HIROSHIMA UNIVERSITY

MASTER THESIS

Invariant Feature Extraction for CNN Classifier and its Application to Medical Diagnosis

Author:
Michiaki UEDA (M204886)

Supervisor:
Professor Takio KURITA
Sub Supervisor:
Associate Professor Miyao JUNICHI
Professor Hiroaki MUKAIDANI

A thesis submitted in fulfillment of the requirements for the degree of Master of Informatics and Data Science

in the

Division of Advanced Science and Engineering
Informatics and Data Science Program

January 20, 2022

Declaration of Authorship

I, Michiaki UEDA (M204886), declare that this thesis titled, "Invariant Feature Extraction for CNN Classifier and its Application to Medical Diagnosis" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Michiaki Ueda

Date: 2021

HIROSHIMA UNIVERSITY

Abstract

Graduate School of Advanced Science and Engineering
Division of Advanced Science and Engineering
Informatics and Data Science Program

Master of Informatics and Data Science

Invariant Feature Extraction for CNN Classifier and its Application to Medical Diagnosis

by Michiaki UEDA (M204886)

In recent years, Deep learning has been successfully applied to a variety of tasks. However, there is a lot of information in the training data that is unnecessary for the task, and it is difficult to automatically remove such unnecessary information from the trained model. For example, in the case of medical diagnosis, the patient variability in the measurement data needs to be ignored. Therefore, I propose two methods for training models by removing unnecessary information and extracting only the information that is relevant to the target task: the first is a model that applies a module called Gradient Reversal Layer. The second is a model that uses Siamese Neural Network. In order to demonstrate the effectiveness of the proposed methods, I tested them on three datasets containing unwanted information: the first is a clothing image classification dataset containing unwanted shift information; the second is a personal classification dataset containing unwanted facial expression information and the third is a patient classification dataset containing unwanted facial expression information. The third is a medical dataset with unwanted variation due to patient differences. In the medical dataset, only the model with the Gradient Reversal Layer was tested. In all experiments, I confirmed the improvement of the desired classification accuracy and the reduction of unwanted information in the features.

Acknowledgements

I would like to express my gratitude to Professor Takio Kurita, Associate Professor Junichi Miyao, Professor Hiroaki Mukaidani, and Assistant Professor Hiroaki Aizawa. They provided the best environment for my research, supported my student life, and helped my research with many ideas. Especially, Professor Kurita always consulted me about my research and helped me with his accurate advice and sharp pointers. Also, Associate Professor Miyao and Assistant Professor Aizawa helped me with their opinions, reference papers, and the introduction of data preprocessing software in our regular seminars. I saw Professor Mukaidani often discussing in other labs, which made me feel that I should do my best. I am also grateful to Professor Yukiko Nakano and Assistant Professor Shogo Miyamoto. They provide many knowledge and dataset. My research was completed thanks to their knowledge and advice. I am also grateful to my lab members and my family.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	**A**rtificial **I**ntelligence
HLAC	**H**igher **o**lder **L**ocal **A**uto-Correlation
SIFT	**S**cale-**i**nvariant **F**eature **T**ransform
CNN	**C**onvolutional **N**eural **N**etworks
GRL	**G**radient **R**eversing **L**ayer
KDEF	the **K**arolinska **D**irected **E**motional **F**aces
ECG	**e**lectro**c**ardio**g**ram
MLP	**M**ulti**l**ayer **P**erceptron
ReLU	**Re**ctified **L**inear **U**nit
SGD	**s**tochastic **g**radient **d**escent
BrS	**Br**ugada **s**yndrome
PCA	**P**rincipal **C**omponent **A**nalysis

Chapter 1

Introduction

In recent years, AI technology has been attracting increasing attention around the world. This is because pattern recognition has led to the development of automation of things, and AI technology is being incorporated into things that are indispensable to humans. In such AI technology, I have been working on the theme of extraction of invariant features. The extraction of invariant features is one of the central themes in pattern recognition. For example, in object detection, the size of the object or its position in the image is not important. In medical diagnosis, the variation of measurement signals due to different patients [14] and devices [5] degrade the performance of disease diagnosis. Many methods have been proposed to extract such invariant features. For example, a shift-invariant feature extraction method called Higher older Local Auto-Correlation (HLAC) has been proposed [23], demonstrating the effectiveness of HLAC features in face recognition [18, 10]. As an application of this, the log-polar transformation was introduced as a preprocessing step to extend the HLAC features to be scale-invariant and applied to face detection[17, 12]. It is not easy to extend such invariant features in the 2D image plane to 3D by camera projection. An invariant feature extraction method has also been proposed to solve this difficulty by introducing a projected motion group [29]. Scale-invariant feature transform (SIFT) has also been proposed as a feature detection algorithm to detect and describe local features of an image[21, 7]. In recent machine learning, the aforementioned Convolutional Neural Networks (CNN) are often used for object recognition, speech recognition, image retrieval, and natural language processing. Even in these cases, invariant feature extraction is an important topic. In the recognition of the number in the image, the numbers are centered by using their positional information, and then the pre-processed images can be fed into the deep network. However, this approach cannot be used if the pre-processing of the data is complex and inaccurate, or if the information required to perform the pre-processing of the data is not satisfactorily available. Another approach is to train CNN to remove unnecessary variations (variant information) by using a large number of training samples. For example, in the field of face recognition, CNNs are used to extract posture invariant features for pose invariant face recognition [1] and metric learning is often used

to extract invariant features [13]. It is also possible to extract rotation-invariant features with metric learning [20]. Invariant feature extraction is also important in medical diagnostics due to the variability of measurement signals as mentioned above. Learning, for example, to ignore the variability by dividing the ECG analysis into multiple tasks for each content has been proposed [15]. Few-shot learning, which uses a network pre-trained on different datasets to train an additional dataset for the desired bit of data, can also extract robust features[19].

In this study, I propose two methods for extracting invariant features. The first is a method that explicitly removes variant information by applying a module called Gradient Reversing Layer (GRL), which works by inverting the learning gradient, and extracts only the invariant features necessary for the task. Specifically, I classify the variant information at the same time as I classify the target class labels. I attempted to remove the variant information by using a gradient inversion layer for learning the classification of the variant information. The second method is to remove the variant information by referring to the Siamese Neural Network model. Specifically, I use data with different variant information as input pairs for the Siamese Neural Network. Under these conditions, I calculated the similarity between the target classifications and the feature vectors, which is a characteristic structure of the Siamese Neural Network.

In order to demonstrate the effectiveness of the proposed method, I conducted experiments under the assumption of three types of variant information. The first dataset is Fashion Mnist [31], which is a fashion classification dataset with variant information called Shift. On this dataset, I tested the above two methods. The next dataset is the Karolinska Directed Emotional Faces (KDEF) [4] personal classification dataset. Since this dataset contains images of seven facial expressions that differ from person to person, I used these facial expressions as variant information for validation. Finally, I considered the application of electrocardiogram (ECG) data to medical diagnosis. This is to identify Brugada syndrome, a heart disease. It is necessary to construct the model in which important information for the classification of the Brugada syndrome is extracted but variant variations due to the differences of the patients. This dataset was validated using only the first proposed method, the GRL-based method. The second proposed method, the Siamese Neural Network model, will be tested in the future.

This paper is organized as follows. First, Section 2 reviews related studies. Section 3 gives details and experimental results of feature extraction methods using GRL. In Section 4, the details and experimental results of the feature extraction method using the Siamese Neural Network model are presented. Section 3 shows the details and experimental results of feature extraction using the Siamese Neural Network model, and Section 5 provides conclusions and future work.

Chapter 2

Related Works

2.1 Neural Network

Neural networks are a type of machine learning used in a wide range of fields. It is said that[2] the origins of this research can be traced back to the 1940's, when researches([22], [30], [25], [26]) using biological systems to mathematically process information were conducted. The error backpropagation method is a learning method for the parameters of this neural network. This method is explained in Section 2.1.1 using the multilayer perceptron, which is considered to be the most practical of neural networks. In addition, CNN, which was created by taking a hint from the visual information processing of the brain in deep learning, a topic of artificial intelligence in recent years, will be explained in 2.1.2.

2.1.1 Multilayer Perceptron

Multilayer Perceptron (MLP), as introduced in [9], is a method that has been primarily studied in neural networks. It is generally difficult to fit a complete numerical model to a variety of incomprehensible problems. Therefore, nonlinear functions can be modeled and trained to generalize when presented with new, unseen data. MLP consists of a simple system of interconnected neurons, as shown in Figure 2.1. It can also be understood by dividing it into an input layer, an output layer, and several hidden layers. The computation in the hidden layer is to derive the hidden layer unit z_j from the input unit x_i as shown below.

$$z_j = h(\sum_{i=1}^{I} w_{ji}^{(1)} x_i + x_0) \tag{2.1}$$

where the 0-th unit is the bias and also w_{ji} is the weight connecting the i-th input unit and the j-th hidden layer unit. Furthermore, $h(\cdot)$ is the activation function, which includes the sigmoid function $h_{sigmoid}$ and the Rectified Linear Unit (ReLU) function h_{ReLU}. The equations for each are as follows

$$h_{sigmoid}(x) = \frac{1}{1 + \exp(\mathrm{x})} \tag{2.2}$$

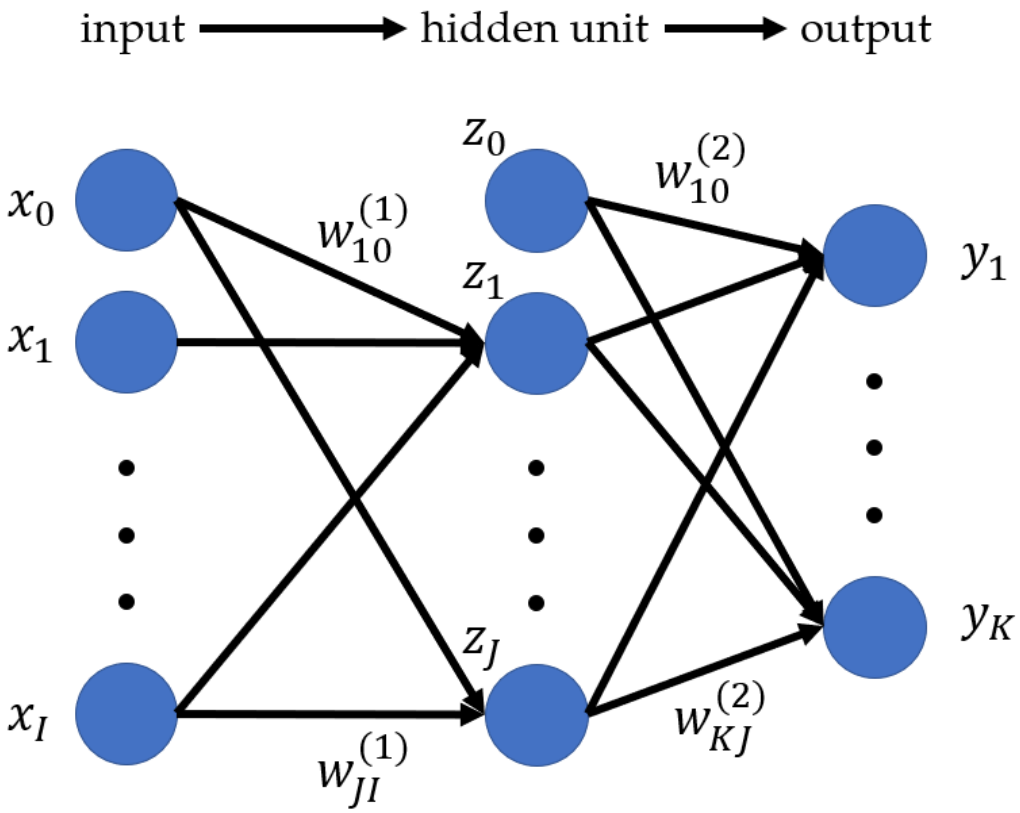


FIGURE 2.1: Multilayer Perceptron

$$h_{ReLU}(x) = \begin{cases} 0 & (x < 0) \\ x & (x \geq 0) \end{cases} \tag{2.3}$$

Similarly, the output unit y_k can be written from the hidden layer unit z_i using the weight w and the bias z_0 of the hidden layer between them as follows

$$y_k = h(\sum_{j=1}^{J} w_{kj}^{(2)} z_i + z_0). \tag{2.4}$$

The goal of MLP is then to estimate the target vector $\mathbf{t}$ from the input. For its estimation, the error backpropagation method proposed by Rumelhart et al. [27] is used. Backpropagation is an algorithm that propagates the obtained error from each unit of the output layer, calculates the difference of each error, and finds the weight parameter that minimizes the error. The error function to be minimized is as follows

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n||^2 \tag{2.5}$$

where n is the number of samples.

For this error function, I can minimize it by considering the gradient in all neurons and using the stochastic gradient descent method (SGD). As an example, the

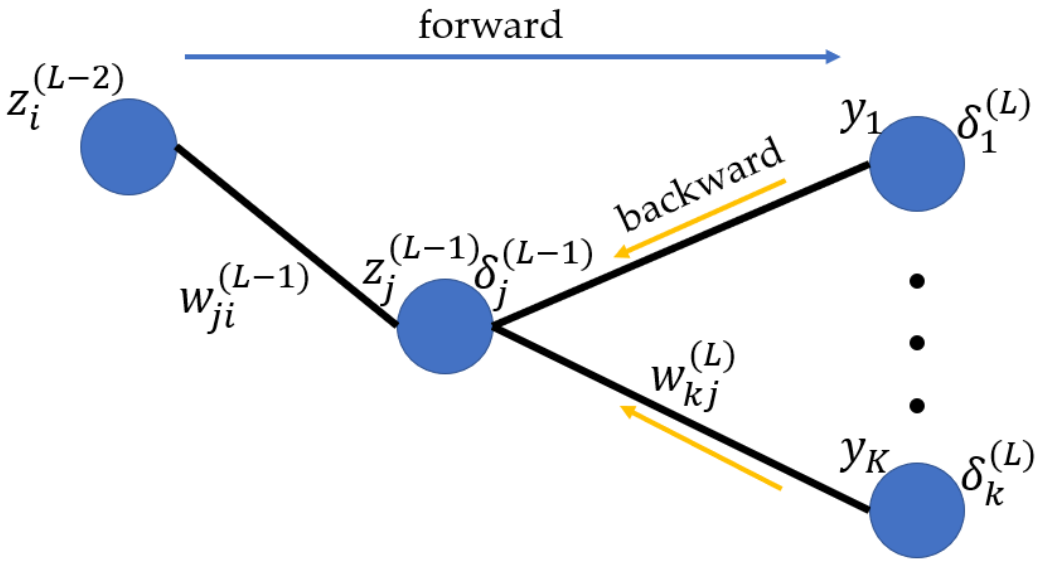


FIGURE 2.2: backward propagation

SGD for $w_{ji}^{(l)}$ is

$$w_{ji}^{(l)} \leftarrow w_{ji}^{(l)} - \mu \frac{\partial E_n}{\partial w_{ji}^{(l)}} \tag{2.6}$$

where μ is the learning coefficient and l is the layer number. The procedure for finding the gradient about w will be shown next with reference to Figure 2.2. I simplify the forwards to the hidden layer as in Equation (2.1) as follows

$$z_j^{(L-1)} = h(a_j^{(L-1)}) \tag{2.7}$$

where L is the Number of layers. Here I introduce a useful notation

$$\delta_j^{(L-1)} \equiv \frac{\partial E_n}{\partial a_j^{(L-1)}} \tag{2.8}$$

where δ is often referred to as the error. Then, considering the gradient for $w_{ji}^{(L-1)}$, it can be transformed as follows

$$\frac{\partial E_n}{\partial w_{ji}^{(L-1)}} = \frac{\partial E_n}{\partial a_j^{(L-1)}} \frac{\partial a_j^{(L-1)}}{\partial w_{ji}^{(L-1)}} \tag{2.9}$$

$$= \delta_j^{(L-1)} z_i^{(L-2)}. \tag{2.10}$$

Also, by the chain rule for $\delta_j^{(L-1)}$, the partial derivative is

$$\delta_j^{(L-1)} \equiv \frac{\partial E_n}{\partial a_j^{(L-1)}} \tag{2.11}$$

$$= \sum_k \frac{\partial E_n}{\partial a_k^{(L)}} \frac{\partial a_k^{(L)}}{\partial a_j^{(L-1)}} \tag{2.12}$$

$$= h'(a_j^{(L-1)}) \sum_k w_{kj}^{(L)} \delta_k^{(L)}. \tag{2.13}$$

Furthermore, the error function for the n-th sample is

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2. \tag{2.14}$$

Then the output layer error $\delta_k^{(L)}$ is

$$\delta_k = \frac{\partial E_n}{\partial y_{nk}} = \frac{\partial \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2}{\partial y_{nk}} = y_{nk} - t_{nk} \tag{2.15}$$

So far, I have explained how to train on the n-th sample. Usually, this kind of learning is not done for each sample, but for a subset of multiple samples, called a mini-batch. Therefore, the iterations when all samples are trained are different from the number of samples. The unit for learning all samples is called an epoch.

2.1.2 Deep Convolutional Neural Network

Convolutional neural networks (CNNs) have been exploding in research since 2012 when Krizhevsky et al. [16] showed that their discrimination accuracy was significantly better than conventional methods. Many CNNs can be divided into two parts: the part that overlays the convolutional layer and the part that overlays the all-combining layer. Most of the CNNs can be divided into two parts: the convolutional part and the full concatenative part. The full concatenative part has the same structure as the MLP mentioned above. Therefore, the most distinctive part of a CNN is that it contains convolutional layers. Figure 2.3 shows an example of how the convolutional layer works. The convolution prepares weights of a size corresponding to a small region, called the kernel size. Using these weights, the output of the neuron is calculated as follows

$$z_j = h(\mathbf{w}^T \mathbf{x}_j + x_0). \tag{2.16}$$

where $w = [w_1, \ldots, w_{kernel\ size}]^T$ and $x_j = [x_j, \ldots, x_{j+kernel\ size}]^T$. Such convolutional operations are usually performed with multiple weights per convolutional layer. For each weight, one neuron vector is generated. Therefore, the next layer will generate neuron vectors for multiple channels. The gradient descent method is often used

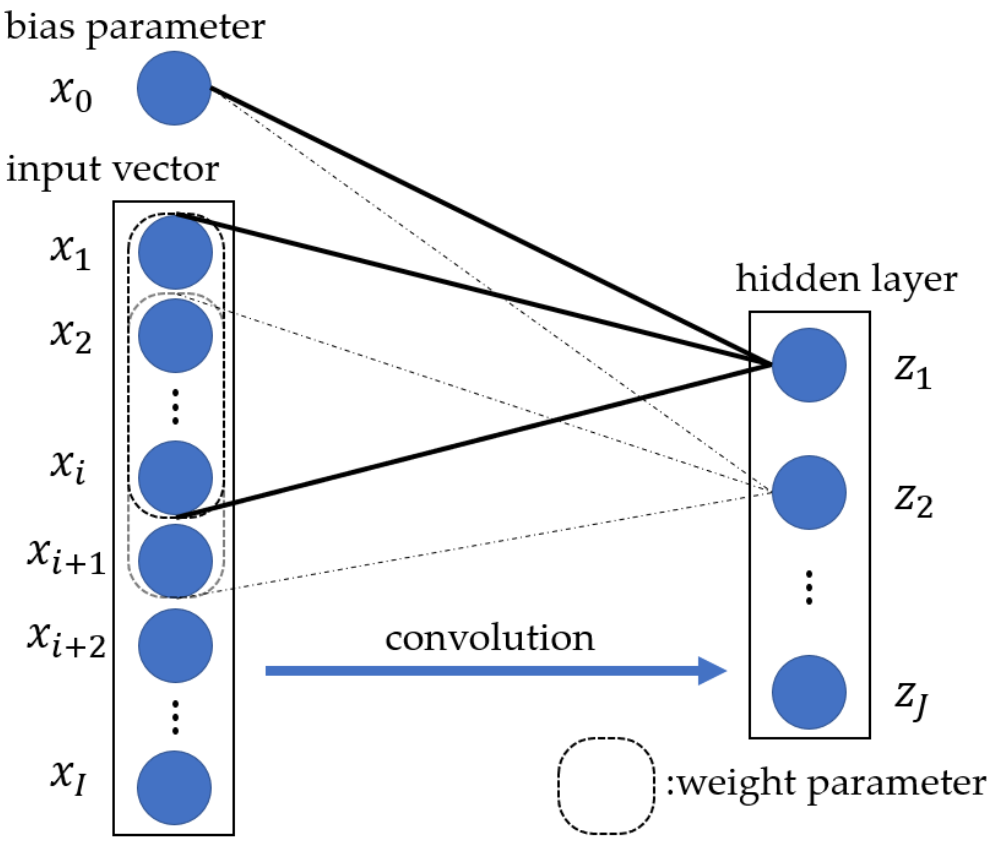


FIGURE 2.3: convolution

to learn the parameters, as in Equation 2.6. The weights and biases are updated as follows, respectively

$$\mathbf{w} \leftarrow \mathbf{w} - \mu \frac{\partial E_n}{\partial \mathbf{w}} \tag{2.17}$$

$$x_0 \leftarrow x_0 - \mu \frac{\partial E_n}{\partial x_0}. \tag{2.18}$$

Chapter 3

Invariant Feature Extraction using Gradient Reversal Layer

3.1 Dataset containing variant information

3.1.1 Shift variant imfomation

For this experiment, I used Fashion Mnist [31] . Fashion Mnist is one of the basic datasets in machine learning for classifying images of clothes and shoes. Fashion Mnist does not originally contain Shift variant information. For this experiment, I will include it by preprocessing. The fact that the classification target is shifted in the image can reduce the classification accuracy. I extract invariant features for this shift information and compare the classification accuracy.

raw data

The raw data is a 28x28 pixel image labeled with 10 classes. An example image of the raw data is shown in Figure 3.1. There are 60,000 training samples and 10,000 test samples.

preprocess

This section describes the pre-processing to include Shift variant information. First, the perimeter of the original image is padded to make a 64 x 64 pixel image. Then, based on a two-dimensional uniform random number, the classification target is shifted so that it does not go outside the image. Figure 3.2 shows an example of an image with these preprocessing steps applied. The center of the classification target is shifted into nine regions, which are labeled as variant information (nine class labels are applied to the nine regions).

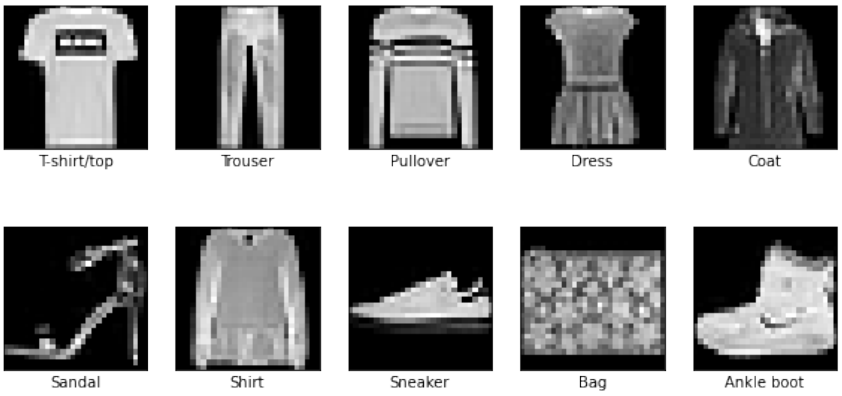


FIGURE 3.1: Raw image of Fashon Mnist

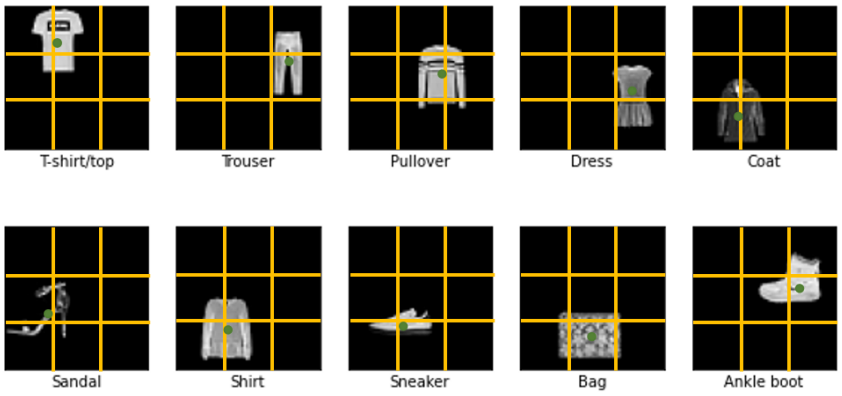


FIGURE 3.2: Shifted image of Fashon Mnist. The yellow line represents the boundary line when dividing into nine regions, and the green point represents the center of the classification target.

3.1.2 Facial expression variant imfomation

In Section 3.1.1, I introduced Shift variant information, a data set that can also be removed by preprocessing. In this experiment, I experimented with a dataset containing facial expression variant information, which is difficult to remove by preprocessing.

FIGURE 3.3: Raw image of the Karolinska Directed Emotional Faces dataset

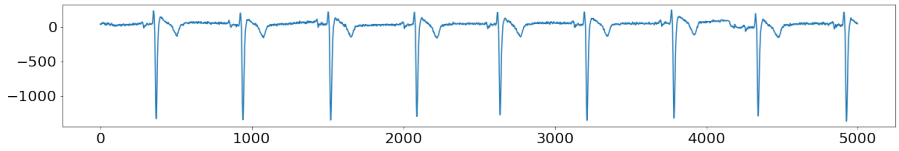
FIGURE 3.4: Preprocessed the Karolinska Directed Emotional Faces dataset

raw data

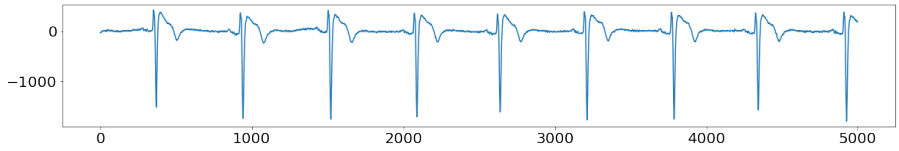
I used the Karolinska Directed Emotional Faces (KDEF) dataset [4] as the dataset containing facial expression variant information. This dataset consists of 70 individuals with 7 facial expressions (afraid, angry, disgusted, happy, neutral, sad, surprised). An example of this data set is shown in Figure 3.3. For simplicity, I use only the front-facing images, although I also have images of the faces from various angles. Therefore, there are 490 face images in the data set. These are prepared as training data, and a dataset of different images of the same person and the same expression is prepared as test data.

preprocess

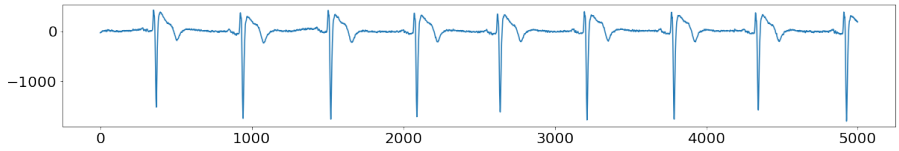
Preprocess this data so that it is easy to use. The face area is cropped and resized to make a 64x64 image. An example of an image with this preprocessing is shown in Figure 3.4.



(A) V_1 lead



(B) V_2 lead



(C) V_2 lead

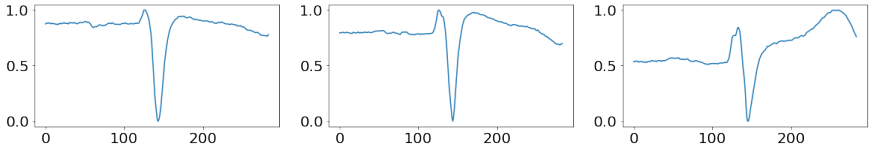
FIGURE 3.5: BrS patient No.0 raw data

3.1.3 Patient variant imfomation

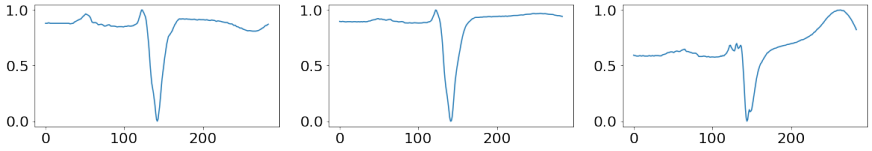
So far, I have introduced a data set for experiments on 2D data. In this section, as an example, I adapt actual clinical data as 1D data. For this experiment, I used electrocardiogram (ECG) data. Specifically, I focused on one of the heart diseases, Brugada syndrome (BrS), which is a hereditary arrhythmogenic disease. BrS is characterized by right ventricular ST-segment elevation and is diagnosed according to the HRS/EHRA/APHRS expert consensus statement [24] when a spontaneous or drug-induced type 1 Brugada ECG The diagnosis is made when a pattern is recorded at least once. However, diagnosis of BrS using a 12-lead ECG remains a challenging task in routine clinical practice. The ECG data were given by a total of 125 subjects, 95 BrS patients, and 30 healthy subjects. Among the BrS patients, 13 patients were labeled as high risk of cardiac arrest and 82 patients were labeled as low risk of cardiac arrest. This data will be validated in two tasks. The data are validated in two tasks: a two-class classification task for healthy subjects and BrS patients (NvB dataset), and a two-class classification task for high and low risk of cardiac arrest (LvH dataset). In both tasks, there is one piece of information that I believe is commonly unnecessary. This information is the difference between patients. I will compare the accuracy of the tasks by performing invariant feature extraction on this information.

raw data

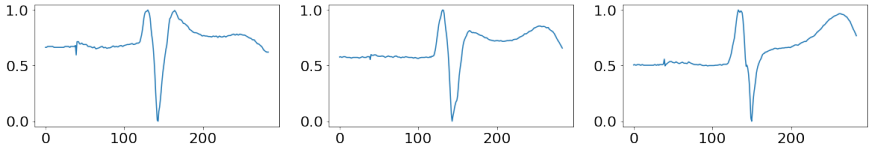
In the raw data, there are 10 seconds of ECG data per patient (sampling frequency is 500 Hz). 12 types of signals are present in the ECG data, of which those related to BrS are leads V_1, V_2 and V_3. The leads V_1, V_2 and V_3 data of one person is shown in Figure 3.5.



(A) patient No.0:(left)V_1 lead (center)V_2 lead (right)V_3 lead



(B) patient No.1:(left)V_1 lead (center)V_2 lead (right)V_3 lead



(C) patient No.3:(left)V_1 lead (center)V_2 lead (right)V_3 lead

FIGURE 3.6: Example of BrS data with cropping and normalization

preprocess

For this study, three types of preprocessing were applied. These are cropping, normalization and Singular value decomposition (SVD) to reveal features. First, let's talk about cropping. The R-peak is calculated by taking the absolute value of the raw data finding the maximum value in the cropping range and then extracting 284 samples as one beat centered on the R-peak. 315 beats for healthy subjects, 124 beats for high-risk BrS patients, and 804 beats for low-risk BrS patients were extracted.

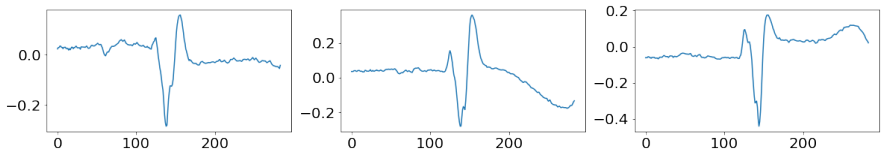
The next step is normalization. In normalization, I used a method to make the number of each beat range from 0 to 1. I used the following formula for normalization

$$\tilde{\mathbf{b}} = \frac{\mathbf{b} - \min(\mathbf{b})}{\max(\mathbf{b}) - \min(\mathbf{b})}. \tag{3.1}$$

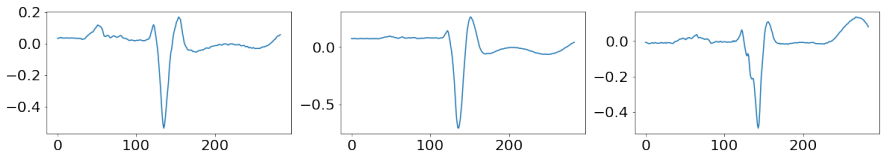
where $\mathbf{b}$ is the beat before normalization, $\tilde{\mathbf{b}}$ is the normalized beat. Examples of data from several people with these preprocesses are shown in Figure 3.6. Finally, SVD is used to reveal the features. This removes the average features in the data. First, the calculation of SVD on the training data matrix X_{tr} is shown in the following equation.

$$U_{tr} D_{tr} V_{tr} = X_{tr} \tag{3.2}$$

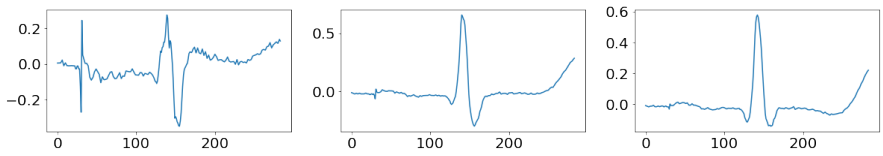
where U_{tr}, V_{tr} is the matrix of eigenvalue vectors of $X_{tr}X_{tr}^T$ and $X_{tr}^T X_{tr}$, respectively. D_{tr} is a matrix with the diagonal components of the non-negative square roots of the eigenvalues of $X_{tr}X_{tr}^T$. The most valuable component of D_{tr} is replaced by zero and reconstructed to reveal the feature. For the test data matrix X_{te}, the V_{tr} of the training



(A) patient No.0:(left)V_1 lead (center)V_2 lead (right)V_3 lead



(B) patient No.1:(left)V_1 lead (center)V_2 lead (right)V_3 lead



(C) patient No.3:(left)V_1 lead (center)V_2 lead (right)V_3 lead

FIGURE 3.7: EExample of BrS data preprocessed by SVD

data is used and calculated as follows.

$$U_{te}V_{tr} = X_{te} \tag{3.3}$$

Here, in U_{te}, the 0 vector is replaced with the column corresponding to the column that was replaced with 0 in the training data and reconstructed. An example of the data of several people with this preprocessing is shown in Figure 3.7.

3.2 Gradient Reversal Layer

The Gradient Reversal Layer (GRL) was proposed in the field of domain adaptation as an element to unlearn domain features[8]. Domain adaptation is a sub-discipline of deep learning in which models learned in one domain is adapted to other domains. Since domain features are not important to improve generalization performance, it makes sense not to train domain features. Ganin et al. proposed a network architecture called domain-adversarial neural network (DANN), in which GRL is used. The DANN consists of three parts (label predictor, domain classifier, and feature extractor), and the feature extractor is split into two parts for estimation. By applying GRL between the feature extractor and the domain classifier, I can achieve an architecture that does not learn domain features. Since GRL has no learning parameters, it can be considered to be a pseudo-function as in the following equation

$$G(\mathbf{z}) = \mathbf{z} \tag{3.4}$$

$$\frac{\partial G}{\partial \mathbf{z}} = -I \tag{3.5}$$

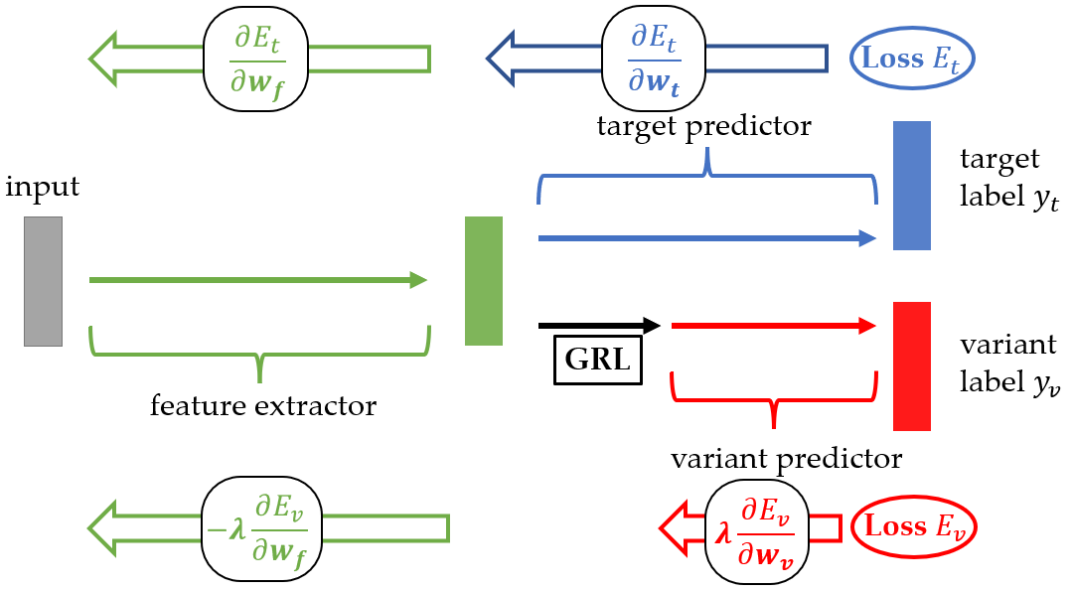


FIGURE 3.8: Invariant Feature Extraction Model using GRL

where I is an identity matrix. I believe that this function of not learning unimportant features can be applied to invariant feature extraction. For this reason, I introduce the model proposed in Section 3.3.

3.3 Invariant Feature Extraction Model using GRL

The architecture of the invariant feature model using GRL is shown in Figure 3.8, where GRL consists of three parts (target predictor, variant predictor, and feature extractor). The weighting parameters of the feature extractor, target predictor, and variant predictor are $\mathbf{w}_f$, $\mathbf{w}_t$ and $\mathbf{w}_v$ respectively. By applying GRL between the feature extractor and the variant predictor, I have achieved an architecture where variant features are not learned. Specifically, GRL, which does nothing during forward, reverses the error during backward. Therefore, the weight parameter derivative $\frac{\partial E_v}{\partial \mathbf{w}_f}$ in the feature extractor of the error E_v calculated from the output of the variant predictor is subtracted. Also, the error E_v calculated from the output of the variant predictor and the error E_t calculated from the output of the target predictor are combined using the hyperparameter λ as follows

$$E = E_t + \lambda E_v. \tag{3.6}$$

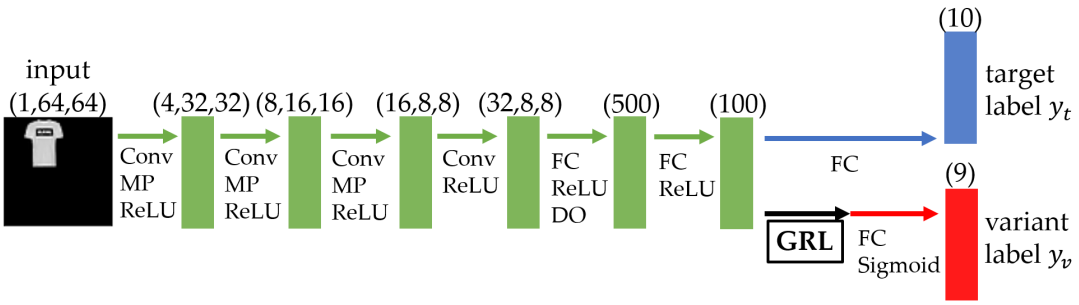


FIGURE 3.9: Invariant Feature Extraction Model using GRL for Fashion Mnist

3.4 Experiments

3.4.1 Shift variant information

Architecture

The network architecture used in this experiment is shown in Figure 3.9. Here, Conv, MP, ReLU, DO, and Sigmoid are convolutional layer, max pooling layer, ReLU function, dropout, and sigmoid function, respectively. the shapes of certain data input are described on the layers, respectively. In this model, three kernel sizes are used to extract features in four convolutional layers and one fully connected layer. This model extracts features in four convolutional layers with kernel size of 3 and one fully connected layer. 100-dimensional feature vectors are obtained from these layers, which are used to estimate the target label from the pure fully connected layer and the variant label from the fully connected layer with GRL.

Loss and Learning Parameter

The error between the estimated label and the ground truth was measured by softmax cross entropy. I also summed them up as in Equation 3.6 and optimized them using Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.001). The batch size was 256, the number of epochs was 300, and the dropout ratio was 0.5.

Results

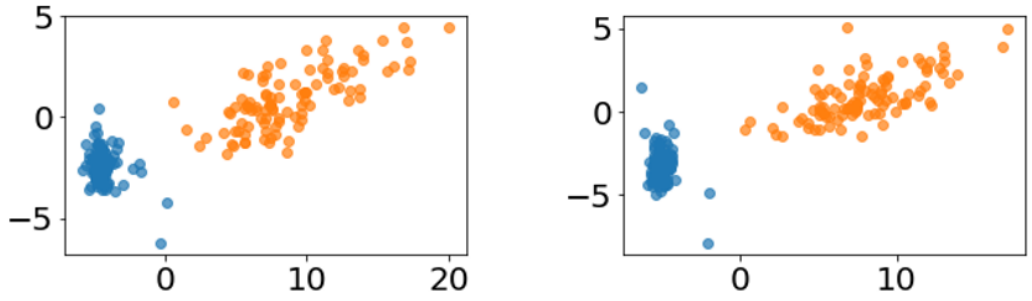
The results of the accuracy of the target label for the test sample are shown in Table 3.1. Here, the range of λ is between 0 and 2.0. A value of λ of 0 is equivalent to no GRL, so this is used as the baseline. I observe the best accuracy when λ is 2.0, which is a 1.01% improvement in accuracy.

TABLE 3.1: Target accuracy scores of the shift invariant experiment using GRL model

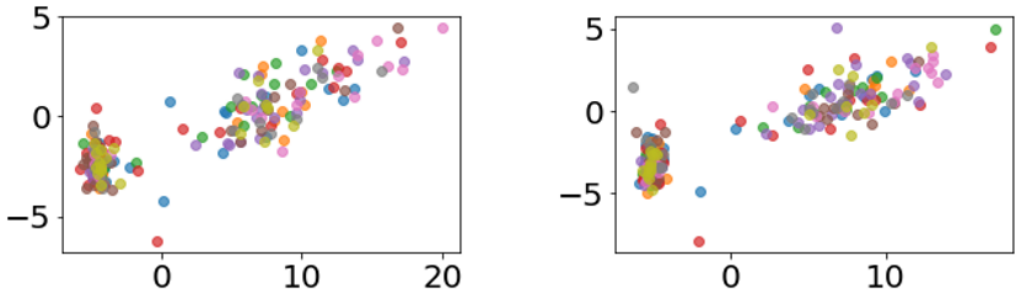
λ	target accuracy[%]
0.0	82.52
0.5	82.80 (+0.28)
1.0	83.51 (+0.99)
1.5	83.02 (+0.50)
2.0	**83.53 (+1.01)**

TABLE 3.2: The logistic regression score in the shift invariant experiment using GRL model

λ	score
0.0	0.1354
2.0	**0.1252**



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

FIGURE 3.10: PCA result in the shift invariant experiment using GRL model: (left)$\lambda = 0$ (right)$\lambda = 2.0$

Feature Analysis

I will analyze the trend in the features extracted by GRL. For this purpose, I compare the baseline with the highest accuracy of λ at 2.0. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure 3.9 with 100 written on top) with the variant label The 100-dimensional vector is reduced to 10 dimensions

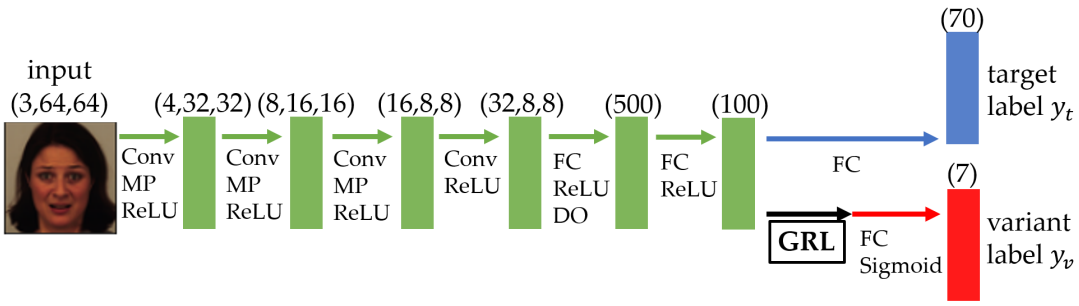


FIGURE 3.11: Invariant Feature Extraction Model using GRL for KDEF dataset

by principal component analysis (PCA) [11]. After the 100-dimensional vector was dimensionally reduced to 10 by PCA, the logistic regression scores are shown in Table 3.2. When λ is 2.0, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 2.0 were visualized in two dimensions by PCA respectively. The training data is used for visualization, two of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 3.10. I could see that the classes were more cohesive when λ was 2.0 compared to the baseline.

3.4.2 Facial expression variant information

Architecture

The network architecture used in this experiment is shown in Figure 3.11. In this model, three kernel sizes are used to extract features in four convolutional layers and one fully connected layer. 100-dimensional feature vectors are obtained from these layers, which are used to estimate the target label from the pure fully connected layer and the variant label from the fully connected layer with GRL.

Loss and Learning Parameter

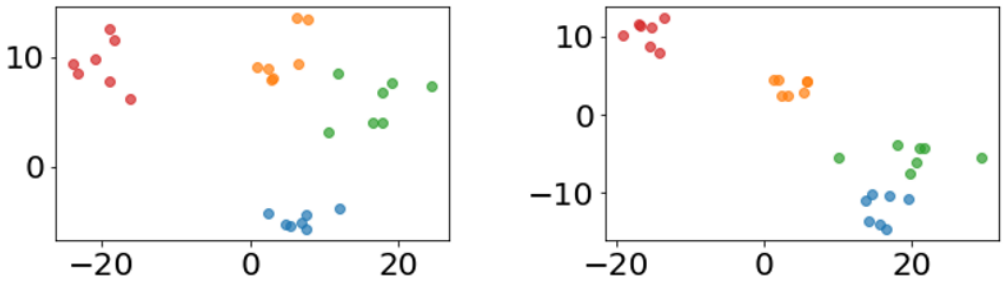
The error between the estimated label and the ground truth was measured by softmax cross entropy. I also summed them up as in Equation 3.6 and optimized them using Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.001). The batch size was 32, the number of epochs was 500, and the dropout ratio was 0.5.

Results

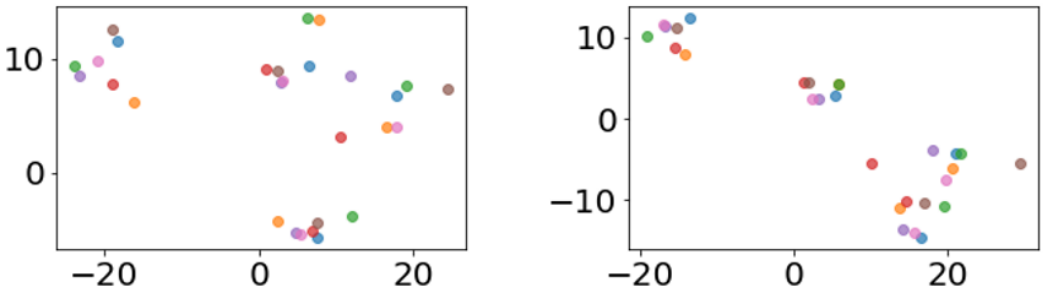
The results of the accuracy of the target label for the test sample are shown in Table 3.3. Here, the range of λ is between 0 and 2.0. A value of λ of 0 is equivalent to no GRL, so this is used as the baseline. I observe the best accuracy when λ is 0.5, which is a 0.61% improvement in accuracy.

TABLE 3.3: Target accuracy scores of the facial expression invariant experiment using GRL model

λ	target accuracy[%]
0.0	95.51
0.5	**96.12 (+0.61)**
1.0	95.10 (-0.01)
1.5	94.08 (-1.43)
2.0	94.48 (-1.03)



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

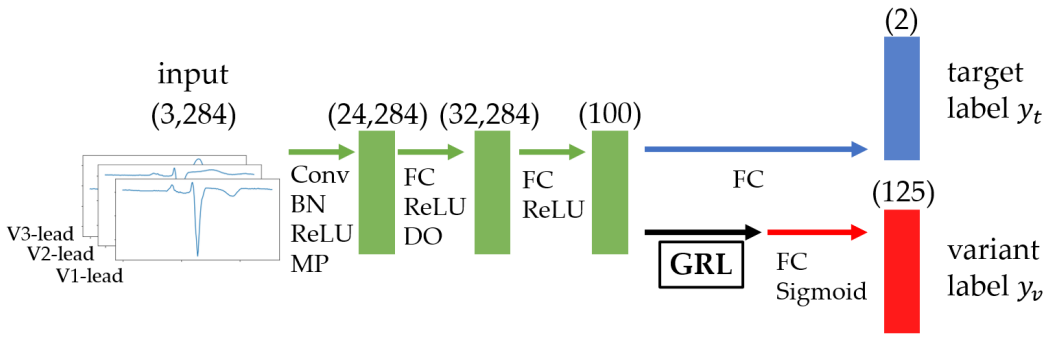
FIGURE 3.12: PCA result in the facial expression invariant experiment using GRL model: (left)$\lambda = 0$ (right)$\lambda = 0.5$

Feature Analysis

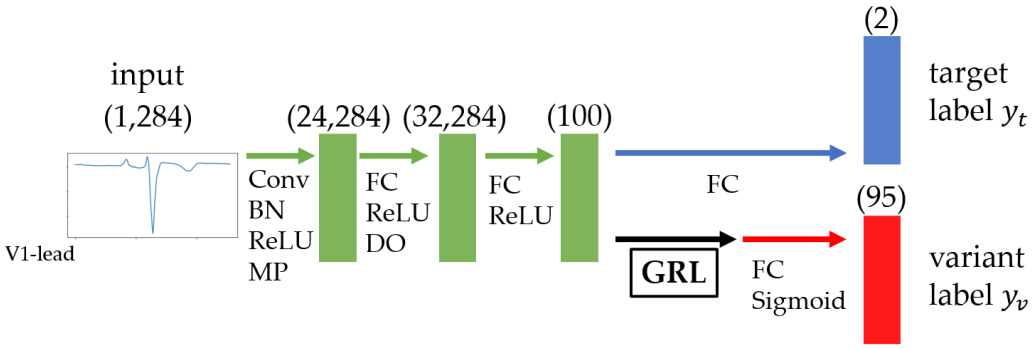
TABLE 3.4: The logistic regression score in the facial expression invariant experiment using GRL model

λ	score
0.0	0.1775
0.5	**0.1448**

I will analyze the feature vectors in the same way as in Section 3.4.1. For this purpose, I compare the baseline with the highest accuracy of λ at 0.5. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure



(A) Invariant Feature Extraction Model using GRL for NvB dataset



(B) Invariant Feature Extraction Model using GRL for LvH dataset

FIGURE 3.13: Invariant Feature Extraction Model using GRL for BrS dataset

3.11 with 100 written on top) with the variant label The 100-dimensional vector is reduced to 10 dimensions by PCA. After the 100-dimensional vector was dimensionally reduced to 10 by PCA, the logistic regression scores are shown in Table 3.4. When λ is 0.5, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 0.5 were visualized in two dimensions by PCA respectively. The training data is used for visualization, 4 of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 3.12. I could see that the classes were more cohesive when λ was 0.5 compared to the baseline.

3.4.3 Patient variant information

Architecture

The network architecture used in this experiment is shown in Figure 3.13. In this model, 21 kernel sizes are used to extract features in four convolutional layers and one fully connected layer. 100-dimensional feature vectors are obtained from these layers, which are used to estimate the target label from the pure fully connected layer and the variant label from the fully connected layer with GRL. For the NvB dataset,

I used three channels of leads V_1, V_2 and V_3 as input data, but for the LvH dataset, I used only leads V_1 as input data. The reason for this is explained in the results section of Section 3.4.3.

Loss and Learning Parameter

The error between the estimated label and the ground truth was measured by softmax cross entropy. I also summed them up as in Equation 3.6. For these errors, I optimize the NvB dataset with Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.001). The batch size was 30, the number of epochs was 300, and the dropout ratio was 0.1. For the LvH dataset, I optimize with Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.01). The batch size was 30, the number of epochs was 300, and the dropout ratio was 0.5. In both datasets, I learn to avoid an imbalance in the target label within the batch size.

Results

TABLE 3.5: Target accuracy scores of the patient invariant experiment using GRL model

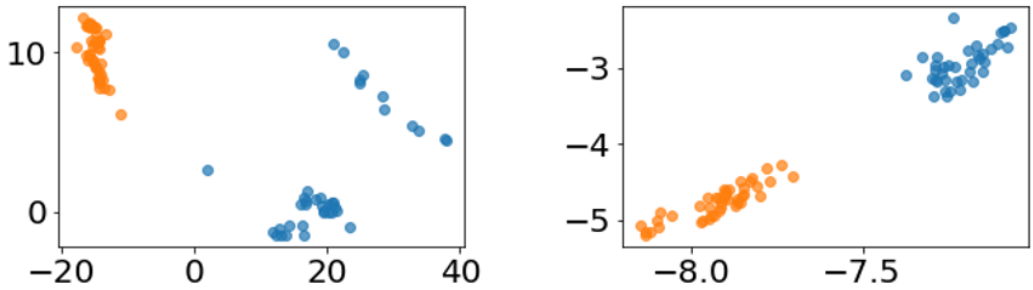
λ	NvB dataset target accuracy[%]	LvH dataset target accuracy[%]
0.0	92.97	62.17
1.0	93.10 (+0.13)	63.00 (+0.83)
2.0	93.52 (+0.95)	64.68 (+2.51)
3.0	94.84 (+1.87)	**66.08 (+3.91)**
4.0	**96.86 (+3.89)**	65.63 (+3.46)

The results of the accuracy of the target label for the test sample are shown in Table 3.5. Here, the range of λ is between 0 and 4.0. A value of λ of 0 is equivalent to no GRL, so this is used as the baseline. I was able to observe hyperparameters λ that exceeded the baseline in both datasets. In this experiment, I was not able to include the results for the three channels of leads V_1, V_2 and V_3 for the LvH dataset. This is because in the baseline experiment ($\lambda = 0$), the test accuracy was 51.32%, and I could not train it well.

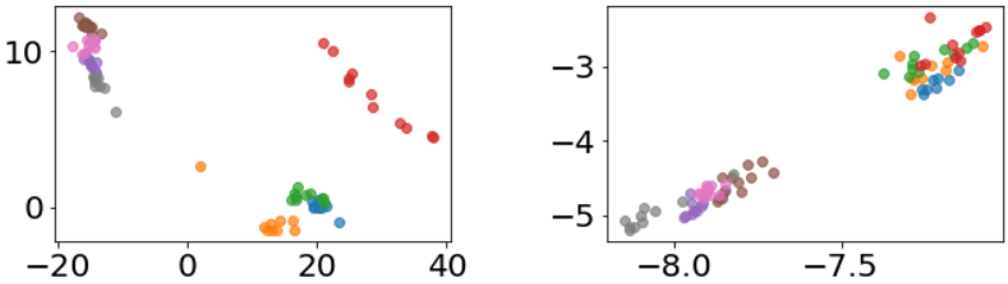
Feature Analysis NvB dataset

TABLE 3.6: The logistic regression score in the patient invariant experiment (NvB dataset) using GRL model

λ	score
0.0	0.9975
4.0	**0.9578**



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

FIGURE 3.14: PCA result in the patient invariant experiment (NvB dataset) using GRL model: (left)$\lambda = 0$ (right)$\lambda = 4.0$

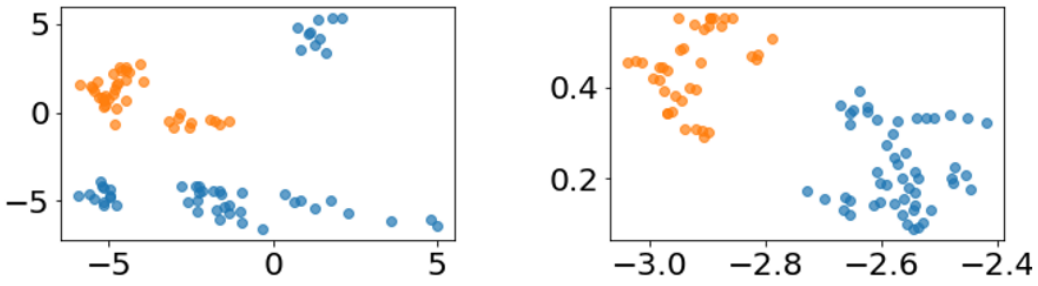
I will analyze the feature vectors in the same way as in Section 3.4.1. For this purpose, I compare the baseline with the highest accuracy of λ at 4.0. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure 3.13 with 100 written on top) with the variant label The 100-dimensional vector is reduced to 10 dimensions by PCA. After the 100-dimensional vector was dimensionally reduced to 10 by PCA, the logistic regression scores are shown in Table 3.6. When λ is 4.0, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 4.0 were visualized in two dimensions by PCA respectively. The training data is used for visualization, two of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 3.14. I could see that the classes were more cohesive when λ was 4.0 compared to the baseline. In addition, when I look at the data colored by the variant label, I can see that there are data that are clustered by the variant label. If I look at this, I can see an example of how data with the same target label, which were separated by the baseline, can come together regardless of the variant label.

Feature Analysis LvH dataset

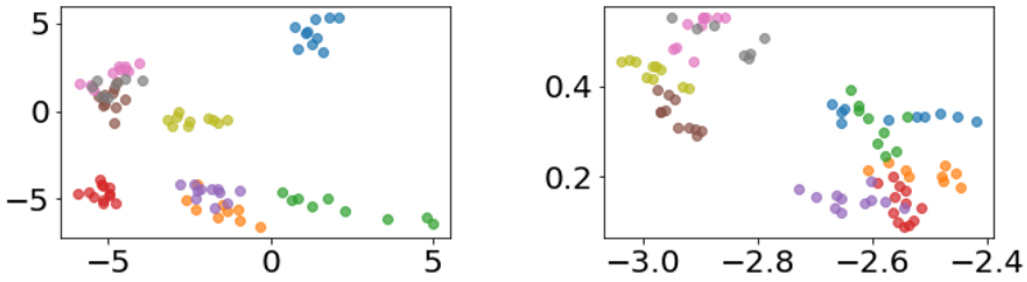
I will analyze the feature vectors in the same way as in Section 3.4.1. For this purpose, I compare the baseline with the highest accuracy of λ at 3.0. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure 3.13 with

TABLE 3.7: The logistic regression score in the patient invariant experiment (LvH dataset) using GRL model

λ	score
0.0	0.9975
3.0	**0.9578**



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

FIGURE 3.15: PCA result in the patient invariant experiment (LvH dataset) using GRL model: (left)$\lambda = 0$ (right)$\lambda = 3.0$

100 written on top) with the variant label The 100-dimensional vector is reduced to 4 dimensions by PCA. After the 100-dimensional vector was dimensionally reduced to 4 by PCA, the logistic regression scores are shown in Table 3.7. When λ is 3.0, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 3.0 were visualized in two dimensions by PCA respectively. The training data is used for visualization, two of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 3.15. I could see that the classes were more cohesive when λ was 3.0 compared to the baseline. In addition, as in the case of the NvB dataset, we were able to confirm that the data was clustered by variant label, and we were able to observe data that gathered regardless of the variant label due to the effect of GRL.

Chapter 4

Invariant Feature Extraction using Siamese Network

4.1 Siamese Neural Network

Siamese Neural Network [3] is one of the leading methods for deep metric learning. In fact, it has been used in face verification [6] and Signal Classification [28] and its effectiveness has been proven. Specifically, as shown in Figure 4.1, two samples are paired and used as input to a Neural Network with shared weights, and the similarity of the extracted features is measured. After that, I learn to make the similarities closer if the categories of the pair are the same and to make the similarities apart if the categories are different. Loss for this learning is explained in Section 4.1.1.

4.1.1 Contrastive Loss Function

Contrastive Loss works in such a way that it chooses to make the similarity smaller or larger depending on the input pair. First, I will use the L1 norm to describe the similarity, although many similarities can be used. In this case, the similarity is given by the following equation

$$E_{L1} = ||Z_1 - Z_2|| \tag{4.1}$$

For this E_{L1}, when Z_1 and Z_2 are in the same category (genuine pair), $E_{genuine}$, and when Z_1 and Z_2 are in different categories (imposter pair), $E_{impostor}$. If the label of whether the categories are the same or not is T (1 for same, 0 for different), the Contrastive Loss Function E_C is

$$E_C = TE^2_{genuine} + (1-T)\max(margin - E_{impostor}, 0)^2 \tag{4.2}$$

where *margin* is a parameter that controls how far apart the different categories are, and $\max(\cdot)$ is a function that returns the maximum value.

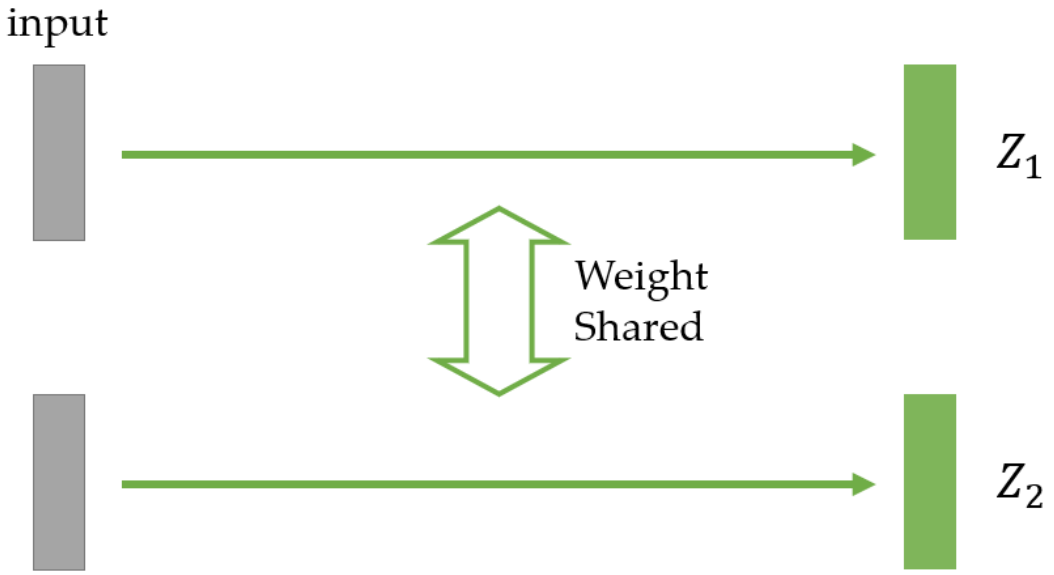


FIGURE 4.1: Siamese Neural Network Architecure

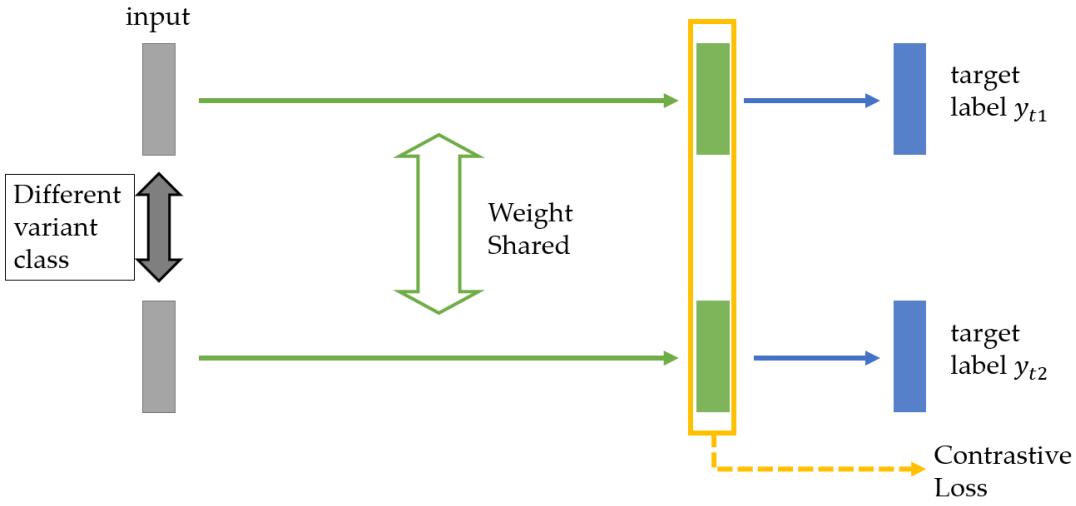


FIGURE 4.2: Invariant Feature Extraction Model using Siamese Neural Network

4.2 Invariant Feature Extraction Model using Siamese Neural Network

Our proposed network architecture for invariant feature extraction, made about the Siamese Neural Network, is shown in Figure 4.2. Since this model allows for supervised learning, the estimated labels of each network are also added to the learning error. Therefore, the final expression for the error is

$$E = E_{t1} + E_{t2} + \lambda E_C \tag{4.3}$$

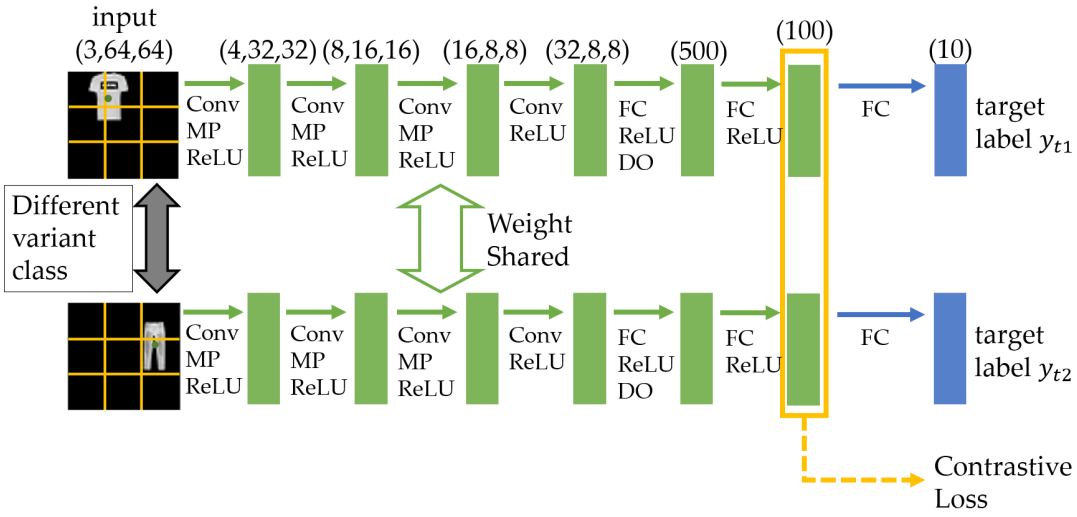


FIGURE 4.3: Invariant Feature Extraction Model using Siamese Neural Network for Fashion Mnist

where E_{t1}, E_{t2} are the errors calculated from the first and second target label estimates and ground truth, respectively, and λ controls the effect of Contrastive Loss Hyperparameters.

4.3 Experiments

4.3.1 Shift variant information

Architecture

The network architecture used in this experiment is shown in Figure 4.3. In this model, I use the same structure as in Section 3.4.1 for feature extraction and target label estimation. The characteristic Contrastive Loss of the Siamese Neural Network is computed on the feature vectors surrounded by the yellow boxes. The input always takes different shift classes (different variant classes).

Loss and Learning Parameter

The error between the estimated label and the ground truth was measured by softmax cross entropy. In Contrastive Loss, a genuine pair is the same fashion class, and an imposter pair is a different fashion class. I also summed them up as in Equation 4.3 and optimized them using Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.001). The batch size was 256, the number of epochs was 300, and the dropout ratio was 0.5.

Results

The results of the accuracy of the target label for the test sample are shown in Table 4.1. Target accuracy is the accuracy of the network output values target label y_{t1}.

TABLE 4.1: Target accuracy scores of the shift invariant experiment using Siamese model

λ	target accuracy [%]
0.0	82.74
0.0005	82.55 (-0.19)
0.001	83.45 (+0.71)
0.0015	**83.57 (+0.83)**
0.002	82.72 (-0.02)

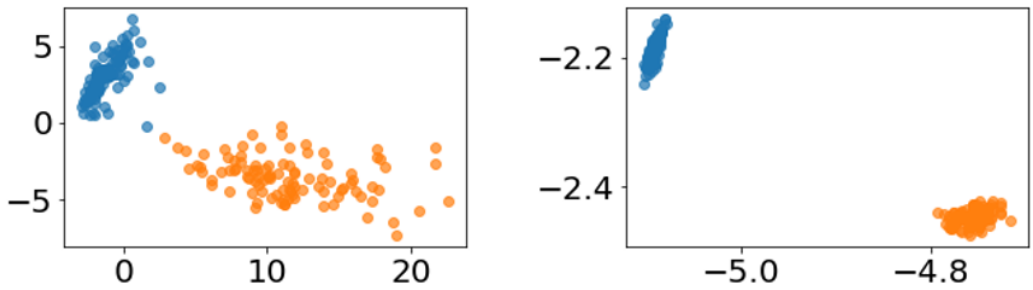
Here, the range of λ is between 0 and 0.002. When the value of λ is 0, there is no effect of Contrastive Loss, so this is used as the baseline. I observe the best accuracy when λ is 0.0015, which is a 0.83% improvement in accuracy.

Feature Analysis

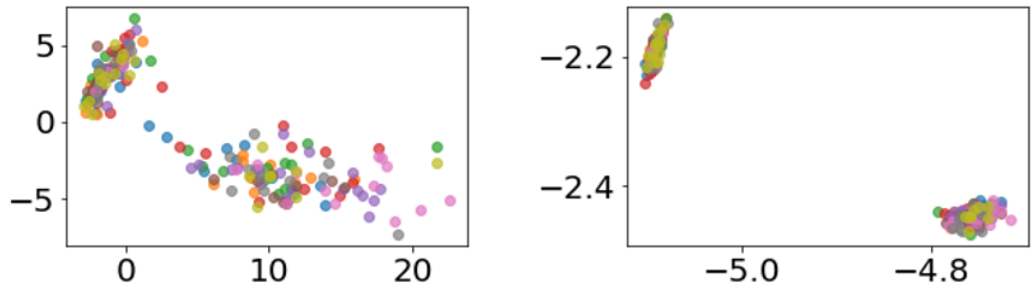
TABLE 4.2: The logistic regression score in the shift invariant experiment using Siamese model

λ	score
0.0	0.1320
0.0015	**0.1230**

I will analyze the feature vectors in the same way as in Section 3.4.1. For this purpose, I compare the baseline with the highest accuracy of λ at 0.0015. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure 4.3 with 100 written on top) with the variant label The 100-dimensional vector is reduced to 10 dimensions by PCA. After the 100-dimensional vector was dimensionally reduced to 10 by PCA, the logistic regression scores are shown in Table 4.2. When λ is 0.0015, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 0.0015 were visualized in two dimensions by PCA respectively. The training data is used for visualization, two of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 4.4. I could see that the classes were more cohesive when λ was 0.0015 compared to the baseline. In addition, it can be seen that the target classes are more strongly grouped than in Section 3.4.1. This may be due to the effect of Contrastive Loss in separating the different classes.



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

FIGURE 4.4: PCA result in the shift invariant experiment using Siamese modell: (left)$\lambda = 0$ (right)$\lambda = 0.0015$

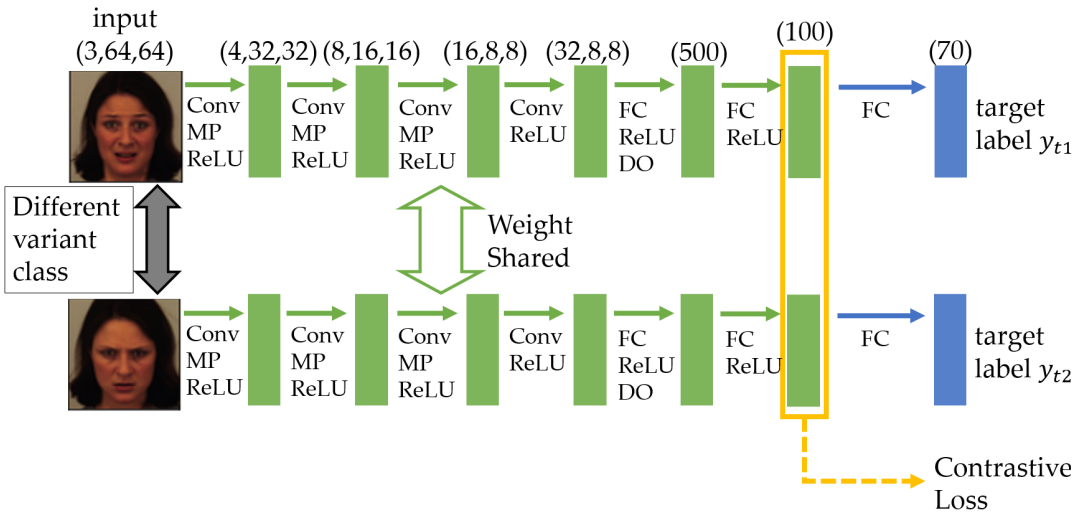


FIGURE 4.5: Invariant Feature Extraction Model using Siamese Neural Network for KDEF dataset

4.3.2 Facial expression variant information

Architecture

The network architecture used in this experiment is shown in Figure 4.5. In this model, I use the same structure as in Section 3.4.2 for feature extraction and target label estimation. The characteristic Contrastive Loss of the Siamese Neural Network

is computed on the feature vectors surrounded by the yellow boxes. The input always takes different facial expressions (different variant classes).

Loss and Learning Parameter

The error between the estimated label and the ground truth was measured by softmax cross entropy. In Contrastive Loss, a genuine pair is the same person, and an imposter pair is a different person. I also summed them up as in Equation 4.3 and optimized them using Momentum SGD (momentum: 0.9, learning rate: 0.01, weight decay: 0.001). The batch size was 32, the number of epochs was 500, and the dropout ratio was 0.5.

Results

TABLE 4.3: Target accuracy scores of the facial expression invariant experiment using Siamese model

λ	target accuracy [%]
0.0	95.51
0.0005	96.32 (+0.81)
0.001	95.91 (+0.40)
0.0015	**96.53 (+1.02)**
0.002	96.53 (+1.02)

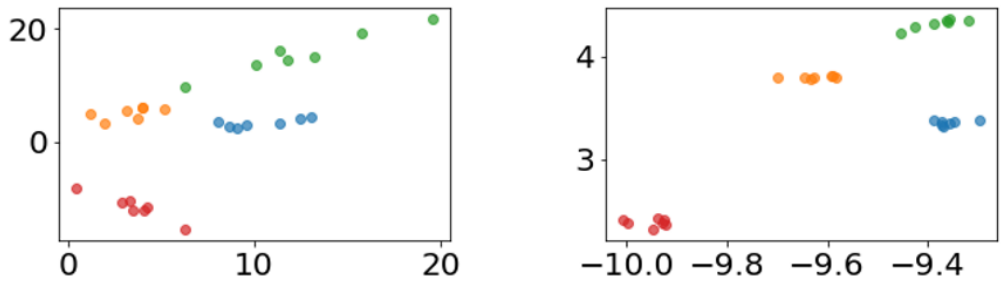
The results of the accuracy of the target label for the test sample are shown in Table 4.3. Target accuracy is the accuracy of the network output values target label y_{t1}. Here, the range of λ is between 0 and 0.002. When the value of λ is 0, there is no effect of Contrastive Loss, so this is used as the baseline. I observe the best accuracy when λ is 0.0015, which is a 1.02% improvement in accuracy.

Feature Analysis

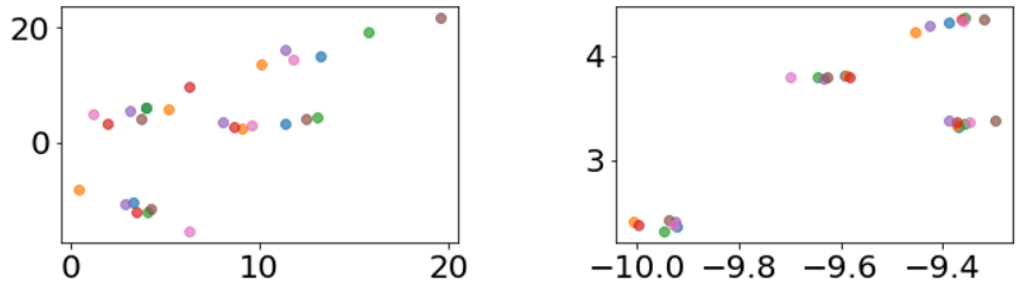
TABLE 4.4: The logistic regression score in the facial expression invariant experiment using Siamese model

λ	score
0.0	0.1734
0.0015	**0.1448**

I will analyze the feature vectors in the same way as in Section 3.4.1. For this purpose, I compare the baseline with the highest accuracy of λ at 0.0015. First, I compare the extracted 100-dimensional feature vectors (the green rectangles in Figure 4.5 with 100 written on top) with the variant label The 100-dimensional vector is reduced to



(A) Colored visualization for each target label



(B) Colored visualization for each variant label

FIGURE 4.6: PCA result in the facial expression invariant experiment using Siamese modell: (left)$\lambda = 0$ (right)$\lambda = 0.0015$

10 dimension by PCA. After the 100-dimensional vector was dimensionally reduced to 10 by PCA, the logistic regression scores are shown in Table 4.4. When λ is 0.0015, the score is lower than the baseline. This may be due to the removal of variant information from the extracted feature vector. Next, the feature vectors for the baseline and when λ was 0.0015 were visualized in two dimensions by PCA respectively. The training data is used for visualization, and two of the target classes are selected and some of the data are used. The results of PCA on a part of train data are shown in Figure 4.6. I could see that the classes were more cohesive when λ was 0.0015 compared to the baseline. Also, as in Shift variant information (Section 4.3.1), the effect of Contrastive Loss shows that each target class is more strongly grouped together than in Section 3.4.2.

Chapter 5

Conclusion

I proposed two methods for end-to-end CNN models for invariant feature extraction using training data containing variant information. The GRL-based method was evaluated on three tasks, and the Siamese model was evaluated on two tasks to demonstrate the improvement in accuracy. I also confirmed that the invariance of the obtained features was increased quantitatively by logistic regression scores and qualitatively by visualization. The proposed method is expected to be effective in cases where data preprocessing is difficult. In future work, I am planning to use the Siamese model for feature extraction on the BrS data set.

Bibliography

[1] Sheikh Bilal Ahmed et al. "On the frontiers of pose invariant face recognition: a review". In: *Artificial Intelligence Review* 53.4 (2020), pp. 2571–2634.

[2] Christopher M Bishop. "Pattern recognition". In: *Machine learning* 128.9 (2006).

[3] Jane Bromley et al. "Signature verification using a "siamese" time delay neural network". In: *International Journal of Pattern Recognition and Artificial Intelligence* 7.04 (1993), pp. 669–688.

[4] Manuel G Calvo and Daniel Lundqvist. "Facial expressions of emotion (KDEF): Identification under different display-duration conditions". In: *Behavior research methods* 40.1 (2008), pp. 109–115.

[5] Minho Choi et al. "Wearable device-based system to monitor a driver's stress, fatigue, and drowsiness". In: *IEEE Transactions on Instrumentation and Measurement* 67.3 (2017), pp. 634–645.

[6] Sumit Chopra, Raia Hadsell, and Yann 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.

[7] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

[8] Yaroslav Ganin et al. "Domain-adversarial training of neural networks". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2096–2030.

[9] Matt W Gardner and SR Dorling. "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences". In: *Atmospheric environment* 32.14-15 (1998), pp. 2627–2636.

[10] Francois Goudail et al. "Face recognition system using local autocorrelations and multiscale integration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.10 (1996), pp. 1024–1028.

[11] Harold Hotelling. "Analysis of a complex of statistical variables into principal components." In: *Journal of educational psychology* 24.6 (1933), p. 417.

[12] Kazuhiro Hotta, Takio Kurita, and Taketoshi Mishima. "Scale invariant face detection method using higher-order local autocorrelation features extracted from log-polar image". In: *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE. 1998, pp. 70–75.

[13] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. "Discriminative deep metric learning for face verification in the wild". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1875–1882.

[14] Shweta H Jambukia, Vipul K Dabhi, and Harshadkumar B Prajapati. "Classification of ECG signals using machine learning techniques: A survey". In: *2015 International Conference on Advances in Computer Engineering and Applications*. IEEE. 2015, pp. 714–721.

[15] Jinlong Ji et al. "A deep multi-task learning approach for ECG data analysis". In: *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*. IEEE. 2018, pp. 124–127.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[17] Takio Kurita, Kazuhiro Hotta, and Taketoshi Mishima. "Scale and rotation invariant recognition method using higher-order local autocorrelation features of log-polar image". In: *Asian Conference on Computer Vision*. Springer. 1998, pp. 89–96.

[18] Takio Kurita, Nobuyuki Otsu, and Tomomasa Sato. "A face recognition method using higher order local autocorrelation and multivariate analysis". In: *International Conference on Pattern Recognition*. IEEE Computer Society Press. 1992, pp. 213–213.

[19] Tianyu Liu et al. "Few-shot learning for cardiac arrhythmia detection based on electrocardiogram data from wearable devices". In: *Digital Signal Processing* 116 (2021), p. 103094.

[20] Yuyan Liu et al. "Rotation-invariant siamese network for low-altitude remote-sensing image registration". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13 (2020), pp. 5746–5758.

[21] G Lowe. "SIFT-the scale invariant feature transform". In: *Int. J* 2 (2004), pp. 91–110.

[22] Warren S McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[23] Nobuyuki Otsu and Takio Kurita. "A New Scheme for Practical Flexible and Intelligent Vision Systems." In: *Proc. IAPR Workshop on Computer Vision*. 1988, pp. 431–435.

[24] Silvia G Priori et al. "HRS/EHRA/APHRS expert consensus statement on the diagnosis and management of patients with inherited primary arrhythmia syndromes: document endorsed by HRS, EHRA, and APHRS in May 2013 and by ACCF, AHA, PACES, and AEPC in June 2013." In: *Heart Rhythm* 10.12 (2013), pp. 1932–1963.

[25] Frank Rosenblatt. *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Tech. rep. Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[26] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[27] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536.

[28] Guangqing Shao, Yushi Chen, and Yinsheng Wei. "Convolutional neural network-based radar jamming signal classification with sufficient and limited samples". In: *IEEE Access* 8 (2020), pp. 80588–80598.

[29] Masaru Tanaka, Takio Kurita, and Shinji Umeyama. "Image understanding via representation of the projected motion group". In: *Pattern recognition letters* 15.10 (1994), pp. 993–1001.

[30] Bernard Widrow and Marcian E Hoff. *Adaptive switching circuits*. Tech. rep. Stanford Univ Ca Stanford Electronics Labs, 1960.

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).