Abalone Age Prediction Employing A Cascade Network Algorithm and Conditional Generative Adversarial Networks*

Zhengjie Wang[1]

Research School of Computer Science, Australian National University
u6259550@anu.edu.au

Abstract. Many machine learning methods are applied to solve classification tasks, such as SVM, Decision Tree, K-Nearest Neighbor. Unfortunately, most of these methods rely heavily on handicraft features and perform poorly on the complex dataset. Artificial Neural Network(ANN) has been proved to be a powerful methodology to build a model. Cascade Correlation (Cascor), a promising dynamic network algorithm, works efficiently and effectively on both regression and classification tasks. We first build a simple feedforward network a biological classification task, abalone age prediction. Then apply Cascor algorithm, and a variant of Cascor, Cascor employing Progressive RPROP(Casper), to improve its performance. The experimental results show that all these neural network solutions perform better than Decision Tree on this task. In the end, we also apply Conditional Generative Adversarial Nets to generate training data, which affects Casper's performance.

Keywords: classification · Artificial Neural Network · Cascor · Casper · GAN · CGAN.

1 Introduction

1.1 Motivation

The abalone dataset[1] is used for the population biology of abalone. It helps abalone researchers to understand the relation between abalones' age and other physical measurements[2]. By predicting abalones' age, researchers can understand the environment better and protect this species.

A multi-label classifier is a good method to do abalone age prediction. Eight physical measurements are taken as input features. The ages are outputs, labeled from 1 to 29. Many classification algorithms perform well on other tasks are applied to this task. But these techniques have a significant loss in accuracy. CLOUDS, a decision tree method, one of the best model in the 1990s, can only achieve 26.4% accuracy on the test set[3]. C4.5, another decision tree method, can achieve 21.5%, while k-NN can achieve only 3.5%[4].

There are three major challenges in this task. The first one is the lack of instances for some classes. Abalones aging 29 is very rare. In the given dataset, there is only one instance of it. Another challenge is the difference among abalones which have similar age is very small. The last challenge is that the instances are not uniformly distributed among all the ages. We get much more instances at age 10 and age 11.

If we can build a successful classification model on this task, this model can also be applied to other classification tasks. It solves a classification task which is lack of data and has a bias in the dataset. The methodologies to overcome the lack of data issue and bias data issue can also be applied to other models.

To solve the lack of data issue, we also propose Conditional Generative Adversarial Nets(CGAN) to generate instances to train our classification model. CGANs have been successfully applied to image generation, text generation and other semi-supervised learning[17]. Our novel work is to apply it to generate samples with few dimensions features, which is a challenge for generative model.

Our main contributions: 1. We apply CasPer network to solve the abalone age prediction problem and prove that it can beat many conventional methods. 2. We apply CGAN to generate few feature vectors. As far as we know, there is no such a work to generate data which consists of sparse features or handicraft features. 3. We apply a classifier to select a generative model.

1.2 Related works

C4.5 is a landmark decision tree algorithm which is still widely used in some classification tasks[3]. It uses information gain to choose the decision nodes. It can handle multiple classification tasks, but a decision tree can easily be overfitting[5].

* Supported by Australian National University

K-NN is an instance based algorithm. It's not so sensitive to noise data, but heavily rely on the distance function. If some features are mixed together in a space, its performance will be drop sharply.

ANN is a family of algorithm inspired by biological neural networks. Signals, which are real numbers, connect different function neurons. These neurons process the inputs with linear functions, then their outputs are fired by non-linear functions. All the weights in a neuron are adjusted by the backpropagation algorithm. Multilayer feedforward networks can approximate any function[6].

The Cascade Correlation algorithm can build a powerful neural network. It trains a neural network by dynamically altering its topology[9]. Cascor is constructed from a simplest network only contents an input layer and an output layer, and both of these layers are fully connected. Then, gradually add hidden units to network one by one. Candidate hidden units are evaluated by correlation with the output layer. This algorithm can lead hidden units to learn more different features from data. It successfully solved two spiral problem with fewer units and less computation time[9]. This methodology has a great effect on follow-up research, such as LoCC[10], CasPer[11]. Constructing neural network dynamically even inspires some deep learning research, such as dynamically changing network's behavior[14].

The Cascade Algorithm Employing Progressive RPROP(Casper)[11] overcomes two drawbacks of Cascor. The main issues are the network trained by Cascor tends to be large and can't generalize well on some problems[11]. CasPer solves these by using different learning rate on the different unit region and employs RPROP, a stochastic gradient descent method[11]. It solves two spiral problem with fewer hidden units and faster training speed.

Generative adversarial network(GAN) is a powerful framework of generative models[16]. Many GAN variants have achieved success in image generation, text generation and domain adaptation[17]. GAN has two major deep neural networks, a generative model G, which represents a distribution p_z(z is a random noise) to match the real data distribution p_d, and a discriminative model D, which learns to discriminate between real and fake data[16]. This framework can be seen as a minimax two-player game[16]. The generator G learns to improve the quality of generated data, while the discriminator D learns to improve its ability to discriminate data. Both the generator and the discriminator aim to minimize their own costs and reach the Nash equilibrium[16]. When the generator can generate data that are totally accepted by the discriminator, the Nash equilibrium is achieved and the generator can generate meaningful data[16].

Conditional generative adversarial network(CGAN) is a conditional version of GAN[18]. It aims to approximate a conditional generative model $p(x|c)$ by adding c as input to both the generator and the discriminator[18]. It's a good solution for generating data with given labels.

2 Method

2.1 Dataset analysis

In the Abalone dataset, nine physical measurements describe an abalone and the last measurement, the number of rings, represents an abalone's age. The number of instances is 4177. Because the range of an abalone's age is an integer between 1 to 29. Thus, we take it as a classification task. Eight physical measurements are input features and they are mapped to 29 classes.

We plot out all the instances in **Fig.1-a**. **Fig.1-a** shows that the distribution of abalone's age approximates to a normal distribution. For some classes, such as 27, 29, the instances are rare. Class 28 even has no instance. This is a big challenge for a classifier to learn patterns in these classes and predict the correct result. Those classes only have few instances become noise for a classifier. So this is a major reason that C4.5, K-Nearest Neighbor perform poor on this task, only achieving 21.5% and 3.57% accuracy respectively.

We also visualize abalone's age distribution by using T-Distribution Stochastic Neighbour Embedding(t-SNE)[7]. For each instance, its embedding consists of all its eight features. After dimensionality reduction by using t-SNE, an instance's features vector(embedding) was mapped to a two-dimensional area and plotted in **Fig. 1-b** and **Fig.1-c**. As we can see from **Fig. 1-b** and **Fig.1-c**, each color indicates a specific class. The number indicates the class index and it is located in the middle of the dots belong to same class. **Fig. 1-b** is for the 29 classes case, while **Fig. 1-c** is for the 3 classes case, which all ages are divided into 3 age-group.

Fig. 1-b and **Fig.1-c** illustrate that all these data are difficult to divide by simple planes. The shape of dimensionality reduction results is similar to spirals in two-spiral problem. This supports the idea that using CasPer network to solve this task. **Fig.1-c** shows that for abalone age prediction, three classes problem is much easier than 29 classes problem. Many instances are very similar to each other although they belong to different classes in **Fig.1-b**. This shows that our current dataset may not have enough features to distinguish different ages. **Fig.1-c** also tell us that these are a strong overlap between different classes, which means some noisy data may be mixed in this dataset. We need more features or more data without noise.

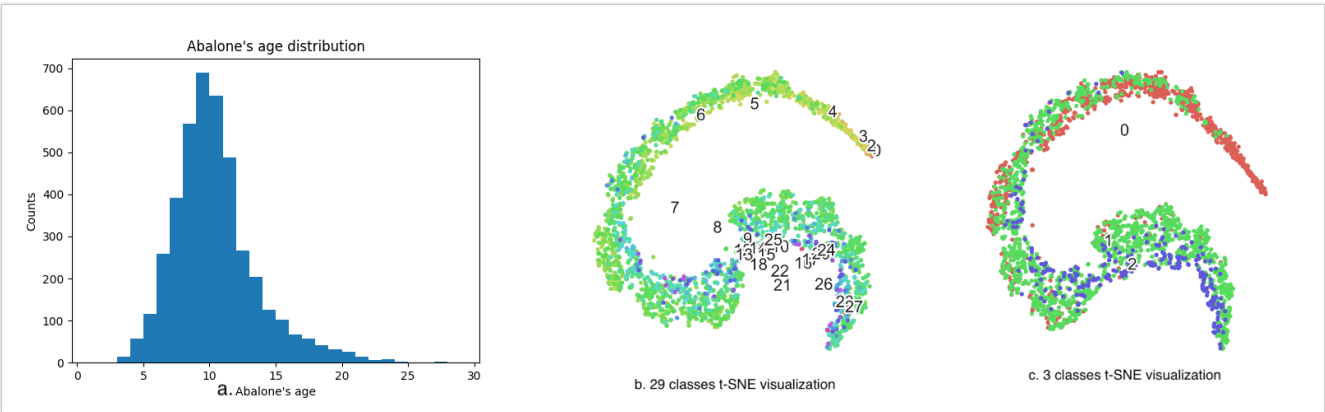

Fig. 1. Abalone's age distribution

For all the features except 'Sex', their values are all continuous float numbers between 0 and 3. Most of the features distribute in the range from 0 to 1. Thus, for the input data, normalization won't contribute much to model's performance. All the weights in ANN units should also start from a small value[8].

2.2 Dataset processing

The feature 'Sex' is an enumerative variable, which contents 'I', 'M' and 'F' to indicate abalone's sex. These values are mapped to 0.01, 0.02 and 0.03 respectively. The main reason to encode 'Sex' in this way because this feature should have similar magnitude with other features[8]. By doing this, our models also achieve better performance.

For using Cross Validation, the dataset is divided into three pieces: training set, validation set, and test set. The proportion of these datasets is 6:2:2, and the original order isn't changed. This method can increase the generalization of a model. The best model can be chosen in terms of the accuracy of the training set and validation set. The test set is used to evaluate the model's real performance.

To avoid the issue that some classes have no enough instances, David treats the dataset as a three-category classification task in their research. Class 1 to 8 are mapped to class 'Young', and class 9-14 and class 15-29 are grouped to the class 'Adult' and 'Old' respectively[12]. This process is only used for comparing the performance with Asim's research[12].

2.3 ANN solution

To solve this task, a three layer feed-forward Artificial Neural Network is built as multiple classifiers. **Fig.2** shows a brief structure of this model. The first layer takes eight features as inputs(N = 8). The hidden layer has 50 units(M = 50), and they are all activated by Rectified Linear Units(ReLU)[13]. In term of the data analysis' result, no negative value is in this dataset. Using ReLU can improve the computational efficiency and our experiment shows that it can improve the performance. The output layer is composed of 29 units(K = 29), and each unit represents a specific class, which is the age of abalone in this case. The output layer is activated by softmax, which is a common technique for the classifier.

This three-layer neural network aims to learn a distribution $P(x|I)$ and infer the age of abalone. I is all the input features of abalone and x is the age of abalone. We use the cross-entropy loss to calculate the loss of this network.

Batch stochastic gradient descent method optimizes this network during the backpropagation stage. The learning rate of SGD optimizer is 0.01.

During the training, all the training data are feed to the network as one batch, and this is also an epoch for this model. The training continuously runs for 1000 epochs. The one achieving the best accuracy on validation set will be chosen as the final model.

The parameters mentioned above are all selected by doing the experiment on the combination of different values.

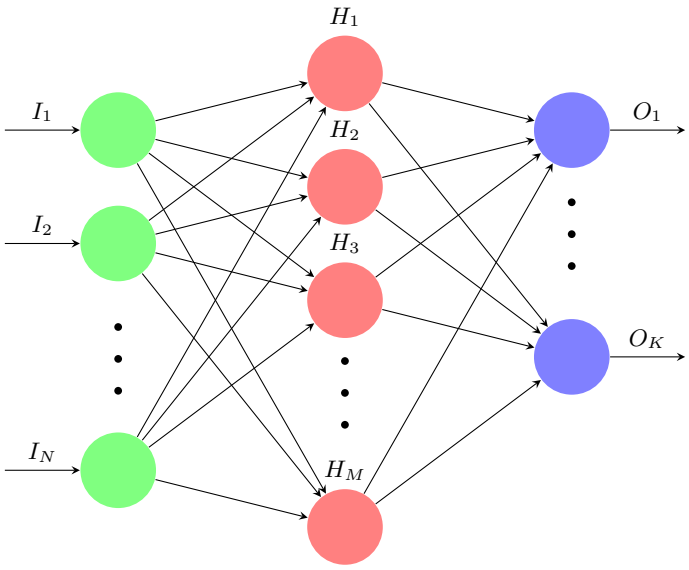

Fig. 2. Three layer neural network

2.4 Cascade Correlation solution

To improve our model's performance, we employ Cascade Correlation architecture[9]. The main reason to apply Cascade is that it successfully solves the two spiral task and get a better result than feedforward network. It has the ability to solve complex regression and classification tasks.

The input layer and output layer are as same as the three-layer network we described in section 2.3. Because cascade correlation algorithm will build the model's hidden layer dynamically. **Fig.3** shows a moment of cascade network during the training stage. The activation function for all hidden units is still ReLU. The cross entropy loss is still used to evaluate this model's loss. Batch stochastic gradient descent optimize this model in the training stage, with learning rate 0.01.This network begins with no hidden units. Training for this backbone will stop until the training loss doesn't decrease anymore or the process reaches 20 epochs.

Then the training will keep on doing if the loss is still high, and hidden units begin to be added to current network one by one.

Firstly, a candidate list composed of 8 candidates is generated. The input of candidate units are trainable, and all units from input layer and pre-existing hidden units are connected to the candidate unit. Candidates don't have any connection with output layer.

Then, all these candidates will be trained in few batches. For each batch, all the other parts of the network are frozen, only the input weights of candidates are trainable. After each batch, we will update candidate units' input weights. This update aims to maximize$S$, the covariance between a candidate unit's output, V and the residual output error of the output layer, E. The correlation in Fahlman's work[9] is defined as:

$$S = \sum_o \left| \sum_p (V_p - \bar{V})(E_{p,o} - \bar{E}_o) \right| \tag{1}$$

where o is the network output layer unit o and $E_{p,o}$ is the error measure on output unit o for pattern p. $\bar{V}$ and $\bar{E}$ are the average value over all observed patterns.

All candidates will be updated at most 20 epochs or until the best candidate is no longer updated. The candidate with largest S will be added to the network and fully connect to the output layer. Its input weights will be frozen. Other candidates will be deleted.

We will keep on training the output layer use the same stop constrain as before. To avoid the network becomes too large, the maximum number of hidden units is set to 50. This is also a stop constrain.

The best model will be chosen in terms of the accuracy on the validation set.

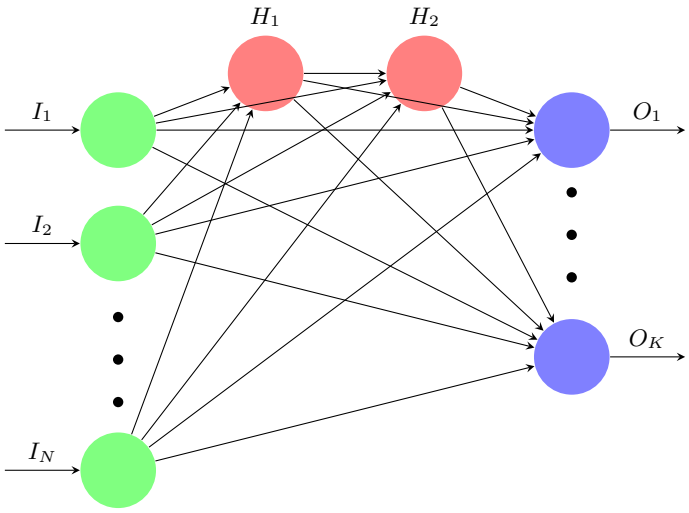

Fig. 3. One moment of Cascade network, the number of hidden units is two

2.5 CasPer solution

The CasPer algorithm successfully solved the frozen weights issue in Cascor algorithm[14]. The ideas, using different learning rate in three different regions and applying a simulated annealing term to the weight decay[11], are not used in this model. The RPROP gradient descent algorithm is adopted to optimize the Cascor model. The learning rate for RPROP optimizer is 0.01.

2.6 CGAN for generating training data

In terms of our analysis on this dataset, this dataset has two serious issues, the lack of instances, only 4177 and mixed with noisy data. To alleviate these issues, we applied CGAN to generate more data. The major reason we choose GAN to generate data is that a well-trained GAN can generate real data[16]. But whether GAN can remove noise in the training data hasn't been proved.

A CGAN model consists of two deep neural network models: a generative model G aims to learn the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G[18]. The basic structure of CGAN model is showed in **Fig.4**. The original CGAN aims to

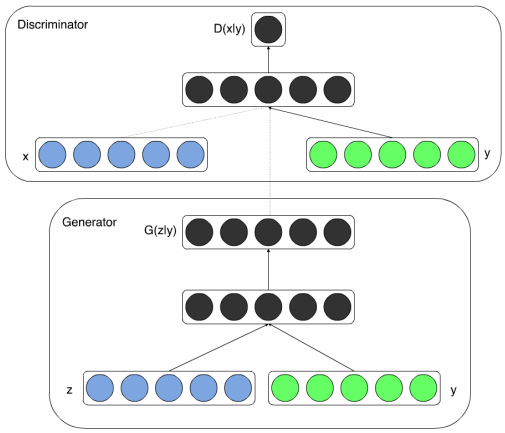

Fig. 4. Conditional Generative Adversarial Net[18]

generate images and generator G and discriminator D are designed with Convolutional Neural Networks. Our goal is to generate a feature vector or an embedding for given label. Therefore, we modify the convolutional layer to full connection layer to capture more latent features among a feature vector. The deconvolutional layers are replaced by full connection layer. The reason that we apply full connection layer is that two full connection layer can approximate any function[6]. For those activation functions between each layer, we choose the sigmoid function to replace the ReLU function because in this dataset, the negative value matters and the network is not too deep, thus we don't have serious gradient vanish or gradient explosion issues.

Our CGAN structures are based on the structure given in original CGAN paper[18]. The structures listed in **Table 1** are similar to each other but the number of parameters. The reason why we keep all these models is that we don't have a proper evaluation metrics to select a model. One possible methods is to apply their generated data to train a specific classification model, if the classifier's performance is improved, we can keep this model. Since we didn't find a paper to support this idea, we will keep all this designs and do experiments on them.

Table 1. CGAN structures

small scale CGAN		medium scale CGAN		large scale CGAN	
G	D	G	D	G	D
FC(64)	FC(16)	FC(128)	FC(32)	FC(256)	FC(64)
BatchNorm + Sigmoid	LeakyReLU	BatchNorm + Sigmoid	LeakyReLU	BatchNorm + Sigmoid	LeakyReLU
FC(256)	FC(256)	FC(512)	FC(512)	FC(1024)	FC(1024)
BatchNorm					
Sigmoid	LeakyReLU	Sigmoid	LeakyReLU	Sigmoid	LeakyReLU
FC(16)	FC(64)	FC(32)	FC(128)	FC(64)	FC(256)
BatchNorm					
Sigmoid	LeakyReLU	Sigmoid	LeakyReLU	Sigmoid	LeakyReLU
FC(8)	FC(1)	FC(8)	FC(1)	FC(8)	FC(1)
Sigmoid					
FC == full connection layer, BatchNorm == Batch normalization layer					

The input of generator G consists of two parts: a two dimension noise vector which sampled from uniform distribution $p_z(z)$, and a one-hot vector of 29 classes. These two vectors are concatenated together as the input. The output of generator G is a vector with 8 dimensions, which represents 8 given features in the real dataset. Discriminator D has the same input as generator G. The output is just a binary value, which indicates the credibility of an input instance.

The training procedure follows a two-player minimax game and the objective function of CGAN defined in Mirza's work[18] is:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x \mid y)] + E_{z\sim p_z(z)}[\log(1 - D(G(z \mid y)))] \tag{2}$$

where x is the real data, and y is the corresponding label. z is the random noise sampled from a distribution $p_z(z)$. In this minimax game, generator G tries to generate data to fool discriminator D, while discriminator D aims to improve its ability to discriminate data.

According to the objective function, training is split into two steps in each iteration. In the first step, we only train the discriminator D. The output of G and real data are combined together as the input of D, and only update weights belong to D. In the second step, we only train the generator G. The output of G is the input of D, and only update weights belong to G. Since the final output is always from D, which is a binary classifier, we choose cross entropy as the loss function and Adam[19] as the optimizer. The learning rate is 0.0002.

The training will last for 250 epochs, and we will apply our training data described in section 2.2 to training our CGAN model. After that, we will use this model to generate 1000 instances for each class and mix these data with training data to training the CasPer model.

3 Results and Discussion

To evaluate this model, we compare our model with three popular methods in the 1990s, C4.5, K-Nearest Neighbor[12], CLOUDS[12]. C4.5 and CLOUDS are classification methods based on decision tree methodology[3]. K-Nearest Neighbor is a basic statistic machine learning algorithm and widely used in clustering tasks. These methods

are applied to this dataset, but the preprocessing is different. To comparing with these methods, two different experiments are performed.

3.1 Results of training with original dataset

Table 2. 29 classes baseline

Model	Accuracy(Training)	Accuracy(Validation)	Accuracy(Test)	Accuracy(1044 instances)
C4.5	-	-	-	0.215
k=5 Nearest Neighbor	-	-	-	0.0357
CLOUDS	-	-	-	0.264
Three layer Neural Network (100 hidden units)	0.1822	0.1952	.1976	0.1820
Three layer Neural Network (500 hidden units)	0.2173	0.2204	0.2275	0.2011
Three layer Neural Network (1000 hidden units)	0.2257	0.2419	0.2599	0.2251
Cascade(50 hidden units)	0.2101	0.2156	0.2395	0.2079
Cascade(100 hidden units)	0.2089	0.2431	0.2443	0.2117
CasPer(5 hidden units, only RPROP)	**0.2783**	0.2910	0.2994	0.2615
CasPer(50 hidden units, only RPROP)	0.2747	0.2838	**0.3078**	**0.2711**
CasPer(100 hidden units, only RPROP)	0.2576	**0.2982**	0.3042	0.2625

In the first experiment, all the age of abalone is mapped to 29 classes[1]. **Table 2** shows the result. For our model, the entire dataset is split into three datasets using the original order, with the proportion, 6:2:2. Thus, only first 60% is applied to train our model. The rest data are used for doing test and validation. For baseline models, they all use the first 3133 instances(75% data) to train their model. The rest data (1044 instances, 25% data) to test the performance. Our model also does the test on the 1044 instances dataset, and this dataset doesn't have any overlap with our training dataset.

On the 1044 instances test set, CasPer model with 50 hidden units beats all other models, achieving 27.11% accuracy. On our split dataset, CasPer model achieves the best result.

The Cascade model can learn more different features. The result in **Table 2** shows that the Cascade model with 50 hidden units achieves similar accuracy as three-layer network with 500 hidden units. When training the candidate hidden units by maximum their correlation with current network residual errors, these candidate units extract hidden features different from the exist hidden units. This result also proof that units in the same layer of a feed forward network easily learn similar features. Therefore Some redundant units exist in such a network. It is necessary to prune some redundant units.

CasPer model significantly improves the neural network's performance and reduce the network's complexity. CasPer model only uses 5 hidden units to get a better performance than all other networks. RPROP gradient descent method extends the Cascade model's ability. The Cascade model aims to maximize the difference of each hidden unit. RPROP gradient descent uses separate adaptive learning rate for each weight. All the weights in a unit can learn more features without relying heavily on the initial value and the same learning rate.

Among all our models, the performance on validation set and test set are better than that on the training set. One main reason is that the training set has different instances from all 29 classes, while validation and test set only has instances from 19 classes. Thus, these datasets may over-estimate our model. Even the 1044 instances dataset, it also only has 20 classes.

The Casper model with 100 hidden units shows that more units can hurt a network's generalization. A large number of units can lead the model focus on the major data. Those minor instances are more likely to be ignored.

In this task, all the given features are very difficult to be linearly separated. The bad performance of baseline methods proof this is a tough task. But the Cascade and CasPer model still perform better than those methods.

In the second experiment, all the age of abalone are mapped to 3 classes(grouping age 0-8, 9-14, 15-29).**Table 3** shows the results. For our model, we still do the same processing as we do in the first experiment. Only modifying the output layer units to three.

Table 3. 3 classes baseline

Model	Accuracy(Training)	Accuracy(Validation)	Accuracy(1044 instances)	Accuracy(Total)
C4.5	-	-	-	0.7225
k Nearest Neighbor	-	-	-	0.6675
Three layer Neural Network (100 hidden units)	0.7049	0.7401	0.7653	0.7280
Cascade(50 hidden units)	0.6962	0.7497	0.7760	0.7280
CasPer(5 hidden units, only RPROP)	**0.7456**	**0.7772**	**0.7928**	**0.7682**

The task is much easier than the first one. For each category, there are more instances for training, but more noise is also added to each category. The decision tree method C4.5 can easily focus on the dominant features during building the tree, while the neighbor vectors of k-NN can represent their class better[12].

In the second experiment, CasPer's performance still dominants all other model's. Other two neural network models can also perform well, but they don't improve much compared to C4.5.

This experiment proved that CasPer model has the ability to denoise. For combined classes, much noise is mixed with the data. Instances located on the boundary, such as age 14 and age 15, their features are very similar to each other. These similarities can confuse a classifier. Meanwhile, instances located in the same class, but has a big gap, such as age 15 and age 29, is quite different. These difference can crash a classifier easily. To some extent, CasPer model overcomes these issues successfully.

3.2 Generated Dataset

It's difficult to evaluate the quality of generated data. A common method is to sample some generated data and judge its quality by human beings. For example, we can submit the generated text or images to Amazon MTurk and use human feedback to evaluate our work. For generated images, we can also use Inception Score[20] to evaluate its diversity and reality. But in our case, each generated instance is an 8 dimension vector. Thus, we still apply t-SNE[7] to visualize the generated data. The first row of **Fig.5** shows the loss changes of three different scale CGAN models.

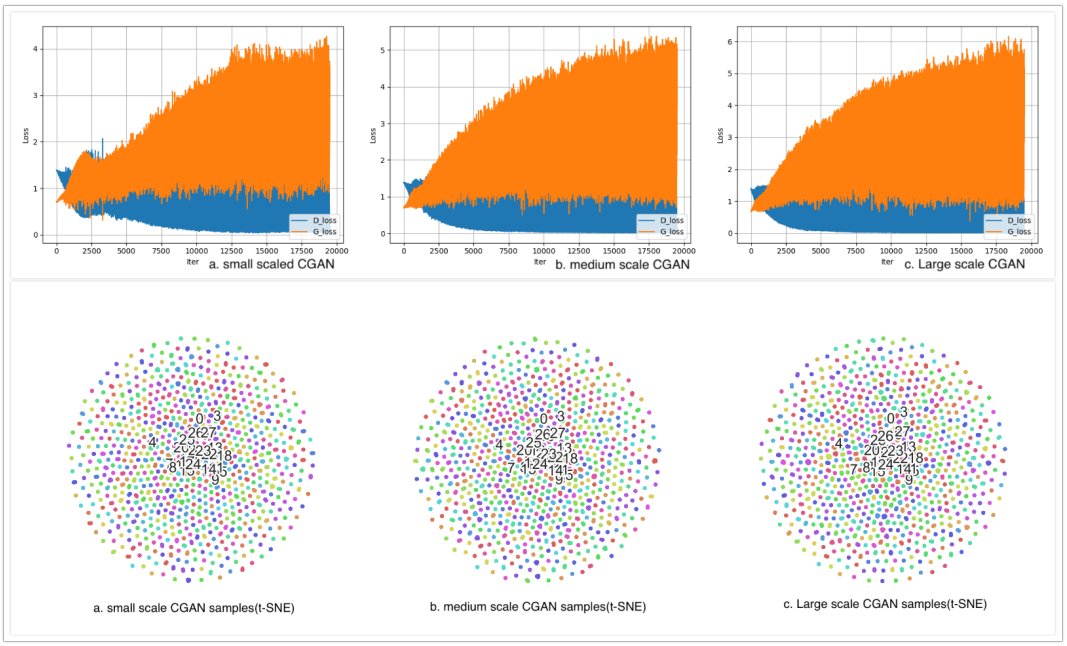

Fig. 5. training loss and generated data

Although these losses fluctuate more frequently and acutely, the loss of D and the loss of G tend to a stable upper

bound and lower bound. The minimum value of discriminator and generator also tend to be smaller than that at the beginning. An interesting phenomenon is that the upper bound of discriminator getting larger as the scale of the network growing. One possible reason is that our training dataset is much smaller than the number of parameters in the model. Many weights can not be trained sufficiently, leading to the growing of loss upper bound. Another possible reason is that in the large scale CGAN, the discriminator is likely to remember all the real data, and no matter how the generator tweak its weights, the discriminator can always figure out the output of the generator are fake data.

The second row of **Fig.5** illustrates the sampled generated data with t-SNE. Comparing the distribution image in **Fig.1**, the generated dataset doesn't show a spiral shape, which means CGAN model may fail to learn the distribution of real data. The generated data are mixed together in the diagram. The noise data in the training data set may affect the model. Our CGAN model didn't recognize the noisy data and ignore them successfully. Although the generated data from three models are quit similar to each other, data from small scale CGAN have more overlap than other two models. We can see that the labels in small scale model's t-SNE diagram are more close to each other than that of the other two model. These means the center of each clustering are close to each other and the dataset has more overlap.

3.3 Results of training with original data and generated data

After training our CGAN models, we sampled 1000 instances for each class from the trained CGAN model(e.g. 29000 instances from each model). Then, we mix the generated data with the training dataset in section 2.2. The results are shown in **Table 4**.

In **Table 4**, 'real training' means the training dataset we get from the original dataset. 'small CGAN' means the dataset sampled from small scale CGAN model(so do 'medium CGAN' and 'large CGAN'). CasPer(50) is CasPer model with 50 hidden units, which is the best model we get so far.

Table 4. 29 classes experiment

Model	Training set	Accuracy(Training)	Accuracy(Validation)	Accuracy(Test)	Accuracy(1044 instances)
CasPer(50)	real training	0.2747	0.2838	**0.3078**	**0.2711**
CasPer(50)	real training + small CGAN	0.8680	0.1210	0.1305	0.1322
CasPer(50)	real training + medium CGAN	**0.9299**	**0.1557**	**0.1629**	**0.1542**
CasPer(50)	real training + large CGAN	0.8976	0.1365	0.1401	0.1389

As we can see from **Table 4**, after add generated data to training dataset, the performance sees a sharp decline, comparing to the model trained on the original training dataset. On the training dataset, the model trained on mixed dataset seems to achieve an excellent improvement, and the highest accuracy reaches 92.99%. But the confusion matrix shows that, the model fits very well on the generated data rather than real data. Since the generated data dominants the training dataset. The model trained on mixed dataset can achieve a great result.

Because the model trained on mixed dataset fits very well on the generated data, it failed to learn more patterns from the real dataset. This means that the generated data is not real enough to help the model improve its performance. The generated dataset have some patterns which are easy to learn, and CasPer model learned them sufficiently.

On the other hand, among all the model trained on a mixed dataset, the one trained on 'real training + medium CGAN' dataset achieve the best result. This proves that the data generated by medium scale CGAN are better than the data generated by other CGANs. Thus, we can use a classifier to evaluate a generative model. If the generated dataset can improve the performance of a classifier, it should be a better generative model, vice versa.

4 Conclusion and Future Work

4.1 Conclusion

In this task, the CasPer network performs well over other models. The architecture of a neural network, optimizer and activate function play big roles in constructing a neural network. Dynamically constructing a network is proofed to be a powerful method for this task. It's still a valuable guideline for deep learning, such as perform different behaviors in different condition[14].

Applying CGAN to generate training data for improving the classifier's performance doesn't work well. The number of instances may be not enough to train a GAN model and the noisy data in the training set also affects the training process.

It's a good idea to use a classifier to evaluate a generative model. Our experiments show that this method can efficiently illustrate the difference between different generative model.

4.2 Future work

In the future, we will focus on building a classification model which can perform well with limited training data and noisy dataset.

Firstly, it is important to do a detailed analysis of the given dataset. By doing this, we can clearly understand the challenge in a specific task and which methods should be applied in this task. Then, using a reasonable method for encoding features is helpful for building model[14]. To analyze the functionality with the distinctiveness[15] of CasPer network should be an interesting task, which allows me to get a better understanding of the details of this network. We can also know more limitation of CasPer network.

Secondly, as we analyzed in section 2, for many classes, there are only very few data for some classes. It's difficult to extract useful features for these classes. Meanwhile, the age of abalone is quite like a Gaussian distribution. If we assume all the instances are mapped to an independent Gaussian distribution $N(\theta, \Sigma)$, we can use a neural network to approximate such a distribution $P(x|features)$.

In the end, we can do research on meta learning, reinforcement learning, and other GAN network. These study areas aim to build a model can learn from few shot examples. We will be inspired by these methods.

References

1. Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
2. Nash, Warwick J, and Tasmania. Marine Laboratories, 1978. The population biology of Abalone (Haliotis species) in Tasmania. I. Blacklip abalone (H rubra) from the North Coast and the Islands of Bass Strait. Sea Fisheries Division, Marine Research Laboratories - Taroona, Department of Primary Industry and Fisheries, Tasmania
3. Alsabi, K, Ranka, S, Singh, V, 1998. CLOUDS: A Decision Tree Classifier for Large Datasets
4. Waugh, S.G, 1995. Extending and Benchmarking Cascade-Correlation
5. Chawla, N.V., 2003 C4.5, and Imbalanced Datasets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure. Proceedings of the ICML.
6. Hornik, K, Stinchcombe, M, White, H, 1989. Multilayer feedforward networks are universal approximators. Neural networks.
7. Maaten, L, of machine learning research, H.-G., 2008. Visualizing data using t-SNE. Journal of machine learning research.
8. Bustos, R.A, and Gedeon, T.D., 1995. Decrypting Neural Network Data: A GIS Case Study. Artificial Neural Nets and Genetic Algorithms.
9. Fahlman, SE, Lebiere, C., 1990. The cascade-correlation learning architecture. Advances in neural information processing.
10. Khoo, S, Gedeon, T.D., 2008. Generalisation Performance vs. Architecture Variations in Constructive Cascade Networks. International Conference on Neural Information
11. Treadgold, NK, Gedeon, T.D., 1997. A cascade network algorithm employing progressive RPROP. International Work-Conference on Artificial
12. Asim, A, Li, Y, Xie, Y, Zhu, Y, Peng, J, 2002. Data Mining for Abalone.
13. Nair, V, Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. Appearing in Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 2010
14. Odena, A., Lawson, D., Olah, C., 2017. Changing Model Behavior at Test-Time Using Reinforcement Learning. arXiv.
15. Gedeon, T.D., Harris, D, Network Reduction Techniques, Proc. Int. Conf. on Neural Networks Methodologies and Applications, AMSE, San Diego, vol. 2, pp. 25-34, 1991.
16. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative Adversarial Networks. arXiv.
17. Lucic, M., Kurach, K., Michalski, M., Gelly, S., Bousquet, O., 2017. Are GANs Created Equal? A Large-Scale Study. arXiv.
18. Mirza, M., Osindero, S., 2014. Conditional Generative Adversarial Nets. arXiv.
19. Kingma, D., Ba, J., 2014. Adam: A Method for Stochastic Optimization. arXiv.
20. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., 2016. Improved Techniques for Training GANs. arXiv.