UNCERTAINTIES IN CONTINUAL LEARNING

by

JUSTIN LO

SEMESTER THESIS (PRACTICAL WORK)

in

COMPUTER VISION AND GEOMETRY GROUP

of

ETH ZURICH

2023

Supervisor:
Hermann Blum
Janis Postels
Prof. Dr. Marc Pollefeys

Contents

Abstract iii

List of Figures iv

List of Tables vi

1 Introduction 1

2 Continual Learning 4

2.1 Catastrophic Forgetting 4

2.2 Continuous Learning Strategies 5

2.2.1 Learning without Forgetting 6

2.2.2 Elastic Weight Consolidation 7

2.2.3 Synaptic Intelligence 8

2.2.4 Experience Replay 8

3 Uncertainty Quantification and Calibration 10

3.1 Uncertainty Quantification Strategies 10

3.1.1 Monte Carlo Dropout 10

3.1.2 Deep Ensembles 11

3.2 Uncertainty Quantification Metrics 12

3.2.1 Uncertainty Decomposition 12

3.2.2 Signal-to-Noise Ratio 13

3.3 Calibration Metrics 14

4 Main Experiments 16

4.1 Dataset details 16

4.1.1 Permuted MNIST 16

4.1.2 DomainNet 17
4.1.3 Other datasets 17
4.2 Experiment details 18

5 Analysis **20**
5.1 Uncertainty transitioning between experiences 20
5.2 Decomposition of uncertainty 22
5.3 Correlations between metrics 22

6 Conclusion and Future Work **28**

Bibliography **29**

A Detailed Experiment Plots **35**

Abstract

Uncertainties in Continual Learning

by

Justin Lo

in

ETH Zurich

In continual learning, deep learning models incrementally learn more classes or tasks over time. However, they need to mitigate the problem of catastrophic forgetting where previously learned knowledge is forgotten. In addition, uncertainty plays a crucial role in many real world applications. Hence, we want the continual learning models to also accurately quantify uncertainty. In this report, we seek to understand various continual learning strategies and uncertainty quantification methods. These are combined by conducting experiments on continual learning baselines to analyse the trends and correlations of accuracy and uncertainty metrics while applying the various continual learning techniques.

List of Figures

2.1 Catastrophic forgetting in domain-incremental learning 5

5.1 Uncertainty decomposition between experiences 21
5.2 Mutual Information of different CL methods 23
5.3 Accuracy using Deep Ensembles for Permuted MNIST and DomainNet 24
5.4 SCE using Deep Ensembles for Permuted MNIST and DomainNet . . . 25

A.1 Accuracy of Permuted MNIST - MCD 35
A.2 Entropy of Permuted MNIST - MCD 36
A.3 MI of Permuted MNIST - MCD 36
A.4 ECE of Permuted MNIST - MCD 37
A.5 SCE of Permuted MNIST - MCD 37
A.6 UCE of Permuted MNIST - MCD 38
A.7 Signal-to-Noise Ratio of Permuted MNIST - MCD 38
A.8 Accuracy of Permuted MNIST - Ensembles 39
A.9 Entropy of Permuted MNIST - Ensembles 39
A.10 MI of Permuted MNIST - Ensembles 40
A.11 ECE of Permuted MNIST - Ensembles 40
A.12 SCE of Permuted MNIST - Ensembles 41
A.13 UCE of Permuted MNIST - Ensembles 41
A.14 Signal-to-Noise Ratio of Permuted MNIST - Ensembles 42
A.15 Accuracy of DomainNet - MCD 42
A.16 Entropy of DomainNet - MCD 43
A.17 MI of DomainNet - MCD 43
A.18 ECE of DomainNet - MCD 44
A.19 SCE of DomainNet - MCD 44

A.20 UCE of DomainNet - MCD 45
A.21 Signal-to-Noise Ratio of DomainNet - MCD 45
A.22 Accuracy of DomainNet - Ensembles 46
A.23 Entropy of DomainNet - Ensembles 46
A.24 MI of DomainNet - Ensembles 47
A.25 ECE of DomainNet - Ensembles 47
A.26 SCE of DomainNet - Ensembles 48
A.27 UCE of DomainNet - Ensembles 48
A.28 Signal-to-Noise Ratio of DomainNet - Ensembles 49

List of Tables

5.1 Proportion of Entropy attributed to Mutual Information 22
5.2 Correlation between Accuracy and Entropy 26
5.3 Correlation between Accuracy and Mutual Information 26
5.4 Correlation between SCE and Mutual Information 27
5.5 Correlation between UCE and SNR 27

Chapter 1

Introduction

State-of-the-art machine learning algorithms focus on obtaining high accuracy on a variety of individual tasks. However, for many applications, training a model for this purpose is insufficient. Firstly, there are situations where there is a continuous data stream. This results in changes in the distribution of the data which we want the system to account for. In these cases, the system should not only learn from current experiences, but also maintain information about previous experiences. Secondly, there are many situations where we also want to understand the reliability of the predictions. We hope that our algorithms will provide us uncertainty estimates of the predictions so that we can make informed decisions or utilise the uncertainty to improve subsequent learning. One application where these two problems are prevalent is in an autonomous vehicle. The machine may be pretrained with a set of capabilities but in new environments, we would want the vehicle to adapt to the new environment while still maintaining its original knowledge. In addition, with the high safety risks, it is important to be aware of the uncertainties to allow more informed decisions.

The first problem of learning a set of tasks in a sequential manner is also known as continual learning. Continual learning is where data arrives in possibly non-independent-identically-distributed manner, yet we want to the system to perform well on the entire set of tasks. This creates a balance of learning the current data and keeping information from previous tasks. If we do not attempt to balance these factors, systems will tend to forget information learnt from earlier experiences, a phenomenon known as "catastrophic forgetting" [18, 37]. In their paper on continual learning, van de Ven and Tolias (2019) [45] classify continual learning into three types

of scenarios, namely task-incremental learning, domain-incremental learning, and class-incremental learning. In this report, we will be focusing domain-incremental learning. For this situation, we want to be able to solve tasks learnt so far, but unlike task-incremental learning, we are not provided information on which task the object is from. In contrast to class-incremental learning, we only need to solve the task that we are provided at test time while not needing to identify the task-ID of our problem. This is in line with our goal where we experience tasks that share a similar structure but have a changing marginal probability distribution of inputs between tasks. Although similar to domain adaptation problems such as transfer learning, the continual learning problem not only focuses on the performance of the new task, but also aims to maintain good performances on the older tasks [15].

The second problem is related to a field of uncertainty quantification and calibration. Although state-of-the-art neural networks have been providing increased accuracy, it has been observed to be produce results that are overconfident [11, 21]. This could result in users making uninformed decisions which could lead to dire consequences. In an ideal situation, if a system is not able to confidently provide a prediction from one source, it should then be able to adjust to rely on other sources to perform a more reliable action. Providing confidence intervals allows users to better interpret a model and is becoming increasingly important in various applications [7, 32]. To evaluate the effectiveness of uncertainty estimations, there have been several works that have proposed calibration metrics and corresponding methods to improve these metrics [11, 22, 50].

Even though these two issues are important and have been individually investigated, there has been limited work exploring the combination of both problems. There has been work that view the continual learning problem using a Bayesian perspective by maintaining a distribution over the model parameters and recursively calculating a posterior distribution to update the predictive distributions. To maintain tractable calculations, variational concepts are utilised to obtain variational continual learning [5, 35, 42]. These methods utilise the uncertainty that exists within models, as well as some supplementing techniques to improve the likelihood estimates, to alleviate the problem of catastrophic forgetting within continual learning. However, the works still focus on the accuracy of the individual tasks and only look at this metric within the experiments. There is little evaluation of the

behaviour of uncertainty of these predictions and leaves a gap in understanding this crucial component of the continual learning setting. Hence, this report seeks to investigate the continual learning setting and the behaviours of various methods in terms of their accuracy, uncertainty estimation and calibrations. The report seeks to investigate the correlations between various metrics among various continual learning methods to better understand the behaviours of these methods and how uncertainty within continual learning can be classified.

In Chapter 2 and Chapter 3, the report dives into the concepts and relevant previous works within continual learning and uncertainty quantification respectively. These chapters introduce the methods that are used in conducting the experiments within the report. Chapter 4 introduces the benchmark datasets that are utilised and the experiment setup for conducting the experiments while Chapter 5 shares the experiment results and provides an analysis of various phenomena observed. Lastly, Chapter 6 summarises the main results and discusses possible future research directions based on the observations within the report.

Chapter 2

Continual Learning

2.1 Catastrophic Forgetting

Catastrophic forgetting occurs when models forget past information when learning from new experiences. This issue has been widely researched in the past [10, 31, 44] and methods have been developed to mitigate the effects of it. Typically, catastrophic forgetting is characterised by the loss of test accuracy observed while learning a future experience compared to the accuracy that was observed while learning the experience. Similar to [3], we define the accuracy $a_{k,j}$ to be the accuracy of the test-set of the j-th task after training the network sequentially from tasks 1 to k. In that case, forgetting can be defined as the difference between the maximum accuracy and the current accuracy of the model. This provides an estimate of how much knowledge the model has forgotten about the previous tasks given the current state. The forgetting for the j-th task after the model has been trained up till the k-th task, where $k > j$ is defined as

$$f_j^k = \max_{l \in 1,\cdots,k-1} a_{l,j} - a_{k,j}$$

In this case, a higher value of f_j^k would imply that the model has forgotten a higher level of knowledge previously learnt. A classical example of catastrophic forgetting is demonstrated in Figure 2.1, where we can see the accuracy of the experience increases when the system learns from the current training experience, but slowly decreases overtime when future training experiences are learnt. This decrease represents forgetting and demonstrates the need for continual learning techniques.

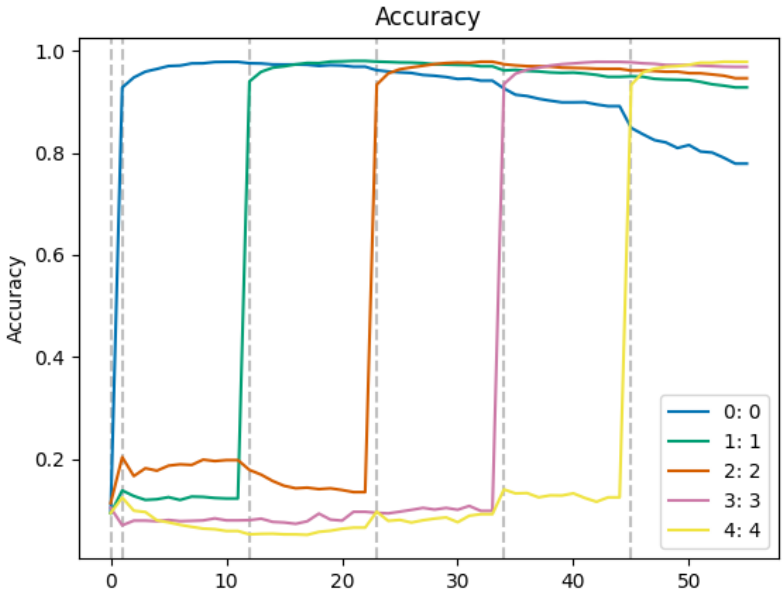


Figure 2.1: Catastrophic forgetting in domain-incremental learning. The dotted lines represent the beginning of various training experiences. As evidenced from the blue curve, the accuracy of experience 0 increases when it is trained, but subsequently decreases when other experiences are trained. This decrease denotes the catastrophic forgetting that has occurred overtime.

2.2 Continuous Learning Strategies

In continual learning, there have been various strategies proposed to mitigate the problem of catastrophic forgetting. In this report, we group the strategies into three categories based on the characterisations provided by [29], [37] and [47]. Thereafter, we experiment with methods from each category to provide a comparison of the different strategies.

The first approach is the use of regularization to impose constraints on the neural network weight updates. This method directly targets the plasticity-stability dilemma where the loss of the network is adjusted to balance between the plasticity, which enables it to learn from the new experience, and network stability, which ensures the preservation of past knowledge. Within this set of approaches, they can be split into two forms of regularization [49]. On one hand, functional approaches penalize changes in the input-to-output mapping of the neural network. These encourage the output using the previous neural network to be similar to the output

by the updated neural network. We investigate one such method known as Learning Without Forgetting (§ 2.2.1). The other form of regularisation is labelled as structural regularisation. These methods include a regularisation penalty to the loss which dissuades the change of parameters. This report explores this class of regularisation with Elastic Weight Consolidation (§ 2.2.2) and Synaptic Intelligence (§ 2.2.3).

The second approach can be classified as replay-based approaches (§ 2.2.4). These methods use a memory buffer to store a subset of previous data to mitigate catastrophic forgetting. These data will then be combined with current data while training new experiences to ensure that the information from past experiences is not lost.

The final group consists of expansion-based methods which make use of architectural changes to account for the new information in current tasks. During the project, we have explored the use of Copy Weights with Reinit (CWR) [26, 29] which extends the output layer and makes use of a set of temporary weights for training, and a set of consolidated weights for inference. However, it was determined that the method was not appropriate and meaningful for the datasets investigated and hence the results are not discussed within this report.

It is noteworthy that these methods are not mutually exclusive, and it is possible to combine various methods within the same continual learning experience. For example, we could combine a replay-based strategy while still applying regularization to the loss.

2.2.1 Learning without Forgetting

Learning without Forgetting (LwF) was proposed by Li and Hoiem [24] as a regularisation approach that focuses on regularizing the output. The model is split into a multi-head problem where there are separate task-specific parameters for each task. Hence, when learning a new task n, the network consists of a set of shared parameters, θ_s, task-specific parameters from the old tasks, θ_0, and a newly randomly initialized set of parameters, θ_n. The loss term is split into three terms as follows:

$$\lambda_0 L_{old}(Y_0, \hat{Y}_0) + L_{new}(Y_n, \hat{Y}_n) + R(\hat{\theta}_s, \hat{\theta}_0, \hat{\theta}_n) \tag{2.1}$$

The first term represents the Knowledge Distillation loss which encourages the outputs of one network to approximate the output of another. In this case, we record the output of old tasks for the new data, Y_0, and try to minimize the loss with the outputs using the new network, $\hat{Y}_0$. In this way, the algorithm attempts to preserve the stability of the model. The second loss term is the regular cross-entropy loss and accounts for the loss of the current experience being learnt. λ_0 is then used as a hyperparameter to balance this plasticity-stability dilemma of the first two terms. The last term is a weight regularization for the stochastic gradient descent process to prevent overfitting.

Learning without Forgetting is a simple process and has been proven to work well in certain situations, but with the downside of requiring a linear increase computational cost with respect to the number of learned tasks, as well as additional memory constraint of storing the predictions of the new data using the old network (Y_0) [29].

2.2.2 Elastic Weight Consolidation

Elastic Weight Consolidation (EWC) was developed by Kirkpatrick et al. in 2017 [18] and follows the form of $L_{total} = L_{current} + \lambda L_{reg}$. Unlike LwF which regularizes the output, EWC penalises the change of network weights which are deemed important in evaluating old tasks. This method is motivated by a Bayesian perspective to maximise the posterior probability $\log p(\theta|D)$ using the following decomposition:

$$\log p(\theta|D) = \log p(D_{new}|\theta) + \log p(\theta|D_{old}) - \log p(D_{new}) \tag{2.2}$$

In the above equation, $\log p(\theta|D_{old})$ represents the information about old tasks. However, the term is intractable and hence is approximated using a Gaussian distribution centered about the parameters of the new tasks, θ^*_A and precision matrix given by the diagonal values of the Fisher information matrix, F.

Using this approximation, the loss of EWC is then formulated as the following:

$$L(\theta) = L_{new}(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{old,i})^2 \tag{2.3}$$

This forms the loss of the current tasks, regularized using a quadratic penalty on the importance of each parameter i. In this way, the parameters will not be regularised

equally. Instead, weights that are more important for old tasks will be less flexible and not be adjusted as much during the gradient descent process, while other weights which affect previous tasks less can be adjusted more easily.

2.2.3 Synaptic Intelligence

Synaptic Intelligence (SI) [49] follows a similar approach to EWC, but uses a different metric to determine the importance of each parameter. Instead of using the Fisher Information matrix, the authors argue for the use of a more computationally efficient calculation that approximates the importance of a parameter by the contribution it has to the change in the total loss. The weight importance of parameter k, ω_k, is calculated as a function of the sum over the weight trajectories in previous tasks.

Overall, this method uses a quadratic surrogate loss, L_n^* for the n-th task as follows:

$$L_n^* = L_n + c\sum_k \omega_k^n(\theta_k^* - \theta_k)^2 \tag{2.4}$$

where c acts as the hyperparameter that controls the plasticity-stability balance and θ_k^* are the weights at the end of the previous task. The main advantage of this approach over EWC is that the weight importance ω_k can be calculated during the stochastic gradient descent process and no additional computations are required.

2.2.4 Experience Replay

Experience replay methods, also known as rehearsal methods, utilise a replay buffer to store a portion of previous data. This set of strategies has been well researched within reinforcement learning [6] to aid off-policy algorithms. In continual learning, a replay buffer is used to construct mini batches of data during training that are composed of new and old data. This mitigates the problem of catastrophic forgetting by having information of old tasks stored within the buffer and replayed during newer tasks. The disadvantage of this method is the memory overhead that is required in storing previous experiences within the buffer. However, Rolnick et al. (2019) [39] has shown that even a small buffer size is sufficient to strongly mitigate the detriments of catastrophic forgetting.

There have been extensive research to adapt rehearsal strategies to optimize which experiences should be stored or pseudo-rehearsal strategies which utilise generative methods to reduce the need to store past data [16, 28, 40]. For simplicity, this report explores the naïve rehearsal strategy, which randomly stores a limited number of past experiences in the memory buffer and replays it in future tasks. However, more extensive research could be conducted in the future on other experience replay methods to analyse their impact on uncertainty estimations.

Chapter 3

Uncertainty Quantification and Calibration

3.1 Uncertainty Quantification Strategies

Uncertainty quantification is crucial for users to know how much trust to put in predictions provided by a system. This is especially important in situations where a wrong prediction can result in major detrimental effects [20]. In active learning or reinforcement learning contexts, knowledge of uncertainty bounds is also to help an agent balance the exploration-exploitation dilemma [8]. There have been a wide variety of uncertainty quantification methods that have been researched, including various complex methods that modify the neural network architectures [1, 5, 33, 46]. Although effective, these methods are difficult to implement and are computationally expensive. In this report, we aim to explore uncertainty quantification strategies that are effective yet simple to implement. Two such methods are Monte Carlo Dropout (§ 3.1.1) and the use of Deep Ensembles (§ 3.1.2).

3.1.1 Monte Carlo Dropout

Dropout was initially proposed by Hinton et al. (2012) [13] as a way to regularise neural networks and prevent overfitting by prevent co-adaptations of features. By applying dropout, each hidden unit has a chance of being omitted from the network. With different hidden units being removed during train time, a variety of networks can be trained. At test time, a mean network is used where all hidden units are activated, hence functioning as an averaging over a large number of networks.

Monte Carlo Dropout (MC Dropout) utilises this concept to perform variational inference in complex neural networks [8, 9, 17]. The goal of approximate variational inference is to find some distribution q that minimises the Kullback-Leibler (KL) divergence between itself with the true posterior. With dropout, the approximating distribution, q, can be viewed as a Bernoulli distribution where a weight is set to 0 with probability p, and θ with probability $1 - p$. To calculate the predictive uncertainty,

$$p(y^*|x^*, D_{train}) = \int p(y^*|x^*, \omega)p(\omega|D_{train})d\omega \tag{3.1}$$

$$\approx \int p(y^*|x^*, \omega)q(\omega)d\omega \tag{3.2}$$

$$\approx \frac{1}{M}\sum_{m=1}^{M} p(y^*|x^*, \hat{\omega_m}) \tag{3.3}$$

where $\hat{\omega_m} \sim q(\omega)$ and corresponds to a neural network with weights given by θ as described earlier.

This means that by performing dropout during prediction, we can accumulate the results of various stochastic forward passes through the model and calculate various uncertainty metrics. In the context of classification problems, this can be further represented as

$$p(y^*|x^*, D_{train}) = \frac{1}{M}\sum_{m=1}^{M} Softmax(f^{\hat{\omega}_m}(x^*)) \tag{3.4}$$

where $f^{\hat{\omega}_m}$ represents a network using the dropout distribution $q(\omega)$.

The main advantage of MC Dropout is the simplicity of implementation and inexpensive computation since it only involves forward passes equivalent to the normal dropout process.

3.1.2 Deep Ensembles

Deep ensembles was introduced by Heskes [12] as a bootstrap method, where each model is trained on different subsets of the training data. It is proposed that n_{run} networks should be executed to determine the mean and variance of the bootstrap. The bootstrapping process is useful for algorithms such as convex optimization where there is no intrinsic randomization in the training process. However, it has been observed that for neural networks, this is not needed and is instead detrimental to

the learning process. Neural networks have many local optima and rely on large number of data points to learn the concept, hence performing multiple runs of the neural network with different initializations is sufficient to create an ensemble of results [21]. The ensemble forms a uniformly weighted mixture model and in the context of the classification problem, the predicted probabilities can be averaged to gather a final prediction for the model.

3.2 Uncertainty Quantification Metrics

3.2.1 Uncertainty Decomposition

Uncertainty can be split into epistemic and aleatoric uncertainty. Epistemic uncertainty, also known as model uncertainty, refers to the uncertainty of the model parameters and measures how uncertain we are about the model that generated the data. For example, the presence of noisy data or lack of training data could affect a model's ability to learn an accurate representation. This can be solved by having more data or a more accurate network, and hence is known as the reducible part of uncertainty. On the other hand, aleatoric uncertainty, otherwise referred to as data uncertainty, refers to the noise from the data due to the randomness of the data generation process or noise that may exist from measurement machines. For example, aleatoric uncertainty could arise from flipping a coin, where even the best model will not be able to provide an assured answer and hence result in variability in predictions. This is generally considered an irreducible portion of uncertainty. There are many works that have explored this decomposition and proposed methods to separate and quantify epistemic and aleatoric uncertainty [12, 19, 33]. This report explores an information-theoretic approach derived from the Shannon entropy and has been discussed by several works [4, 9, 17, 41].

To estimate uncertainty, a possible measure that can be used is the entropy of the predictive distribution. In the classification setting, where we have discrete outcomes of classes $1, \dots c$, this can be represented as

$$H[p(y^*|x^*)] = -\sum_{c=1}^{C} p(y^* = c|x^*) \log p(y^* = c|x^*) \tag{3.5}$$

However, this represents the total uncertainty, and does not distinguish between the epistemic and aleatoric uncertainty. In some cases, it may be useful to have this information so that we understand areas that the model may be poorly expressed, compared to areas where it may simply contain noisy results. Therefore, we investigate the relationship between entropy and mutual information to aid in this decomposition.

$$I(\omega; y^*|x^*) = H[p(y^*|x^*)] - E_{p(\omega)}H[p(y^*|x^*, \omega)] \tag{3.6}$$

In Equation 3.6, we represent the conditional mutual information I between the weights ω and the predictions, y^*, given an input x^*. Equivalently, this represents the information gained about the model parameters if we were to receive a label for a new point. If we are certain about a prediction, it implies that majority of the variability of the output comes from the input. Therefore, we would expect the information gain to be low. Conversely, if there is high mutual information between the model parameters and conditional output, it means that there is high uncertainty, and we will gain information from knowing the data label.

Using this decomposition, the mutual information represents the model's epistemic uncertainty. The first term of Equation 3.6 is the entropy, as discussed in Equation 3.5. Hence, the second term represents the aleatoric uncertainty of the model. Using the approximations provided by uncertainty quantification methods as described in § 3.1, we are able to calculate the predictive distribution, entropy and mutual information.

3.2.2 Signal-to-Noise Ratio

As an alternative method to quantify uncertainty, we propose to use signal-to-noise ratio (SNR). This metric is commonly used in communication and audio systems and represents how much data is being transmitted. There have been a variety of works that utilise SNR [2, 14, 30, 48] for various purposes. In this report,

we define SNR a given data point i as

$$\bar{x_{i,c}} = \sum_{m=1}^{M} [Softmax(f^m(x))]_c \tag{3.7}$$

$$SNR_{i,c} = \frac{\bar{x_{i,c}}}{Variance(f(x_{i,c}))} \tag{3.8}$$

$$SNR_i = \sum_{c=1}^{C} \vec{SNR}_{i,c} \tag{3.9}$$

where m represents a network derived from the uncertainty quantification process, the variance is taken over the M networks, and c is a class within the C-classes classification problem. The SNR can then be aggregated by taking the average over all samples.

3.3 Calibration Metrics

Besides evaluating a model's uncertainty, we want to measure how reliable a model's confidence is. In an ideal situation, we want the predicting probability to match the true likelihood of correctness. Formally, perfect calibration is defined by Guo et al. (2017) [11] as

$$P(\hat{Y} = c|\hat{P} = p) = p \tag{3.10}$$

for all probability values $p \in [0, 1]$ and class labels $c \in C$. Intuitively, this means that if the model predicts a class c with a probability of 0.7, the model should be correct 70% of the time.

To quantify the gap between the confidence and accuracy, the use of Expected Calibration Error (ECE) was proposed. Since the probability space is continuous, there is a need to split the predictions into various bins for calibration. As proposed within [11] and used within this report's implementations, the bins are split into equally spaced subintervals and the maximal probability value for each data point is used. The ECE can then be calculated as follows:

$$ECE = \sum_{b=1}^{B} \frac{n_b}{N} |acc(b) - conf(b)| \tag{3.11}$$

where $b = 1 \dots B$ represent the bins, n_b is the number of predictions within a specific bin b and N is the total number of data points. The accuracy (acc) is calculated as

the percentage of correct predictions within a bin b, while the confidence ($conf$) is calculated as the average probability of the predicted output allocated to the bin b.

One limitation of ECE is that it only uses the confidence of the class corresponding to the maximal probability value. By utilising the maximal value, the metric does not account for the calibration of the remaining classes. When Naeini et al. (2015) first introduced ECE [34], the problem was a binary setting and hence the metric accurately captured the calibration. However, the datasets used within this report are multi-class problems and hence ECE is less representative of the calibration error. To alleviate this problem, Nixon et al. (2020) [36] proposed the use of Static Calibration Error (SCE). SCE further splits the computations for class as seen in Equation 3.12 such that the accuracy and confidence values are calculated for the n_{bc} data points for each class label c within bin b.

$$SCE = \frac{1}{C}\sum_{c=1}^{C}\sum_{b=1}^{B}\frac{n_{bc}}{N}|acc(b,c) - conf(b,c)| \tag{3.12}$$

Another proposed calibration metric was proposed by Laves et al. (2020) [22] which aims to quantify perfect calibration of uncertainty. Instead of comparing accuracy to the confidence, the Uncertainty Calibration Error (UCE) is quantified by the difference between the error and uncertainty (Equation 3.15).

$$err(b) = 1 - acc(b) \tag{3.13}$$

$$uncer(b) = \frac{1}{n_b}\sum_{i\in b} H(p_i) \tag{3.14}$$

$$UCE = \sum_{b=1}^{B}\frac{n_b}{N}|err(b) - uncer(b)| \tag{3.15}$$

Similar to the solution proposed for SCE, we also applied a class-based averaging for UCE to account for the multi-class setting.

Chapter 4

Main Experiments

4.1 Dataset details

In this report, we aim to investigate the uncertainties off various continual learning methods, with a focus on domain-incremental learning. As such, a variety of suitable datasets were selected for experimentation. Ultimately, we utilised the Permuted MNIST dataset (§ 4.1.1) and DomainNet dataset (§ 4.1.2) to report the results in Chapter 5. However, we also experimented with the PACS and CLEAR dataset, which are briefly discussed in § 4.1.3.

4.1.1 Permuted MNIST

The Permuted MNIST dataset was first introduced in [10] to show the effects of catastrophic forgetting. It is an adaptation of the MNIST dataset, which involves a ten-digit classification of hand-written digits. In the Permuted MNIST variant, the pixels of the original images are permuted randomly. Within each experience, the pixels share the same permutation configuration, whereas between each experience, there is a different permutation used.

In this report, we utilise Permuted MNIST with five experiences, corresponding to five different permutations of the original MNIST dataset. These experiences are trained sequentially, representing five different domains in a continual learning setting.

For all experiments, we used a Multilayer Perceptron with two hidden layers of 256 and 128 hidden units respectively. Each layer was followed by a ReLU activation

and a dropout of probability $p = 0.5$ was included in the model architecture. Each experience had different permutations of the same 60,000 train images and were tested on the respective permutations of 10,000 test images.

4.1.2 DomainNet

DomainNet is a dataset introduced by Peng et al. (2019) [38] to address unsupervised domain adaptation. This represents a more realistic scenario compared to Permuted MNIST and contains six different domains, namely (1) clipart: a collection of clipart images, (2) infograph: infographic images with a specific object, (3) painting: artistic depictions of the objects, (4) quickdraw: drawings by players who played the game "Quick Draw", (5) real: real-world images and photos and (6) sketch: sketches of the objects. The full dataset consists of nearly 570,000 images from 345 classes. However, to allow more manageable training of the dataset, the 50 classes with the most images were selected, resulting in 115,083 images across the six domains. Example classes include feathers, penguins, swords, and windmills. The dataset was split into a 70-30 train-test split within each domain to facilitate the experiments.

As a more extensive dataset, we required a complex neural network architecture to handle the intricacies. In the experiments, we utilised EfficientNet-B0 which aims to optimize neural network performance by balancing network depth, width and resolution [43]. The main benefit of EfficientNet is the smaller network and faster computations when employing the network architecture. To enable Monte Carlo Dropout for uncertainty quantification, the networks were modified to include dropout layers between the convolution layers with a dropout probability of 0.1. To prevent external factors from affecting uncertainty estimation analysis, no pretraining was used for the network. Standard augmentation methods were applied to the images to aid neural network training.

4.1.3 Other datasets

In addition to the two datasets, investigations were also conducted with the PACS [23] and CLEAR [25] datasets, both of which are focused on domain adaptation. The datasets were also trained using the EfficientNet-B0 architecture as described

in § 4.1.2.

PACS features four domains, namely photo, art paintings, cartoons, and sketches. Although the dataset was suitable for our desired purpose, the PACS dataset featured only 9991 images across all domains. This resulted in the training process being unstable and the models being unable to produce good results.

The CLEAR dataset features images sorted over a time period from 2004 to 2014. These 11 years represent the shift in domain and hence 11 sequential experiences for the continual learning problem. The dataset was much larger in size, mitigating the problem that was experienced in the PACS dataset. However, the domains were extremely similar; for example, a cosplayer was still a human regardless of year, and a DSLR camera still shared similar features across the time period. Therefore, there was little evidence of forgetting and the problem did not suit the goals of the report.

4.2 Experiment details

We utilised Avalanche [27] as a Python library to assist with the experiments. The library is aimed towards Continual Learning research and was used as the basis of implementation for various continual learning techniques in our experiments.

For each training configuration, five separate models were trained to form the deep ensemble (§ 3.1.2). One of the models were then selected to perform Monte Carlo Dropout (§ 3.1.1) where 10 stochastic forward passes were used. To compare various continual learning strategies, we applied the following methods: (1) naïve baseline, where no continual learning was applied, (2) Learning without Forgetting (§ 2.2.1), (3) Elastic Weight Consolidation (§ 2.2.2), (4) Synaptic Intelligence (§ 2.2.3), (5) Replay using 5% of the experience dataset size(§ 2.2.4) and (6) Replay using 15% of the experience dataset size. Using each continual learning strategy and uncertainty quantification method, we tracked the loss, accuracy, uncertainty-related metrics (as described in § 3.2.1), signal-to-noise ratio (§ 3.2.2) and calibration-related metrics (§ 3.3). These metrics were calculated after periodic epochs for each test experience, allowing analysis of the trends and correlations between the metrics.

To evaluate the correlations, only the metrics of test experiences that have been trained on were considered (if the model has been trained on experiences 0 and 1, and is currently training on experience 2, we would only consider the metrics of the

test data from experiences 0, 1 and 2. The test data from the other experiences will then be not considered . For all the evaluated test experiences, the two chosen metrics (X and Y) are then combined into a single collection and the Pearson correlation coefficient (Equation 4.1) is then calculated.

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{4.1}$$

The Pearson correlation coefficient is measured in the range of -1 to 1, where a value of -1 signifies total negative linear correlation, 0 being no linear correlation, and 1 signifying total positive linear correlation. To check for possible non-linear relationships, the use of the Spearman's rank correlation coefficient was also experimented with. However, this generally weakened the correlation strengths and hence were omitted in the final results.

Chapter 5

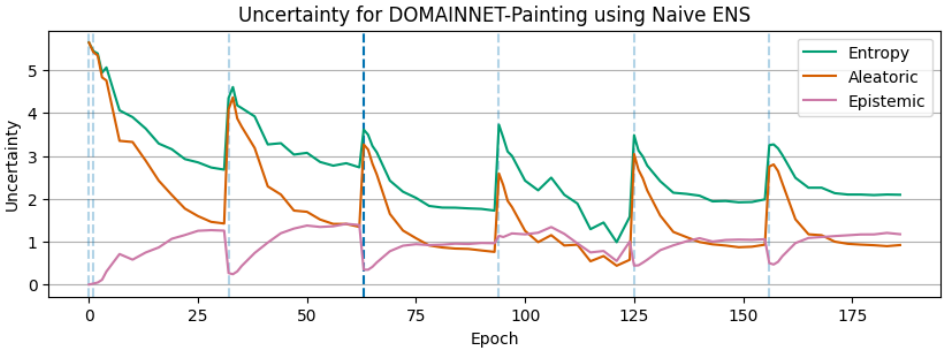
Analysis

In this chapter, we discuss a summary of the key results that were analysed during this project. A more detailed display of the results for the various methods and metrics is listed in Appendix A.

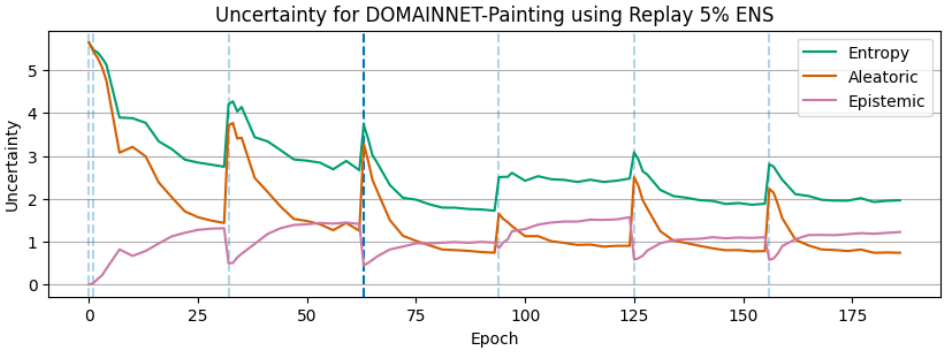
5.1 Uncertainty transitioning between experiences

While performing experiments on the various datasets, one interesting observation was the change in uncertainty overtime. When analysing the real-world datasets, we noticed that there was a sharp change in uncertainty when a new experience was encountered. As seen in Figure 5.1, whenever there is a new experience, the entropy expectedly increases. However, this increase in entropy is largely attributed to aleatoric uncertainty. It is observed that the aleatoric uncertainty increases while the epistemic uncertainty decreases. A possible explanation for this phenomenon is that when the model is first learning a new experience, it views the new information as noise, while becoming confident of the 'noise model' that it is predicting. Overtime, the model learns from the new experience and learns a network that can attribute the uncertainty towards epistemic uncertainty instead.

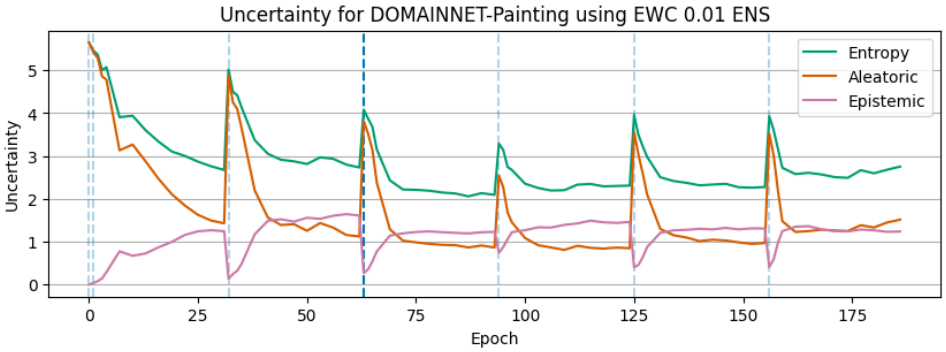
This trend is noticed across both uncertainty quantification methods (Monte Carlo Dropout and Deep Ensembles), as well as various continual learning techniques applied. Due to the instability in the initial training of each experience, this motivated us to conduct future analysis by splitting between (A) epochs early within an experience and (B) epochs in the later epochs of the experience.



(a) Uncertainty without continual learning method



(b) Uncertainty using Replay



(c) Uncertainty using EWC

Figure 5.1: Uncertainty decomposition of the test dataset containing the painting domain within DomainNet. The dotted blue lines represent the transition to learning each new domain (experience), and the dotted dark blue line represents when the painting domain is being learnt

5.2 Decomposition of uncertainty

A key portion of the project was to understand the decomposition of uncertainty. To gain better insight into how epistemic and aleatoric uncertainty is split, we calculated the entropy and mutual information at the various epochs. Then, the proportion of mutual information (epistemic uncertainty) compared to the entropy (total uncertainty) was calculated. These values were calculated for the test experiences after the respective experiences have been trained.

Method	PMNIST-MCD	PMNIST-ENS	DOMNET-MCD	DOMNET-ENS
EWC	0.6871	0.3199	0.4646	0.5404
Replay 5%	0.4391	0.2087	0.3826	0.6015
Replay 15%	0.4205	0.1963	0.3947	0.6055
Naïve	0.3446	0.1256	0.3114	0.5606
SI	0.3418	0.1273	0.3131	0.5335
LwF	0.0750	0.0181	0.0541	0.1522

Table 5.1: Proportion of Entropy (Total uncertainty) attributed to Mutual Information (Epistemic Uncertainty) for the 4 experiments, respectively Permuted MNIST using MCD, Permuted MNIST using Deep Ensembles, DomainNet using MCD and DomainNet using Deep Ensembles

As seen in Table 5.1, the replay and Elastic Weight Consolidation (EWC) methods attribute a relatively high percentage of uncertainty towards epistemic uncertainty. On the other hand, using Learning without Forgetting (LwF) resulted in a significantly lower proportion of epistemic uncertainty, being at least 3 times less than the next lowest proportion. To further verify this observation, we also ensured that these relationships were not simply due to the differences in accuracy or entropy. As seen in Figure 5.2, we can also visually observe that the raw values of mutual information for LwF is significantly lower compared to the other continual learning methods that we employed.

Overall, it is noteworthy that as a method that regularizes the output, LwF also causes a shift in the decomposition of uncertainty towards aleatoric uncertainty.

5.3 Correlations between metrics

As discussed in Chapter 4, we calculated the correlations between various metrics, splitting between the initial epochs and later epochs. In this section, we highlight

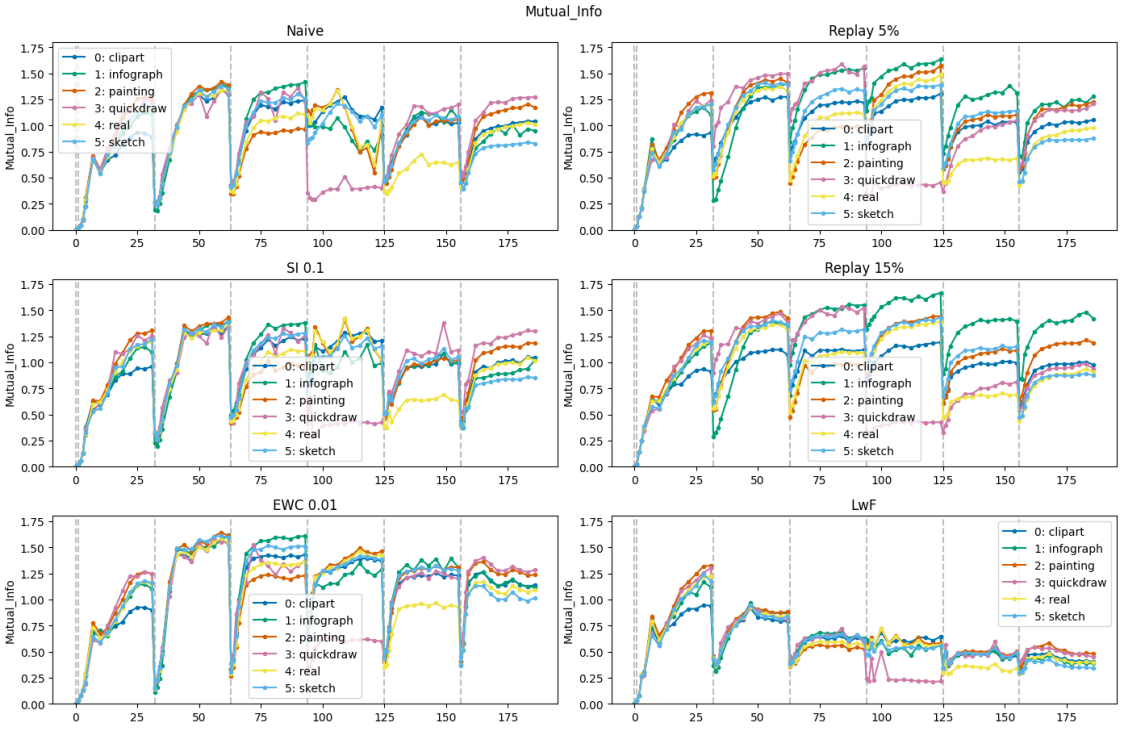
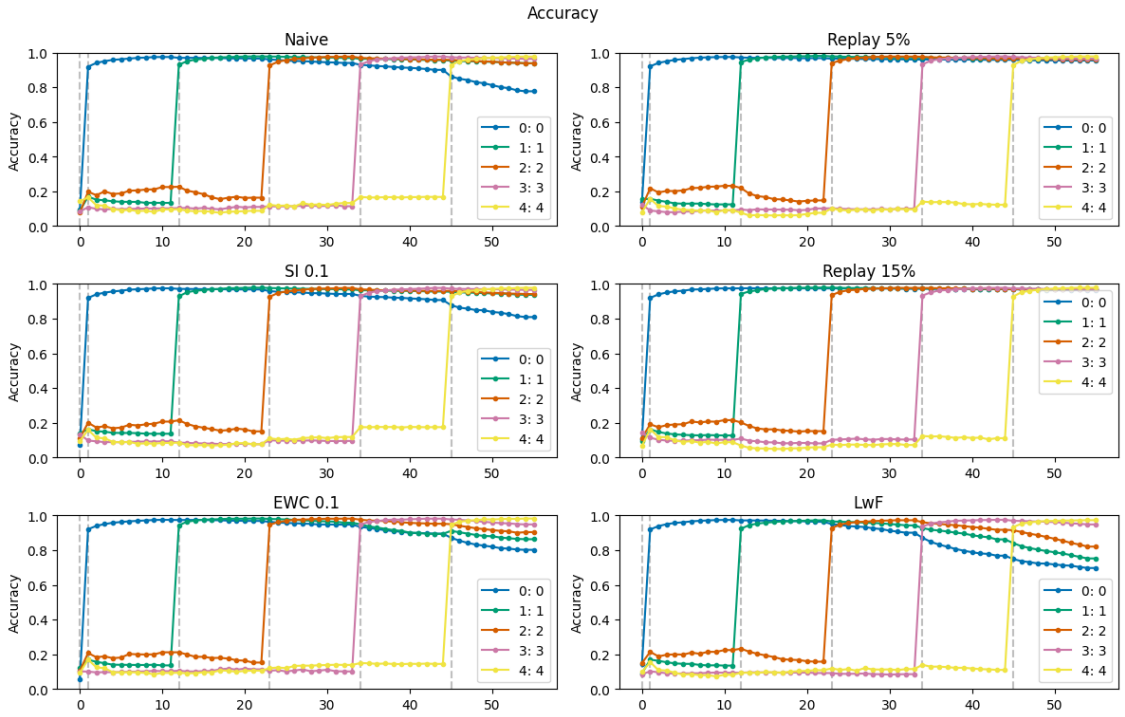


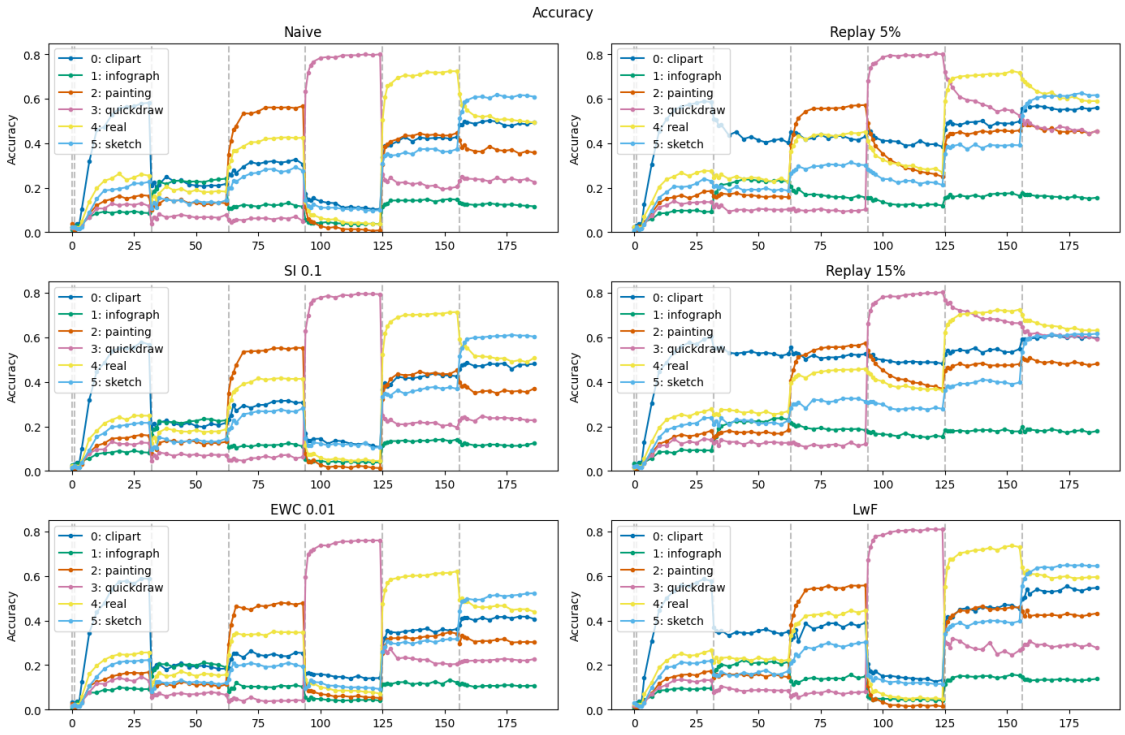
Figure 5.2: Mutual Information of different continual learning methods on the DomainNet dataset

the relationships that are significant and discuss how the various continual learning methods affect this correlation. By investigating these relationships, we hope to make use of certain metrics as a proxy for other important metrices when they may otherwise be unavailable. In particular, more attention was given to the relationships which correlated with either accuracy or the calibration error. An overview of the trends of the accuracy and static calibration error (SCE) can be viewed in Figure 5.3, Figure 5.4.

Across all the continual learning strategies and uncertainty quantification methods, by only observing the first few epochs of the testing tasks, there was a strong negative correlation between the accuracy and entropy. This meant that observing a decreased entropy generally meant that the accuracy was increasing Table 5.2. By comparing the two datasets and two uncertainty quantification methods, we observed that this correlation was relatively more pronounced in the replay-based methods while having the weakest relationship when we use EWC to train the models.

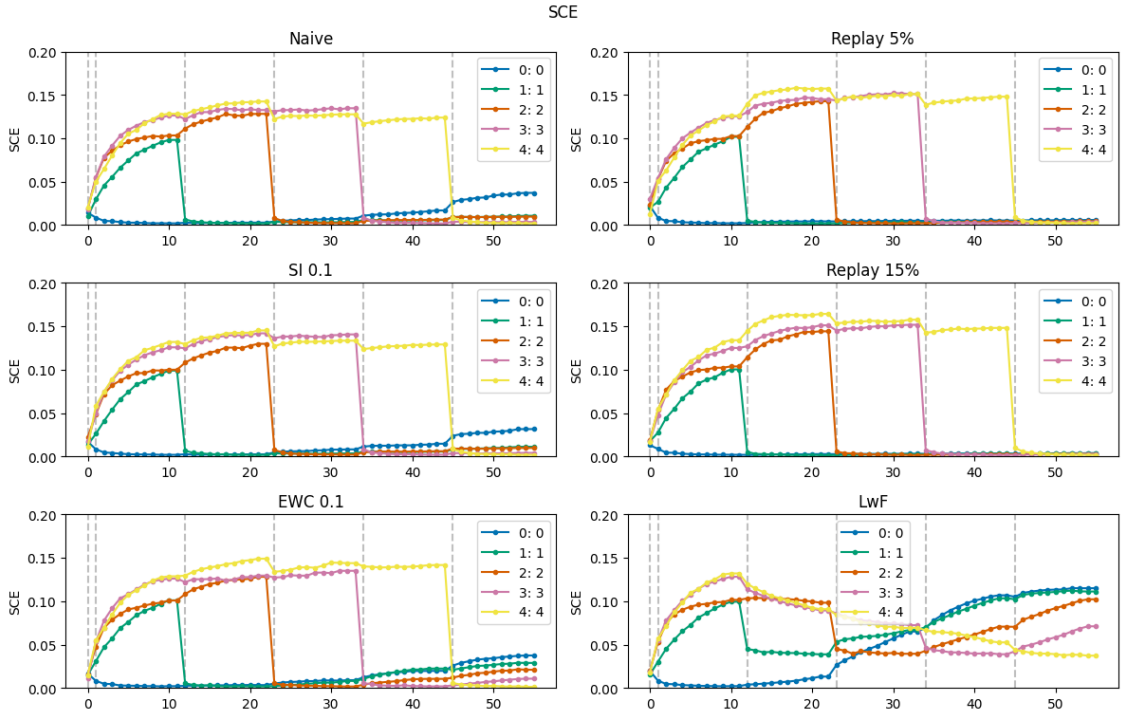


(a) Accuracy of Permuted MNIST

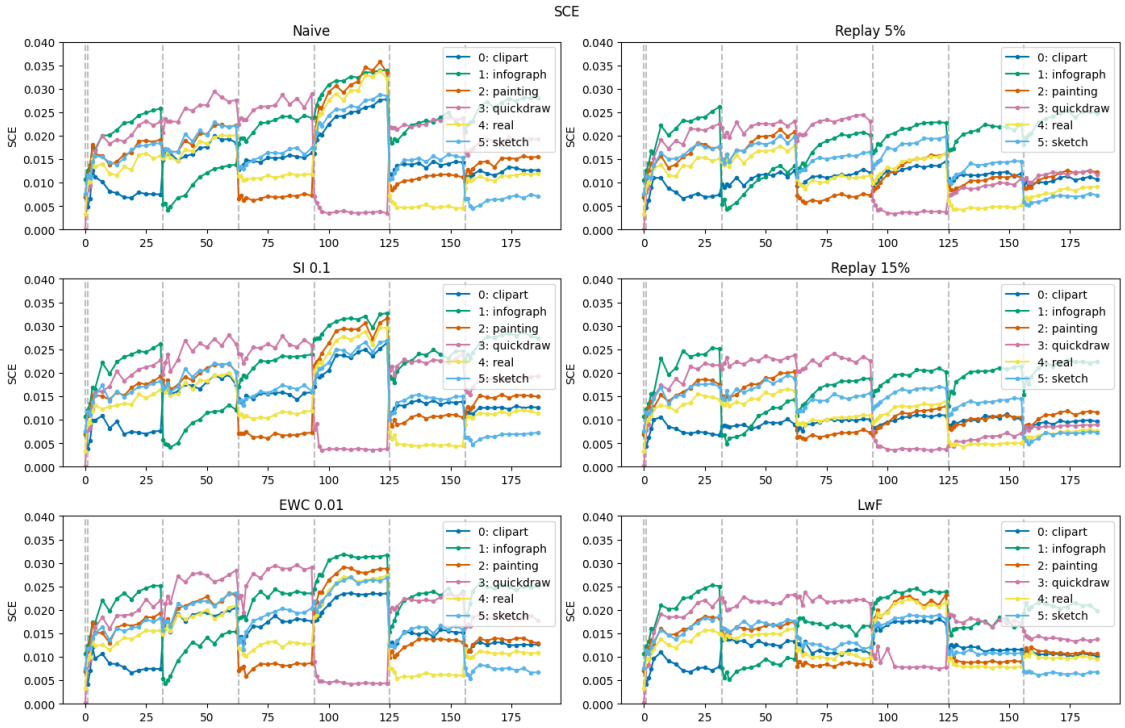


(b) Accuracy of DomainNet

Figure 5.3: Accuracy using Deep Ensembles for Permuted MNIST and DomainNet



(a) SCE of Permuted MNIST



(b) SCE of DomainNet

Figure 5.4: Static Calibration Error (SCE) using Deep Ensembles for Permuted MNIST and DomainNet

Method	PMNIST-MCD	PMNIST-ENS	DOMNET-MCD	DOMNET-ENS
Naïve	-0.9579	-0.9579	-0.5052	-0.6247
LwF	-0.6837	-0.7147	-0.6733	-0.7481
EWC	-0.5148	-0.8839	-0.4680	-0.5373
SI	-0.9406	-0.9540	-0.5141	-0.6015
Replay 5%	-0.8760	-0.9188	-0.6586	-0.7891
Replay 15%	-0.9604	-0.9760	-0.7354	-0.8363

Table 5.2: Correlation between Accuracy and Entropy **during initial** epochs of the experiences

However, after the initial epochs, looking only at the epistemic uncertainty (mutual information) seems to be a better indication of the accuracy. There is also an inverse relationship between accuracy and mutual information, though less strong as compared to relationship observed in the first few epochs. As seen in Table 5.3, the continual learning methods have differing impact on this relationship, with the replay methods exhibiting a relatively stronger relationship in the Permuted MNIST dataset, but a weaker inverse relationship in the DomainNet dataset. However, Learning without Forgetting (LwF) does not exhibit this strong relationship between accuracy and mutual information. As observed earlier, this could be due to the proportion of epistemic uncertainty for LwF. As a method that allocates most of its uncertainty to aleatoric uncertainty, it also results in a weaker relationship between mutual information and accuracy. Instead, we observe that while employing LwF, entropy remains a better indication of accuracy for the later epochs.

Method	PMNIST-MCD	PMNIST-ENS	DOMNET-MCD	DOMNET-ENS
Naïve	-0.9627	-0.9838	-0.3130	-0.6143
LwF	-0.4138	-0.8282	-0.0585	-0.4247
EWC	-0.7019	-0.9437	-0.4122	-0.7604
SI	-0.9718	-0.9756	-0.3435	-0.6948
Replay 5%	-0.6270	-0.7052	-0.8227	-0.8704
Replay 15%	-0.6149	-0.3438	-0.8205	-0.8842

Table 5.3: Correlation between Accuracy and Mutual Information **after initial** epochs of the experiences

Besides metrics which correlate with accuracy, calibration metrics are also of importance. By analysing the mutual information, we observed that there is a positive correlation between the SCE and mutual information. Contrary to the previous

correlations, this correlation was evident even without splitting the experiences into an 'initial' and 'later' phase (Table 5.4). Similar to the relationship between mutual information and accuracy, the LwF strategy does not exhibit this correlation as strongly. In the DomainNet dataset evaluated using Monte Carlo Dropout, there was almost no linear correlation observed between the metrics.

Method	PMNIST-MCD	PMNIST-ENS	DOMNET-MCD	DOMNET-ENS
Naïve	0.9684	0.9798	0.3921	0.4704
LwF	0.4819	0.8769	-0.1025	0.1678
EWC	0.9800	0.9765	0.3379	0.4895
SI	0.9724	0.9770	0.3955	0.5103
Replay 5%	0.6227	0.6493	0.8191	0.7401
Replay 15%	0.8074	0.5625	0.8578	0.8186

Table 5.4: Correlation between SCE and Mutual Information **across all** epochs of the experiences

Another noteworthy relationship was observed between the uncertainty calibration error (UCE) and signal-to-noise ratio (SNR). When observing the later epochs of each test experience, there was an observed negative correlation between the UCE and SNR(Table 5.5). However, this relationship was less consistent across the various continual learning strategies and uncertainty quantification methods.

Method	PMNIST-MCD	PMNIST-ENS	DOMNET-MCD	DOMNET-ENS
Naïve	-0.3233	-0.8239	-0.0861	-0.4856
LwF	-0.5494	-0.5233	0.0990	-0.1205
EWC	-0.6408	-0.7345	-0.3675	-0.4291
SI	-0.3994	-0.8044	-0.5686	-0.5017
Replay 5%	-0.8749	-0.3485	-0.5544	-0.4443
Replay 15%	-0.8327	-0.6658	-0.5670	-0.4468

Table 5.5: Correlation between UCE and Signal-to-Noise Ratio **after initial** epochs of the experiences

Overall, observing the correlation between metrics that are easily computable, and metrics that we may desire, is a useful tool when faced with unknown data. Due to how the correlations were calculated in our experiments, these correlations apply without the knowledge of the task identity, which is useful in real-world situations.

Chapter 6

Conclusion and Future Work

This report aims to understand the uncertainties in a domain-incremental setting. By conducting experiments on the Permuted MNIST dataset and DomainNet dataset as benchmarks, we analysed various uncertainty quantification and calibration metrics while utilising Monte Carlo Dropout and Deep Ensembles. We observed how various metrics correlated with each other and whether we could use easily computable metrics as a proxy to estimate other important ones.

Looking ahead, there are two extensions which are useful to investigate based on the observations represented. Firstly, investigations could be conducted on improving the stability between experiences. As discussed in § 5.1, there are drastic changes when a new domain is introduced. Further investigation could be conducted to allow a more stable training dynamic. Another potential research direction is investigating out-of-distribution data. In our experiments, we analysed metrics form test data that had already been trained on. However, in real-world settings, there is no guarantee that we will always encounter such data. Therefore, it may be meaningful to investigate how uncertainty is affected by out-of-distribution data.

Bibliography

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks", 2015. arXiv: 1505.05424 [stat.ML].

[2] Y. Cai, Z. Wang, and D. Goyal, "22 - applications of terahertz technology in the semiconductor industry", in *Handbook of Terahertz Technology for Imaging, Sensing and Communications*, ser. Woodhead Publishing Series in Electronic and Optical Materials, D. Saeedkia, Ed., Woodhead Publishing, 2013, pp. 624–640, ISBN: 978-0-85709-235-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780857092359500227.

[3] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence", in *Computer Vision – ECCV 2018*, Springer International Publishing, 2018, pp. 556–572. [Online]. Available: https://doi.org/10.1007%2F978-3-030-01252-6_33.

[4] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning", 2018. arXiv: 1710.07283 [stat.ML].

[5] S. Farquhar and Y. Gal, "A unifying bayesian view of continual learning", 2019. arXiv: 1902.06494 [stat.ML].

[6] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, and W. Dabney, "Revisiting fundamentals of experience replay", 2020. arXiv: 2007.06700 [cs.LG].

[7] L. Frau, G. A. Susto, T. Barbariol, and E. Feltresi, "Uncertainty estimation for machine learning models in multiphase flow applications", *Informatics*, vol. 8, no. 3, 2021, ISSN: 2227-9709. [Online]. Available: https://www.mdpi.com/2227-9709/8/3/58.

[8] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning", 2016. arXiv: 1506.02142 [stat.ML].

[9] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data", 2017. arXiv: 1703.02910 [cs.LG].

[10] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks", 2015. arXiv: 1312.6211 [stat.ML].

[11] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks", 2017. arXiv: 1706.04599 [cs.LG].

[12] T. Heskes, "Practical confidence and prediction intervals", in *Advances in Neural Information Processing Systems*, M. C. Mozer, M. Jordan, and T. Petsche, Eds., vol. 9, MIT Press, 1996. [Online]. Available: https://proceedings.neurips.cc/paper/1996/file/7940ab47468396569a906f75ff3f20ef-Paper.pdf.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", 2012. arXiv: 1207.0580 [cs.NE].

[14] D. S. Holmes, "Signal to noise ratio-what is the right size ?", 2007.

[15] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, "Re-evaluating continual learning scenarios: A categorization and case for strong baselines", 2019. arXiv: 1810.12488 [cs.LG].

[16] D. Isele and A. Cosgun, "Selective experience replay for lifelong learning", 2018. arXiv: 1802.10269 [cs.AI].

[17] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" 2017. arXiv: 1703.04977 [cs.CV].

[18] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks", *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017. [Online]. Available: https://doi.org/10.1073%2Fpnas.1611835114.

[19] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? does it matter?" *Structural Safety*, vol. 31, no. 2, pp. 105–112, 2009, Risk Acceptance and Risk Communication, ISSN: 0167-4730. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167473008000556.

[20] Y. Lai, Y. Shi, Y. Han, Y. Shao, M. Qi, and B. Li, "Exploring uncertainty in deep learning for construction of prediction intervals", 2021. arXiv: 2104.12953 [cs.LG].

[21] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles", 2017. arXiv: 1612.01474 [stat.ML].

[22] M.-H. Laves, S. Ihler, K.-P. Kortmann, and T. Ortmaier, "Calibration of model uncertainty for dropout variational inference", 2020. arXiv: 2006.11584 [cs.LG].

[23] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization", 2017. arXiv: 1710.03077 [cs.CV].

[24] Z. Li and D. Hoiem, "Learning without forgetting", 2017. arXiv: 1606.09282 [cs.CV].

[25] Z. Lin, J. Shi, D. Pathak, and D. Ramanan, "The clear benchmark: Continual learning on real-world imagery", 2022. arXiv: 2201.06289 [cs.CV].

[26] V. Lomonaco and D. Maltoni, "Core50: A new dataset and benchmark for continuous object recognition", 2017. arXiv: 1705.03550 [cs.CV].

[27] V. Lomonaco, L. Pellegrini, A. Cossu, A. Carta, G. Graffieti, T. L. Hayes, M. D. Lange, M. Masana, J. Pomponi, G. van de Ven, M. Mundt, Q. She, K. Cooper, J. Forest, E. Belouadah, S. Calderara, G. I. Parisi, F. Cuzzolin, A. Tolias, S. Scardapane, L. Antiga, S. Amhad, A. Popescu, C. Kanan, J. van de Weijer, T. Tuytelaars, D. Bacciu, and D. Maltoni, "Avalanche: An end-to-end library for continual learning", in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, ser. 2nd Continual Learning in Computer Vision Workshop, 2021.

[28] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning", 2022. arXiv: 1706.08840 [cs.LG].

[29] D. Maltoni and V. Lomonaco, "Continuous learning in single-incremental-task scenarios", 2019. arXiv: 1806.08568 [cs.LG].

[30] M. J. Marquez, "A bayesian approach to the inference of parametric configuration of the signal-to-noise ratio in an adaptive refinement of the measurements", 2012. arXiv: 1208.2048 [astro-ph.IM].

[31] M. McCloskey and N. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem", English (US), *Psychology of Learning and Motivation - Advances in Research and Theory*, vol. 24, no. C, pp. 109–165, Jan. 1989, ISSN: 0079-7421.

[32] A. A. Mishra, A. Edelen, A. Hanuka, and C. Mayes, "Uncertainty quantification for deep learning in particle accelerator applications", *Phys. Rev. Accel. Beams*, vol. 24, p. 114 601, 11 Nov. 2021. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevAccelBeams.24.114601.

[33] A. Murad, F. A. Kraemer, K. Bach, and G. Taylor, "Probabilistic deep learning to quantify uncertainty in air quality forecasting", *Sensors*, vol. 21, no. 23, p. 8009, Nov. 2021, ISSN: 1424-8220. [Online]. Available: http://dx.doi.org/10.3390/s21238009.

[34] M. P. Naeini, G. F. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using bayesian binning", in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, ser. AAAI'15, Austin, Texas: AAAI Press, 2015, pp. 2901–2907, ISBN: 0262511290.

[35] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning", 2018. arXiv: 1710.10628 [stat.ML].

[36] J. Nixon, M. Dusenberry, G. Jerfel, T. Nguyen, J. Liu, L. Zhang, and D. Tran, "Measuring calibration in deep learning", 2020. arXiv: 1904.01685 [cs.LG].

[37] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review",, 2018. [Online]. Available: https://arxiv.org/abs/1802.07569.

[38] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation", in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1406–1415.

[39] D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, and G. Wayne, "Experience replay for continual learning", 2019. arXiv: 1811.11682 [cs.LG].

[40] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay", 2017. arXiv: 1705.08690 [cs.AI].

[41] L. Smith and Y. Gal, "Understanding measures of uncertainty for adversarial example detection", 2018. arXiv: 1803.08533 [stat.ML].

[42] S. Swaroop, C. V. Nguyen, T. D. Bui, and R. E. Turner, "Improving and understanding variational continual learning", 2019. arXiv: 1905.02099 [stat.ML].

[43] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks", 2020. arXiv: 1905.11946 [cs.LG].

[44] S. Thrun and T. M. Mitchell, "Lifelong robot learning", *Robotics and Autonomous Systems*, vol. 15, no. 1, pp. 25–46, 1995, The Biology and Technology of Intelligent Autonomous Agents, ISSN: 0921-8890. [Online]. Available: https://www.sciencedirect.com/science/article/pii/092188909500004Y.

[45] G. M. van de Ven and A. S. Tolias, "Three scenarios for continual learning", 2019. arXiv: 1904.07734 [cs.LG].

[46] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics", in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11, Bellevue, Washington, USA: Omnipress, 2011, pp. 681–688, ISBN: 9781450306195.

[47] S. Yu, H. Cha, H. Lee, S. Mo, J. Park, and H. Kang, "Transfer and continual learning", University lecture presented in AI602: Recent Advances in Deep Learning, 2022. [Online]. Available: https://alinlab.kaist.ac.kr/resource/2022_AI602_Lec09.pdf.

[48] T. Yuan, W. Deng, J. Tang, Y. Tang, and B. Chen, "Signal-to-noise ratio: A robust distance metric for deep metric learning", 2019. arXiv: 1904.02616 [cs.CV].

[49] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence", 2017. arXiv: 1703.04200 [cs.LG].

[50] J. Zhang, B. Kailkhura, and T. Y.-J. Han, "Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning", 2020. arXiv: 2003.07329 [cs.LG].

Appendix A

Detailed Experiment Plots

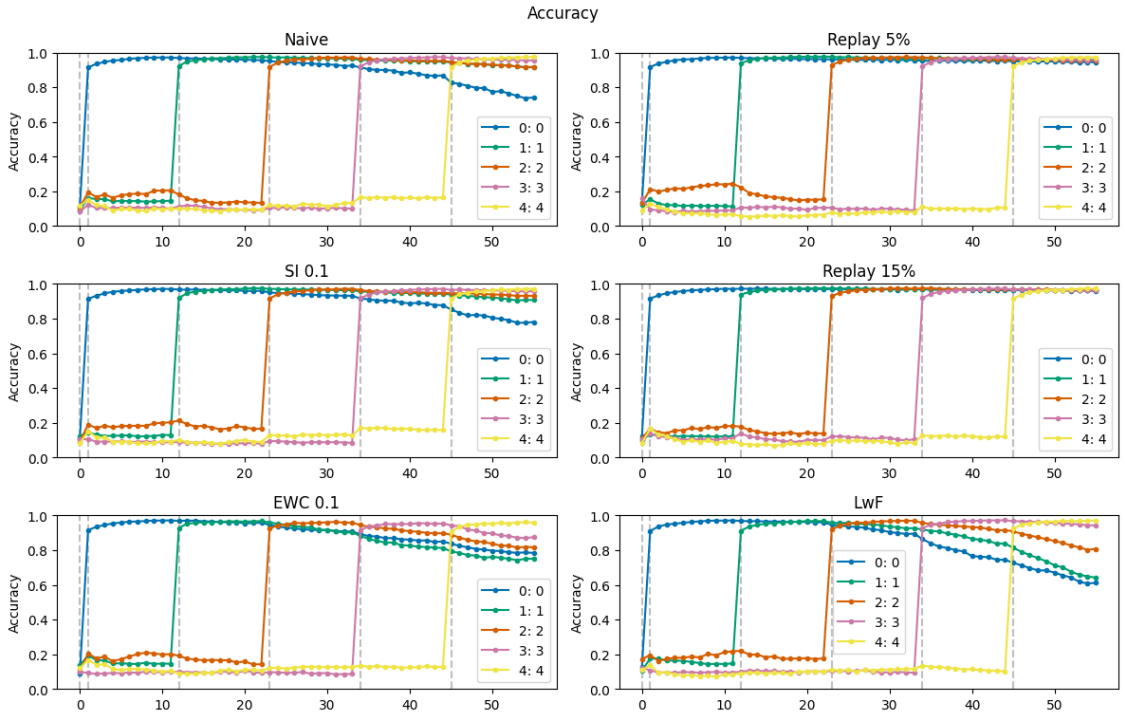


Figure A.1: Accuracy of Permuted MNIST - MCD

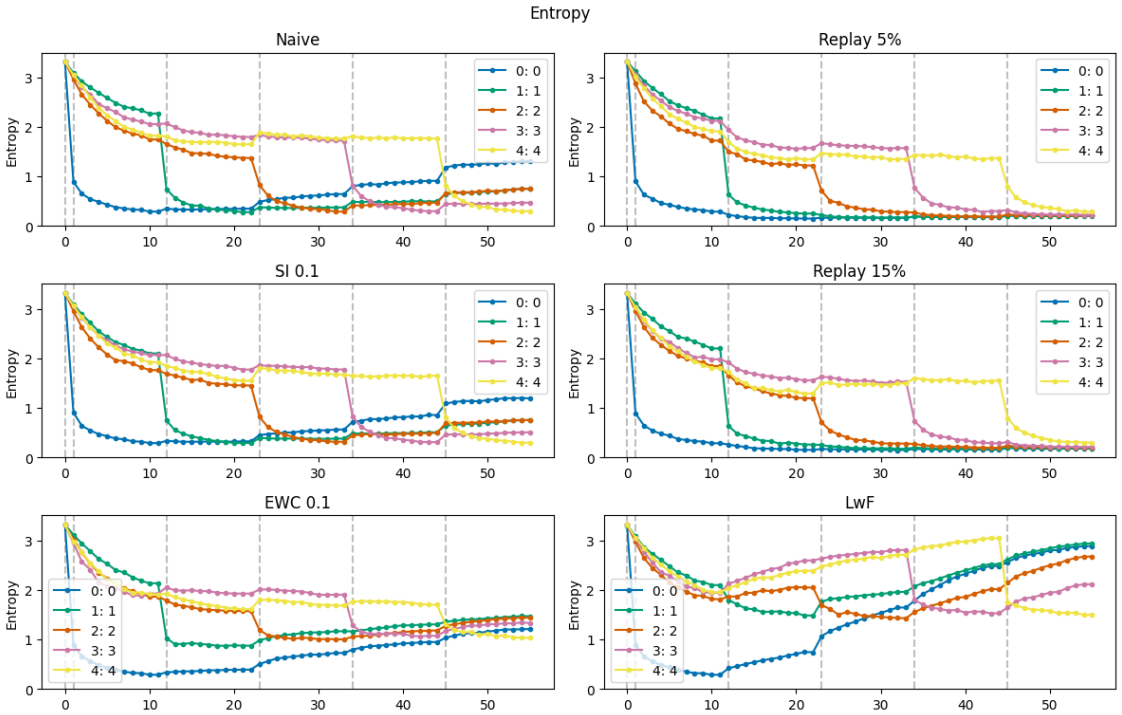


Figure A.2: Entropy of Permuted MNIST - MCD

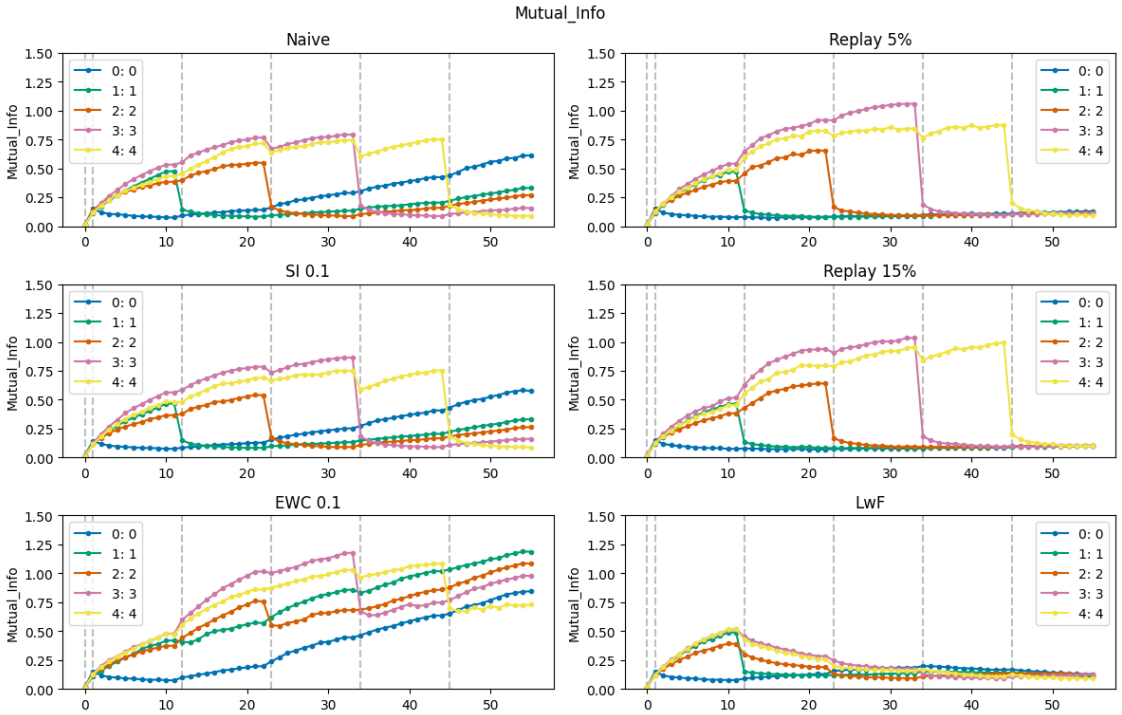


Figure A.3: MI of Permuted MNIST - MCD

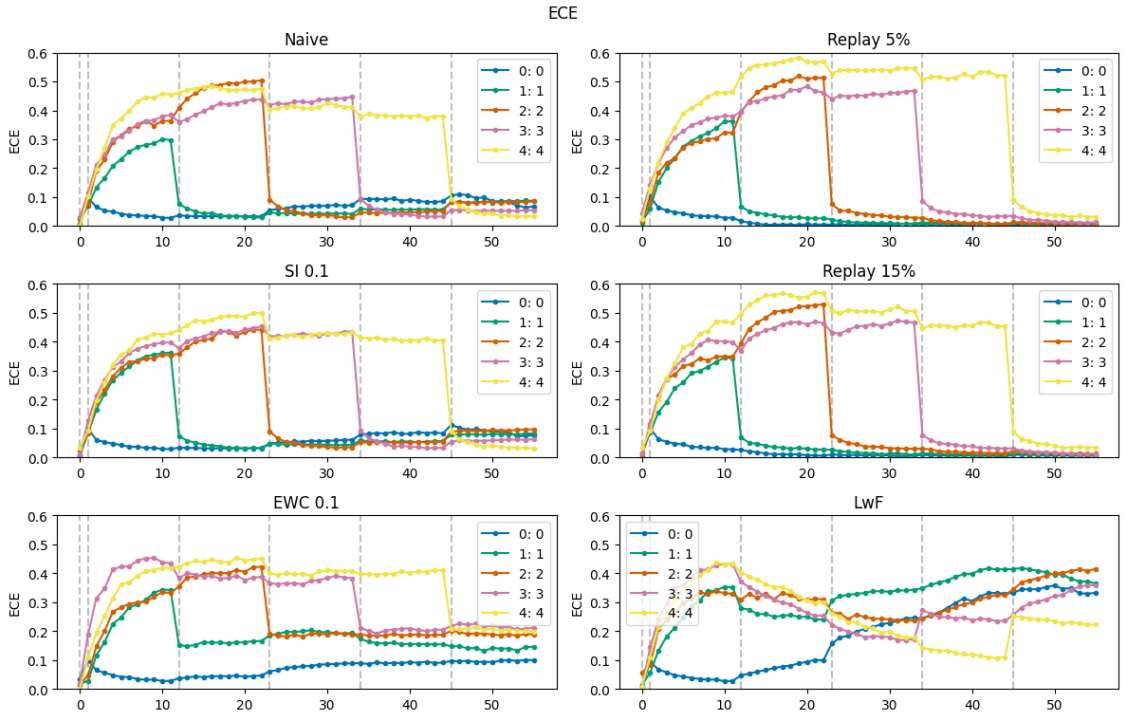


Figure A.4: ECE of Permuted MNIST - MCD

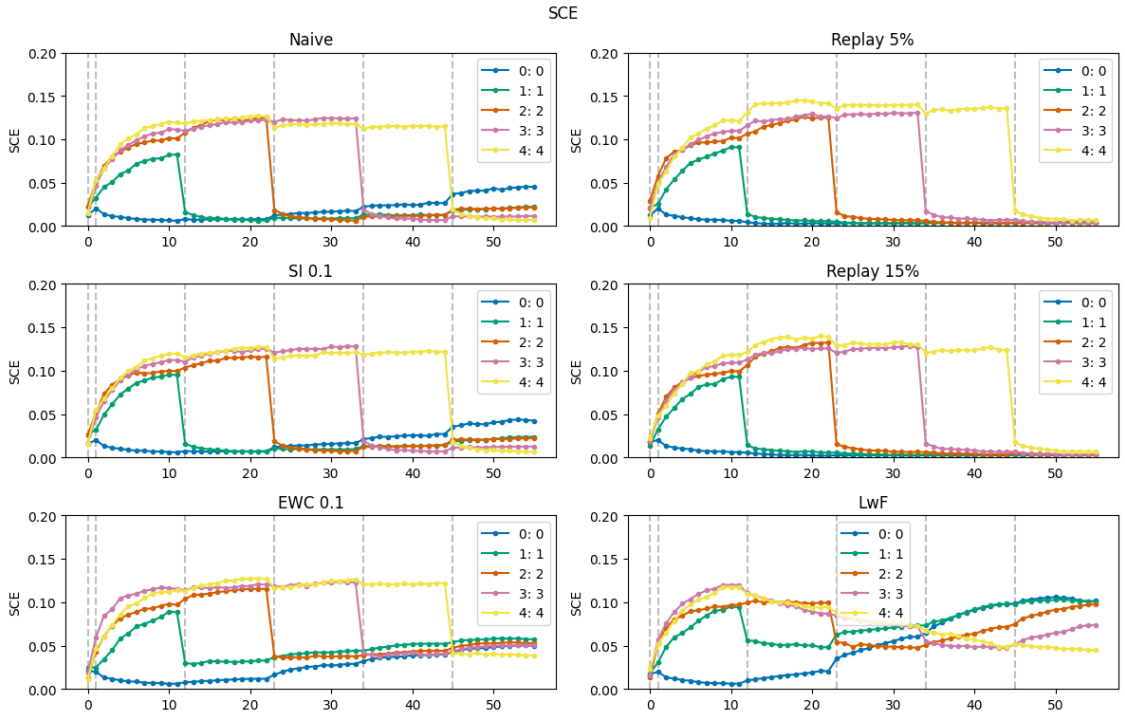


Figure A.5: SCE of Permuted MNIST - MCD

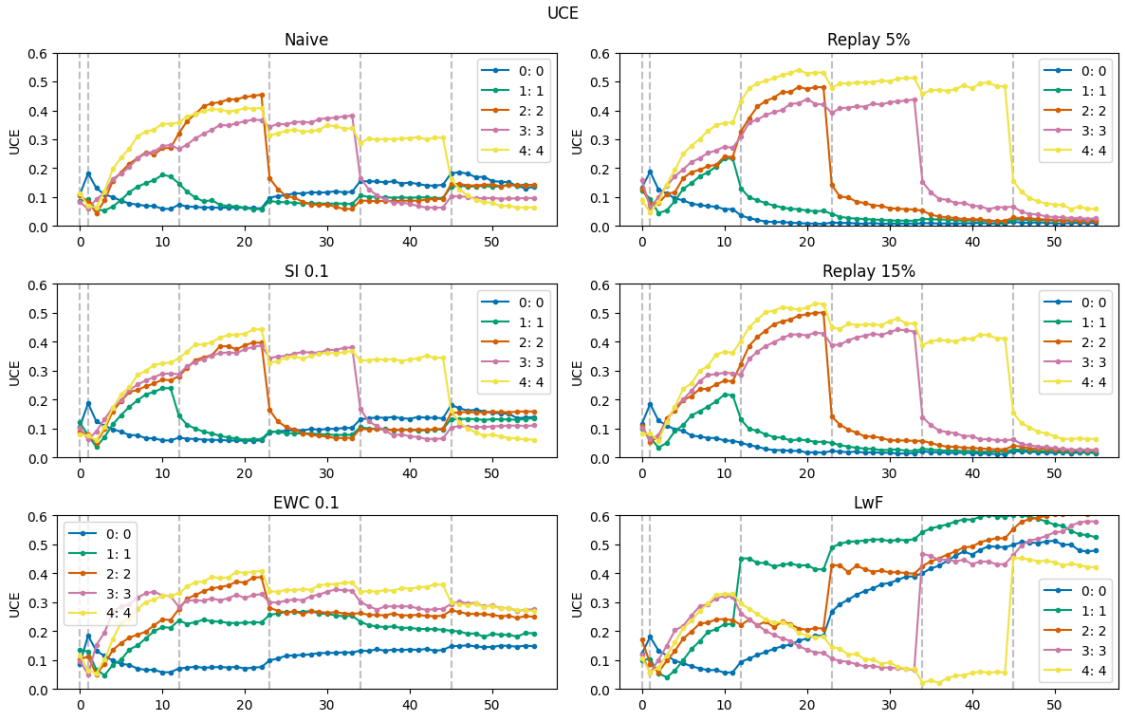


Figure A.6: UCE of Permuted MNIST - MCD

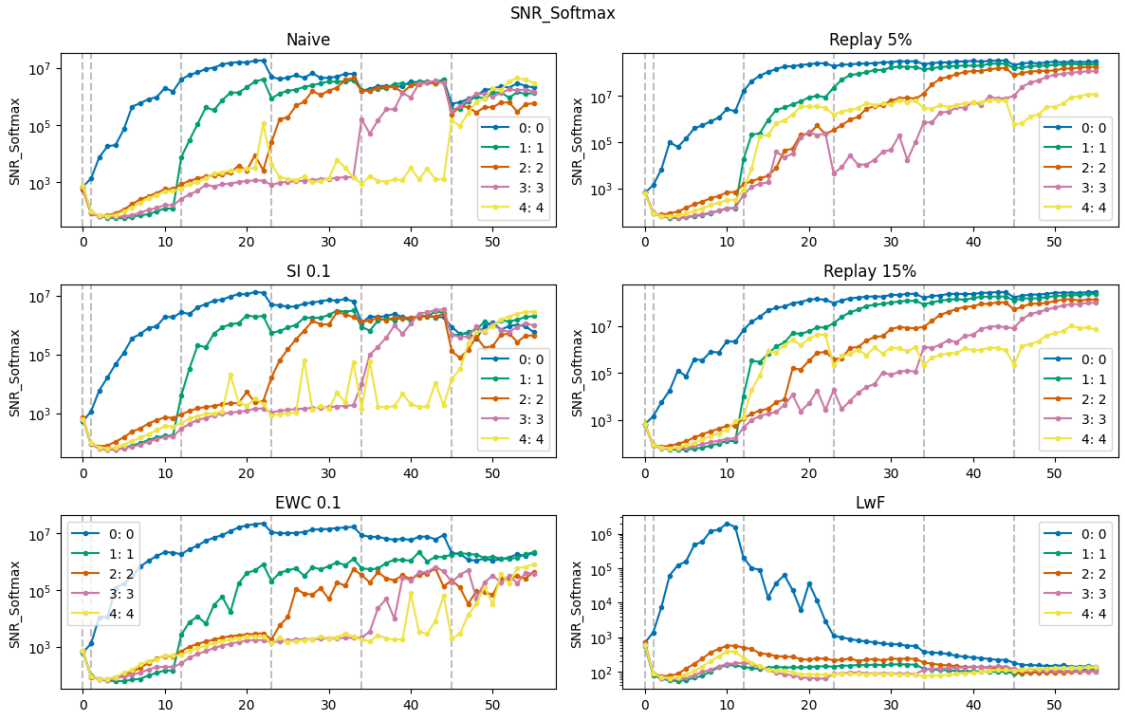


Figure A.7: Signal-to-Noise Ratio of Permuted MNIST - MCD

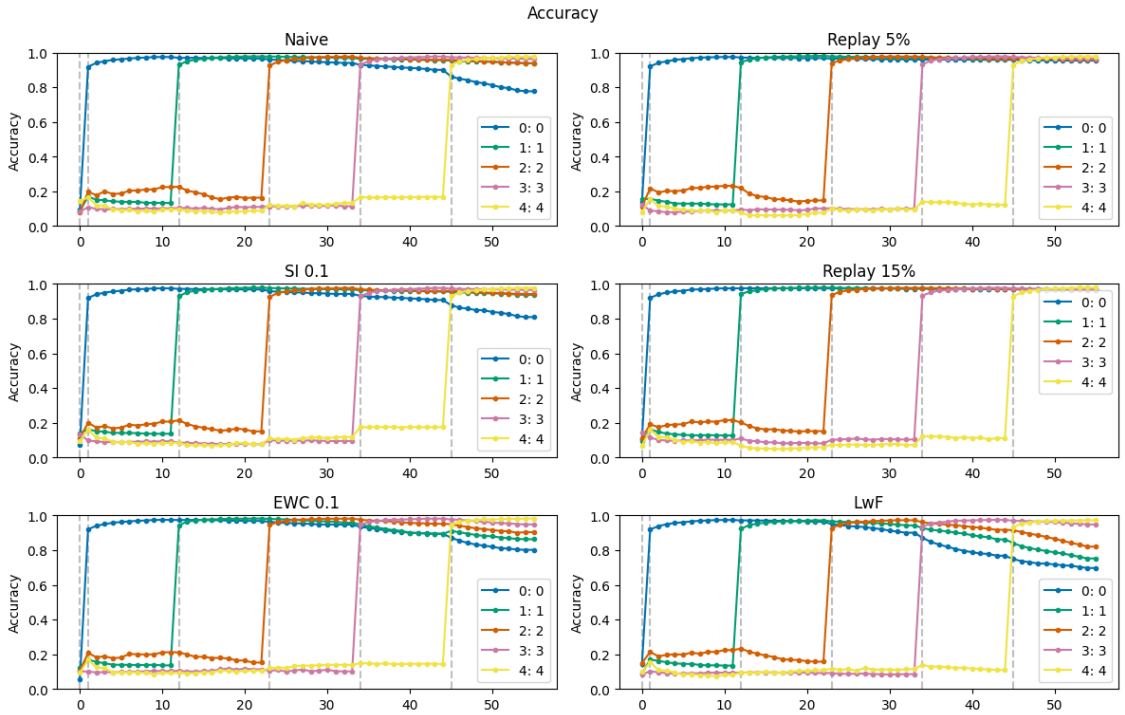


Figure A.8: Accuracy of Permuted MNIST - Ensembles

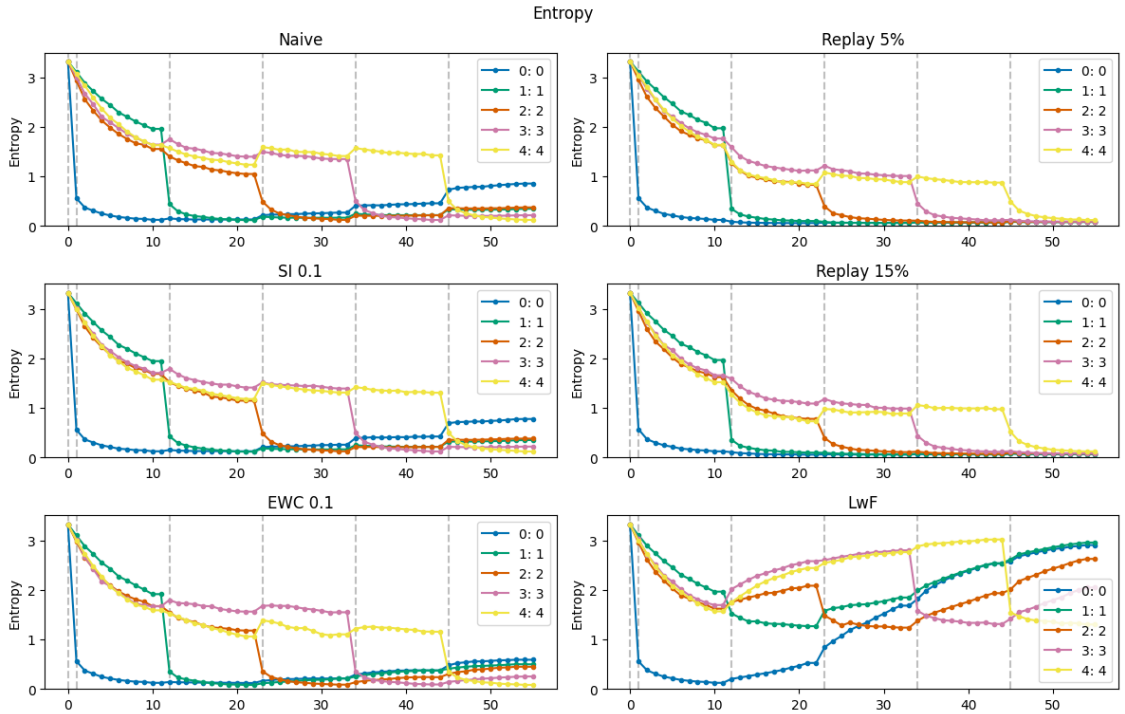


Figure A.9: Entropy of Permuted MNIST - Ensembles

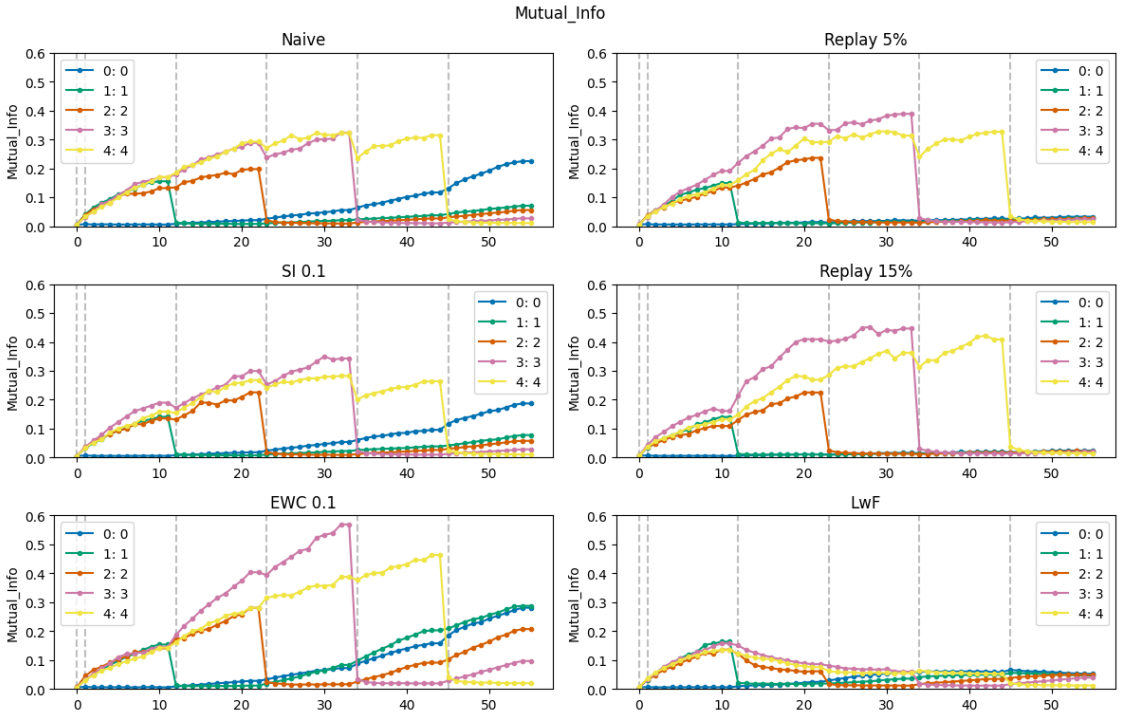


Figure A.10: MI of Permuted MNIST - Ensembles

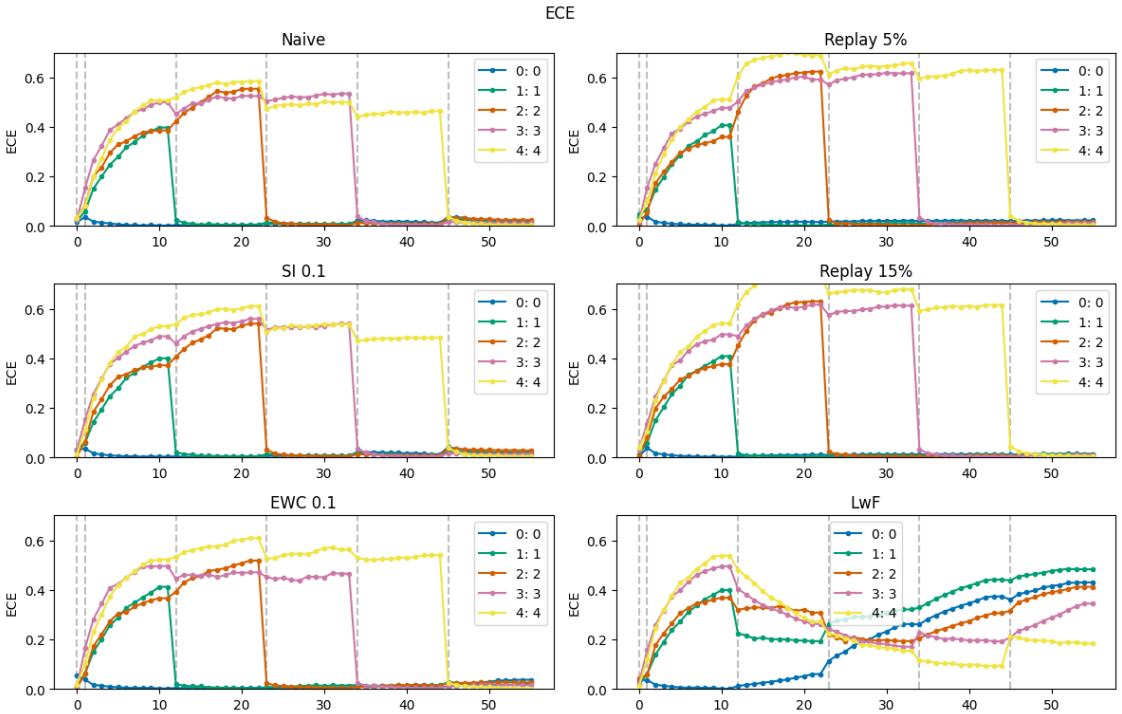


Figure A.11: ECE of Permuted MNIST - Ensembles

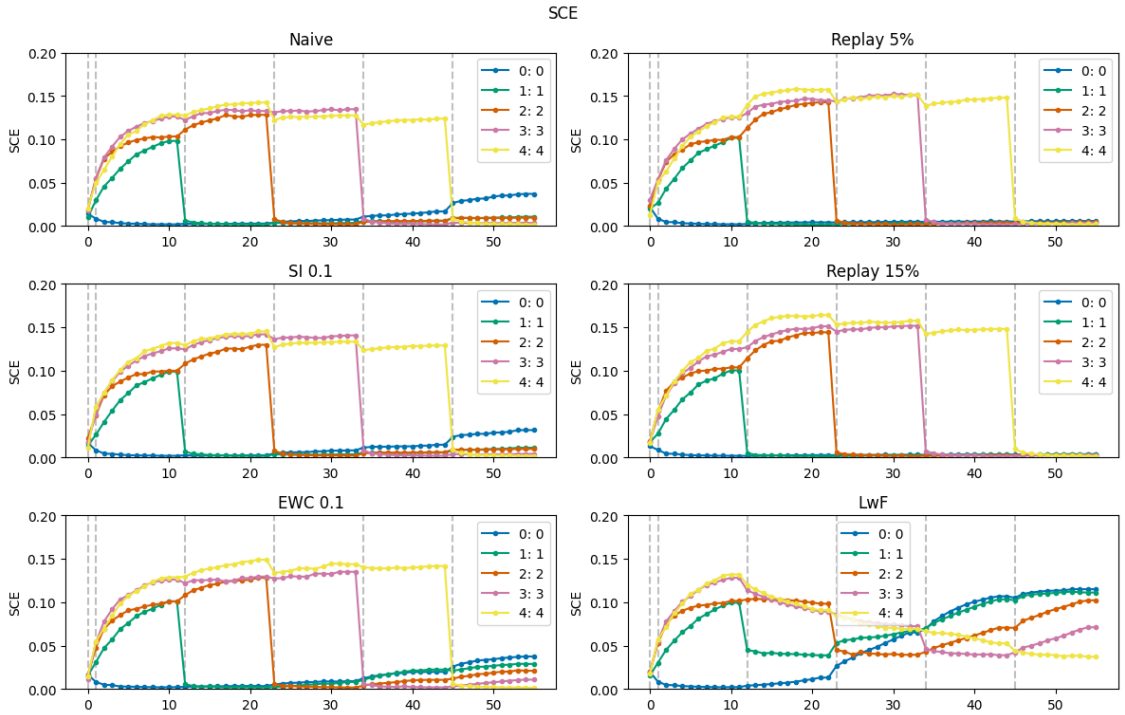


Figure A.12: SCE of Permuted MNIST - Ensembles

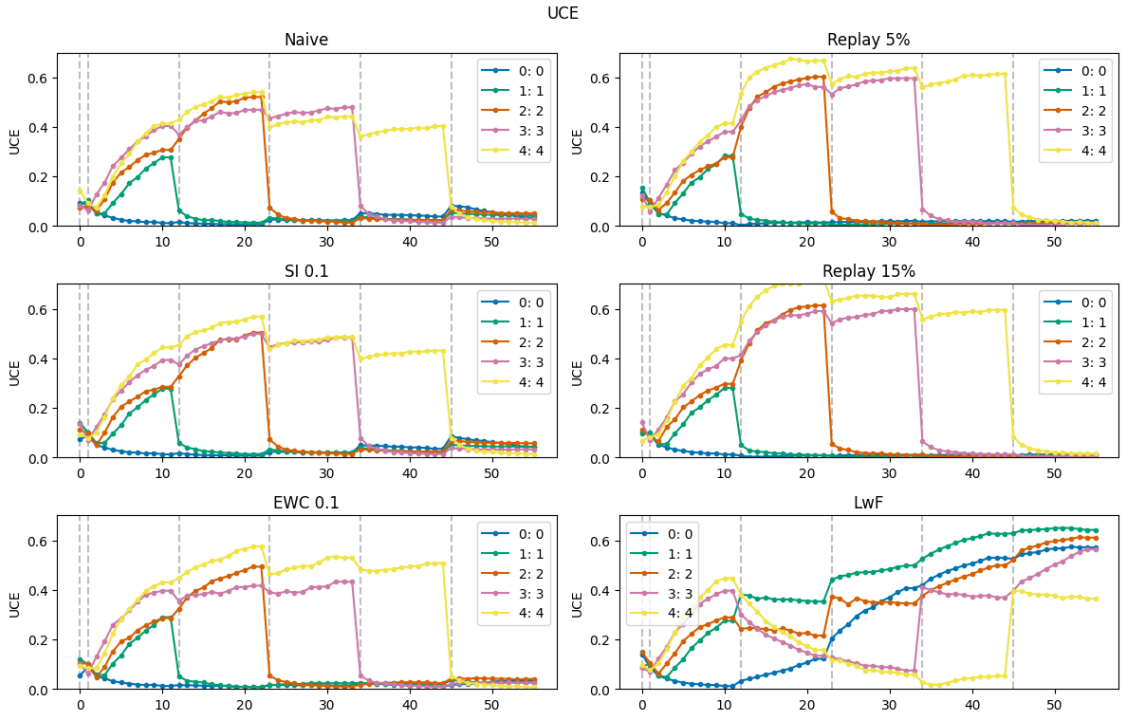


Figure A.13: UCE of Permuted MNIST - Ensembles

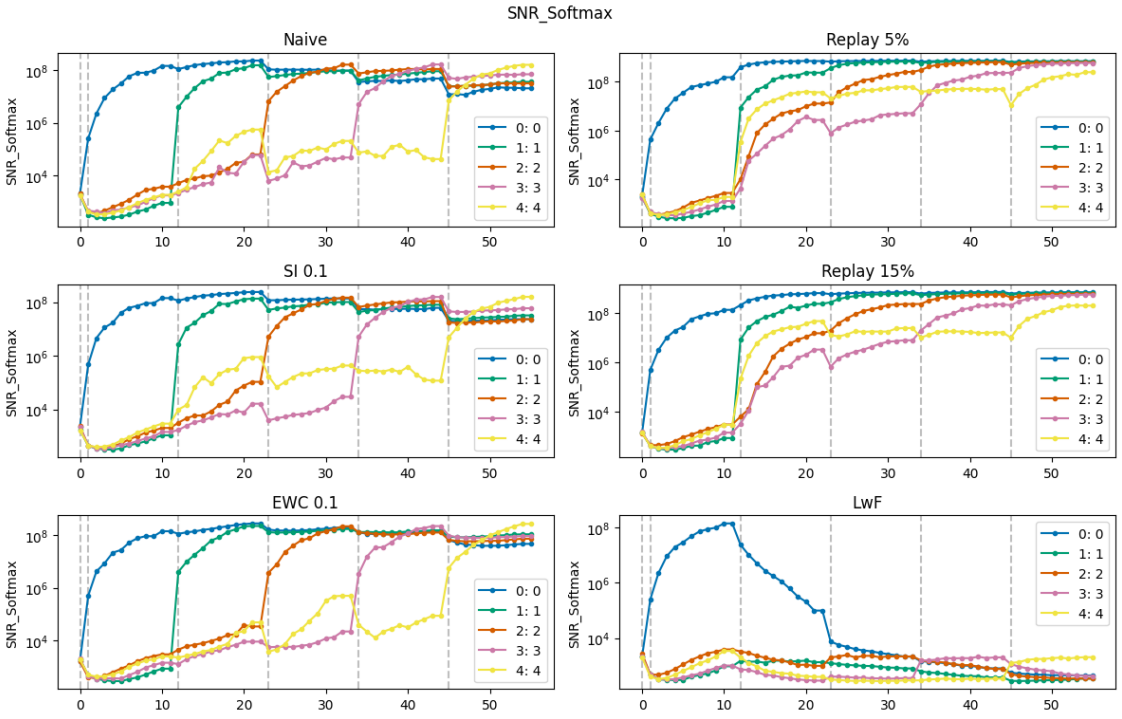


Figure A.14: Signal-to-Noise Ratio of Permuted MNIST - Ensembles

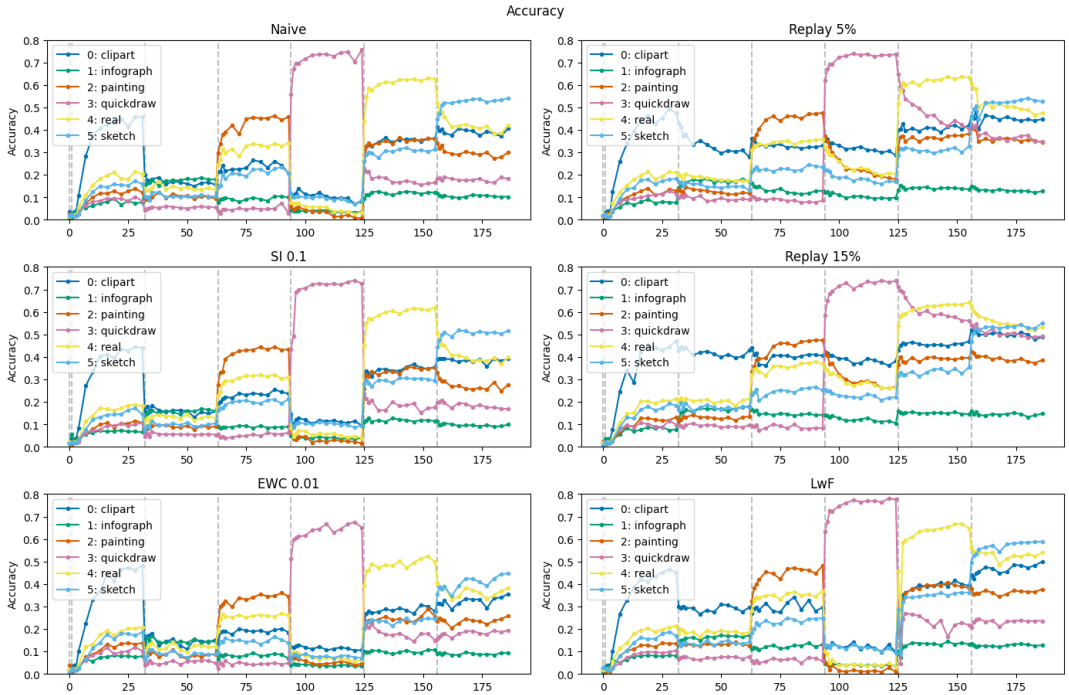


Figure A.15: Accuracy of DomainNet - MCD

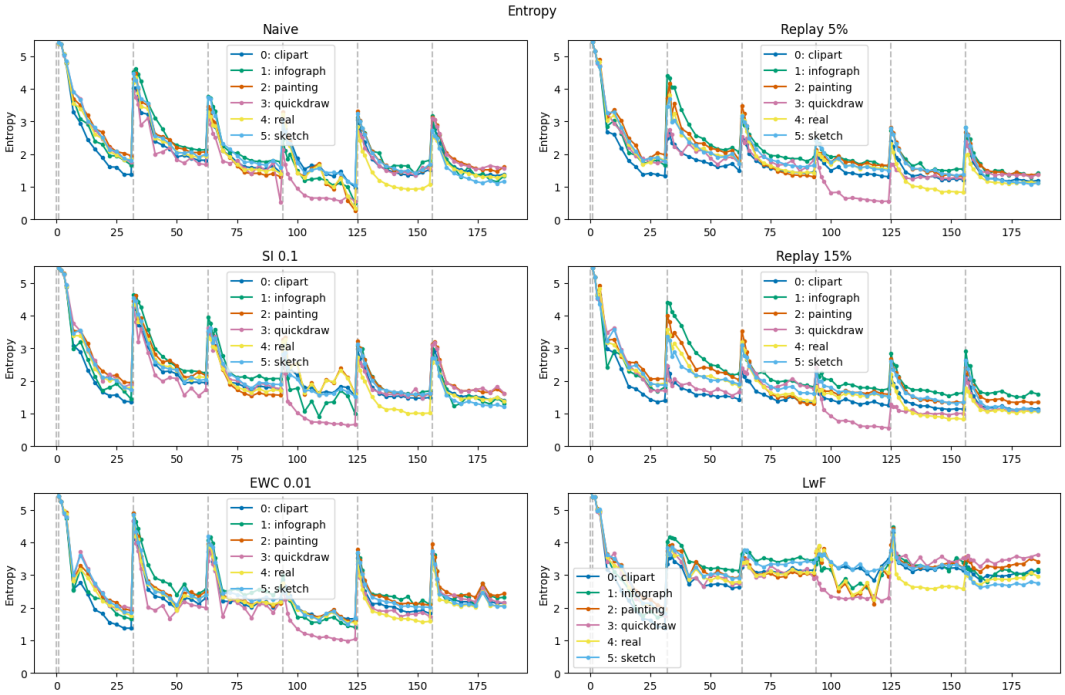


Figure A.16: Entropy of DomainNet - MCD

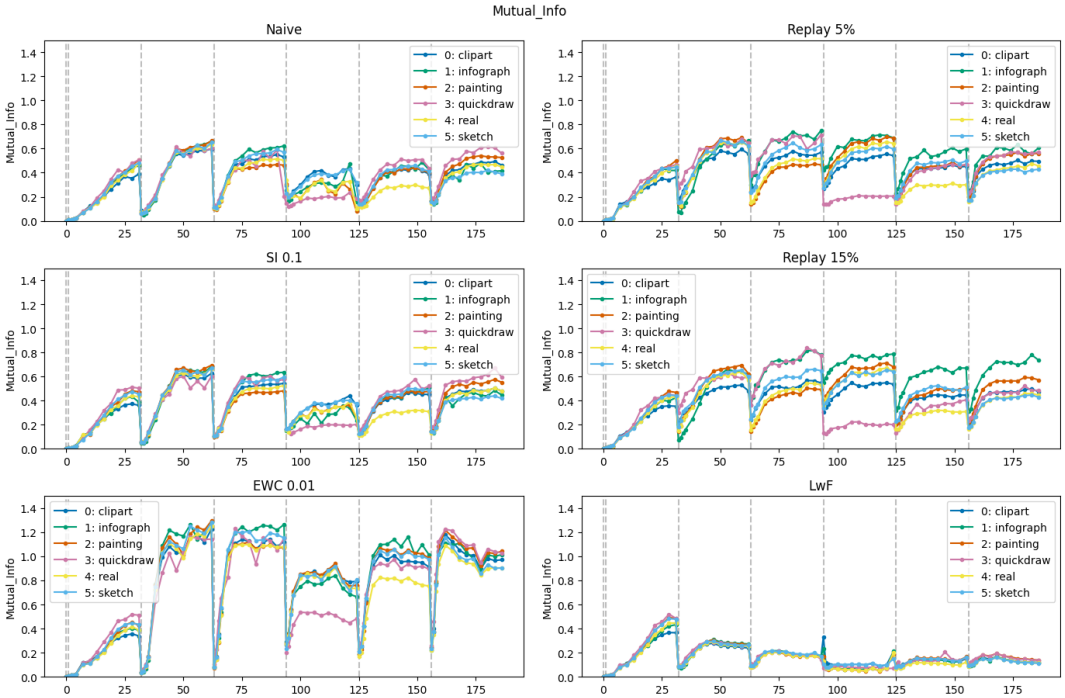


Figure A.17: MI of DomainNet - MCD

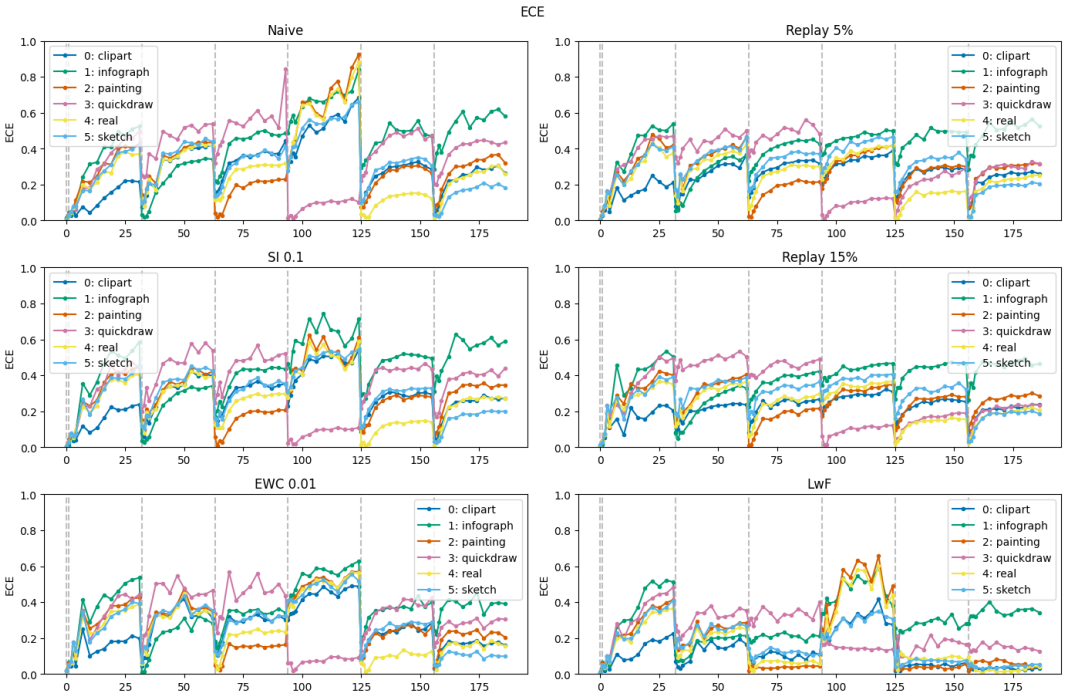


Figure A.18: ECE of DomainNet - MCD

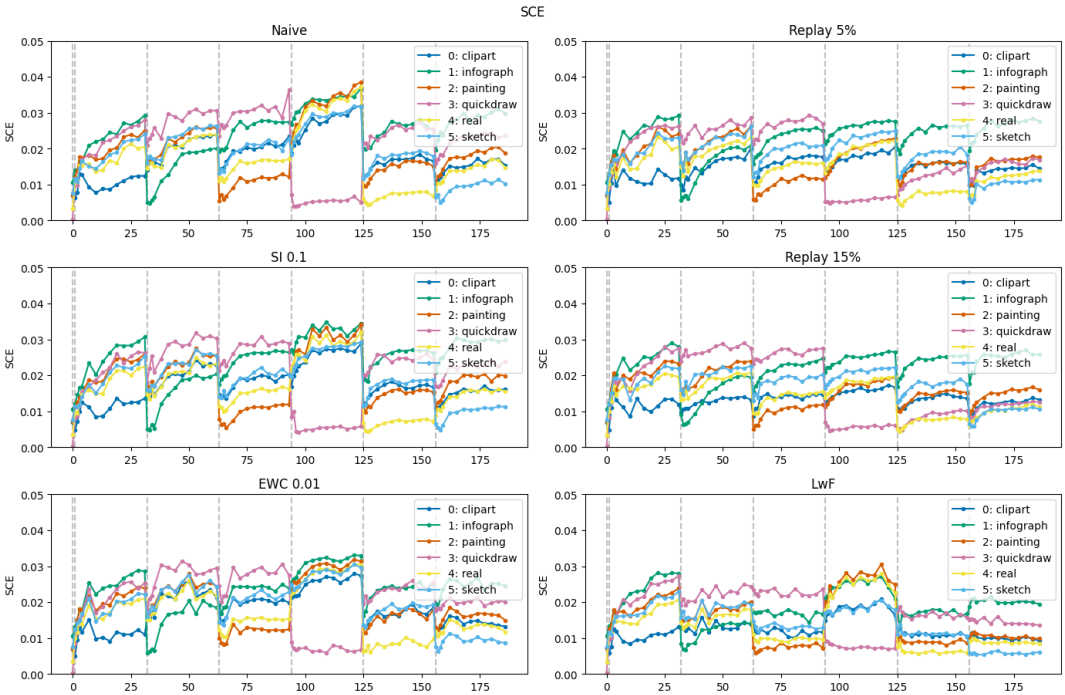


Figure A.19: SCE of DomainNet - MCD

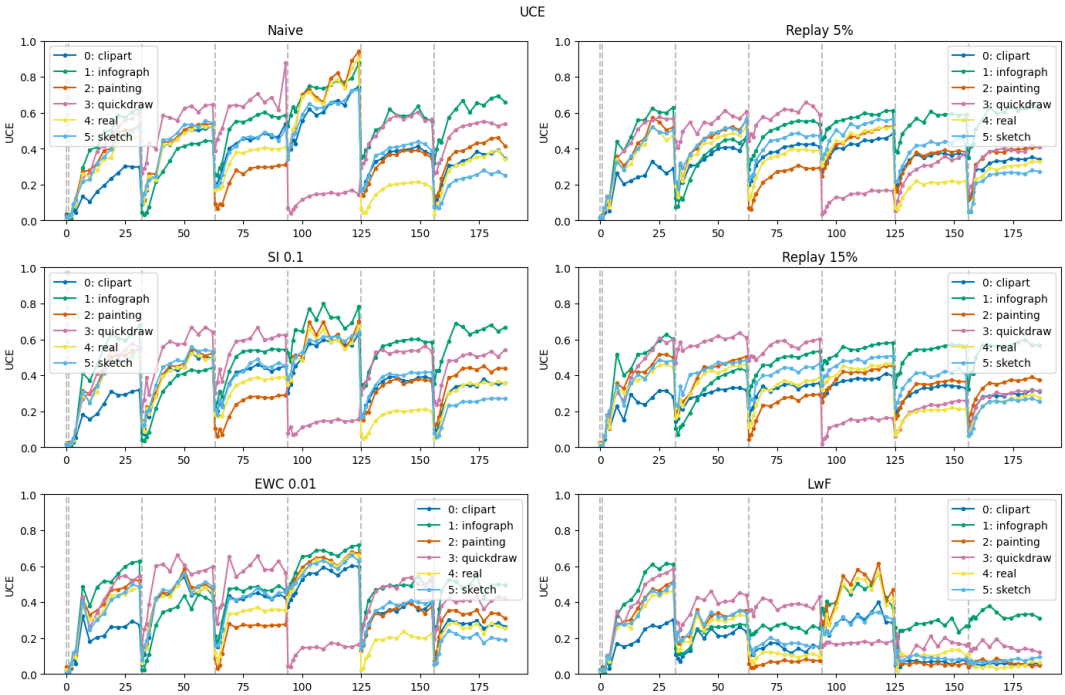


Figure A.20: UCE of DomainNet - MCD

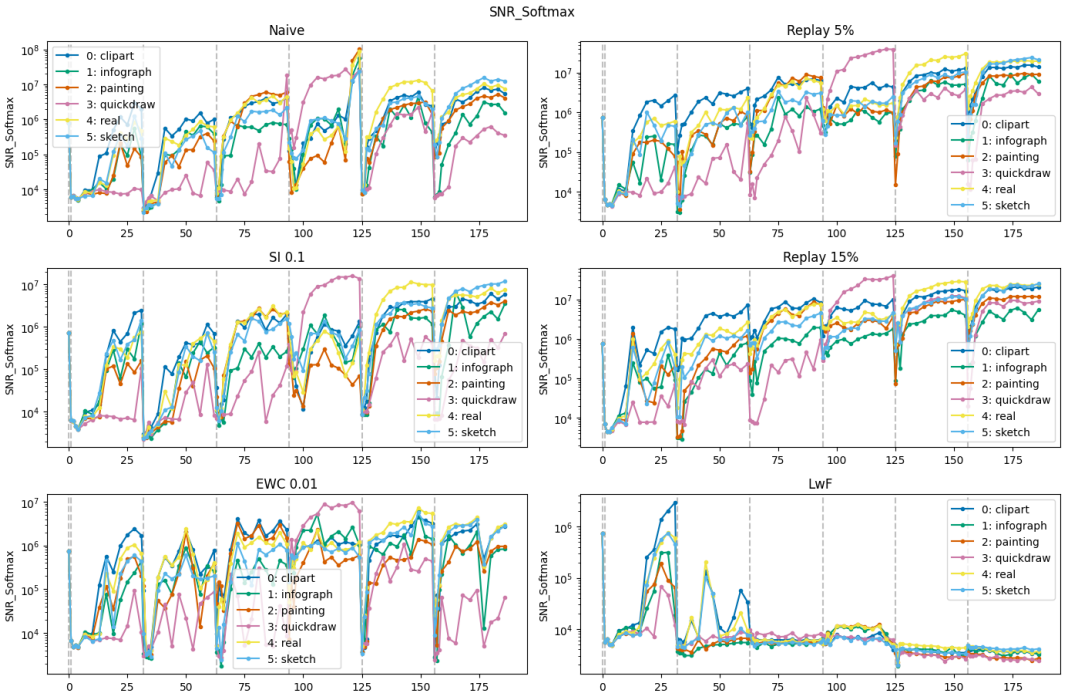


Figure A.21: Signal-to-Noise Ratio of DomainNet - MCD

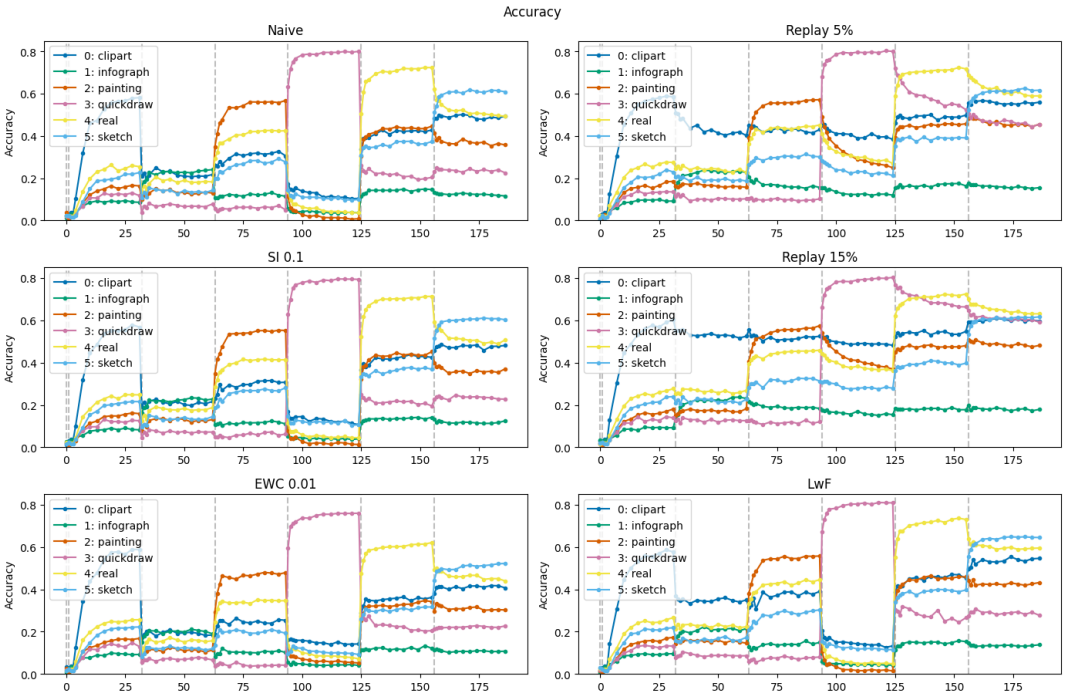


Figure A.22: Accuracy of DomainNet - Ensembles

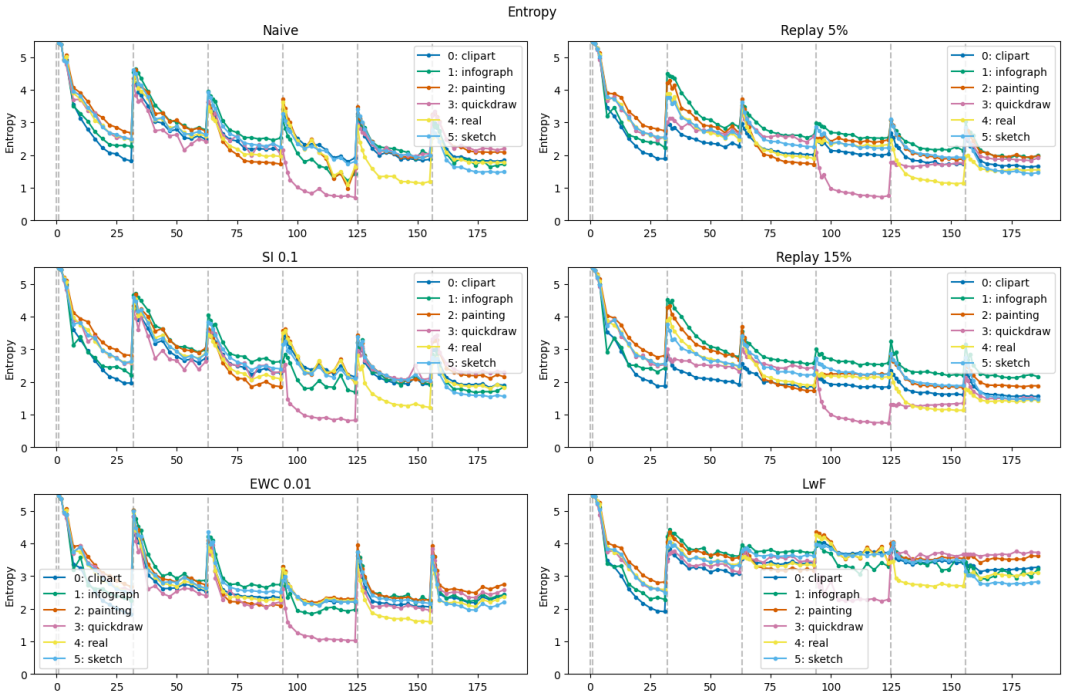


Figure A.23: Entropy of DomainNet - Ensembles

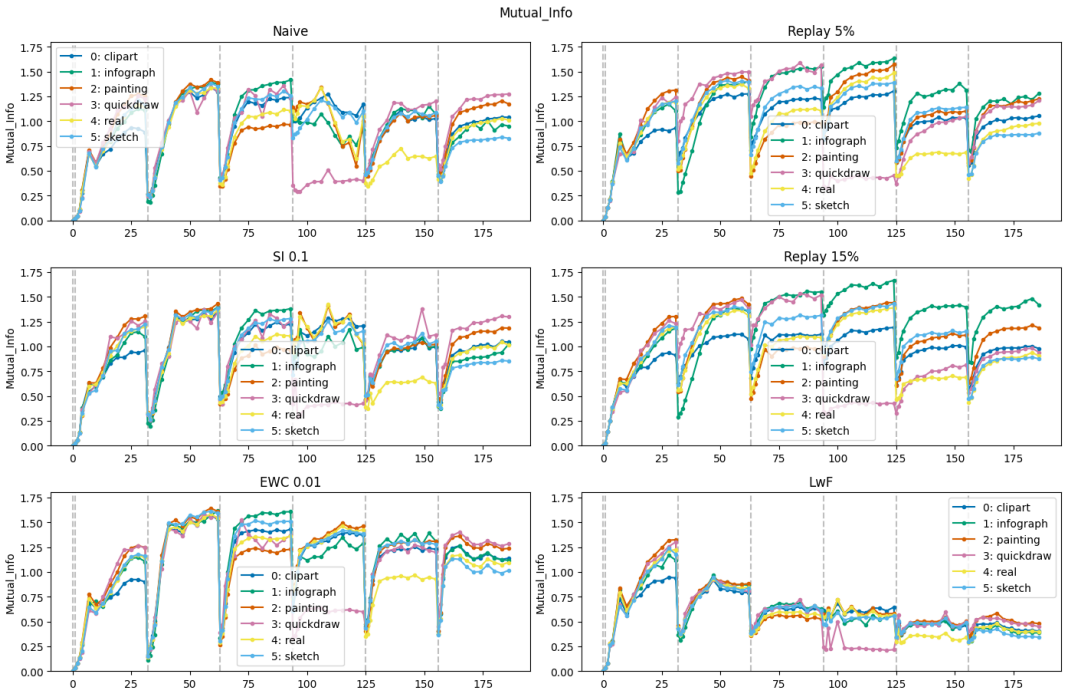


Figure A.24: MI of DomainNet - Ensembles

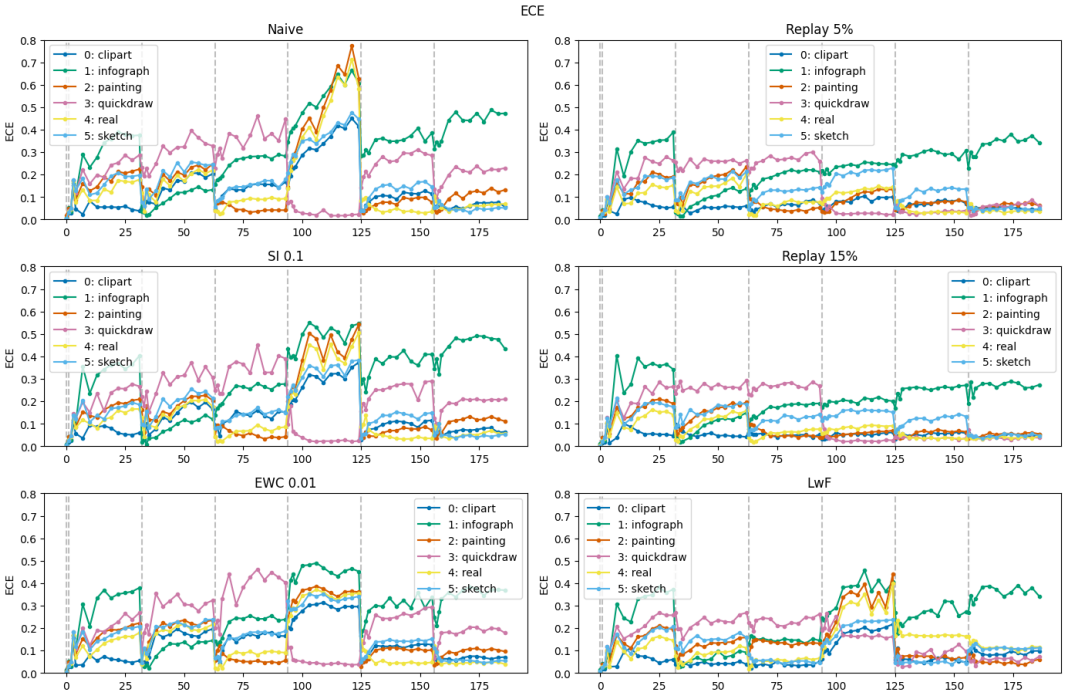


Figure A.25: ECE of DomainNet - Ensembles

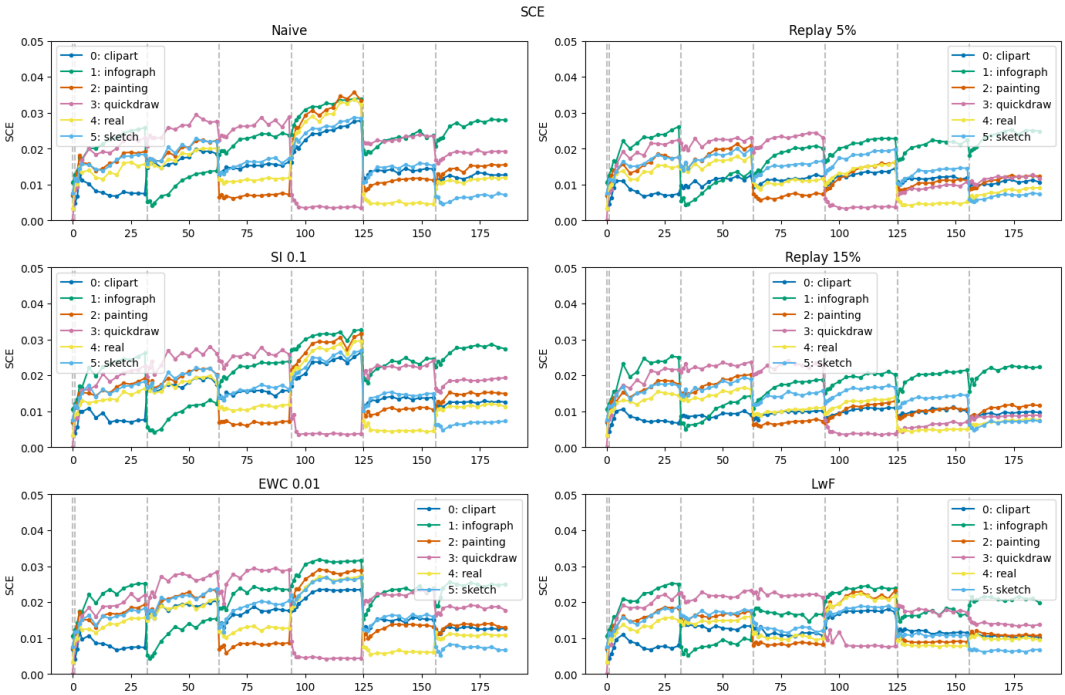


Figure A.26: SCE of DomainNet - Ensembles

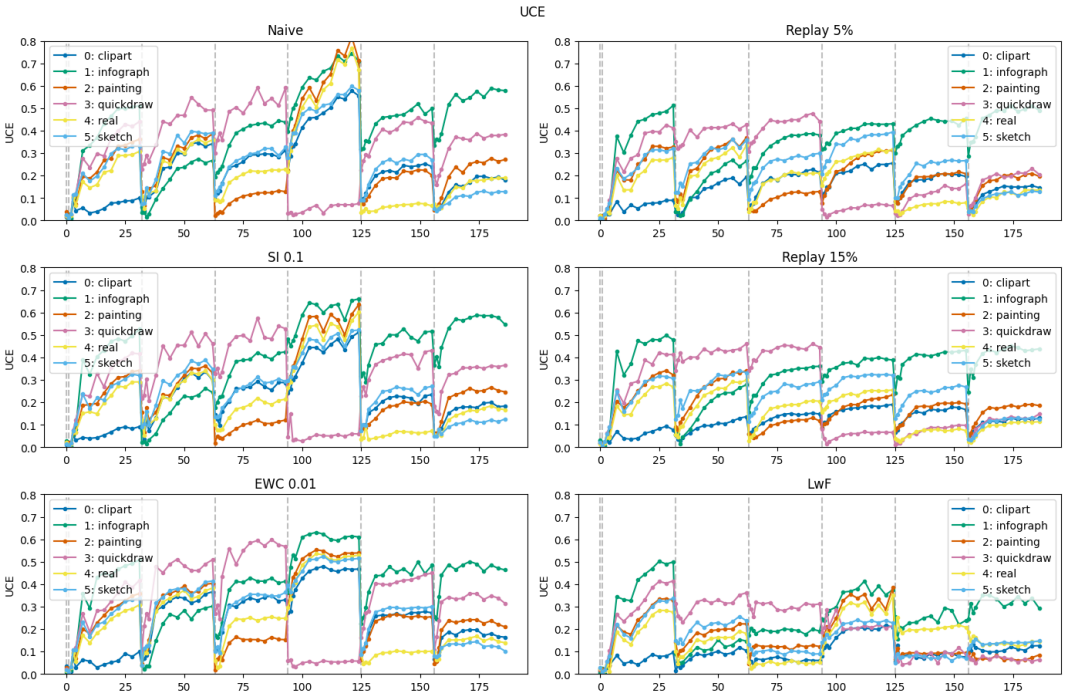


Figure A.27: UCE of DomainNet - Ensembles

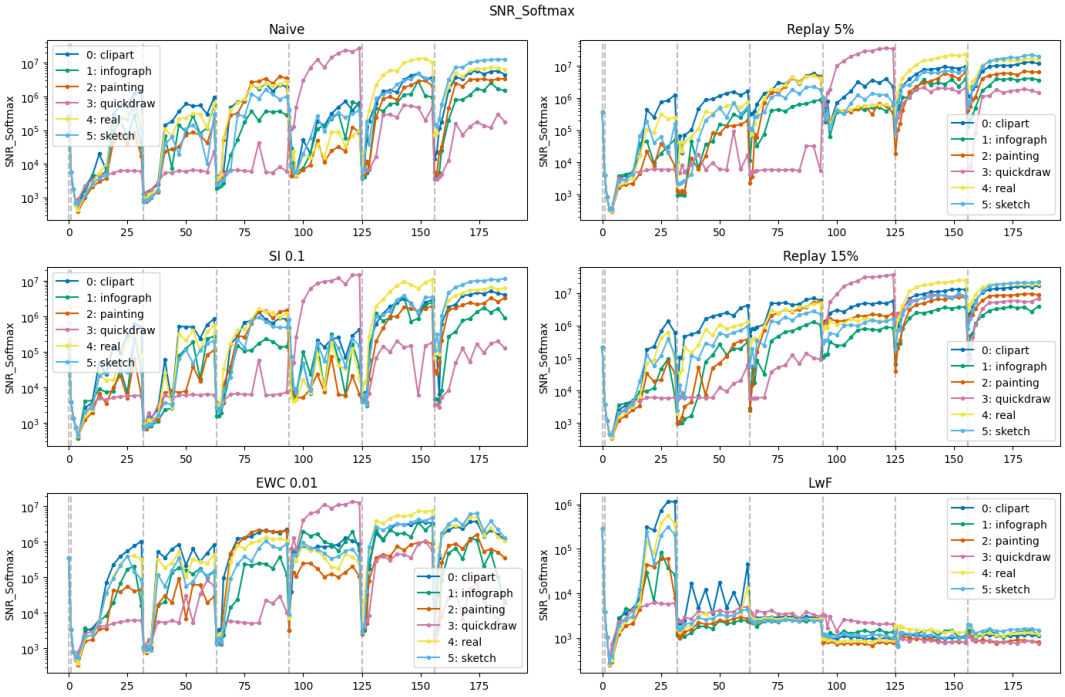


Figure A.28: Signal-to-Noise Ratio of DomainNet - Ensembles