): 94.11			
Params size (MB): 25.27			
Estimated Total Size (MB): 119.95			

Figure 3. Finetuning GoogleNet Model

Pre-processing the input will be planned as follows:

1. Since the board and the pieces may have different colors, images will be converted to gray-scale version.
2. Gray-scale images will be converted again to 3 channel images. Because models accept 3 channel images.
3. The inputs will be resized to 224x224. Because models accept this dimension.
4. Input will be converted to tensor format in order to use torch
5. Normalize the input using the mean and standard deviation. To normalize the input image dataset, the mean and standard deviation of the pixels data is used as per the standard values suggested by the PyTorch.

3.1 Goals of Project

This project will have 3 purposes.

- One goal is to build two different models able to learn how to generate FEN labels. FEN stands for Forsyth–Edwards Notation which is explained in out of scope section.
- Second goal is to show that as it is said in “Going Deeper with Convolutions” paper, GoogleNet has a better performance than AlexNet by using a real dataset.
- It is also aimed to show how this situation changes with different hyper parameters (eg learning rate, data augmentation, regularization etc.).

3.2 Impact of Solution

In this project, it will be showed whether GoogleNet will perform better than AlexNet even with different hyper parameter selections, data augmentation applying or different regularization methods. In addition, if there is enough time, methods to improve the models can be applied. For example: reducing the sizes of filters used or applying a 1x1 bottleneck convolution for dimension reduction to AlexNet model.

3.3 SOTA

AlexNet [6] is an important study in 2012 that made convolutional neural network models and deep learning popular again. At that time, they presented a state of art architecture that gave very good results according to traditional machine learning and computer vision algorithms. AlexNet is an 8-layer structure (5: Conv - 3 FC). It has more filter and convolution layers per layer [6]. It consists of 11x11, 5x5, 3x3, convolutions, max-pooling, droupout, data augmentation, ReLU activation, SGD momentum. ReLU activations are added after each convolution fully connected layer.

It is a breaking point in the image classification problem by providing a sudden increase in classification. The network achieved a top-5 error of 15.3% compared to the second place top-5 error rate of 26.2% in ImageNet ILSVRC competition. The architecture of AlexNet is given in figure 4.

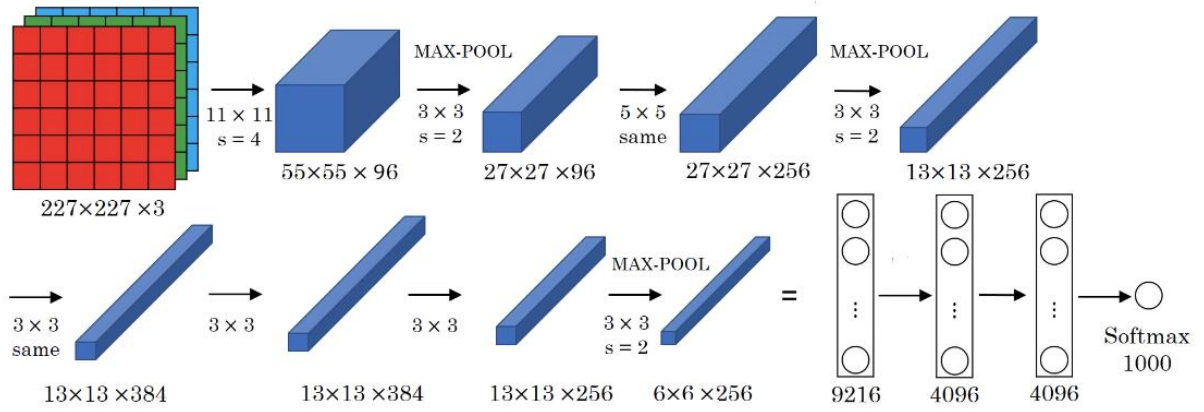


Figure 4. AlexNet Architecture

The winner of the ILSVRC 2014 contest was GoogLeNet (Inception V1) from Google [1]. the first 5 error rate is reached to 6.67%. They presented both a deep and wide architecture. The architecture consisted of inception modules. The Inception Module is shown in the figure 5. 1x1, 3x3, 5x5 convolutions and max pooling are used in the module. They used 1x1 filters called bottleneck to reduce the dimension and so that the number of parameters. Thus, the computational cost was lower. The architecture consists of 22 layers (27 layers with pooling layers). The first layers are simple convolution layers. Then it consists of 9 inception modules. Last layers are dropout and linear (softmax) classification layers.

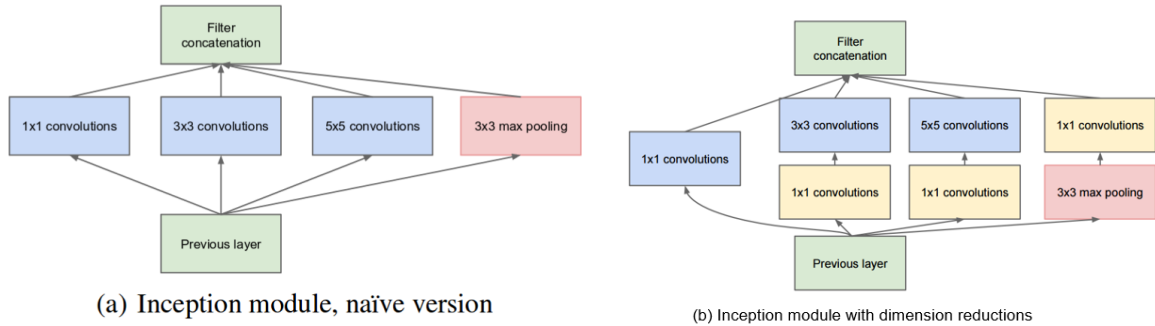


Figure 5. Inception Modules [1]

These two important architectures in CNN algorithms have been used by those concerned in computer vision areas. Since these architectures are trained with high parameter numbers and by using strong hardware powers, they offer us ready and pre-trained models. Thus, we can use these ready-made models to solve different problems. This is called transfer learning. A comprehensive review on transfer learning is provided by Pan & Yang (2010) [8]. This article shows how to implement a transfer learning solution for image classification problems. Transfer learning is a popular method of computer vision because it saves us time and allows us to create accurate models (Rawat & Wang 2017) [9]. While solving a different problem, we take advantage of previous learning and do not have to start from scratch. Thus, we use pre-trained models as transfer learning in computer vision areas. That's why AlexNet

and GoogleNet are used as pre-trained models in this project. Since the first layers extract the general features, the low layers in the models will be frozen. Yosinski et al. (2014) stated this [10]: “if first-layer features are general and last-layer features are specific, then there must be a transition from general to specific somewhere in the network”.

In the paper titled as “Evaluation of Pre-Trained Convolutional Neural Network Models for Object Recognition”, AlexNet and GoogleNet pre-trained models were used for image detection problem. According to this paper, even though AlexNet and GoogLeNet yield similar accuracy which is 99.65%, GoogleNet achieved better results [11]. In my project, I will plan to use these models for image classification problem. On the other hand, according to the Rahmathunneesa and Muneer’s work, AlexNet performed faster and gave better results in terms of performance metrics compared to the GoogleNet for the categorization of glioma grades (brain tumors) [12]. Also, they said that, GoogLeNet performs intermediate performance in terms of all metrics. In another work which is “Transfer learning with pre-trained deep convolutional neural networks for serous cell classification”, GoogleNet gave better performance compared to AlexNet in classification problem [13]. In addition, to improve the classification performance, the number of the training samples was increased by using data augmentation techniques.

3.3.1 Novel contributions

In this project, as contributions to the existing solutions available, the following differences will be stated. In the project, two important CNN architectures, AlexNet and GoogleNet, will be used in the classification problem and these architectures will be evaluated in terms of performance, speed and confusion. The strengths and weaknesses of these two pre-trained architecture will be discussed. Finetuning methods will be applied to the models. Architectures will not be trained from the scratch, the first layers will be frozen, the higher layers that extract more specific features will be trained. In addition, the models will be retrained using different hyper parameters (learning rate, regularization etc.) and the results will be compared. Thus, it will be observed whether there will be an improvement in the performance of the models if different parameters are used. Finally, if there will be enough time, it is planned to add a 1x1 bottleneck layer to the AlexNet architecture in order to reduce the number of parameters. Thus, how its performance changes will be examined.

3.4 Risk Assessment

Possible Risk	Risk Reason	Contingency Plans
Different resolutions on dataset	Dataset	Adjust the brightness/contrast of images, convert to grayscale
Limited data	Dataset	Apply different fine tuning approach
Incorrect data	Dataset	Check data, filter correct ones
Low model accuracy/high error rate	Training strategy or model building errors	Check the model, apply hyper parameter optimization, use different learning rate selection
Some samples may have poor quality	Dataset	Reduce resolution for each image
Training takes very long time	Model/technical	Train on GPU, change GPU model, train parallel on GPU, freeze more layers, change hyper parameters
Random Noise	Dataset	Add distributions of Gaussian, uniform and salt-and pepper noise.
Vanishing gradient descent	Model	Reduce layer size, find more data, use RNN approaches, use RELU
Overfitting	Model	Use regularization, dropout, separate train, validation and test datasets

Table 1. Risk Assessment of the Project

4 PROJECT SCOPE

This SOW shall apply to the tasks, services and terms detailed below:

4.1 Work Breakdown Structure (WBS)

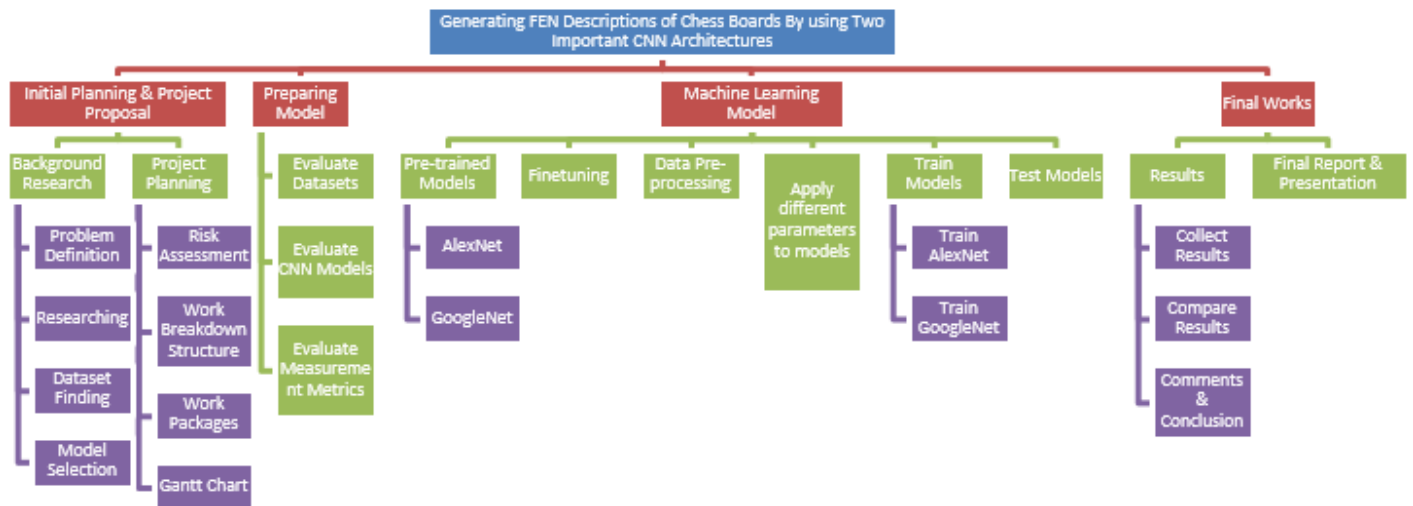


Figure 6. Example Work Breakdown Structure (WBS)

4.2 Work Packages

WP 1	Initial Planning & Project Proposal		
		30.04.2021	M1
Objectives: This work package will cover determining the problem and the project subject, doing research on the subject, determining the project schedule and the project management.			
Tasks <ul style="list-style-type: none"> <input type="checkbox"/> Background Research <input type="checkbox"/> Project Planning 			
Deliverables and Milestones: <p>D1: Project proposal and presentation</p>			

WP 2	Preparing Model		
	M1	10.05.2021	M4
Objectives: This work package will cover preliminary preparations to create models for the problem. In this section, all technical features of the models to be used and the metrics to compare the results will be decided.			
Tasks <ul style="list-style-type: none"> <input type="checkbox"/> <i>Evaluate Datasets</i> <input type="checkbox"/> <i>Evaluate CNN Models</i> <input type="checkbox"/> <i>Evaluate Measurement Metrics</i> 			
Deliverables and Milestones: <p>MS2.1: All technical features of the models decision</p> <p>MS2.2: Measurement metrics decision</p> <p>D2: Decision of the project process</p>			

WP 3	Machine Learning Model		
	M4	05.06.2021	M7
Objectives: This work package will cover creating models, pre-processing dataset, training models and testing models.			
Tasks <ul style="list-style-type: none"> <input type="checkbox"/> <i>Pre-trained Models</i> <input type="checkbox"/> <i>Finetuning</i> <input type="checkbox"/> <i>Data Pre-processing</i> <input type="checkbox"/> <i>Applying different hyper parameters</i> <input type="checkbox"/> <i>Model training</i> <input type="checkbox"/> <i>Model testing</i> 			

Deliverables and Milestones:

MS3.1: Creation of the models

MS3.2: Model training

MS3.3: Model testing

D3: Progress presentation

MS4: Creation of the models with different hyper parameters

D4: Models

WP 4	Final Works		
	M7	18.06.2021	M8
Objectives: This work package will cover collecting, evaluating, comparing the results, preparing a final report with all the information obtained.			
Tasks <input type="checkbox"/> <i>Results</i> <input type="checkbox"/> <i>Final Report and Presentation</i>			
Deliverables and Milestones: MS5.1: Collection and comparing results MS5.2: Comment and conclusion about the work D5.1: Results D5.2: Final Report and final presentation			

4.3 Out of Scope

The following are considered OUT OF SCOPE for this contract:

Forsyth–Edwards Notation (FEN) [14]

Forsyth - Edwards Notation (FEN) is a standard notation that describes a particular board position of a chess game with a text line using only the ASCII characters. The goal of FEN is to obtain all the necessary information to restart a game from a specific location. FEN was discovered by journalist David Forsyth and developed by Steven J. Edwards with the use of computers.

A FEN record contains six fields. The separator between fields is a space. The fields are:

1. Piece placement (from White's perspective). Each rank is described, *starting with rank 8* and ending with rank 1; within each rank, the contents of each square are described from file "a" through file "h". Following the Standard Algebraic Notation (SAN), each piece is identified by a single letter taken from the standard English names (pawn = "P", knight = "N", bishop = "B", rook = "R", queen = "Q" and king = "K"). White pieces are designated using upper-case letters ("PNBRQK") while black pieces use lowercase ("pnbrqk"). Empty squares are noted using digits 1 through 8 (the number of empty squares), and "/" separates ranks.
2. Active color. "w" means White moves next; "b" means Black moves next.
3. Castling availability. If neither side can castle, this is "-". Otherwise, this has one or more letters: "K" (White can castle kingside), "Q" (White can castle queenside), "k" (Black can castle kingside), and/or "q" (Black can castle queenside). A move that temporarily prevents castling does not negate this notation.
4. En passant target square in algebraic notation. If there's no en passant target square, this is "-". If a pawn has just made a two-square move, this is the position "behind" the pawn. This is recorded regardless of whether there is a pawn in position to make an en passant capture.^[6]
5. Halfmove clock: This is the number of halfmoves since the last capture or pawn advance. The reason for this field is that the value is used in the fifty-move rule.
6. Fullmove number: The number of the full move. It starts at 1, and is incremented after Black's move.

Figure 7. Explanations of the FEN Fields (Directly taken from [14])

An example FEN output of a chess board is shown in figure 8.

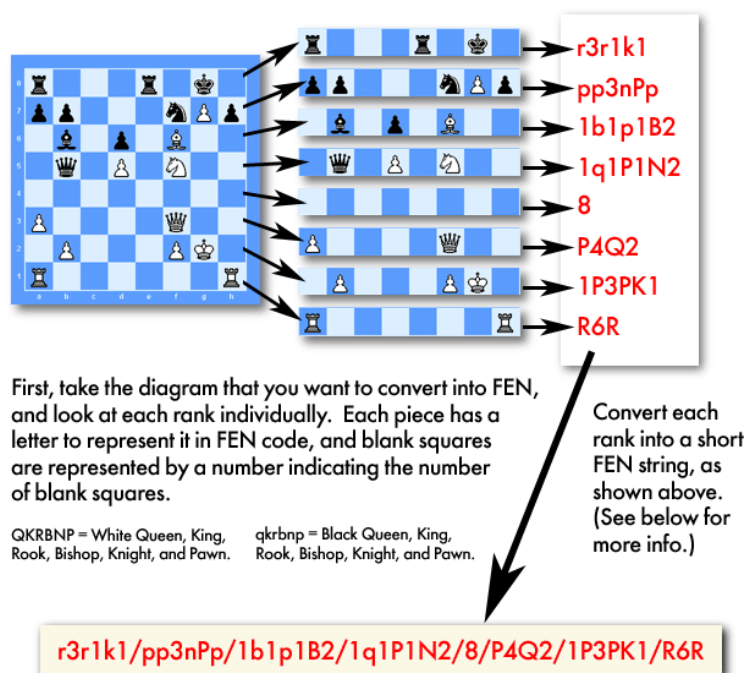


Figure 8. An example FEN output of a chess board [15]

In this project, the 4th, 5th and 6th fields were not used in the FEN labels used. In addition, labels are in a filename in Forsyth–Edwards Notation format, but with dashes instead of slashes.

5 ASSUMPTIONS

Project assumptions typically revolve around constraints such as time, hardware, and scope. The assumptions of this project can be listed as follows.

1. All the resources needed to complete the project, both information and material, will be accessible.
2. While working on the project, I will have the resources needed to complete tasks on time, from the necessary equipment, software, electricity throughout the project life.
3. All equipment and hardware will remain operational throughout the project cycle.
4. The overall scope of the project will not change throughout project life cycle. However, tasks may undergo minor changes during detailing.
5. Training of the models will not exceed the duration of the project.
6. The dataset will be in the quality, amount and format suitable for the project. The images used will be labeled in the appropriate format.

6 MILESTONES and DELIVERABLES

6.1 Deliverables and Milestone Tables

Deliverable (D)	Description	Date	Milestone (MS)
D1	Project Proposal and Presentation	30.04.2021	MS1
D2	Decision of the Project Process	10.05.2021	MS2
D3	Progress presentation	28.05.2021	MS3
D4	Models	05.06.2021	MS4
D5.1	Results	18.06.2021	MS5
D5.2	Final Report and final presentation	18.06.2021	MS5

Table 1. Deliverable Table

6.2 Project Schedule (Gantt Chart)

Weeks	M1	M2	M3	M4	M5	M6	M7	M8
Starting	25.04.2021	02.05.2021	09.05.2021	16.05.2021	23.05.2021	30.05.2021	06.06.2021	13.06.2021
WP 1	D1							
WP 2		MS2.1 MS2.2	MS2.2 MS2.2	D2				
WP 3				MS3.1 MS3.2 MS3.3	D3	MS4	D4	
WP 4					MS5.1	MS5.2	D5.1	D5.2

Figure 9. Gantt Chart

7 References

- [1] C. Szegedy et al., "Going Deeper with Convolutions", arXiv.org, 2014. [Online]. Available: <https://arxiv.org/abs/1409.4842>.
- [2] "A Brief History of Go | American Go Association", Usgo.org. [Online]. Available: <https://www.usgo.org/brief-history-go>.
- [3] "International Chess Federation", Fide.com. [Online]. Available: <https://www.fide.com/>.
- [4] D. Silver et al., "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm", arXiv.org, 2017. [Online]. Available: <https://arxiv.org/abs/1712.01815>.
- [5] [Online]. Available: <http://www.image-net.org/> <http://www.image-net.org/challenges/LSVRC/>.
- [6] Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.
- [7] "Chess Positions", Kaggle.com. [Online]. Available: <https://www.kaggle.com/koryakinp/chess-positions>. [Accessed: 27- Apr- 2021].
- [8] Pan, S.J. and Yang, Q., 2010. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), pp.1345–1359.
- [9] Rawat, W. and Wang, Z., 2017. Deep convolutional neural networks for image classification: A comprehensive review. Neural computation, 29(9), pp.2352–2449.
- [10] Yosinski, J., Clune, J., Bengio, Y. and Lipson, H., 2014. How transferable are features in deep neural networks?. In Advances in neural information processing systems (pp. 3320–3328).

- [11] Zabir, M. & Fazira, N. & Ibrahim, Zaidah & Sabri, Nurbaity. (2018). Evaluation of Pre-Trained Convolutional Neural Network Models for Object Recognition. International Journal of Engineering and Technology(UAE). 7. 95-98. 10.14419/ijet.v7i3.15.17509.
- [12] A. P. Rahmathunneesa and K. V. Ahammed Muneer, "Performance Analysis of Pre-trained Deep Learning Networks for Brain Tumor Categorization," 2019 9th International Conference on Advances in Computing and Communication (ICACC), 2019, pp. 253-257, doi: 10.1109/ICACC48162.2019.8986151.
- [13] Baykal, E., Dogan, H., Ercin, M.E. et al. Transfer learning with pre-trained deep convolutional neural networks for serous cell classification. Multimed Tools Appl 79, 15593–15611 (2020). <https://doi.org/10.1007/s11042-019-07821-9>.
- [14] "Forsyth–Edwards Notation - Wikipedia", En.wikipedia.org. [Online]. Available: https://en.wikipedia.org/wiki/Forsyth%E2%80%93Edwards_Notation#cite_note-pgn_spec-1.
- [15] "Forsyth-Edwards Notation F.A.Q.", Chessgames.com. [Online]. Available: <https://www.chessgames.com/fenhelp.html>.