Improving Generalizations of Deep Learning Using Angular Visual Hardness

Chunshan Liu - CL74@rice.edu   Huiming Lin - HL68@rice.edu

1. Introduction

Convolutional neural network has reached human-level accuracy in the past few years. But insights into this black box is still limited. For the image classification problem, some studies ([1]; [21]; [15] in [1]) try to explain the behavior of Convolutional Neural Networks (CNN) by measuring model hardness which indicates the classification difficulty of an image for a CNN. By examining what kinds of images are considered hard for a model, we can get a deeper understanding of how the model works, and design new models and training methods to improve classification accuracy.

A recent paper [1] proposes the Agular Visual Hardness (AVH) score to measure the classification hardness of an image. The paper states several nice properties of the index. The most important property for this project is that AVH is consistent with model generality. In other words, models with better generalization ability (smaller test error) have lower AVH score throughout the training process. This indicate the possibility to design better loss functions and better training processes using AVH. As far as we know, all the existing researches on AVH focuses on its properties based CNN for the ImageNet classification problem. [2] uses an idea that is similar to AVH, but it redefines the dot product as the product of a magnitude function and a angle function for every neuron, which is very different from our project.

The rest of the report is divided into five parts. Part two gives the definition of AVH and lists the properties of AVH score. In part three, we discuss how to incorporate AVH into the loss function in image classification problems. The performance of new loss functions is examined on CIFAR10 with a modified LeNet-5. In the next part, we propose a new training process called Smart Augmentation which utilizes the AVH scores of images. We observe an improvement in classification on both modified AVH loss and Smart Augmentation. In part 5, we extend AVH from image classification using CNN to other deep learning models and applications. The example we use is the text classification problem based on Recurrent Neural Network (RNN). Part 6 is the summary of the project and future work.

2. Angular Visual Hardness

2.1 Definition

Angular visual hardness is a score defined on CNN with softmax cross-entropy loss for image classification problems. Denote N as the sample size, $i = 1, \dots, N$, and the number of image categories as C, $j = 1, \dots, C$. Figure 1 ([6]) demonstrates a CNN model used for image classification. It contains an input layer, several hidden layers and an output layer. Denote the output feature of the last hidden layer as a row vector a^i. In the output layer,

$$f^i = W \cdot a^{i\prime},$$

where W is a weight matrix with C rows $w_1, \dots, w_C$, and f^i is the logit. The softmax function with cross-entropy loss is then

$$L = \frac{1}{N}\sum_i L^i = \frac{1}{N}\sum_i -log\left(\frac{e^{f^i_{y_i}}}{\sum_j e^{f^i_j}}\right).$$

Decompose the dot product in f we get

$$L_i = -\log\left(\frac{e^{||w_{y_i}||\cdot ||a^i||\cdot\cos(\theta_{y_i})}}{\sum_j e^{||w_j||\cdot ||a^i||\cdot\cos(\theta_j)}}\right),$$

where $||w_{y_i}||$ is the weights of the target class, $||w_{y_i}||\cdot||a^i||$ is the magnitude information and θ is the angle between feature a^i and classifier w_j. The model confidence of a single sample is defined as $\frac{e^{||w_{y_i}||\cdot ||a^i||\cdot\cos(\theta_{y_i})}}{\sum_j e^{||w_j||\cdot ||a^i||\cdot\cos(\theta_j)}}$, which indicates that if $\cos(\theta_{y_i})$ is closer to one and θ is smaller, we have more confidence that image i is from the target class y_i.

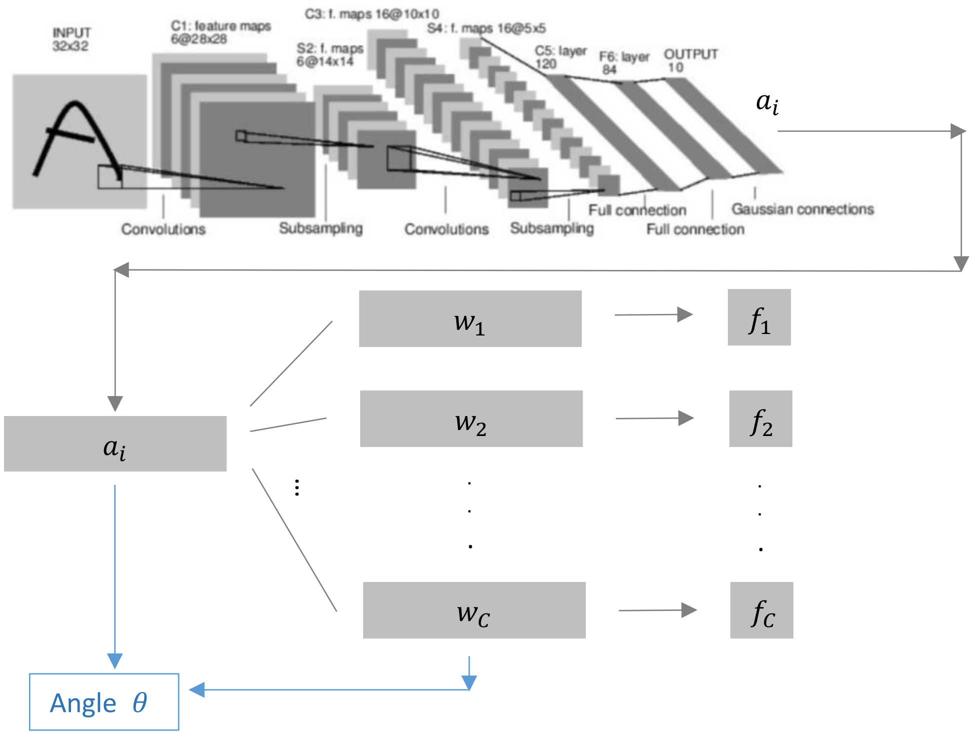


Figure 1: The structure of CNN for image classification

Based on the decomposition of dot product, Angular Visual Hardness is defined as

$$AVH(a, w_y) = \frac{\mathcal{A}(a, w_y)}{\sum_{j=1}^{C}\mathcal{A}(a, w_j)},$$

where $\mathcal{A}\,(a, w_y)$ is the angle between feature and the classifier in the target class. Since the angles are positive, the regularized angle is a score between 0 and 1. According to the definition of model confidence, if an image has a lower AVH, it is easier for the model to classify. The idea is that the angle between feature embedding and weight accounts for the inter-class differences while the L2 norm of the feature embedding accounts for intra-class variation ([2]). The hardness is characterized by the inter-class variation, a.k.a. the angle.

2.2 Properties of AVH

In this section, we will list the relevant properties of AVH mentioned in [1] and discuss how to modify the loss and the training processes based on these properties. Details about the modifications are in Section 3 and Section 4. The losses, accuracies and AVH scores in [1] are calculated on the ImageNet validation set.

First, as shown in Figure 2 ([1]), during the training process, AVH hits a plateau when the accuracy and loss are still improving. This indicates that increasing the norm of features is easier than decreasing the AVH score in the training process. However, AVH is more important than the feature norm because it is the key to decide which class the image is classified to ([2]). As a result, it is necessary to design loss functions that focus on minimizing angles. We put AVH in the loss functions in two ways: the additive way and the multiplicative way. The new loss functions will force the model to minimize AVH more aggressively than the original softmax cross-entropy loss.

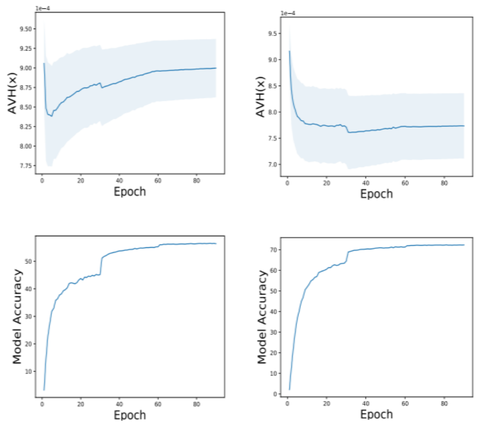


Figure 2: Better models have better AVH scores; left: AlexNet; right: VGG19 ([1]).

Second, AVH's is consistent with human visual hardness. The validation set of ImageNet provides frequencies of human selection. By comparing AVH scores with human selection frequencies, it is shown in [1] that images with higher human selection accuracy have smaller AVH throughout the training process. In other words, if an image is considered hard for human, it has a higher angular visual hardness score. Therefore, AVH is a consistent and valid representation of visual hardness. In Section 4, we implement this property by using AVH to select augmented images. Results show that augmented images neither too hard nor too simple can improve model performance.

Last, models with better generalization ability have smaller AVH scores. An example is shown in Figure 2. VGG19 has a higher validation accuracy than AVH and the AVH score of VGG19 is lower throughout the training process. What's more, the AVH score of VGG19 decreases as the model accuracy increases while the AVH score of AlexNet increases after a few epochs. As a result, adding AVH to the loss function or incorporating it in the augmentation process may improve model generality.

3. Loss Function with AVH

3.1 Incorporate AVH in the loss function

As mentioned in section 2, the AVH score hits a plateau when the loss is still. It is argued in [1] that at the beginning of training, the model will first minimize the AVH score instead of increasing the feature norm. It is because AVH is the key to which class the image is classified to. According to [2], the angle between weights (classifiers) in different classes counts for variations between classes while the norm counts for variations inside classes. Therefore, the model will first increase the angle between weights and decreases the angle between features and weights. After the AVH angles are hard to minimize and hit a plateau, the model will increase the feature norm of images with a small AVH. By increasing the norm of easy images, the dot product will increase and the loss will decrease. But the increase in feature norm will not improve the classification accuracy as effectively as AVH. In addition, models with higher validation accuracy have lower AVH throughout the training process.

This leads to the idea of forcing the loss function to consider more about the inter-class variations and minimize AVH more aggressively throughout the training process. We propose two ways to combine AVH with the softmax cross-entropy loss function: the additive way and the multiplicative way.

1. Additive AVH loss function: the new loss function with additive AVH is defined as
$$L = \frac{1}{N}\sum_i [L^i + f(AVH^i)],$$
where f is an increasing function and L^i is the cross-entropy loss.
2. Multiplicative AVH loss function: the new loss function with multiplicative AVH is defined as
$$L = \frac{1}{N}\sum_i [L^i \cdot f(AVH^i)],$$
where f is an positive increasing function and L^i is the cross-entropy loss.

3.2 Results of additive AVH loss function

We use new losses in a simple CNN model and examine whether the new loss functions will improve model generalization. The CNN structure used here is a modified LeNet5 with batch normalization and no data augmentation. The dataset is the CIFAR10 dataset instead of the ImageNet dataset in the original paper. CIFAR10 has 10 classes with 1000 training images and 100 test images in each class.

Figure 3 shows results of the baseline model on the left and results of the additive AVH loss model on the right. The baseline model is the modified LeNet5 with the original softmax cross-entropy loss. The new loss function demonstrated in the figure uses $f = AVH^2$. We tried other f as well, and it turns out the square of AVH is more comparable to the scale of the original loss and has a good classification accuracy.

In Figure 3, x-axis is the epoch number, grey lines are results from multiple runs of the model, and the blue line is the mean of results from all runs. Both the baseline and the new models are trained on 30 epochs with Adam optimizer. In the baseline model, the average training accuracy is 94.1%, with a lowest accuracy of 93.6% and a highest of 94.4%. The average test accuracy is 50.3%, with a lowest accuracy of

49.3% and highest of 52%. The final average AVH is 0.0861, with a lowest of 0.0858 and a highest of 0.0868.

For the additive AVH model, the average training accuracy is 86.7%, with a lowest accuracy of 84.8% and a highest of 88%. The average test accuracy is 52.3%, with a lowest accuracy of 50.4% and highest of 54.3%. The final average AVH is 0.079, with a lowest of 0.0781 and a highest of 0.0804.

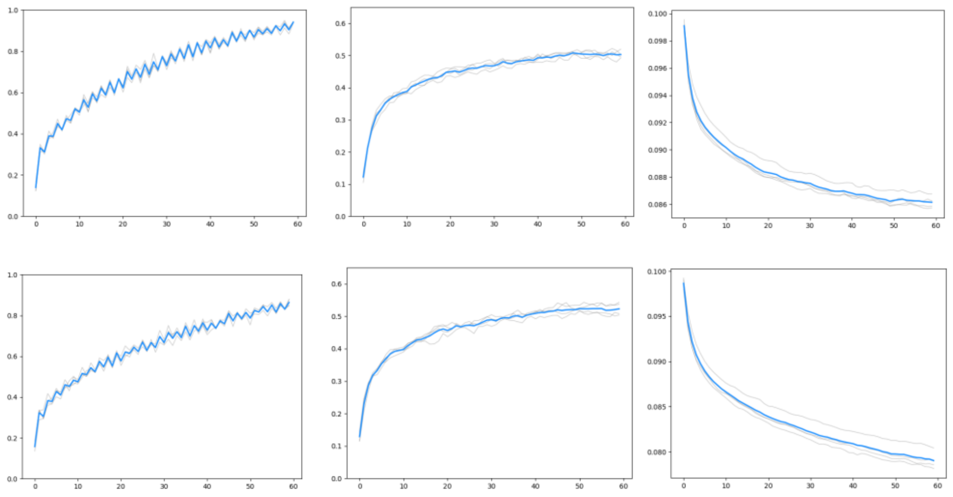


Figure 3: top row: results from the baseline model; bottom row: results from the additive AVH model;

left: training accuracy; middle: test accuracy; right: test AVH (average of all images),

According to the above results from multiple runs, our new additive AVH model has a lower training accuracy, a higher test accuracy, and a lower AVH score as the result of the new loss function. During the second half of the training process, AVH of the original CNN decreases slower than the new model, which is expected because AVH is added to the loss function.

In summary, by adding AVH to the loss function, the CNN fits the training worse but is better at generalizing to the test dataset. The test accuracy increases 2% on average. The seed in code and the results are on github.

3.3 Results of multiplicative AVH loss function

The performance of multiplicative AVH loss is checked using the same dataset, the same CNN structure and the same baseline model. The new model is also trained on 30 epochs.

Figure 4 shows results of the multiplicative AVH loss model. The new loss function that generates Figure 4 is

$$L = \frac{1}{N} \sum_{i} [L^{i} \cdot \exp(AVH^{i})].$$

By choosing the exponential function, we put relatively more penalty on the hard images. We also tried other f such as the square and the square root, but we did not observe significant differences.

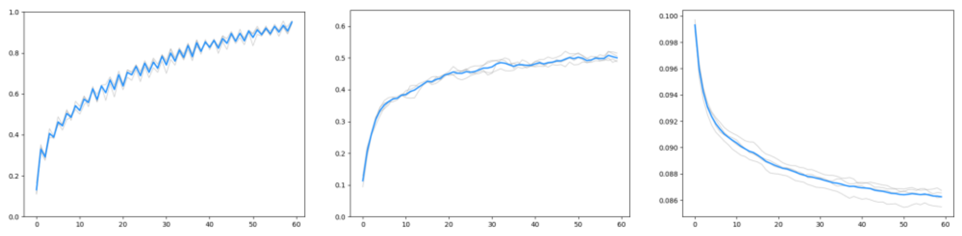


Figure 4: the multiplicative AVH model;

left: training accuracy; middle: test accuracy; right: test AVH.

For the multiplicative AVH model, the average training accuracy is 95.05%, with a lowest accuracy of 94.3% and a highest of 95.6%. The average test accuracy is 50%, with a lowest accuracy of 48.8% and highest of 51.5%. The final average AVH is 0.0863, with a lowest of 0.0855 and a highest of 0.0867.

According to the above results from multiple runs, the AVH model is almost the same as the original model. The AVH reaches about the same number as the original model even if it is added to the loss function. Since the new loss function is not very effective at penalizing AVH, its loss and accuracy are very similar to the original model.

In summary, the multiplicative AVH model is not as effective as the additive AVH model.

4. Smart Augmentation With AVH

4.1 Incorporate AVH in the training process

Augmentation is a common way to improve model performance. [3] explored data augmentation as a type of regularization. Large number of augmentations can be produced by making small changes to the data we have, such as rotating, adding random noise, etc. With the augmented samples, the model may be more robust to small changes to the original samples and may be forced to learn features that are more robust and essential to the classification task.

However, since numerous augmentations can be generated, a natural question is: are all these augmented images useful? Intuitively, if the augmented image is an easy example, it doesn't contribute much to improving model generalization; if it is a hard example, it may be an outlier to the population images. Adding large number of outliers to the original sample can mislead the training process to a wrong direction. Therefore, it might make more sense to use only part of the augmented images. Therefore, we propose a selective augmentation procedure based on the AVH score of the augmented images. The AVH score is calculated for each augmented image after loading the pre-trained baseline Lenet-5 model.

The augmented images are generated in the following way: for each training sample, randomly generate its augmentation using either random rotation (with the random degree of rotation between 20% on the left and 20% on the right), adding Gaussian random noise (with mean 0 and standard deviation 0.01), or horizontal flip. Since there are 10,000 training sample, 10,000 augmented images are generated correspondingly.

4.2 Results of Smart Augmentation with AVH

After loading the baseline LeNet-5 model, we calculate the AVH score for each augmented image. It turns out that many of the them are hard examples: their AVH is high and the accuracy on augmentation is only 0.1988. In fact, the average AVH score for training sample is 0.0675, while the average AVH for augmented images is 0.0958. This may be due to the low quality of the training data: the size of the training sample is only 28 by 28 and making a small change may drastically decreases the image quality. Figure 5 and Figure 6 are examples of augmented image with relatively high and low AVH score, corresponding to hard and easy augmented images.

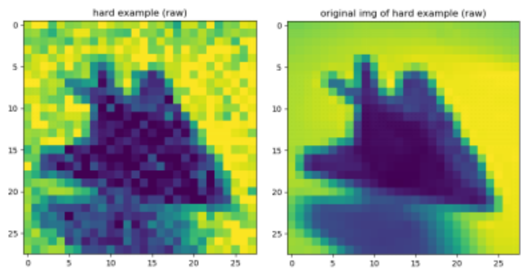


Figure 5: hard example, augmented with random noise, AVH=0.108.

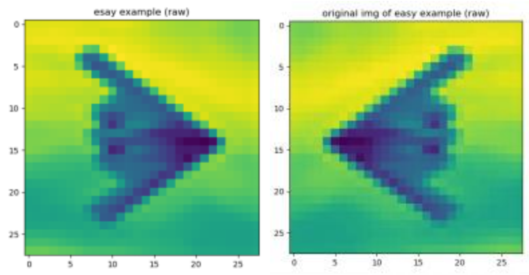


Figure 6: easy example, augmented with horizontal flip, AVH=0.072.

We then compare the following 3 methods. The baseline method is to train the LeNet-5 model with cross entropy loss after adding all 10,000 augmented images to the training sample. We call this method M0. The second method is to train the LeNet-5 model (with cross entropy loss) after adding a randomly selected 20% of the 10,000 augmented images. We call this method M1. The third method is to train the LeNet-5 model (with loss function being the sum of cross entropy loss and the square of AVH) after adding only selected 20% of the 10,000 augmented images. Specifically, only augmented images with AVH in the 2%-22% quantiles are added to the training samples. We call this method M2. Thus, M0 has 20,000 samples during the training process, M1 and M2 has 12,000 samples during the training process.

The following results are the mean from 5 replications, each with training epoch=25. On average, method M0 has a test accuracy 54.1%, method M1 has a test accuracy 54.7%, method M2 has a test accuracy 59.1%. We plot the mean test accuracy for each method in Figure 7. On average, method M2 has improved over M0 and M1 by more than 4%. This advocates our assumption that using AVH score to select augmentation that have some noise but are not too different from the training samples may help improve generalization without misleading the model.

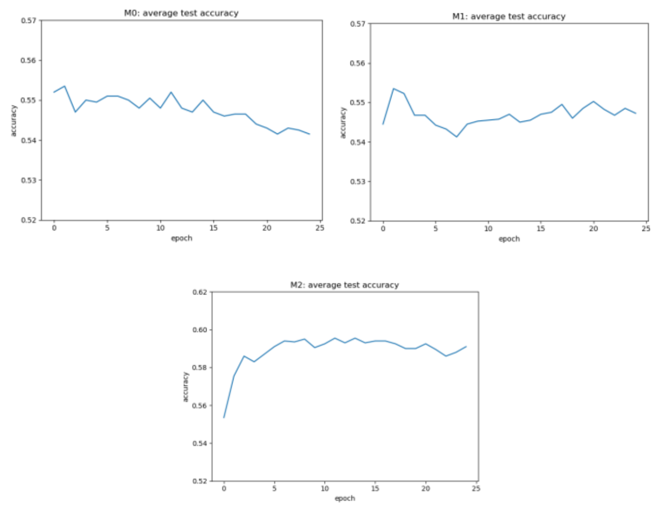


Figure 7: average test accuracy of 5 runs.

5. RNN with AVH

5.1 Incorporate AVH in RNN

Previous works on AVH only focus on state-of-art CNNs for image classification problems. But it is also interesting to check whether the AVH score can be extended to other deep learning setups as well. Because ImageNet is the only dataset we can find that provides human classification accuracy, it is unable to check whether AVH in other applications is also consistent with human classification hardness. However, it is still of interest to check whether the properties of AVH on CNN image classification holds for other deep learning tasks. Is AVH consistent with training loss and training accuracy during the training process? Is AVH of a sample consistent with its model prediction accuracy? Will the AVH score reach a plateau before the training loss converges? Will models with better generalization have smaller AVH throughout the training process? If these are true, can we improve model generality using the methods proposed in the previous two sections?

The idea of extending AVH is as follows. First, recall that AVH is based on the decomposition of dot product in the last output layer. Second, if the dot product between features and weights has higher value, the loss is smaller. As long as the output layer of a network satisfies these two conditions, it makes sense to look at the normalized angle between features and weights in the last layer. For applications other than image classification, the extended AVH is not a measure of visual hardness, but it is still a reasonable hardness score for the model.

The specific extension we explore in this project is to incorporate AVH in RNN for text classification. The final output layer we use is a softmax cross-entropy loss, so the calculation of AVH is the same as the CNN model in previous sections. We train RNNs on the IMDb dataset of movie reviews, examine the properties of AVH and add AVH into the softmax cross-entropy loss function.

5.2 IMDb data and data processing

The IMDb dataset consists of movies reviews from IMDb and are labeled by positive or negative sentiment ([4], [5]). This dataset can be directly downloaded using keras. The dataset consists of 25,000 training sequences and 25,000 test sequences. In keras, a dictionary from words to integers is applied to the processed text data. After the transformation, each text sample is encoded as a sequence of integers.

A word embedding of size 32 is used to convert each word into a vector. The length of sequences varies, and a padding of size 200 is used to truncate long sentences to 200 words and fill up short sentences with zeros. A dynamic RNN with LSTM cell of size 100 is used as the base RNN model. In dynamic RNN, if a sentence’s length is less than 200, the sentence of original length is fed into the model instead of the padded length 200. Different embedding sizes, padding sizes and hidden layer sizes are implemented, but no significant difference is observed.

The whole process is coded in tensorfow. If we use keras instead of tensorflow to build the RNN, the test accuracy will increase about 10% even if the same RNN structure is used. We use tensorflow in the project because first, it is easy to do tracking and modifications; second, the tensorflow backend of keras does not have the cosine function, so it is very difficult to create a keras tensor for AVH and track the gradient flow of weights from it.

5.3 The properties of AVH in RNN

The properties of AVH during the training process is demonstrated using the baseline model. The baseline model is a dynamic RNN with LSTM cell of size 100. The original softmax cross-entropy loss is used. Trace plots of the indexes of interest are in Figure 8 and Figure 9. AVH is the mean AVH of all samples. Mean magnitude of weights is the mean of magnitudes of weights in the last layer from all classes. Mean of magnitude of features computes the mean of magnitude of features from all images.

It is not surprising to see that AVH decreases a lot at the beginning of the training process then hits a plateau. Our second finding is that compared to the magnitude of weights and features, AVH is more consistent with the loss and accuracy on the test data. AVH stops improving when loss and accuracy stop improving, but the mean magnitude of features continue to increase. This means that the angle between feature and weight has a stronger influence on model performance than magnitudes. Therefore, it is necessary to design better loss functions as the previous section.

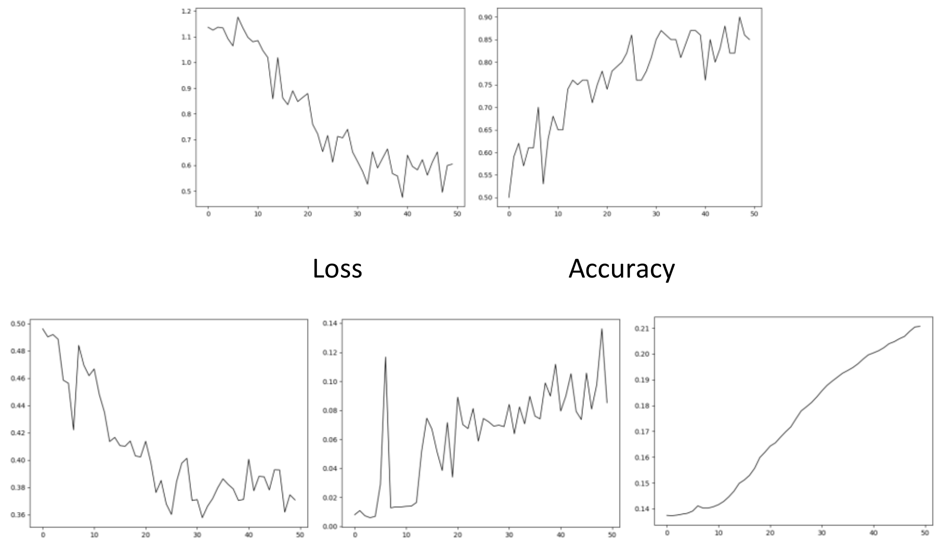


Figure 8: results of the baseline model, training data.

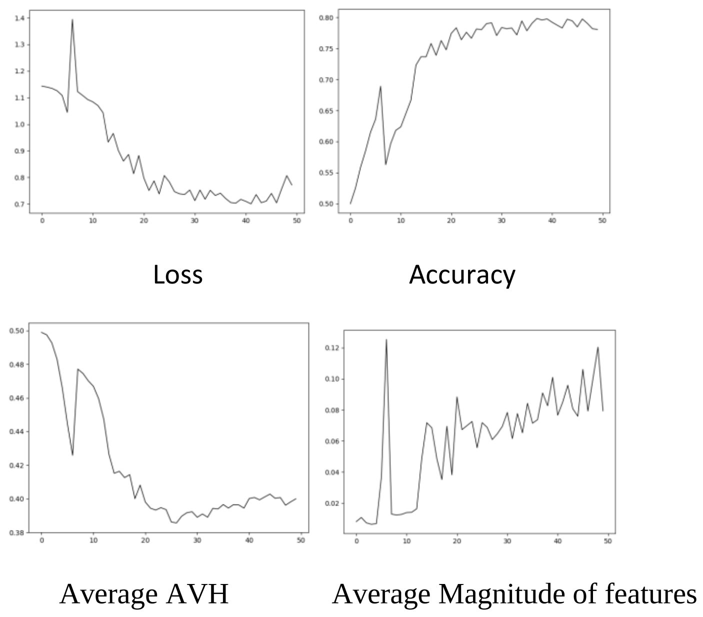


Figure 9: results of the baseline model, test data.

What’s more, we also tried other RNN structures with different embedding sizes, padding sizes and hidden layer sizes. We notice that the models with better generalization (smaller test error) have smaller AVH. This is consistent with the CNN setup.

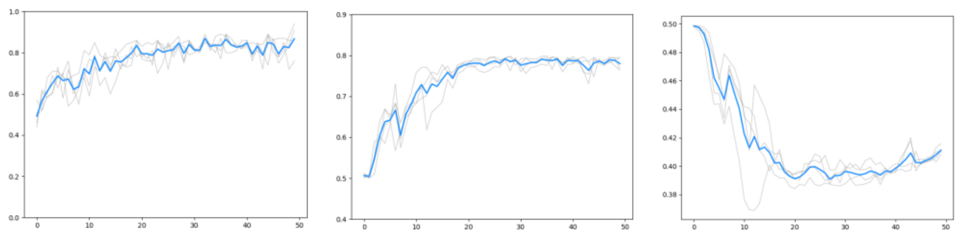
5.3 Incorporate AVH in the loss function

As discussed in 5.2, it is reasonable to design loss functions that penalize more on the angle between feature and target weight. We implemented the same additive and multiplicative AVH loss functions as the previous session and show the results in Figure 10 and Figure 11.

In the baseline model, the average training accuracy is 86.75%, with a lowest accuracy of 76% and a highest of 94%. The average test accuracy is 78%, with a lowest accuracy of 76.42% and highest of 79.49%. The final average AVH on test set is 0.4112, with a lowest of 0.4083 and a highest of 0.4158.

According to Figure 10, the additive AVH loss results in an unstable training process. Although it sometimes reaches a higher test accuracy than the original model, it also occasionally generates very bad results.

Figure 11 is the results from multiplicative AVH loss. It indicates that the multiplicative AVH loss is very similar with the original loss.



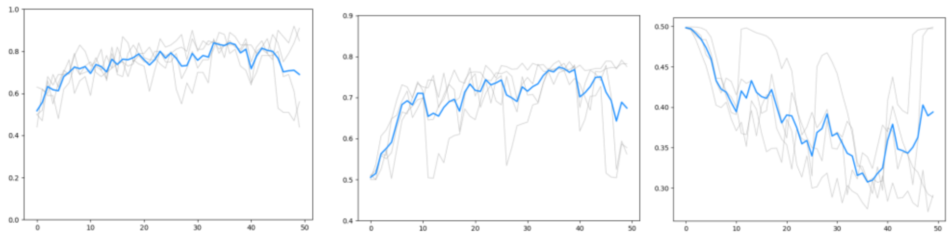


Figure 10: top row: results from the baseline RNN; bottom row: results from the additive AVH model;

left: training accuracy; middle: test accuracy; right: test AVH.

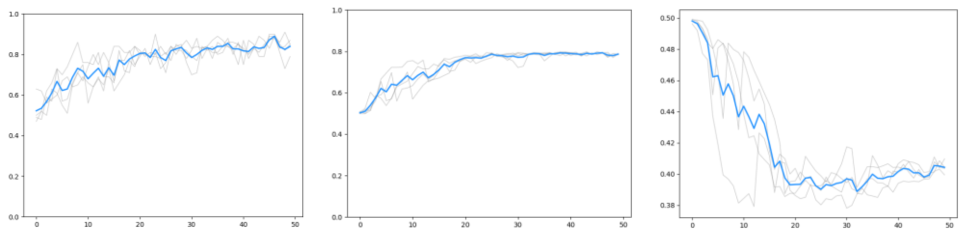


Figure 11: the multiplicative AVH model with RNN;

left: training accuracy; middle: test accuracy; right: test AVH.

6. Summary

In this project, we use two methods to improve CNN for image classification. One is the modified loss, and the other one is Smart Augmentation. Results show that an additive AVH loss with augmented images neither too hard nor too simple can improve model performance. We also extend AVH score to RNN and examine its performance. Future work would be to design better loss functions or training processes that incorporates AVH or design a better score for visual hardness or model hardness.

Citations

1. Chen, Beidi, et al. "Angular Visual Hardness." arXiv preprint arXiv:1912.02279 (2019).

2. Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M Rehg, and Le Song. "Decoupled networks." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2771–2779, 2018.

3. Chiyuan Zhang et al. "Understanding deep learning requires rethinking generalization". In:CoRRabs/1611.03530 (2016). arXiv:1611.03530.

4. https://keras.io/datasets/

5. https://www.imdb.com/interfaces/

6. https://cpb-us-e1.wpmucdn.com/blogs.rice.edu/dist/1/7022/files/2019/11/Back.pdf

Appendix

All the code files are in https://github.com/chunshanl/ELEC576final.git. The seed used are all included in code.

GPU access: personal laptops. Memory: 16384 RAM.

Work assignment: The two of us discussed the idea together. We coded together and constantly discussed details in coding. Each file has multiple versions before it is finalized. For the finalized results presented in the report, Huiming Lin contributes more to the CNN models and Chunshan Liu contributes more to the NLP models.