One-shot Learning with Absolute Generalization

1st Hao Su
School of Information and Electronics
Beijing Institute of Technology
Beijing, China
suhao_hao@126.com

Abstract—One-shot learning is proposed to make a pretrained classifier workable on a new dataset based on one labeled sample from each pattern. However, few of researchers consider whether the dataset itself supports one-shot learning. In this paper, we propose a set of definitions to explain what kind of datasets can support one-shot learning and propose the concept “absolute generalization”. Based on these definitions, we proposed a method to build an absolutely generalizable classifier. The proposed method concatenates two samples as a new single sample, and converts a classification problem to an identity identification problem or a similarity metric problem. Experiments demonstrate that the proposed method is superior to baseline on one-shot learning datasets and artificial datasets. The code can be get https://github.com/qqsuhao/One-shot-Learning-with-Absolute-Genelization

Index Terms—One-shot learning, Absolute Genelization

I. INTRODUCTION

Deep learning models have achieved great success in image classification [1] [2]. However, in practical applications, the popularization of deep learning models still suffer from a lot of problems. The prominent problem is that deep learning models need a large number of labeled samples to train [3] [4], and once the scenes are changed, previously trained models may not work normally although the scenes change does not alter semantics about human cognition. For example, if we train a model to recognize samples with a certain texture of background, we may need to retrain the model once the texture of the background changes into another style. That is, the generalization of the model is limited, which makes it difficult for the model trained on dataset A to work on dataset B. Fortunately, studies related to one-shot learning or few-shot learning (FSL) in recent years may provide a solution to the problem [5].

The one-shot learning was first proposed by [6] and [7], aiming to learn characteristics of novel patterns from a few of labeled samples and use them for classification [8]. If one labeled sample is taken from each of the 20 novel patterns for learning, it is called 20-way one-shot learning [9]. In general, FSL methods can be roughly divided into three categories [10] [11]: (1) making augmentation and regularization based on prior knowledge to avoid over-fitting of the model [12] [13]; (2) constraining hypothesis space based on prior knowledge, including multitask learning [14] [15], embedding learning [16] [17] and so on [18]; (3) optimizing the search strategy for parameters in hypothesis spaces based on prior knowledge [19] [20].

Most datasets for one-shot learning are divided into two subsets: training set and probe set, and none of these patterns of the probe set have ever appeared in training set. The trained one-shot learning models are expected to learn through few of labeled probe samples and be able to classify other unlabeled probe samples. Therefore, it is regarded that one-shot learning models are generalizable, which can work on datasets that have never been seen before based on limited prior knowledge. If we regard the training set and probe set as two different datasets A and B, respectively, then one-shot learning models could be regarded as a general model on both datasets. Most researchers dedicate to study how to improve the generalizability by adapting models themselves. However, few of them consider the relationship between the distribution of data and the generalizability of models. That is, the generalizability of models may depend on whether the datasets support models to generalize. For example, if datasets A and B are totally different, those models trained on A certainly could not work on B. But if A and B are semantically identical, but distributed differently, those models trained on A must could be generalized on B from the perspective of human cognition.

In this paper, we define an “absolute generalization” for classification from the perspective of distribution of dataset. That is, as long as the distribution of dataset A and B satisfies certain conditions, models trained on the dataset A will be able to work on dataset B. We analyze the distribution of concatenated samples coupled by two samples and propose a method to make existing methods absolutely generalizable. The proposed method is aimed to measure the similarity between a pair of samples without any metric measurement. Finally, we compare our method with the baseline produced by siamese networks for verification [15] [9].

The rest of paper is organized as follows. In the second section, we review some related work. In the third section, we make some definitions about “absolute generalization” and propose our method to build a absolutely generalizable classifiers. Next, we apply experiments compared with siamese networks as baselines in Section 4. Finally, we point out the remaining problems and future research directions.

II. RELATED WORK

A. Image Classification

Deep learning has achieved great success in image classification [21]. Most of these classifiers are essentially data fitting
depending on labels. Consider the forerunner of convolution neural networks for recognition, LeNet [22], which succeeded on MNIST dataset for classification. LeNet fits training data to corresponding one-hot code, where appropriate optimizers, structures, active functions and loss functions need to be considered. However, for example, if we simply reverse the color of input images during probe phase, these classifiers may not work normally. Although the simple operation does not disturb human cognition, these classifiers may take a fatal blow. It is obvious that weights in the hidden layers highly depend on the distribution of training data. Once the distribution of probe data is changed, no matter whether the change alters semantics of probe data, the classifier will not make a difference. In a broad sense, we can regard that the generalizability of classifier is low.

B. One-shot learning

From the perspective of generalization, these one-shot learning models are aimed to improve the generalizability on novel datasets. We can regard a one-shot labeled sample as a template. That is, these one-shot learning models recognize unlabeled samples through predicting whether they belong to the same pattern with the template [5]. Recently, a number of one-shot learning models have been developed. In [9], Koch et al. employed deep siamese networks for one-shot learning on Omniglot dataset [8]. The siamese networks were first introduced in the early 1990s by Bromley and LeCun to solve signature verification problem [23], which aimed to measure the similarity between a pair of samples. In [15], LeCun et al. employed two weights-shared subnetworks (usually either CNNs or autoencoders) to extract features of a pair samples, respectively, as shown in Figure. 1. Then, they used a contrastive energy function which contained dual terms to decrease the energy of identical pattern pairs and increase the energy of different pattern pairs.

![Fig. 1. The structure of siamese networks.](image)

III. Methodology

A. Absolute Generalization

The concept of generalization describes that how well the model works in the probe set of a dataset. To distinguish the generic concept, we define “absolute generalization” for two datasets \( A \) and \( B \):

**Definition 1:** The absolute generalization refers to that a model which is trained in the dataset \( A \) could be employed on the dataset \( B \) without extra modification, and we call that the model is absolutely generalizable.

In above definition, the two datasets are different, which is defined as follows:

**Definition 2:** The dataset \( A \) is distinguished from the dataset \( B \) if \( f_A(x) \neq f_B(x) \).

Where, \( f_A(x) \) and \( f_B(x) \) are data distributions of the dataset \( A \) and \( B \), respectively. It’s noted that the model with absolute generalization not always exists for any two datasets. Next, we give the existing conditions.

Assume that the dataset \( A \) contains two patterns which are denoted as \( \omega_0 \) and \( \omega_1 \), respectively. So does the dataset \( B \). Generally, we use the maximum posterior estimation (MAP) to make decision in the dataset \( A \):

\[
\omega_i = \arg \max_i f_A(\omega_i|x)
\]

\[
= \arg \max_i \frac{f_A(x|\omega_i)P_A(\omega_i)}{f_A(x)}(i = 0, 1)
\]

\[
= \arg \max_i f_A(x|\omega_i)P_A(\omega_i),(i = 0, 1)
\]

Where, \( P_A(\omega_i) \) denotes prior probability, which is a constant and could be ignored if it is 0.5. Denote \( f_A(x|\omega_i) \) as a likelihood distribution of the dataset \( A \). Due to \( f_A(x|\omega_i) \neq f_B(x|\omega_i) \) and \( f_A(x) \neq f_B(x) \), or some models ignore \( f_A(x) \), the classifier trained in the dataset \( A \) could not be employed in the dataset \( B \). However, we can relax the constraints to make the “absolute generalization” available.

Assume that the observed sample \( x \) in dataset \( A \) and dataset \( B \) is generated by the following mapping:

\[
x = g(z_1, z_2)
\]

Where, denote \( g(\cdot) \) as a mapping from latent space to sample space. Denote \( z_1 \) and \( z_2 \) as two independent latent variable. As shown in following, we convert the data distribution to latent variable distribution:
As long as we can isolate the effects of

\[ f_A(x) \rightarrow f_A(z_1)f_A(z_2) \]  

\[ f_A(x;\omega_i) \rightarrow f_A(z_1;\omega_i)f_A(z_2) \]  

\[ f_A(z_1) = \sum_{i=0}^{1} f_A(z_1;\omega_i)P_A(\omega_i) \]  

\[ f_B(x) \rightarrow f_B(z_1)f_B(z_2) \]  

\[ f_B(x;\omega_i) \rightarrow f_B(z_1;\omega_i)f_B(z_2) \]  

\[ f_B(z_1) = \sum_{i=0}^{1} f_B(z_1;\omega_i)P_B(\omega_i) \]  

Because the distribution of \( z_2 \) has nothing to do with patterns, we call \( f_A(z_2) \) as the background distribution of dataset \( A \).

\textbf{Definition 3}: The classifier with the absolute generalization between two datasets exits if \( f_A(z_1;\omega_i) = f_B(z_1;\omega_i) \) and \( P_A(\omega_i) = P_B(\omega_i) \), \( i=0,1 \). In this case, the classifier is

\[ \omega_i = \arg \max_i \frac{f_A(z_1;\omega_i)P_A(\omega_i)}{f_A(z_1)} \]

\[ \omega_i = \arg \max_i \frac{f_B(z_1;\omega_i)P_B(\omega_i)}{f_B(z_1)} \]

As long as we can isolate the effects of \( z_2 \) when we construct a classifier in the dataset \( A \), the classifier is absolutely generalizable and could be applied in the dataset \( B \).

\section{Proposed Method}

Denote sample matrices of dataset \( A \) and \( B \) as \( X_A \in \mathbb{R}^{m \times N_A} \) and \( X_B \in \mathbb{R}^{m \times N_B} \), respectively. \( m \) denotes dimension of samples. \( N_A \) and \( N_B \) denote number of the samples in dataset, respectively. Each column of \( X_A \) is a sample vector and so does \( X_B \). Samples of two patterns in dataset \( A \) are denoted as \( x_{j}^{A_{0}}; j = 0, \ldots, n_{A_{0}} \) and \( x_{j}^{A_{1}}; k = 0, \ldots, n_{A_{1}} \), respectively. Where \( n_{A_{0}} + n_{A_{1}} = N_A \).

Consider converting classification problems of two patterns to distinguishing whether two samples belong to an identical pattern or different patterns. For dataset \( A \), concatenate samples belonging to an identical pattern and denote them as \( x_{iden}^{iden} \)

\[ X_{iden}^A = \{ [x_{A_{0}}^{A_{0}}; k = 0, \ldots, n_{A_{0}}] \cup [x_{A_{1}}^{A_{1}}; k = 0, \ldots, n_{A_{1}}] \} \]

concatenate samples belonging to different patterns and denote as \( x_{diff}^{diff} \). We consider that classifiers in the sample space spanned by \( X_{iden}^A \) and \( X_{diff}^A \) are still executable in the sample space spanned by \( X_{iden}^B \) and \( X_{diff}^B \) if the two datasets satisfy Definition 3.

We take a simple example to explain the observation. Assume that \( m = 2 \) and the dimension of concatenated samples is 4. As shown in Figure 2, samples of dataset \( A \) are distributed in 2-dim plane \( S1 \) with two patterns represented by red and blue, respectively. The 2-dim plane \( S2 \) is a duplicate of \( S1 \). To be able to show the 4-dim space, regard \( S1 \) and \( S2 \) as two basis plane. The two orange hollow circles denote concatenate samples belonging to different patterns and two black hollow circles denote concatenate samples belonging to an identical pattern. Note that two black hollow circle are located in a hyperplane passing the origin, denoted as blue plane \( S3 \). Moreover, two orange hollow circles are located on both sides of the hyperplane \( S3 \). In the example, if the two datasets \( A \) and \( B \) satisfy Definition 3, the slope of decision bound in the two datasets are the same, which confirms that the normal vector of hyperplane \( S3 \) is constant. As long as we solve the normal vector in dataset \( A \), it could be used in dataset \( B \) for classification directly.

We use MNIST dataset to further explain our method. We concatenate two samples from digits 0 and 1 of MNIST as a new sample and generate the sample set \( X_{diff}^A \) and \( X_{iden}^A \). We regard the raw sample as dataset \( A \) and \( A = X_{diff}^A \cup X_{iden}^A \). Then, we modify the distribution of raw data but do not alter its semantics, such as flipping image color, adding
noise and replacing texture of background. We regard these modified dataset as dataset \( \mathcal{B} \). Because dataset \( \mathcal{A} \) and \( \mathcal{B} \) have the same semantic, they satisfy Definition 3. Due to their different distributions, we regard they are two different datasets according to Definition 2. As shown in Figure 2, we used t-sne [24] to reduce the dimension of concatenated samples and use different colors to indicate different concatenate patterns. For concatenated samples from raw data (i.e. dataset \( \mathcal{A} \)) denoted as blue and red, the distribution of them is similar to our assumption in Figure 2. The concatenated samples from identical patterns are distributed nearly to the same hyperplane as other modified datasets in Figure 3.

C. Models for Neural Networks

Our method is to couple two samples together as one new sample by concatenating them along with a dimension. In image processing, we directly concatenate two vectors flattened by images when employing MLP, as shown in Figure 4(a). When employing CNN, we can concatenate two images along with the channel dimension, as shown in Figure 4(b).

As explained in Section III, the output of our model represents the normalization distance from the concatenated sample to the hyperplane. Thus, a positive or negative output indicates that the sample is on both sides of the plane. We also could regard the output as a probability of the concatenated sample belonging to \( \mathcal{X}_{\text{diff}} \), if limiting output to \([0, 1]\). If outputs indicate distances, employ mean square error (MSE) as loss function; else if they indicate probabilities, binary cross entropy (BCE) is employed as loss function.

D. Comparison with Siamese Networks

Siamese networks [23] and [22] firstly proposed the idea of coupling two samples together, where the author employed a distance measurement to describe the similarity between two hidden features extracted by two identical networks, respectively. However, in our method, we regard the concatenated sample as a new sample and analyze its distribution properties aiming to approve the generalizability of classifiers.

From the perspective of loss function, the siamese networks employ contrastive loss function based on p-norm distance. But in our method, it could be regarded that the type of distance metric is left to the neural networks to decide by itself.

The training time and inference time of our model is half that of siamese networks because our model only takes one forward propagation.

IV. EXPERIMENTS

A. Experiments on MNIST

The MNIST dataset [22] contains 70,000 samples of 10 patterns. Each sample is a single channel image of 28x28 with black background and white foreground. Due to our attention on “absolute generalization”, we only use the digit 4 and 9 in our experiment. We train our model on the raw dataset and test the model on some modified samples to valid the performance of our model. For comparison, we employ siamese neutral networks (SNN) as a contrast.

Fairly, the same structure of MLP is employed in our model and SNN. As shown in Figure 5, the whole structure is employed in our model and the first 4 layers are employed in SNN as a subnetwork that shares weights. The image samples are reshaped as vectors at the input layer of the network. The variables’ dimension drops by half as they pass through a full connection layer. Because that the output of Tanh is in block \([-1, 1]\), we use Tanh as activate layers according to 4.

In training phase, we train models with 100 epoches. For each epoch, 23582 pairs of samples are chosen randomly in the MNIST dataset with batch-size 256. We use Adam optimizer with learning rate 0.001. In probe phase, besides raw probe dataset of MNIST, some modified samples are produced with noise or different styles based on the raw probe dataset of MNIST, which are regarded as another dataset different from the MNIST dataset. As shown in Table 1, we have 10 different probe datasets including raw data, flipped black and white, adding salt-pepper noise with various noisy pixel densities, adding Gaussian noise with various variances, adding various texture styles on background. We use area under the curve (AUC) of the receiver operating characteristics (ROC) and F1-scores to measure the classification performance, where F1-score is the harmonic average of precision and recall. We
TABLE I
EXPERIMENT ON DIGITS 4 AND 9 OF MNIST.

|                  | RAW                  | FLIPPED              | SALT PEPPER NOISE(0.2) | SALT PEPPER NOISE(0.5) | SALT PEPPER NOISE(0.9) |
|------------------|----------------------|----------------------|------------------------|------------------------|------------------------|
| SAMPLES          | ![Raw Samples](image1) | ![Flipped Samples](image2) | ![Salt Pepper Noise(0.2) Samples](image3) | ![Salt Pepper Noise(0.5) Samples](image4) | ![Salt Pepper Noise(0.9) Samples](image5) |
| SSIM             | 0.317                | 0.279                | 0.316                  | 0.279                  | 0.060                  | 0.049                  | 0.016                  | 0.013                  | 0.0001                  | 0.0001                  |
| AUC (OURS)       | 0.989                | 0.986                | **0.951**              | **0.951**              | **0.711**              | **0.507**              |
| (SNN)            | 0.997                | 0.646                | 0.901                  | **0.543**              | 0.498                  |
| (LENET)          | 0.994                | 0.725                | 0.974                  | 0.820                  | 0.517                  |
| F1-SCORE (OURS)  | 0.971                | 0.971                | **0.895**              | **0.708**              | **0.669**              |
| (SNN)            | 0.983                | 0.687                | 0.837                  | 0.666                  | 0.665                  |
| (LENET)          | 0.993                | 0.784                | 0.961                  | 0.787                  | 0.673                  |

|                  | GAUSSIAN NOISE(0.5) | GAUSSIAN NOISE(0.9) | GAUSSIAN NOISE(1.5) | STYLE 1 | STYLE 2 |
|------------------|----------------------|----------------------|----------------------|---------|---------|
| SAMPLES          | ![Gaussian Noise(0.5) Samples](image6) | ![Gaussian Noise(0.9) Samples](image7) | ![Gaussian Noise(1.5) Samples](image8) | ![Style 1 Samples](image9) | ![Style 2 Samples](image10) |
| SSIM             | 0.140                | 0.114                | 0.053                 | 0.039               | 0.021                 | 0.016                 | 0.167                 | 0.142                 | 0.060                 | 0.046                 |
| AUC (OURS)       | **0.988**            | **0.954**            | **0.897**             | **0.984**            | **0.973**             |
| (SNN)            | 0.993                | 0.875                | 0.701                 | 0.593                | 0.544                 |
| (LENET)          | 0.994                | 0.986                | 0.971                 | 0.663                | 0.694                 |
| F1-SCORE (OURS)  | **0.965**            | **0.896**            | **0.828**             | **0.953**            | **0.924**             |
| (SNN)            | 0.973                | 0.811                | 0.691                 | 0.677                | 0.674                 |
| (LENET)          | 0.991                | 0.970                | 0.937                 | 0.749                | 0.759                 |

Note that the “salt pepper noise(0.2)” denotes the density of salt pepper noise is 0.2. The “gaussian noise(0.5)” denotes the variance of gaussian noise is 0.5. So do others.

operate each experiment 10 times and get the mean and standard deviation. As references, we use average Structural SIMilarity (SSIM) [25] to measure the similarity between pairs of samples.

Besides, we use LeNet trained with digits 4 and 9 of raw training dataset to compared with our model. Note that the input of LeNet is one image, which is different from SNN and ours. That is, the datasets LeNet used is different from SNN and ours. Thus, the AUC and F1-socre of LeNet is just a reference.

As shown in Table. I, when using simple structure of MLP, SNN is awkward in probe datasets adding salt-pepper noise and adding various texture styles on background. However, our model is better at these probe dataset. The LeNet does a good job on probe datasets adding noise. However, it has a poor performance on “flipped”, “style1” and “style2”. Due to the input of our model is a concatenated sample coupling two images, where one of them provides prior knowledge when another is regarded as a probe sample, our model has better performance.

B. Experiments for Face Identification

We conduct the experiment on ORL [26] faces Dataset. The dataset contains 40 patterns of faces with 10 images in each pattern. All images are stored in grayscale with image size $92 \times 112$. For each pattern of faces, the images were captured at different times, under different lighting, with different facial expressions (eyes open/closed, smiling/not smiling) and facial details (with glasses/without glasses). All images were taken against a dark, uniform background, with the front face (some slightly sideways).

Fairly, we employ the same convolution neutral networks in our model and SNN, as shown in Figure. 6, the difference between two models is the input and output of the network. Note that we use ReLU and Sigmoid as activate layers rather than Tanh. Because that the distributions of concatenated samples are complex when the patterns in a dataset is more
TABLE II
EXPERIMENTS ON ORL FACE DATASETS.

| SAMPLES          | RAW | FLIPPED | SALT PEPPER NOISE(0.1) | GAUSSIAN NOISE(50) |
|------------------|-----|---------|------------------------|---------------------|
|                  | ![Image](image1) | ![Image](image2) | ![Image](image3) | ![Image](image4) |
| SSIM             | 0.575 | 0.385 | 0.568 | 0.392 | 0.096 | 0.065 | 0.047 | 0.037 |
| AUC (OURS)       | **0.943** (0.006) | **0.910** (0.010) | **0.935** (0.007) | **0.894** (0.005) |
|                  | (SNN) | 0.943 (0.007) | 0.844 (0.019) | 0.509 (0.012) | 0.606 (0.014) |
| F1-SCORE (OURS)  | **0.900** (0.005) | **0.866** (0.009) | **0.875** (0.005) | **0.821** (0.009) |
|                  | (SNN) | 0.882 (0.016) | 0.794 (0.018) | 0.665 (0.018) | 0.677 (0.021) |

TABLE III
EXPERIMENTS ON Omniglot DATASETS.

| SAMPLES          | RAW | FLIPPED | SALT PEPPER NOISE(0.5) | GAUSSIAN NOISE(0.9) | STYLE1 | STYLE2 |
|------------------|-----|---------|------------------------|---------------------|--------|--------|
|                  | ![Image](image5) | ![Image](image6) | ![Image](image7) | ![Image](image8) | ![Image](image9) | ![Image](image10) |
| 1-SHOT 5-WAY     | **0.974** (0.006) | **0.910** (0.014) | **0.427** (0.028) | **0.190** (0.021) | **0.823** (0.015) | **0.581** (0.017) |
|                  | (SNN) | 0.968 (0.006) | 0.896 (0.018) | 0.661 (0.028) | 0.380 (0.027) | 0.743 (0.023) |
| 1-SHOT 20-WAY    | **0.916** (0.014) | **0.442** (0.028) | **0.190** (0.021) | **0.823** (0.015) | **0.581** (0.017) | **0.958** (0.006) |
|                  | (SNN) | 0.896 (0.018) | 0.661 (0.028) | 0.380 (0.027) | 0.743 (0.023) | 0.942 (0.010) |
| 5-WAY 5-WAY      | **0.823** (0.015) | **0.581** (0.017) | **0.495** (0.030) | **0.942** (0.010) | **0.813** (0.021) | **0.964** (0.012) |
|                  | (SNN) | 0.743 (0.023) | 0.495 (0.030) | 0.942 (0.010) | 0.813 (0.021) | 0.887 (0.015) |
| 5-WAY 20-WAY     | **0.861** (0.018) | **0.581** (0.017) | **0.958** (0.006) | **0.861** (0.018) | **0.970** (0.007) | **0.821** (0.009) |
|                  | (SNN) | 0.813 (0.021) | 0.942 (0.010) | 0.887 (0.015) | 0.956 (0.011) | 0.879 (0.014) |
| 20-WAY 5-WAY     | **0.956** (0.014) | **0.956** (0.014) | **0.861** (0.018) | **0.970** (0.007) | **0.964** (0.012) | **0.821** (0.009) |
|                  | (SNN) | 0.956 (0.014) | 0.956 (0.014) | 0.861 (0.018) | 0.970 (0.007) | 0.821 (0.009) |

than 2, we only use 0 and 1 to denote the label of identical pattern samples and different pattern samples, respectively.

In the experiment, we reshape all images to the size of 100 × 100. In training phase, we use 20 patterns of faces to train our model and SNN. We train models with 100 epoches. For each epoch, 660 pairs of samples are chosen randomly which are evenly split between the identical patterns sample pair and the different patterns sample pair. In probe phase, we evaluate models on probe datasets including the rest 20 patterns of faces. Similarly to the Section IV-A, we produce some images as different datasets based on the raw probe dataset.

As shown in Table. II, SNN has lower recognition rate for probe samples with salt-pepper noise compared with our model. our model has better performance than SNN in face identification.

C. Experiments on Omniglot for One-Shot Learning

The Omniglot dataset [27] is a classical dataset for one-shot learning, which is collected by Brenden Lake and his collaborators. The dataset contains handwritten character images from 50 alphabets ranging from well-established international languages. All images are divided into a 40 alphabet background set and a 10 alphabet evaluation set, which are used for training and probe phase, respectively.

For fairness, we employ the similar structure in our model and SNN as reference [9]. In training phase, no data augmentation is used. In probe phase, we produce 5 datasets based on the raw probe dataset, including “flipped”, adding salt-pepper noise with density 0.5, adding gaussian noise with variance 0.9 and styles transformation.

As shown in Table. III, our model perform better than SNN on most probe datasets. However, on the “flipped” dataset,
Our model perform worse. Theoretically, the distribution of “flipped” dataset is a mirror of raw dataset, which could be classified as accurate as raw dataset. However, in Section IV-B and IV-C the accuracy on the “flipped” dataset is far lower than its on the raw model, which may be because that weights of CNN in our model still tightly depend on the distribution of background.

V. CONCLUSION

In practical applications, we always expect that a trained model could deal with other similar classification task. One-shot learning gives a solution for the multi-task need. However, most researchers focus on improving the classification performance of model, few of them consider whether the datasets they employed are fit for one-shot learning.

In this paper, we proposed the concept “absolute generalization” in order to explain what kind of datasets were fit for one-shot learning. We believed that a classifier with absolute generalizability can be obtained when the datasets satisfied certain conditions. We proposed a method to build an absolutely generalizable classifier. In the method, a new dataset was produced by concatenating two samples of raw datasets. In the new dataset, we converted a classification problem to an identity identification problem or a similarity metric problem. The distribution of the new dataset hid a constant hyperplane which supported an absolutely generalizable classifier.

Because open source datasets cannot satisfy our conditions, we produced some artificial datasets based on open source datasets. However, these artificial datasets had a great challenge and practical significance. Experiments showed that the proposed method was superior to the baseline method, which confirmed that our concerns did influence the baseline method.

However, we found that the proposed method performed poorly when combined with CNN. So in the future, we will continue to study our method based on CNN. Besides, we will try to concatenate samples in higher dimensions for few-shot learning.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Communications of the ACM, vol. 60, no. 6, pp. 84–90, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[3] Q. Fan, W. Zhuo, C.-K. Tang, and Y.-W. Tai, “Few-shot object detection with attention-rpn and multi-relation detector,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[4] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, “Few-shot object detection via feature reweighting,” in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019.

[5] N. O’Mahony, S. Campbell, A. Carvalho, L. Krpalkova, G. V. Hernandez, S. Harapanahalli, D. Riordan, and J. Walsh, “One-shot learning for custom identification tasks; a review,” Procedia Manufacturing, vol. 38, pp. 186–193, 2019.

[6] L. Fei-Fei, R. Fergus, and P. Perona, “One-shot learning of object categories,” IEEE transactions on pattern analysis and machine intelligence, vol. 28, no. 4, pp. 594–611, 2006.

[7] L. Fei-Fei et al., “A bayesian approach to unsupervised one-shot learning of object categories,” in Proceedings Ninth IEEE International Conference on Computer Vision. IEEE, 2003, pp. 1134–1141.

[8] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, “Learning to compare: Relation network for few-shot learning,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1199–1208.

[9] G. Koch, R. Zemel, and R. Salakhutdinov, “Siamese neural networks for one-shot image recognition,” in ICML deep learning workshop, vol. 2. Lille, 2015.

[10] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Generalizing from a few examples: A survey on few-shot learning,” ACM Computing Surveys (CSUR), vol. 53, no. 3, pp. 1–34, 2020.

[11] S. Kadam and V. Vaidya, “Review and analysis of zero, one and few shot learning approaches,” in Intelligent Systems Design and Applications, A. Abraham, A. K. Cherukuri, P. Melin, and N. Gandhi, Eds. Cham: Springer International Publishing, 2020, pp. 100–112.

[12] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, “Meta-learning with memory-augmented neural networks,” in International conference on machine learning, 2016, pp. 1842–1850.

[13] I. Ziko, J. Dolz, E. Granger, and I. B. Ayed, “Laplacian regularized few-shot learning,” in International Conference on Machine Learning. PMLR, 2020, pp. 11 660–11 670.

[14] Y. Zhang, H. Tang, and K. Jia, “Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data,” in Proceedings of the european conference on computer vision (ECCV), 2018, pp. 233–248.

[15] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), vol. 1. IEEE, 2005, pp. 539–546.

[16] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra et al., “Matching networks for one shot learning,” Advances in neural information processing systems, vol. 29, pp. 3630–3638, 2016.

[17] E. Triantafillou, R. Zemel, and R. Urtasun, “Few-shot learning through an information retrieval lens,” in Advances in Neural Information Processing Systems, 2017, pp. 2255–2265.

[18] K. Allen, E. Shelhamer, H. Shin, and J. Tenenbaum, “Infinite mixture prototypes for few-shot learning,” in Proceedings of the 36th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 232–241.

[19] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” arXiv preprint arXiv:1703.03400, 2017.

[20] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, “One-shot video object segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 221–230.
[21] H. Pham, Q. Xie, Z. Dai, and Q. V. Le, “Meta pseudo labels,” arXiv preprint arXiv:2003.10580, 2020.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[23] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, “Signature verification using a “siamese” time delay neural network,” International Journal of Pattern Recognition and Artificial Intelligence, vol. 7, no. 04, pp. 669–688, 1993.

[24] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” Journal of machine learning research, vol. 9, no. Nov, pp. 2579–2605, 2008.

[25] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612, 2004.

[26] F. S. Samaria and A. C. Harter, “Parameterisation of a stochastic model for human face identification,” in Proceedings of 1994 IEEE workshop on applications of computer vision. IEEE, 1994, pp. 138–142.

[27] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, “One shot learning of simple visual concepts,” in Proceedings of the annual meeting of the cognitive science society, vol. 33, no. 33, 2011.