Deep Convolutional Neural Network with Independent Softmax for Large Scale Face Recognition

Yue Wu†, Jun Li†, Yu Kong†, Yun Fu†‡
†Department of Electrical & Computer Engineering, Northeastern University, Boston, USA
‡College of Computer & Information Science, Northeastern University, Boston, USA
{yuewu, yukong, yunfu}@ece.neu.edu, junl.mldl@gmail.com

ABSTRACT

In this paper, we present our solution to the MS-Celeb-1M Challenge. This challenge aims to recognize 100k celebrities at the same time. The huge number of celebrities is the bottleneck for training a deep convolutional neural network of which the output is equal to the number of celebrities. To solve this problem, an independent softmax model is proposed to split the single classifier into several small classifiers. Meanwhile, the training data are split into several partitions. This decomposes the large scale training procedure into several medium training procedures which can be solved separately. Besides, a large model is also trained and a simple strategy is introduced to merge the two models. Extensive experiments on the MSR-Celeb-1M dataset demonstrate the superiority of the proposed method. Our solution ranks the first and second in two tracks of the final evaluation.

Keywords

Convolutional Neural Network; Face Recognition; Independent Softmax

1. INTRODUCTION

Large scale image classification [14] has made a series of breakthroughs led by Deep Convolutional Neural Network(CNN) [9, 10]. Many other computer vision problems benefit from the deep models of CNN, such as face verification [17, 15], object detection [3, 13] and object segmentation[11]. However, by introducing the MS-Celeb-1M [5], this large scale dataset brings new challenges in image classification task. First, up to 100k classes and 10M images are considerably larger than previous publicly available datasets [14, 17, 19, 21, 8]. Second, the provided training set only covers 75% of classes in their measurement set. These challenges come up with two interesting problems: 1) *how to build a system to classify 100k classes in a short time given such large scale data?* 2) *what will happen if the model does not see the class during training?*

MM '16, October 15-19, 2016, Amsterdam, Netherlands

DOI: http://dx.doi.org/10.1145/2964284.2984060

Recently, to solve the large scale text classification problem, Yu *et al.* proposed a generic empirical risk minimization (ERM) model [20], and Rai *et al.* employed a Bayesian multi-label learning via positive labels (BMLPL) model [12]. However, the inference of BMLPL is expensive with increasing the number of labels, and ERM is difficult to extend to more large scale labels as it contains the low-rank regularization. These methods focus on text data. In image classification with CNN, it's easy to think of making the number of neurons in the last layer of a CNN directly equal to the number of classes. But as mentioned by [1], many technical challenges exist in training such kind of big model, such as the storage and seeking of massive data, GPU memory, synchronization between GPUs. These problems may increase the training time drastically. Moreover, the gradient-based optimization [4] of CNN models with a large number of output neurons has not been deeply studied. The convergence of the model is also unclear. Thus, it is difficult for training a large model that directly handles all classes at the same time.

Meanwhile, a straight answer of the second problem is that the model will return the most similar class as the result. In face recognition, this means that the classifier will find the most look like person from all people in training. But how confident the classifier will be? Taken the binary classification as an example, how could we know if a distractor sample does not belong to both two classes? The answer is we can't. But if the classifier gives a fifty-fifty prediction, it has a bigger chance to be a distractor than a sample which has 99% probability to be one class. Extended to the multi-classes problem, if the classifier give all predictions of one sample the same probability $1/N$ where N is the number of classes, this sample has a bigger chance to be a distractor. This gives us an insight that top-1 probability of one sample for multi-classes classification can also be viewed as an indicator of the confidence that this sample belongs to these classes. The higher the probability is, the more confident this sample is in, vice versa.

In this paper, we address the large scale images classification problem by introducing an ***independent softmax*** model. Instead of directly training a classifier to predict all classes at the same time, we train several independent networks with each a softmax function to predict probabilities for a part of classes. By doing this, the large number of classes and images are split into several small groups. The scale of classification problem for each group is reduced and can be solved using some well-established techniques (*e.g.* Torch [2]). This model is based on one proposition that if a

classifier does not contain one class, the prediction for images of that class will give lower score than the classifier which contains. As a special case, if all classifiers don't contain that class, all of them will make relative lower predictions compared with classes they have.

We present comprehensive experiments on MS-Celeb-1M [5] to evaluate our method. We show that the large scale classification problem can decomposed into several small classification problems using the independent softmax model. On the MS-Celeb-1M challenge, we obtain excellent results by the independent softmax model. Our final solution has 73.4% and 48.6% coverage@P=95%, which rank the first in random set and second in hard set, respectively. Moreover, when using precision equal to 99%, we rank first in both random set with 52.5% coverage and hard set with 33.6% coverage.

2. METHODS

In this section, we will describe the details of our method to the classification with 100k classes. We first review the softmax model. Then we introduce the independent softmax model. A fusion of the two models is employed to get the final result.

2.1 Softmax Model (SM)

The softmax function can be expressed as

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}, \tag{1}$$

where x_i is the i-th dimension output, N is the number of dimension which is usually equal to the number of classes. We can treat $f(x_i)$ as the probability of the i-th class. When making a prediction, a sample is assigned the class which has the largest probability, *i.e.*,

$$F(x_i) = \max_{i \in [1,N]} f(x_i). \tag{2}$$

Softmax function encounters the training problem when the number of class N is large. First, the huge parameters in the last layer not only make the forward-backward propagation slow but also make the synchronization difficult when using multiple GPUs. In practice, we find that to finish a standard 90 epochs training on the whole MS-Celeb-1M dataset, this model may need 4 months using two Titan X GPUs. Second, the convergence is also not guaranteed. In the beginning, we try to optimize the weights from random initialization. But the training loss doesn't decrease after one epoch. Thus we pre-train a model with about 10,000 classes and fine-tune it in the whole dataset. However, before the final evaluation, only 3 epochs are finished. From this we can observe that the large number of classes and images make the training difficult.

2.2 Independent Softmax Model (ISM)

To tackle the the large scale classes problem, we introduce a hypothesis, which is described as

Hypothesis: *a sample has the same probability to be any class in a softmax model if it does not belong to the classes for training.*

More precisely, the prediction of class i for a sample that is not in N classes can be written as

$$F(x_i) = \frac{1}{N}. \tag{3}$$

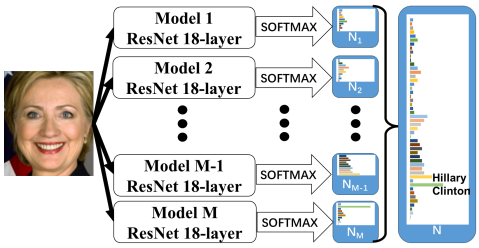

Figure 1: Independent softmax model

In other words, all classes has the same probability to be the final prediction of a sample from an unseen class. However, the probabilities for all classes cannot be the same in real world. This comes from the fact that there should always exist some class that is more similar to that sample than other classes. Instead, the key proposition of that hypothesis we utilized in ISM is

Proposition: *if a classifier does not contain one class, the prediction for images of that class will get lower score than the classifier which contains.*

Mathematically, this can be formalized as

$$F(x_{i_p}) < F(x_{i_q}), \tag{4}$$

where $p \neq q$ and $p, q \in [1, M]$. The q-th fold contains the ground-truth class i_q while the p-th fold doesn't. This proposition is validated by experiments in section 3.3 with a detailed discussion.

In our independent softmax model, the N classes are split into M folds. The k-th fold contains N_k classes, which means $N = \sum_{k=1}^{M} N_k$. Based on Eq 4, the prediction for one sample can be written as

$$\begin{aligned} \max_{k \in [1,M]} F(x_{i_k}) &= \max_{k \in [1,M]} \max_{i_k \in [1,N_k]} f(x_{i_k}) \\ &= \max_{k \in [1,M], i_k \in [1,N_k]} \frac{e^{x_{i_k}}}{\sum_{j=1}^{N_k} e^{x_j}}. \end{aligned} \tag{5}$$

According to Eq 5, probabilities for all classes are first calculated in each fold. Then the class which has the largest probability is chosen as the final prediction, which is shown in Figure 1. Different from SM, ISM separates the large number of classes into several small group of classes. This results in a partition of classes. The training images are also split according to the partition of classes. For each fold of data, it is still a classification problem. The training for them can be distributed into different machines. Moreover, due to the partition, the scale of data and classes is not that large anymore. The convergence is guaranteed according to the previous knowledge and the training time for each model is also acceptable.

2.3 Fusion of SM and ISM

To fuse the two models, we first use a nonlinear mapping to map the two scores in the same scale. After mapping, the two scores can be compared directly. To learn the mapping function, we randomly sampled 10,000 images from the validation set to make predictions using two models. Afterwards, the images that have the correct results from both two models are selected, witch results in 4,660 images. Figure 2 illustrates these scores after sorting. We can observe that the output of the two models varies in different scale. This happens because the two models are trained in different ways and data. In Figure 3, the scores from model SM and

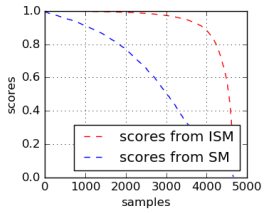

Figure 2: Scores from two models.

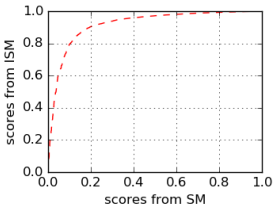

Figure 3: Non-linear mapping curve.

ISM are taken as the x-axis and y-axis, respectively. This is the curve which will be used to map the scores generated by model SM to model ISM.

Function tanh and sigmoid are first used to fit that curve independently but show bad results. Finally, a linear combination of function tanh and sigmoid is utilized. The fitted function is as following

$$s_2^1 = a/(1 + e^{-b \times s_2}) + c \times \tanh(d \times s_2), \quad (6)$$

where s_2 is the score of model A, s_2^1 is the mapped score and a, b, c, d are 0.3, 4.25, 0.7, 13.44, respectively.

The fusion algorithm is described in Algorithm 1. Our intuition is to utilize the good performance of model SM at Coverage@P=99% , which means when model SM gives a high score, it is very likely to be correct. Thus if the score of model SM is higher than a predefined threshold θ, we will use this score as the final result. The θ is set to 0.95 empirically. More details can be found in section 3.3.

Algorithm 1 Two models fusion

```
procedure FUSION(Model SM score s_1, Model ISM score s_2, threshold θ)
    Compute the mapped score s_2^1 via Eq. 6
    if s_2^1 > θ then
        Use s_2^1 as the final result
    else
        Use s_1 as the final result
```

3. EXPERIMENTS

In this section, we first review the dataset and evaluation metric. Then we go through the implementation details and the experiment results.

3.1 Dataset and Evaluation Metric

We evaluate our method on the MS-Celeb-1M dataset [5] that contains 99,892 classes. The models are trained on the provided 8.46 million images and evaluated on a development set with 1k images. We also obtain a final result on 50k test images, reported by the challenge organizer. It is notable that the training set only cover about 75% of classes in the measurement set, which means the upper bound of recognition recall is 75%. Besides, two different kinds of data are utilized for development and testing. The first kind of data have big variations and are harder, which are referred as dev1. The other kind of data are randomly selected and are easier, which are referred as dev2. We evaluate the recognition coverage at both precision 95% and 99%, which are defined as

$$\begin{aligned} \text{precision} &= c/m, \\ \text{coverage} &= m/n, \end{aligned} \quad (7)$$

where for n images in the measurement set, c images are correct among the recognized m images.

3.2 Implementation Details

To train and validate classification models, the 8.46 million images are first randomly split into a training set and a validation set. For each class, 90% of its images are for training and the rest 10% are for validation.

ISM. The dataset is further partitioned into 5 folds manually. We split the dataset by alphabetical order. The details of each fold is in Table 1. The 18-layer model from Deep Residual Network [6] are employed for the classification of each fold. A torch implement [2] is utilized. We replace the last layer with a new layer which has the corresponding classes number. Five 18-layer models are trained for five folds independently. For each model, The batch size is set to 128. The initial learning rate is set to 0.1. Each model is trained for 90 epochs. The learning rate decreases by 10 times every 30 epoch. The training takes about 7 GB GPU memory. These models are trained on a Titan X or a K40c or 2 GTX 970. Each model needs about one week to finish training.

Table 1: Dataset Partition Information

Fold	Classes	Train Images	Val Images
1	10,001	760,656	89,461
2	23,000	1,743,649	205,377
3	23,000	1,740,138	204,921
4	23,000	1,742,208	205,092
5	20,890	1,578,945	185,807

SM. For the softmax model, it is also a 18-layer Deep Residual Network model trained on the whole dataset. It has 99,892 output which each one is corresponding to one class. This model utilizes the fold 1 model as pre-trained weights and then is fine-tuned on the whole dataset. It is trained on 2 titan X with a batch size 256. It takes total 16 GB GPU memory. The training of this model is very slow. Before the challenge final evaluation, we can only train the model for 3 epochs. The first two epochs are with learning rate 0.01 and the third epoch uses 0.001.

3.3 Experiment Results

The proposition is validated in the beginning. Then we show the results of ISM, SM and Fusion of the two models. Final evaluation results are also given.

Proposition Validation. We take the model trained from the second fold to validate the Equation 4. In the 1k development images, classes of 198 images belong to the second fold. In Figure 4, scores of these images from the second model are marked as scores from "in". Scores of the rest 802 images from the second model are marked as scores from "out". The percentage of samples are taken as the x-axis and scores are used as the y-axis. From Figure 4, we can

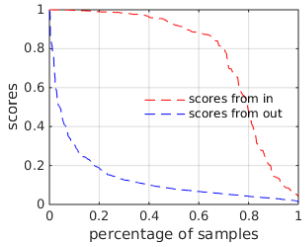

Figure 4: Results of model ISM from 'in' and 'out' data.

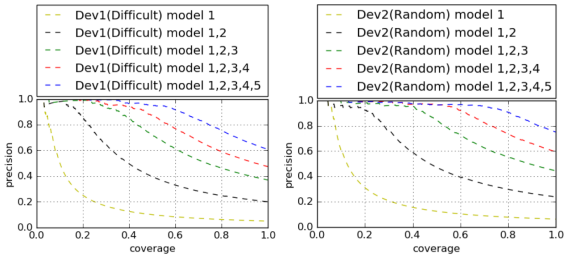

Figure 5: Results of ISM with different number of small models

Table 2: Results of ISM with different number of small models

Model	Coverage@P=95%		Coverage@P=99%	
	dev1	dev2	dev1	dev2
1	3.0	5.8	3.0	4.4
1,2	13.4	17.0	3.0	4.4
1,2,3	28.4	40.6	21.2	9.80
1,2,3,4	37.4	56.4	24.8	29.4
1,2,3,4,5	49.6	71.0	32.8	24.2

observe that 70% of the "in" images have higher scores than 0.8. Meanwhile, about only 1% of "out" images get higher scores than 0.8. This demonstrates that a softmax classifier will give lower scores to the images which classes are not in its training set, but give higher scores to the images which classes are in. Thus, the proposition is reasonable.

ISM. First we evaluate the performance of model ISM. Five independent models are utilized. When predicting, we vary the number of models from one to five which are corresponding to the fold number in Table 1. Results are shown in Figure 5 and Table 2. We can observe that when more models are added, the better the results are. This is because when more models are added, more classes are covered. It is also consistent with that more data lead to higher performance. However, when the model 5 is added, the coverage with precision 99% of dev2 decreases from 29.4% to 24.2%. This makes no sense because the 24.2% is even lower than 32.8% of dev1 which is obvious harder. We believe this is due to the limited samples of the development set. Similar phenomena are also shown in the fusion of two models.

Multi-scale Testing. Usually, in convolutional neural network, multi-scale testing [7, 16, 18], *i.e.*, ten crops or dense crops, gives better results. But since the final evaluation is tested online. we can not afford aggressive cropping approaches. Besides, We also observe that the faces are almost in the center of aligned images. Thus, we do evaluation on the original center-crop, two-crop and ten-crop [9]. Two-crop which takes the center crop of the original and flipped image are a simple version of ten-crop. The results are shown in Figure 6 and Table 3. We can observe

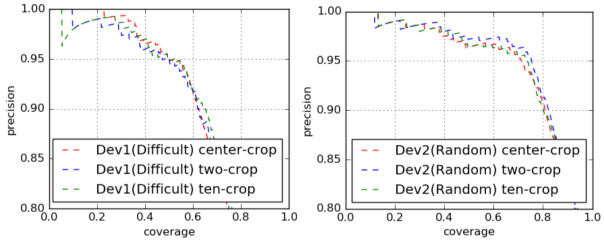

Figure 6: Results of model ISM with different testing views

Table 3: Results of model ISM with different testing views

Model	Coverage@P=95%		Coverage@P=99%	
	dev1	dev2	dev1	dev2
Center-crop	49.6	71.0	32.8	24.2
Two-crop	50.0	75.2	21.6	22.0
Ten-crop	53.2	73.6	27.4	24.6

that when using precision 95%, two-crop and ten-crop has similar performance but are both better than center-crop. Considering the speed issue, our final submission is based on two-crop.

SM. Similar results are also observed from model SM, which are not shown due to the limit space.

Fusion. Results for fusion of SM and ISM are illustrated in Figure 7 and Table 4. We find the model SM is better than model ISM when the precision is high. But the precision of SM drops faster than ISM with the increase of coverage. Fusion of SM and ISM achieve better performance than both SM and ISM except the coverage with precision 99%.

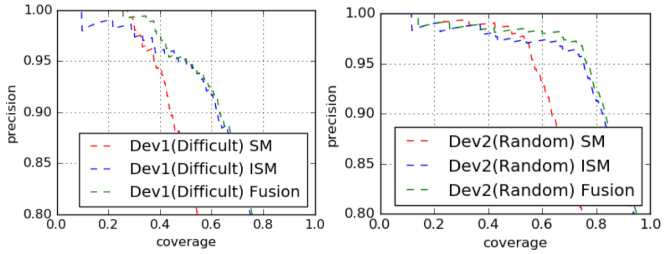

Figure 7: Fusion of SM and ISM

Table 4: Fusion of SM and ISM

Model	Coverage@P=95%		Coverage@P=99%	
	dev1	dev2	dev1	dev2
ISM	50.0	75.2	21.6	22.0
SM	37.6	57.0	29.4	42.4
Fusion	50.2	76.6	34.6	25.6

Final Evaluation. The final result on the final 5k test images are in Table 5. We can see that the at 95%, the results are similar to the results on development set except the coverage with precision 99% of dev2. This demonstrates that small development set makes that result unstable.

Table 5: Results on the final testing set

Model	Coverage@P=95%		Coverage@P=99%	
	dev1	dev2	dev1	dev2
Test	48.6	73.4	33.6	52.5

4. CONCLUSIONS

In this paper, we proposed an independent softmax model to solve the large scale face recognition problem. The model partitions the large scale classification problem into several small classification problems, which can be distributed and easily solved. The good performance in MS-Celeb-1M challenge 2016 demonstrates the superiority of our method. In the future, we plan to investigate the relationship between different models under the independent softmax model.

5. ACKNOWLEDGMENTS

This work is supported in part by the NSF CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, and U.S. Army Research Office Young Investigator Award W911NF-14-1-0218.

6. REFERENCES

[1] http://mxnet.readthedocs.io/en/latest/tutorials/imagenet_full.html.

[2] https://github.com/facebook/fb.resnet.torch/.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[4] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks.

[5] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. *arXiv preprint arXiv:1607.08221*, 2016.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[7] A. G. Howard. Some improvements on deep convolutional neural network based image classification. *arXiv preprint arXiv:1312.5402*, 2013.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[12] P. Rai, C. Hu, R. Henao, and L. Carin. Large-scale bayesian multi-label learning via topic-based label embeddings. In *Proceedings of the Neural Info. Processing Systems*, 2015.

[13] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[15] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[16] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[17] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898, 2014.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[19] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[20] H. Yu, P. Jain, P. Kar, and I. Dhillon. Large-scale multi-label learning with missing labels. In *Proceedings of the Int'l Conf. Machine Learning*, 2014.

[21] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.