An Implementation of Altered Weighted Majority Algorithm on Machine Learning Models

Zhihao Chen, Haikang Deng, Zhanyi Ding, Zehao Xiao

{zhihao, haikang, zhanyi, xzhzhzha}@live.unc.edu

1. Introduction

We introduce Altered Weighted Majority Algorithm (AWMA) that aggregates predictions given by multiple experts based on their performances. As the original Weighted Majority Algorithm (Littlestone & Warmuth, 1994) is designed for binary cases, we extend the algorithm so that it applies to multi-class settings. In AWMA, the hyperparameter ϵ is tuned down to reduce the effect caused by performance difference among experts. Otherwise, after more than 10,000 rounds of update, the algorithm will be assigning all weights to the best expert regardless of how other experts behave. Testing the algorithm on CIFAR-10 (Krizhevsky, n.d.) dataset, we showed that, when given a set of comparable experts, AWMA draws collective wit of experts and adds robustness to produced prediction, whereas in other cases, when given a set of experts with uneven performances, AWMA finds the best expert and converges to its predictions.

2. Design

In this project, we apply AWMA on five machine learning models who predict the label of given images. The CIFAR-10 dataset consists of 60,000 32×32 color images in 10 classes—airplane, automobile, cat, dog, etc. The dataset is randomly divided into four subsets, model training set, model test set, WMA training set, and WMA test set, with 25000, 5000, 20000, 10000 examples respectively. The two former subsets are responsible for training and testing of machine learning models, while the two latter subsets are reserved for AWMA.

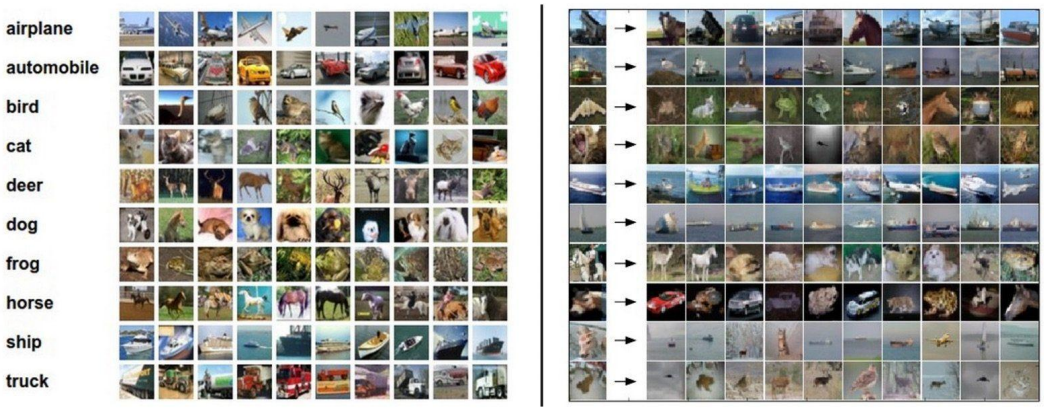

Figure 1. CIFAR-10 dataset examples (Krizhevsky, n.d.)

Five machine learning models used are: Logistic Regression, Multi-layer Perceptron, Convolutional Neural Network, Residual Network, and SVM. They are trained and evaluated on model training set and model test set before AWMA kicks in. In practice, these five models behave differently and produce predictions that are not alike.

- **Logistic Regression (LR):** a combination of ten logistic regressors. The label with highest logit is selected as prediction.
- **Multi-layer Perceptron (MLP):** a neural network with input layer, one hidden layer, and output layer. Image features are unrolled to 1d vector and fed into the network for prediction.
- **Convolutional Neural Network (CNN):** a neural network that extracts features using convolution and pooling. Generally works well with images.
- **Residual Network (ResNet):** a neural network consists of convolution blocks and residual connection. With high potential in representing images, residual network is widely used in computer vision tasks. Note: two versions are provided to satisfy experiment setting—ResNet-tiny and ResNet-base (He, et. al, 2015).
- **Support Vector Machine (SVM):** a supervised machine learning method based on support vectors (Hearst et. al, 1998).

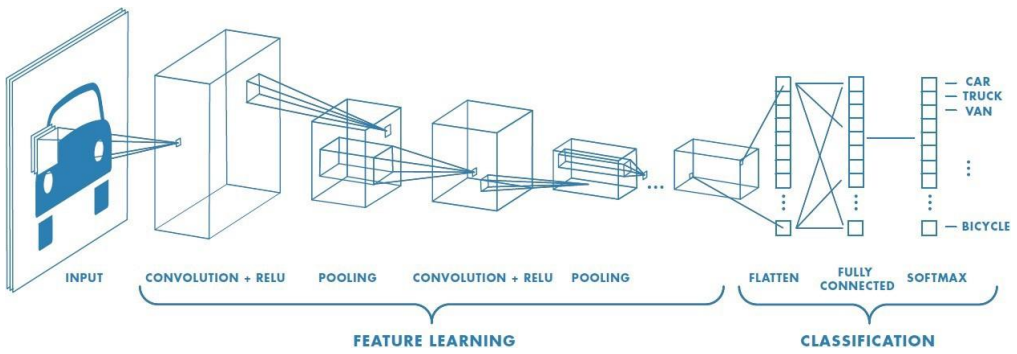

Figure 2. Convolutional Neural Network for Classification (Saha, 2021)

In the balanced setting, three of the five models (MLP, CNN and ResNet-tiny) have comparable accuracy around 50%. Whereas in the skewed setting, ResNet-tiny is replaced by ResNet-base, which has accuracy of 77%, while other models remain unchanged. We conduct experiments under these two settings and see how AWMA reacts.

Specifically, for both balanced setting and skewed setting, we first perform weight update and weighted prediction on the AWMA training set, with each expert starting with equal share of weight. Each example in the dataset is defined as one iteration of AWMA. The procedure is given as

1. Input Image to models and collect their predictions.

2. Aggregate predictions from models based on their weights and select the class that has the most votes as AWMA prediction.
3. Observe true labels and penalize models that give wrong predictions by scaling down their weights.
4. Repeat 1-3 in the next iteration.

By doing so we generate AWMA predictions for the dataset, which could be analyzed for accuracy and compared to each model's accuracy. In addition, a set of weights is produced which reflects the performance of each model in the AWMA training set. Note that during experiments, AWMA is not given the information of experts' accuracies on any of the dataset. It has to figure out how these experts perform and work out an aggregate prediction by itself. To further investigate the AWMA's capability, we evaluate it on the AWMA test set. This time, no weight update is performed, we freeze the set of weight produced previously on the AWMA training set.

In addition, the randomized version of Altered Weighted Majority Algorithm (RAWMA) is constructed and tested on the same trials as AWMA. RAWMA is similar to AWMA except that instead of aggregating weighted predictions, it draws an expert from the uniform distribution of expert weights, where experts with higher weight are more likely to be sampled. RAWMA takes the prediction suggested by that expert.

3. Experiments

3.1 Equal Performance Experts Experiments

In the following experiments, we implemented the AWMA with five experts. We further divided the algorithm into randomized and deterministic versions. Note that the hyperparameter ϵ also played an important role in the process of choosing experts and making predictions. We'll analyze it later. We call it the *Altered* Weighted Majority Algorithm because we now have ten potential classes for each prediction, i.e. (0: airplane, 1: automobile, … , 9: truck). In examples we have seen before, WMA was designed for binary classification problems (yes/no), in which case either class receiving more than 50% weighted vote will be considered as the prediction given by WMA. However, in the altered version, with 10 possible classes, we may not have a class that receives more than 50% weighted vote given by experts. Therefore, instead of requiring a class to have more than a certain portion of vote, we change AWMA to select the class receiving the most weighted vote, which is implemented using argmax method. In the following experiments when there are experts whose performances are comparable, we call it the Equal Performance Experts Experiment (EPEE).

After we trained the five deep learning models on the model training set, we got accuracy $[0.37, 0.48, 0.49, 0.49, 0.24]$ on the model test set, of which MLP, CNN, and ResNet-tiny performed almost equally well. We first run the deterministic AWMA on its training set. The

initial weights for experts are $[1, 1, 1, 1, 1]$, which normalizes to $[0.2, 0.2, 0.2, 0.2, 0.2]$, representing that each model has the same amount of voice in the prediction. For each image in the training set, AWMA aggregates model predictions based on their weight, and chooses the label with the highest weights voted as its prediction. After observing the actual label for the example, experts who gave wrong prediction in round t will get their weights reduced by a positive factor $\epsilon < 1$.

In the following example, we illustrate a round of AWMA updates. The way we determine the predicted result for each individual observation is like a voting process: each expert has a chance to vote/predict for one label with his own normalized weight; at the end of each round, the label which received the most voting weight will be accepted as the predicted result for this observation. For example, if the current weights are $[0.1, 0.2, 0.2, 0.4, 0.1]$, and the predictions are $[1, 2, 2, 1, 6]$, then the prediction for this individual observation would be label 1, since label 1 received a weight of $0.1 + 0.4 = 0.5$, which is higher than weight received by any other labels. Then, we observe the true label to be 1, which corresponds to our predicted result. Therefore, the weight of expert #1 and expert #4 stays the same, while weight of other experts would decrease by a factor of (1-ϵ).

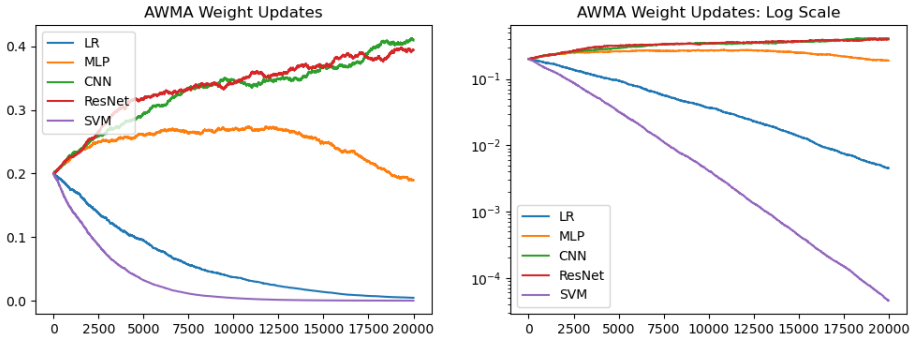

Figure 3. Weight updates of AWMA on training set in EPEE

After iterating over 20000 data examples, we got the final weights for each expert as $[0.045, 0.19, 0.41, 0.395, 0.00045]$. Then we fix these weights and conduct the test on the AWMA test set, which yields an accuracy of 0.53. Notice the prediction result improved by 0.04 compared to the best single experts.

In our randomized altered weighted majority algorithm(RAWMA) process, weights for expert are initialized to $[1, 1, 1, 1, 1]$, which normalizes to $[0.2, 0.2, 0.2, 0.2]$. In each round, RAWMA samples an expert with probability w_i, which is the normalized weight of the corresponding expert. RAWMA then takes this expert's prediction as its own prediction for this round. Code implementation of this sampling process is simple—we generate a random number ranged from 0 to1. If it falls in the interval $[0, w_1)$, we sample expert #1. If the number falls in

$[w_1, w_1 + w_2)$, we sample expert #2, etc. After observing the true label in every round, we update weights of the experts who made wrong prediction by multiplying a factor of 1-ϵ.

3.2 Skewed Setting

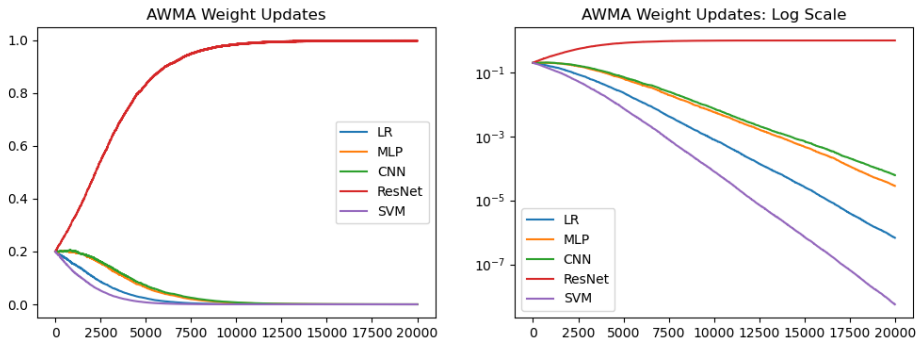

Figure 4. Weight updates of AWMA on training set in Skewed Setting

In the skewed setting, ResNet-tiny is replaced by ResNet-base which has accuracy of 77.34%, which is significantly higher than the other four models. We apply AWMA on the same training and test datasets as in EPEE and result in very different conditions. As shown in the figure above, as we iterate over the dataset, ResNet-base's weight skyrockets and converges to 1 while other models' weights decrease to 0. The algorithm assigns ResNet-base a dominant weight so that its prediction becomes the AWMA's prediction. Experiments in the skewed setting shows that AWMA has prediction accuracy of 75.68% on the training set, and 76.48% for the test set (freezing weights), which are close to accuracy of the ResNet-base model. In this case, due to the big performance gap between models, AWMA fails to achieve a better performance than the single best model. Yet, it finds the best model promptly and achieves a similar accuracy. In addition, RAWMA, in the skewed setting, achieves slightly lower accuracy on the training set and equally good accuracy on the test set, when weights are fixed.

Comparable Setting

Content\Model	LR	MLP	CNN	ResNet-tiny	SVM	**AWMA**	**RAWMA**
Accuracy (Model sets)	38.44%	48.98%	49.84%	49.56%	25.46%		
Accuracy (AWMA train set)	37.21%	47.62%	49.75%	49.65%	24.40%	**52.56%**	47.86%
Accuracy (AWMA test set)	37.10%	48.48%	49.52%	49.90%	25.20%	**53.08%**	49.69%
Weight	4.53529E-03	**1.90340E-01**	**4.09770E-01**	**3.95310E-01**	4.55570E-05		

Skewed Setting

Content\Model	LR	MLP	CNN	ResNet-base	SVM	**AWMA**	**RAWMA**
Accuracy (Model sets)	38.44%	48.98%	49.84%	77.34%	25.46%		
Accuracy (AWMA train set)	37.21%	47.62%	49.75%	**76.76%**	24.40%	75.68%	72.36%
Accuracy (AWMA test set)	37.10%	48.48%	49.52%	**76.48%**	25.20%	**76.48%**	76.47%
Weight	6.77846E-07	2.84483E-05	6.12440E-05	**9.99910E-01**	5.61873E-09		

(Table 1. Experiment results)

4. Result analysis

We could see from the result that using the AWMA algorithm with an ϵ of $\sqrt{\log(5)/20000}/5$ for the EPEE, we successfully improved the prediction accuracy to 0.53 for the test set, which is higher than each individual models' accuracy. Since we had three models with similar accuracy yet each of them might be good at classifying different labels, the AWMA allows us to leverage each model's advantages and thus improving the robustness of our prediction, instead of simply trusting the best individual model. For example, using the final weight $[0.045, 0.19, 0.41, 0.395, 0.00045]$, if the predicted result for each individual model is $[1, 2, 5, 2, 1]$, then our AWMA predicted result would be label 2, instead of label 5 suggested by model #3.

However, for the skewed setting, where there is an expert with significantly better accuracy than others, its weight will be dominant at the end of the weight update. Like in our experiment, the weight for model 4 becomes 0.9. In fact, if the weight of an individual expert becomes greater than 0.5, AWMA predicted result would simply be the same as the best expert, because its individual weight is already greater than the weight sum of all other experts, which guarantees that in each round the predicted result will be the same as the best model. For such a situation, AWMA would fail to increase the robustness of the model. Nevertheless, it could serve as a method for identifying the best model by checking the final weight.

Another important thing to notice is the effect of choosing different ϵ's. For EPEE, if we just use an $\epsilon = \sqrt{\log(5)/20000}$, the final weight of the best expert would be slightly over 0.5. Since the dataset is large, a slight difference in accuracy would result in a big gap between weights. Therefore, we reduce ϵ and make sure none of the individual experts have final weight greater than 0.5. Because of the EPEE setting, such final weights would allow the AWMA to increase the robustness of our prediction. The ϵ value we used for our AWMA model is $\sqrt{\log(5)/20000}/5$, as larger ϵ would make the best expert weights greater than 0.5, yet lower ϵ would reduce the AWMA prediction accuracy.

RAWMA does not work well in EPEE, which is not at all surprising. Because RAWMA is not using the combined wisdom of several models; in essence, it samples from the group of experts and chooses one exact model to trust. Therefore, on the testing set, RAWMA's accuracy can be calculated as an expectation: it is the weighted sum of individual models' accuracies. Therefore, theoretically speaking, RAWMA would not yield an accuracy higher than the best model, if not in adversarial settings. Our two experiments(EPEE and Skewed Setting) both confirmed this conclusion, even though the difference between the best expert and Skewed Setting is smaller. As there is one dominant model with final weight close to 1, and the randomization process thus becomes meaningless since it keeps choosing the dominant expert as the best one. We get confirmation from the testing result: the accuracy is 0.7647, which is very close to that of the best model.

5. Conclusion

For scenarios where several best models are comparable, AWMA is a good strategy to increase the robustness of the prediction by taking into consideration the collective wisdom of each individual expert. Note that we should tune the hyperparameter ϵ to be small enough that none of the experts would have a final weight > 0.5, yet large enough to best increase the prediction accuracy. For scenarios where there is a dominant expert, AWMA could find the best expert and converge to its predictions. Whereas, RAWMA generally has equal or worse performance than the best expert unless in adversarial settings.

Reference

1. Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2015). *Deep Residual Learning for Image Recognition.*

2. Hearst, M., Dumais, S., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their Applications, 13(4), 18-28.*

3. Krizhevsky, A. (n.d.). *CIFAR-10 Dataset.* CIFAR-10 and CIFAR-100 datasets. Retrieved May 2, 2022, from https://www.cs.toronto.edu/~kriz/cifar.html

4. N. Littlestone, & M.K. Warmuth (1994). The Weighted Majority Algorithm. *Information and Computation, 108(2), 212-261.*

5. Saha, S. (2021, December 7). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. Medium. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

6. *Kaggle: Your Home for Data Science*. (n.d.). Kaggle. https://www.kaggle.com/code/scratchpad/notebook852b8a91bc/edit

7. *Training a Classifier — PyTorch Tutorials 1.11.0+cu102 documentation*. (n.d.). PyTorch. https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

8. Kuangliu. (n.d.). *GitHub - kuangliu/pytorch-cifar: 95.47% on CIFAR10 with PyTorch*. GitHub. https://github.com/kuangliu/pytorch-cifar

Appendix

AWMA

Group6

May 5, 2022

- **Input:** $0 < \epsilon < 1$
- **Initialize:** $w_j \to 1, j = 1, \ldots, N$
- **Iteration:** For $i \in [T]$:
  - $\widehat{W_{k,i}} = \sum_{j=1}^{N} \mathbb{I}_{p(j)=k} w_{j,i}$, for $j \in [T],\ k \in [C]$, where C is the number of classes
  - $\widehat{y_i} = \arg\max_k \widehat{W_{k,i}}$
  - Observe real value $y_i \ \Rightarrow w_{j,i+1} = w_{j,i}(1-\epsilon)^{\mathbb{I}_{(y_i \neq \widehat{y_{j,i}})}}$, where $\widehat{y_{j,i}}$ is the prediction given by expert j in round i
- **Output:** final weight.