Simon Fraser University

CMPT415 - Special Research Projects

Malware Evolution Detection

Chen Zhao (301308092)

January 11, 2021

Contents

Malware Evolution Detection

Chen Zhao
email cza94@sfu.ca

1 Introduction

Artificial Intelligence (AI) along with its branches namely Machine Learning (ML) and Deep Learning at the moment has made extraordinary success and attracted tremendous researchers in fields such as physics, biology, and manufacturing. Current neural networks are successful on various decision related problems, but they often fail silently and produce overconfident or miscalibrated predictions when there is a shift in the input distribution. This can be summarized as model uncertainty or dataset shifting. Dataset shifting occurs when the environment is updating frequently as data has been collected over a long period of time.

As computer viruses have become ever more sophisticated and troublesome after the introduction of the PC, malware has been experiencing a high speed of evolution which directly increases the complexity for malware detection. The increasingly shorter evolution cycle of malware leads malware detection to be extremely time-sensing in many applications, such as computer virus, worms and so on. Malware evolving every minute increases the complexity to acquire and train an effective model to predict on every new generation of malware. To evaluate the accuracy and uncertainty when processing new malware, uncertainty measurement is critical for risk assessment and decision making which has led to shifts towards the use of Bayesian uncertainty.

Epistemic uncertainty and aleatoric uncertainty are the two main types of uncertainty which are caused by limited training data and knowledge and the natural stochasticity of observations respectively. Epistemic uncertainty is an excellent representation on detecting the uncertainty of a model's predictions; building models which 'know when they don't know'. Bayesian neural networks are one approach to measure epistemic uncertainty which assigns every hyperparameter a probability distribution such as weight and bias. In a Bayesian neural network, all weights and biases have a probability distribution attached to them. When doing classification, every time of network execution yields a new set of sampled weights and biases instead of a single set of output values. The set of output values represents a probability distribution on output values and hence confidence and uncertainty can be evaluated in each of the outputs.

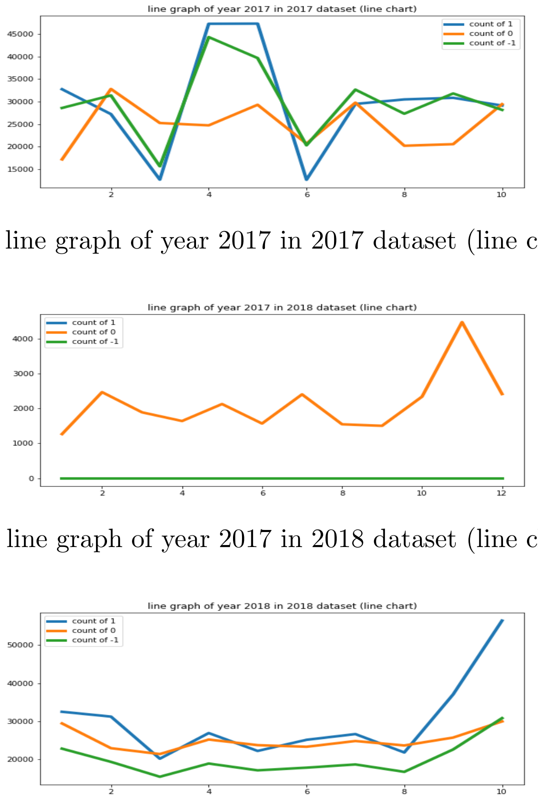
The objective is to measure the uncertainty of malware detection using SWAG, Deep ensemble models and visualize the calibration of confidence by reliability diagrams. Besides, two statistical hypothesis testing approaches, Kolmogorov-Smirnov (KS) test and mean maximum discrepancy(MMD), are used to test concept shift between EMBER data 2017 and 2018.

2 Data Preprocessing

The EMBER [1] dataset is a collection of features from PE files that serve as a benchmark dataset for researchers. The EMBER2017 dataset contained features from 1.1 million PE files scanned in or before 2017 and the EMBER2018 dataset contains features from 1 million PE files scanned in or before 2018. Raw features are extracted to JSON format and included in the publicly available dataset. Vectorized features can be produced from these raw features and saved in binary format from which they can be converted to CSV, dataframe, or any other format. The second EMBER dataset release consisted of version 2 features calculated over samples collected in or before 2018. In conjunction with the second release, the version 2 features are included from the samples collected in 2017. Combining the data from 2017 and 2018 will allow longer longitudinal studies of the evolution of features and PE file types.

To keep EMBER2017 and EMBER2018 datasets integrity and adapt to the training session, data are extracted and separated into sets in terms of months in 2017 and 2018. Both are transformed into input data including 2381 columns through vectorizing and normalizing after dropping the ‘time’ column and selecting only rows labeled 0 and 1. In the flowing line charts, the appearance numbers of 1, 0 and -1 are counted based on their corresponding appearing time. For example, year 2017 in both EMBER2017 and EMBER2018 datasets have similar trend from January to October and data of November and December in year 2017 are included in EMBER2018 dataset, which can be regarded as the extension of EMBER2017 dataset at some extent. Apparently, data labeled in the same year from different dataset have similar distribution, which means data from two dataset are considered corresponding to each other even if more data are stored in the EMBER 2017 dataset. Thus, data in year 2017 from ember 2017 and data in year 2018 from ember 2018 dataset can be used as training dataset for Malware Evolution Detection.

In order to evaluate the uncertainty in EMBER(Endgame Malware BEnchmark for Research) dataset, data associated with different timestamps which will be set as testing units to create the distribution of predicted accuracy yielded by deep ensemble models and statistically used to calculate distance between timestamps. Both approaches will reflect the magnitudes of uncertainty and dataset shifting, which can be used to define if malware evolution appears when the malware data in EMBER2018 has been shifted from EMBER2017 dataset.



(a) line graph of year 2017 in 2017 dataset (line chart)

(b) line graph of year 2017 in 2018 dataset (line chart)

(c) line graph of year 2018 in 2018 dataset (line chart)

Figure 1

After data processing, 2381 columns of features(X) and one column of label(y) are set as input data for both year 2017 and year 2018. I will set 2017 January and February data as training data and 2017 March data as validation data. Then, a distribution of entropy is returned by importing the testing data which includes whole year data of year 2017 or year 2018. In the two approaches illustrated in section (dataset shifting detection), each data of year 2017 and 2018 with 2381 columns of features are tested on Kolmogorov-Smirnov (KS) test and Maximum mean Discrepancy to yields corresponding p-value and raising related hypothesis.

3 Uncertainty Measurement

3.1 SWAG

SWA-Gaussian (SWAG) [2], a simple, scalable, and general purpose approach for uncertainty representation and calibration in Bayesian deep learning. The key idea of SWAG is that the SGD iterates, with a modified learning rate schedule, act like samples from a Gaussian distribution; SWAG fits this Gaussian distribution by capturing the SWA mean and a covariance matrix, representing the first two moments of SGD iterates. I use this Gaussian distribution as a posterior over neural network weights, and then perform a Bayesian model average, for uncertainty representation and calibration.

Standard training of deep neural networks (DNNs) proceeds by applying stochastic gradient descent on the model weights with the following update rule:

$$\triangle\theta_t = -\eta_t\left(\frac{1}{B}\sum_{i=1}^{B}\triangledown_\theta logp(y_i|f_\theta(x_i)) - \frac{\triangle_\theta logp(\theta)}{N}\right)$$

where the learning rate is η, the ith input and label are $\{x_i, y_i\}$, the size of the whole training data is N, the size of batch is B, and the DNN, f, has the weight parameters θ^2. The loss function is a negative log likelihood training does not represent uncertainty in the predictions or parameters θ.

The main idea of SWA is to run SGD with a constant learning rate schedule starting from a pre-trained solution, and to average the weights of the models it traverses. Denoting the weights of the network obtained after epoch i of SWA training θ_i , the SWA solution after T epochs is given by $\theta_{SWA} = \frac{1}{T}\sum_{i=1}^{T}\theta_i$

To maintain a running avearage for rach weight, a diagonal covariance approximation can be used based on the following rules:

$$\overline{\theta^2} = \frac{1}{T}\sum_{i=1}^{T}\theta_i^2$$

$$\sum_{diag} = diag(\overline{\theta_2} - \theta_{SWA^2})$$

where the square of θ_{SWA}^2 and θ_i^2 are applied elementwise.

The resulting approximate posterior distribution, $N(\theta_{SWA}, \sum_{diag})$, is adapted to measure and predict the results in terms of the square of deviation. Thus, the magnitude of uncertainty can be used to evaluate the calibration of our current structure of model and weights to reflect the process of malware evolution.

3.2 Deep Ensemble Model

Deep ensembles [3] have been empirically demonstrated as an advanced approach for optimizing accuracy, uncertainty and out-of-distribution(OOD) of deep learning models. Bayesian neural networks, which learn distributions over the parameters of the network, are theoretically well-motivated by Bayesian principles, but do not perform as well as deep ensembles in practice, particularly under dataset shift. Broadly, there are two classes of ensembles: randomization-based approaches, where the ensemble members can be trained in parallel without any interaction, and boosting-based approaches where the ensemble members are fit sequentially. While deep ensembles were theoretically motivated by the bootstrap and non-bootstrap, deep ensembles trained with random initialization also perform well to measure uncertainty. In the EMBER dataset, a random initialization technique is applied to create distribution of entropy and predict on data with different timestamps. To be specific, deep ensemble models will use 2017 January and February data as training dataset and 2017 March data as validation dataset.

Given the input features x, I use a neural network to model the probabilistic predictive distribution $P_\theta(y|x)$ over the labels, where θ are the parameters of the neural network. Let M denote the number of NNs in the ensemble and $\{\theta_m\}_{m=1}^{M}$ denote the parameters of the ensemble. I mainly use randomization-based approaches to ensemble five (M=5) NNs by assigning different initialized weights and set same parameters θ to train these five NNs on the same training and validating data. After training, these give neural networks are stacked into one ensemble model. By using the deep ensemble models to make predictions on two datasets with different timestamps, the deep ensembles model yields a distribution of entropy for each dataset. Two distributions collected from two testing datasets with different timestamps can be used to distinguish if two testing datasets belong to the same population based on some statistical approaches, such as Kolmogorov-Smirnov (KS) [4] test. KS test is a nonparametric test of the equality of continuous or discontinuous, one-dimensional probability distributions that can be used to compare two samples selected from the same population. To compare if the distribution of entropy from EMBER 2017 and 2018 are different, Kolmogorov-Smirnov test calculates a proper p-value on two distributions of entropy to define if the null hypothesis, the samples drawn from the same distribution, are ought to be rejected, which reflects whether the two testing datasets are from the same distribution.

To compare if the distribution of entropy from EMBER 2017 and 2018 are different, I adopt the Kolmogorov-Smirnov (KS) test to calculate a proper p-value on the variances to define if the null hypothesis, the samples drawn from the same distribution, are ought to be rejected, which reflects whether the two predictive datasets are from the same distribution.

3.3 Reliability Diagrams

Calibrated confidence estimates are essential for model interpretability and stability. Humans have a natural cognitive intuition for probabilities. Good confidence estimates provide a valuable extra bit of information to establish trustworthiness with the user, especially for neural networks, whose classification decisions are often difficult to interpret.

Reliability diagrams [5], are a visual representation of model calibration showing confidence of predicting probability estimates representative of the true correctness likelihood, are clear to visualize the uncertainty and the magnitude of confidence. These diagrams plot expected sample accuracy in terms of a function of confidence. A model is perfectly calibrated, if the diagram should map the identity function. Miscalibration shows when any deviation differed from a perfect diagonal identity function. To plot this diagram in finite samples, true labels, predicted results, and confidence of prediction are input parameters separated into M number of bins. The accuracy and confidence of each bin are yielded by function:

$$acc(B_m) = \frac{1}{|B_m|} \sum_{i_m} 1(\hat{y}_i = y_i) conf(B_m) = \frac{1}{|B_m|} \sum_{i_m} \hat{p}_i$$

While reliability diagrams are useful visual tools, it is more convenient to have a scalar summary statistic of calibration. Since statistics comparing two distributions cannot be comprehensive, previous works have proposed variants, each with a unique emphasis. One notion of miscalibration is the difference in expectation between confidence and accuracy. Expected Calibration Error or ECE approximates by partitioning predictions into M equallyspaced bins (similar to the reliability diagrams) and taking a weighted average of the bins' accuracy/confidence difference. The difference between accuracy and confidence for a given bin represents the calibration gap.

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} acc(B_m) - conf(B_m)$$

4 Dataset shifting Detection

4.1 Kolmogorov-Smirnov (KS) test

To compare if the distribution of entropy are different, I adopt the Kolmogorov-Smirnov (KS) test [6], univariate testing, to calculate a proper p-value to define if the null hypothesis, the samples drawn from the same distribution, are ought to be rejected, which reflects whether the two datasets are from the same distribution. Kolmogorov-Smirnov test can also directly detect dataset shifts based on each feature. Through combining KS test with Bonferroni Correction and collect p-values for each features, concept shift can be detected from the comparison between the mean of p-values and pre-defined alpha divided by the number of features.

$$Z = sup_z |F_p(z) - F_q(z)|$$

Where the largest difference Z of the cumulative density functions(CDF) and F_p and F_q are the empirical CDFs of the source and target data, respectively. Under the null hypothesis, Z follows the Kolmogorov distribution.

After data processing, 2381 columns of features are set as input data. KS test will be applied to all features sequentially which yields a list of p-values and raise the issue of multiple hypothesis testing. As I cannot make strong assumptions about the (in)dependence among the tests, I rely on a conservative aggregation method, notably the Bonferroni correction, which rejects the null hypothesis if the minimum p-value among all tests is less than α/K (where α is the significance level of the test). While several less conservative aggregations methods have been proposed, they typically require assumptions on the dependencies among the tests.

$$avg(p) < \frac{\alpha}{N}$$

where N is the column number of dataset (amount of features)

If the value of average p-value is less than the divided significance level, the null hypothesis should be rejected which means the two datasets are from different distributions.

4.2 Maximum mean Discrepancy

Except testing the distribution of entropy and multiple univariate testing, maximum mean discrepancy(MMD) [6] can be used to test dataset shift for multivariate two-sample testing. Since MMD is an alternative method of Kolmogorov-Smirnov and Bonferroni Correction, the same input and output values in KS testing are imported and yielded, which use 2381 columns of features (vectorized 2017 and 2018 ember data without labels) as input and a p-value as output result. Based on the samples of two distributions, MMD allows us to distinguish between two probability distributions p and q based on the mean embeddings $\mu_p$and μ_q of the distributions in a reproducing kernel Hilbert space F, formally

$$MMD(F,p,q) = |\mu_p - \mu_p|_F^2$$

$$MMD^2 = \frac{1}{m^2-m}\sum_{i=1}^{m}\sum_{j\neq 1}^{m} k(x_i,x_j) + \frac{1}{n^2-n}\sum_{i=1}^{n}\sum_{j\neq 1}^{n} k(x'_i,x'_j) - \frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} k(x_i,x'_j)$$

where I use a squared exponential kernel $k(x,\tilde{x}) = e^{||x-\tilde{x}||^2}$ and set σ to the median distance between points in the aggregate sample over p and q . A p-value can then be obtained by carrying out a permutation test on the resulting kernel matrix.

5 Experiments

5.1 SWAG

A distribution of entropy is returned by SWAG by importing training data which includes 2381 columns of features and one column of labels from Janurary, February, March in 2017. To detect if year 2017 data and year 2018 data are sampled from the same distribution, the two distributions of entropy of year 2017 and 2018 are tested by Kolmogorov-Smirnov (KS) test with testing data EMBER2017 and EMBER2018 which yields a p-value. A feed-forward neural network are implemented, applied to SWAG [2] and used to measure the uncertainty by using distribution of entropy.

Given the values of entropy calculated by SWAG model, the average entropy value of year 2018 data (0.6469766409268951) is slightly greater than the average entropy value of year 2017 data (0.6255155877562545). Comparing if these two distributions of entropy are calculated and collected from the same dataset through KS test, the p value (0.013475889875863678) is less than the corresponding p-value, for example, 0.05. Thus, the null hypothesis, the samples drawn from the same distribution, is ought to be rejected, which potentially reflects year 2017 in EMBER2017 differs from year 2018 in EMBER2018.

Entropies table with five samples					
EMBER 2017 (year 2017)	0.62316189	0.60817403	0.61578962	0.62316189	0.59983093
EMBER 2018 (year 2018)	0.5822031	0.5822031	0.69314717	0.5822031	0.5822031
P value:0.013475889875863678					

5.2 Deep Ensemble Models

By importing the same input data as SWAG, the same output, a distribution of entropy, is returned by deep ensemble model. To detect if year 2017 data and year 2018 data are sampled from the same distribution, the two distributions of entropy tested on year 2017 and 2018 data are measured by Kolmogorov-Smirnov (KS) test which yields a p-value used to determine the hypothesis of KS test. I fully construct five neural networks, stacking them into ensemble model and use KS test from scipy.stat [7] to calculate p-values.

By constructing a simple structure of single neural network, each single neural network is trained and validated on 2017 January, February and March data. After stacking five single neural networks into one bayesian neural network, the uncertainty can be evaluated based on the distribution of entropy returned from ensemble model. To measure if the two samples from year 2017 and 2018 are from the same dataset, two distributions of entropy can be yielded by this ensemble model and imported to a KS test to calculate a p-value.

Five examples from distributions of Entropy						
EMBER 2017 (year 2017)	0.187127873	0.034988113	0.187127830	0.187127873	0.187127873	...
EMBER 2018 (year 2018)	0.187127873	0.187127873	0.187127873	0.187127830	0.034988112	...
P value: $2.172\,874\,402\,159\,926\,4 \times 10^{-40}$						

Given the table above, the null hypothesis, the samples drawn from the same distribution, is ought to be rejected, since the corresponding value (0.56327) when alpha equals to 0.05 is larger the p-value ($2.172\,874\,402\,159\,926\,4 \times 10^{-40}$) resulted by KS test. Thus, year 2017 in EMBER 2017 and year 2018 in EMBER 2018 are from different sample or EMBER2018 has been experienced some evolution.

5.3 Reliability Diagrams

In this section, I measure the calibration for both single neural network stacked in deep ensemble model and overall deep ensemble model by visualization of reliability diagrams and quantify the calibration by Expected Calibration Error (ECE) value. The attached reliability graphs [8] are ploted by the function provided by Matthijs Hollemans and I implement the ECE function.

Given figure 3 in appendix section two, all five single neural network models result in high accuracy with most of the predictions aggregated over ninety percent and the deep ensemble model is slightly over confident, since the average of accuracy is less than the identity function. Their corresponding ECE values are ranged between 1.33% to 3.38%.

	Sub Model 1	Sub Model 2	Sub Model 3	Sub Model 4	Sub Model 5
ECE	1.85%	1.33%	2.87%	3.38%	1.90%

By using ECE as the primary empirical metric to measure calibration, the ECE value of deep ensemble model (0.67%), shown in figure 2, is less than the lower bound of the above ECE range. The lower ECE value represents this deep ensemble model is more calibrate than every single sub neural network.

5.4 Kolmogorov-Smirnov (KS) test

Kolmogorov-Smirnov (KS) test, univariate testing, are evaluated on every feature of ember 2017 and 2018 to subsequently combine the p-values from each test, raising the issue of multiple hypothesis testing. Then Bonferroni Correction is used to counteract the problem of multiple comparisons which rejects the null hypothesis if the minimum p-value among all tests is less than α divided by number of features(K). By using KS test function from scipy.stat [7], I construct function to apply KS test to all test of features sequentially and yield a averaged p-value based on Bonferroni Correction.

Given the experiments result, the significance level(0.05) divided by K is

$$2.099\,958\,000\,839\,983\,3 \times 10^{-5}\,\mathrm{m\,s^{-1}}$$

and the minimum p-value among all tests selected by Bonferroni correction is 0. As the calculated p value is smaller than the averaged significance level, the null hypothesis of Kolmogorov-Smirnov test will be rejected that these two dataset are drawn from different populations or the year 2018 malicious data has experienced some evolution.

5.5 Maximum mean Discrepancy

Maximum mean discrepancy imports the year 2017 and year 2018 to calculate a distance on the space of probability measures and then carrying out a permutation test on the result kernel matrix. Since MMD is for multivariate two-sample testing which needs more space then KS test, tremendous space are required to test on EMBER 2017 and EMBER 2018 dataset. By deploying MMD on two sub-sample datasets , January and February data of year 2017 are used to determine if January data and February data are from same distribution. Under the limitation of resources, Maximum mean discrepancy is still executing on data appeared in Jan and February 2017. The function of MMD is constructed by a PyTorch library torch-two-sample [9].

6 Conclusion

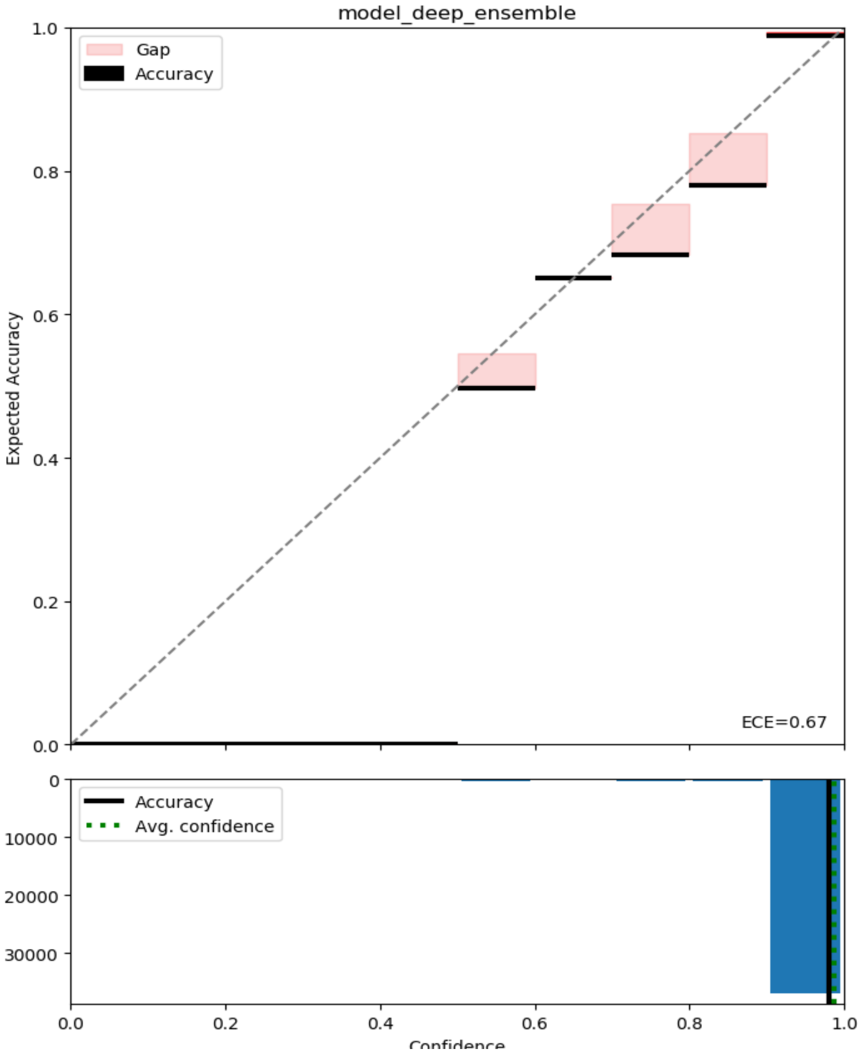
In this paper, I apply SWAG and deep ensemble approaches to measure the uncertainty of the predictions. SWAG model iterates SGD from a Gaussian distribution to yield a distribution of entropy and deep ensemble models with random initialization build a distribution of entropy as well. Both distributions from year 2017 in EMBER2017 dataset and year 2018 in EMBER2018 are tested by Kolmogorov-Smirnov (KS) test with null hypothesis whether these sample distributions are from the same population. Based on my result, this null hypothesis are rejected, which potentially reflects the testing data(year 2017 and 2018) are from different distribution or malware data in EMBER 2018 experience some evolution comparing to EMBER2017 data.

Besides, I use reliability diagrams to visualize the calibration of each single neural network stacked in deep ensemble models and the calibration of deep ensemble model. The difference of calibration is hard to distinguish from the diagrams, but the magnitude of calibration evaluated by expected ealibration error or ECE values are measurable. Since the ECE values of all sub neural networks of deep ensemble model are larger than the ECE value of the deep ensemble model and all these ECE values of sub-models are also lower than 5%, the deep ensemble model is calibrate and trustable.

For the conclusion of section four(Dataset shifting Detection), both Maximum mean discrepancy and Kolmogorov-Smirnov (KS) test directly detect dataset shifting by importing vectorized EMBER 2017 and EMBER 2018 dataset. Kolmogorov-Smirnov test are tested on each features sequentially, while MMD are tested on all features directly as a multivariate testing. The p-value returned by KS test is less than the corresponding values of significant value, which illustrates that data of year 2017 and year 2018 differs from each other. Under the limitation of resources, Maximum mean discrepancy is currently not executable on EMBER datasets but is executing on sub dataset, such as data appeared in Jan and February 2017.

7 Appendix

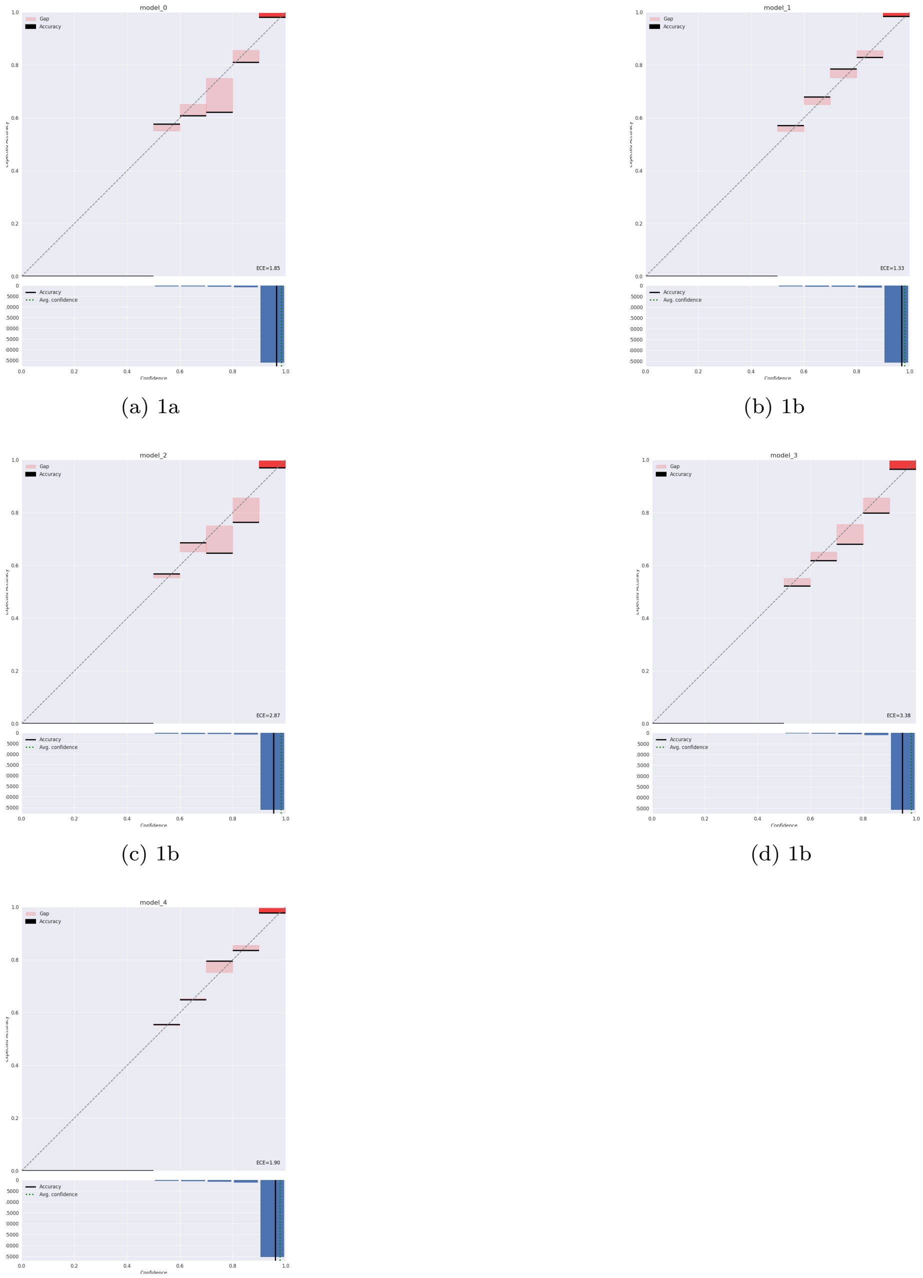
7.1 Reliability Diagrams for Deep ensemble model



(a) Reliability Diagram of Deep Ensemble Model

Figure 2

7.2 Reliability Diagrams for five sub neural network of Deep ensemble model



(a) 1a

(b) 1b

(c) 1b

(d) 1b

(e) 1b

Figure 3: plots of reliability diagrams for five single neural networs

References

[1] H. S. Anderson and P. Roth, "EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models," *ArXiv e-prints*, Apr. 2018.

[2] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, "A simple baseline for bayesian uncertainty in deep learning," in *Advances in Neural Information Processing Systems*, pp. 13153–13164, 2019.

[3] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles." `https://arxiv.org/abs/1612.01474`, Dec 2016.

[4] Wikipedia contributors, "Kolmogorov–smirnov test," 2020. [Online; accessed 2-December-2020].

[5] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks." `https://arxiv.org/abs/1706.04599`, Jun 2017.

[6] S. Rabanser, S. Günnemann, and Z. C. Lipton, "Failing loudly: An empirical study of methods for detecting dataset shift." `https://arxiv.org/abs/1810.11953`, Oct 2018.

[7] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[8] M. Hollemans, "reliability diagram." `https://github.com/hollance/reliability-diagrams`, July 2020.

[9] josipd, "torch-two-sample." `https://github.com/josipd/torch-two-sample`, May 2020.