

**AN EMOTION RECOGNITION SYSTEM
INTEGRATED WITH HUMAN
INTELLIGENCE**

TAN SWEE YANG

UNIVERSITI TUNKU ABDUL RAHMAN

**AN EMOTION RECOGNITION SYSTEM INTEGRATED WITH
HUMAN INTELLIGENCE**

TAN SWEE YANG

**A project report submitted in partial fulfilment of the
requirements for the award of Bachelor of Science Software
Engineering with Honours**

**Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

May 2023

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature :



Name : Tan Swee Yang


ID No. : 1904180

Date : 6/10/2023

APPROVAL FOR SUBMISSION

I certify that this project report entitled **AN EMOTION RECOGNITION
SYSTEM INTEGRATED WITH HUMAN INTELLIGENCE** was
prepared by **TAN SWEE YANG** has met the required standard for submission
in partial fulfilment of the requirements for the award of Bachelor of Science
Software Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	
Supervisor	:	<u>Prof. Ts. Dr. Yau Kok Lim</u>
Date	:	<u>6 October 2023</u>

Signature	:	<hr/>
Co-Supervisor	:	<hr/>
Date	:	<hr/>

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2023, Tan Swee Yang. All right reserved.

ABSTRACT

Reinforcement learning (RL) is a powerful tool for solving real-world problems, but achieving high accuracy often relies on complex reward functions. Incorporating human intelligence into RL rewards enhances reliability and safety. This research focuses on improving the Two-State Q-Learning (TS-QL) approach. TS-QL uses QL to provide a second perspective to the Convolutional Neural Network (CNN) models on misclassified images by performing simple image transformation actions such as rotations. TS-QL has shown promise in enhancing the testing accuracy of a facial emotion recognition (FER) system. This research addresses four problems of TS-QL: a) the use of a single QL to learn unrelated optimal policies; b) non-optimal QL hyperparameters; c) inaccurate reward function; d) randomness in action selection. To address these problems, this research introduces Two-State Q-Learning with Human Feedback (TS-QL-HF), which aims to achieve these research objectives: a) to design specialized QL agents, each dedicated to learning distinct and unrelated optimal policies; b) to integrate, program, investigate the reinforcement learning approach with human inputs using a suitable set of hyperparameters to improve the accuracy of the FER system and the agent learning convergence rate, and provide conclusions using simulations; c) to propose, program, investigate an efficient and intelligent action selection strategy suitable for the deterministic environment in the reinforcement learning approach and provide conclusions using simulations; d) to program and investigate the effects of DQL, a reinforcement approach from previous studies in the fully deterministic environment. The improved FER system is expected to have the following deliverables: a) the multi-objective of QL is changed to a single objective; b) optimal QL hyperparameters are utilized in the QL of the FER system; c) human feedback is incorporated into the reward function of TS-QL in the FER system as TS-QL-HF; d) an intelligent and efficient action selection strategy is implemented in TS-QL-HF; e) human participation is minimized in the learning process to reduce potential human error. To evaluate the proposed approach, three experiments are conducted: a) CNN models performance comparison; b)

action selection strategies performance comparison; c) QL and Double QL Performance Comparison. The experiments are run using the PyCharm IDE, and the specific settings of research scenarios, hyperparameters and CNN network architecture are showed in the results and discussions chapter. The findings of this research have practical implications in fields like healthcare, education, robotics, and marketing in dynamic environments with challenging reward design.

TABLE OF CONTENTS

Table of Contents

LIST OF TABLES	10
LIST OF FIGURES	11
LIST OF SYMBOLS / ABBREVIATIONS	13
CHAPTER 1 14	
INTRODUCTION	14
1.1 Background of the Problem	16
1.2 Problem Statement	17
1.3 Importance of the Study	18
1.4 Project Objectives	19
1.5 Proposed Approach	19
1.6 Hypotheses	21
1.7 Project Scope	21
CHAPTER 2 22	
LITERATURE REVIEW	22
2.1 Introduction	22
2.2 Review of CNN models	23
2.3 Review of Action Selection Strategies	24
2.4 Review of Existing QL Extensions	25
2.5 Performance of Existing Approaches	27
2.6 Convergence Rate of Existing DQN Approaches	27
2.7 Summary	32
CHAPTER 3 33	
METHODOLOGY AND WORK PLAN	33
3.1 Introduction	33
3.2 Planning	33
3.3 Literature Review	33
3.4 Implementation	34
3.5 Interpretation and Report	34
3.6 Flowchart	35

3.7	Image Loading and Preprocessing	36
3.8	CNN Training	39
3.9	Q-learning	43
3.10	Action Selection Strategies	45
3.11	Environment Reward Function	48
3.12	QL Update Function	48
3.13	DQL Update Function	48
3.14	Q-table Mean Function	49
3.15	Iterative Q-Learning	49
3.16	WBS	58
3.17	Gantt Chart	60
3.18	Summary	62
CHAPTER 4	63	
RESULTS AND DISCUSSION	63	
4.1	Introduction	63
4.2	CNN Models Performance Comparison	68
4.3	TS-QL and TS-QL-HF Action Selection Strategies Performance Comparison	70
4.4	QL and Double QL Performance Comparison	73
4.5	Summary	75
4.6	Recommendations For Future Works	75
REFERENCES	77	
APPENDICES	80	

LIST OF TABLES

Table 2.2	CNN models used in the FER system from the previous work.	23
Table 2.3	Existing action selection strategies.	25
Table 2.4	Existing Solutions Comparisons	26
Table 2.5	Mean and median scores across all 57 Atari games, measured in percentages of human performance.	27
Table 2.6	Existing Approaches Convergence Rate Comparison	28
Table 3.8.4	Details of layers used in the CNN models.	43
Table 3.10	The characteristics, advantages, and disadvantages of the action selection strategies used in this FER system.	45
Table 3.15	The characteristics, advantages, and disadvantages of the approaches used in this FER system.	49
Table 4.1.1	Hyperparameters used for training InceptionV3, ResNet50, and MobileNetV2 CNN models.	64
Table 4.1.2	Hyperparameters used in TS-QL, TS-DQL, TS-QL-HF, and TS-DQL-HF models.	65
Table 4.1.6	Table of comparison on the one-shot RBED, harmonic RBED, and random action selection strategies.	67
Table 4.2	Averaged score of the models. The F1 scores used are macro as the classes are imbalanced.	69
Table 4.3	Average accuracy and F1 score of one-shot RBED, harmonic RBED, and random selection used in TS-QL and TS-QL-HF compared to the baseline. The baseline is the accuracy and F1 score of InceptionV3 model before using QL.	71
Table 4.4	Average accuracy and F1 score of TS- compared to the baseline. The baseline is the accuracy and F1 score of InceptionV3 model before using QL.	74

LIST OF FIGURES

Figure 2.6.1	Learning Curves of DQN and DDQN on Linear and Polynomial Learning Rates.	29
Figure 2.6.2	Detailed learning curves for rank-based (red) and proportional (blue) prioritization, as compared to the uniform Double DQN baseline (black) on a selection of games.	30
Figure 2.6.3	Learning Curves for Squared Error DA (red) and SA (green) on game environment "The Corridor".	31
Figure 3.6	High-Level Overview of the FER System.	35
Figure 3.7	Image Loading and Preprocessing Flowchart.	37
Figure 3.8.1	InceptionV3 Custom Classifier Architecture.	40
Figure 3.8.2	ResNet50 Custom Classifier Architecture.	41
Figure 3.8.3	MobileNetV2 Custom Classifier Architecture.	42
Figure 3.17	Gantt Chart on the Project.	60
Figure 4.1.4	Sample images from the database.	66
Figure 4.2	Performance comparison of InceptionV3, ResNet50, and MobileNetV2.	68
Figure 4.3	Performance comparison of one-shot RBED (red), harmonic RBED (yellow), and random selection strategies (blue) used in TS-QL and TS-QL-HF on the 'best' misclassified image.	70
Figure 4.4	Performance comparison of Traditional QL (red) and DQL (blue) in TS-QL and TS-QL-HF on the 'best' misclassified image.	73
Figure A.2	Action Stats of the Previous QL Approach using InceptionV3, ResNet50, and MobileNetV2.	80
Figure A.3	Confusion matrixes of InceptionV3, ResNet50, and MobileNetV2. X-axis represents the actual emotion	83

labels, and Y-axis represents the predicted emotion labels.

Figure A.4	Report on InceptionV3, ResNet50 and MobileNetV2 on recall, precision, F1 score and accuracy.	84
Figure A.5	The model accuracy and model loss of InceptionV3, ResNet50, and MobileNetV2.	85
Figure B.1	TS-DQL-HF Flowchart.	87

LIST OF SYMBOLS / ABBREVIATIONS

ε	Epsilon value
$Q(s, a)$	State-action pair value of QL (Q-value)
α	Learning rate of QL
γ	Discount factor of QL
r_t	Reward in time t
a_t	Action in time t
$\forall_s \in S$	For all state in state space
$\forall_a \in A_s$	For all action in action state space
s_t	State in time t
Q^A	Q-table A
Q^B	Q-table B
\bar{Q}	Mean of Q-table A and B
$\overline{Q(s, a)}$	State-action pair value of \bar{Q} (Mean Q-value)
a^*	Best current action in Q-table A
b^*	Best current action in Q-table B

CHAPTER 1

INTRODUCTION

Facial emotion recognition (FER) refers to a computer's ability to understand and differentiate human facial emotions through artificial intelligence (AI) technologies. This capability is critical in human interaction and communication and has a significant impact on various fields, such as healthcare, marketing, public safety, and education (Andalibi & Buss, 2020).

Artificial Neural Network (ANN), one of the most powerful and popular tools, is a computational processing system that draws significant inspiration from the way the biological nervous system in the human brain operates. ANN is a network that mainly consists of a high number of interconnected computational nodes, commonly referred to as neurons. These neurons collaborate to learn from input data and improve the network's final output. One of the largest limitations of ANN is the high computational complexity requirement to computer image data, preventing it to be applicable to image classification tasks such as FER. Convolutional neural network (CNN) is a form of ANN primarily used for solving complex image-driven pattern recognition due to its ability to reduce the parameters required to set up the model thus reducing the computational complexity requirement, making it highly suitable for image-focused tasks (O'Shea & Nash, 2015). CNN finds patterns within images by extracting various features, a set of characteristics from images and encoding them into the architecture. Throughout the years, researchers have introduced state-of-the-art pretrained CNN models such as InceptionV3, ReNet50, and MobileNetV2 where the learnt features can be transferred to another task using a technique called transfer learning, expediting the training process significantly (Bird & Faria, 2018).

Reinforcement learning (RL) is a machine learning approach where an agent learns an optimal policy of a task in Markovian domains by interacting with the environment in a trial-and-error manner. Q-learning (QL) is a model-free RL algorithm that learns the optimal policy from action outcomes using Q-values, which is the expected discounted reward for executing action in a state

while following a policy, without requiring the building of domain maps, the state-action transition probabilities. The environment of QL can be deterministic or stochastic. Deterministic environment refers to an environment where the outcome of every action taken by the agent is predictable and consistent throughout the episodes. Stochastic environment refers to an environment where the outcome of every action taken by the agent not predictable and inconsistent throughout the episodes (Sutton & Barto, 2005). The Q-values are represented in a lookup table called Q-table (Watkins & Dayan, 1992). When a QL agent learns the optimal policy, the Q-values converges and are shown as plateau in the learning curve (Van Hasselt, n.d.). Exploration is the process of taking arbitrary actions to gain more information about the environment while exploitation is the process of taking the optimal action. While learning more information about the environment is important, it is important to perform exploitation because it allows the agent to take advantage of what it has learnt and take the actions with high reward probability. Therefore, balancing exploration and exploitation is a crucial process in QL learning process (baeldung, 2023). QL has two important hyperparameters that can significantly impact the performance of the model. The first hyperparameter is the learning rate, α , which determines the rate where an agent updates its q-values based on the new information. A learning rate of 1 will make an agent fully accept the latest information without considering the previously learnt information while a learning rate of 0 will make an agent learn nothing as it does not accept any new information. The second hyperparameter is the discount factor, γ , which determines the weight of an agent in valuing immediate rewards compared to future rewards. A discount factor of 0 will make an agent short-sighted, as it does not consider future rewards at all while a discount factor of 1 will make an agent consider immediate and future reward equally (Sutton & Barto, 2005).

In recent years, RL has gained much attention in computer vision and researchers have proposed various approaches, such as object localization and adaptive image resolution to improve computer vision (Hafiz, n.d.). Recently, (Ilona, 2020) has utilized an approach which addresses the sampling efficiency problem in QL, and two-state QL (TS-QL) proposed by (Hafiz, n.d.) to enhance

the accuracy of a FER system by finding introducing randomness into the features of misclassified images. The approach is shown to provide a small but significant accuracy improvement on pretrained state-of-the-art CNN model called InceptionV3 in classifying facial emotions using only three actions, highlighting its sampling efficiency. This chapter discusses the background of the problem, problem statement, importance of study, project objectives, hypotheses, proposed approach, and project scope.

1.1 Background of the Problem

The goal of TS-QL is to find optimal policies for the image transformation actions on the images misclassified by a CNN model. However, it suffers from slow and unstable convergence. As the number of misclassified images increases in a real-world FER system where the environment is complex and stochastic, it makes this approach impractical to be applied. There are several major root causes that contribute to these issues.

The first major root cause is the usage of non-optimal learning rate and discount factor. It is found that the environment is fully deterministic, which makes the learning rate of 1 and discount factor of 0 optimal. TS-QL uses a learning rate of 0.4 and a discount factor of 0.3. This issue is found to unnecessarily slows down the convergence significantly. This is because in a fully deterministic environment, there is no ambiguity in the consequences of actions. Consequently, the agent can confidently update its q-value using the true reward received while ignoring long-term consequences.

The second major root cause is the inaccurate representation of the reward function. TS-QL uses the standard deviation of feature map as the reward function; however, it has been shown that it does not accurately represent the ground truth, or the true consequences of actions on the effectiveness of identifying correct facial emotions. Rather, it is an arbitrary evaluation that maximizes the standard deviation of the feature maps, which only promotes the diversity of the feature maps. This issue is found to limit the accuracy enhancement potential of the approach.

The third major root cause is the randomness of action selection. TS-QL uses an arbitrary action selection strategy, which involves randomly exploring

the entire action-state space to find the optimal policy. By exploring every part of the environment, the agent wastes time interacting with irrelevant parts of it (Tijmsma et al., 2017). This issue is found to significantly slow down the learning process of the agent towards the optimal policy for each image, therefore leading to a slow convergence, and a struggle in adapting to new policies.

The fourth major root cause is unnecessary multi-objective learning. TS-QL uses a single QL to learn unrelated optimal policies in each misclassified image. This causes the agent to struggle in learning new policies using unrelated knowledge and leading to the agent wasting time unlearning past knowledge before learning the new optimal policy. This issue is found to slow down convergence even further and causes a struggle in adapting towards new policies.

To address these issues, the learning rate and discount factor of TS-QL is changed to align with the fully deterministic environment. Next, an alternative reward function, which aligns with the ground truth, is defined to provide an accurate representation of the consequence of actions in classifying facial emotions. Furthermore, a more focused and intelligent action selection strategy is adopted in this project. The ideal strategy should intelligently balance exploration and exploitation based on its time spent exploring the environment (Maroti, 2019). Finally, separate QLs are used to learn the unrelated optimal policies. These improvements expedite the convergence rate, prevent unrelated knowledge interference, and increase the accuracy provided by the approach.

1.2 Problem Statement

The TS-QL is expected to have an efficient and robust convergence, while maximizing its accuracy in classifying facial emotions. However, the approach suffers from slow convergence, an inability to adapt to new policies, and lower accuracy. Furthermore, the approach has been shown to achieve lower accuracy. There are four root causes that contribute to these issues. The first root cause is the usage of non-optimal QL hyperparameters. TS-QL uses a set learning rate and discount factor that is non-optimal to a fully deterministic environment. This causes the agent to waste time calculating q-values that consider the past and future rewards when there is no ambiguity in the consequences of actions. Consequently, it slows down the convergence rate. The second root cause is the

reward function is inaccurate when the standard deviation of the feature map is used since it does not accurately represent the ground truth of action consequences. As a result, the accuracy of classifying facial emotions is reduced. The third root cause is the randomness in action selection. Currently, actions are selected randomly, which increases the time incurred for exploring the environment, including the irrelevant parts (Tijsma et al., 2017). Consequently, the convergence rate is lower, and there is a struggle in adapting to new policies. The fourth root cause is the use of a single QL to learn unrelated optimal policies. This causes a struggle in learning new policies using unrelated knowledge and this problem leads to the agent wasting time unlearning past knowledge before learning the new optimal policy. Consequently, the convergence rate is lowered, and there is a struggle in adapting to new policies. These explain the need for a set of redefined QL hyperparameters, a more focused and intelligent action selection strategy, and an alternative reward function that together learns a single objective.

1.3 Importance of the Study

In recent years, RL has gained much attention among researchers in various fields. Throughout the years, researchers have proposed various action selection strategies such as e-greedy and softmax to strike a balance between exploration and exploitation (Tijsma et al., 2017). However, these action selection strategies are designed for a stochastic environment and do not address a suitable action selection strategy for a fully deterministic environment. Researchers have also proposed various QL approaches for instance double QL (DQL), deep QN (DQN), deep double QL (DDQN), and dueling architecture which contributes to improving the convergence rate (Van Hasselt, n.d.), (Li, 2017), (Z. Wang et al., 2016). There are several research gaps unexplored by these studies despite these studies making substantial progress. First, these studies only consider stochastic environments and do not investigate the effects of the proposed QL approaches in the studies in a fully deterministic environment. Secondly, they assume the use of well-designed reward functions and fail to address problems where it is difficult and impractical to find an accurate reward function. This research aims to address these research gaps by integrating human intelligence

into QL, proposes a suitable action selection strategy for the fully deterministic environment and investigates the effects of a proposed QL approach in previous studies, DQL in a fully deterministic environment. As of today, RL is far from achieving intelligence comparable to humans, lacking crucial skills such as creativity, imagination, and critical thinking. Consequently, RL algorithms often require a tremendous amount of data to learn effectively. Integrating human knowledge into RL can significantly reduce its data required, increase its reliability and robustness, and build explainable RL systems (Deng et al., 2020). The findings of this study will have practical implications for various fields, such as healthcare, education, robotics, and marketing with complex and dynamic environments where it's difficult to design accurate reward functions (Deng et al., 2020) (Andalibi & Buss, 2020b).

1.4 Project Objectives

The objectives of this program are:

- To integrate, program, and investigate the reinforcement learning approach with human inputs using a suitable set of hyperparameters to improve the accuracy of the FER system and the agent learning convergence rate, and provide conclusions using simulations.
- To propose, program, and investigate an efficient and intelligent action selection strategy suitable for the deterministic environment in the reinforcement learning approach and provide conclusions using simulations.
- To program and investigate the effects of DQL, a reinforcement approach from previous studies in the fully deterministic environment.

1.5 Proposed Approach

This research proposes two-state QL with human feedback (TS-QL-HF), which is a novel approach integrated with human feedback, that improves convergence rate, robustness, and the accuracy of identifying facial emotions when compared to TS-QL. TS-QL-HF has four main features.

Firstly, to address the usage of non-optimal hyperparameters, TS-QL-HF utilizes a learning rate of 1 and discount factor of 0. By using this set of hyperparameters, the convergence rate is increased significantly as the agent do

not waste time considering the past and future rewards in a fully deterministic environment where the outcome of each action is predictable and consistent throughout the episodes.

Secondly, to address the usage of inaccurate reward function, TS-QL-HF introduces human reward feedback as a substitute for immediate rewards. The purpose is to augment the system with the reliability and accuracy of human intelligence in the reward function. By integrating human knowledge into RL, it improves the accuracy of the FER system.

Thirdly, to address the randomness in action selection, TS-QL-HF introduces a novel action selection strategy designed for a fully deterministic environment, one-shot reward-based ϵ decay (one-shot RBED). One-shot RBED decays its ϵ to 0 once a positive reward is received. If no positive reward is received, the ϵ will remain at 1. By using this strategy, the convergence rate is increased as the agent does not waste time exploring the environment after the optimal action has been found while balancing exploration and exploitation intelligently. It is important to note that two actions that result in a positive are considered equal optimal action as they provide the same value in accuracy enhancement of the FER system.

Lastly, to address the unnecessary multi-objective learning, TS-QL-HF approach separates the learning of each misclassified image into separate QL tasks. This eliminates the time wasted unlearning past knowledge before learning the new optimal policy, improving the convergence rate and robustness of TS-QL.

However, the effectiveness of using human feedback in TS-QL-HF for FER systems is dependent on the quality of human input. The quality of human feedback can be affected by factors such as fatigue, lack of attention, and inconsistency. This can lead to human errors in the labeling of the emotions, which can adversely affect the performance of the system.

1.6 Hypotheses

H1: Learning rate of 1 and discount factor of 0 reduces the number of iterations required to converge while not causing any performance penalty.

H2: TS-QL-HF improves the accuracy of the FER system and reduces the number of iterations required to converge compared to TS-QL.

H3: One-shot RBED and harmonic RBED improves the accuracy of TS-QL and TS-QL-HF.

H4: Using double QL (DQL) improves the accuracy of TS-QL-HF and TS-QL.

1.7 Project Scope

The improved FER system is expected to have the following deliverables:

1. Optimal QL hyperparameters are utilized in the QL of the FER system to speed up convergence rate.
2. Human feedback is incorporated into the reward function of TS-QL in the FER system as TS-QL-HF to improve accuracy of the FER system.
3. An intelligent and efficient action selection strategy is implemented in TS-QL-HF to speed up convergence rate.
4. The multi-objective of QL is changed to single objective to improve robustness and speed up convergence rate.
5. Human participation is minimized in the learning process to reduce potential human error.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

CNN is a major contributor to the accuracy of the FER system in predicting facial emotions. This literature review discusses the characteristics and differences between the models InceptionV3, ResNet50, MobileNetV2. These models are selected because they are promising models for FER system used by (Ilona, 2020). (Szegedy et al., n.d.), (He et al., 2015), (Sandler et al., 2018), (*Keras Applications*, n.d.) provides a comprehensive insight into the three CNN models.

Choosing a suitable action selection strategy plays a crucial role in reducing the randomness in action selection. Throughout the years, there have been a few commonly used action selection strategies such as the softmax selection. This literature review discusses the softmax selection (Whiteson et al., 2007), epsilon selection (Whiteson et al., 2007), and reward bases epsilon decay RBED (Maroti, 2019), and proposes two action selection strategies designed for deterministic environments. These strategies use different techniques to balance exploration and exploitation. Epsilon selection and softmax are selected because they are widely used, and they have been shown to be highly effective (Whiteson et al., 2007). RBED are selected because the concept of decaying epsilon based on rewards is expected to work well in the deterministic environment.

There are many contributing factors that lead to the slow and unstable convergence of QL approaches. One of the major factors is it suffers from the curse of dimensionality when scaled, which leads to a significant decrease in the convergence rate. The Q-table becomes too large to store in memory and impractical to learn from as the state-action space grows exponentially. To handle high-dimensional data, (Mnih et al., 2015) proposed the DQN algorithm, a variant of Q-learning. Over the years, researchers have proposed various extensions to improve the DQN algorithm and address the problems faced by traditional QL. These extensions include Double DQN (DDQN), Prioritized

Experience Replay (PER), and Dueling Architecture (DA), proposed by (Van Hasselt, n.d.), (Schaul et al., n.d.), and (Z. Wang et al., 2016). These extensions enhance the performance and significantly speed up the convergence rate of the DQN algorithm. This literature discusses the DQN approach and the existing extensions, and proposes a human feedback solution to further improve the convergence rate of QL. A paper by (Li, 2017) provides valuable insights into the DQN algorithm and its advancements. These extensions of DQN instead of extensions of QL are selected because DQN is a suitable tool when the action space is extended. The action space is expected to be extended in the future works where a more complex dataset is used.

2.2 Review of CNN models

Table 2.2 shows the description, advantages, and disadvantages of each CNN model used in the FER system from the previous work (Ilona, 2020). The CNN models reviewed in this section are InceptionV3 (Szegedy et al., n.d.), ResNet50 (He et al., 2015), and MobileNetV2 (Sandler et al., 2018). The goal of this review is to understand the differences between the models to help select a suitable model for this FER system. A real-life FER system is expected to be efficient and inexpensive. However, the cost of RL and human feedback is expensive, which makes deeper CNN models an ideal choice. Based on this review of models, InceptionV3 appears to be an ideal model for this FER system.

Table 2.2: CNN models used in the FER system are from the previous work (Ilona, 2020) comparison.

CNN models	Description	Advantage	Disadvantage
InceptionV3	Utilizes Inception modules with various kernel sizes for improved feature extraction.	High accuracy.	Moderate efficiency, may not be suitable for mobile and embedded devices.

ResNet50	Employs residual connections to mitigate vanishing gradient problem.	Moderate accuracy.	Highly inefficient, not suitable for mobile and embedded devices.
MobileNetV2	Uses depth wise separable convolutions for lightweight yet accurate models.	Very efficient on mobile and embedded devices.	Low accuracy on complex tasks.

2.3 Review of Action Selection Strategies

Table 2.3 shows the description, advantages, and disadvantages of three action selection strategies in RL. The action selection strategies reviewed in this section are SoftMax (Whiteson et al., 2007), ϵ -greedy (Whiteson et al., 2007), and reward based ϵ -decay (RBED) selection (Maroti, 2019). The goal of this review is to explore the techniques used in existing strategies. In SoftMax selection, selecting actions with probabilities proportional to their estimated values in SoftMax selection is unnecessary due to the deterministic environment. In the ϵ -greedy selection, when a best action occurs, it is unnecessary to explore any other actions because multiple positive actions lead to the same amount of accuracy enhancement to the FER system. In the RBED, the ϵ -decay formula is acting overly conservative, and it is not suitable for this environment where the environment is relatively simple. However, the idea of decaying ϵ based on rewards in RBED appears to be a promising approach for this environment where outcomes of the actions are predictable and consistent throughout episodes. Therefore, this research proposes two novel new action selection strategies, which are one-shot RBED and harmonic RBED.

Table 2.3: Existing action selection strategies.

Strategy	Description	Advantages	Disadvantages
SoftMax Selection	Selects actions with probabilities proportional to their estimated values.	Focus on exploring the most promising suboptimal action.	Higher computational cost.
Epsilon Greedy Selection	Chooses the best action most of the time and explores with a small probability ϵ .	Simplicity and ease of implementation.	May converge to suboptimal solutions without proper tuning of ϵ .
Reward Based Decay (RBED)	Adjusts exploration rate based on agent's reward performance. Exploration rate decreases only when a certain reward threshold is surpassed.	Adapts exploration rate dynamically to agent's learning progress.	Slows down convergence significantly due to the choice of ϵ -decay formula.

2.4 Review of Existing QL Extensions

Table 2.4 shows the description of each algorithm, advantages, and disadvantages. The three algorithms are modifications to the standard DQN that aim to improve its convergence rate by addressing various problems with DQN. DQN addresses the limitation of QL in high-dimensional action-state spaces. DDQN addresses the problem of overestimation bias, PER addresses the issue of experience replay, and DA enhances network architecture.

Table 2.4: Existing Solutions Comparisons

Algorithm	Description	Advantages	Disadvantages
DQN (Deep Q-Network)	Uses a deep neural network to approximate the Q-function.	Can handle high-dimensional action-state spaces.	Can overestimate Q-values, especially early in training.
Double DQN	Addresses the overestimation problem of DQN by using a separate network to select actions.	Reduces overestimation of Q-values.	Still requires a lot of training data to converge.
Prioritized Experience Replay	Prioritizes experiences based on their estimated learning potential, with more important experiences sampled more frequently.	Can lead to faster learning by focusing on important experiences.	Can be more complex to implement and tune.
Dueling Architecture	Separates the Q-function into a state value function and an advantage function, which allows for better estimation of Q-values.	Can handle situations where different actions have similar values.	Can be more complex to implement and may require more training time.

2.5 Performance of Existing Approaches

Table 2.5 compares the performance gain of dueling architecture on three algorithms: DQN, DDQN, PER using mean and median scores from an experiment conducted by (Z. Wang et al., 2016). The experiment was conducted on two metrics: 30 no-ops and human start, across all 57 Atari games. As shown in the table, the combination of DDQN, PER, and DA (Prior. Duel Clip) scores the highest on both metrics with mean and median of 595.9%, 567.0%, 172.1% and 115.3% respectively. The combination of DDQN, PER, and DA (Prior. Duel Clip) increased the performance of traditional DQN on both metrics with mean and median of 364%, 347.4%, 93% and 46.8% respectively. The combination of DDQN and DA (Duel Clip) increased the performance of traditional DQN on both metrics with mean and median of 145.2%, 124%, 72.4% and 48.6%. In conclusion, the table shows that the dueling architecture combined with the algorithms outperforms the baseline algorithms and traditional DQN significantly.

Table 2.5: Mean and median scores across all 57 Atari games, measured in percentages of human performance. (Z. Wang et al., 2016)

	30 no-ops		Human Starts	
	Mean	Median	Mean	Median
Prior. Duel Clip	591.9%	172.1%	567.0%	115.3%
Prior. Single	434.6%	123.7%	386.7%	112.9%
Duel Clip	373.1%	151.5%	343.8%	117.1%
Single Clip	341.2%	132.6%	302.8%	114.1%
Single	307.3%	117.8%	332.9%	110.9%
Nature DQN	227.9%	79.1%	219.6%	68.5%

2.6 Convergence Rate of Existing DQN Approaches

Table 2.6 summarizes the experimental results and comparison of the three solutions that improve the convergence rate of DQN. The table provides information on the experiment used for each approach, the improvement comparison made and the explanation of how the convergence rate is improved. DDQN and DA improved the convergence rate by addressing the bias overestimation problem of the DQN. PER improves the convergence rate of

DQN by selectively sampling experiences from the agent's memory based on their level of importance.

Table 2.6: Existing Approaches Convergence Rate Comparison

Solution	Experiment	Improvement in Convergence Rate	Explanation
DDQN	Gambling game of roulette and a small grid world with considerable randomness in rewards	Significant improvement compared to DQN in both linear and polynomial learning rates	DDQN reduces overestimation of Q-values and increases stability of the learning process.
PER	Selection of Atari games	Proportional PER converges faster than Ranked-Based PER and uniform DDQN	PER enables the agent to learn from important experiences more frequently and efficiently, leading to faster convergence.
DA	The Corridor game environment with 5, 10, and 20 actions	Consistently outperforms Single Architecture used in DDQN and PER	DA separates the estimation of state values and action advantages, allowing the agent to learn the value of different actions more effectively in each state.

2.6.1 DDQN

Figure 2.6.1 compares the convergence rate of DDQN with traditional DQN from an experiment conducted by (Van Hasselt, n.d.) on the gambling game of roulette and a small grid world, both of which had considerable randomness in rewards. The first row shows average rewards per time step. The second row shows the maximal action value in the starting state S . Averaged over 10,000 experiments. The discount factor was 0.95 in all experiments, and the learning rate was either linear or polynomial. The straight line represents the DQN while the dotted line represents DDQN. As shown in the figure, DDQN converges significantly faster in both learning rates. Additionally, the figure shows that polynomial learning rate performs better than linear learning rate. In conclusion, the figure shows that DDQN is a improves the convergence rate of DQN significantly.

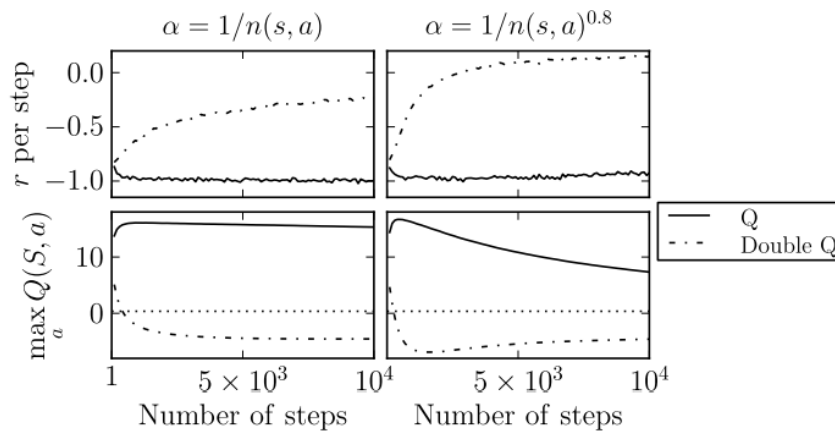


Figure 2.6.1: Learning Curves of DQN and DDQN on Linear and Polynomial Learning Rates (Van Hasselt, n.d.).

2.6.2 PER

Figure 2.6.2 compares the convergence rate of Proportional PER (Blue) and Ranked-Based PER (Red) with uniform DDQN (Black) from an experiment conducted by (Schaul et al., n.d.) on a selection of Atari games. The middle lines show the average scores and the shaded area around them shows the variation between scores. The green dashed lines show the human scores. As shown in the figure, Proportional PER converges faster compared to Ranked-Based PER and uniform DDQN. Ranked-Based PER converges faster than uniform DDQN significantly. In conclusion, the figure shows that PER improves the convergence rate of DDQN significantly.

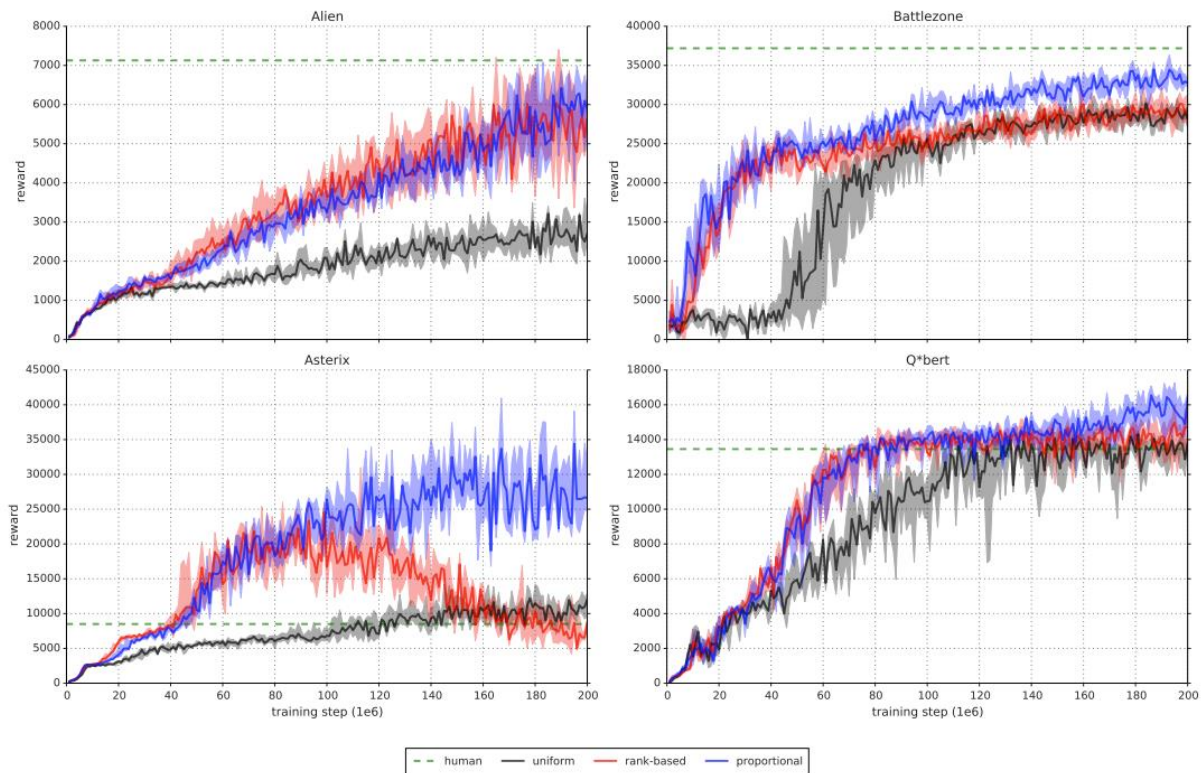


Figure 2.6.2: Detailed learning curves for rank-based (red) and proportional (blue) prioritization, as compared to the uniform Double DQN baseline (black) on a selection of games (Schaul et al., n.d.).

2.6.3 DA

Figure 2.6.3 compares the convergence rate of DA with traditional Single Architecture used in DDQN on game environment called "The Corridor". The plot (a) shows the game environment. The agent's actions are going up, down, left, right and no action. The plots (b), (c) and (d) show the squared error for policy evaluation with 5, 10, and 20 actions on a log-to-log scale. The red line represents the single-stream network while the green line represents the DA. As the figure shows, DA consistently outperforms a conventional single-stream network (SA) used in DDQN and PER, with the performance gap increasing with the number of actions. In conclusion, the figure shows that the DA improves the convergence rate of SA significantly.

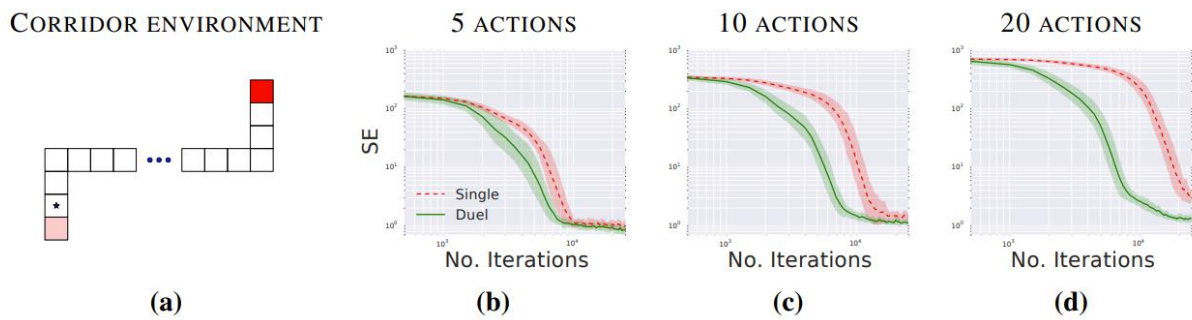


Figure 2.6.3: Learning Curves for Squared Error DA (red) and SA (green) on game environment "The Corridor" (Z. Wang et al., 2016).

2.7 Summary

This section discussed three CNN models used in the FER system. InceptionV3 is the model chosen as it provides a higher accuracy compared to ResNet50 and MobileNetV2. Then, this section discussed three existing action selection strategies. Due to the need for a strategy suitable for deterministic environment, this research proposes the novel one-shot RBED strategy.

This section also discussed DQN, and its three extensions proven to accelerate convergence rate. DQN allows QL to handle large state-action space by using neural network to estimate the Q-function. DDQN reduces the overestimation of Q-values by using separate networks to estimate Q-functions. PER enables the agent to learn from important experiences. DA separates the Q-function into a state value function and an advantage function. DQN is a significant improvement over QL, enabling it to handle high-dimensional action-state spaces. DDQN, PER, and DA have further enhanced the performance and convergence rate of DQN by addressing issues such as overestimation bias, experience replay, and network architecture. However, TS-QL does not suffer from high-dimensional action-state space due to the utilization of a small action-state space to solve a small problem, making the added complexity unnecessary. Furthermore, these approaches did not consider integrating human feedback as immediate reward, which fails to address the inaccurate reward function used in TS-QL. Despite the research gaps in the approaches, double QL proposed by (Van Hasselt, n.d.) shows that the concept of using a double estimator in DDQN is applicable and effective in traditional QL to reduce the overestimation of Q-values without overcomplicating the model. To address the research gaps, this research investigates the effectiveness of using human feedback in TS-QL as immediate rewards. Finally, this research investigates the effects of using QL and DQL alongside human feedback as immediate reward and provides conclusions.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

This chapter discusses the overview of research methodology which consists of four stages: planning, literature review, implementation, interpretation and report. Then, this chapter illustrates the workflow of the FER system using flowcharts and pseudocodes. Finally, this chapter outlines the specific steps to achieve research objectives, providing a detailed work plan to outline tasks and timelines planned for each stage of the project using a work breakdown structure (WBS) and Gantt chart.

3.2 Planning

Planning is the first stage of this research to ensure that this research will be conducted in an organized manner to achieve the research objectives effectively within the deadline given. The planning identifies the problem statement, research objectives, developing proposed approach, proposed solution, formulating hypotheses, and developing project scope. Additionally, the planning also consists of creating detailed WBS and Gantt chart that outlines the main tasks with the duration of the subtasks.

3.3 Literature Review

The literature review stage investigates the CNN models, common action selection strategies, extensions of QL, and identifies gaps in the existing research to be addressed in this research. The search is conducted using the keywords related to RL and QL on reputable academic databases, such as Google Scholar and IEEE Xplore, by relevance and date. The papers are selected based on their relevancy to expediting the convergence rate in QL and the quality of the sources.

3.4 Implementation

The implementation stage implements the three CNN models InceptionV3, ResNet50, and MobileNetV2. Then, the stage implements the approaches TS-QL, TS-QL-HF, TS-DQL, and TS-DQL-HF with action selection strategies one-shot RBED, harmonic RBED, and random selection. Python is used to program the approaches. The libraries used are ConvolutionalNeuralNetworks, Matplotlib, Numpy, OpenCV, Pandas, Scipy, Keras, Tensorflow, PIL, SimpleJson. The IDE used is PyCharm.

3.5 Interpretation and Report

The interpretation and report stage compares the performance of the approaches with the action selection strategies. First, the details of CNN layers, hyperparameters of CNN and QL, performance metrics, datasets used, experiment environments are determined. Then, three experiments are performed to evaluate the performance of TS-QL and TS-QL-HF with the action selection strategies and choose an optimal strategy. After that, the QL extension DQL is experimented with TS-QL and TS-QL-HF to evaluate the performance of DQL. The results are reported using graphs created using Python with Matplotlib library and tables created using Microsoft Word.

3.6 Flowchart

The purpose of this FER system is to accurately recognize human facial emotions from images using CNN and TS-DQL-HF. Figure 3.6 illustrates the high-level workflow of the FER system. The flowchart is organized into four main modules: image loading and preprocessing, CNN training, TS-DQL-HF, and performance evaluation. The TS-DQL-HF contains a submodule called Interactive DQL (I-DQL). The novelty of this research lies in the submodule I-DQL. This section provides a detailed workflow of the FER system.

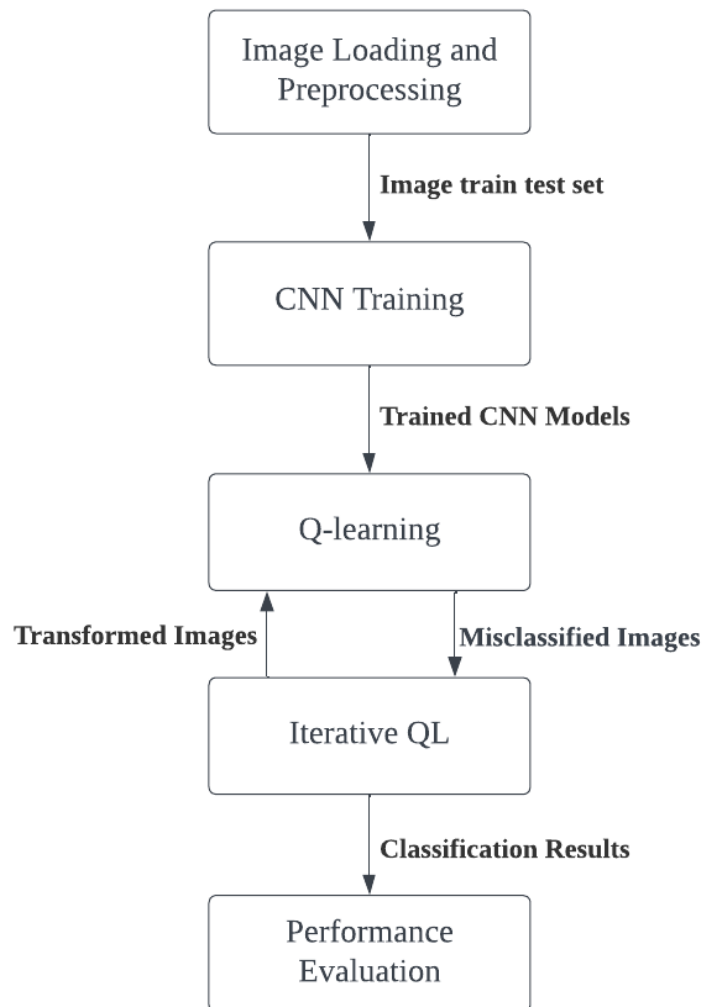


Figure 3.6 High-Level Overview of the FER System

3.7 Image Loading and Preprocessing

Figure 3.7 and Algorithm 1 illustrate the workflow and the detailed steps for image loading and the preprocessing module. The objective to program the reinforcement learning approach integrated with human inputs is benefited from this module. Image pre-processing converts loaded unstructured image data into structured image data so that it is usable for the CNN model to perform training and classify emotions. Image pre-processing has been a crucial step to process raw images.

The input of this module is images from the dataset FERG_DB_256 (Aneja Deepali and Colburn, 2017). This module involves five processes. The first process loads the images. The second process resizes the images into the size of 75x75. The third process removes the 4th dimension (transparency layer) of the images. The fourth process normalizes the images to the [0, 1] range using the normalization equation (1), where x represents an image.

$$X' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

The fifth process splits the images into 90% training set and 10% as testing set. The output of this module is the training and testing sets of the preprocessed images in a usable and effective format for the CNN training module to perform training.

The advantage of this module is reducing the training time and increases the inference speed of the CNN model, therefore enhancing the training efficiency of the FER system (Nelson, 2020). This module is useful for this FER system because of the constraints of limited computation power and memory in this research.

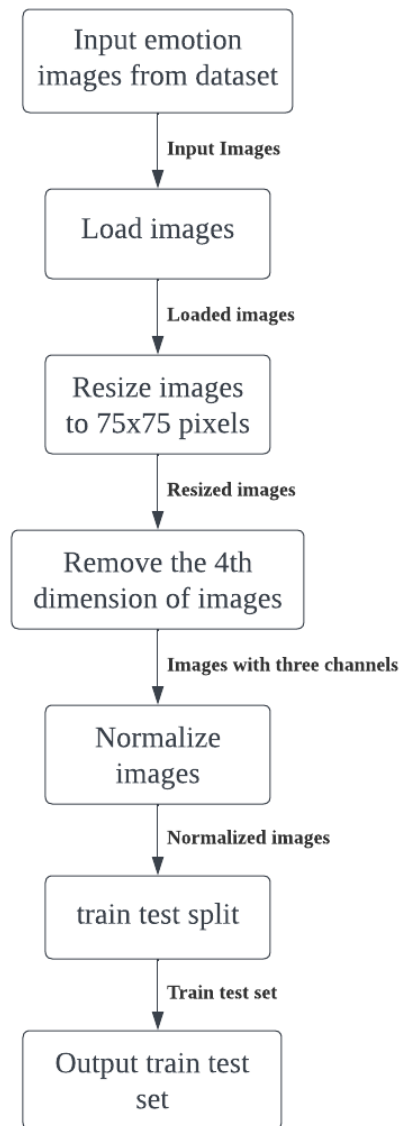


Figure 3.7: Image Loading and Preprocessing Flowchart

ALGORITHM 1: IMAGE LOADING AND PREPROCESSING

input: images from facial emotions dataset

output: train test set

```
1  loaded images  $\leftarrow$  load input images
2  for every image as img do
3      | resized img  $\leftarrow$  resize img to 75x75 pixels
4      | img with three channels  $\leftarrow$  remove 4th dimension of img
5      | normalized img  $\leftarrow$  normalize img using (1)
6  end for
7  training and testing sets  $\leftarrow$  split dataset into 90% training and 10%
   testing sets
```

3.8 CNN Training

The objective to program the reinforcement learning approach integrated with human inputs is benefited from this module. The goal of this module is to create three CNN models: InceptionV3, ResNet50, MobileNetV2 with custom classifier layers and perform training to classify seven facial emotions classes: anger, disgust, fear, joy, neutral, sadness and surprise.

The CNN models used are pretrained and custom classifier layers are added. During the training of the networks, the pretrained layers are frozen to prevent overriding their weights. The custom classifier layers are adopted from (Hafiz, n.d.). Figure 3.8.1 to 3.8.3 illustrates the architecture of the custom classifier layers. The details of layers of the CNN models are shown in the results and discussions chapter.

This module provides several advantages to this FER system. The most beneficial advantage of using InceptionV3 model, a CNN model, which reduces the number of parameters in artificial neural network (ANN) (Albawi, Mohammed and Al-Zawi, 2018). This enhances the performance of the FER system. Besides, CNN are proven to achieve higher accuracy on large datasets for image classification purposes such as FER compared to other methods due to its capabilities of joint feature and classifier learning (Gu *et al.*, 2018), therefore enhances the accuracy of the FER system. Using this CNN model is suitable for this FER system because it provides better accuracy and performance for FER to classify emotions with the constraints of computational power and memory.

3.8.1 InceptionV3 Network Architecture

The final layer of the default topless InceptionV3 model is a pooling layer with the output size of $8 \times 8 \times 2048$. The layer is connected to the four custom classifier layers. The first layer is a flatten layer with the output size of 2048. The purpose of this layer is to convert the results of the pooling layer into a single long continuous linear vector. The second layer is a dense layer with the output size of 1024. The purpose of the dense layer is to classify the results of the flatten layer into 1024 categories. The third layer is a dropout layer with the output size of 1024. The purpose of the dropout layer is to prevent overfitting during training. The final layer is a dense layer with the size of 7. The purpose of the dense layer is to classify the result of the dropout layer into the seven emotion categories.

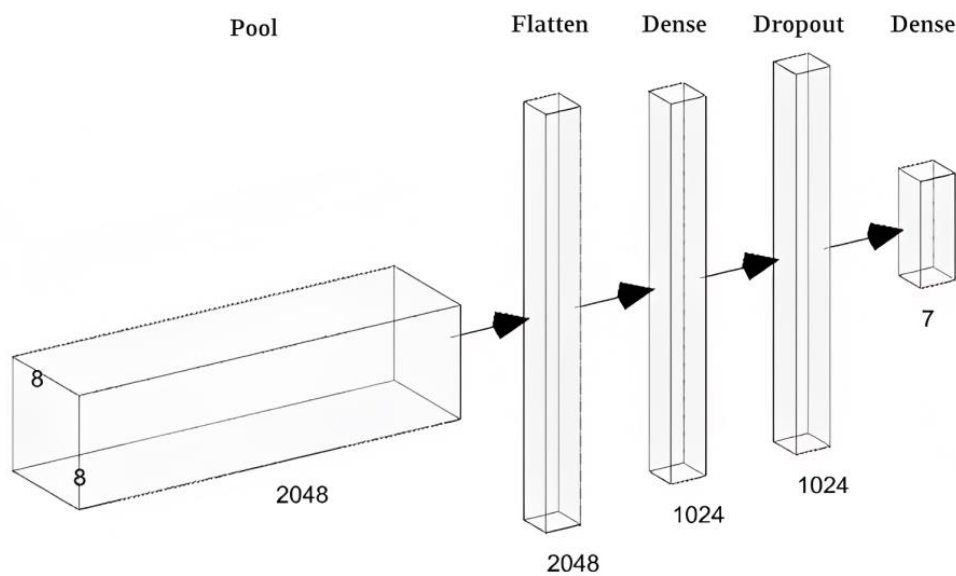


Figure 3.8.1: InceptionV3 Custom Classifier Architecture.

3.8.2 ResNet50 Network Architecture

The final layer of the default topless ResNet50 model is a convolutional layer with the output size of $3 \times 3 \times 2048$. The layer is connected to the five custom classifier layers. The first layer is a global average pooling layer with the output size of 1024. The purpose of this layer is to reduce the spatial dimensions of the output feature maps by the convolutional layer. The second to the fourth layers are dense layers with the size of 1024. The purpose of these layers is to classify the results of the global average pooling layer into 1024 categories. The final layer is a dense layer with the size of 7. The purpose of this layer is to classify the results of the previous dense layer into 7 emotion categories.

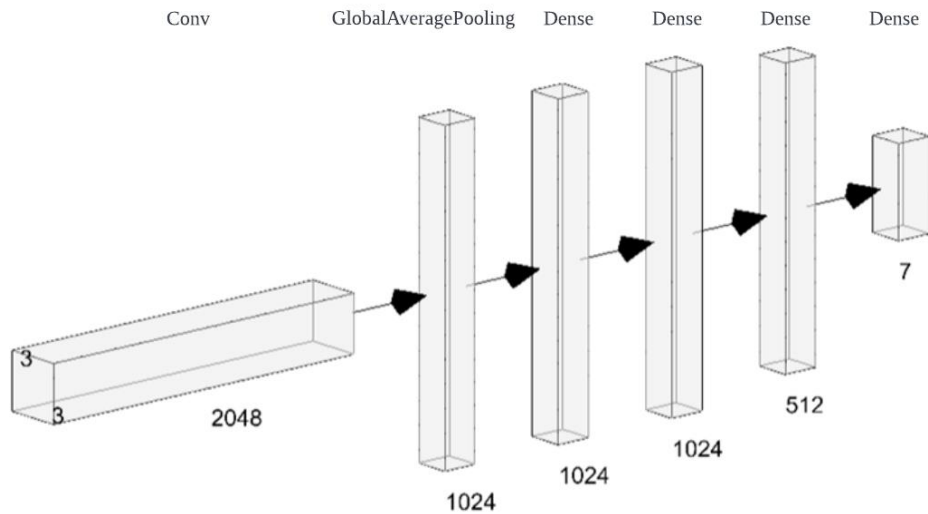


Figure 3.8.2: ResNet50 Custom Classifier Architecture.

3.8.3 MobileNetV2 Network Architecture

The final layer of the default topless MobileNetV2 model is a ReLU layer with the output size of $3 \times 3 \times 1028$. The layer is connected to the two custom classifier layers. The first layer is a global average pooling 2d layer. The purpose of this layer is to reduce the spatial dimensions of the output feature maps by the ReLU layer. The final layer is a dense layer. The purpose of this layer is to classify the results of the previous dense layer into 7 emotion categories.

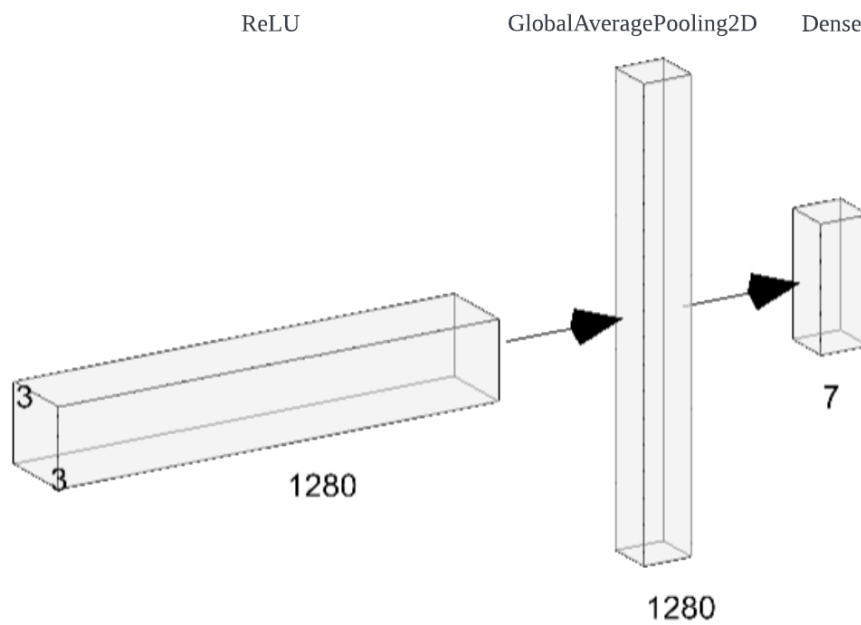


Figure 3.8.3: MobileNetV2 Custom Classifier Architecture.

3.8.4 Details of layers used in CNN Models

Table 3.8.4 shows the details of layers used in the CNN models. Parameters refer to the weights and biases in the model. The trainable parameters refer to the additional parameters created by the custom classifier layers. Depths refer to the number of layers in the model.

Table 3.8.4: Details of layers used in the CNN models.

Model	Total Parameters	Trainable Parameters	Total Depth
InceptionV3	23.9M	2.1M	192
ResNet50	28.3M	12.6M	111
MobileNetV2	2.27M	0.4M	106

3.9 Q-learning

Algorithm 2 shows the detailed steps for the QL module, which uses the submodule QL execution. There are four QL approaches implemented as the submodule. The approaches are TS-QL, TS-QL-HF, TS-DQL, and TS-DQL-HF. All the objectives are achieved in this module. The CNN model used is InceptionV3 as it is proven to provide the highest accuracy in this FER system. The strategies implemented are one-shot RBED, harmonic RBED, and random selection. All the approaches use one-shot RBED as it performs optimally in this deterministic environment as shown in the result and discussions chapter.

The inputs of this module are the trained CNN model, and the testing set. The module involves six processes. The first process performs initial classification on the testing set and obtains initial predictions. The second process executes a QL approach and obtains Q of misclassified images, the output Q -table from QL execution submodule. The third process chooses $\text{argmax}Q(s, a)$ from Q as the optimal action. The fourth process applies the optimal action on the misclassified images. The fifth process performs the second classification on the misclassified images and obtains updated predictions. The sixth process replaces misclassified predictions with updated predictions. The output of this module are updated predictions.

This module provides a significant advantage to the FER system. It provides different human perspectives, which are new, to the classifier. This enhancement has been shown to provide a minor accuracy as improvement compared to the state-of-the-art InceptionV3 model used in the FER system. Using the QL is suitable for the FER system because a minor improvement in accuracy has significant effects. For instance, accurate classification in image classification tasks like FER.

Algorithm 2: Q-Learning

input: trained InceptionV3 model, testing set
output: updated predictions

- 1 preds \leftarrow perform initial classification on testing set
- 2 **for** (all misclassified images) as mis_img **do**
- 3 $Q \leftarrow$ QL execution
- 4 optimal_action \leftarrow select optimal action from Q
- 5 transformed mis_img \leftarrow apply optimal_action on mis_img
- 6 new_pred \leftarrow perform second classification on transformed mis_img
- 7 preds \leftarrow update misclassified preds with new_pred
- 8 **end for**

3.10 Action Selection Strategies

The action selection strategies used in this FER system are one-shot RBED, harmonic RBED, and random selection. Table 3.10 shows the characteristics, advantages, and disadvantages of the strategies.

Table 3.10: The characteristics, advantages, and disadvantages of the action selection strategies used in this FER system.

	Characteristics	Advantages	Disadvantages
One-shot RBED	Stops exploration once a positive reward is received.	Highly efficient in a fully deterministic environment.	Not suitable for stochastic environment.
Harmonic RBED	Allows a small amount of exploration after a positive reward is received.	Slightly lower efficiency in a fully deterministic environment, can be suitable in a stochastic environment.	Not optimal in a fully deterministic environment.
Random Selection	Selects actions randomly throughout all episodes.	Can be useful in a fully stochastic environment.	Inefficient due to not performing exploitation.

3.10.1 One-Shot RBED

Algorithm 3 shows the detailed steps for the proposed one-shot RBED action selection strategy. The objective to propose, program, investigate an efficient and intelligent action selection strategy suitable for the deterministic environment in the reinforcement learning approach and provide conclusions using simulations is achieved in this strategy. The ε value decaying process has four processes. The first process initializes ε to 1. The second process gets the reward r , which can in the form of environmental feedback or human feedback containing the value 0 or 1. The third process checks if the current r is 1, if it is, the ε is decayed to 0. If it is not, the ε will be remained as 0. The

advantage of this strategy is it performs exploitation aggressively so that it fully utilizes the fact that in this deterministic environment, where the outcome of each action is the same throughout the episodes. This strategy is suitable in this FER system because of the deterministic nature of the environment.

Algorithm 3: On-Shot RBED

```

1    $\epsilon \leftarrow$  initialize  $\epsilon = 1$ 
2   for each episode do
3        $a \leftarrow$  select action using  $\epsilon$ -greedy with  $\epsilon$  value
4        $r \leftarrow$  get reward
5       if ( $r = 1$ )
6            $\epsilon = 0$ 
7       end if
8   end for

```

3.10.2 Harmonic RBED

Algorithm 4 shows the detailed steps for the proposed harmonic action selection strategy. The objective to propose, program, investigate an efficient and intelligent action selection strategy suitable for the deterministic environment in the reinforcement learning approach and provide conclusions using simulations is achieved in this strategy. The ϵ value decaying process has four processes. The first process initializes ϵ to 1. The second process gets the reward r , which can in the form of environmental feedback or human feedback containing the value 0 or 1. The third process checks if the current r is 1, if it is, the ϵ is decayed using the harmonic sequence formula in (2). If it is not, the ϵ will be remained as 0. The advantage of this strategy is it allows a small amount of exploration after receiving a reward to maximize cumulative reward. This strategy is suitable for a stochastic environment where the outcome of each action is unpredictable throughout the episodes. However, this research analyzes the effects of this strategy on this deterministic environment.

$$\varepsilon = \frac{1}{i+1^2} \quad \text{Eq. (2)}$$

Algorithm 4: Harmonic RBED

```

1       $\varepsilon \leftarrow$  initialize  $\varepsilon = 1$ 
2      for each episode do
3           $a \leftarrow$  select action using  $\varepsilon$ -greedy with  $\varepsilon$  value
4           $r \leftarrow$  get reward
5          if ( $r = 1$ )
6              Decay  $\varepsilon$  using (2)
7          end if
8      end for

```

3.10.3 Random Selection

The objective to propose, program, investigate an efficient and intelligent action selection strategy suitable for the deterministic environment in the reinforcement learning approach and provide conclusions using simulations is achieved in this strategy. The strategy is relatively simple as it only contains one step. It randomly selects an action throughout all episodes regardless of the rewards received by the agent. The strategy is suitable for this FER system when the collection of human feedback is expensive or impractical.

3.11 Environment Reward Function

Algorithm 5 shows the reward function of the environmental reward used by TS-QL and TS-DQL. The reward function works as follows, if $m_1 > m$, it maps the r as 1, else, it maps the r as -1.

Algorithm 5: Environmental Reward Function

```

1  if  $m_1 > m$ 
2      |  $r = 1$ 
3  else
4      |  $r = -1$ 
5  end if

```

3.12 QL Update Function

The QL update function is used to update the values in the Q-table for the approach TS-QL and TS-QL-HF.

$$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} (Q(s', a')) - Q(s, a)] \quad \text{Eq. (3)}$$

3.13 DQL Update Function

The DQL update function is used to update the values in the Q-table for the approach TS-DQL and TS-DQL-HF. The update function works as follows: if Q^A is selected, it updates $Q^A(s, a)$ using the Q^A update function (3), where a^* is the current best action in Q^A .

$$Q^A(s, a) = Q^A(s, a) + \alpha(s, a)(r + \gamma Q^B(s', a^*) - Q^A(s, a)) \quad \text{Eq. (4)}$$

If Q^B is selected, the $Q^B(s, a)$ is updated using the Q^B update function in Equation (4), where b^* is the current best possible action in Q^B .

$$Q^B(s, a) \leftarrow Q^B(s, a) + \alpha(s, a)(r + \gamma Q^A(s', b^*) - Q^B(s, a)) \quad \text{Eq. (5)}$$

3.14 Q-table Mean Function

The Q-table mean function is used to calculate the mean table of Q-table A and Q-table B in DQL.

$$\overline{Q(s, a)} = \frac{Q^A(s, a) + Q^B(s, a)}{2} \quad \text{Eq. (5)}$$

3.15 Iterative Q-Learning

The following approaches are the submodule of the Q-learning module. The approaches used are two-state Q-learning (TS-QL), two-state Q-learning with human feedback (TS-QL-HF), two-state double Q-learning (TS-DQL), and two-state double Q-learning with human feedback (TS-DQL-HF). This research proposes TS-QL-HF as it is shown to provide the highest accuracy with the most efficient convergence. All the approaches use one-shot RBED as the action selection strategy as it is shown to provide the optimal convergence. Table 3.15 shows the characteristics, advantages, and disadvantages of these approaches.

Table 3.15: The characteristics, advantages, and disadvantages of the approaches used in this FER system.

	Characteristics	Advantages	Disadvantages
TS-QL	Improves the accuracy of the FER system by providing a second perspective to the CNN model.	Suitable when human feedback is too expensive or impractical to collect.	Limited accuracy enhancement.
TS-QL-HF	Uses human feedback as immediate reward in substitution of environmental reward.	Provides a higher accuracy compared to TS-QL.	Not suitable when human feedback is too expensive or impractical to collect.
TS-DQL	Uses double Q-learning reduce the overestimation of q-	Reduces the overestimation of q-values.	Slows down convergence rate.

	values in traditional QL.		
TS-DQL-HF	Uses double Q-learning reduce the overestimation of q-values in traditional QL.	Reduces the overestimation of q-values.	Slows down convergence rate and decreases accuracy of the FER system.

3.15.1 TS-QL

Algorithm 6 shows the detailed steps for TS-QL. The objective to integrate, program, investigate the reinforcement learning approach with human inputs using a suitable set of hyperparameters to improve the accuracy of the FER system and the agent learning convergence rate, and provide conclusions using simulations is achieved in this approach.

The input of this approach is a misclassified image, and the output of this approach is the Q-table after completed the episodes. This approach involves ten processes. The first process initializes the Q-table by zeros for all state and action and ε by 1. The following processes repeats until the episodes end. The second process selects an action using ε -greedy with the ε -value. The third process gets the feature map in the last output layer of a topless CNN model using the input misclassified image. The third process calculates m , the standard deviation of the feature maps. The fourth process applies the action on the misclassified image. The fifth process gets m_1 the feature map in the last output layer of a topless CNN model using the modified misclassified image. The sixth process gets r using the environment reward function specified above. The seventh process checks if r is 1, then it maps the next state to 0 and decays ε to 0, else, it maps the next state to 1. The eighth process transits the current state to the next state. The ninth process update $Q_{s,a}$ using the Q-learning update function (3).

This approach provides a significant advantage to the FER system. TS-QL provides a second perspective, which is new, to the CNN models. This enhancement has been shown to provide a minor accuracy as improvement compared to the state-of-the-art InceptionV3 model used in the FER system.

Using TS-QL is suitable for the FER system because a minor improvement in accuracy has significant effects. For instance, accurate classification in image classification tasks like FER.

Algorithm 6: TS-QL

```

input: misclassified image
output:  $Q$ 
1   $Q \leftarrow$  initialize  $Q$  for  $\forall_s \in S$  and  $\forall_a \in A_s$  by 0,  $\varepsilon = 1$ 
2  for each episode do
3       $a \leftarrow$  select action using  $\varepsilon$ -greedy with  $\varepsilon$  value
4      feature maps  $\leftarrow$  get the feature map in the last output layer of
        a CNN model (topless) using misclassified image
5       $m \leftarrow$  calculate the standard deviation of the feature maps
6      modified image  $\leftarrow$  apply action on misclassified image
7      feature maps  $\leftarrow$  get the feature map in the last output layer of
        a CNN model (topless) using modified image
8       $m_1 \leftarrow$  calculate the standard deviation of the feature maps
9       $r \leftarrow$  gets  $r$  using environmental reward function
10     if ( $r = 1$ )
11          $s_t = 0$ 
12         decay  $\varepsilon$  to 0
13     else
14          $s_t = 1$ 
15      $s_t \leftarrow s_{t+1}$ 
16     update  $Q_{s,a}$  using (3)
17 end for
18  $Q \leftarrow Q$  after completed the episodes

```

3.15.2 TS-QL-HF

Algorithm 7 shows the detailed steps for TS-QL-HF. The objective to integrate, program, investigate the reinforcement learning approach with human inputs using a suitable set of hyperparameters to improve the accuracy of the FER system and the agent learning convergence rate, and provide conclusions using simulations is achieved in this approach.

The input of this approach is a misclassified image, and the output of this approach is the Q-table after completed the episodes. This approach involves six processes. The first process initializes the Q-table by zeros for all state and action and ε by 1. The following processes repeats until the episodes end. The second process selects an action using ε -greedy with the ε -value. The third process gets the r from the human evaluator. The reward can be either 1 or -1. The fourth process checks if r is 1, then it maps the next state to 0 and decays ε to 0, else, it maps the next state to 1. The fifth process transits the current state to the next state. The sixth process updates $Q_{s,a}$ using the Q-learning update function (3).

This approach provides a significant advantage to the FER system. TS-QL-HF uses an accurate reward function as a substitution of the standard deviation of feature map. This enhancement is shown to provide a higher accuracy compared to TS-QL in the FER system. Using TS-QL-HF is suitable in this FER system because the cost of collecting human feedback as immediate is inexpensive and practical to collect.

Algorithm 7: TS-QL-HF

input: misclassified image
output: Q

```

1   $Q \leftarrow$  initialize  $Q$  for  $\forall_s \in S$  and  $\forall_a \in A_s$  by 0,  $\varepsilon = 1$ 
2  for each episode do
3       $a \leftarrow$  select action using  $\varepsilon$ -greedy with  $\varepsilon$  value
4       $r \leftarrow$  gets  $r$  from human evaluator
5      if ( $r = 1$ )
6           $s_t = 0$ 
7          decay  $\varepsilon$  to 0
12     else
13          $s_t = 1$ 
14      $s_t \leftarrow s_{t+1}$ 
15     update  $Q_{s,a}$  using (3)
16 end for
17  $Q \leftarrow Q$  after completed the episodes

```

3.15.3 TS-DQL

Algorithm 8 shows the detailed steps for TS-DQL. The objective to program and investigate the effects of DQL, a reinforcement approach from previous studies in the fully deterministic environment is achieved in this approach.

The input of this approach is a misclassified image. The output of this module is \bar{Q} . This approach involves nine processes. The first process initializes the Q-table by zeros for all state and action and ε by 1. The following processes repeats until the episodes end. The second process selects an action using ε -greedy with the ε -value. The third process gets the feature map in the last output layer of a topless CNN model using the input misclassified image. The third process calculates m , the standard deviation of the feature maps. The fourth process applies the action on the misclassified image. The fifth process gets m_1 the feature map in the last output layer of a topless CNN model using the modified misclassified image. The sixth process gets r using the environment reward function specified above. The seventh process checks if r is 1, then it maps the next state to 0 and decays ε to 0, else, it maps the next state to 1. The eighth process randomly selects Q^A or Q^B to be updated. If Q^A is selected, it updates $Q^A(s, a)$ using the Q^A update function (4), where a^* is the current best action in Q^A . If Q^B is selected, the $Q^B(s, a)$ is updated using the Q^B update function (5), where b^* is the current best possible action in Q^B . The ninth process calculates the mean of Q^A and Q^B , \bar{Q} using (6).

This approach provides a disadvantage to the FER system. The DQL is shown to cause convergence rate to slow down without any accuracy gain by reducing the overestimation of q-values. This is due to the deterministic nature of the environment. Therefore, using this approach is not suitable for this FER system.

Algorithm 8: TS-DQL

```

input: misclassified image
output:  $\bar{Q}$ 
1  Q-table  $\leftarrow$  initialize  $Q^A, Q^B$  for  $\forall_s \in S$  and  $\forall_a \in A_s$  by 0,  $\varepsilon = 1.0$ 
2  for each episode do
3      a  $\leftarrow$  select action using  $\varepsilon$ -greedy with  $\varepsilon$  value
4      feature maps  $\leftarrow$  get the feature map in the last output layer of
      a CNN model (topless) using misclassified image
5       $m \leftarrow$  calculate the standard deviation of the feature maps
6      modified image  $\leftarrow$  apply action on misclassified image
7      feature maps  $\leftarrow$  get the feature map in the last output layer of
      a CNN model (topless) using modified image
8       $m_1 \leftarrow$  calculate the standard deviation of the feature maps
9       $r \leftarrow$  gets r using environmental reward function
10     if ( $r = 1$ )
11          $s_t = 0$ 
12         decay  $\varepsilon$  to 0
13     else
14          $s_t = 1$ 
15      $s_t \leftarrow s_{t+1}$ 
16     if update A
17         update  $Q^A$  using (4)
18     else if update B
19         update  $Q^B$  using (5)
20     end if
21 end for
22  $\bar{Q} \leftarrow$  calculate mean of  $Q^A$  and  $Q^B$  using (6)

```

3.15.4 TS-DQL-HF

Algorithm 9 shows the detailed steps for TS-DQL-HF. The objective to program and investigate the effects of DQL, a reinforcement approach from previous studies in the fully deterministic environment is achieved in this approach.

The input of this approach is a misclassified image. The output of this module is \bar{Q} . This approach involves nine processes. The first process initializes the Q-table by zeros for all state and action and ε by 1. The following processes repeats until the episodes end. The second process selects an action using ε -greedy with the ε -value. The third process gets the r from the human evaluator. The fourth process checks if r is 1, then it maps the next state to 0 and decays ε to 0, else, it maps the next state to 1. The fifth process randomly selects Q^A or Q^B to be updated. If Q^A is selected, it updates $Q^A(s, a)$ using the Q^A update function (4), where a^* is the current best action in Q^A . If Q^B is selected, the $Q^B(s, a)$ is updated using the Q^B update function (5), where b^* is the current best possible action in Q^B . The sixth process calculates the mean of Q^A and Q^B , \bar{Q} using (6).

This approach provides a disadvantage to the FER system. The DQL is shown to cause convergence rate to slow down and reduces the accuracy gain by reducing the overestimation of q-values. This is due to the deterministic nature of the environment. Therefore, using this approach is not suitable for this FER system.

Algorithm 9: TS-DQL-HF

input: misclassified image
output: \bar{Q}

```

1  Q-table  $\leftarrow$  initialize  $Q^A, Q^B$  for  $\forall s \in S$  and  $\forall a \in A_s$  by 0,  $\varepsilon = 1.0$ 
2  for each episode do
3      a  $\leftarrow$  select action using  $\varepsilon$ -greedy with  $\varepsilon$  value
9      r  $\leftarrow$  gets r from human evaluator
10     if (r = 1)
11          $s_t = 0$ 
12         decay  $\varepsilon$  to 0
13     else
14          $s_t = 1$ 
15      $s_t \leftarrow s_{t+1}$ 
16     if update A
17         update  $Q^A$  using (4)
18     else if update B
19         update  $Q^B$  using (5)
20     end if
21 end for
22  $\bar{Q} \leftarrow$  calculate mean of  $Q^A$  and  $Q^B$  using (6)

```

3.16 WBS

A Work Breakdown Structure (WBS) is created for task management of the project. It contains all the key deliverables for the project. The WBS of this project is listed as follow:

0.0 An Emotion Recognition System Integrated with Human Intelligence

1.0 Planning

- 1.1 Create Work Breakdown Structure (WBS)
- 1.2 Create Gantt Chart
- 1.3 Register Project Title
- 1.4 Registered Project Title
- 1.5 Study Project Background
- 1.6 Define Project Objectives
- 1.7 Define Project Scope
- 1.8 Identify Research Novelty
- 1.9 Define Problem Statement
- 1.10 Define Importance of Study
- 1.11 Completed Planning

2.0 Literature Review

- 2.1 Analyze CNN Models Used
- 2.2 Analyze Existing Action Selection Strategies
- 2.3 Analyze Extensions of Q-Learning
- 2.4 Compare Convergence of the Extensions
- 2.5 Identify Research Gap
- 2.6 Formulate Hypotheses
- 2.7 Propose TS-QL-HF
- 2.8 Completed Literature Review

3.0 Implementation

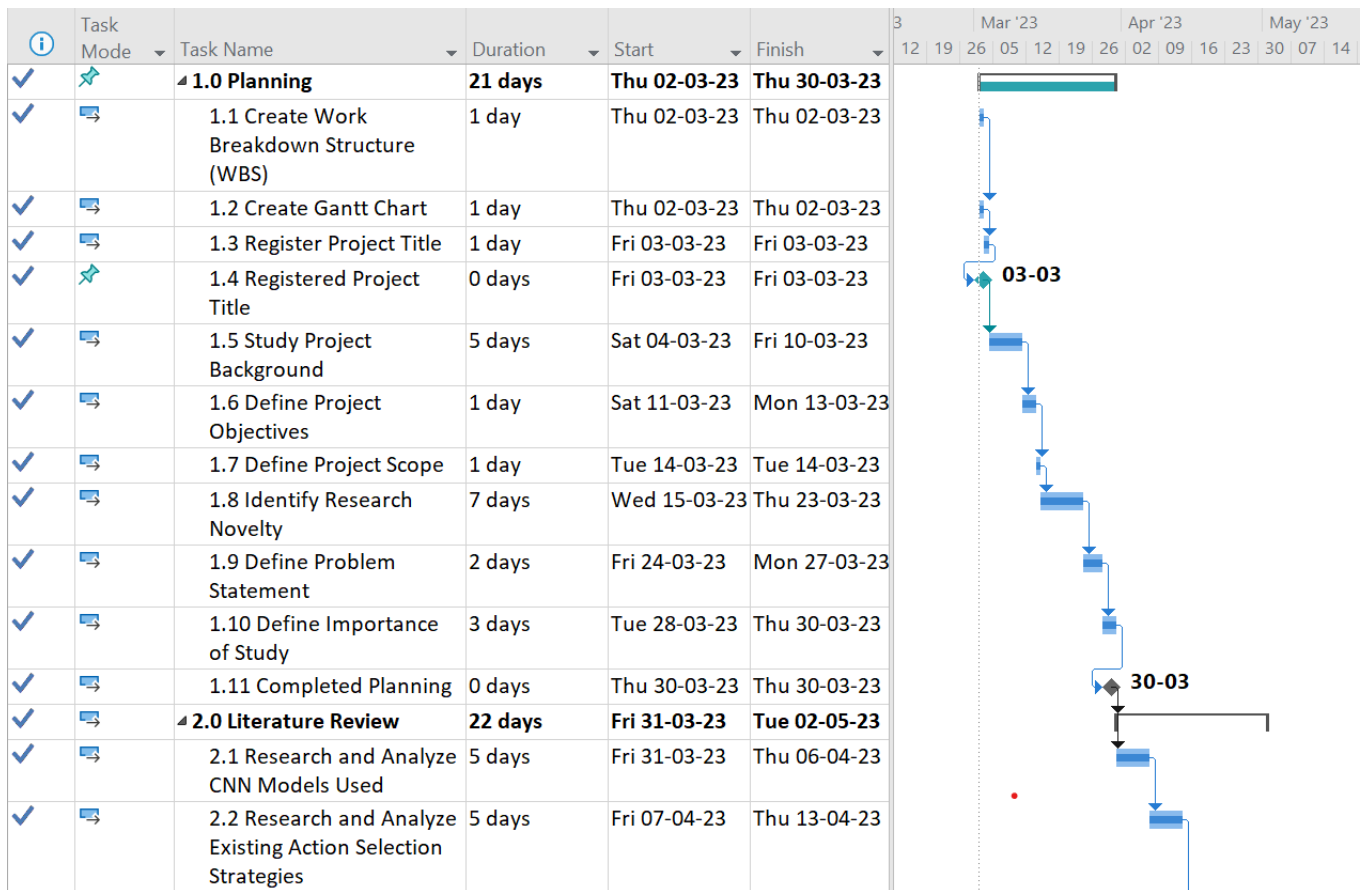
- 3.1 Define Methodology
- 3.2 Identify Research Method
- 3.3 Create Flowchart
- 3.4 Create Pseudocode
- 3.5 Implement TS-QL, TS-QL-HF, TS-DQL and TS-DQL-HF
- 3.6 Completed Implementation

4.0 Interpret and Report

- 4.1 Identify CNN Hyperparameters
- 4.2 Identify QL Hyperparameters
- 4.3 Identify Performance Metrics
- 4.4 Identify Experiment Environment
- 4.5 Compare Implemented Action Selection Strategies
- 4.6 Compare Performance of CNN models
- 4.7 Compare Performance of TS-QL and TS-QL-HF with Action Selection Strategies
- 4.8 Compare Performance of TS-QL and TS-QL-HF with Double Q-Learning
- 4.9 Completed Performance Comparison
- 4.10 Provide Conclusions
- 4.11 Provide Recommendations for Future Works
- 4.12 Completed Interpretation and Report

3.17 Gantt Chart

A Gantt Chart is created to outline the project schedule according to the WBS detailed with start and end dates. Figure 3.17 shows the Gantt Chart of this research. The Gantt Chart is separated by the following research phases: planning, literature review, implementation, interpretation and report. The earliest expected research completion time is on 10/8/2023. No slacks are available for this research as all the tasks are mandatory for the research's success. All the tasks are completed, and the all the milestones are achieved.



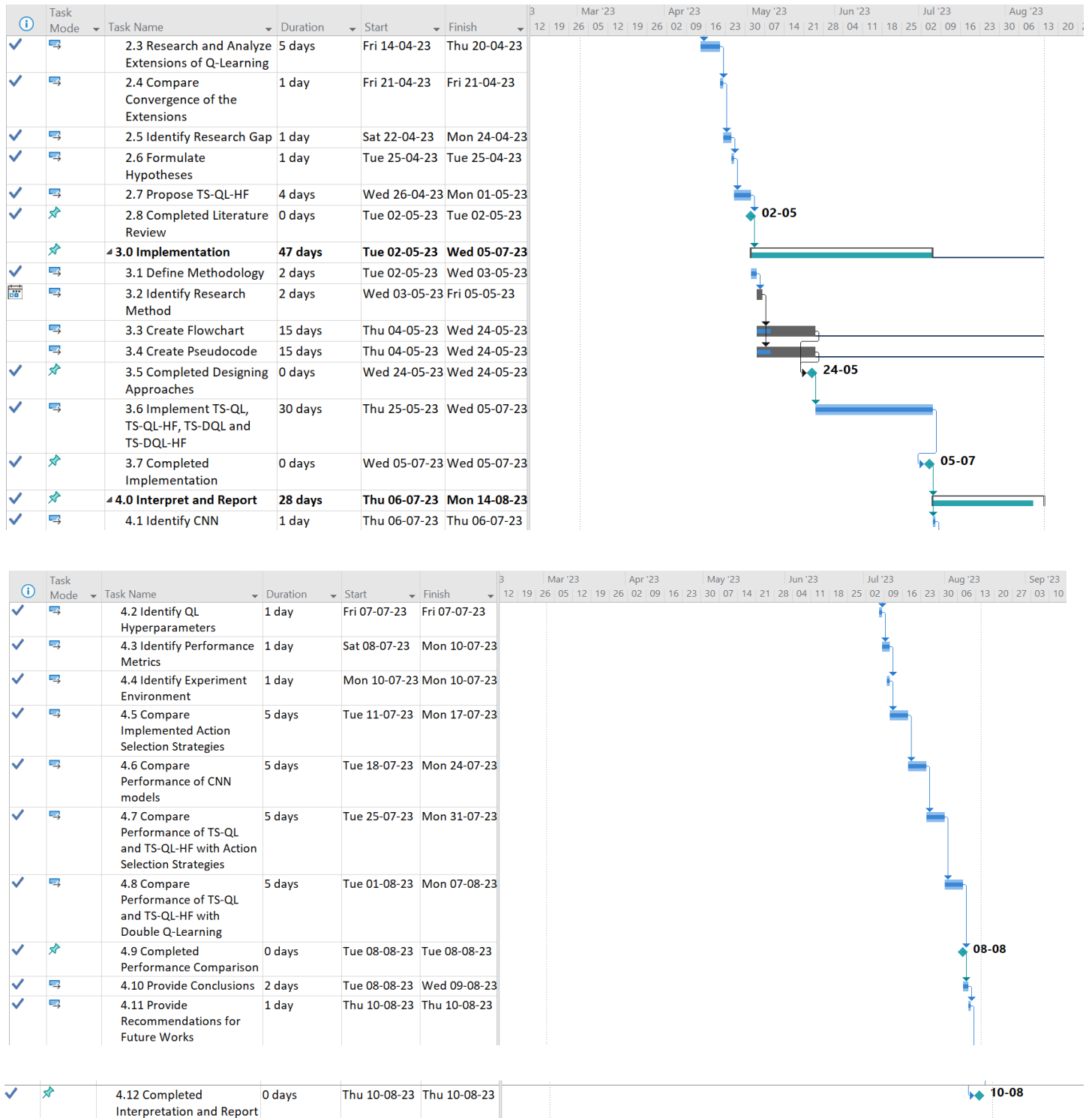


Figure 3.17 Gantt Chart on the Project.

3.18 Summary

This chapter discussed each phase of the research methodology to integrate human intelligence into the FER system. This chapter has discussed the workflow of the proposed FER system. Additionally, this chapter also discussed the project planning through WBS and Gantt chart.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter analyzes three state-of-the-art CNN models to identify a suitable model for this FER system. Next, this chapter compares the effects of using one-shot decay, harmonic decay, and random action selection strategies on TS-QL and TS-QL-HF to investigate the effects of each strategy on convergence rate and accuracy and to select the optimal strategy for the FER system. Finally, this chapter compares the performance of TS-QL, TS-DQL, TS-QL-HF, and TS-DQL-HF to investigate the effectiveness of reducing overestimation of Q-values using DQL for the FER system. The subsequent experiments consider the research scenario where the facial emotion dataset used is consistent and well controlled, which means the faces in the dataset are facing a similar direction and the dataset does not have any obstructions. The experiments also consider the research scenario where the human feedback received is highly reliable. Lastly, the experiments also consider the research scenario where the QL environment is fully deterministic, where the outcome of every action taken by the agent is predictable and consistent throughout the episodes. The preliminary results are shown in the appendix.

4.1.1 Hyperparameters used in CNN Models

Table 4.1.1 outlines the hyperparameters used in InceptionV3, ResNet50, and MobileNetV2 CNN models. Adam is an efficient optimizer algorithm proposed by (Kingma & Ba, 2014) which utilizes the momentum method to automatically adjust its learning rate. Learning rate is an important tuning parameter that determines the step size in each iteration. The batch size is another important tuning parameter used to determine the number of samples used in each iteration (Murphy, n.d.). The hyperparameters used are adopted from the previous works (Ilona, 2020) and (Hafiz, n.d.) without any fine tuning as this is not the primary focus of this research.

Table 4.1.1: Hyperparameters used for training InceptionV3, ResNet50, and MobileNetV2 CNN models.

	CNN Models
Optimizer	Adam
Learning rate (α)	0.001
Batch size	20

4.1.2 Hyperparameters used in QL Models

Table 4.1.2 outlines the hyperparameters used in TS-QL, TS-DQL, TS-QL-HF, and TS-DQL-HF models. The learning rate, α , in QL is an important turning parameter that determines the balance of exploration and exploitation of prior knowledge. $\alpha = 0$ makes an agent learn nothing as it exclusively exploits prior knowledge, while $\alpha = 1$ makes an agent consider only the most recent information thus ignoring prior knowledge to explore possibilities. $\alpha = 1$ is optimal during a fully deterministic environment (Sutton & Barto, 2005). The discount factor, γ , in QL is another important tuning parameter that determines the importance of future rewards. $\gamma = 0$ makes an agent short sighted by only considering current reward, while $\gamma = 1$ makes an agent strive for long-term high reward (Sutton & Barto, 2005). Therefore, for all QL models in the experiments, α is changed to 1.0 and γ is changed to 0.0.

Table 4.1.2: Hyperparameters used in TS-QL, TS-DQL, TS-QL-HF, and TS-DQL-HF models.

	QL Models
Learning rate (α)	1.0
Discount factor (γ)	0.0
Q-values initialization	0.0
Number of episodes	60

4.1.3 Performance Metrics

The performance comparisons in this research utilize two types of performance metrics, relating to the FER system and QL. Accuracy and F1 score are used for the FER system. F1 score is the harmonic means of precision and recall. On the other hand, max q-values and cumulative rewards are used for QL. Max q-value represents the optimal q-value in the current state and action and cumulative reward represents the amount of reward the agent has received so far. The max q-values are used to determine convergence while the cumulative rewards are used to determine the quality of the QL. It is determined that the convergence and cumulative reward does not represent the true quality of the learnt policy. Therefore, the accuracy and F1 score are used to evaluate the true quality of the policy learnt by the QL.

4.1.4 Dataset Used

The dataset used is (*FERG - V7 Open Datasets*, n.d.). FERG is a database of six cartoon characters (3 males and 3 females) called Ray, Malcolm, Jules, Bonnie, Mery and Aia. The dataset contains 55769 labeled facial expressions and the images of each character are grouped into 7 basic human emotions classes: anger, disgust, fear joy, neutral, sadness and surprise (*FERG - V7 Open Datasets*, n.d.). The dataset is consistent and well controlled, as all the faces in the dataset are front facing and the images do not contain any obstructions.



Figure 4.1.4: Sample images from the database. Top row (left to right): character Aia with expression 'fear', character Bonnie with expression 'joy' and character Jules with expression 'disgust'. Bottom row (left to right): character Malcolm with expression 'sadness', character Mery with expression 'neutral' and character Ray with expression 'surprise' (*FERG - V7 Open Datasets*, n.d.).

4.1.5 Experiment Environment

Cross validation is a widely used data resampling method used to estimate the true prediction performance of a model (Berrar, 2018). The variability of the model performance is calculated by using the standard deviation on the performance metrics used. The dataset is separated into 10 folds cross validation randomly. In each fold, the data is split into 90% training 10% testing sets. To validate the performance of the FER system during training, the training set is further split into 80% training and 20% validation set.

4.1.6 Action Selection Strategies

Action selection strategies play a crucial role in the performance of QL as it dictates the actions to be taken. The action selection strategies compared in the experiment called action selection strategies performance comparison are one-shot RBED, harmonic RBED, and random action selection strategy. The idea of reward based ϵ decay (RBED) is inspired by (Maroti, 2019). The one-shot RBED and harmonic RBED are novel strategies that are designed for this scenario where the environment is fully deterministic and the reward function using human feedback received are highly reliable. These two strategies focus highly on exploitation efficiency which reduces the expensiveness of collecting

human feedback by avoiding unnecessary explorations once a positive reward is obtained. This aggressive exploitation is suitable in this scenario because of the inexistence of suboptimal action as any action that results in the positive outcome provides the same amount of enhancement for the FER system.

Table 4.1.6: Table of comparison on the one-shot RBED, harmonic RBED, and random action selection strategies.

Strategy	Description	Advantage	Disadvantage
One-shot RBED	<ul style="list-style-type: none"> - Decays ϵ to 0 once a positive reward is received. - Maintains ϵ at 1 until a positive reward is obtained. 	Very efficient in exploiting learnt policies.	Not suitable in complex problems where suboptimal solutions exist.
Harmonic RBED	<ul style="list-style-type: none"> - Decays ϵ using harmonic sequencing when a positive reward is received. - Maintains ϵ at 1 until a positive reward is obtained. 	Efficient in exploiting learnt policies.	Slower in exploiting learnt policies compared to one-shot decay.
Random action selection	<ul style="list-style-type: none"> - Selects actions randomly without considering rewards. 	Ensures constant exploration.	Lacks the ability to exploit learnt knowledge.

4.1.7 Overestimation of Q-values

DQL addresses the issue of overestimating q-values due to the usage of the max operator in traditional QL and it is proven to cause a slower and unstable convergence. However, the experiments done in the previous works (Li, 2017; Van Hasselt, n.d.; van Hasselt et al., 2015) is in stochastic environments thus failing to address the effects of DQL in fully deterministic environments. In the following experiment on DQL, the experiment investigates the effects of overestimation of q-values and the effects of using DQL in a fully deterministic environment.

4.2 CNN Models Performance Comparison

The goal of this experiment is to compare the CNN models and identify a suitable model to be used in the performance comparison in the specified experiment scenario and using the specified hyperparameters. The condition for suitable are the least amount of misclassified image to minimize the cost of collecting human feedback.

Figure 4.2 shows the performance comparison of the three CNN models: InceptionV3, ResNet50, and MobileNetV2 on the FER system while table 4.2 shows the average score of the models.

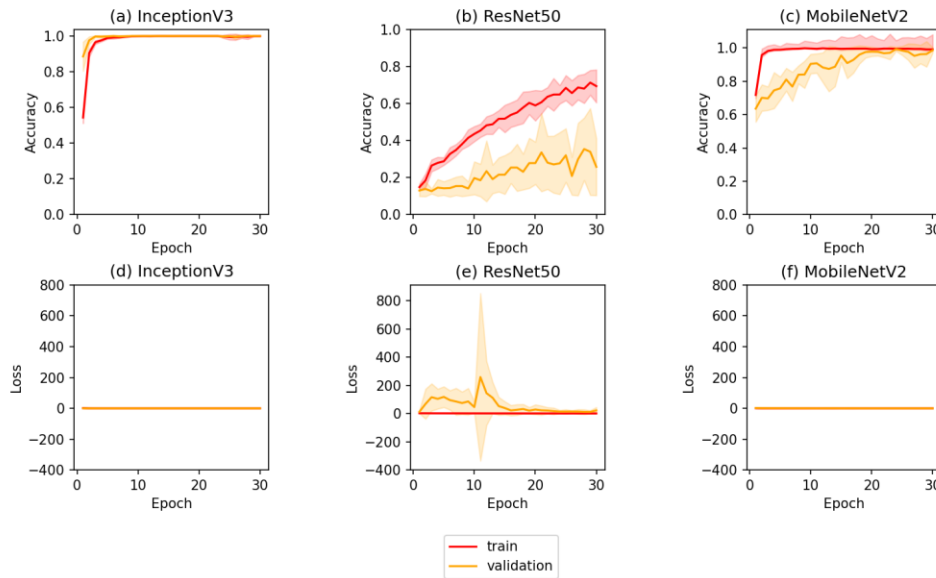


Figure 4.2: Performance comparison of InceptionV3, ResNet50, and MobileNetV2. The x-axis is the number of training iterations, and the y-axis is the accuracy of the models. The first row shows the training (red) and validation (yellow) accuracy of each model respectively. The second row shows the training (red) and validation (yellow) loss of each model respectively. The shaded area shows the standard deviation, the range of accuracy and loss achievable by the models. The average accuracy and loss are obtained by averaging the results over 10 folds. The standard deviations are obtained by calculating the standard deviation of the results over 10 folds. ResNet is shown to overfit due to the high number of parameters used. InceptionV3 is shown to provide a stable accuracy with the most efficient convergence rate.

Table 4.2: Averaged score of the models. The F1 scores used are macro as the classes are imbalanced. ResNet50 is shown to provide a very low accuracy due to overfitting caused by the high number of parameters used. InceptionV3 is shown to provide the best accuracy, loss, and F1 score.

	Accuracy	Loss	F1 Score
InceptionV3	0.9849 ± 0.0168	0.0497 ± 0.0566	0.9826 ± 0.0199
ResNet50	0.2621 ± 0.1485	21.1658 ± 23.1655	0.1571 ± 0.1635
MobileNetV2	0.9373 ± 0.0296	0.2377 ± 0.1574	0.9355 ± 0.0300

The results show that InceptionV3 converges most efficiently while providing the highest validation accuracy and the lowest average loss compared to the other models. ResNet50 is unable to converge and provides the lowest average accuracy and highest loss. The validation accuracy is significantly lower than training accuracy which clearly indicates overfitting. The reason for overfitting is because the model is shown to have a very high number of parameters. MobileNetV2 converges slower than InceptionV3 and provides a slightly lower accuracy compared to InceptionV3.

This experiment concludes by selecting InceptionV3 as the base model for this FER system as it provides the highest average accuracy and lowest average loss despite being a slower model compared to MobileNetV2 due to the expensiveness of collecting human feedback for the QL model as the number of misclassified images increases.

4.3 TS-QL and TS-QL-HF Action Selection Strategies Performance Comparison

The goal of this experiment is to investigate the following: a) the effects of human feedback in TS-QL; b) the effects of using different action selection strategies on TS-QL and TS-QL-HF. Figure 4.3 shows the effect of one-shot the three action strategies on the max q-values and cumulative reward of TS-QL, before applying human feedback, and TS-QL-HF, after applying human feedback. Table 4.3 shows the effects of using human feedback, and the three action selection strategies on accuracy and F1 score of the FER system.

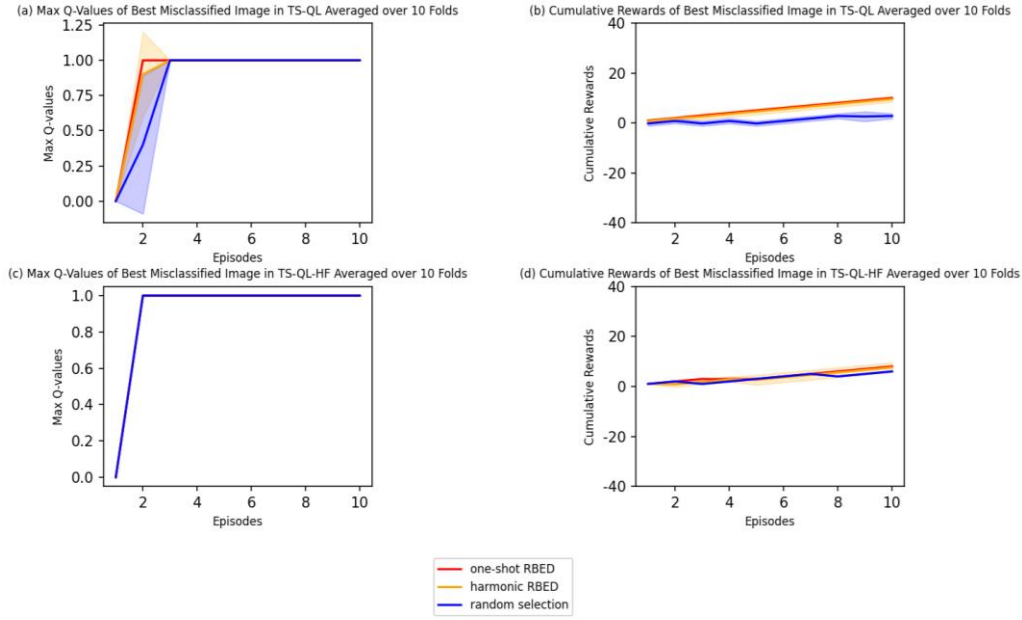


Figure 4.3: Performance comparison of one-shot RBED (red), harmonic RBED (yellow), and random selection strategies (blue) used in TS-QL and TS-QL-HF on the ‘best’ misclassified image. The ‘best misclassified image’ represents the misclassified image that has the highest max q-values in each fold. The first row shows the effects of the action selection strategies on TS-QL (without human feedback) on max q-values of the image. The second row shows the effects of the action selection strategies used in TS-QL-HF (human feedback) on max q-values and cumulative reward. In the first column, x-axis is the number of training episodes, and y-axis is the max q-values, the optimal action in each state and action. In the second column, x-axis is the number of training episodes, and y-axis is the cumulative rewards, the total reward in each episode. The shaded area shows the standard deviation, the range of max q-values and cumulative rewards achievable. One-shot RBED is shown to provide the most efficient

convergence with the highest cumulative reward consistently. TS-QL-HF is shown to provide a more stable convergence despite the action selection strategies used.

Table 4.3: Average accuracy and F1 score of one-shot RBED, harmonic RBED, and random selection used in TS-QL and TS-QL-HF compared to the baseline. The baseline is the accuracy and F1 score of InceptionV3 model before using QL. Action selection strategies is shown to not improve nor worsen the accuracy and F1 score due to the deterministic environment. TS-QL-HF is shown to improve the accuracy and F1 score due to the usage of human feedback.

	Accuracy	F1 Score
Baseline	0.9849 ± 0.0168	0.9826 ± 0.0199
TS-QL-HF One-Shot RBED	0.9936 ± 0.0137	0.9933 ± 0.0140
TS-QL-HF Harmonic RBED	0.9936 ± 0.0137	0.9933 ± 0.0140
TS-QL-HF Random Selection	0.9936 ± 0.0137	0.9933 ± 0.0140
TS-QL One-Shot RBED	0.9893 ± 0.0143	0.9882 ± 0.0165
TS-QL Harmonic RBED	0.9893 ± 0.0143	0.9882 ± 0.0165
TS-QL Random Selection	0.9893 ± 0.0143	0.9882 ± 0.0165

One-Shot RBED max q-values converges the most efficiently and yields the most average cumulative reward for TS-QL compared to other strategies. One-shot RBED has the same efficient max q-values convergence compared to other strategies but receives a higher average cumulative reward compared to random selection while yielding the same average cumulative reward as harmonic RBED for TS-QL-HF. These observations show that one-shot RBED is the most efficient in exploiting learnt policies.

Harmonic RBED max q-values converges slightly slower than one-shot RBED similarly to random selection but yields the same amount of cumulative reward as one-shot RBED for TS-QL compared to other strategies. Harmonic RBED max q-values has the same efficient converges compared to other strategies but yielding a similar amount of cumulative reward as one-shot RBED for TS-QL-HF. These observations show that harmonic RBED is slightly more conservative in exploiting learnt policies without losing its cumulative reward.

Random selection max q-values converges the slowest while yielding the lowest cumulative reward compared to other strategies for TS-QL. Random selection max q-values converges similarly to other strategies efficiently while receiving the lowest cumulative reward compared to other strategies compared to other strategies. These observations show that by selecting actions randomly without considering rewards, this strategy wastes time exploring unimportant parts of the environment thus yielding a lower average cumulative reward.

TS-QL-HF has a lower average cumulative reward for the action selection strategies used while the accuracy and F1 score is higher compared to TS-QL. This shows that human feedback is a more accurate reward function compared to the standard deviation of feature map, which leads to the average cumulative reward of TS-QL-HF being more accurately representing the small accuracy and F1 score enhancement provided by these two approaches. However, it is worth noting that TS-QL-HF is more time costly due to the time spent collecting human feedback. The action selection strategies do not affect the average accuracy and F1 score FER system for both TS-QL and TS-QL-HF, this is due to the nature of the deterministic environment, which causes the timing or sequence of actions to be irrelevant, making the accuracy and F1 score of the FER system independent from the action selection strategy.

This experiment concludes by selecting One-Shot RBED as the action selection strategy for this FER system due to its efficiency in exploiting learnt knowledge and ability to increase the average cumulative reward for TS-QL while performing similarly to Harmonic RBED and random selection in TS-QL-HF in this research scenario. This experiment also concludes that TS-QL proves to be a suitable approach when the prediction performance of the FER system is critical, whereas TS-QL-HF proves to be a suitable approach when the prediction accuracy and F1 score of the FER system is critical as both approaches is shown to be capable of improving the average accuracy and F1 score of the FER system.

4.4 QL and Double QL Performance Comparison

The goal of this experiment is to investigate the effects of using DQL in TS-QL and TS-QL-HF. Figure 4.4 shows the effects of q-values overestimation and the effects of using DQL in TS-QL and TS-QL-HF on the max q-values and cumulative reward of TS-QL, before applying human feedback, and TS-QL-HF, after applying human feedback. Table 4.4 shows the effects of q-values overestimation and the effects of using DQL in TS-QL and TS-QL-HF on accuracy and F1 score of the FER system.

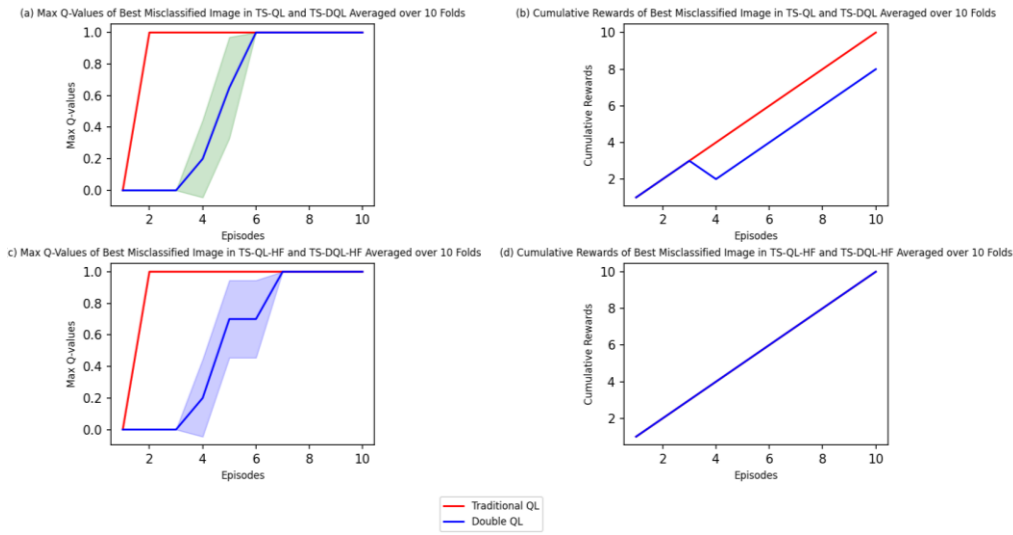


Figure 4.4: Performance comparison of Traditional QL (red) and DQL (blue) in TS-QL and TS-QL-HF on the ‘best’ misclassified image. The ‘best misclassified image’ represents the misclassified image that has the highest max q-values in each fold. The first row shows the effect of using DQL on TS-QL (without human feedback). The second row shows the effect of using DQL of TS-QL-HF (With human feedback). In the first column, x-axis is the number of training episodes, and y-axis is the max q-values, the optimal action in each state and action. In the second column, x-axis is the number of training episodes, and y-axis is the cumulative rewards, the total reward in each episode. The shaded area shows the standard deviation, the range of max q-values and cumulative rewards achievable. Both TS-QL and TS-QL-HF is shown to have a slower convergence rate after using DQL.

Table 4.4: Average accuracy and F1 score of TS- compared to the baseline. The baseline is the accuracy and F1 score of InceptionV3 model before using QL. The accuracy of TS-QL-HF is shown to be reduced. This shows that DQL is not suitable in a deterministic environment.

	Accuracy	F1 Score
Baseline	0.9849 ± 0.0168	0.9826 ± 0.0199
TS-DQL-HF	0.9914 ± 0.0142	0.9904 ± 0.0164
TS-QL-HF	0.9936 ± 0.0138	0.9925 ± 0.0162
TS-DQL	0.9893 ± 0.0143	0.9882 ± 0.0165
TS-QL	0.9893 ± 0.0143	0.9882 ± 0.0165

TS-DQL is shown to act more conservatively thus slowing down the convergence rate of TS-QL. At the same time, the cumulative reward yielded is also reduced. The accuracy and F1 score of TS-DQL remains the same compared to TS-QL. TS-DQL-HF is also shown to act more conservatively thus slowing down the convergence rate of TS-QL-HF. At the same time, the cumulative reward yielded is also reduced. The accuracy and F1 score of TS-DQL-HF is reduced compared to TS-QL-HF. These observations show that acting conservatively in a deterministic nature can cause a minor performance loss. However, it is worth noting that using DQL can be beneficial in a research scenario where the QL environment is stochastic. This experiment concludes that traditional QL preferred over DQL for this experiment scenario.

4.5 Summary

The chapter has investigated the CNN models and concludes by using InceptionV3 as the base model because it provides the highest accuracy compared to other models thus requires the least human input. Next, this chapter investigates the effect of using human feedback and different action selection strategies and concludes that using human feedback provides performance improvement on the FER system while the action selection strategies do not make a difference. Finally, this chapter investigated the effect of using DQL in the QL approaches and concludes that DQL is suitable for deterministic environment as it is shown decreases the performance of the FER system.

4.6 Recommendations For Future Works

This research recommends the following to improve the TS-QL-HF approach. Firstly, a hybrid version of TS-QL and TS-QL-HF that dynamically learns from environmental rewards and human feedback reward can be explored to dynamically learn from environmental rewards. In real-time FER system, collecting human feedback to improve the application is impractical and expensive. While occasional human feedback collection remains valuable for its higher accuracy enhancement compared to environmental rewards, relying solely on it is not practical. By utilizing the hybrid version, it offers a balanced and practical approach for improving the accuracy of real-time FER system without depending solely on human input.

Next, DQN can be explored in a bigger action space. In a real-time FER system, the images are highly complex thus the misclassification will be increased significantly due to unpredictable scenarios such as occlusion, making it is highly beneficial to increase the action space as the training set increases. There is lack of research on using DQN in a deterministic environment, thus, the effectiveness of DQN in the environment remains unknown.

Finally, the results of the QL can be extended as an intelligent image augmentation. Image augmentation is the process of generating new transformed images that increases the diversity of input images fed to CNN models (*What Is Image Data Augmentation?*, n.d.). However, image augmentation has been arbitrary therefore slowing down the training time of CNN models significantly. This approach has the potential to reduce the amount

of image augmentation needed by only selecting the image augmentation that is shown to improve the accuracy of the CNN models. This approach has a potential to pave a new way in traditional image augmentation.

By exploring these recommendations, this research not only enhances the TS-QL and TS-QL-HF approach for real-time FER systems but also provide valuable insights in a wider field in computer vision and can be widely applicable in various areas such as marketing, medical, and robotics

REFERENCES

- Andalibi, N., & Buss, J. (2020a, April 21). The Human in Emotion Recognition on Social Media: Attitudes, Outcomes, Risks. *Conference on Human Factors in Computing Systems - Proceedings*. <https://doi.org/10.1145/3313831.3376680>
- Andalibi, N., & Buss, J. (2020b, April 21). The Human in Emotion Recognition on Social Media: Attitudes, Outcomes, Risks. *Conference on Human Factors in Computing Systems - Proceedings*. <https://doi.org/10.1145/3313831.3376680>
- baeldung. (2023, March 24). *Epsilon-Greedy Q-learning* / Baeldung on Computer Science. Baeldung. <https://www.baeldung.com/cs/epsilon-greedy-q-learning>
- Berrar, D. (2018). Cross-validation. In *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics* (Vols. 1–3, pp. 542–545). Elsevier. <https://doi.org/10.1016/B978-0-12-809633-8.20349-X>
- Bird, J. J., & Faria, D. R. (2018). *A Study on CNN Transfer Learning for Image Classification Sim2Real: From Simulation to Real Robotic Application using Deep Reinforcement Learning and Knowledge Transfer View project PhD Thesis: A Socially Interactive Multimodal Human-Robot Interaction Framework through Studies on Machine and Deep Learning View project*. <https://www.researchgate.net/publication/325803364>
- Deng, C., Ji, X., Rainey, C., Zhang, J., & Lu, W. (2020). iScience Integrating Machine Learning with Human Knowledge. *ISCIENCE*, 23, 101656. <https://doi.org/10.1016/j.isci>
- FERG - V7 Open Datasets*. (n.d.). Retrieved July 8, 2023, from <https://www.v7labs.com/open-datasets/ferg>
- Hafiz, A. M. (n.d.). *Image Classification by Reinforcement Learning with Two-State Q-Learning*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*. <http://arxiv.org/abs/1512.03385>

- Ilona, T. (2020, November 26). *Deep Reinforcement Learning in emotion recognition*.
https://github.com/ilonatommy/DLR_FacialEmotionRecognition
- Keras Applications*. (n.d.). Retrieved September 12, 2023, from
<https://keras.io/api/applications/>
- Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization*.
<http://arxiv.org/abs/1412.6980>
- Li, Y. (2017). *Deep Reinforcement Learning: An Overview*.
<http://arxiv.org/abs/1701.07274>
- Maroti, A. (2019). *RBED: Reward Based Epsilon Decay*.
<http://arxiv.org/abs/1910.13701>
- Murphy, K. P. (n.d.). *Machine learning : a probabilistic perspective*.
- O'Shea, K., & Nash, R. (2015). *An Introduction to Convolutional Neural Networks*. <http://arxiv.org/abs/1511.08458>
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). *MobileNetV2: Inverted Residuals and Linear Bottlenecks*.
<http://arxiv.org/abs/1801.04381>
- Schaul, T., Quan, J., Antonoglou, I., Silver, D., & Deepmind, G. (n.d.). *PRIORITIZED EXPERIENCE REPLAY*.
- Sutton, R., & Barto, A. (2005). *Reinforcement Learning: An Introduction*.
<http://incompleteideas.net/sutton/book/ebook/the-book.html>
- Szegedy, C., Vanhoucke, V., Ioffe, S., & Shlens, J. (n.d.). *Rethinking the Inception Architecture for Computer Vision*.
- Tijmsma, A. D., Drugan, M. M., & Wiering, M. A. (2017, February 9). Comparing exploration strategies for Q-learning in random stochastic mazes. *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*.
<https://doi.org/10.1109/SSCI.2016.7849366>
- Van Hasselt, H. (n.d.). *Double Q-learning*.
- van Hasselt, H., Guez, A., & Silver, D. (2015). *Deep Reinforcement Learning with Double Q-learning*. <http://arxiv.org/abs/1509.06461>
- Wang, H., & Zheng, H. (2013). Model Validation, Machine Learning. *Encyclopedia of Systems Biology*, 1406–1407.
https://doi.org/10.1007/978-1-4419-9863-7_233

- Wang, Z., Schaul, T., Hessel, M., & Lanctot, M. (2016). *Dueling Network Architectures for Deep Reinforcement Learning* Hado van Hasselt.
<https://www.youtube.com/playlist?list=>
- Watkins, C. J. C. H., & Dayan, P. (1992). *Q-Learning* (Vol. 8).
- What Is Image Data Augmentation?* (n.d.). Retrieved October 1, 2023, from
<https://www.picsellia.com/post/image-data-augmentation>
- Whiteson, S., Taylor, M. E., & Stone, P. (2007). Empirical studies in action selection with reinforcement learning. *Adaptive Behavior*, 15(1), 33–50.
<https://doi.org/10.1177/1059712306076253>

APPENDICES

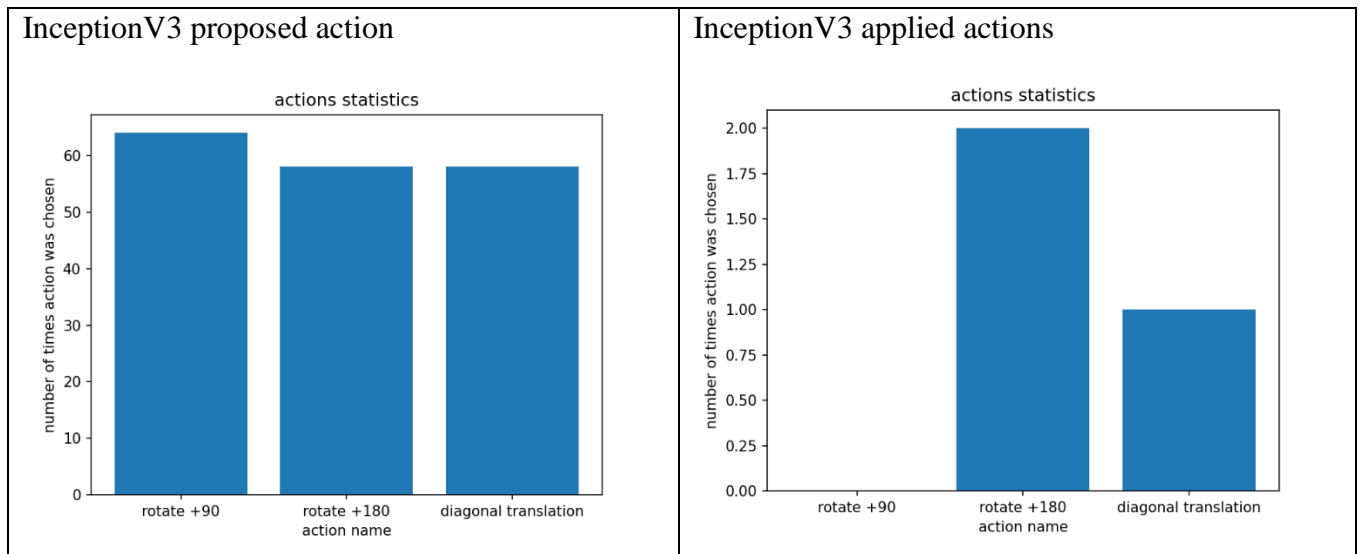
Appendix A: Preliminary Results

A.1 Appendix Overview

The appendices show the preliminary results of TS-QL that leads to the results in the results and discussions chapter. Then, the appendices show the flowchart of TS-DQL-HF.

A.2 Action Selection Stats

Figure A.2 illustrates the number of each proposed actions and selected optimal actions. The previous QL approach uses an arbitrary action selection strategy. In the first column, Figure 4.3 shows the optimal action has the same probability of being selected as the suboptimal action and the worst performing action. This shows that a lot of time has been wasted in exploring suboptimal actions. In the second column, Figure 4.3 shows in InceptionV3 and MobileNetV2, rotate +180 is selected as optimal action, in ResNet50, rotate +90 is selected as optimal action.



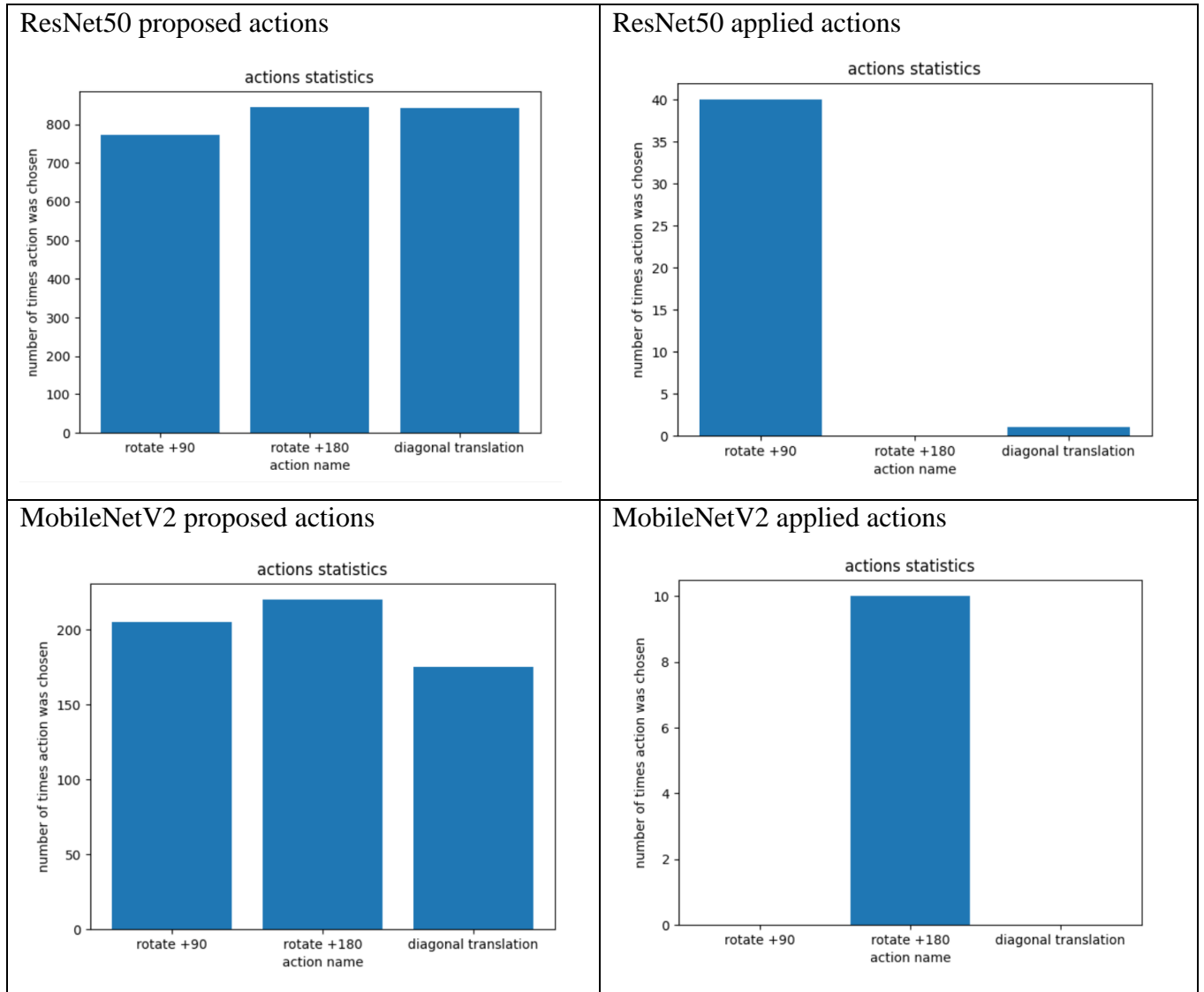
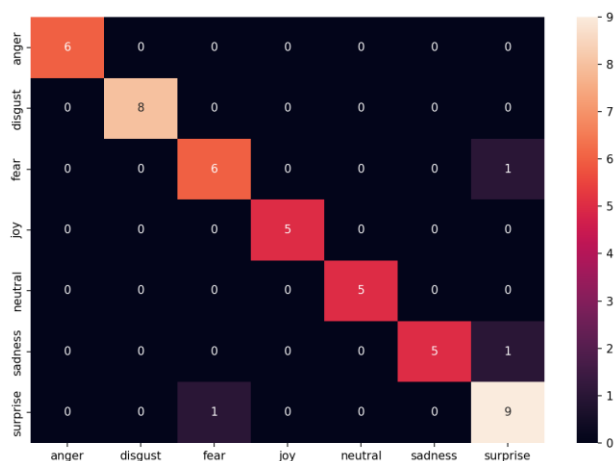


Figure A.2: Action Stats of the Previous QL Approach using InceptionV3, ResNet50, and MobileNetV2.

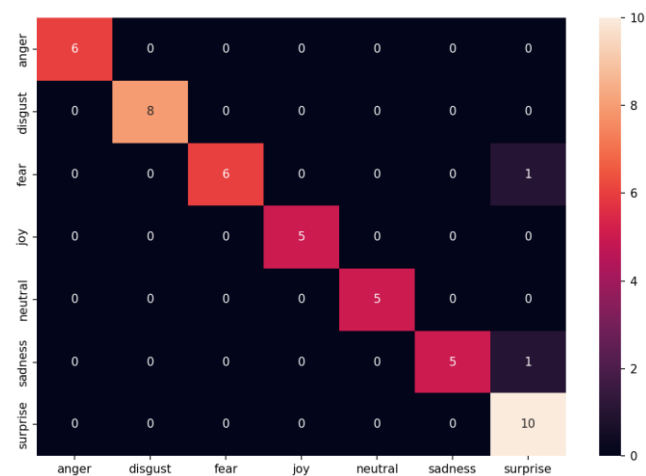
A.3 Confusion Matrix

Figures A.3 illustrate the confusion matrix which tells the number of misclassifications in each emotion label. Figures 4.4 show that, in InceptionV3, the number of misclassified images decreased from 3 to 1. In ResNet50 and MobileNetV2 the number of misclassified images did not change. Further investigation is required for these two models.

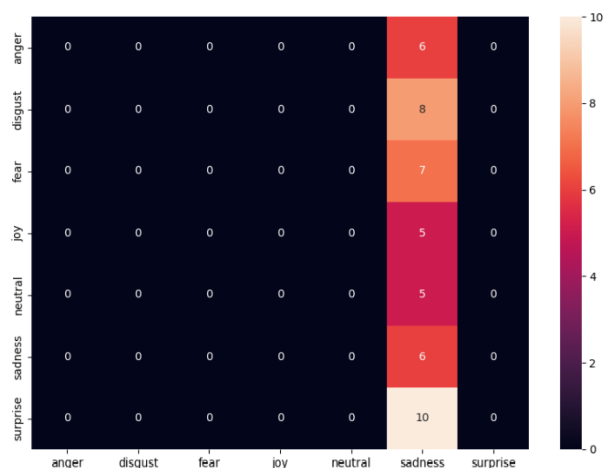
InceptionV3 Before QL



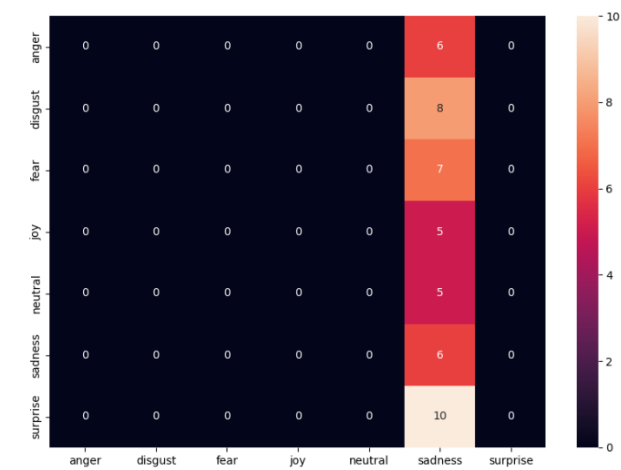
InceptionV3 after QL



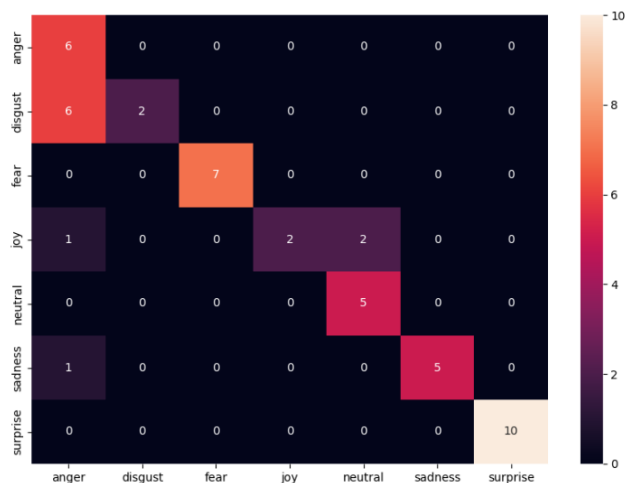
ResNet50 before QL



ResNet50 after QL



MobileNetV2 before QL



MobileNetV2 after QL

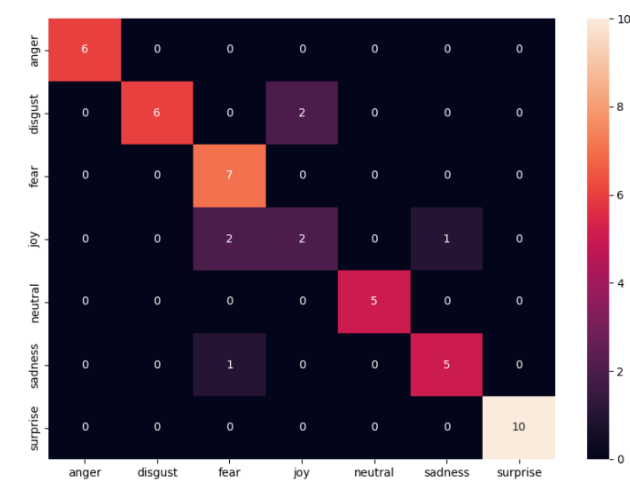


Figure A.3: Confusion matrixes of InceptionV3, ResNet50, and MobileNetV2. X-axis represents the actual emotion labels, and Y-axis represents the predicted emotion labels.

A.4 Report

Figures A.4 shows the recall, precision, F1 score and accuracy. In InceptionV3, figures 4.5 show that the accuracy increased from 0.93 to 0.95 while the F1 score increased from 0.94 to 0.95. In ResNet50, figures 3.5 show that the accuracy and F1 score does not change. In mobileNetV2, the accuracy increased from 0.79 to 0.87 while the F1 score increased from 0.76 to 0.85 despite the learning curve showing continuous divergence throughout the entire learning process. Further investigation is required on ResNet50 and MobileNetV2.

InceptionV3 report before QL					InceptionV3 report after QL				
recall: 0.9414965986394559					recall: 0.9557823129251701				
precision: 0.953617810760668					precision: 0.9591836734693878				
F1 score: 0.9461966604823748					F1 score: 0.9556158127586699				
report:	precision	recall	f1-score	support	report:	precision	recall	f1-score	support
0	1.00	1.00	1.00	6	0	0.86	1.00	0.92	6
1	1.00	1.00	1.00	8	1	1.00	1.00	1.00	8
2	0.86	0.86	0.86	7	2	0.86	0.86	0.86	7
3	1.00	1.00	1.00	5	3	1.00	1.00	1.00	5
4	1.00	1.00	1.00	5	4	1.00	1.00	1.00	5
5	1.00	0.83	0.91	6	5	1.00	0.83	0.91	6
6	0.82	0.90	0.86	10	6	1.00	1.00	1.00	10
accuracy			0.94	47	accuracy			0.96	47
macro avg	0.95	0.94	0.95	47	macro avg	0.96	0.96	0.96	47
weighted avg	0.94	0.94	0.94	47	weighted avg	0.96	0.96	0.96	47
accuracy: 0.9361702127659575					accuracy: 0.9574468085106383				

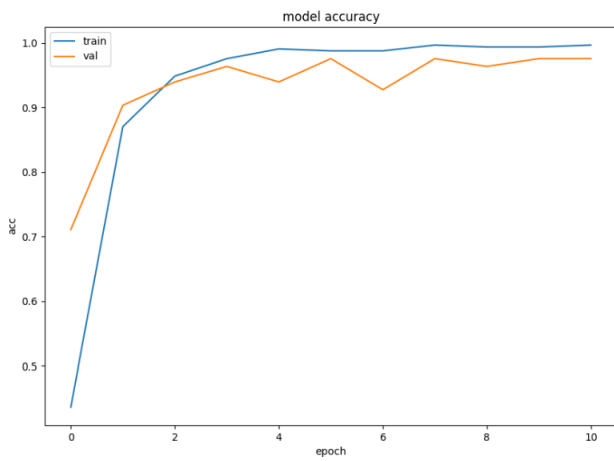
<div>ResNet50 before QL</div> <div><div>recall: 0.14285714285714285</div><div>precision: 0.0182370820668693</div><div>F1 score: 0.032345013477088944</div><table><tr><td>report:</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>0.00</td><td>0.00</td><td>0.00</td><td>6</td></tr><tr><td>1</td><td>0.00</td><td>0.00</td><td>0.00</td><td>8</td></tr><tr><td>2</td><td>0.00</td><td>0.00</td><td>0.00</td><td>7</td></tr><tr><td>3</td><td>0.00</td><td>0.00</td><td>0.00</td><td>5</td></tr><tr><td>4</td><td>0.00</td><td>0.00</td><td>0.00</td><td>5</td></tr><tr><td>5</td><td>0.13</td><td>1.00</td><td>0.23</td><td>6</td></tr><tr><td>6</td><td>0.00</td><td>0.00</td><td>0.00</td><td>10</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.13</td><td>47</td></tr><tr><td>macro avg</td><td>0.02</td><td>0.14</td><td>0.03</td><td>47</td></tr><tr><td>weighted avg</td><td>0.02</td><td>0.13</td><td>0.03</td><td>47</td></tr></table><div>accuracy: 0.1276595744680851</div></div>	report:	precision	recall	f1-score	support	0	0.00	0.00	0.00	6	1	0.00	0.00	0.00	8	2	0.00	0.00	0.00	7	3	0.00	0.00	0.00	5	4	0.00	0.00	0.00	5	5	0.13	1.00	0.23	6	6	0.00	0.00	0.00	10	accuracy			0.13	47	macro avg	0.02	0.14	0.03	47	weighted avg	0.02	0.13	0.03	47	<div>ResNet50 after QL</div> <div><div>recall: 0.14285714285714285</div><div>precision: 0.0182370820668693</div><div>F1 score: 0.032345013477088944</div><table><tr><td>report:</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>0.00</td><td>0.00</td><td>0.00</td><td>6</td></tr><tr><td>1</td><td>0.00</td><td>0.00</td><td>0.00</td><td>8</td></tr><tr><td>2</td><td>0.00</td><td>0.00</td><td>0.00</td><td>7</td></tr><tr><td>3</td><td>0.00</td><td>0.00</td><td>0.00</td><td>5</td></tr><tr><td>4</td><td>0.00</td><td>0.00</td><td>0.00</td><td>5</td></tr><tr><td>5</td><td>0.13</td><td>1.00</td><td>0.23</td><td>6</td></tr><tr><td>6</td><td>0.00</td><td>0.00</td><td>0.00</td><td>10</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.13</td><td>47</td></tr><tr><td>macro avg</td><td>0.02</td><td>0.14</td><td>0.03</td><td>47</td></tr><tr><td>weighted avg</td><td>0.02</td><td>0.13</td><td>0.03</td><td>47</td></tr></table><div>accuracy: 0.1276595744680851</div></div>	report:	precision	recall	f1-score	support	0	0.00	0.00	0.00	6	1	0.00	0.00	0.00	8	2	0.00	0.00	0.00	7	3	0.00	0.00	0.00	5	4	0.00	0.00	0.00	5	5	0.13	1.00	0.23	6	6	0.00	0.00	0.00	10	accuracy			0.13	47	macro avg	0.02	0.14	0.03	47	weighted avg	0.02	0.13	0.03	47
report:	precision	recall	f1-score	support																																																																																																											
0	0.00	0.00	0.00	6																																																																																																											
1	0.00	0.00	0.00	8																																																																																																											
2	0.00	0.00	0.00	7																																																																																																											
3	0.00	0.00	0.00	5																																																																																																											
4	0.00	0.00	0.00	5																																																																																																											
5	0.13	1.00	0.23	6																																																																																																											
6	0.00	0.00	0.00	10																																																																																																											
accuracy			0.13	47																																																																																																											
macro avg	0.02	0.14	0.03	47																																																																																																											
weighted avg	0.02	0.13	0.03	47																																																																																																											
report:	precision	recall	f1-score	support																																																																																																											
0	0.00	0.00	0.00	6																																																																																																											
1	0.00	0.00	0.00	8																																																																																																											
2	0.00	0.00	0.00	7																																																																																																											
3	0.00	0.00	0.00	5																																																																																																											
4	0.00	0.00	0.00	5																																																																																																											
5	0.13	1.00	0.23	6																																																																																																											
6	0.00	0.00	0.00	10																																																																																																											
accuracy			0.13	47																																																																																																											
macro avg	0.02	0.14	0.03	47																																																																																																											
weighted avg	0.02	0.13	0.03	47																																																																																																											
<div>MobileNetV2 before QL</div> <div><div>recall: 0.7833333333333333</div><div>precision: 0.8775510204081634</div><div>F1 score: 0.759121830550402</div><table><tr><td>report:</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>0.43</td><td>1.00</td><td>0.60</td><td>6</td></tr><tr><td>1</td><td>1.00</td><td>0.25</td><td>0.40</td><td>8</td></tr><tr><td>2</td><td>1.00</td><td>1.00</td><td>1.00</td><td>7</td></tr><tr><td>3</td><td>1.00</td><td>0.40</td><td>0.57</td><td>5</td></tr><tr><td>4</td><td>0.71</td><td>1.00</td><td>0.83</td><td>5</td></tr><tr><td>5</td><td>1.00</td><td>0.83</td><td>0.91</td><td>6</td></tr><tr><td>6</td><td>1.00</td><td>1.00</td><td>1.00</td><td>10</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.79</td><td>47</td></tr><tr><td>macro avg</td><td>0.88</td><td>0.78</td><td>0.76</td><td>47</td></tr><tr><td>weighted avg</td><td>0.90</td><td>0.79</td><td>0.77</td><td>47</td></tr></table><div>accuracy: 0.7872340425531915</div></div>	report:	precision	recall	f1-score	support	0	0.43	1.00	0.60	6	1	1.00	0.25	0.40	8	2	1.00	1.00	1.00	7	3	1.00	0.40	0.57	5	4	0.71	1.00	0.83	5	5	1.00	0.83	0.91	6	6	1.00	1.00	1.00	10	accuracy			0.79	47	macro avg	0.88	0.78	0.76	47	weighted avg	0.90	0.79	0.77	47	<div>MobileNetV2 after QL</div> <div><div>recall: 0.8547619047619047</div><div>precision: 0.8619047619047618</div><div>F1 score: 0.8512071495264772</div><table><tr><td>report:</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>1.00</td><td>1.00</td><td>1.00</td><td>6</td></tr><tr><td>1</td><td>1.00</td><td>0.75</td><td>0.86</td><td>8</td></tr><tr><td>2</td><td>0.70</td><td>1.00</td><td>0.82</td><td>7</td></tr><tr><td>3</td><td>0.50</td><td>0.40</td><td>0.44</td><td>5</td></tr><tr><td>4</td><td>1.00</td><td>1.00</td><td>1.00</td><td>5</td></tr><tr><td>5</td><td>0.83</td><td>0.83</td><td>0.83</td><td>6</td></tr><tr><td>6</td><td>1.00</td><td>1.00</td><td>1.00</td><td>10</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.87</td><td>47</td></tr><tr><td>macro avg</td><td>0.86</td><td>0.85</td><td>0.85</td><td>47</td></tr><tr><td>weighted avg</td><td>0.88</td><td>0.87</td><td>0.87</td><td>47</td></tr></table><div>accuracy: 0.8723404255319149</div></div>	report:	precision	recall	f1-score	support	0	1.00	1.00	1.00	6	1	1.00	0.75	0.86	8	2	0.70	1.00	0.82	7	3	0.50	0.40	0.44	5	4	1.00	1.00	1.00	5	5	0.83	0.83	0.83	6	6	1.00	1.00	1.00	10	accuracy			0.87	47	macro avg	0.86	0.85	0.85	47	weighted avg	0.88	0.87	0.87	47
report:	precision	recall	f1-score	support																																																																																																											
0	0.43	1.00	0.60	6																																																																																																											
1	1.00	0.25	0.40	8																																																																																																											
2	1.00	1.00	1.00	7																																																																																																											
3	1.00	0.40	0.57	5																																																																																																											
4	0.71	1.00	0.83	5																																																																																																											
5	1.00	0.83	0.91	6																																																																																																											
6	1.00	1.00	1.00	10																																																																																																											
accuracy			0.79	47																																																																																																											
macro avg	0.88	0.78	0.76	47																																																																																																											
weighted avg	0.90	0.79	0.77	47																																																																																																											
report:	precision	recall	f1-score	support																																																																																																											
0	1.00	1.00	1.00	6																																																																																																											
1	1.00	0.75	0.86	8																																																																																																											
2	0.70	1.00	0.82	7																																																																																																											
3	0.50	0.40	0.44	5																																																																																																											
4	1.00	1.00	1.00	5																																																																																																											
5	0.83	0.83	0.83	6																																																																																																											
6	1.00	1.00	1.00	10																																																																																																											
accuracy			0.87	47																																																																																																											
macro avg	0.86	0.85	0.85	47																																																																																																											
weighted avg	0.88	0.87	0.87	47																																																																																																											

Figure A.4. Report on InceptionV3, ResNet50 and MobileNetV2 on recall, precision, F1 score and accuracy.

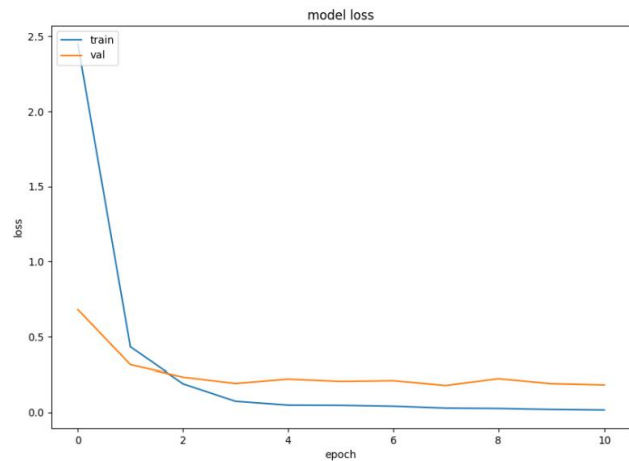
A.5 Training History

Figure A.5 shows that the validation set is not performing as well as the training set in all the model accuracy and model loss graphs. This is because validation data is unseen data that is not used during training. Validation data is used to evaluate the performance of the model (H. Wang & Zheng, 2013).

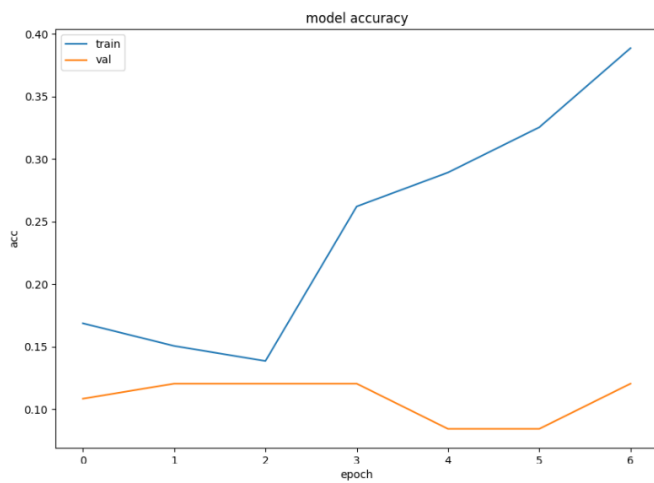
Model accuracy of InceptionV3



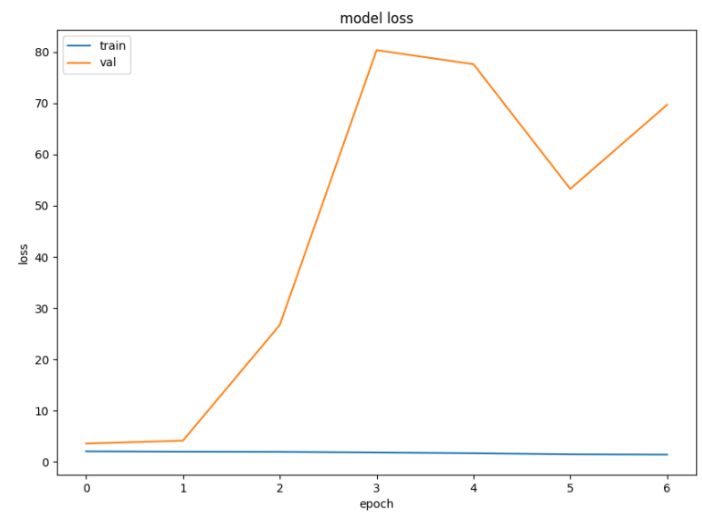
Model loss of InceptionV3



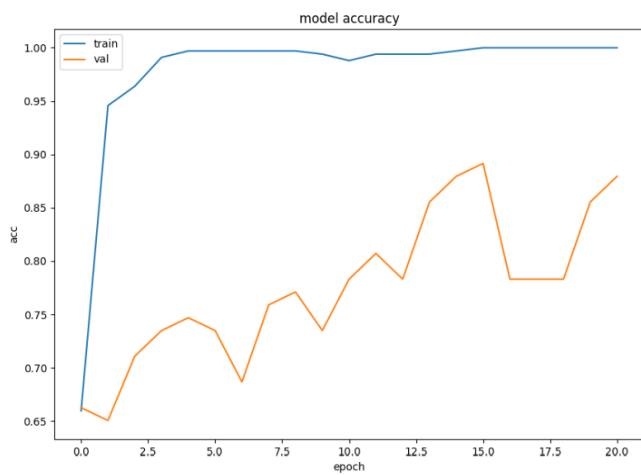
Model accuracy of ResNet50



Model loss of ResNet50



Model accuracy of MobileNetV2



Model loss of MobileNetV2

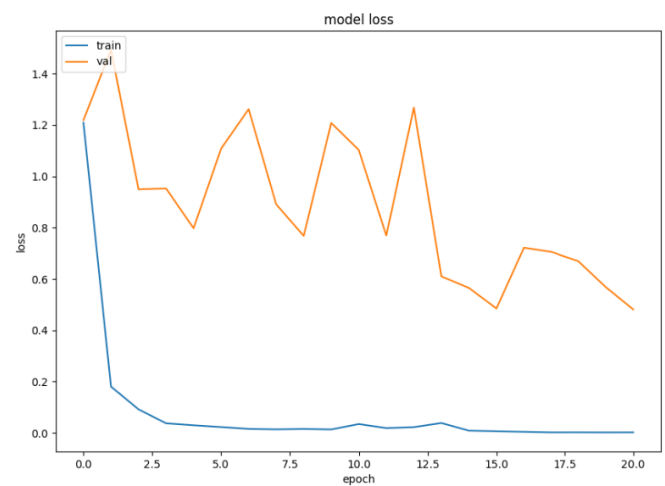


Figure A.5: The model accuracy and model loss of InceptionV3, ResNet50, and MobileNetV2.

Appendix B: Flowchart of TS-DQL-HF

B.1 TS-DQL-HF

Algorithm 8 shows the workflow and detailed steps for the TS-DQL-HF module, which uses the I-DQL submodule. The objective of integrating human inputs into reinforcement learning to recognize misclassified images by CNN is achieved in this module.

The inputs of this module are the trained CNN model, and the testing set. The module involves six processes. The first process performs initial classification on the testing set and obtains initial predictions. The second process performs I-DQL and obtains \bar{Q} of misclassified images, the average of Q^A and Q^B . The Q^A and Q^B values refer to Q-tables A and B. The third process chooses $\text{argmax}_Q(s, a)$ from \bar{Q} as the optimal action. The fourth process applies the optimal action on the misclassified images. The fifth process performs the second classification on the misclassified images and obtains updated predictions. The sixth process replaces misclassified predictions with updated predictions. The output of this module are updated predictions.

This module provides a significant advantage to the FER system. TS-DQL-HF provides different human perspectives, which are new, to the classifier. This enhancement has been shown to provide a minor accuracy as improvement compared to the state-of-the-art InceptionV3 model used in the FER system. Using TS-DQL-HF is suitable for the FER system because a minor improvement in accuracy has significant effects. For instance, accurate classification in image classification tasks like FER.

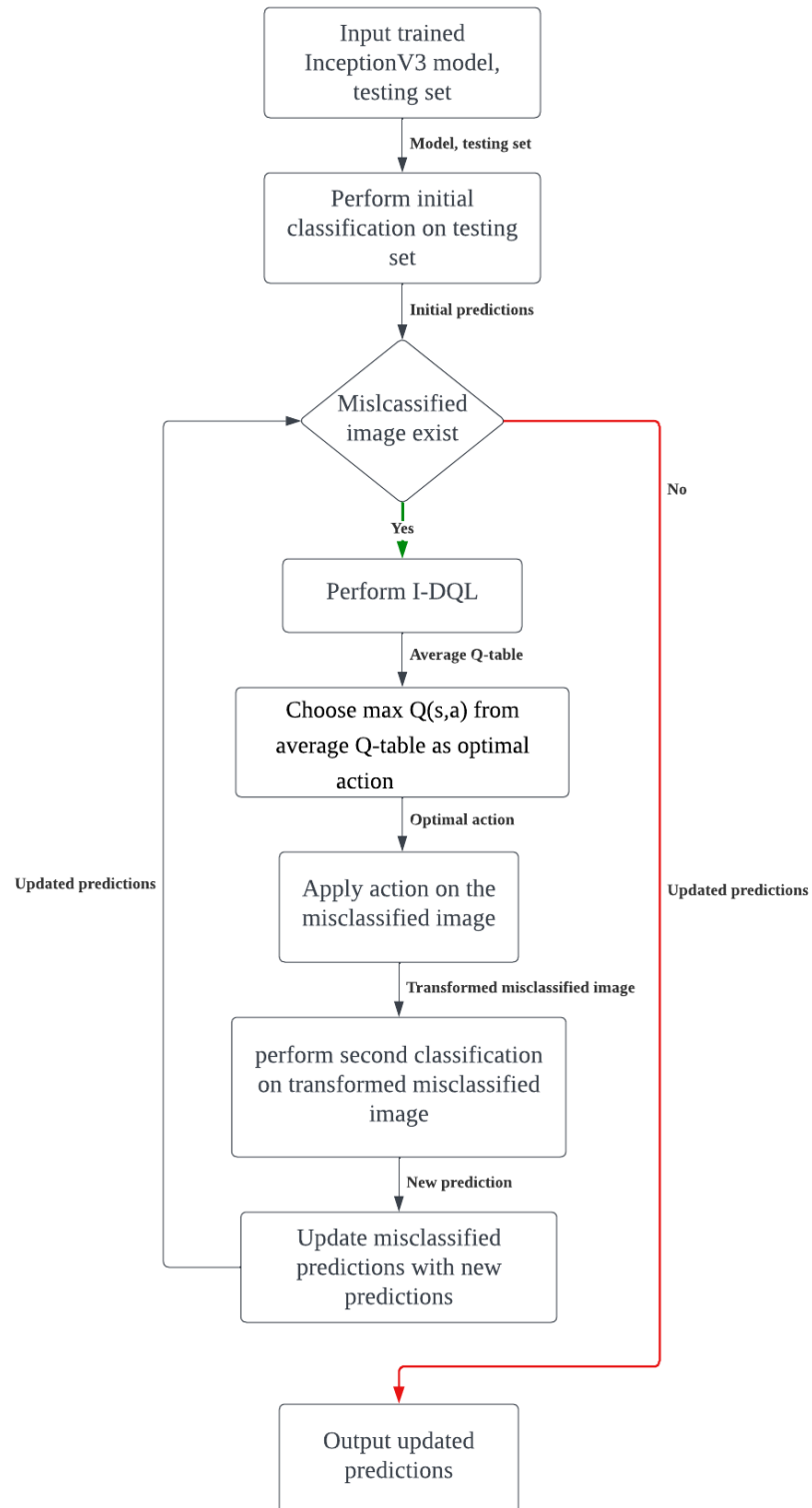


Figure B.1: TS-DQL-HF Flowchart