Exploiting Loss Function Properties for Improved Performance

Snehal Prabhu
UMass Amherst
scprabhu@umass.edu

Shruti Shelke
UMass Amherst
sshelke@umass.edu

Abstract

The training of a neural network is supervised by the loss function, which acts as a learning objective with the goal of improving the accuracy of a model. It does so by managing the weights in neural networks in such a way that the performance is maximized. Generally, individual loss functions are used to train neural networks for object recognition tasks. However, most other loss functions might also have inherent attractive properties that would benefit in training the model. In this research paper, we suggest a combined loss function that exploits the advantages of multiple loss functions that have complimentary properties. We do so by analyzing these properties for a set of functions which are Hinge Loss, Softmax Loss, L-Softmax Loss and A-Softmax Loss and discussing how each loss function benefits the overall training of a neural network. We support this argument through an experimental analysis on an image classification task that employs a fully connected neural network on the CIFAR-10 dataset. We introduce various combinations of losses to train on this model and perform hyper parameter optimization through grid search to assign weights to each loss function in the combined loss. We evaluate the classification accuracies for each loss function and observe an improved performance in the combined loss function as we predicted.

1. Introduction

In neural networks, loss functions perform the duty of evaluating the set of weights so as to maximize the accuracy of our model. In the context of image classification, we cannot modify the data that maps pixel values to class scores. However, we can regulate the set of weights that parameterizes these mappings using loss functions. These objective functions attempt to increase the probability of correct classes and decrease the probability of incorrect classes by ensuring that the predicted class labels correspond as closely as possible to the ground truth labels for each image. Regular loss functions such as SVM and Softmax loss each have inherent properties that make them attractive options to train neural networks on. Some loss functions outshine others in performance under different circumstances but most of them have several advantages that improve accuracy on an individual basis while decreasing it in others. To exploit the superiority of a single loss function while retaining the complimentary properties of other loss functions, we suggest employing newer loss functions that are a combination of multiple commonly used loss functions. In this research paper, we study the nature of the properties of four loss functions, namely Hinge Loss (SVM), Cross-Entropy Loss (Softmax), Large-Margin Loss (L-Margin) and Angular Loss, and argue that their various combinations would allow us to train more effective networks for object verification tasks.

We perform an experimental analysis to support our aforementioned claim in the context of an image classification task. We train a baseline model that is a fully connected neural network which uses the regular softmax function. We further modify the loss function in our baseline model and show how several loss functions perform better than the others on an individual basis, and how combinations of complimentary loss functions gives us a higher classification accuracy in a fully connected neural network for image classification. We substantiate our reasoning by executing a comparative study of the performance of linear combinations of these loss functions using grid search for hyper parameter optimization. These searches provide us with the weights to assign to each loss function in our linear combination of loss function such that accuracy is maximized.

We use the CIFAR-10 dataset on our neural networks model to perform the required tasks for our experimental analysis. This dataset consists of 60,000 images, each of size 32X32. They are divided into ten classes as the following: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. Each class has approximately 6000 images each. The entire dataset is divided into six portions. One portion acts as the testing dataset, one as the validation set and the other four act as the training dataset. The testing dataset contains 1000 images which are randomly chosen from each class, hence a total of 10000 images. The rest of the images go into the training datasets, which may or may not have a uniform distribution of classes per dataset, meaning there could be more images from one class than the others in a particular training dataset. We use the classification accuracy as a metric to compare the performance of different loss functions on our image classification network. We achieve this by calculating the fraction of correctly classified classes over total number of classes. We will do so for each loss function we select and see which loss function gives us a higher accuracy on our image classification neural network.

We expect that the classification accuracy achieved by the model which uses the combined loss function would perform the best out of the models using each loss functions individually. We also expect that the hyper parameters that describe the weights of each loss function in our model will put a larger emphasis on L-Softmax Loss and a slightly lower one on Angular Softmax and SVM as the former is more effective and rigorous. However, in terms of time efficiency, we expect this model to perform the worst. That is because the combination of the loss functions should be able to bring the predicted class very close to the actual class for each image since the influence of all the loss functions is being taken into account on the weights and higher emphasis on L-Softmax Loss gives us a more challenging learning objective. During the experimental analysis, we see that our claims are quantified and proved by the classification accuracy for each neural network trained on these losses.

2. Related Work

The paper 'A Performance Comparison of Loss Functions' does a comparative study of the effect of various loss functions on the performance of the MNIST dataset. While we do a similar comparison analysis of loss functions, our paper focuses more on the logical and mathematical reasoning behind the observation that the loss with a combined sum of various loss functions has the best accuracy as well as a grid search method for optimal hyperparameters, whereas the aforementioned paper focuses more on the observations themselves on a simple linear combination.

In the paper, the authors use a Deep Neural Network for digit classification, which is a highly time consuming process. In an effort to reduce training time, parallelized DNN training as well as usage of better loss functions is essential. The latter is justified by the statement 'the training process of DNN is equivalent to minimizing the output of the loss function from an optimization point of view'[1]. In addition to the regular softmax, they introduce and utilize Modified and Angular softmax losses. They also use Additive-Margin softmax loss, Arcface loss, Center loss, Focal loss and a combined loss of all these losses. Their digit recognition experiment analyzes the effect of these losses on the train accuracy and test accuracy due to digit distribution and convergence time for the DNN.

In terms of network architecture, it has six convolution layers, three max pooling layers, and a fully-connected layer with two to three dimension embeddings. The MNIST dataset that they use has 60,000 images in the training dataset and 10000 in the test dataset, each of size 28x28, containing one of the ten digits in handwritten format and a label defining the digit for each image. They find that the new combined loss 'classified digits more clearly and visibly than the others, and besides the Arcface loss and the Additive-Margin softmax loss required the shortest convergence time in order to meet the predefined accuracy'. Using this study as a background, we conduct similar experiments to support our logical claim that the combined loss performs well for image classification.

Another paper that we studied for our research was 'Large-Margin Softmax Loss for Convolutional Neural Networks'. In this paper, the authors propose a new loss function called large margin softmax (L-softmax), which is inspired by the amalgamation of cross entropy loss with softmax loss, while also encouraging discriminative learning of features. This L-margin loss is described as 'the softmax loss as the combination of a cross-entropy loss, a softmax function and the last fully connected layer'. Unlike the fully connected network that we use on our model, this paper employs a convolutional neural network to perform experiments based on VGG Net in terms of architecture. The advantages of the L-softmax loss function include intra-class compactness and inter-class separability between learned features, increased desired margins and avoids overfitting. Additionally, the discriminative nature of the L-Softmax loss improves the performance of classification. We adapt this loss into our own architecture as work as well and use this paper as a guideline to describe the importance of this loss in our combined loss.

The paper 'Exploring the Role of Loss Functions in Multi-class Classification' provided us with the knowledge and intuition in picking the loss functions for our image classification model whose dataset also has multi-class labels, ten to be specific. This paper attempts to explain the effect of loss functions on the classifier performance and the reasoning behind the trump between loss functions. The paper argues that models with loss functions that over-parameterize also generalize better while maintaining accuracy. They are easier to train and boost testing performance. The authors argue that 'different loss functions have different levels of effective over-parameterization' and this affects the optimization landscape and prove that softmax loss tends to over-parameterize well, leading us to perform grid search of weights for our combination of loss functions in such a way that we emphasize the use of softmax and softmax-like objective functions over others.

3. Study of Properties of Loss Functions

In this section, we study the properties of each loss function that we use in our classifier and what each loss brings to the performance of our model. We reason how some losses outperform the others but also have their own drawbacks, hence a combination of these loss functions allows us to take advantage of the complimentary properties of each individual loss function and hence get a more effective image classifier.

1. Hinge Loss (SVM): The Support Vector Machine loss function works in such a way that it encourages the model to pick the correct class for any given image such that the score exceeds that of the incorrect classes by a given margin. Here given image x_i that has a label y_i corresponding to the correct class, we compute the vector of each label j as a linear function of the weights matrix W and x_i which serves as a score for label j. We similarly compute the same for the label y_i for the correct class and take the maximum between 0 and the difference between the two vectors in addition to the margin that we discussed above.

$$L_i = \sum_{j \neq y_i} \max(0, w_j^T x_i - w_{y_i}^T x_i + \Delta)$$

Figure 1. SVM Loss Function

Logically, 'The loss function quantifies our unhappiness with predictions on the training set'[6]. SVM penalizes not only incorrect predictions but also correct but not confident predictions. It does so in such as way that the penalty is indirectly dependent on the correctness and confidence of the predicted values, i.e., it significantly penalizes extremely wrong predictions, mildly penalizes correct but not confident predictions, and does not penalize confident, correct predictions. It maximizes the margin between the decision boundary and image so that the classification of each image is correct as well as confident. However, once the SVM notes that the margins are satisfied, it will no longer 'optimize the weights in an attempt to "do better" than it is'[6] already because it does not care about individual scores. Furthermore, these scores are not standardized by calibration and are difficult to interpret.

2. Softmax Loss (Cross-Entropy Loss): Softmax loss is the generalization of the binary Logistic Regression classifier to multiple classes. This loss function is more intuitive than SVM as it is based on a maximum likelihood estimate of our model's parameters and can be understood through the lens of a probabilistic interpretation. It allows us to interpret our model's scores as relative probabilities against each other as confidence in each class.

$$f(x_i; W) = Wx_i$$

where ***s*** denotes the mapping function ***f***

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Figure 2. Softmax Loss

We calculate the normalized probabilities for each class for a given test image and get the probability distribution for the data. Using cross entropy we calculate the negative log of these probabilities to get the softmax loss. The goal of a softmax loss classifier is to minimize the negative log likelihood of the correct class i.e., minimize the cross-entropy between the estimated class probabilities. Unlike SVM, the softmax loss is not satisfied the way the SVM loss is, that is because it believes that the loss could always be improved by giving a lower probability to incorrect classes and a higher probability to correct classes. However, this softmax loss also has its shortcomings. Mainly, it does not optimize features to get the best performance possible, i.e., 'it neither increases similarity score for positive pairs nor decreases similarity score for negative pairs'.[1]

To deal with this issue, we modify the boundaries between classes for the softmax loss. With the reduced boundary for each class, we can increase the distance between each class that is every class is better distinguished from another class in the dataset. This way we get the higher probability values for each correct class identified and reduce the probability for all other classes. We also work with different values to introduce an angle shift in the calculation of normalized probabilities and conclude if there is any improvement in the performance of the training model for softmax loss. We achieve both of the following using Large margin Softmax and Angular Softmax respectively.

3. Large Margin Softmax (L-Softmax):

The Large Softmax Loss introduces a margin to our original Softmax Loss in an effort to support 'intra-class compactness' and 'inter-class separability' between learned features by increasing the distance between the classes i.e., it tries to increase compactness between elements of the same class and reduce closeness or increase separation between elements of different classes. This margin can be manipulated using a parameter m, where the value of m decides how large the margin is.

$$L_i = -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_j)}}\right)$$

$$\psi(\theta) = (-1)^k \cos(m\theta) - 2k, \quad \theta \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$$

Figure 3. Large-Margin Softmax Loss

A bigger m increases the margin between classes while also increasing learning difficulty. When m=1, the L-Softmax Loss is the same as our original softmax loss. It has a discriminative nature, hence during visual classification tasks such as ours, it performs well. It is also less susceptible to overfitting due to the inherent nature of the difficult learning objective. The parameter m is a positive integer and our function is monotonically decreasing and k is in the range 0 to m-1 inclusive. Θ is the angle between W and x.

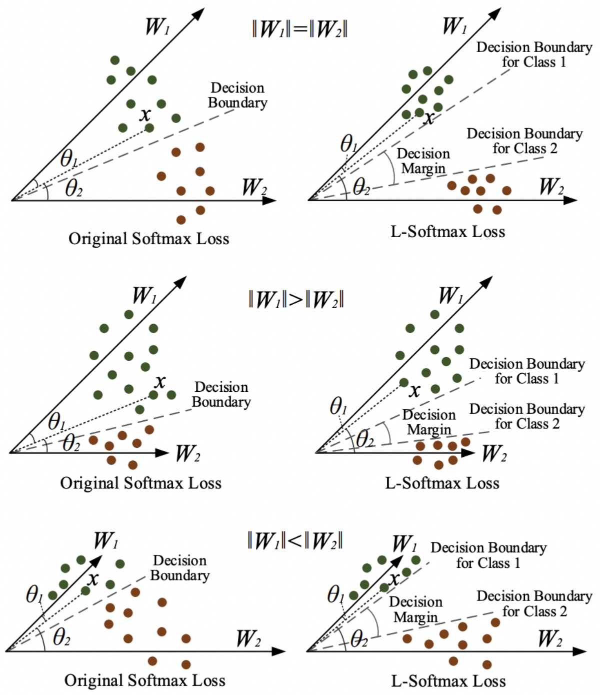


Figure 4. Modification to Softmax Loss

Let us understand the intuition behind this L-Softmax Loss geometrically. Assume that this is a binary classification problem and x belongs to class 1. Looking at the first case when $\|W1\| = \|W2\|$, we want $\|W1\|\|x\|\, cos(\theta 1) > \|W2\|\|x\|\, cos(\theta 2)$ since we want our loss to force $W1 * x > W2 * x$ to correctly classify x to class 1, which is what our regular softmax does. This still holds if $\|W1\|\|x\|\, cos(m * \theta 1) > \|W2\|\|x\|\, cos(\theta 2)$ because $\|W1\|\|x\|\, cos(\theta 1) >= \|W1\|\|x\|\, cos(m * \theta 1) > \|W2\|\|x\|\, cos(\theta 2)$.

Therefore, all of our requirements are still satisfied like they were with regular softmax but now we also have a more rigorous decision boundary due to the parameter m. This is because our classification is dependent on the angle between W and x. In regular softmax loss, we just needed $\theta 1 > \theta 2$ to hold true. By introducing the m parameter in L-Softmax, we ensure that $m\theta 1 > \theta 2$ holds true as well. This allows us to optimize our loss for the same values in both cases, which introducing and adjusting the decision boundary in L-softmax such that the angle between the input and correct class is narrower. Hence, the performance of our classifier with an L-softmax improves significantly as compared to that of a classifier using the regular softmax loss.

Even for cases when $\|W1\| > \|W2\|$ and $\|W1\| < \|W2\|$, the L-Softmax loss proposes a decision margin, which is affected by both the angles between x and W1, W2 as well as the $\|W1\|$ and $\|W2\|$. The larger $\|Wj\|$ is, the larger the angle for the corresponding class while still holding our basic requirements true and hence improving performance.

4. Angular Softmax:

Similar to the goal of Large Margin Softmax, Angular Softmax also seeks to increase intra-class compactness as well as increase inter-class separability. It does so by adding an angular margin on the angle between W and x, which is also parameterized by m. The major difference between L-Softmax loss and Angular Softmax Loss is that the latter normalizes classification weight W into 1, and sets the bias to 0.

$$L_A = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\|x_i\|\cos(m\theta_{y_i i})}}{e^{\|x_i\|\cos(m\theta_{y_i,i})} + \sum_{j=1,j\neq y_i}^{c} e^{\|x_i\|\cos(\theta_{j,i})}}$$

Figure 5. Angular Softmax Loss

This results in a simpler requirement to be fulfilled by the margin, which is simply that $cos(\theta 1) > cos(\theta 2)$ since $\|W1\| = \|W2\|$ and hence $\|W1\|\|x\| = \|W2\|\|x\|$. Therefore adding an angular margin only requires us to ensure that $cos(m\theta 1) > cos(\theta 2)$ where a larger m will give us larger decision boundaries. The advantage of Angular Softmax is that it increases the rate of convergence during training due to the normalization and setting of bias to 0 and increases recognition performance for unviewed images during training.

5. Combined Loss: We use several combinations of the four loss function to avail the benefits of each in a our model. We join Hinge Loss with Softmax since the former works well with unstructured data and is less susceptible to overfitting while the latter is more determined to improve performance. We combine L-Softmax with A-Softmax for the rigorous nature of the former and quick convergence and comprehensive learning of unseen features of the latter. We also add SVM to the L-Softmax and Angular Softmax losses to utilize benefits of all of the loss functions together. The parameter m was given values 2, 3 and 3 for both L-Softmax and A-Softmax losses as these are the most commonly used values. We do not combine regular softmax with this complete combination since it would be redundant. The weights given to each loss function in each of the combinations vary as we try and test hyper parameters using grid search to optimize effectiveness of our loss function on the classification accuracy. However, looking at the general capabilities in terms of performance of each loss function individually, we see the following order in descent: Large Margin Softmax $>$ Angular Softmax $>$ Regular Softmax $>$ Hinge Loss, hence we make an educated guess that proportional weights in the combined loss will give us the best classification accuracy.

4. Technical Approach

The main goal of our experimental analysis is to substantiate our claim that exploiting complimentary properties of different loss functions through linear combinations gives us a better classification model. We achieve this goal by performing a hyperparameter optimization using random search as well as grid search. Here we choose the hyper parameters using grid search which are for the combined loss function, where the hyper parameters determine how much weight each loss (either SVM, L-Softmax and/or Angular Softmax) has. Random search is used for hyper parameters m, λ and bias. From our theoretical analysis, we state that L-Softmax should have more weight as compares to the Angular Softmax and we use this technical approach section to verify our claim.

Our architecture consists of a fully connected network. To study these different losses, we train this fully connected network consisting of a combination of affine layers, batchnorm layers, relu non-linearities and dropout layers. We first train a model on each individual loss that we discussed in the previous section. We use random search to get a range of values for the other hyper parameters such as regularization constants and biases so that it is best suited for that particular loss function. This gives us the best accuracy that each loss function can achieve on an individual basis and we will later use the best accuracies for each loss for a comparison study.

We then observe changes in performance of the classifier network as we modify and combine different loss functions. Our permutations of losses included the following: original (Softmax loss and Hinge loss), (Large Margin Loss and Angular Loss), (Large Margin Loss, Angular Loss and Hinge Loss). To find the proportion of how each loss which contributes towards the combined loss in the most effective way, we fine tuned our hyperparameters such that we had the best possible customized subset to train our model on. To complete this task, we use grid search. Grid search is a technique used to find the optimal parameters of a model which gets the best accuracy such that we define a search space as a grid of hyperparameter values and evaluate every position in that grid [7]. Our search space is defined as the ' volume to be searched where each dimension represents a hyperparameter and each point represents one model configuration'[7]. For the context of this paper, our search space is in the 0.20 to 0.90, with the grid search indices computed in increments of 0.1 and decrements of 0.1 from each side iteratively. We found the best combination hyperparameters for Angular Softmax and L-Softmax and then treated it as a single function while adding SVM to the combined loss. We see the effects of these losses on the performance of our neural network in the results section.

5. Experimental Results

We first train the model individually on each loss function and we compute the classification accuracy for the network. From the table, we observe that Large Margin Softmax with m=4 performs the best out of all the loss functions when each are trained on their own.

Accuracy for Loss Function			
Loss Function	**Train**	**Validation**	**Test**
SVM	62.2	54.1	50.5
Softmax	**64.0**	**55.0**	**53.8**
L-Softmax (m=2)	58.2	54.7	52.0
L-Softmax (m=3)	44.7	44.2	41.7
L-Softmax (m=4)	**63.7**	**56.1**	**55.5**
A-Softmax (m=2)	59.7	55.6	53.0
A-Softmax (m=3)	44.2	43.8	40.3
A-Softmax (m=4)	**63.8**	**53.6**	**54.5**

This observation matches our expectation from our study of properties as L-Softmax loss with a larger m has the most rigorous decision boundary. It is followed by A-Softmax, Softmax and Hinge Loss in that decreasing order, as we predicted. Looking at the graphs in Figure 3:Training Accuracy for Individual Loss Functions and Figure 4: Validation

Accuracy Individual Loss Functions, we observe that the training accuracy and the validation accuracy for the Large-Margin Softmax function and Angular Softmax function for a larger m trains better then lower m values. For training accuracy, as number of iterations increases, L-Softmax and A-Softmax losses, each with parameter m=4 perform almost equally as good. For validation accuracy, as number of iterations increases, L-Softmax with m=4 outperforms all the other losses.

Figure 6. Training Accuracy for Individual Loss Functions

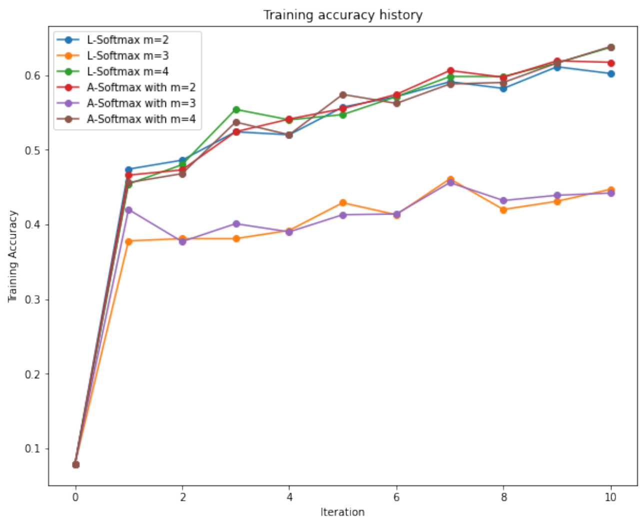
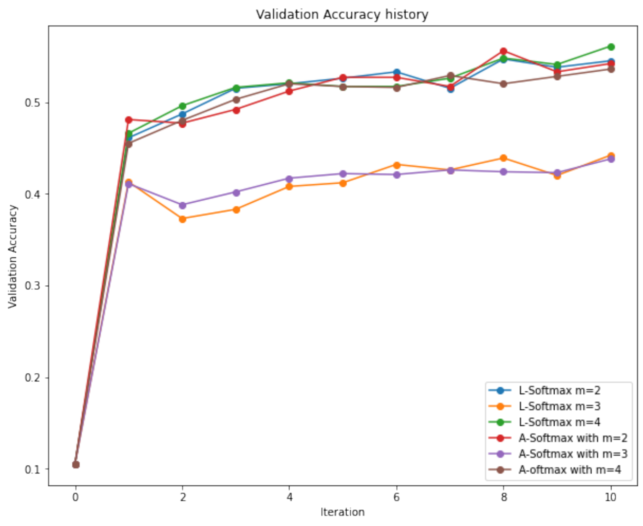


Figure 7. Validation Accuracy Individual Loss Functions



Following this, we implement different combinations of loss functions to inspect any improvements in the performance of the model. We use grid search to get choose the best hyper parameters for finding the weights to combine these loss functions. In the following table, we only discuss the optimized hyper parameters for each combination so as to compare the best accuracy possible for each combination against another.

Accuracy for Combined Loss Functions			
Loss Function	**Train**	**Validation**	**Test**
SVM + Softmax with Higher weight to Softmax	63.2	54.1	50.4
SVM + Softmax with Equal weight to Softmax	63.0	54.2	53.8
SVM + Softmax with Lower weight to Softmax	59.8	53.4	52.2
(weight=0.65) L-Softmax + (weight=0.34) A-Softmax (m=2)	**61.4**	**54.1**	**54.6**
(weight=0.65) L-Softmax + (weight=0.34) A-Softmax (m=3)	42.0	43.5	40.5
(weight=0.65) L-Softmax + A-Softmax (weight=0.34) (m=4)	60.3	55.4	53.8
(weight=0.25) SVM + (weight=0.65) L-Softmax + (weight=0.34) A-Softmax (m=2)	**62.0**	**54.4**	**54.1**
(weight=0.25) SVM + (weight=0.65) L-Softmax + (weight=0.34)A-Softmax (m=3)	36.2	36.8	31.4
(weight=0.25) SVM + (weight=0.65) L-Softmax + (weight=0.34) A-Softmax (m=4)	**64.3**	**55.9**	**55.5**

The combinations of loss function train the neural network better than when the model uses only a single loss function. We observe a slight increase in the test accuracy when we combine the Large-margin softmax loss function with the Angular softmax function with m=4. We get the best accuracy with a combination of the Large-margin softmax, Angular Softmax and SVM. The best of these assign a larger weight to the L-Softmax and a lower weight to SVM.

Looking at the graphs in Figure 5:Training Accuracy for Combined Loss Functions and Figure 6: Validation Accuracy for Combined Loss Functions, we observe that as the number of iterations increase, the loss function consisting of (lowest weight) SVM, A-Softmax and (highest weight) L-Softmax losses each with m=4 begins to get the

Figure 8. Training Accuracy for Combined Loss Functions

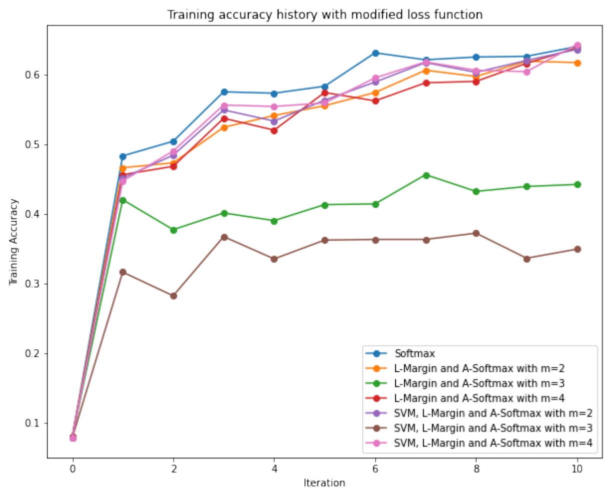
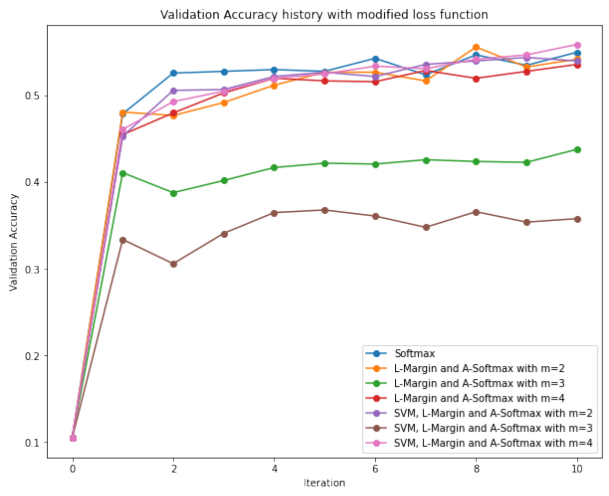


Figure 9. Validation Accuracy for Combined Loss Functions



best accuracy with individual softmax that was the highest initially decreases over time. From Figure 6, we see that for validation accuracy on the combined datasets, this same permutation outperforms all the other combined functions to give us the best accuracy on our image classification model. Through this experimental analysis, we see that our informed prediction that the combination of losses with higher emphasis on L-Softmax with larger m and lower emphasis on A-Softmax performs the best out of all the losses we trained our network on.

6. Conclusion and Future Work

In this research paper, we sought to analyze inherent properties of loss functions such that we could combine them to exploit their advantages in a manner that was complimentary. We did so with four main loss functions: Hinge Loss, Softmax Loss, Large-margin Softmax Loss and Angular Softmax Loss. We studied the benefits of each function and discussed how we could group together all of these in a single loss function to get improved performance. We did an experimental analysis in an image classification task using the CIFAR-10 dataset with a fully connected network to support our theory. This study consisted of a comparative analysis of classification accuracies of individual as well as combined loss functions. The weights given to each loss function in the linear combination was determined through grid search for hyperparameter optimization. All of the observations from our experimental analysis substantiated our argument that a combined loss function consisting of individual loss functions with complimentary properties has better classification accuracy, namely when L-Softmax loss had the highest weight and SVM the least. This particular linear combination allowed us to extract the properties of each loss function such as the rigor of the decision boundary, deterministic and stringent learning objective, high convergence rate and resistance to overfitting respectively.

For future work, this combined loss function model can be extended to other object recognition tasks such as classifying digits, health data etc. We could also use a pretrained CNN such as VGG-Net instead of training a fully connected neural network from scratch to improve effectiveness. We could change hyper-parameter weights to give more emphasis to A-Softmax for open-set face recognition datasets since it performs well when it encounters images in need of classification that it has not seen before in training. Another improvement on our project could be defining a loss function to optimize determination of hyperparameters more dynamically such as meta-learning.

References

[1] Kwantae Cho; Jong-hyuk Roh; Youngsam Kim; Sangrae Cho, 'A Performance Comparison of Loss Functions': *https://ieeexplore.ieee.org/document/8939902*

[2] Y. Wen, K. Zhang, Z. Li and Y. Qiao, 'A discriminative feature learning approach for deep face recognition', European conference on computer vision: *https://link.springer.com/chapter/10.1007/978-3-319-46478-7*$_3$1

[3] Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2999-3007, doi: 10.1109/ICCV.2017.324: *https://ieeexplore.ieee.org/document/8237586*

[4] Chen, S., Liu, Y., Gao, X., Han, Z. 'Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices':

https://link.springer.com/chapter/10.1007/978-3-319-97909-$0_4$6

[5] Cao, L. Shen, W. Xie, O. M. Parkhi and A. Zisserman, "VGGFace2: A Dataset for Recognising Faces across Pose and Age," 2018 13th IEEE International Conference on Automatic Face Gesture Recognition: https://ieeexplore.ieee.org/document/8373813

[6]https://www.maxkohler.com/notes/2018-11-21-cs231n-3/

[7]https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/