Doctoral Thesis

Enhancing Recognition and Improved Processing Speed for Isolated Sign Language Recognition
（手話単語認識における認識率の向上と処理速度の改善）

Noriaki Hori

Feburary, 2024
The Graduate School of Information Science and Technology
Hokkaido University

Abstract

Research on sign language recognition can be divided into two categories: recognition of consecutive signs, like simultaneous sign language interpretation for news programs, and word recognition, called isolated sign recognition. This study focuses on enhancing recognition and improving processing speed for isolated sign language. Examples of concrete applications of sign language word recognition include conversational sign language recognition and support for the deaf and hard of hearing and sign language learners. For example, we believe that signers can be photographed and automatically converted into words or speech sounds to facilitate smooth communication between people who do not use sign language and those who understand sign language. In recent years, various researchers have proposed models for sign language recognition, and the research on sign language recognition has made significant progress but is not yet complete: On a dataset of about 230 words, the recognition rate is above 98% and approaching human recognition levels. Recently, GPUs have become more powerful, enabling real-time processing. In this paper, based on the SAM-SLR (Skeleton Aware Multi-model Sign Language Recognition) model, we propose a method to further enhance the recognition rate by reusing the estimated information for each epoch output during learning, utilizing joint coordinates (Joint) and bone coordinates (Bone) with posture estimation information. Additionally, we present a methodology for re-evaluating when the difference between Top-1 and Top-2 recognition evaluation values is low. Furthermore, we introduce an approach to improve processing speed while preserving the recognition rate and evaluation value, enhancing the practical utility of sign language recognition.

The SAM-SLR, based on our research, secured victory in the 2021 competition utilizing the AUTSL (Ankara University Turkish Sign Language) dataset and achieved an impressive recognition rate of 97.64%. We have conducted research to improve the recognition rate further, as some researchers have reported that the recognition rate of the dataset AUTSL could be increased to just under 99%. The second proposal focuses on the low recognition rate when the difference between the recognition results' Top-1 and Top-2 evaluation values is slight. It is a method that aims for even higher recognition without compromising the high recognition rate of the existing model, re-evaluating the results using a new technique not employed in the existing model. With the recognition rate of the AUTSL dataset approaching human levels to achieve practical

usability, the next challenge revolves around processing speed. To enhance processing speed, the main proposals include maintaining the recognition rate, addressing the parts of the dataset that do not work on current GPUs, seamlessly connecting four independent modalities from one video, and reducing the recognition response time.

The paper consists of six chapters. Chapter 1 describes the background and objective of the research. In particular, it discusses the challenges of improving the recognition rate and processing speed of the SAM-SLR model.

In Chapter 2, we describe related studies on the isolated sign language recognition model. In particular, the model SAM-SLR, on which this study is based, is described in detail.

In Chapter 3, we describe a method to improve the recognition rate based on the SAM-SLR model. Focusing on the high recognition rate of Joint and Bone in the SAM-SLR model's recognition process, we propose a method to reuse the estimated results of Joint and Bone for each epoch during training. Initially, we aggregate the Top-1 evaluation values for each class up to an arbitrary epoch and adopt the class with the highest total Top-1 evaluation value. Additionally, we propose an algorithm that sequentially searches from that arbitrary epoch to epoch 0, taking the evaluation result when the Top-1 corresponds to that class as the result of the arbitrary epoch evaluation. This method is applied to the SAM-SLR recognition methods Joint and Bone, respectively, to improve the recognition rate. Furthermore, the recognition rate can be improved by applying the Joint and Bone results to these higher-level methods' Multi-stream and SAM-SLR models.

In Chapter 4, we discuss further methods to improve the recognition rate based on the SAM-SLR model. The decision to implement re-evaluation is based on conditions such as when the difference between the Top-1 and Top-2 evaluation values from the SAM-SLR recognition results is slight and when a sign, such as one hand, is within the face area. The evaluation method for re-evaluation is to create a triangular mesh based on the positions of facial parts such as eyes, nose, mouth, chin, and cheeks, and to capture the index finger position in the triangular mesh shape so that the positions can be compared relatively even for different facial shapes. This method more accurately captures sign language recognition that touches the mouth, nose, forehead and is evaluated by comparing the positions of all Train and Test data in each class. The recognition rate improved from 97.94% to 98.24%.

In Chapter 5, we describe a method to improve the processing speed to maintain the recognition rate in the SAM-SLR model and reduce the recognition response time. First, since the Optical Flow used in the RGB-Flow modality of the SAM-SLR model does not work on current GPUs, the code was modified to obtain results equivalent to the original method, and four independent modalities were seamlessly connected from one video for recognition processing. Next, to reduce the recognition response time, we reduced processing, parallelized, changed from Python to C++, and utilized internal memory while maintaining the recognition rate. Consequently, the difference between the original and proposed methods of modified code in optical flow processing was almost the same visually for the optical image. There were some numerical differences, but they were within a minor range. Additionally, for the MMPose posture estimation process, the original and proposed methods produced almost the same results. The recognition evaluation values were almost the same, and the recognition rate was also the same. The average recognition response time was reduced from about 4.4 seconds to 0.72 seconds, about six times faster than the serial processing and the proposed method.

In Chapter 6, we discuss the improvement of recognition rate and processing speed for isolated sign language recognition and conclude the paper. This chapter summarizes the study's findings on sign language recognition and processing speed improvement and discusses future developments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Isolated sign language

The inception of computer-based sign language recognition began with image recognition, such as fingerspelling, and has evolved to encompass sign language word recognition, along with sign language video recognition, including conversational sign language. Learning sign language video recognition necessitates a high-performance CPU or GPU for image processing, with simple sign language recognition becoming feasible on GPUs since around 2020. Sign language video recognition can be categorized into two main types: continuous sign language, observed in TV news programs, and "word-based sign language recognition," known as isolated sign language. Isolated sign language consists of word-by-word sign language stored in a video, utilizing the video as a frame image for learning and recognition through an image recognition model. Research on isolated sign language recognition also plays an important role in advancing conversational sign language recognition, which is anticipated to become mainstream in the future.

1.2 Research Background

In this study, our focus is on enhancing both the recognition rate and real-time processing for isolated sign language recognition. Specific applications of isolated sign language recognition include supporting the deaf and aiding sign-language learners. Sign language serves as a means of communication for people who are deaf or hard of hearing; however, it can be a challenging tool for those unfamiliar. Therefore, we envision facilitating smooth communication for individuals who do not use sign language by capturing sign language through photographs and automatically converting it into subtitles and voice for better understanding. Over the past few years, various researchers have proposed models for sign language recognition, and while significant progress has been made, the research is not yet complete. Some sign language datasets are approaching human perception levels, surpassing 98%. With recent improvements in GPU performance, real-time processing has become achievable. Additionally, advancements in smartphone capabilities have led to the emergence of proposals for applications that recognize sign language on smartphones. In this study, we establish the following criteria to facilitate the practical implementation of real-time processing for isolated sign language:

・Enhance the recognition rate of isolated sign language to be equal to or close to the human recognition rate compared to conventional methods.

・Ensure that the response time for sign language recognition following the playback of a video is within 1 second.

However, it is essential to note that in this paper, the term "real-time" is defined as increasing the recognition processing speed as much as possible so that the sign language recognition response time does not appear long. This implies a response within about one second after the sign language video is played.

1.3 Research Problems

This paper has two challenges: improvement in recognition rate and real-time processing. Regarding recognition rate improvement, two methods are proposed based on the Skeleton Aware Multi-Model Sign Language Recognition (SAM-SLR [1]) model from previous research. The first method aims to further enhance the recognition rate by using fewer modalities, assuming the introduction of future mobile devices. According to the recognition results using the Turkish Sign Language Dataset, Ankara University Turkish Sign Language (AUTSL), one of the modalities, Multi-stream, has achieved the highest recognition rate at 96.47%. Also, Multi-stream consists of four streams: Joint, Bone, Joint Motion, and Bone Motion, with different posture position information, and their recognition rates are 95.35%, 95.69%, 93.21%, and 93.29%, respectively.

Furthermore, to improve the recognition rate, we propose a method to reuse the estimated results output for each epoch and enhance the recognition rate for Joint and Bone. The second method improves the recognition rate more than the original method to reach the level of human recognition. While SAM-SLR currently achieves a high recognition rate of 97.94% on our PCs, further enhancing the recognition rate presented a challenge. One researcher assumed that the recognition rate could be increased to just under 99%. The proposed method focuses on the low recognition rate when the difference between the Top-1 and Top-2 evaluation values in the recognition results of the SAM-SLR model is slight and performs re-evaluation. The re-evaluation method is a process of reassessing the relative position of the index finger in the face area at the reference point of the face part. In real-time processing, based on the model from previous research, the four independent modalities are processed in real-time to shorten the recognition response time. The original Optical Flow does not work on current GPUs to perform real-time processing, so modifying the code to produce an output equivalent to the original method is necessary. Additionally, to shorten the recognition response time, the parallel processing and attitude estimation process from the video playback time were reduced from two images to one image.

1.4 Proposed Approaches

In this study, Skeleton Aware Multi-Model Sign Language (SAM-SLR) [1] was utilized as the sign word recognition model, and Ankara University Turkish Sign Language (AUTSL) [2] served as the sign language dataset. We implemented a method of reusing the estimation results for each epoch during learning and introduced another method for re-evaluation. The re-evaluation process involved comparing relative index finger positions based on face part positions, with attention to the observation that the recognition rate tends to be low when there is a small difference between the Top-1 and Top-2 evaluation values of recognition results. Additionally, we addressed speed improvement by fixing the issue with Optical Flow processing, which does not function with the current GPU, leading to a reduction in recognition response time. Subsequently, we elaborate on the research methodologies for each approach.

1.4.1 How to reuse estimation results per epoch

This study focused on the Multi-stream modality, which boasts the highest recognition rate in the SAM-SLR model. Multi-stream is a method that achieves this enhancement through Late Fusion, combining recognition results from four streams: Joint, Bone, Joint Motion, and Bone Motion. Our specific target was the highly recognized Joint and Bone streams. The proposed reuse method involves reusing Top-1 estimation results for each epoch during learning, where Top-1 signifies the estimation result of the class with the highest value. Subsequently, we compared recognition rates for Joint, Bone, Multi-stream, and SAM-SLR, evaluating the proposed approach against conventional methods.

1.4.2 A method to re-evaluate by focusing on the difference in evaluation values after recognition results

In this study, we proposed a re-evaluation method focusing on the low recognition rate when the difference between the Top-1 and Top-2 recognition evaluation values is low in the recognition results of the SAM-SLR model. The evaluation process determines whether to initiate re-evaluation based on the disparity in evaluation values. Specifically, it involves creating a triangular mesh using the face part's position and comparing the index finger's position on the mesh. This method can more accurately capture sign language recognition, especially when fingers touch the mouth or nose. It also elucidates variations in recognition results attributed to differences in recognition rates among conditions such as one hand and both hands, as well as variations in evaluation calculations between 1.0, 2.0, and 3.0.

1.4.3 Process speed improvement method

In this study, the method for improving the processing speed of the proposed method is based on the SAM-SLR model and is divided into two parts. The first step is to modify the four independent modality processes to output the recognition results for each video from all the test data, connect each seamlessly, and output the recognition results by Late Fusion. The second step is to shorten the recognition response time. The problem in the first step is that the software of Optical Flow, which is one of the modalities, does not work with the current GPU, so it was necessary to correct the code to achieve the same results. The results showed that the evaluation values of the original method and the proposed method, the modified code, were approximately the same. In the second stage, the imperative was to reduce processing time in multiple areas. The challenge lay in shortening and parallelizing the posture estimation process, converting Python to C++, and utilizing internal memory. The original method uses a method of processing posture estimation on two images and employing more reliable position information. In the proposed method, attitude estimation processing was performed only for 640x640 images. The results showed similarities between the original method and the outcomes, alongside a 1.6 times increase in processing speed. We also found that if this process is executed only from video playback, the process is completed during video playback. This study compares the case where the four modalities are serially processed with the proposed method to show its effectiveness. We also discuss the differences in evaluation values between the conventional and proposed methods for posture estimation and Optical Flow processes.

1.5 Chapter outlines

The structure of the next chapter is outlined as follows. Chapter 2 provides an overview of previous research on isolated sign language recognition. Specifically, the SAM-SLR model serving as the foundation for this study will be detailed extensively. Chapter 3 introduces a method for enhancing isolated sign language recognition by reusing estimation results for each epoch during machine learning. In Chapter 4, the focus is directed toward addressing the low recognition rate, specifically when the difference between the Top-1 and Top-2 evaluation values in the recognition results of SAM-SLR is slight. Chapter 5 proposes a method designed to increase the recognition speed of the SAM-SLR model. Finally, Chapter 6 provides a summary of the content of this article.

Chapter 2

Related works

2.1 Models of isolated sign language recognition

Sign language recognition (SLR) has achieved great strides in recent years to achieve high recognition accuracy, thanks to the development of practical deep learning architectures and a surge in computational power. Researchers often use methods in which images are preprocessed to obtain finger, arm, nose, and mouth position information through Optical Flow and posture estimation processes, and 3DCNN is used to model spatio-temporal information for sign language recognition. Especially each modality has recently been fused to improve recognition rates further. One example is the SAM-SLR [1] model, which improves performance by fusing multi-modal information. Specifically, on RGB video recognition, three distinct architectures (3DCNN, SSTCN [1], SL-GCN [1]) were employed to extract features from four modalities RGB-Frames [1], RGB-Flow [1], SSTCN, and SL-GCN for independent sign language recognition. For preprocessing, the Denseflow API provided by OpenMMLab [3] with OpenCV and CUDA implemented is used to obtain Optical Flow using the TV-L1 algorithm [4]. MMPose [5] is also used for attitude estimation, which is used to preprocess SSTCN and SL-GCN. In this chapter, we first present the data sets used, the four modalities RGB-Frames, RGB-Flow, SSTCN, and SL-GCN, then the Multi-stream [1] used in SL-GCN, and finally, Late Fusion and GEM for multi-modal ensembles.

The dataset AUTSL [2] was created for a sign language recognition competition and contains Turkish Sign Language. It contains 36,302 video files, each of RGB and Depth Video. (Table 2.1) This data set is publicly available on the Internet and can be easily obtained by anyone. The videos are all 512 x 512 in size, with the signer standing or sitting almost in the center. Only RGB is used in this case. (Fig. 2.1) The number of signs for the words used is 226, which is small compared to other datasets (500 words for SLR500 [6], 2,000 words for WLASL-2000 [7], 1,000 words for MS-ASL [8], and 1,064 words for BSL-1K [9]), but the sample size is large compared to other more significant than the other datasets.

Table 2.1. Summary of isolated sign language datasets.

Dataset	Signs	Signers	Language	Samples			
				Training	Validation	Testing	Total
AUTSL	226	43	Turkish	28,142	4,418	3,742	36,302

Fig. 2.1. Images of dataset AUTSL.

Table 2.2 illustrates the recognition rates of the Test data when exclusively trained on the Train data from the AUTSL dataset, which was utilized in the previously mentioned competition. The table lists various models with publicly available codes. Notably, the SAM-SLR model exhibits a commendable recognition rate, serving as the foundational reference for our proposal to enhance recognition rates and processing speed. A detailed explanation of this model will be provided later in the text. The discussion of other models is reserved for Chapter 4, Introduction.

Table 2.2. Main competition results using Train dataset for AUTSL.

	Models	Accuracy
1	SAM-SLR ver.1	97.62%
2	USTC-SLR	97.62%
3	Novopoltsev et al.	95.72%
4	Coster et al.	92.92%

2.2 Skelton Aware Multi-modal Sign Language Recognition (SAM-SLR) model

SAM-SLR [1] stands for Skeleton Aware Multi-Model Sign Language Recognition, and this model Features four modalities: Multi-stream, Features, RGB-Frames, and RGB-Flow's recognition results are fused with late fusion to increase the recognition rate. (Fig. 2.2)

Starting from the top of the figure, Multi-stream recognizes the positional information of MMPose hands and fingers as input information in the SL-GCN (Sign Language Graph Convolutional Network) model. Features uses a 24x24 heat map of the image as input information for the location of the chin, mouth, shoulders, arms, wrists, fingertips, and the base of the fingertips to be recognized by the SSTCN (separable spatial-temporal convolutional network) model. RGB-Frames crops each image so that the person is in the center and recognizes it with a 3DCNN model. For RGB-Flow, each image is processed by Optical flow, cropped in the same way as RGB-Frames, and recognized by the 3DCNN model. The code for optical flow processing in SAM-SLR is implemented in C++, while the remainder is scripted in Python, with both the learning and recognition processes executed in PyTorch. In addition, all four modalities use position information from the MMPose posture estimation process.

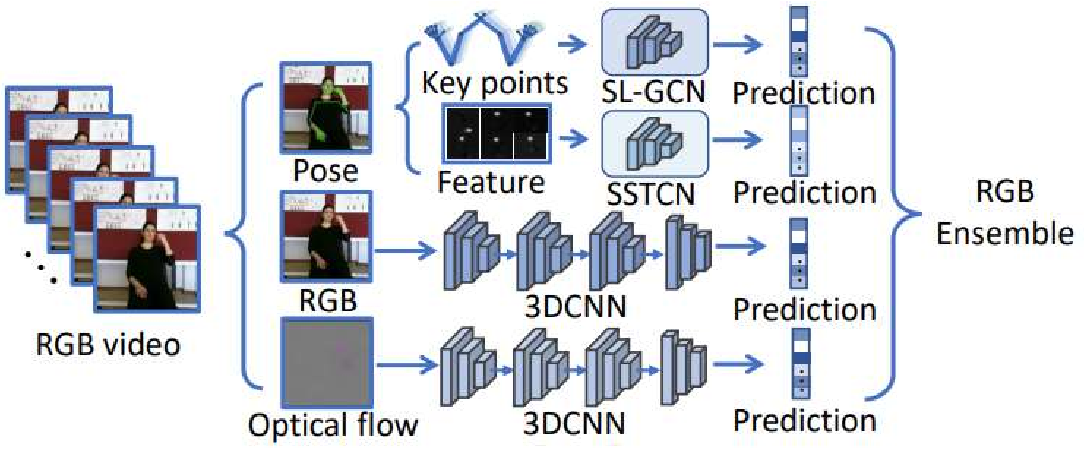


Fig. 2.2. The skeleton aware multi-modal sign language recognition framework (SAM-SLR [1]).

Table 2.3 displays the recognition results for the Validation data trained on the Train data of the AUTSL dataset, showcasing the recognition rate for each stream and modality used in the Multi-stream. Ultimately, the recognition rate increased to 96.96% due to Late Fusion applied to each modality. Notably, the recognition rate for the

Validation data is slightly lower than that of the Test data across each stream and modality. The Multi-stream and Joint also exhibit high recognition rates at 95.45% and 95.02%, respectively. The trend in recognition rates for each stream and modality closely mirrors that observed in the Test data.

Table 2.3. Result of Multi-stream and each modality on AUTSL Validation set.

Stream	Top-1	Top-5	Modality	Top-1	Top-5
Joint	95.02	99.21	Multi-stream	95.45	99.25
Bone	94.70	99.14	Features	94.32	98.84
Motion Joint	93.01	98.85	RGB Frames	94.77	99.48
Motion Bone	92.49	98.78	RGB Flow	91.65	98.76

2.2.1 Multi-stream

First, two images are prepared by creating a 640x640 image from a 512x512 image, one frame at a time from the sign language video. Next, the positional accuracy of the two images is compared, and the better information is used and saved to a file (Ex. signer6_sample1_color.npy). Joint uses the position information of the fingertips and hands as joint positions and generates a file (e.g., data_joint. npy). Bone uses the middle position of the two joints to generate a file (e.g., data_bone.npy). Joint Motion generates a file (e.g., data_joint_motion.npy) using the difference between the positions of the joints in the two frames. Bone Motion generates a file (e.g., data_bone_motion.npy) using the difference between the positions of the bones in the two frames. Bone Motion uses the difference in bone position between the two frames to generate a file (e.g., data_bone_motion.npy).

In the machine learning process, using the SL-GCN (Sign Language Graph Convolutional Network) model, Joint, Bone, Joint Motion, and Bone Motion each have a posture estimation information file (e.g., data_joint.npy, data_bone.npy, data_joint_motion.npy, data_bone_motion.npy) are trained separately as input data. A file of trained weights for each epoch (e.g., sing_joint_final-0.pt, sing_bone_final-0.pt, sing_joint_motion_final-0.pt, sing_bone_motion_final-0.pt, sing_joint_motion_final-0. pt, sing_bone_motion_final-0.pt), respectively, are generated.

The recognition process involves loading a file (.pt file) of data of trained weights into the SL-GCN model, resulting in a file (.pkl) of recognition results. Multi-stream is created by Late Fusion of the evaluation values of the recognition results of the four streams.

Table 2.4. Each process and code file name with Names of example related files for Multi-stream.

Each process	Code file name	Input file examples	Output file examples
Preparing	demo.py	signer6_sample1_color.mp4	signer6_sample1_color.npy
Joint's preparing	sign_gendata.py	signer6_sample1_color.npy	data_joint.npy
Bone's preparing	gen_bone.py	data_joint.npy	data_bone.npy
Joint-Motion's preparing	gen_motion_data.py	data_joint.npy	data_joint_motion.npy
Bone-Motion's preparing	gen_motion_data.py	data_bone.npy	data_bone_motion.npy
Learning	main.py	data_joint.npy data_bone.npy data_joint_motion.npy data_bone_motion.npy	sign_joint_final-0.pt sign_bone_final-0.pt sign_joitn_motion_final-0.pt sign_bone_motion_final-0.pt
Recognition	main.py	data_joint.npy data_bone.npy data_joint_motion.npy data_bone_motion.npy	epoch_0.pkl epoch_0.pkl epoch_0.pkl epoch_0.pkl

The Multi-stream evaluation value is obtained by substituting the estimated results q and weights α = {1.0, 0.9, 0.5, 0.5} for each stream of Joint, Bone, Joint Motion and Bone Motion into the following formula (2.1).

$$q_{Multi-stream} = \alpha_1 q_{Joint} + \alpha_2 q_{Bone} + \alpha_3 q_{JM} + \alpha_4 q_{BM} \tag{2.1}$$

2.2.2 Features

A novel separable spatial-temporal convolutional network (SSTCN) that learns from whole-body skeletal features was devised to exploit the information of whole-body key points fully. The model's creators say they learned from ResNet2+1D [11] that the performance of an action recognition model could be further improved by factorizing the network into temporal and spatial parts. This section describes the data processing procedure for Features. Each image is extracted from a sign language video file and processed. Each image is extracted from a sign language video file and processed. Next, each frame of the image is resized from 512x512 to 384x384. Finally, each RGB color component of the image is divided by 255, then 0.406, 0.456, and 0.485 are added to each respective component, and the result is divided by 0.225, 0.224, and 0.229. This processed image is then used in the posture estimation process by MMPose. (Fig. 2.3)

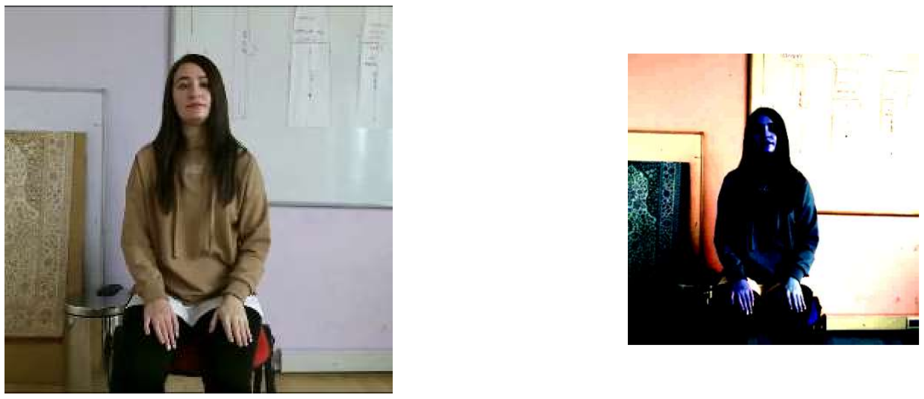

(a) Original image. (b) Resized image with brightness adjustment.

Fig. 2.3. Preparing of image for Features.

The number of images in the input data is adjusted by 60. To always have 60 images, the images are thinned out and adjusted evenly for continuous use. When there are exactly 60 images, use them all, numbered from 0 to 59. If there are 59 images, repeatedly use image 0, resulting in the sequence 0, 0, 1, 2, ..., 57, 58. For cases with 58 or fewer images, continuously use image 0 and multiples of the total number divided by (60 - total number). For instance, with 58 images, use 0 and 29 in a repeating pattern, like 0, 0, 1, 2, ..., 28, 29, 29, 30, …, 56, 57. If there are 61 images, omit image 0 and use images 1 through 60. For cases with 62 or more images, selectively omit image 0 and multiples of (total number + 1) divided by (total number - 60). For example, with 62 images, omit images 0 and 31, and use images 1, 2, ..., 29, 30, 32, 33, ..., 60, 61. With 63 images, omit images 0, 21, and 42, and use images 1, 2, ..., 19, 20, 22, 23, ..., 40, 41, 43, 44, ... 61, 62. This ensures that the total number of images is always adjusted to 60 by selectively omitting and using images continuously. (Table 2.5)

Table 2.5. Adjusting image numbers for a total of 60 images.

Number of Images	Consecutive number	Omitted number	Example results
58	0, 29		0, 0, 1, 2, …, 28, 29, 29, 30, …, 56, 57
59	0		0, 0, 1, 2, …, 57, 58
60			0, 1, 2, …, 58, 59
61		0	1, 2, … 59, 60
62		0, 31	1, 2, …, 29, 30, 32, 33, …, 60, 61
63		0, 21, 42	1, 2, … 19, 20, 22, 23, …, 40, 41, 43, 44, …, 61, 62

A heatmap (96x96) of each of the 33 positions (chin, mouth, shoulders, arms, wrists, fingertips, and base of fingertips) is extracted from the images using the posture estimation process MMPose wholebody_w48_384x384_adam_lr1e-3. The heatmap size is then downsized to 24x24, and this information is used as input data and saved as a file (.pt). The machine learning process uses the training data file (.pth) as input information for the SSTCN model and generates a training file (.pth). The recognition process also causes the SSTCN model to read the training file (.pth) and derive recognition results using the training file (.pt) of test data as input information.

Table 2.6. Each process and code file name with Names of example related files for Features.

Each process	Code file name	Input file examples	Output file examples
Preparing	wholepose_features_extraction.py	signer6_sample1_color.mp4	signer6_sample1_color.pt
Learning	train_parallel.py	signer6_sample1_color.pt	T_Pose_model.pth
Recognition	test.py	signer6_sample1_color.pt	T_Pose_model_test.pkl

2.2.3 RGB-Frames

The modalities RGB-Frames and RGB-Flow (Optical Flow) are modeled using a 3DCNN, using the ResNet2+1D [11] architecture that separates the temporal and spatial convolution of a 3DCNN. The model features the ResNet2+1D-18 variation pre-trained on the Kinetics [12] dataset as the backbone of choice, improving accuracy. In addition, the model is pre-trained on the largest dataset available, SLR500, for RGB Frames. The following describes the data processing procedure for RGB-Frames.

First, frame images are extracted from the sign language video individually. The posture estimation process MMPose is processed from the frame images to create a position information data file. (.npy) The file of each frame image is used to extract the joint position information of the chin, both eyes, both ears, both shoulders, both elbows, both wrists, and both hands, and the left end, right end, top end, and bottom end (Fig. 2.4(a)) are obtained so that the person is centered using the following formula to create a 256x256 image (Fig.2.4(b)) (Table 2.7, Preparing 2). Input Test data to the developed model and get estimation results.

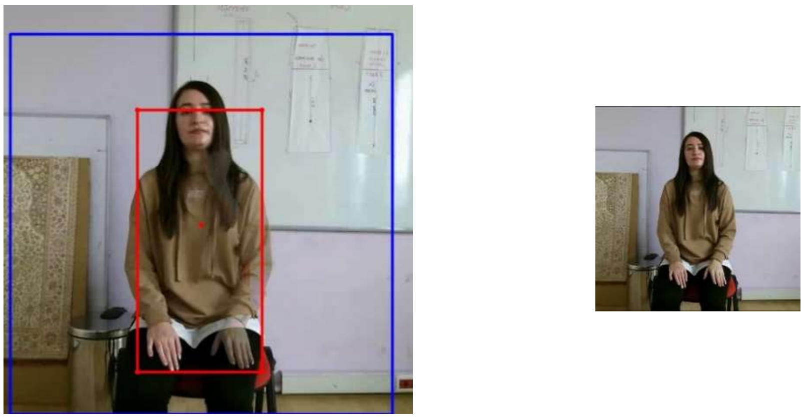

(a) Crop image (b) 256x256 image

Fig. 2.4. Preparing of image for RGB-Frames.

Table 2.7. Each process and code file name with Names of example related files for RGB-Frames.

Each process	Code file name	Input file examples	Output file examples
Preparing 1	demo.py	signer6_sample1_color.mp4	signer6_sample1_color.npy
Preparing 2	gen_frames.py	signer6_sample1_color.mp4 signer6_sample1_color.npy	0000.jpg
Learning	Sign_Isolated_Conv3D_clip.py	0000.jpg	results_epoch_.pkl
Recognition	Sign_Isolated_Conv3D_clip_test.py	0000.jpg	results_epoch_.pkl

2.2.4 RGB-Flow

RGB-Flow (Optical Flow) modality is modeled using a 3DCNN, like RGB-Frames. The following describes the data processing procedure for RGB-Frames. First, frame images are extracted from the sign language video individually. Next, Optical flow processing (Table 2.8, Preparing 2) creates two images (Fig. 2.5(c) and (d)) in x- and y-axis directions from two consecutive frame images (Fig. 2.5(a) and (b)). The optical flow image (Fig. 2.5(e)) employed in the RGB-Flow process is composed by overlaying X-axis optical flow (Fig. 2.5(c)) on the B frame, Y-axis optical flow (Fig. 2.5(d)) on the G frame, and once again, X-axis optical flow on the R frame (Table 2.8, Preparation 3). Finally, similar to the RGB-Frames process, the optical flow image is cropped using positional information from posture estimation to center the person and then resized to 256x256.

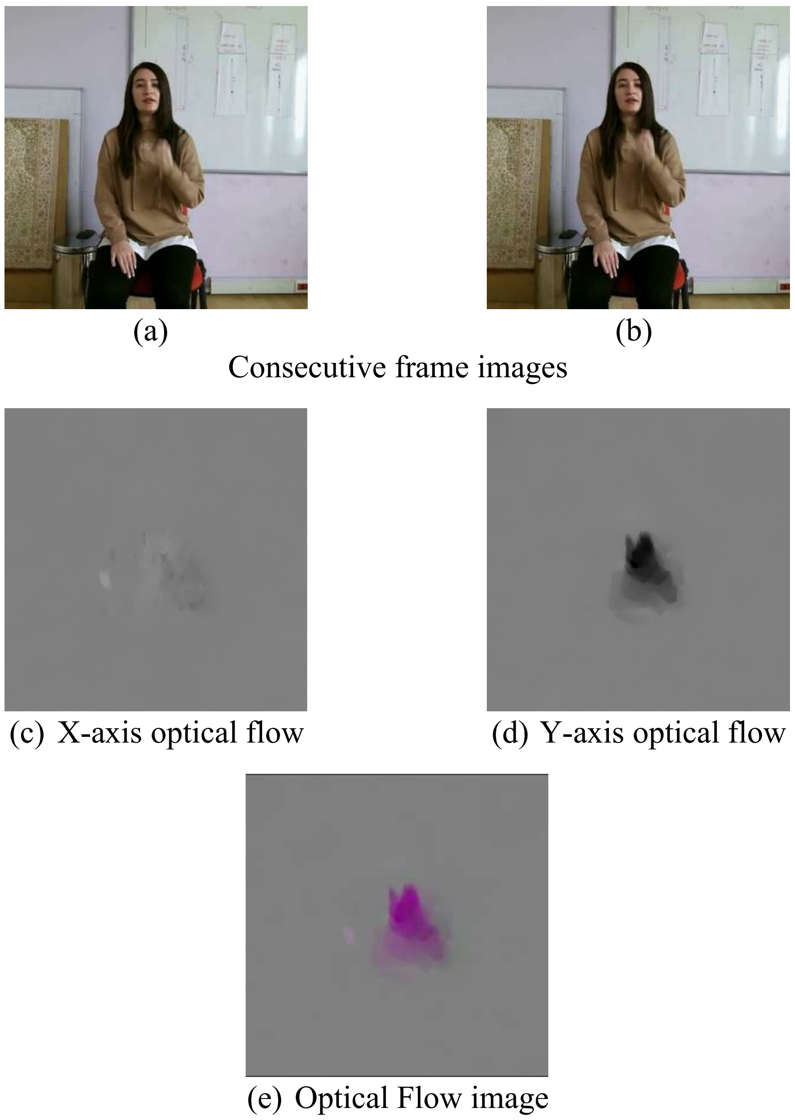


Fig. 2.5. Preparing of image for RGB-Flow.

Table 2.8. Each process and code file name with Names of example related files for RGB-Flow.

Each process	Code file name	Input file examples	Output file examples
Preparing 1	demo.py	signer6_sample1_color.mp4	signer6_sample1_color.npy
Preparing 2	Docker (extract_optical_flow.sh)	signer6_sample1_color.mp4	flow_x_00000.jpg flow_y_00000.jpg
Preparing 3	gen_flow.py	signer6_sample1_color.npy flow_x_00000.jpg flow_y_00000.jpg	0000.jpg
Learning	Sign_Isolated_Conv3D_flow_clip.py	0000.jpg	results_epoch.pkl
Recognition	Sign_Isolated_Conv3D_flow_clip_test.py	0000.jpg	results_epoch.pkl

Chapter 3

Enhancing Sign Language Recognition through the reuse of estimate results by each epoch

3.1 Introduction

It is said that there are 70 million people in the world who use sign language. In our daily life, we sometimes see images of signers on TV news doing simultaneous interpretation, or we can learn it in school. We are doing research on real-time sign language recognition to facilitate communication with signers, and we believe that the speed of recognition results is one of the important aspects. In this study, we focus on an isolated SLR task and use RGB from the dataset [2], which contains videos of Turkish Sign Language. Among various sign language recognition methods, Skeleton Aware Multi-modal SLR (SAM-SLR) [1] won a competition using this dataset and was reported in a workshop [13]. The follow-up, SAM-SLR-v2 [10], has further improved the recognition rate. SAM-SLR and SAM-SLR-v2 fuse the results of multiple modalities to increase the recognition rate. The recognition rate for Multi-stream (SL-GCN), one of the modalities, is 96.47% [10], which is a remarkably high recognition rate among the other modalities. This Multi-stream was obtained by integrating the recognition rates of four streams (Joint, Bone, Joint Motion and Bone motion), two of which used Joint and Bone features, and the respective recognition rates were 95.35% and 95.69%, which were already higher than those of other modalities (Table 3.1). We investigated the possibility of reducing recognition processing by focusing on these two streams Joint and Bone from the four modalities to shorten the response time and improve the recognition rate of each single stream. Then, we propose a method to reuse the estimate results of the training model created for each epoch in the Multi-stream sign language graph convolutional network (SL-GCN) of SAM-SLR [1] in order to further improve the recognition rate. Specifically, in SL-GCN, there are recognition methods using posture information for each stream, and we apply the proposed method to two methods from these streams, Joint and Bone. (Fig. 3.1) For this method, there are two phases. In the first phase, using the training model information created up to the last epoch, only the Top1 ratings of the recognized estimate results using each test data are summed for each class. The class with the highest sum is designated as the final class. In the second phase, we select all the estimate results for which this final class is No. 1 from the estimate results for all the epochs made up to the final epoch. The estimation result closest to the final epoch is then the final estimation result. Our main contributions can be summarized as follows.

1) Since we reuse the recognition results of the per-epoch learning model created by the Multi-stream sign language graph convolutional network (SL-GCN), there is no need to create new per-epoch learning model results.

2) Also, since it reuses the information from the estimate results created for each epoch, there is no need to create new recognition results using test data.

3) The computational complexity of this method is very small, since it is limited to the aggregation and sorting of the Top-1 evaluation values of the recognition results output for each epoch for each test data by class.

4) It can provide a very high recognition rate, especially up to Epoch 150.

5) We report the results of our application to the SAM-SLR models in an ensemble model using proposed Joint and Bone.

Source	Modality	Streams	Preprocessing	Models	Multi-modal Late-fusion Ensemble
RGB Videos	RGB-Frames		Frames	3DCNN	Global Ensemble Model (SAM-SLR-v2 [10])
	RGB-Flow		Optical flow	3DCNN	
	Features		Pose Features	SSTCN	
	Multi-stream	**Joint**	Pose Graph 2D	SL-GCN	**Model-free Late Fusion (SAM-SLR [1])**
		Bone			
		Joint Motion			
		Bone Motion			

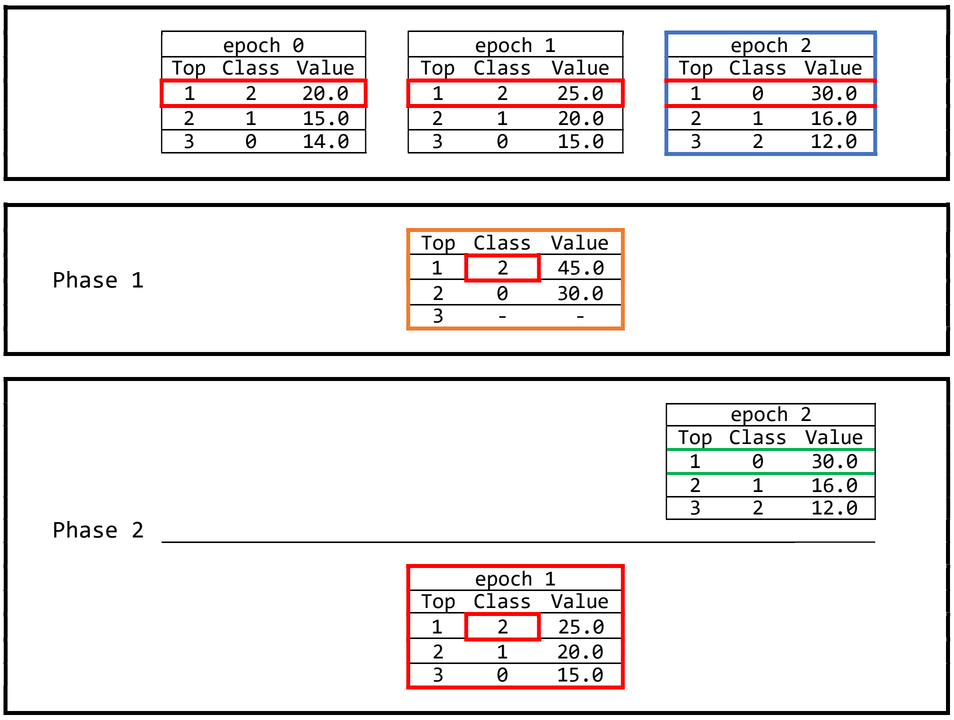
Fig. 3.1. Concept of the skeleton aware multi-modal sign language recognition framework (SAM-SLR [1], v2 [10]). Bold include proposed Joint & Bone.

Table 3.1. Result of each modality and Multi-stream SL-GCN on AUTSL Test set. [1]

Modality	Top-1	Top-5	Stream	Top-1	Top-5
RGB Frames	95.00	99.47	Joint	**95.35**	99.49
RGB Flow	90.41	98.74	Bone	**95.69**	99.55
Features	93.37	99.28	Motion Joint	93.21	99.12
Multi-stream	**96.47**	99.76	Motion Bone	93.29	99.31

3.2 Methodology

We propose a method to improve the recognition rate by reusing the estimate results for each epoch output by the recognition method using the Joint and Bone features used in SAM-SLR. (Fig. 3.1) The proposed method will be a method that reuses and re-evaluates all the estimate results of the training model made before that epoch to obtain the estimate results of a given epoch. (Fig. 3.3) Generally, when a learning model is used for recognition, a single learning model created at a single epoch produces estimate results (Fig. 3.2 blue box). In the proposed method, for example, when recognizing Epoch 2, the estimate results of the training models of Epoch 0, 1, and 2 are used. There are two main steps in this method: (Fig. 3.2) In Phase 1, the Top-1 values are used from the estimate results of each epoch, and the Top-1 values are summed for each class, and the largest one is the selected class. Figure 3-2 illustrates that, for example, in the case of epoch 2 of the learning model, in Phase 1, using the estimate results of epochs 0, 1, and 2, the Top1 estimates at each epoch are used to select class 0 and 2 as candidates. The total value is 30.0 and 45.0, respectively, and class 2 is selected. (Fig. 3.2 orange box) In Phase 2, the selected class selects the Top-1 estimates from the estimates of each epoch. Then, the most recently created epoch estimation result is adopted from among them. In Phase 2, the epochs when Top1 is in class 2 are epochs 1 and 0. The closest to epoch 2 is epoch 1, which is selected. (Fig. 3.2 red box)



epoch 0		
Top	Class	Value
1	2	20.0
2	1	15.0
3	0	14.0

epoch 1		
Top	Class	Value
1	2	25.0
2	1	20.0
3	0	15.0

epoch 2		
Top	Class	Value
1	0	30.0
2	1	16.0
3	2	12.0

Top	Class	Value
1	2	45.0
2	0	30.0
3	-	-

epoch 2		
Top	Class	Value
1	0	30.0
2	1	16.0
3	2	12.0

epoch 1		
Top	Class	Value
1	2	25.0
2	1	20.0
3	0	15.0

Fig. 3.2. Method of the reuse of estimate results by each epoch.

```
Data: results[epochs][classes]
class_num ← 226
sum[]  ← 0
for i ← 0 to final_epoch do
max_value ← 0.0
  class_top ← 0
  for j ← 0 to class_num - 1 do
    if results[i][j] > max_value then
      max_value ← results[i][j]
      class_top ← j
end
  end
sum[class_top]←sum[class_top]+max_value
  top_class[i] ← class_top
end
max_sum ← 0
for i ← 0 to class_num - 1 do
if sum[i] > max_sum then
    max_sum ← sum[i]
    select_class ← i
  end
end
select_epoch ← 0
for i ← final_epoch to 0 step -1 do
if top_class[i]=select_class then
select_epoch ← i
  break
  end
end
return results[select_epoch]
```

Fig. 3.3. Algorithm of the reuse of estimate results by each epoch.

Table 3.2 and 3.3 show examples of what epochs were chosen. Table 3.2 shows how many epochs ago the adopted epoch is in epochs 150 to 229 of Joint and Bone, respectively. For Joint, 98.19% of the epochs were the same, 0.54% were one epoch earlier, and 0.24% were two epochs earlier. In Bone, the same epoch was 97.84%, 0.55%, and 0.25%, respectively. These results show that most of the epochs remain the same, with about 2% being replaced. Table 3.3 shows some examples of epoch changes for epoch 200 Joint and Bone. In the Bone example, there were 3,742 test data, of which 73 chose epochs different than epoch 200. Most of the epochs that were changed were also near epoch 200, such as 199 and 198. However, in case No. 724 in the test data, epoch 137 was chosen 63 epochs ago. The result of this case is epoch 137, because there were many Top-1s in the corresponding class from epoch 0 to 137. For the correct answer in this case, it was another applicable class after epoch 138.

The model is simple in its methodology, the method is computationally inexpensive, and the estimate results previously produced can be reused. Specifically, regarding the computational cost of this method, the advantage is that when creating a training model, the best training model is created by using training data and learning, building up epoch 0, 1, 2..., etc. In this method, the training model created in the process of creating the training model is used to Since it outputs the estimated results of the test data, there is no need to create a new learning model. If the estimate results are output from test data at each epoch, the data can be reused and used in this method.

Table 3.2. The rate of selected epoch. (Joint and Bone)

	epoch before	same epoch	1	2	3 or more	Total
Joint	Amount	293,938	1,608	723	3,091	299,360
	Rate	98.19%	0.54%	0.24%	1.03%	100%
Bone	Amount	292,901	1,644	749	4,066	299,360
	Rate	97.84%	0.55%	0.25%	1.36%	100%

Table 3.3. Changed selected epoch on epoch. (Joint and Bone)

Test Data No	42	149	417	438	…	3,522	3,534	3,681
Joint	199	198	187	197		195	165	196
Test Data No	26	171	186	…	724	…	3,634	3,681
Bone	199	198	177		137		199	193

3.3 Results

We used the published code used in SAM-SLR [1], created training models for each method on our PC, used test data, and produced estimate results. Our PC specs are AMD 3960x (CPU), RTX-3090 (3 GPUs), 128GB (RAM), Ubuntu 18.04 (OS) and we did most of the processing on this PC. The comparison method consists of two patterns: the conventional method of SAM-SLR [1] and the application of the proposed method for Joint and Bone. To know the impact of the proposed method, we measured the recognition rate in three locations: Stream SL-GCN Joint and Bone, Multi-stream SL-GCN, and Multi-ensemble Late fusion. On section C and D the optimal parameters were given manually from 0.0 to 2.0. The test data consists of 3,742 RGB videos, each video containing one of 226 different signs. The evaluation method outputs an estimation result for 226 different signs per video. For the evaluation method of recognition rate, the number of correct answers is counted as Top-1 when the correct answer is the highest in the evaluation value for that class, and Top-5 when the correct answer is included in the top 5.

$$Recognition\,rate\;of\;Top\text{-}n = \frac{Total\;number\;of\;Top\text{-}n}{3{,}742} \tag{3.1}$$

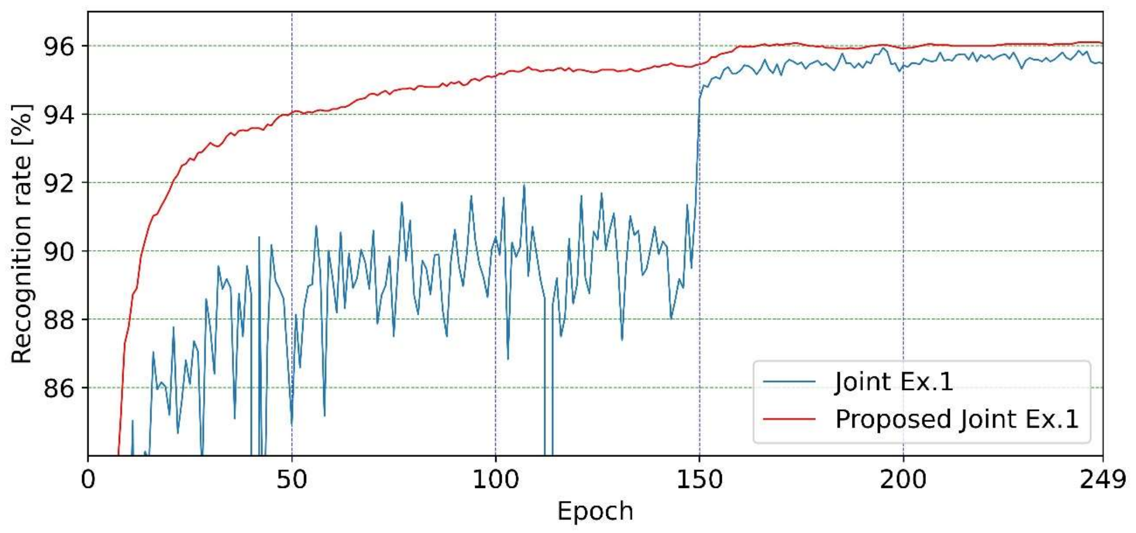
3.3.1 Stream SL-GCN Joint

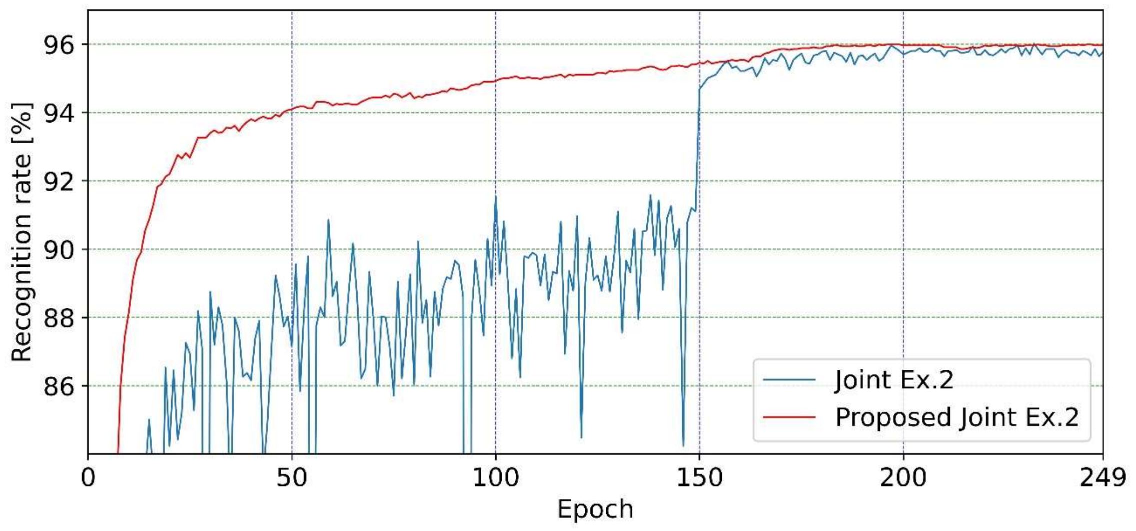
For the Top1 recognition rate, SAM-SLR Joint was 95.35% for paper [10], although it was unclear at which epoch, and the highest value was 95.94% in our PC environment, which was 96.10% for the proposed method, 0.16 percentage points higher. (Table 3.4, Example 1) The average recognition rate for the 80 epochs from epoch 150 to 229 was 95.46% for the original method and 95.96% for the proposed method, which is 0.50 percentage points higher. (Fig. 3.4, Example 1) The reason for the improved recognition rate after epoch 150 in the graph of Joint for the original method is that the learning rate is lowered from 0.1 to 0.01 in the default setting to suppress oscillations during training. This phenomenon is also observed in Bone.

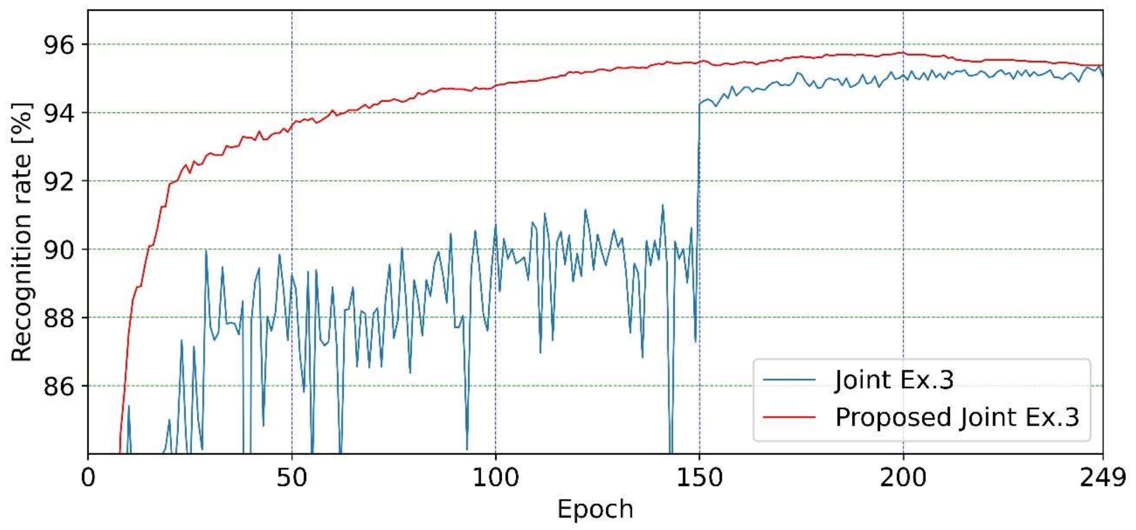
Table 3.4. Results of single stream (Example 1) of SAM-SLR and Our PC using SAM-SLR code on AUTSL Test set.

Streams	SAM-SLR-v2 [10]		Our PC		
	Top -1	Top-5	Top-1	Top-5	Average of Top-1*
Joint (Example 1)	95.35	99.49	95.94	99.63	95.46
Proposed Joint (Example 1)	-	-	**96.10**	**99.65**	**95.96**
Bone (Example 1)	95.69	99.55	96.02	99.65	95.38
Proposed Bone (Examples 1)	-	-	**96.18**	**99.65**	**95.99**
Joint Motion	93.21	99.12	93.61	99.36	-
Bone Motion	93.29	99.31	93.77	99.41	-
Multi-stream	96.47	99.76	96.82	**99.87**	96.38
Multi-stream + **Proposed Joint & Bone (Example 1)**	-	-	**97.01**	**99.87**	**96.48**

* from epoch 150 to 229


96
94
92
90
88
86
Recognition rate [%]
0
50
100
150
200
249
Epoch
Joint Ex.1
Proposed Joint Ex.1


96
94
92
90
88
86
Recognition rate [%]
0
50
100
150
200
249
Epoch
Joint Ex.2
Proposed Joint Ex.2


96
94
92
90
88
86
Recognition rate [%]
0
50
100
150
200
249
Epoch
Joint Ex.3
Proposed Joint Ex.3

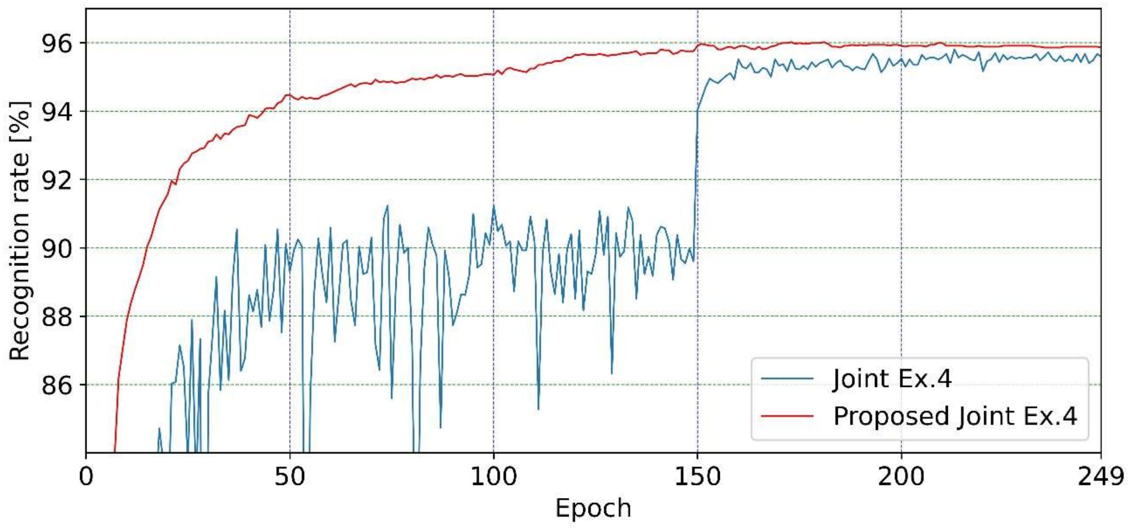


Fig. 3.4. The four comparison graphs of recognition rate for Joint and Proposed Joint across each training sessions with random initial conditions using AUTSL test set. All remaining graphs are in Appendix 3.

After an international conference presentation, we conducted nine additional trials, and the results are as follows. The proposed method was adapted to each training model and validated. For graphs Example 1 to Example 4, the results are almost similar, showing high recognition rates from the initial epoch. Examples 2, 3, and 4 in Figure 3.4 are from three newly trained models with different initial conditions. As with previous methods, Table 3.5 summarizes the peak recognition rate and its changes up to epoch 249 before and after applying the proposed method across ten trials, as well as the average recognition rate and its variations from epoch 150 to 229, along with the results of the T-test for recognition rates between epoch 150 and 249. The average peak recognition rate increased by 0.21 points while Example 2 depressed by 0.03. The average recognition rate increased by 0.49 points for each result. In the context of this analysis, the null hypothesis (H0) assumes no difference in recognition accuracy before and after application, while the alternative hypothesis (H1) posits a significant improvement. Rejecting the null hypothesis at a 0.001 significance level provides evidence of a statistically significant positive impact on recognition accuracy post-application. This rejection supports the alternative hypothesis, indicating that the reuse method has improved recognition performance. Table 3.6 presents a list of results indicating reductions in recognition rates following the application of the Reuse method after epoch 150. In epochs 150 to 249 over ten trials, the recognition rate declined by 15 data points out of 1000, resulting in an average drop of 0.053 points. The most significant decrease, 0.16 points, occurred in epoch 249 of Example 6, where the recognition rate fell from 95.67% to 95.51%. This decline is attributed to the rapid increase in the recognition rate in epoch 247, rising from 95.22%, which could not be effectively managed. The recognition rates for ten trials from epoch 0 to epoch 249 are listed at the end of the report. (Appendix 1)

Table 3.5. The ten comparison results of recognition rate for Joint and Proposed Joint across each training sessions with random initial conditions, including T-test outcomes.

	Peak Recognition Rate (Epoch 0 - 249)			Average of Top-1 (Epoch 150 - 229)			T-test for accuracy changes before and after applying Reuse method per epoch (Epoch 150 - 249)			
	Reuse method [%]		Improved	Reuse method [%]		Improved				
		✓			✓		MD	SDD	*t*-stat	*p* value
Ex.1	95.94	96.10	0.16	95.46	95.96	0.50	0.48	0.028	28.5	7.04^{-50} ***
Ex.2	96.02	95.99	-0.03	95.60	95.84	0.24	0.23	0.021	16.0	1.38^{-29} ***
Ex.3	95.35	95.75	0.40	94.90	95.58	0.68	0.60	0.068	23.2	4.17^{-42} ***
Ex.4	95.80	96.02	0.22	95.34	95.91	0.57	0.53	0.077	19.0	3.70^{-35} ***
Ex.5	95.51	95.86	0.35	95.10	95.57	0.47	0.44	0.030	25.2	3.42^{-45} ***
Ex.6	95.67	95.80	0.13	95.25	95.59	0.34	0.30	0.029	17.4	2.99^{-32} ***
Ex.7	95.67	95.67	0.00	95.12	95.53	0.41	0.35	0.072	12.9	3.39^{-23} ***
Ex.8	95.48	95.67	0.19	95.02	95.53	0.51	0.46	0.098	14.6	1.24^{-26} ***
Ex.9	95.64	95.80	0.16	95.26	95.65	0.39	0.33	0.071	12.5	2.79^{-22} ***
Ex.10	95.56	96.04	0.48	95.13	95.90	0.77	0.70	0.060	28.6	2.79^{-50} ***
Avg.	95.66	95.87	0.21	95.22	95.71	0.49	0.44	0.074	51.4	3.61^{-283} ***

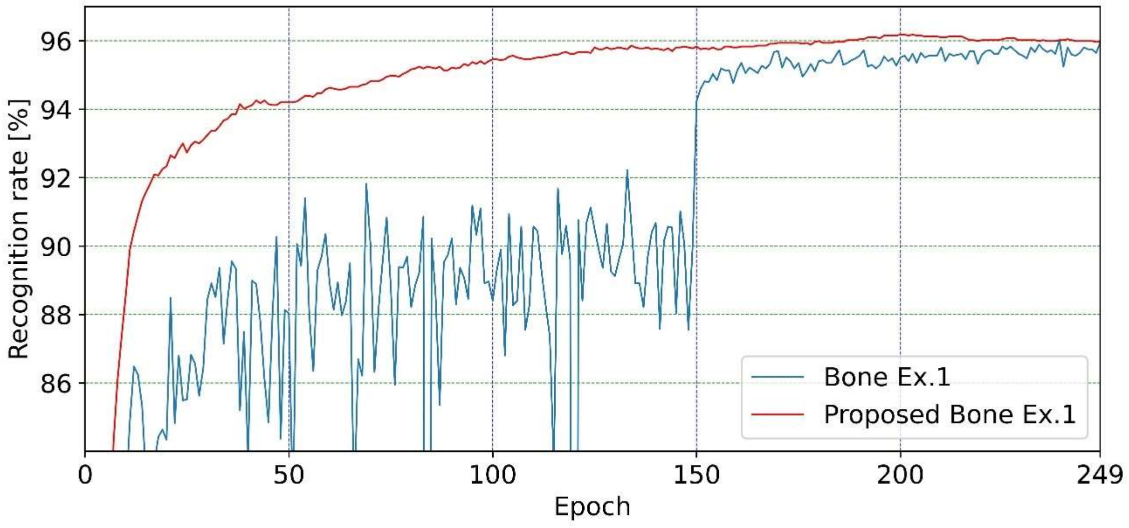
Note. MD = Mean of differences, SDD = Standard deviation of differences, ***$p < 0.001$

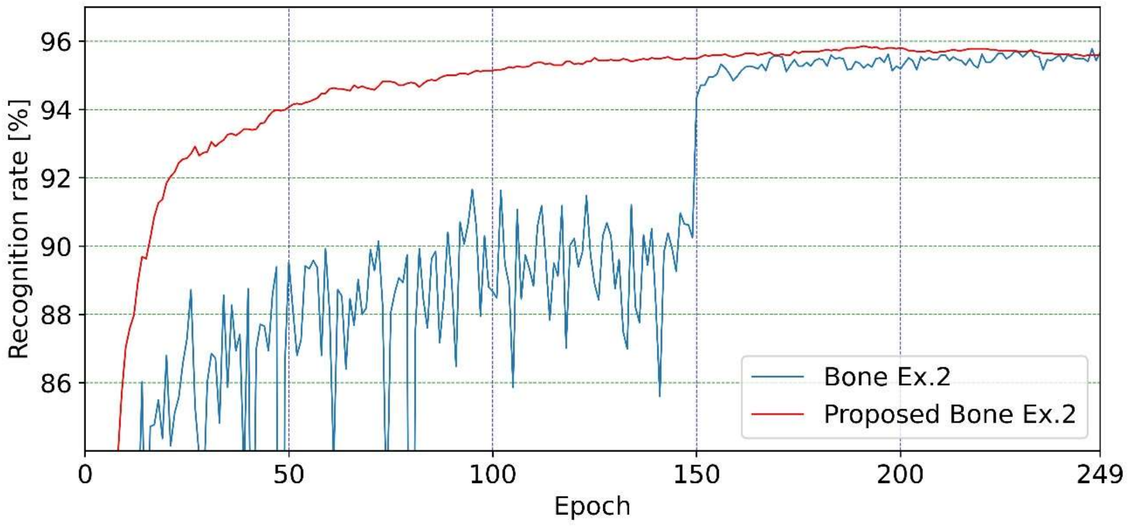
Table 3.6. Some results of recognition rate from ten trials of training Joint and Proposed Joint with random initial conditions. * All data is displayed in Appendix 1.

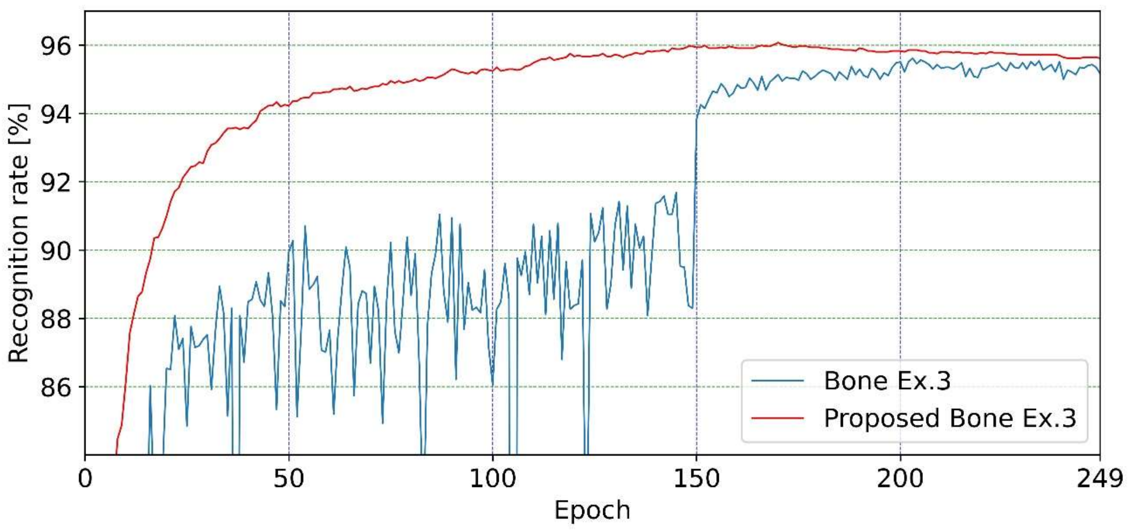
Epoch No	Example 2		Example 5		Example 6		Example 7		Example 9	
Reuse		✓		✓		✓		✓		✓
159	95.35	95.51	**95.16**	95.14	95.00	95.40	94.79	95.54	95.03	95.62
217	**95.94**	95.91	95.30	95.72	95.35	95.62	95.32	95.62	95.48	95.56
218	95.70	95.88	95.24	95.72	95.24	95.62	95.46	95.64	**95.62**	95.56
219	95.70	95.91	95.27	95.72	95.30	95.62	**95.67**	95.64	95.46	95.56
220	95.83	95.96	95.38	95.70	95.30	95.59	**95.67**	95.62	95.43	95.59
223	95.75	95.96	95.27	95.72	95.32	95.62	95.46	95.62	**95.62**	95.54
225	95.75	95.96	95.19	95.67	95.35	95.62	**95.64**	95.59	95.56	95.56
226	95.86	95.96	95.14	95.64	95.27	95.62	95.38	95.56	**95.62**	95.56
232	**96.02**	95.96	95.11	95.59	95.56	95.59	95.56	95.62	95.46	95.56
234	95.70	95.96	95.19	95.59	95.54	95.59	**95.67**	95.64	95.56	95.56
235	95.86	95.96	95.27	95.56	95.40	95.56	95.64	95.64	**95.62**	95.56
236	95.86	95.94	95.19	95.56	**95.62**	95.59	95.46	95.64	**95.59**	95.56
246	95.67	95.99	95.51	95.56	**95.51**	95.48	95.48	95.62	95.51	95.64
249	95.78	95.96	95.32	95.56	**95.67**	95.51	95.56	95.59	95.62	95.64

3.3.2 Stream SL-GCN Bone

Similar to the above method, for the Top1 recognition rate, SAM-SLR's Bone was 95.69% [10], with the highest value in our PC environment being 96.07%, which was 96.18% for the proposed method, 0.16 percentage points higher. (Table 3.4, Example 1) In all the different environments, Bone had a higher recognition rate than Joint. The average recognition rate for the 80 epochs from epoch 150 to 229, measured on our PC environment, was 95.38% for the original method and 95.99% for the proposed method, 0.61 percentage points higher. (Fig. 3.5, Example 1)







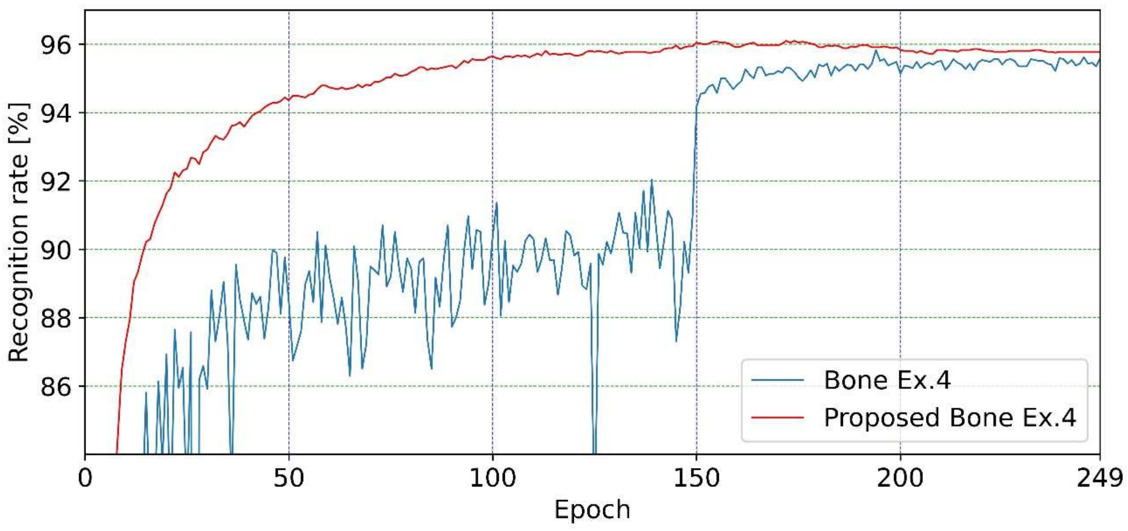


Fig. 3.5. The four comparison graphs of recognition rate for Bone and Proposed Bone across each training sessions with random initial conditions using AUTSL test set. All remaining graphs are in Appendix 4.

As with the Joint validation in the previous section, we conducted nine additional trials, and the results are as follows. The proposed method was adapted to each training model and validated. For graphs Example 1 to Example 4, the results are almost similar, showing high recognition rates from the initial epoch. Examples 2, 3, and 4 in Figure 3.5 are from three newly trained models with different initial conditions. As with previous methods, Table 3.7 summarizes the peak recognition rate and its changes up to epoch 249 before and after applying the proposed method across ten trials, as well as the average recognition rate and its variations from epoch 150 to 229, along with the results of the T-test for recognition rates between epoch 150 and 249. The average peak recognition rate increased by 0.27 points. The average recognition rate increased by 0.56 points for each result. In the context of this analysis, the null hypothesis (H0) assumes no difference in recognition accuracy before and after application, while the alternative hypothesis (H1) posits a significant improvement. Rejecting the null hypothesis at a 0.001 significance level provides evidence of a statistically significant positive impact on recognition accuracy post-application. This rejection supports the alternative hypothesis, indicating that the reuse method has improved recognition performance. Table 3.8 presents a list of results indicating reductions in recognition rates following the application of the Reuse method after epoch 150. In epochs 150 to 249 over ten trials, the recognition rate declined by 4 data points out of 1000, resulting in an average drop of 0.0825 points. The most significant decrease, 0.19 points, occurred in epoch 247 of Example 2, where the recognition rate fell from 95.78% to 95.59%. This decline is attributed to the rapid increase in the recognition rate in epoch 246, rising from 95.40%, which could not be effectively managed. The recognition rates for ten trials from epoch 0 to epoch 249 are listed at the end of the report. (Appendix 2)

Table 3.7. The ten comparison results of recognition rate for Bone and Proposed Bone across each training sessions with random initial conditions, including T-test outcomes.

	Peak Recognition Rate (Epoch 0 - 249)			Average of Top-1 (Epoch 150 - 229)			T-test for accuracy changes before and after applying Reuse method per epoch (Epoch 150 - 249)			
	Reuse method [%]		Improved	Reuse method [%]		Improved				
		✓			✓		MD	SDD	*t*-stat	*p* value
Ex.1	96.02	96.18	0.16	95.38	95.99	0.61	0.55	0.065	21.7	1.06^{-39} ***
Ex.2	95.78	95.86	0.08	95.33	95.70	0.37	0.31	0.049	14.2	7.16^{-26} ***
Ex.3	95.62	96.07	0.45	95.11	95.87	0.76	0.68	0.151	17.4	3.37^{-32} ***
Ex.4	95.83	96.10	0.27	95.25	95.91	0.66	0.59	0.121	17.1	1.49^{-31} ***
Ex.5	95.62	95.72	0.10	95.14	95.58	0.44	0.41	0.040	20.3	2.00^{-37} ***
Ex.6	95.62	95.88	0.26	95.20	95.68	0.48	0.44	0.037	22.8	1.50^{-41} ***
Ex.7	96.07	96.31	0.24	95.69	96.04	0.35	0.32	0.033	17.5	2.27^{-32} ***
Ex.8	95.70	96.02	0.32	95.27	95.81	0.54	0.49	0.044	23.6	9.97^{-43} ***
Ex.9	95.43	95.75	0.32	94.89	95.54	0.65	0.61	0.041	29.8	1.46^{-51} ***
Ex.10	95.38	95.91	0.53	95.02	95.76	0.74	0.73	0.063	29.1	1.33^{-50} ***
Avg.	95.71	95.98	0.27	95.23	95.79	0.56	0.51	0.082	56.6	5.69^{-314} ***

Note. MD = Mean of differences, SDD = Standard deviation of differences, $^{***}p < 0.001$

Table 3.8. Some results of recognition rate from ten trials of training Bone and Proposed Bone models with random initial conditions. * All data is displayed in Appendix 2.

Epoch No	Example 2		Example 7	
Reuse		✓		✓
224	95.64	95.72	**96.07**	96.04
232	**95.75**	95.72	96.02	96.07
247	**95.78**	95.59	95.99	96.10
249	**95.67**	95.59	95.83	96.10

3.3.3 Multi-stream SL-GCN

Multi-stream SL-GCN is a recognition method that uses four posture estimation features, Joint, Bone, Joint motion, and Bone motion, each of which is calculated using the evaluation value of the recognition result, using formula (2.1). In order to include the evaluation values of proposed Joint and Bone in the proposed method, two terms are added to the formula (2.1) as follows.

$$\begin{aligned} q_{Multi-stream} = \alpha_1 q_{Joint} + \alpha_2 q_{Bone} + \alpha_3 q_{JM} + \alpha_4 q_{BM} \\ +\alpha_5 q_{Proposed\ Joint} + \alpha_6 q_{Proposed\ Bone} \end{aligned} \tag{3.2}$$

For the recognition rate of Multi-stream SL-GCN, SAM-SLR gave 96.47% [10]. In our PC environment, the recognition rate was 96.82% when using the published code for SAM-SLR and α = {1.0, 0.9, 0.5, 0.5, 0.0, 0.0} used in that code. The highest recognition rate was 97.01% when using the proposed method proposed Joint, Bone with added evaluation values and weights α = {0.3, 0.6, 0.65, 0.35, 0.7, 1.1}. (Table 3.9) The application of the proposed method also improved the recognition rate for the majority of the epochs. For the average recognition rate for the 80 epochs from epoch 150 to 229 in our PC environment, the original method gave 96.38% and the proposed method gave 96.48%, 0.10 percentage points higher. (Fig. 3.6)

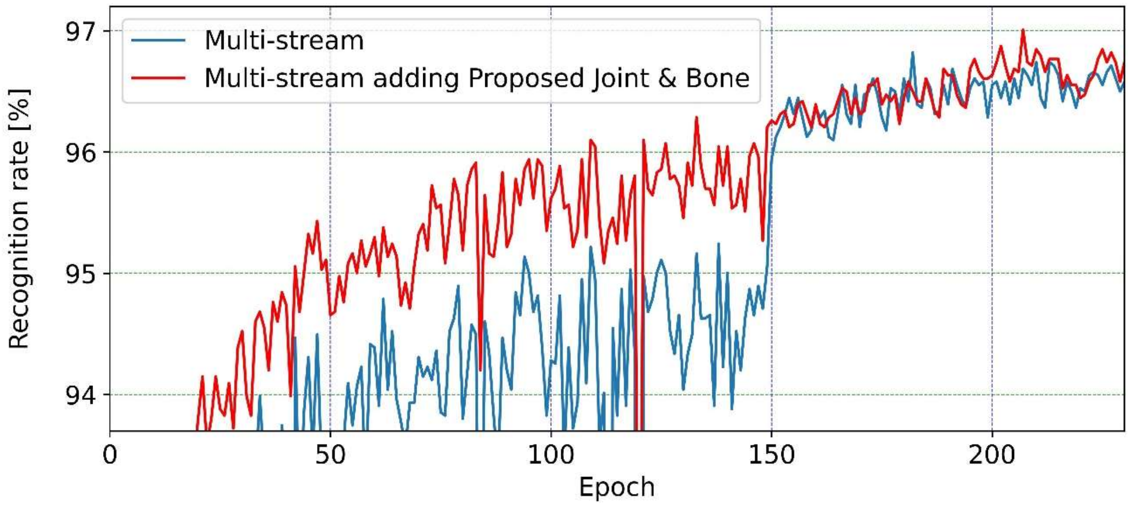


Fig. 3.6. Multi-stream evaluated using AUTSL test set.

Table 3.9. Performance our ensemble results evaluated on AUTSL Test set.

Modality	SAM-SLR-v2		Our PC	
	Top-1	Top-5	Top-1	Top-5
RGB Frames	95.00	99.47	95.27	99.57
RGB Flow	90.41	98.74	89.98	99.01
Features	93.37	99.28	94.41	99.44
Multi-stream	96.47	99.76	96.82 96.69	**99.87** 99.79
Multi-stream **+ Proposed Joint & Bone**	-	-	**97.01**	99.81
SAM-SLR-v2[10]	98.00	100	-	-
SAM-SLR [1]	97.62	100	97.94	100
SAM-SLR [1] **+ Proposed Joint & Bone**	-	-	**98.05**	100

3.3.4 Multi-ensemble Late-fusion Ensemble

Finally, we evaluated the estimate results of the four modalities with three different Late Fusion methods.

SAM-SLR-v2[10] (Global Ensemble Model)

SAM-SLR [1] (Model-free Late Fusion) (Our PC)

SAM-SLR [1] + Proposed Joint and Bone (Ours)

Table 3.9 shows the recognition rate results for each modality. RGB Frames, RGB Flow, Features, and Multi-stream are the recognition rates published in SAM-SLR-v2 [10] and the highest recognition rate up to Epoch 229 in Our PC. The top row of the Multi-stream column shows the recognition rate when α = {1.0, 0.9, 0.5, 0.5, 0.0, 0.0} and the bottom row when α = {0.0, 0.0, 0.55, 0.55, 1.0, 1.1}. (Table 3.10)

Table 3.10 and 3.11 show the Late Fusion weights used in the three models. There are two Late Fusion methods.

SAM-SLR-v2 is a method where each weight β is pre-trained and when the results for each modality are input, they are estimated according to the optimal weights. We have included graphs and recognition rates for SAM-SLR-v2 from the paper [1], as there does not appear to be any code available for this method. Also, the published code and Ensemble parameters for SAM-SLR using β ={1.0, 0.9, 0.4, 0.4} and the proposed method using *β*={1.0, 1.1, 0.1, 0.5}. (Table 3.11)

SAM-SLR-v2

For SAM-SLR-v2, we have visually generated the graph in Figure 3.7 from the paper, with a recognition rate of 98.00%. Especially after epoch 150, it has remained stable around 97.9%.

SAM-SLR

From the SAM-SLR-v2 document, the recognition rate was 97.62%. The graph in Figure 3.7 plots the recognition rate running on our PC using the published code. The highest recognition rate was 97.94%. (Table 3.9) The graph in Figure 3.7 shows that the recognition rate has remained around 97.8% since epoch 150.

SAM-SLR + Proposed Joint and Bone

The graph in Figure 3.7 shows the results of Joint and Bone estimation using the SAM-SLR code with *α* = {0.0, 0.0, 0.55, 0.55, 1.0, 1.1} for Multi-stream, instead of using the original Joint and Bone estimate results. We also used *β* = {1.0, 1.1, 0.1, 0.5} in the Ensemble parameters. The highest recognition rate was 98.05%. (Fig. 3.7 and Table 3.9)

Table 3.10. Multi-stream parameters *α*.

Models	Joint	Bone	Joint Motion	Bone Motion	Proposed Joint	Proposed Bone
SAM-SLR-v2[10]	1.0	0.9	0.5	0.5	-	-
SAM-SLR [1]	1.0	0.9	0.5	0.5	-	-
SAM-SLR [1] **+ Proposed Joint & Bone**	-	-	0.55	0.55	1.0	1.1

Table 3.11. Ensemble parameters *β*.

Models	Multi-stream	RGB Frames	RGB Flow	Features
SAM-SLR-v2[10]	Learning the ensemble weights			
SAM-SLR [1]	1.0	0.9	0.4	0.4
SAM-SLR [1] **+ Proposed Joint & Bone**	1.0	1.1	0.1	0.5

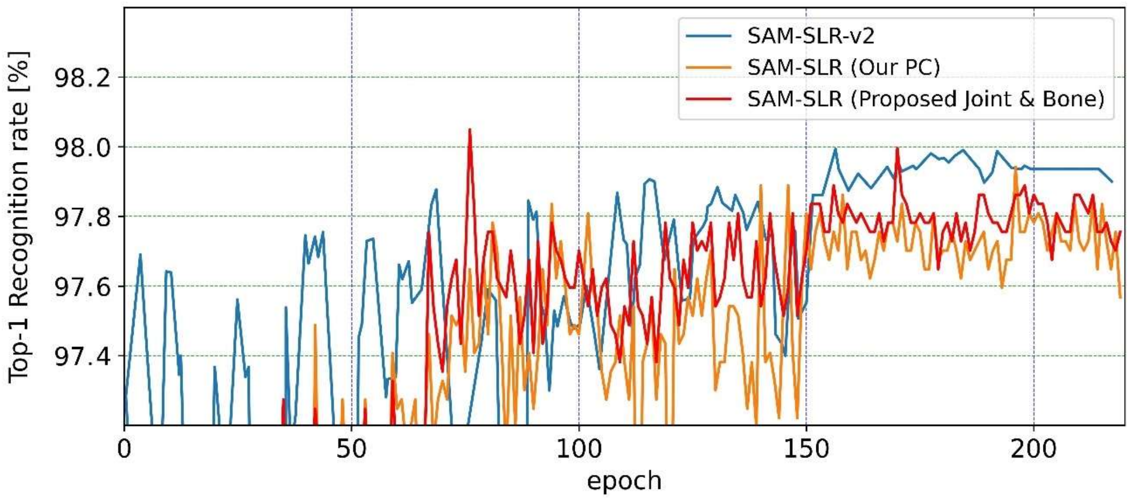


Fig. 3.7. Each SAM-SLR evaluated using AUTSL Test set.

3.4 Conclusion

In this study, we proposed a method to improve the recognition rate by using the training model of any epoch we want to use for recognition estimation and the estimate results from the training model created before that epoch. As a result, the average recognition rate from epochs 150 to 229 for the ten trials was improved by 0.49 points for Joints and 0.56 points for Bones compared to the original method. The method is also easy to implement because it reuses training models and estimate results that have already been created, so the computational complexity is very low and the calculation method is simple. The significance of this method is that it can contribute greatly to the improvement of the recognition rate when used alone, such as when there is no other modality, under the same conditions, but when the results of multiple modalities are enumerated, there may be competition with the parts recognized by other modalities, making it difficult to achieve significant improvement. However, when ensembling the results of multiple modalities, it is difficult to achieve a significant improvement because of the competition with other modalities' recognition. With the proposed method applied, Joint and Bone exceeded the recognition rate before application in almost all epochs. Late Fusion of the applied results with the results of other modalities resulted in a final recognition rate of 98.05%, exceeding the highest recognition rate of 98.00% for version 2 of SAM-SLR. In the future, we would like to come up with a new method for the Late Fusion part to exceed 98% for all epochs.

Chapter 4

Enhancing Sign Language Recognition through Re-Evaluation Method by Index Finger Position using Face Part Position Criterion

4.1 Introduction

It is said that there are 70 million people in the world who use sign language. According to official World Organization statistics for 2022 (http://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss accessed on 9 April 2023), approximately 430 million people (more than 5% of the world's population) will need rehabilitation to address their disabling hearing loss. It is also estimated that by 2050, nearly 2.5 billion people will have some degree of hearing loss, and at least 700 million will require hearing rehabilitation. Sign language recognition research has two main categories: isolated sign language and continuous sign language. The former are datasets of sign language videos ranging from 100 to 2000 words. Currently, the recognition rate for a vocabulary of 200 words exceeds 98%, which is getting close to the level of human recognition. We are working on the former and believe that increasing the recognition rate of isolated signs can be used in future studies of continuous signs. In the field of sign language recognition, various researchers have proposed systems with high recognition rates. Sign language recognition (SLR) has made significant progress in achieving high recognition accuracy in recent years with the development of practical deep learning architectures and rapid improvements in computing power. In sign language video recognition, a standard method is to convert video into a frame-by-frame in pre-processing, obtain position information of fingers, arms, nose, mouth, etc., by optical flow and posture estimation process for the image, and model spatiotemporal information using Three-Dimensional Convolutional Neural Network (3DCNN) [14] in the machine learning process of sign language recognition. In particular, models using posture estimation with MMPose [5], OpenPose [15], or MediaPipe [16] have been proposed and tend to have higher recognition rates [1, 10, 17–31]. Recently, there has also been an increase in restudies that use multiple recognition methods and fuse their results to further improve the recognition rate. In particular, the fusion of individual modalities has been performed recently to further improve the recognition rate.

In this study, we focus on an isolated SLR task and use RGB video from a dataset AUTSL [2] of Turkish Sign Language. This dataset was used in the competition used in the workshop and is readily available and easy to use. The baselines used in this competition were CNN, FPM [32], BLSTM [33], and model of attention, with a recognition rate of 49.23%. Sincan et al. summarized this competition's results [13]. The main results using the RGB dataset used in this study are shown below. The recognition rate results below also differ because the training data included training and validation data. In contrast, only training data were used in this study, with the former having higher recognition rates.

- Coster et al. proposed a model Video Transformer Network-Pose flow (VTN-PF) that provided posture information or hand geometry from RGB video data frame by frame to the Video Transform Network. They achieved a recognition rate of 92.92% [34].
- The Wenbinwuee team trained multiple models for RGB video recognition using RGB, optical flow, and person segmentation data, obtained the final prediction for each model using SlowFast, SlowOnly, and Temporal Shift Module (TSM), and fused the results, and obtained a result of 96.55% [14].
- The rhythmblue6 team proposed an ensemble framework consisting of multiple neural networks (Inflated 3D (I3D), Semantics-Guided Neural (SGN), etc.) and implemented the University of Science and Technology of China-Sign Language Recognition (USTC-SLR) model for isolated characters, 97.62% [14].
- Jiang et al. obtained 98.42% for Skeleton Aware Multi-modal Sign Language Recognition (SAM-SLR), 97.62% for the first version [1], and 98.00% for the second version [10] when trained on training data alone.

The main results published on the website of AUTSL Benchmark (https://paperswithcode.com/sota/sign-language-recognition-on-autsl Accessed on 9 April 2023) are as follows.

- Novopoltsev et al. proposed a real-time recognition system [30] using the Video Swin transformer [35] and Multiscale Vision Transformers (MViT) [36] models. They achieved a recognition rate of 95.72% and 2–3 predictions per second on CPU.
- Ryumin et al. proposed audio-visual speech recognition using spatio-temporal features (STF) and long-short term memory (LSTM) models. The model is especially characterized by the incorporation of lip information. They achieved a recognition rate of 98.56% and demonstrated a real-time process using mobile devices [31].

In addition to the above, the results published in the paper are as follows.

- Hrúz et at. analyzed 2 appearance-based approaches, I3D and TimeSformer, and 1 pose-based approach, SPOTER, which achieved recognition rates of 96.37% and 97.56% for test and validation datasets, respectively [26].
- Al-Hammadi et al.'s proposed architecture consists of a few separable Three-Dimensional Graph Convolution Network (3DGCN) layers, which are enhanced by a spatial attention mechanism. They achieved a recognition rate of 93.38% [27].
- We proposed a method to reuse the estimation results produced at each epoch based on SAM-SLR, which improved the recognition rate to 98.05% [29].

This section describes the SAM-SLR model used in the proposed methodology. It should be noted that this model was also used in the previously proposed paper [29], so the description is almost identical. In RGB video sign language recognition, there are four modalities: RGB-frames, RGB-flow, features, and Multi-stream, each of which independently performs sign language recognition and extracts features.

We will next talk about pre-processing, the four modalities, and late fusion.

- For pre-processing, the RGB stream uses the TVL1 algorithm [4], an OpenCV, and CUDA implementation of the Denseflow API provided by OpenMMLab [3], to extract the optical flow. The features and Multi-stream modalities used MMPose [5] to extract the pose estimation.

The four modalities are RGB-frames, RGB-flow, features, and Multi-stream, each of which independently performs sign language recognition and extracts features.

- The RGB-frames and RGB-flow modalities are modeled in a 3DCNN [14] using the ResNet2+1D [11] architecture, which separates the temporal and spatial convolution of a 3DCNN. The model chooses the Res-Net2+1D-18 variant for its backbone, which is pre-trained on the Kinetics dataset [12]. In addition, it is pre-trained on the most extensive available SLR dataset SLR500 [22] for the RGB frame to further improve the accuracy.
- A separate spatial–temporal convolutional network (SSTCN [1]) was developed to learn from the entire skeleton to fully extract information from key points throughout the body.
- A Multi-stream sign language graph convolutional network (SL-GCN [1]) was designed to model the embedding dynamics using the whole-body key points extracted by the pre-trained whole-body posture estimator. The estimated results and weights of the Joint, Bone, Joint motion, and Bone motion streams were multiplied and used as the evaluation value of the Multi-stream modality. The highest recognition rate among the modalities was achieved by Multi-stream, which had a recognition rate of 96.47%.

- There are two versions of late fusion.
- The first version [1] proposes ensemble model-free late fusion, a simple late fusion approach that fuses predictions from all modalities. All modalities were manually weighted using {1.0, 0.9, 0.4, 0.4}. The recognition result was 97.62%.

In the second version of SAM-SLR [10], a learning global ensemble model was proposed because finding the optimal weights for fusion is time consuming. The method was pre-trained in a neural network with the score of each modality as input and each weighted as output. The recognition result was 98.00%. For the other datasets, the recognition rates were also high for WLASL2000 [37] after pre-training on SLR500 [22] and the BSL-1K [38] dataset (Table 4.1).

Table 4.1. Summary of isolated sign language datasets.

Datasets	**Language**	**Glosses**	**Signers**	**Samples**
AUTSL [2]	Turkish	226	43	36,302
SLR500 [22]	Chinese	500	50	125,000
WLASL2000 [37]	American	2000	119	21,083
BSL-1K [38]	British	1064	40	273,000

The AUTSL dataset used in this study already achieved a recognition rate of over 98% in 2021. Thus, datasets and models, which are already in a high recognition rate, are in a situation where the recognition rate cannot be easily improved. The test data for this dataset contain datasets that are difficult for humans to judge, so it may be difficult to answer all of them correctly. In addition, a report on the recognition rate of the validation set for this dataset showed that the percentage of samples correctly recognized by at least one model was 98.96% [26], suggesting the possibility of eventually achieving recognition rates of around 99%. We proposed a re-evaluation method to further improve the recognition rate. In 2023, before our proposal, Ryumin et al. proposed a model to reach 98.56% [31].

In our approach, we decided to re-evaluate the results from Top-1 to Top-3, noting that the recognition rate tends to be lower when the difference between the Top-1 and Top-2 scores among the recognition results of the first version of SAM-SLR is slight.

The evaluation method was based on the fact that although the positions of facial parts such as eyes, nose, and mouth differ from person to person, by creating a triangle with the facial parts as vertices and connecting the vertices, the coordinates of the index finger of the dominant hand can be converted to the coordinates inside the triangle. This method is based on the fact that the position of the finger can be captured from the position of the facial parts and compared with four locations where the finger is considered stationary.

Our main contributions can be summarized as follows. Based on the results of the SAM-SLR evaluation, we re-evaluated only the data with low recognition rates to maintain the recognition rate. The proposed recognition method, which is not used in SAM-SLR, independently reflects the recognition rate and can achieve a higher recognition rate than that of SAM-SLR. Since the information converted from the finger position information to the face part coordinates from the training data is stored in advance and only converted to face part coordinates from the test data at the time of re-recognition, the computation is reduced. We will report comparative results when the proposed method is applied to each of the first versions of SAM-SLR (hereafter referred to as SAM-SLR) and the previously proposed model. This method can be applied to machine learning in the future.

4.2 Methodology

4.2.1 Dataset

The AUTSL [2] dataset was created for sign language recognition competitions and contains Turkish Sign Language with 36,302 video files each of RGB and depth video (Table 4.2). This dataset is publicly available on the Internet and can be easily obtained by anyone. The videos are all 512 × 512 in size, with the signer standing or sitting almost in the center. Only RGB video is used.

Table 4.2. A statistical summary of the AUTSL [2] Dataset. ([29])

Dataset	**Signs**	**Signers**	**Language**	**Frames**	**Samples**			
					Training	**Validation**	**Testing**	**Total**
AUTSL	226	43	Turkish	57–157	28,142	4418	3742	36,302

4.2.2 Architecture of SAM-SLR

SAM-SLR uses four methods from the sign language videos, including optical flow, 3DCNN, and posture estimation, and synthesizes the results of each estimation to achieve a high recognition rate. On the Turkish sign language dataset, the first version achieves a recognition rate of 97.62%, and the second version achieves a recognition rate of 98.00%. In addition, the method we proposed in the previous study (Fig. 4.1), which reuses the estimation results of the training models created in each of the previous epochs of Joint and Bone, 2 streams used in the Multi-stream modality, and one of the modalities of SAM-SLR, improved the recognition rate to 98.05% [29]. However, a higher recognition rate could not be achieved. Therefore, a new recognition method is needed to seek a higher recognition rate, and we have three perspectives on the new recognition method. The first perspective is a method to re-evaluate the 2% portion that is not recognized from the recognition results of the first version of SAM-SLR (hereafter referred to as SAM-SLR). The second perspective was that some characters touching the mouth, eyes, nose, etc., could not be recognized because the position of the fingers could not be accurately detected. The third perspective detects those index fingers that stay longer near each facial part from the training information in the dataset, captures their relative position, creates a heatmap for each sign language, and compares it with the relative position of the index finger in the test data for recognition processing.

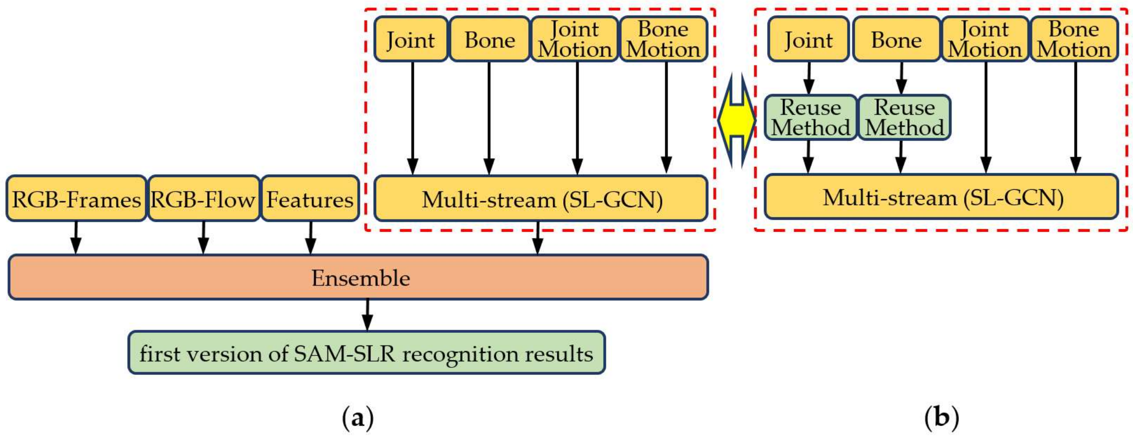


Fig. 4.1. Concept of skeleton aware multi-modal sign language recognition framework (SAM-SLR) (RGB version). (a) Original [1, 10] and (b) Last ours, our proposed Joint and Bone streams in the Multi-stream modality [29].

4.2.3 Overview of the Re-Evaluation Method

The flowchart diagram (Fig. 4.2) illustrates the overall process of the re-evaluation method. First, as a pre-processing step, all sign language video data in the dataset are automatically flipped and aligned so that the dominant hand is on the right. Next, the input data are re-evaluated if they meet 3 conditions (the difference between the Top-1 and Top-2 evaluation values from the SAM-SLR recognition results is within 2.0, the sign language is one-handed, and there are 4 index finger position data in the face area of the test data). Finally, the results of this evaluation are replaced with the SAM-SLR results. This set of procedures was automatically determined by the amount of movement of each left and right finger when judging one-handed signs, and any outlier information in the posture estimation was excluded from obtaining the correct index finger position information.

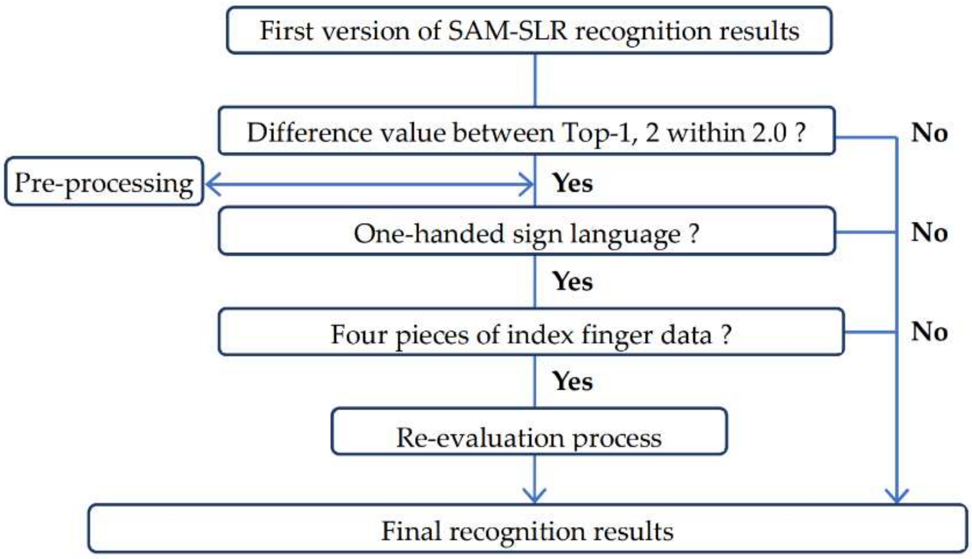

First version of SAM-SLR recognition results
Difference value between Top-1, 2 within 2.0 ?
No
Pre-processing
Yes
One-handed sign language ?
No
Yes
Four pieces of index finger data ?
No
Yes
Re-evaluation process
Final recognition results

Fig. 4.2. Flowchart of the re-evaluation process. Example: difference value is within 2.0, one-handed sign language and 4 pieces of finger data.

4.2.3.1 Difference Value between Top-1 and Top-2

For the first method, Table 4.3 shows the difference between the Top-1 and Top-2 scores and the percentage of correct responses for the test data of the training model at epoch 196 of the SAM-SLR. Note that the maximum value of this difference was up to 14.0. This table shows that a low percentage of correct answers characterizes those with a slight difference between the Top-1 and Top-2 scores. This is because the sign languages in these classes are considered to be similar and difficult to distinguish. If some recognition methods from a different perspective than the one used in SAM-SLR could be used, the current recognition rate could be exceeded. In particular, with a difference of 2.0 in the evaluation value, the recognition rate is 68.94% (Table 4.3). Hypothetically, if we could achieve a recognition rate of about 75% with a new method in the 132 tests data, we would be able to increase the number of correct answers in the test data by about 6%, which would increase the number of correct answers by about 8 data equivalents. As a result, the number of correct answers in the SAM-SLR would increase from 3665 to 3673, and the overall recognition rate would reach 98.16%.

Table 4.3. Value of difference between Top-1 and Top-2 on the first version of SAM-SLR results.

Value of Difference	**Number of Matched**	**Number of Correct**	**Number of Incorrect**	**Top-1 Acc (%)**
1.0	64	38	26	59.38
2.0	132	91	41	68.94
3.0	226	171	55	75.66
…				
9.0	1984	1907	77	96.12
…				
14.0	3742	3665	77	97.94

Conversely, suppose the recognition rate of this method is about 75%. In this case, the recognition rate for a difference of 3.0 in the evaluation value is as high as 75.66%, so it is expected that the recognition rate will decrease when using this method, and the more significant the difference in the evaluation value, the lower the recognition rate. Therefore, we will consider the case where the difference in the evaluation values is 2.0.

Regarding the range of classes to be re-evaluated, the correct answers for Top-1, Top-2, and Top-3 were 3665, 3727, and 3739, respectively. The recognition rates were 97.94%, 99.60%, and 99.92%, respectively, and since the range up to Top-3 can almost cover the whole range, the scope of the re-evaluation was limited to Top-3.

4.2.3.2 One-Handed or One- or Two-Handed Sign Language

The default for the second conditional branch in the flowchart was set to one-handed sign language. The two reasons for this are that the re-evaluation method is aimed at sign language that touches parts of the face area using the index finger, many of which require one-handed sign language, and to improve the recognition rate. In Chapter 3, we examine one-handed or one- or two-handed sign language cases.

4.2.3.3 Number of Index Finger Data

The default for the last conditional branch in the flowchart is 4 samples of index finger position information. That is because a small sample size will include sign language in which the index finger just passes through the face area while setting a larger sample size will reduce the number of test data that meet the criteria. In Chapter 3, the number of samples of this index finger is verified from 2 to 7.

4.2.4 Index Finger Position in Face Area Using Face Part Position Criterion

The second method is to touch parts of the face with the index finger. OpenPose [15] is used for posture estimation. The position of the parts of the face varies from person to person, such as the size of the face or the elongation. In order to absorb this difference in position, a skeleton (Fig. 4.3) composed of triangles is created with each facial part as a vertex, and the number of the triangle in which the index finger is located and the position of the index finger within that triangle are stored. This allows for more accurate position comparisons. To capture the relative facial structure, we used the positional information of the face's eyes, nose, mouth, cheeks, and chin used in SAM-SLR (yellow dots) to create a new vertex (green dots) to fit the contour of the face. The upper vertex was stretched to 2.5 times its height when the height from the midpoint of the cheek edges and eyes on each side to the intersection with the perpendicular line was set to 1. The lower vertex was further stretched to the same height when the height to the intersection of the cheek edges and the perpendicular line from the chin was set to 1. The outer left and right vertex positions were created from the left and right cheek vertexes and the chin vertex, with their lower vertexes twice as high as the diagonals perpendicular to each intersection of the cheek edges and the chin. To capture the relative position of the fingers, a mesh of 17 triangles was created using these vertices.

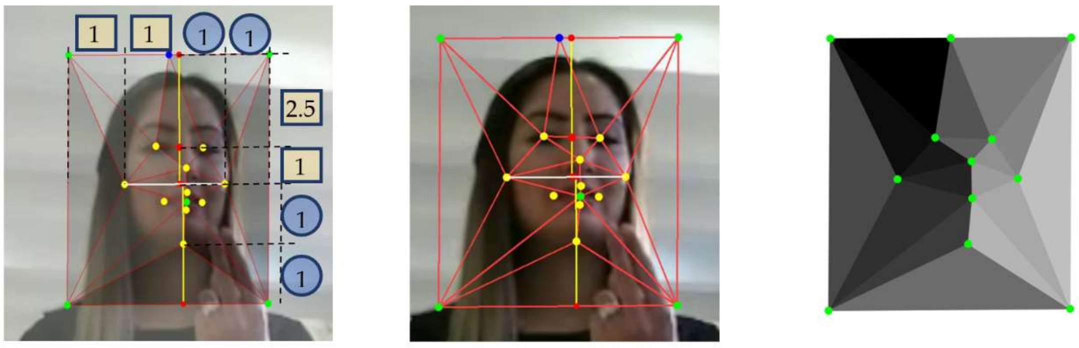


Figure 4.3. Structure of face triangle mesh.

Figure 4.4 shows the sign language with the index finger pointing to the mouth, but in the top row, the positioning of the index finger is very different. In the lower row, the coordinates of the index finger are recorded on the triangle mesh based on the facial parts, and when the coordinates within the triangles (red line) of the trajectory data (b) are transformed affine to the coordinates within the triangles (blue line) of (a) and (c), the positions of each index finger are very close.

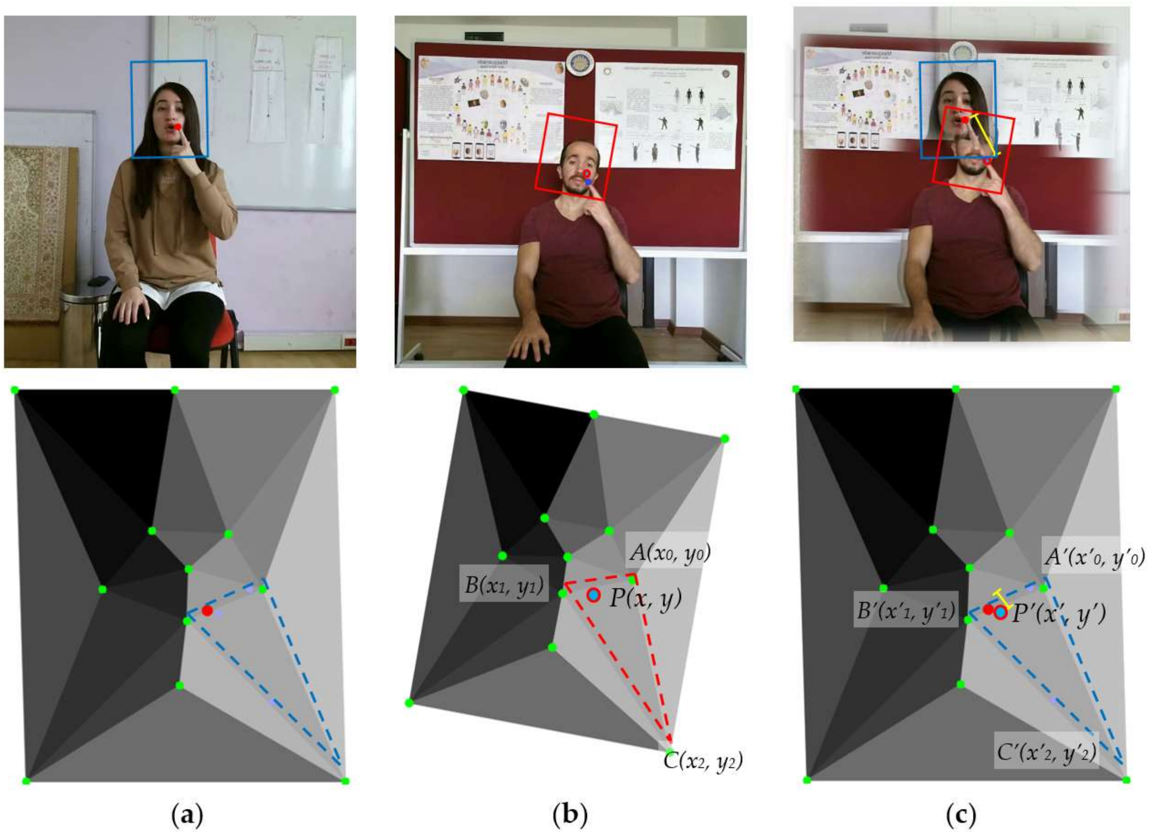


Fig. 4.4. Improvement of index finger position in face area using face part position criterion from the AUTSL [2] dataset: (a) test data, (b) training data, and (c) distance differences of the positioning of the index finger.

4.2.5 Process of Staying Fingertip Decision

We will now discuss the decision about the object to be recognized using the new method. As mentioned above, the target should have a slight difference between the Top-1 and Top-2 values, be a one-handed sign language, have no outliers, and have all four pieces of finger information in the face area. One-handed sign language is used for signs expressed by touching parts of the face with the index finger. The reason for the "no outliers" mentioned above is that if there are outliers, the structure of the triangular network will be significantly disrupted when it is created. The reason for the outliers is that the position of a hand in the face area cannot be detected accurately because the hand hides the face. In addition, the method that uses the positions of the cheeks, chin, and eyes to capture the facial structure cannot process the face correctly because of the collapse of the facial contours.

In this study, we examined the index finger within the face region to capture the state of the index finger within the face region, which is the smallest distance the index finger moves within the face region. The method was to sum the distance traveled between three frames, one before and one after each frame, and those with the shortest distance traveled were judged to have stayed longer in that location. We decided to extract the Top-4 shortest travel distances in the order of shortest travel distance. Fingers in the three frames before and after the adopted t, i.e., seven frames, were judged to belong to the location at the adopted t and excluded from subsequent adoption decisions.

$$\sum_{t=i-1}^{i=2} \sqrt{(x_{t+1} - x_t)^2 + (y_{t+1} - y_t)^2} \tag{4.1}$$

Figure 4.5 shows the motion of the index finger, with the red dot indicating the adopted location. The Table 4.4 on the right shows the actual position information of the index finger and the total distance traveled. The yellow cells show the immediately before the adopted area that cannot be adopted again, the red letters show the actual adopted location, and the light blue cells show the second candidate area. The second possible candidate location is the light blue area from t = 14...17, 25...32, where t = 18...24 is excluded because 21 was adopted just before. The shortest, t = 17, was accepted at 3.2361. The last matches are 21, 17, 29, and 25.

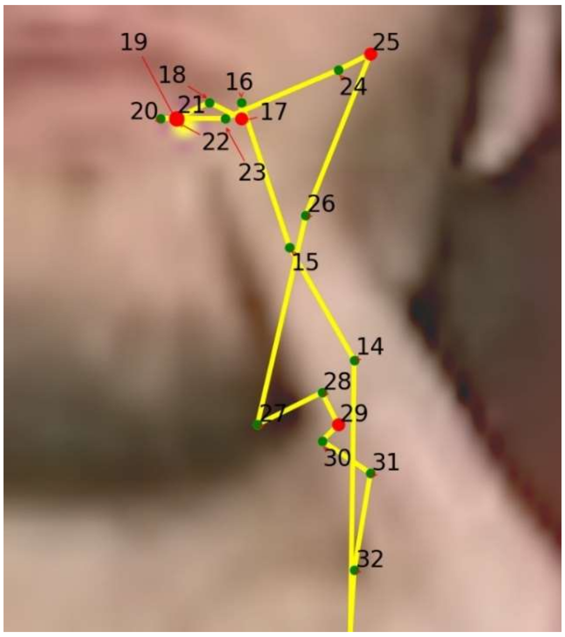


Fig. 4.5. Example of moving fingertip from the AUTSL [2] dataset. (Each number at the fingertip in the figure is the frame number.)

Table 4.4. Process of staying fingertip decision.

t	x	y	d_i	1st	2nd	3rd	4th
14	266	130	13.0064		−3		
15	262	123	17.5491		−2		
16	259	114	10.4868		−1		
17	259	115	3.2361		0		
18	257	114	4.4721	−3	1		
19	255	115	3.2361	−2	2		
20	254	115	2.0000	−1	3		
21	255	115	1.0000	0			
22	255	115	3.0000	1			-3
23	258	115	10.6158	2			-2
24	265	112	9.8518	3			-1
25	267	111	13.0064				0
26	263	121	24.1120			−3	1
27	260	134	17.8138			−2	2
28	264	132	6.7082			−1	3
29	265	134	3.6503			0	
30	264	135	5.0198			1	
31	267	137	9.6883			2	
32	266	143	53.1784			3	

4.2.6 Re-Evaluation Process

The Top-4 coordinates at which the index finger remained within the face region are recorded for training and test data. An affine transformation matrix was obtained to transform the coordinates of the training data collected for each class into the coordinates corresponding to the Top-1 face part coordinates of the test data. The variables in the following equation are the coordinates A(x_0, y_0), B(x_1, y_1), and C(x_2, y_2) of the triangles corresponding to the coordinates of the training data, and the triangles of the test data and their coordinates A'(x'_0, y'_0), B'(x'_1, y'_1) and C'(x'_2, y'_2) corresponding to the triangles of the training data.

$$\begin{pmatrix} x'_0 & x'_1 & x'_2 \\ y'_0 & y'_1 & y'_2 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ 1 & 1 & 1 \end{pmatrix} \tag{4.2}$$

The equation for finding the coordinates using the affine transform matrix:

$$\begin{aligned} \hat{x} &= dx + by + c \\ \hat{y} &= dx + ey + f \end{aligned} \tag{4.3}$$

To create the histogram data, the image area of $\hat{x}$ and $\hat{y}$ was 480 in height and width, and the bin was 300 in the histogram.

$$X = \frac{\text{width}}{\text{bins}} \hat{x},\ Y = \frac{\text{height}}{\text{bins}} \hat{y} \tag{4.4}$$

Histogram equation:

$$Histogram(X,\ Y) = \frac{1}{n} \sum_{i=0}^{fingers(\hat{x},\ \hat{y})} H(X,\ Y) \tag{4.5}$$

After creating a histogram using the data $H(X,\ Y)$ transformed for the histogram, the data were divided by the amount of data n after tabulation to equalize each class's different amounts of data.

Gaussian filter equation:

$$g(x,\ y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)} \tag{4.6}$$

Histogram equation:

$$Heatmap(X,\ Y) = Histogram(X,\ Y) \cdot g(X,\ Y) \tag{4.7}$$

There are two types of heatmaps: absolute scores (Fig. 4.6, top) and relative scores (Fig. 4.6, bottom).

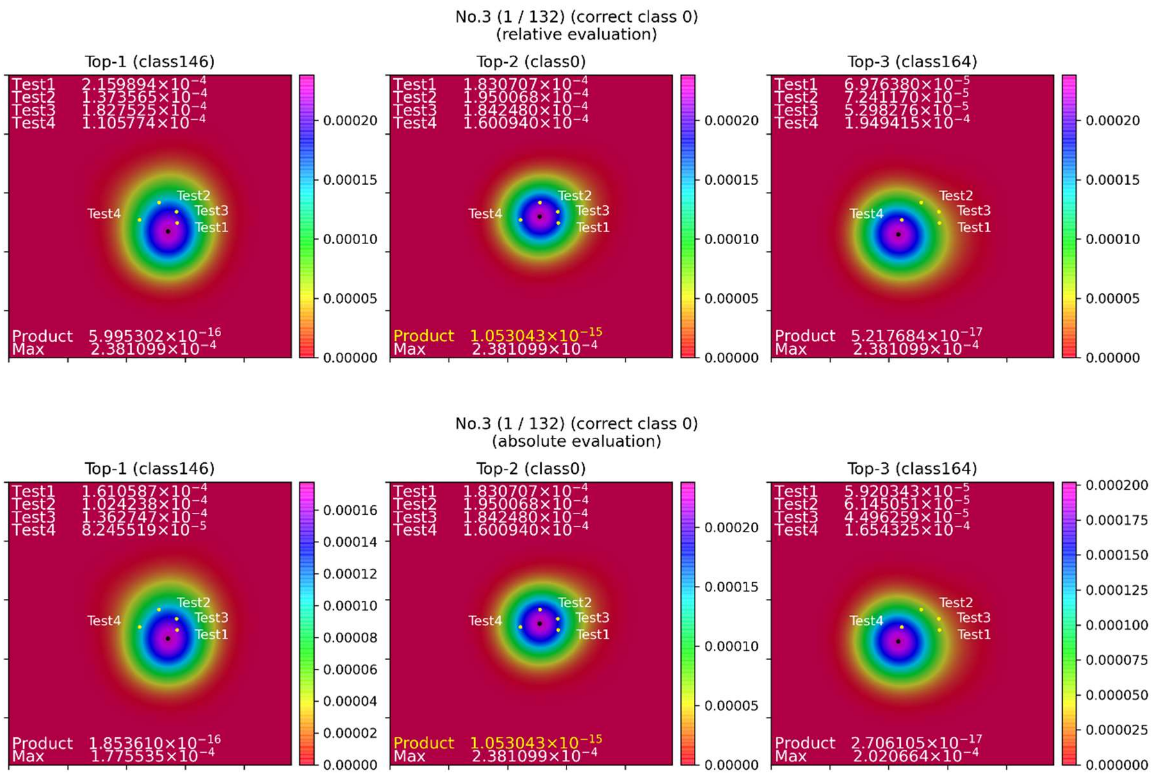


Fig. 4.6. Example of absolute and relative evaluation heatmaps. (The heatmap with the highest 'Product' in yellow is ultimately chosen.) * All data is displayed in Appendix 5.

The heatmap uses this equation, and sigma applied a Gaussian filter of 24.

The absolute product equation:

$$Absolute\ Product_{Top\text{-}n} = \Pi Heatmap(X,\ Y)_{Top\text{-}n} \tag{4.8}$$

For the absolute rating heatmap, we created a heatmap of the top 1, 2, and 3 histogram data for each class candidate. We obtained the product of the heatmap values at the four coordinates of the test data.

The relative product equation:

$$High\ Absolute\ Product_{Top\text{-}n} = Max\big(\Pi Heatmap(X,\ Y)_{Top\text{-}n}\big)$$

$$rate_{Top\text{-}n} = \frac{Max\left(High\ Absolute\ Product_{Top\text{-}1,\ 2,\ 3}\right)}{High\ Absolute\ Product_{Top\text{-}n}} \tag{4.9}$$

$$Relative\ Product_{Top\text{-}n} = \Pi Heatmap(X,\ Y)_{Top\text{-}n} \cdot rate_{Top\text{-}n}$$

The relative rating value heatmap was created by multiplying the highest absolute rating value in the Top-1, 2, and 3 classes by the highest value in each class so that the highest value in each class is the same. We obtained the product of the heatmap values at the four coordinates of the test data. The plotted positions of Test1 to Test4 are the positions of the index fingers of the test data in order of most extended stay, and each value is the heatmap value of the training data at that position.

There are three scoring methods: The first and second are based on the product of the values of the four finger positions of the test data in the heatmap of the finger positions of the training data made for each class, using the heatmap of absolute evaluation values for the former and the heatmap of relative evaluation values for the latter. The third method is an evaluation method that integrates these two heatmaps, where the first place is given three points, the second place two points, and the third place one point, and each class is added up, and if there is a tie for the first place, the class with the best relative evaluation value is selected.

Figure 4.6 shows the heatmap output for Top-1 to Top-3, where the correct answer for a given test data is class 0. From left to right, the SAM-SLR recognition results show that Top-1 is class 146, Top-2 is class 0, and Top-3 is class 164. The absolute evaluation of the top row shows that the product of Top-2 has the highest value, and Top-2 is selected as the re-evaluation result. The black dots in the heatmap's center indicate the maximum value's location. In the relative evaluation at the bottom, the highest value of the maximum relative evaluation for each class is 2.381099×10^{-4}, which is the maximum value of Top-2, so each rate in Equation (7) is multiplied on each heatmap so that the maximum value for each class is this value. This results in the values of Top-1 and Top-3 being 5.995302×10^{-16} and 5.217684×10^{-17}, respectively, and the value of the product of Top-2 being 1.053043×10^{-15}, which is the same as the value of the absolute score. Since this value is the highest, Top-2 is selected as the re-evaluation result.

4.3 Results

In this section, we present the results of re-evaluation experiments based on the models (1) SAM-SLR and (2) SAM-SLR with the last our model.

4.3.1 PC Environments

We used the published code used in SAM-SLR [1], created training models for each method on our PC, used test data, and produced estimated results. The development environment and evaluation methods are based on the same criteria as the previously proposed paper [29]. Our PC specs are AMD 3960x (CPU), RTX-3090 (3 GPUs), 128 GB (RAM), AMD 3950x (CPU), RTX-2080Ti (GPU), 64GB (RAM), and Ubuntu 18.04 (OS).

4.3.2 Summary of Results

The test data consist of 3,742 RGB videos, each video containing 1 of 226 different signs. The evaluation method outputs an estimation result for 226 different signs per video. For the evaluation method of recognition rate, the number of correct answers is counted as Top-1 when the correct answer is the highest in the evaluation value for that class.

$$\textit{Recognition rate of Top-1} = \frac{\textit{Total number of Top-1}}{3742} \tag{4.10}$$

Each parameter of the heatmap used in this study has a histogram interval of 300 and a heatmap sigma of 24. There are four scoring models: SAM-SLR, the previously proposed model (Last ours), and Ours-1 and Ours-2, which add the method proposed here to these two models. The recognition rates were 97.94%, 98.05%, 98.24%, and 98.21%, respectively (Table 4.5). Last our model is a method to improve the recognition rate of SAM-SLR by reusing the evaluation values of the training models generated at each epoch of the Joint and Bone streams, which are part of the Multi-stream model using the SAM-SLR posture estimation. When the data and the PC used were run in the same environment, including SAM-SLR, the recognition rate of the proposed method Ours-1 showed the highest value, 0.3 points higher than SAM-SLR, and 0.21 points higher than the second version of SAM-SLR in the paper.

Table 4.5. Performance of our re-evaluation results (with and without fine-tuning using the validation set) evaluated on the AUTSL test set.

Model	Fine-Tune*	Publication	Our PC
Baseline [2]	-	49.22	-
VTN-PF [34]		92.92	-
Enhanced 3DGCN [27]	No	93.38	-
MViT-SLR [11]	-	95.72	-
Wenbinwuee team [13]	Yes	96.55	-
Neural Ens. [26]	No	96.37	-
USTC-SLR [13]	Yes	97.62	-
Second version of SAM-SLR [10]	No	98.00	-
STF + LSTM [31]	-	98.56	-
SAM-SLR [1]	No	97.62	97.94
+ Last ours (proposed Joint and Bone streams)[29]	No	-	98.05
+ Ours-1 (re-evaluation method)	No	-	98.24
+ Ours-2 (Last ours and re-evaluation method)	No	-	98.21

* With validation data

The Table 4.6 shows the results of the recognition rates for absolute, relative, and combined absolute and relative ratings and the average recognition rate at epochs 150 to 229 using the 2 models, Ours-1 and Ours-2. The order of the highest recognition rates was higher for the integrated, relative, and absolute scores, and the model was slightly higher for Ours-1. For the integrated score, the results were 98.24% and 98.21% for each model, respectively. In that order, the average recognition rate was also higher for the integrated, relative, and absolute scores, with a slightly higher rate for the Ours-2 model. For the integrated score, the results were 98.00% and 98.04% for each model, respectively. Compared with the results of the second version of SAM-SLR, created visually from the paper but only as a reference due to different PC environments, the last proposed method (Last ours) achieved 98.05%. However, the average recognition rate of the second version compared with the graph was better. On the other hand, due to improvements, Ours-1 achieved the highest recognition rate of 98.24% and Ours-2 had the highest average recognition rate of 98.04%. Moreover, focusing on the point of concern in this graph, there are times when the recognition rates of SAM-SLR and Ours-1 drop significantly, about 6 times between epochs 80 and 150, but there are also times when the model using posture estimation drops significantly at the same time, which could be the cause. Applying the previous model improved this significant drop, and the Ours-2 with the Last our model applied remained stable and high. In the graph, the recognition rate improves dramatically from epoch 150. This is because the posture estimation model used in SAM-SLR is set to suppress the learning rate to one-tenth of the standard rate when learning from epochs 150 to 200, contributing to the improved recognition rate. Therefore, we started evaluating the average recognition rate at epoch 150. (Fig. 4.7)

Table 4.6. Performance of our re-evaluation results evaluated by the evaluation method on the AUTSL Test set.

Based Model	**Re-evaluation Method (Finger position evaluation)**	**Acc.**	**Average of Acc. From epoch 150 to 229**
SAM-SLR [1] (Ours-1)	-	97.94	97.73
	Absolute	98.05	97.82
	Relative	98.16	97.92
	Absolute and relative	98.24	98.00
SAM-SLR with proposed Joint and Bone streams [29] (Ours-2)	-	98.05	97.79
	Absolute	98.00	97.82
	Relative	98.13	97.96
	Absolute and relative	98.21	98.04

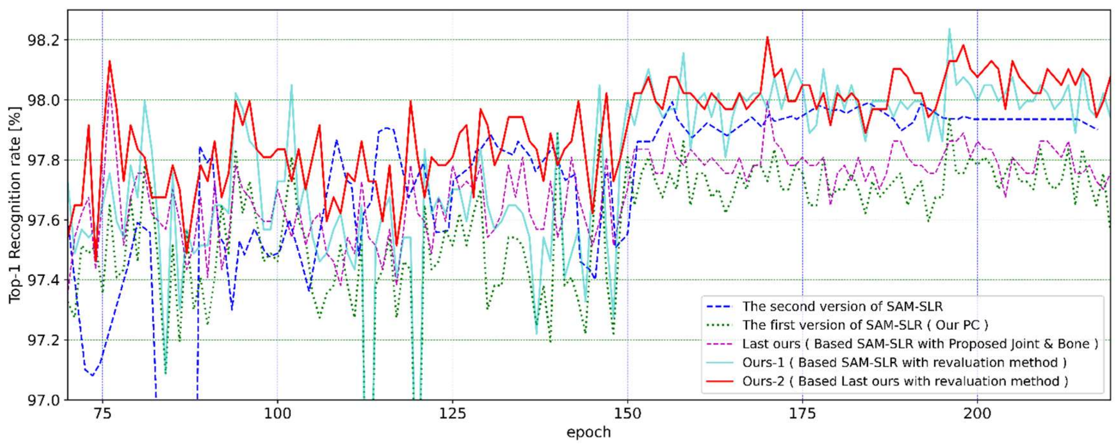


Fig. 4.7. Performance of our re-evaluation results from each epoch evaluated by models on the AUTSL test set. (The second version of SAM-SLR, the first version of SAM-SLR, and Last ours [29].)

4.3.3 The Performance of Each Conditional Branch

The recognition rate before introducing this method was 97.94%, and after the introduction, the recognition rate improved except the rate for conditional branches on one or two-handed sign language and 3 pieces of index finger data was 97.92%. The average recognition rate for these cases was 98.09%, an improvement of 0.15 percentage points. In addition, the maximum recognition rate was 98.24% for one-handed sign language, with 4 pieces of index finger dataset as the default for the conditional branch.

For the first conditional branch, regarding the difference value between Top-1 and Top-2, we predicted in the previous section that a difference of 2.0 would be optimal. The average recognition rate before the introduction was 69.94%, but from Table 4.7, it can be observed that the average recognition rate for the entire test data after the re-evaluation was 72.98% and the high recognition rate was 77.27%.

Table 4.7. Performance of absolute and relative re-evaluation results evaluated depending on one-handed sign language or one- or two-handed sign language and the number of finger data by the evaluation method on the AUTSL test set.

	Number of Index Finger Data					
	2	3	4	5	6	7
	One-handed sign language					
Total number	132	132	132	132	132	132
Number of correct	98	95	102	99	97	95
Correct rate (%)	74.24	71.97	77.27	75.00	73.48	71.97
Number of re-evaluations	41	41	35	30	20	13
Number of correct	30	27	29	22	17	12
Correct rate (%)	73.17	65.86	82.86	73.33	85.00	92.31
Final recognition rate (%)	98.13	98.05	98.24	98.16	98.10	98.05
	One- or two-handed sign language					
Total number	132	132	132	132	132	132
Number of correct	93	90	99	98	96	94
Correct rate (%)	70.45	68.18	75.00	74.24	72.72	71.21
Number of re-evaluations	54	51	42	33	21	14
Number of correct	35	30	32	23	17	12
Correct rate (%)	64.81	58.82	76.19	69.70	80.95	85.71
Final recognition rate (%)	98.00	97.92	98.16	98.13	98.08	98.02

For the second conditional branch, one-handed sign language, the recognition rate improves when introduced from the beginning since many sign language signs that touch parts of the face area using the index finger are one-handed sign language signs. In the case of one-handed sign language, the average was 98.12%, while in the case of one or two-handed sign language, i.e., without this decision, the average was 98.05%. In addition, the number of re-evaluated cases averaged 30.0 for one-handed sign language, while it was 35.8 for one or two-handed sign language, with the latter having 10 more cases.

The third conditional branch, the number of index finger data, was higher, with an average of 98.20% when there were 4 samples and 97.99% when there were 3 samples. When looking at the re-evaluated cases, the number of cases and their recognition rate averaged 47.5 cases and 68.42% when there were 2 samples. At the same time, 13.5 cases and 88.89% were recognized when there were 7 samples. When the number of samples was small, the number of cases increased, and the recognition rate was low, while when the number of samples was large, the number of cases decreased, and the recognition rate tended to be high.

4.3.4 The Performance of Each Epoch

The graph in Figure 4.8 shows the cases where the difference between the Top-1 and Top-2 evaluation values is within 2.0 using the Ours-1 method, the number of cases retested, and the percentage of correct answers for each. The number of cases within 2.0 and the number of cases resurveyed at epoch 0 were 3557 and 651, respectively, and the number gradually decreased, averaging 136.8 and 39.1, respectively, from epoch 150 to epoch 229. The percentage of resurveyed cases averaged 26.87% for epochs 0 to 229. Although some cases were excluded for outliers or other reasons, the percentage of signs touching the face in one-handed sign language is just under 30%. Regarding the percentage of correct answers in the resurvey and the percentage of correct answers within 2.0, the averages from epochs 150 to 229 were 79.8% and 73.3%, respectively. Within 2.0 was the best for this method since a difference in scores greater than 2.0 could result in a lower recognition rate. In addition, since about 70% of the areas were still not re-examined, the recognition rate could be improved in the same way by trying another analysis method using the re-evaluation method proposed in this study.

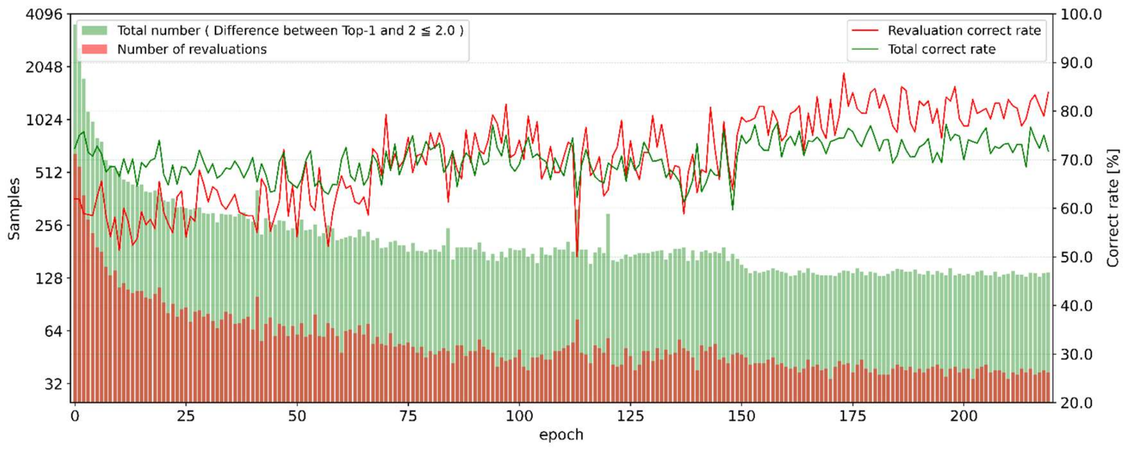


Fig. 4.8. The difference between the Top-1 and Top-2 evaluation values is within 2.0, the number of the re-evaluation and the percentage of correct answers for each.

4.4 Conclusion

In the previous method, we proposed a method to reuse the results of the recognition estimation of the training model, which reached 98.05%. Compared with the SAM-SLR, the results were generally good in the recognition rate of each epoch after epoch 150. However, there were few cases exceeding 98%; therefore, in this study, we proposed a method to accurately capture the location of the index finger in the face region to improve the recognition rate. As a result, we found that the difference between Top-1 and Top-2 of the SAM-SLR estimation results was within 2.0. In the method of re-evaluation when four location data of the stopped finger could be obtained, the recognition rates of the SAM-SLR-based method and the previously proposed method were, respectively, 98.24% and 98.21%, with the former being better and improving by up to 0.3, 0.24, and 0.19 percentage points, respectively, when compared with the first version and the second version of SAM-SLR, and the previous method. The average recognition rate from epochs 150 to 229 was 97.79% for the previously proposed method and 98.00% and 98.04 for the proposed former and latter, respectively, improving by 0.21 and 0.25 points, respectively. Since this method was relatively independent of the SAM-SLR recognition methods, it is thought that it could reflect the recognition results without being affected by other modalities.

- The recognition rate with slight differences in top-1 and top-2 evaluation values in the SAM-SLR recognition results is low (e.g., 75% or less).
- The recognition rate up to Top-3 is close to 100%.
- In the face area, no special recognition processing was performed.

Our re-evaluation method can process the re-evaluation by an average of 0.101s per video using CPU when setting each conditional branch is the default after the SAM-SLR recognition results process.

Chapter 5

Improved processing speed of Isolated Sign Language Recognition

5.1 Introduction

We are researching real-time sign language recognition to facilitate communication with signers, and the speed of recognition results is one of the critical aspects. Various researchers have proposed models with high recognition rates for sign language recognition. Sign language recognition has significantly improved in recent years in achieving high recognition accuracy. This progress can be attributed to the development of practical, deep-learning architectures and enhanced computational power with GPUs. In sign language video recognition, the standard approach involves pre-processing the video by converting it into frames, extracting positional data on fingers, arms, nose, mouth, and others by using techniques like postural estimation and utilizing a Three-Dimensional Convolutional Neural Network (3DCNN) [14] in the machine learning stage for recognizing sign language. Among these models, models such as MMPose [5], OpenPose [15], and MediaPipe [16] are used for the posture estimation process and tend to have higher recognition rates than other image processing. Furthermore, current proposals not only strive for high recognition rates using a single model but also utilize multiple recognition models and fuse their results to enhance the recognition rate. There are two main types of sign language recognition datasets: isolated signs and continuous signs.

This study focuses on the isolated SLR task and uses the RGB video of the Turkish isolated sign language dataset AUTSL. This dataset was used in a workshop competition. It contains 226 words each of RGB and Depth sign language videos, and the highest sign language recognition system has a recognition rate of over 98%, approaching the level of human recognition. Moreover, some researchers say this dataset's recognition rate may increase to under 99%. We have worked on improving recognition rates [29, 39], but we decided that the recognition rate was reaching a high level and decided to work on research on real-time processing. The real-time orphaned symbol recognition system proposed in this study is based on the public code of the first version of the Skeleton Aware Multi-modal SLR (SAM-SLR), which is the competition's winner.

Our proposed system performs real-time sign language recognition while concurrently playing a single sign language video, resulting in recognition outcomes comparable to those achieved through the original method. Initially, we focused on identifying software capable of rapidly generating alternative optical flow, given the original method's incompatibility with commercially available hardware. Subsequently, we customized the code of the selected alternative software to generate results almost comparable to the optical flow software used in the original method. In the Results section, we confirmed slight differences in the output results. We updated our software to the latest versions, which included PyTorch and OpenCV, and integrated the processing of the four independent modalities into a single unified process. Next, our primary focus was reducing and parallelizing the processing to save time while maintaining comparable recognition results to those of the original method.

- The latest hardware and software have improved processing speed.
- While the original method utilized MMPose for posture estimation with two different image sizes to enhance accuracy, our proposed method used a single image size of 640x640 to streamline processing and accelerate the procedure.
- Parallelizing the image processing enhances speed.
- Reducing the file accesses and using internal memory optimizes speed.

5.2 Methodology

We designed it to respond in real time based on the published code of the SAM-SLR model [1]. In order to realize real-time processing, there are a wide variety of things, such as improving the hardware environment, improving the software environment used, seamless recognition processing per video, parallelizing the recognition processing, and improving the code.

5.2.1 Improved hardware environments

Improvements in CPU and GPU computers are required for image recognition, especially in GPU performance, enabling real-time processing of sign language recognition. (Fig. 5.1) The processing power of the RTX 4090 used this time makes it possible to perform the processing of MMPose and Optical flow, which are used for estimation of each frame, at the same time when playing back 30 fps sign language videos.

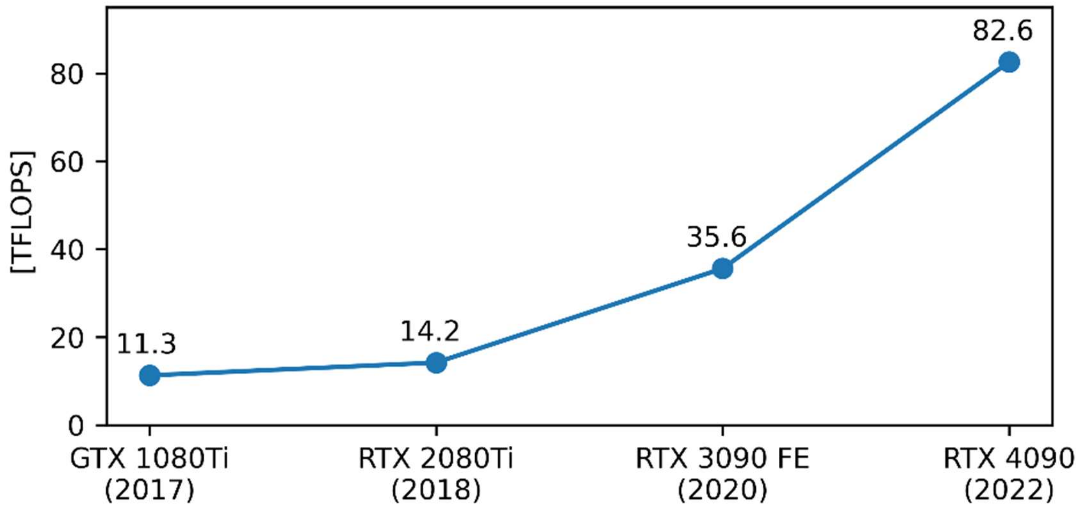


Fig. 5.1. GPU Benchmark. (FP32) [40 - 42]

5.2.2 Improved Software environments

Although MMPose, Optical flow, and model SL-GCN using attitude estimation use PyTorch, the processing speed performance is different due to different versions, especially in version 2.0, which is slightly faster. Also, PyTorch can be selected for PyTorch installation in the Ubuntu environment, but after checking both, in this case, using Anaconda (mini condo) was able to speed up the processing speed slightly. There are reports that mini conda is often faster.

5.2.3 Seamless recognition processing for each video

Published code is designed to process all data in bulk for machine learning and testing data verification. Therefore, it is common for the data required for machine learning to be individually processed by a plurality of processes and outputted files for each file. A seamless flow of independent processes is required while simultaneously playing video or shooting sign language to transition from the original procedural code to real-time processing. Also, the published code was released in 2020, and the software version has been upgraded, the hardware architecture has improved, and the code can only be operated by correcting the code.

5.2.3.1 Seamless recognition processing for each video. (Example of RGB-Flow)

The following is a flow of procedure differences of RGB-Flow modality for the original system and real-time system.

Original system

(1) From the 3,742 sign language videos of the test data, the posture information is saved to the temporary file using MMPose for each video.

(2) Create an Optical flow image from 3,742 test data.

(3) The center position of the person is calculated from the posture information of the temporary file for each video.

(4) Recognition processing with the 3DCNN model.

Real-time System

(1) Extraction of posture information by MMPose per each image during play-back.

(2) Optical flow image creation per each image during playback.

(3) Centering and resizing of Optical flow images after playback.

(4) Recognition processing with 3DCNN model.

5.2.3.2 Alternative Optical flow Processing

There are two problems with the optical flow processing: it did not work on current GPUs, and the output result was different from the original if alternative software running on current GPUs was used as is (Fig. 5.2). First, we discuss the current GPU operating issue. The optical flow software used in the original code was provided on Ubuntu 16 using Docker. It worked on the RTX-2000 series, but was found not to work on the RTX-3000 series or later, so alternative software was sought and modified to make the code as equivalent as possible. This alternative optical flow [3] is available at the following address. (https://github.com/open-mmlab/denseflow) Next, we checked the output results of the alternative software and found that it differed from the original. Figure 5.2 displays the output results of the original and the Alternative's Optical flow processing. Optical flow processing requires two images as it is based on the difference between two frame images. The first image, No. 11, exhibits the most noticeable difference. We believed that the disparity between the two images would significantly impact the recognition process. Therefore, we checked the code of the alternative software to obtain equivalent results and found the following.

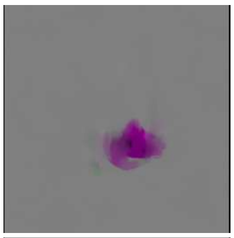
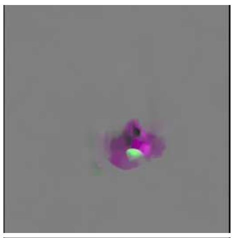
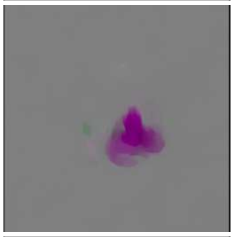
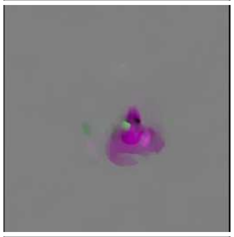
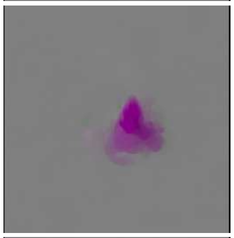
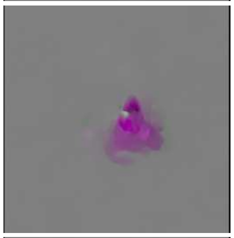
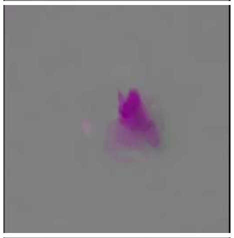
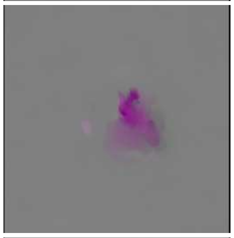
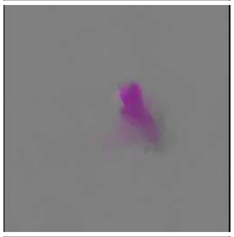
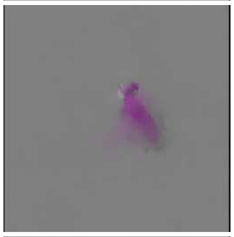
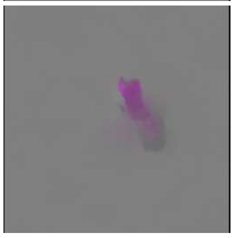
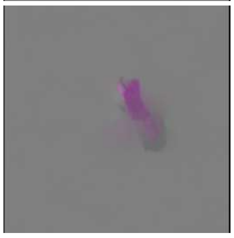
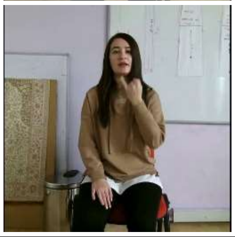
Frame No	Original triming images	No	Introduced method (OpenCV 2.4.13)	Alternative Method (OpenCV 3)
11		11		
12		12		
13		13		
14		14		
15		15		
16		16		
17				

Fig. 5.2. Differences of optical flow images between introduced and alternative methods. (Test dataset : signer34_sample1.mp4)

This alternative software is from the same series that was initially provided using Docker, but the main differences lie in the available GPU environment and the optical flow algorithm used. The optical flow algorithm used in both software is OpenCV's TVL1, while the code used in the original was based on OpenCV version 2.4.13 (Fig. 5.3). In contrast, the code used in the alternative software was based on OpenCV version 3 (Fig 5.4(a) and 5.5(a)). Our proposed code is shown in Figure 5.4(b) and 5.5(b), with the text areas highlighted in red being relevant to the modification.

```
void cv::gpu::OpticalFlowDual_TVL1_GPU::operator ()(const GpuMat& I0, const
GpuMat& I1, GpuMat& flowx, GpuMat& flowy)
{

    // create the scales
    for (int s = 1; s < nscales; ++s)
    {
        gpu::pyrDown(I0s[s - 1], I0s[s]);
        gpu::pyrDown(I1s[s - 1], I1s[s]);

       if (useInitialFlow)
        {
            gpu::pyrDown(u1s[s - 1], u1s[s]);
            gpu::pyrDown(u2s[s - 1], u2s[s]);

            gpu::multiply(u1s[s], Scalar::all(0.5), u1s[s]);
            gpu::multiply(u2s[s], Scalar::all(0.5), u2s[s]);
        }

    }

    // pyramidal structure for computing the optical flow
    for (int s = nscales - 1; s >= 0; --s)
    {

        // scale the optical flow with the appropriate zoom factor
        gpu::multiply(u1s[s - 1], Scalar::all(2), u1s[s - 1]);
        gpu::multiply(u2s[s - 1], Scalar::all(2), u2s[s - 1]);
    }
}
```

Fig. 5.3. Code excerpts for the original method’s optical flow. (tvl1flow.cpp)

```
class CV_EXPORTS_W OpticalFlowDual_TVL1 : public DenseOpticalFlow
{
public:

             double scaleStep = 0.8,

};
```

(a) Alternative method.

```
class CV_EXPORTS_W OpticalFlowDual_TVL1 : public DenseOpticalFlow
{
public:

             double scaleStep = 0.5,

};
```

(b) Proposed method.

Fig. 5.4. Optical Flow Code Change: ‘scaleStep_’ from 0.8 to 0.5.

```
void OpticalFlowDual_TVL1_Impl::calcImpl(const GpuMat& I0, const GpuMat& I1, GpuMat& flowx, GpuMat& flowy, Stream& stream)
{

    for (int s = 1; s < nscales_; ++s)
    {
        cuda::resize(I0s[s-1], I0s[s], Size(), scaleStep_, scaleStep_, INTER_LINEAR, stream);
        cuda::resize(I1s[s-1], I1s[s], Size(), scaleStep_, scaleStep_, INTER_LINEAR, stream);

        if (useInitialFlow_)
        {
            cuda::resize(u1s[s-1], u1s[s], Size(), scaleStep_, scaleStep_, INTER_LINEAR, stream);
            cuda::resize(u2s[s-1], u2s[s], Size(), scaleStep_, scaleStep_, INTER_LINEAR, stream);

            cuda::multiply(u1s[s], Scalar::all(scaleStep_), u1s[s], 1, -1, stream);
            cuda::multiply(u2s[s], Scalar::all(scaleStep_), u2s[s], 1, -1, stream);
        }

    }

    for (int s = nscales_ - 1; s >= 0; --s)
    {

        // scale the optical flow with the appropriate zoom factor
        cuda::multiply(u1s[s - 1], Scalar::all(1/scaleStep_), u1s[s - 1], 1, -1, stream);
        cuda::multiply(u2s[s - 1], Scalar::all(1/scaleStep_), u2s[s - 1], 1, -1, stream);
    }
}
```

(a) Alternative method.

```
void OpticalFlowDual_TVL1_Impl::calcImpl(const GpuMat& I0, const GpuMat& I1, GpuMat& flowx, GpuMat& flowy, Stream& stream)
{

    for (int s = 1; s < nscales_; ++s)
    {
        cuda::pyrDown(I0s[s - 1], I0s[s]);
        cuda::pyrDown(I1s[s - 1], I1s[s]);

        if (useInitialFlow_)
        {
            cuda::pyrDown(I0s[s - 1], I0s[s]);
            cuda::pyrDown(I1s[s - 1], I1s[s]);

            cuda::multiply(u1s[s], Scalar::all(scaleStep_), u1s[s], 1, -1, stream);
            cuda::multiply(u2s[s], Scalar::all(scaleStep_), u2s[s], 1, -1, stream);
        }

    }

    for (int s = nscales_ - 1; s >= 0; --s)
    {

        cuda::multiply(u1s[s - 1], Scalar::all(1/scaleStep_), u1s[s - 1], 1, -1, stream);
        cuda::multiply(u2s[s - 1], Scalar::all(1/scaleStep_), u2s[s - 1], 1, -1, stream);
    }
}
```

(b) Proposed method.

Fig. 5.5. Optical Flow Code Change: ‘resize’ to ‘pyrDown’.

The main modifications involve replacing the 'resize' function with 'pyrDown' and adjusting the 'scaleStep_' constant from 0.8 to 0.5. This aligns the code with the SAM-SLR implementation and improves consistency with the previous optical flow images.

5.2.4 Improvements and Treatments of Each Process

Real-time processing includes pre-processing, video playback (sign language shooting), image processing, image recognition, and recognition results. Figure 5.6 shows the real-time processing method. In pre-processing, the recognition model is loaded in advance so that the recognition result is output as soon as the data is entered. In real-time sign language recognition, if one frame is 30 fps, there is 0.033s, so if the processing required for one frame of the image can be completed within this time, then the required processing time is the response time after the end of sign language. Therefore, if a process requires more than 0.033s, that amount will directly affect the response time. In this case, we tried to use this one-image playback time to perform MMPose and Optical flow processing. We were able to reduce processing time and find new, faster methods. Also, PyTorch's data loader is too slow to process in real time, so we prepared our own. After the playback (shooting), the data is processed to the required shape for each model. Recognize each model, integrate each recognition result, and make a final decision.

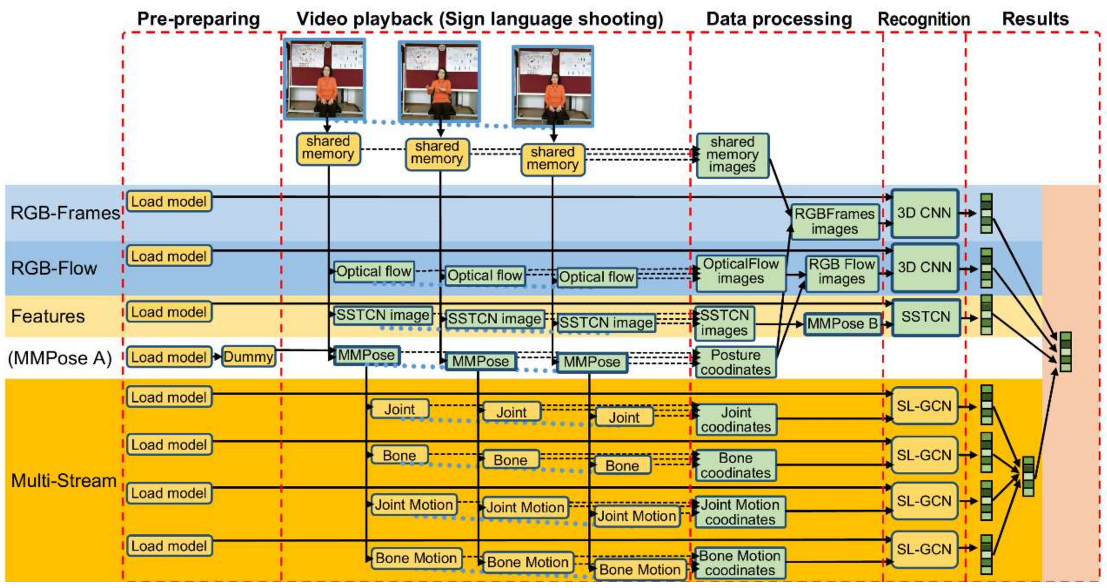


Fig. 5.6. Flow chart of real-time isolated sign language recognition processing.

5.2.4.1 Preparation Processing

In preparation, loading each recognition model before video playback (sign language shooting) allows you to move the load time forward. In addition, when using the recognition model, the first thing recognized is often slower than the one recognized after that. Therefore, if the dummy data is used to perform the recognition operation, the data to be recognized can be operated in an optimal state.

5.2.4.2 Unique Data Processing

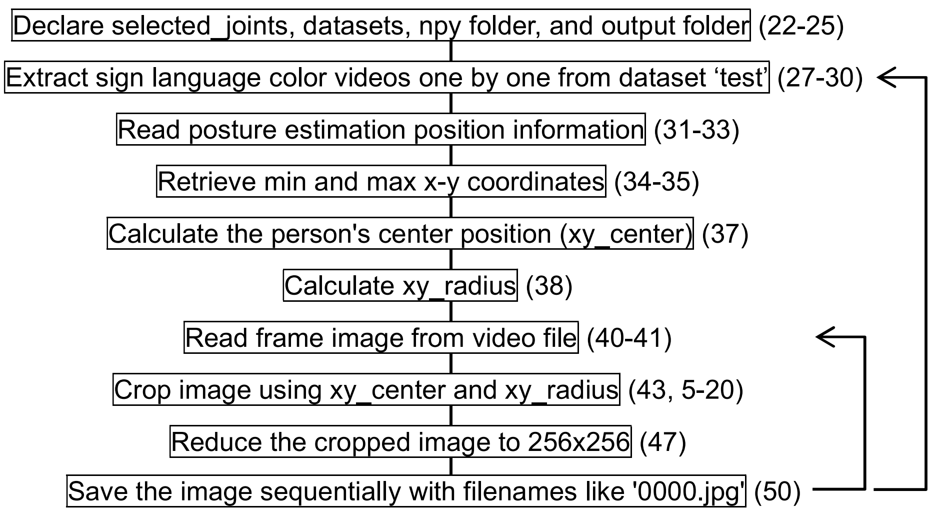
The data loader provided by PyTorch learns many data or processes the test data. At this time, the data loader prepared by PyTorch takes much time, so it takes much time to process the video data of the data required for processing. Changed the label to code that is passed directly to PyTorch.

5.2.4.3 Processing Possible During Playback

The interval of one frame is 0.033s when 30 fps. If the MMPose and Optical flow processing can be done within this time frame, it will be possible to recognize sign languages in real time. As for MMPose, it was found that the processing time would take a considerable amount of time, and the response time would be considerable if implemented as it was, so we considered reducing the processing. The reduction of processing is described in the next section.

5.2.4.4 Python to C++ and internal memory

We have ported the components of RGB-Frames and RGB-Flow, responsible for preprocessing images prior to learning, from Python to C++, to enhance image processing task speed. We will illustrate the Python script 'gen_frames.py' (Fig. 5.7) as an example, introduce C++ ported code, and explain leveraging internal memory for file storage. 'gen_frames.py' serves as a preprocessing code for RGB-Frames. As described in Chapter 2 (RGB-Frames), the script processes posture estimation from sign language videos. After saving the position information in an npy file, the code utilizes this information to crop the image, centering the person. Each resulting frame is then sequentially numbered and saved as '0000.jpg' for use in machine learning. The ‘gen_frames.py’ file includes both the main code and the ‘crop’ function. Here's an overview of the key steps:


Declare selected_joints, datasets, npy folder, and output folder (22-25)
Extract sign language color videos one by one from dataset 'test' (27-30)
Read posture estimation position information (31-33)
Retrieve min and max x-y coordinates (34-35)
Calculate the person's center position (xy_center) (37)
Calculate xy_radius (38)
Read frame image from video file (40-41)
Crop image using xy_center and xy_radius (43, 5-20)
Reduce the cropped image to 256x256 (47)
Save the image sequentially with filenames like '0000.jpg' (50)

(a) Algorithm of 'gen_frames.py'. (In parentheses is the code's line number.)

```
 5 def crop(img, center, radius, size=512):
 6   scale = 1.3
 7   radius_crop = (radius * scale).astype(np.int32)
 8   center_crop = (center).astype(np.int32)
 9   rect = (max(0,(center_crop-radius_crop)[0]), max(0,(center_crop-radius_crop)[1]),m
   in(512,(center_crop+radius_crop)[0]), min(512,(center_crop+radius_crop)[1]))
11   img = img[rect[1]:rect[3],rect[0]:rect[2],:]
12   if img.shape[0] < img.shape[1]:
13     top = abs(img.shape[0] - img.shape[1]) // 2
14     bottom = abs(img.shape[0] - img.shape[1]) - top
15     img = cv2.copyMakeBorder(img, top, bottom,0,0,cv2.BORDER_CONSTANT,value=(0,0,0))
16   elif img.shape[0] > img.shape[1]:
17     left = abs(img.shape[0] - img.shape[1]) // 2
18     right = abs(img.shape[0] - img.shape[1]) - left
19     img = cv2.copyMakeBorder(img,0,0,left, right, cv2.BORDER_CONSTANT,value=(0,0,0))
20   return img
21
22 selected_joints = np.concatenate(([0,1,2,3,4,5,6,7,8,9,10], [91,95,96,99,100,103,104
   , 107,108,111],[112,116,117,120,121,124,125,128,129,132]), axis=0)
23 folder = 'test' # 'train', 'val'
24 npydir = 'test_npy/npy3' # 'train_npy/npy3', 'val_npy/npy3'
25 outdir = 'test_frames' # 'train_frames' 'val_frames'
26
27 for root, dirs, files in os.walk(folder, topdown=False):
28   for name in files:
29     if 'color' in name:
30       cap = cv2.VideoCapture(os.path.join(root, name))
31       npy = np.load(os.path.join(npydir, name + '.npy')).astype(np.float32)
32       npy = npy[:, selected_joints, :2]
33       npy[:, :, 0] = 512 - npy[:, :, 0]
34       xy_max = npy.max(axis=1, keepdims=False).max(axis=0, keepdims=False)
35       xy_min = npy.min(axis=1, keepdims=False).min(axis=0, keepdims=False)
36       assert xy_max.shape == (2,)
37       xy_center = (xy_max + xy_min) / 2 - 20
38       xy_radius = (xy_max - xy_center).max(axis=0)
39       index = 0
40       while True:
41         ret, frame = cap.read()
42         if ret:
43           img = crop(frame, xy_center, xy_radius)
44         else:
45           break
46         index = index + 1
47         img = cv2.resize(img, (256,256))
48         if not os.path.exists(os.path.join(outdir, name[:-10])):
49           os.makedirs(os.path.join(outdir, name[:-10]))
50         cv2.imwrite(os.path.join(outdir, name[:-10], '{:04d}.jpg'.format(idx)), img)
```

(b) Code excerpts for RGB-Frames's crop. (gen_frames.py)

Fig. 5.7. Original code for RGB-Frames image process. (gen_frames.py)

In changing from this Python code to C++, this time code using internal memory is added. In ‘rgb_frames.cpp’, there is the main code (Fig. 5.8(a)) and the functions ‘read_npy’ (Fig. 5.8(b)), ‘init_mmap_frames’ (Fig. 5.8(c)), ‘mmap_frames_proc' (Fig. 5.8(d)), ‘xy_max_min’ (Fig. 5.8(e)), and ‘crop’ (Fig. 5.8(f)). The following is the program flow.

1. Read the posture position information with function ‘read_npy’ (Fig. 5.8(b)).
2. Initialize internal memory with function ‘init_mmap_frames’ (Fig. 5.8(c)).
3. Execute the function ‘mmap_frames_proc’ (Fig. 5.8(d)):
   3-1. Call the internal memory ‘mm1’ of frame images.
   3-2. Call function ‘xy_max_min’ (Fig. 5.8(e)).
      3-2-1. Read position information from the specified joint number
      3-2-2. Obtain the min and max x-y coordinates.
   3-3. Allocate internal memory ‘mm_frames’ for saving trimmed images.
      3-3-1. Read frame images from internal memory ‘mm1’ with a ‘for’ statement.
      3-3-2. Call function ‘crop’ (Fig. 5.8(f)) and crop the image using ‘xy_center’ and ‘xy_radius’.
      3-3-3. Reduce the cropped image to the size of 256x256.
      3-3-4. Save the cropped image to internal memory ‘mm_frames’.

```
#include <fcntl.h>
#include <sys/mman.h>
#include <unistd.h>
#include <opencv2/opencv.hpp>

using namespace std;
float npy1[250][133][3] = {0};

int main(int argc, char *argv[]){

    string file_name = argv[1] , file_name2;
    file_name2 = file_name.substr(0, strlen(file_name.c_str() ) - 4);
    std::string filename = "/home/h1/CVPR21Chal-SLR/real/npy3/";
    filename += file_name2;
    filename += ".npy";

    int frame_len = read_npy(filename);

    std::string filename_frames = "/tmp/mmap_frames";
    std::size_t n_frames = 256 * 256 * 3 * 250;
    int ret = init_mmap_frames(filename_frames, n_frames);

    struct timespec req = {0, 150000};
    nanosleep(&req, NULL);

    int ret2 = mmap_frames_proc(filename_frames, n_frames, frame_len);

    return 0;
}
```

Fig. 5.8(a). Code Conversion: Python to C++. (function ‘main’)

```
int read_npy(string filename){
    std::ifstream file(filename, std::ios::binary);
    if (!file)
    {
        std::cout << "failed" << filename << std::endl;
        // return 1;
    }

    uint16_t headerLen;
    file.read(reinterpret_cast<char*>(&headerLen), 2);
    headerLen = ((headerLen >> 8) & 0x00FF) | ((headerLen << 8) & 0xFF00);
    std::string headerStr(headerLen, '\0');
    file.read(&headerStr[0], headerLen);

    size_t shapePos = headerStr.find("'shape': (");
    shapePos += 10;
    size_t shapeEndPos = headerStr.find(")", shapePos);
    std::string shapeStr = headerStr.substr(shapePos, shapeEndPos - shapePos);

    std::vector<size_t> shape;
    size_t start = 0;
    size_t end = shapeStr.find(",", start);
    while (end != std::string::npos)
    {
        size_t size = std::stoi(shapeStr.substr(start, end - start));
        shape.push_back(size);
        start = end + 1;
        end = shapeStr.find(",", start);
    }
    size_t lastSize = std::stoi(shapeStr.substr(start, shapeStr.length() - start));
    shape.push_back(lastSize);
    file.seekg(8 + 2 + 118);
    file.read(reinterpret_cast<char*>(&npy1), sizeof(float) * shape[0]*133*3);
    file.close();
    return shape[0];
}
```

Fig. 5.8(b). Code Conversion: Python to C++. (function 'read_npy')

```
int init_mmap_frames(string filename_frames, int n_frames){
    std::ofstream file_frames(filename_frames, std::ios::binary | std::ios::out);
    file_frames.seekp(n_frames - 1);
    file_frames.write("", 1);
    file_frames.close();

    return 0;
}
```

Fig. 5.8(c). Code Conversion: Python to C++. (function 'init_mmap_frames')

```
int mmap_frames_proc(string filename, int n_frames, int frame_len){
    const int n250 = 512 * 512 * 3 * 250;
    std::size_t n1_frames = 256 * 256 * 3;

    int fd = open("/tmp/mmaptest1", O_RDONLY);
    if (fd < 0) {
        std::cerr << "Failed to open file." << std::endl;
        return 1;
    }

    void* mm1 = mmap(NULL, n250, PROT_READ, MAP_SHARED, fd, 0);
    if (mm1 == MAP_FAILED) {
        std::cerr << "Failed to mmap." << std::endl;
        close(fd);
        return 1;
    }

    cv::Point2d  xy_max, xy_min;
    std::tie(xy_max, xy_min) = xy_max_min(frame_len);
    int fd_frames = open(filename.c_str(), O_RDWR);
    void* mm_frames = mmap(0, n_frames, PROT_WRITE, MAP_SHARED, fd_frames, 0);
    cv::Mat img;
    cv::Point2d  xy_center = (xy_max + xy_min) / 2.0 - cv::Point2d { 20, 20 };
    float xy_radius = std::max(xy_max.x - xy_center.x, xy_max.y - xy_center.y);

    for (int i1 = 0; i1 < frame_len; i1++){
        off_t offset = i1 * 512 * 512 * 3;
        uchar* buf = static_cast<uchar*>(mm1) + offset;
        img = cv::Mat(cv::Size(512, 512), CV_8UC3, buf);
        cv::Mat cropped_img = crop(img, xy_center, xy_radius);
        cv::resize(cropped_img, cropped_img, cv::Size(256, 256));
        cv::cvtColor(cropped_img, cropped_img, cv::COLOR_BGR2RGB);
        std::memcpy(static_cast<char*>(mm_frames) + n1_frames * i1, cropped_img.data,
 n1_frames);
    }
    munmap(mm1, n250);
    close(fd);
    return 0;
}
```

Fig. 5.8(d). Code Conversion: Python to C++. (function ‘mmap_frames_proc’)

```
std::tuple<cv::Point2d, cv::Point2d> xy_max_min(int frame_len){

    std::vector<int> selected_joints = { 0,1,2,3,4,5,6,7,8,9,10,
                                         91,95,96,99,100,103,104,107,108,111,
                                         112,116,117,120,121,124,125,128,129,132 };
    int num_joints = selected_joints.size();

    cv::Point2d  npy2[250][31];

    for (int i1 = 0; i1 <  frame_len; i1++){
      for (int i2 = 0; i2 < num_joints; i2++){
          npy2[i1][i2].x = 512 - npy1[i1][selected_joints[i2]][0];
          npy2[i1][i2].y = npy1[i1][selected_joints[i2]][1];
      }
    }

    cv::Point2d  xy_max = { -std::numeric_limits<float>::max(), -
 std::numeric_limits<float>::max() };
    cv::Point2d  xy_min = { std::numeric_limits<float>::max(),
 std::numeric_limits<float>::max() };

    for (int i = 0; i < frame_len; i++) {
        for (int j = 0; j < num_joints; j++) {
            xy_max.x = std::max(xy_max.x, npy2[i][j].x);
            xy_max.y = std::max(xy_max.y, npy2[i][j].y);
            xy_min.x = std::min(xy_min.x, npy2[i][j].x);
            xy_min.y = std::min(xy_min.y, npy2[i][j].y);
        }
    }

    return std::forward_as_tuple(xy_max, xy_min);
}
```

Fig. 5.8(e). Code Conversion: Python to C++. (function ‘xy_max_min’)

```
cv::Mat crop(cv::Mat image, cv::Point2d center, double radius, int size = 512)
{
    double scale = 1.3;
    cv::Point2d radius_crop = cv::Point2d(int(radius * scale), int(radius * scale) );
    cv::Point2d center_crop = center;

    int x1 = std::max(0, static_cast<int>(center_crop.x - radius_crop.x));
    int y1 = std::max(0, static_cast<int>(center_crop.y - radius_crop.y));
    int x2 = std::min(512, static_cast<int>(center_crop.x + radius_crop.x));
    int y2 = std::min(512, static_cast<int>(center_crop.y + radius_crop.y));

    cv::Rect rect(x1, y1, x2 - x1, y2 - y1);
    cv::Mat cropped_image = image(rect);

    if (cropped_image.rows < cropped_image.cols)
    {
        int top = std::abs(cropped_image.rows - cropped_image.cols) / 2;
        int bottom = std::abs(cropped_image.rows - cropped_image.cols) - top;
        cv::copyMakeBorder(cropped_image, cropped_image, top, bottom, 0, 0,
 cv::BORDER_CONSTANT, cv::Scalar(0, 0, 0));
    }
    else if (cropped_image.rows > cropped_image.cols)
    {
        int left = std::abs(cropped_image.rows - cropped_image.cols) / 2;
        int right = std::abs(cropped_image.rows - cropped_image.cols) - left;
        cv::copyMakeBorder(cropped_image, cropped_image, 0, 0, left, right,
 cv::BORDER_CONSTANT, cv::Scalar(0, 0, 0));
    }

    return cropped_image;
}
```

Fig. 5.8(f). Code Conversion: Python to C++ (function 'crop')

In the C++ code, internal memory was handled by the function 'mmap'. By using internal memory, file input/output is accelerated by using internal memory, which is faster than a hard disk, and the same memory space can be used between different languages such as Python and C++. Finally, the transition from Python to C++ and the optimized utilization of internal memory significantly improve the processing speed.

5.3 Results

We report on the recognition results of the test data in the original system and the real-time system proposed here, including the environment in which the measurements were made, an overview of the measurement results, details of the measurement time for each process, and details of the evaluation values for each process.

5.3.1 Measurement environment

We reworked the code for the real-time system based on the published code and performed the measurements in the hardware and software environments shown in Table 5.1. The following is a list of points that were noted during the measurement.

- The PC was rebooted every time before starting the measurement.
- Computer rooms were kept below 20 degrees Celsius to maintain computer performance, and computers were cooled by fans or other means.
- The trained models and initial system evaluations used in this study were created on the same PC environment used in the previous paper [29]. The primary PC environment was AMD 3960x CPU and RTX 3090 GPU.

Table 5.1. Hardware and Software Specifications.

Hardware	Specifications
CPU	Intel i9-13900K
GPU	ASUS TUF-RTX4090-O24G-GAMING
Mother Board	ASRock Z690 PRO RS
Memory	128GB
NVME	Intel Solid State Drive. 2048GB
Software	Specifications
OS	Ubuntu 20.04.6 LTS
NVIDIA driver	530.30.02
CUDA version	12.1
PyTorch version	2.0.1 (mini conda)
NVCC version	3.8.16
gcc version	V10.1.243

5.3.2 Overview

We introduced four methods to enhance processing speed and enable simultaneous recognition during playback. These methods involve parallel processing, optical flow processing using a single image, and utilizing Python to C++ conversion in conjunction with internal memory. Measurements were taken three times, and the resulting values were averaged over them. The graph in Figure 5.9 shows the average time of each process per video. For the 3,742 test data, the average recognition response time after the end of playback was 0.7248s for a single video playback time of 1.9648s in proposed method. To compare with our proposed method, we tested two serial processing approaches. The first processes recognition after playback, utilizing Python to C++ conversion and internal memory. The second processes recognition simultaneously with playback. In the first approach, the average recognition response times were 4.3995s, 4.4033s, and 4.4012s, resulting in an overall average of 4.4013s. In the second approach, the average recognition response times were 3.2583s, 3.2510s, and 3.2526s, resulting in an overall average of 3.2540s. Our proposed method demonstrated an average response time of 0.7248s, while the two serial processing methods recorded response times of 4.4013s and 3.2540s. Comparing with serial processing 1 and 2, our proposed method achieved an overall improvement of 83.5% and 77.7%, resulting in an average response speed enhancement of 6.07 times and 4.49 times, respectively. Before development, we considered real-time sign language recognition practical if the response time was within 1 second.

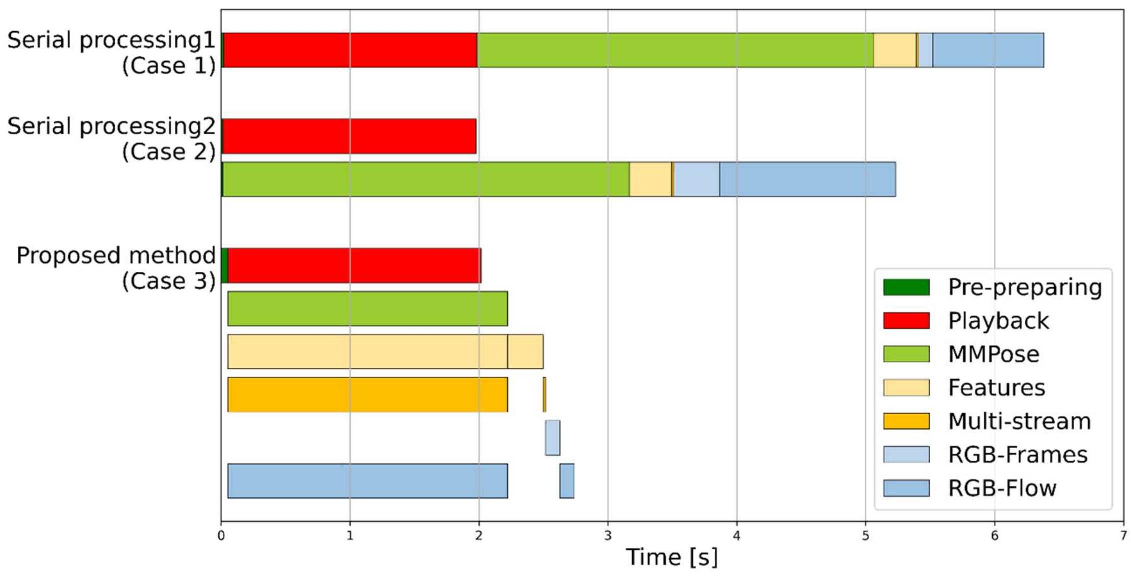


Fig. 5.9. Average processing time for serial processing and proposed method processing.

Considering that the real-time recognition response would be prolonged to process four modalities, we studied further improving the recognition rate of Multi-stream, the modality with the highest recognition rate. We increased the recognition rate from 96.82% to 97.01% [29]. The results of this study showed that even with the processing of the four modalities, the recognition response time is practical using current PC equipment. We also found that there are two time-consuming processes after video playback. The first is that Features requires about 0.3s to obtain posture information using information from all frame images, making it the most time-consuming of the post-video playback processes. Second, RGB-Frames and RGB-Flow take more time to process after video playback because they calculate the center position of the person from the posture estimation information of all frames and process the images so that the person is in the center.

Table 5.2 shows the recognition rates for SAM-SLR model and proposed method, respectively. The recognition rate results for SAM-SLR model and proposed method were the same, as were the recognition results for each test data. The recognition rate was calculated using Equation (5.1).

$$Accuracy\ rate = \frac{Number\ of\ correct\ answers}{3{,}742} \tag{5.1}$$

Table 5.2. Results of recognition rate for SAM-SLR model and proposed method.

	Recognition rate	Number of correct answers	Total number
SAM-SLR model	97.94%	3,665	3,742
Proposed method	97.94%	3,665	3,742

The evaluation values of each modality were also almost the same, which will be discussed in detail later.

5.3.3 Each processing time

The processing performed during playback is the image processing used in Optical flow, MMPose, and SSTCN, which can be processed in a single-frame image. The average playback time per video is 1.9648s, while the average is 2.1729s, indicating a delay of about 0.2s. If this problem can be improved, the response time can be reduced by 0.2s. One way to improve this problem is to use even faster GPUs in the future. As for software improvement, it is possible to rewrite the code from Python to C++ and to advance parallel processing. Regarding the measurement time for each modality, the modality with the fastest response time was Multi-stream with a response time of approximately 0.02s. Next was RGB-Flow and RGB-Frames at about 0.12s, followed by Features at 0.28s. These results suggest that if the posture estimation process can be processed during video playback, it will provide the fastest response time and can be processed in real-time for mobile devices.

Table 5.3. Average time for each processing for proposed method.

Processing	First	Second	Third	Average
Pre-preparing	0.0508	0.0502	0.0509	0.0506
Playback	1.9648	1.9648	1.9648	1.9648
Optical flow, MMPose, SSTCN image, Each stream	**2.1779**	**2.1686**	**2.1722**	**2.1729**
Postures coordinates	**0.0003**	**0.0003**	**0.0003**	**0.0003**
SSTCN images	0.0000	0.0000	0.0000	0.0000
SSTCN MMPoseB	0.1524	0.1524	0.1524	0.1524
SSTCN recognition	0.1220	0.1220	0.1220	0.1220
Features	**0.2744**	**0.2744**	**0.2744**	**0.2744**
Multi-stream coordinates	0.0017	0.0017	0.0017	0.0017
Multi-stream recognition	0.0172	0.0172	0.0172	0.0172
Multi-stream	**0.0189**	**0.0189**	**0.0189**	**0.0189**
RGB-Frames images	0.0687	0.0679	0.0688	0.0685
RGB-Frames recognition	0.0446	0.0445	0.0445	0.0445
RGB-Frames	**0.1133**	**0.1124**	**0.1133**	**0.1130**
RGB-Flow images	0.0660	0.0654	0.0655	0.0656
RGB-Flow recognition	0.0444	0.0444	0.0444	0.0444
RGB-Flow	**0.1104**	**0.1097**	**0.1099**	**0.1100**
Recognition processing	**2.6952**	**2.6844**	**2.6891**	**2.6896**
Response Time	**0.7304**	**0.7196**	**0.7243**	**0.7248**

We have already discussed average processing times in the overview. Here we use the table of average processing times between the two serial processing and the proposed method in Table 5.4. The three cases differ with and without the four processing speed acceleration options. Case 1 corresponds to the graph of serial processing 1 in Fig. 5.9, where the option 'Python to C++ and internal memory' was implemented, resulting in an average recognition time of 4.4010s. The average response time was naturally the same because recognition processing started after the completion of video playback. Case 2 corresponds to the graph of serial processing 2 in Fig. 5.9, where the option 'simultaneously process with playback' was utilized. The average recognition time was

5.2188s, and the average response time was 3.2540s; recognition processing is performed simultaneously with playback, leading to an improvement of 1.9648 seconds. Case 3 corresponds to the graph of the proposed method in Figure 5.9, where all four options are implemented. The average recognition time is 2.6896s, and the average response time is 0.7248s—about 6 and 4.5 times faster than the average response times of 4.4010s and 3.2540s for cases 1 and 2, respectively.

Table 5.4. Average processing time for various speed improvement options in difference cases.

Various speed improvement options	Case 1: Serial processing1	Case 2: Serial processing2	Case 3: Proposed method
Processing Possible During Playback		✓	✓
Parallel processing			✓
MMPose using single image			✓
Python to C++ & Internal memory	✓		✓
Processing			
Pre-preparing	0.0180	0.0143	0.0506
Playback	1.9648	1.9648	1.9648
Optical flow, MMPose, SSTCN image, Each stream	**-**	**-**	**2.1729**
MMPose only	**3.0792**	**3.1520**	**-**
Postures coordinates	**0.0000**	**0.0001**	**0.0003**
SSTCN images	0.0586	0.0571	0.0000
SSTCN MMPoseB	0.1490	0.1497	0.1524
SSTCN recognition	0.1211	0.1208	0.1220
Features	**0.3287**	**0.3277**	**0.2744**
Multi-stream coordinates	0.0013	0.0000	0.0017
Multi-stream recognition	0.0170	0.0178	0.0172
Multi-stream	**0.0183**	**0.0178**	**0.0189**
RGB-Frames images	0.0690	0.3127	0.0685
RGB-Frames recognition	0.0444	0.0446	0.0445
RGB-Frames	**0.1133**	**0.3573**	**0.1130**
RGB-Flow images	0.8172	1.3194	0.0656
RGB-Flow recognition	0.0442	0.0445	0.0444
RGB-Flow	**0.8614**	**1.3639**	**0.1100**
Recognition processing	**4.4010**	**5.2188**	**2.6896**
Response Time	**4.4010**	**3.2540**	**0.7248**

The difference between the recognition process in Case 1 and Case 2 lies in the implementation of the option 'Python to C++ and Internal memory.' This difference results in recognition processing times of 4.4010s and 5.2188s, respectively, indicating a speedup of 0.8178s. (Table 5.5) In Table 5.4, it is evident that only in Case 3, the option 'MMPose using single image' is implemented for MMPose processing, resulting in a time of 2.1729s. However, the actual MMPose time is shorter than this since it encompasses the processing of Optical Flow, SSTCN image, and Each stream. When MMPose is processed independently while playing back the video, the playback time and MMPose processing time align closely, suggesting an MMPose processing time of approximately 2 seconds, a value close to the average video playback time of 1.9648s. Considering that the MMPose processing times for Case 1 and Case 2 were 3.0792s and 3.1520s, respectively, averaging 3.1156s, MMPose processing time for Case 3 was 1.1156s faster when assumed to be 2 seconds. This speedup is due to the option 'MMPose using single image'. (Table 5.5) The difference between Case 2 and Case 3 lies in the implementation of three options: 'Parallel processing,' 'MMPose processing,' and 'Python to C++ and Internal memory.' This led to a reduction in recognition processing time from 3.2540s to 0.7248s, indicating a collective speedup of 2.5292s. As mentioned earlier, 'Python to C++ and Internal memory' speeds up the process by 0.8178s, and 'MMPose' by 1.1156s. This suggests that the 'parallel processing' option can accelerate the process by 0.5958s. (Table 5.5) We found that it is important to consider whether there is room for speed-up in all processes in order to perform speed-up processing.

Table 5.5. Improvement of recognition processing time for various speed improvement options.

Various speed improvement options	Time without using option [s]	Time with option [s]	Improved Time with other options [s]	Improved Time [s]
Parallel processing	3.2540	0.7248	0.8178 1.1156	0.5958
MMPose using single image	3.1156	2.0000	-	1.1156
Python to C++ & Internal memory	5.2188	4.4010	-	0.8178

5.3.4 Each processing scores

Table 5.6 and 5.7 show the original and real-time systems' evaluation value of the sign language recognition result. The sample result is the first sample video in the list of test data. The Late Fusion method in SAM-SLR is used for the evaluation values of each stream Joint, Bone, Joint Motion, Bone Motion and each modality RGB-Frames, RGB-Flow, Features, Multi-stream. The total value is obtained by multiplying each coefficient $\alpha = \{1.0, 0.9, 0.5, 0.5\}, \beta = \{0.9, 0.4, 0.4, 1.0\}$.

$$Score_{Multi-stream} = \alpha_1 q_{Joint} + \alpha_2 q_{\mathrm{Bone}} + \alpha_3 q_{Joint\ Motion} + \alpha_4 q_{Bone\ Motion} \tag{5.2}$$

$$Score_{SAM-SLR} = \beta_1 q_{RGB-Frames} + \beta_2 q_{RGB-Flow} + \beta_3 q_{Features} + \beta_4 q_{Multi-stream} \tag{5.3}$$

There coefficients and the formula (5.2) and (5.3) for calculating the evaluation value are the same as the first version of the SAM-SLR [1]. Also shown below are sample results (Table 5.6 and 5.7), with the original code evaluation values on the upper row and our results on the lower row. All three times measurements had the same value. The numbers represent the respective evaluation values, and the numbers in parentheses are the evaluation values multiplied by the respective coefficients. The results are generally comparable.

Two main differences exist between the information used in the original system and the proposed real-time system. First, the original system used two different image sizes, 512 and 640, for the posture estimation process using MMPose, and the most reliable one was chosen for each posture estimation result. However, the survey results showed that the 640 result was used in most cases. In addition, since the image size took about 1.6 times longer to process than 640 only, only 640 was used this time to save time. This resulted in some differences. The ones used for this result are Multi-stream, RGB-Frames, and RGB-Flow. Second, the Optical Flow software introduced in the original system did not work with the current system, so another software was used. This may have resulted in slightly different images from the Optical flow images in the original system, resulting in different evaluation values.

Table 5.6. Examples of the score of each stream. (The upper row is SAM-SLR model, the lower row is proposed method)

	Rate	Top-1	Top-2	Top-3
Gloss No	(SAM-SLR model)	133	106	146
	(Proposed method)	133	106	146
Joint	1.0	9.894 (9.894)	5.678 (5.678)	4.207 (4.207)
		9.896 (9.896)	5.672 (5.672)	4.206 (4.206)
Bone	0.9	9.437 (8.493)	5.292 (4.763)	5.215 (4.693)
		9.430 (8.487)	5.298 (4.768)	5.219 (4.697)
Joint Motion	0.5	9.025 (4.513)	4.691 (2.346)	4.953 (2.476)
		9.001 (4.501)	4.691 (2.346)	4.950 (2.475)
Bone Motion	0.5	11.095 (5.547)	6.965 (3.482)	5.555 (2.777)
		11.072 (5.536)	6.957 (3.479)	5.531 (2.766)
Multi-stream		(28.447)	(16.268)	(14.154)
		(28.419)	(16.264)	(14.144)

Table 5.7. Examples of the score of each modality. (The upper row is SAM-SLR model, the lower row is proposed method)

	Rate	Top-1	Top-2	Top-3
Gloss No	(SAM-SLR model)	133	106	146
	(Proposed method)	133	106	146
RGB-Frames	0.9	5.581 (5.023)	-0.676 (-0.609)	-0.853 (-0.768)
		5.536 (4.982)	-0.727 (-0.655)	-0.894 (-0.805)
RGB-Flow	0.4	6.015 (2.406)	-1.052 (-0.421)	-0.296 (-0.119)
		5.863 (2.345)	-1.094 (-0.438)	-0.292 (-0.117)
Features	0.4	7.723 (3.089)	0.253 (0.101)	0.142 (0.057)
		7.726 (3.090)	0.259 (0.104)	0.147 (0.059)
Multi-stream	1.0	28.447 (28.447)	16.268 (16.268)	14.154 (14.154)
		28.419 (28.419)	16.264 (16.264)	14.144 (14.144)
SAM-SLR		(38.965)	(15.339)	(13.324)
		(38.838)	(15.275)	(13.282)

Comparing them to Table 5.7, Features had almost the same content with the same values up to the second few. On the other hand, the Top-1 values for RGB-Flow were different by as much as 0.2081 between 6.015 and 5.863, suggesting that the MMPose and Optical flow processing had an impact. Despite these differences, it is clear that the evaluations were ranked correctly. The final recognition results are also considered to be good. Figure 5.10 shows the difference in the output results of the two software programs for optical flow. The image (a) was created by the software introduced in the original system, the image (b) was created by the alternative method, and the image (c) was created by the proposed method nearing original results using alternative method.

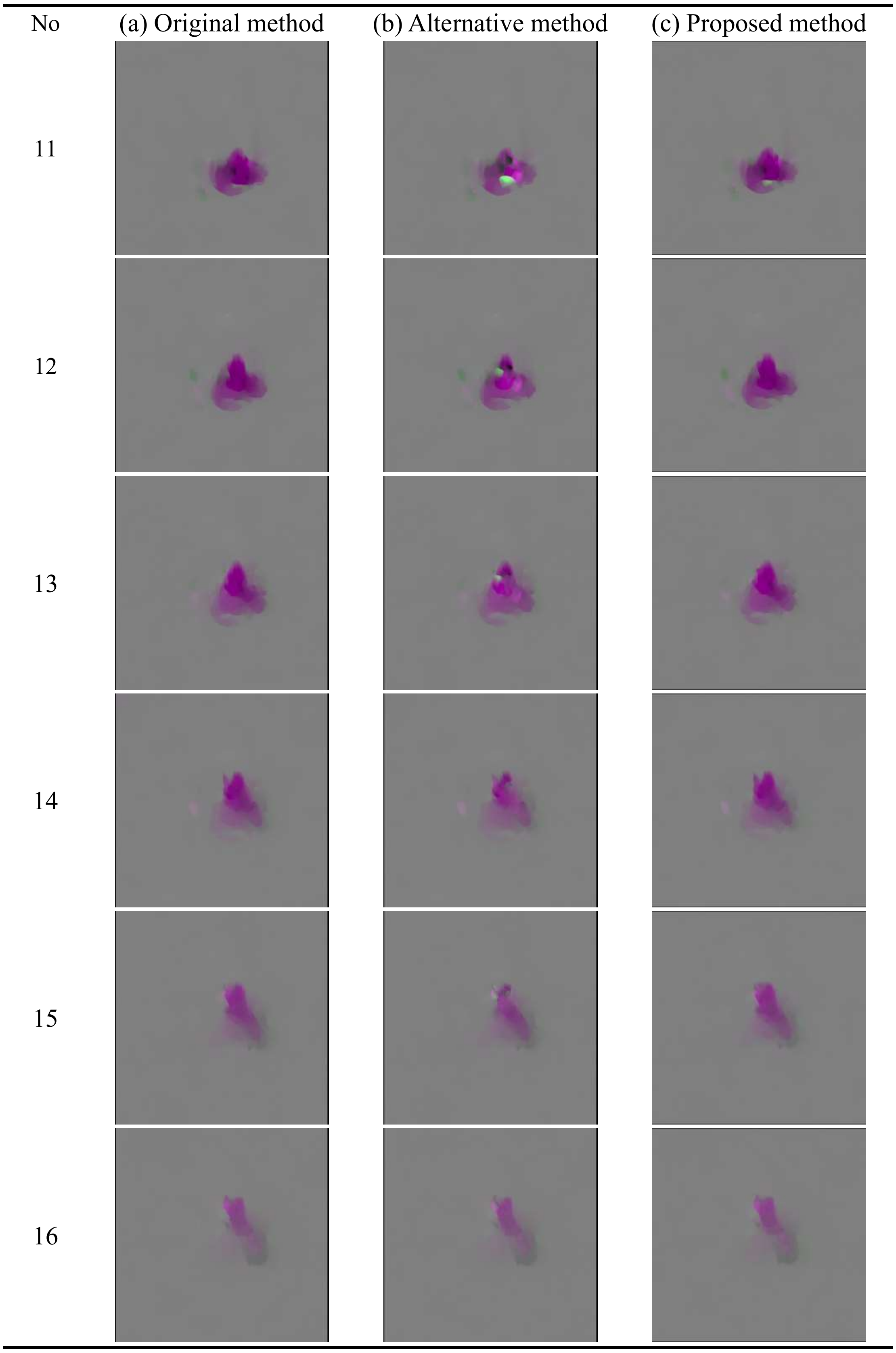


Fig. 5.10. Optical flow images: (a) Introduced method, (b) Alternative method, and (c) Proposed method nearing original results using alternative method.

5.4 Conclusion

In order to perform real-time processing using the PC equipment available in the summer of 2023 and using the SAM-SLR code, the recognition results were maintained, and the recognition response time was 0.7248s. Regarding practicality, this response time is an advantage to respond within 1 second. The practicality of this technique can be used to recognize simple sign language and can be applied to sign language learners, such as sign language coaching and gesture operation commands. In the future, if a higher speed is required, it will be possible to shorten the processing time and reduce the response time by using the current model by speeding up the model itself. Also, real-time sign language recognition will be possible even in the environment of smartphones. In addition, since the latest GPU performance can handle most of the required processing for each frame, the time for image processing and recognition processing after all frames are finished will be the bottleneck of response time. Further validation with other datasets will allow us to find new improvements in real-time processing, which is one of our future tasks. Finally, we are considering developing smartphone software, such as an application [43]. In addition, the consideration of the release of the code will be done after the series of projects is finished.

Chapter 6

Conclusion

6.1 Conclusion

In Chapter 1, the research background and purpose are described. We mainly mentioned the issues for improving the recognition rate and realizing the processing speed of the isolated sign language recognition model SAM-SLR from previous research. In Chapter 2, we describe related research on the isolated sign language model. In particular, he described in detail the model of the previous study, SAM-SLR, which is the basis of this study.

In Chapter 3, we delve into enhancing the recognition rate of isolated sign language by reusing estimation results per epoch. In Chapter 3, we delve into enhancing the recognition rate of isolated sign language by reusing estimation results per epoch. The distinctive contribution of this study involves the summation of Top-1 evaluation values for each class from the estimation results across epochs during the machine learning of sign language recognition. The average results from ten trials conducted before and after applying the proposed method reveal noteworthy improvements. Specifically, for the 'Joint' stream, the highest recognition rate increased by 0.21 percentage points, advancing from 95.66% to 95.87%, while the average recognition rate for epochs 150 to 229 improved by 0.49 percentage points, rising from 95.22% to 95.71%. Similarly, for the 'Bone' stream, the highest recognition rate improved by 0.27 points, reaching 95.98% from 95.71%, and the average recognition rate for epochs 150 to 229 increased by 0.56 points, progressing from 95.23% to 95.79%.

In Chapter 4, we detail the improvement in the recognition rate of isolated sign language by re-evaluating the position of the index finger relative to the face parts' positional reference. The novelty of this study lies in a method that further increases the recognition rate by addressing instances of low recognition when the difference between Top-1 and Top-2 evaluation results is minimal, without compromising the high recognition rate of the SAM-SLR model. The re-evaluation method captures the position within a triangular mesh formed by face parts such as eyes, nose, mouth, cheeks, and chin. This allows the recognition system to capture different face shapes in relative positions and evaluate the index finger's position in the face area by comparing test data and training data. The optimal method improved the recognition rate from 97.94% in the past to 98.24%.

Chapter 5 discusses improving processing speed for isolated sign language recognition. The novelty of this research is that in real-time sign language recognition using the sign language dataset AUTSL, the average response time was reduced from 4.4 seconds for serial processing to 0.72 seconds for the proposed method while maintaining a high recognition rate of 97.74%. This is an improvement in the recognition rate compared to other researchers' proposals, where the processing speed was faster, but the recognition rate was about 95%. The original point is that four modalities are processed simultaneously to improve the processing speed, and the evaluation values of the estimated results are almost maintained while reducing the processing. This enables equal and independent results from the SAM-SLR model across the four modalities, offering the potential for real-time processing when high recognition rates are achieved with newly created learning model results.

6.2 Future work

We aim to explore the prospects for isolated sign language recognition, which could have potential applications in continuous sign language recognition. (Fig. 6.1) For instance, one approach involves iteratively repeating the sign language recognition processing at short intervals. Upon achieving an accuracy of 90% or higher, we can adapt the corresponding word and proceed with the recognition processing for the next word. This iterative process could enhance overall recognition accuracy by combining various methods.

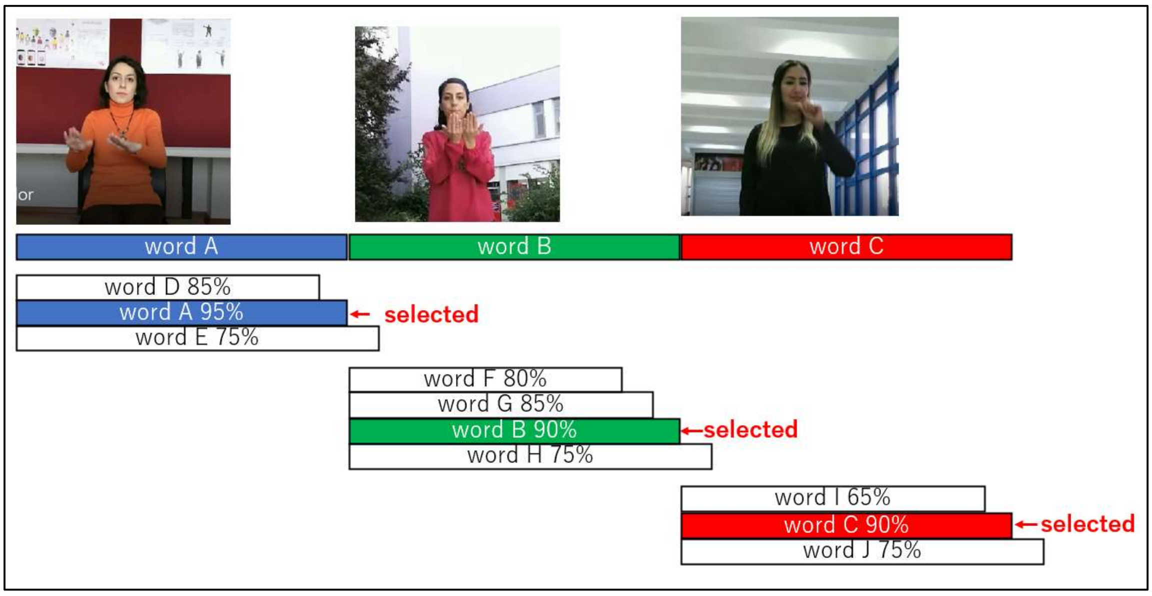


Fig. 6.1. Application to Continuous Sign Language Recognition.

References

[1] S. Jiang, B. Sun, L. Wang, Y. Bai, K. Li, and Y. Fu, "Skeleton aware Multi-modal sign language Recognition," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3413–3423.

[2] O. M. Sincan and H. Y. Keles, "AUTSL: A large scale multi-modal Turkish sign language dataset and baseline methods," IEEE Access, vol. 8, pp. 181340–181355, 2020.

[3] S. Wang, Z. Li, Y. Zhao, Y. Xiong, L. Wang, and D. Lin, "Denseflow," 2020. [Online]. Available: https://github.com/open-mmlab/denseflow (accessed on 26 February 2023).

[4] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L1 optical flow," in Proceedings of Joint Pattern Recognition Symposium, Springer, 2007, pp. 214–223.

[5] M. Contributors, "OpenMMLab pose estimation toolbox and benchmark," 2020. [Online]. Available: https://github.com/open-mmlab/mmpose

[6] J. Zhang, W. Zhou, C. Xie, J. Pu, and H. Li, "Chinese sign language recognition with adaptive HMM," in Proceedings of IEEE International Conference on Multimedia and Expo, 2016, pp. 1–6.

[7] D. Li, C. Rodriguez, X. Yu, and H. Li, "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison," in Proceedings of IEEE Winter Conference on Applications of Computer Vision, 2020, pp. 1459–1469.

[8] H. Vaezi Joze and O. Koller, "Ms-asl: A large-scale data set and benchmark for understanding American Sign Language," in The British Machine Vision Conference, 2019.

[9] S. Albanie, G. Varol, L. Momeni, T. Afouras, J. S. Chung, N. Fox, and A. Zisserman, "Bsl-1k: Scaling up co-articulated sign language recognition using mouthing cues," in European Conference on Computer Vision, 2020, pp. 35–53.

[10] S. Jiang, B. Sun, L. Wang, Y. Bai, K. Li, and Y. Fu, "Sign Language Recognition via Skeleton-Aware Multi-Model Ensemble," arXiv preprint arXiv:2110.06161, 2021.

[11] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in Proceedings of IEEE Computer Vision and Pattern Recognition, 2018, pp. 6450–6459.

[12] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier, and A. Zisserman, "A short note about kinetics-600," arXiv preprint arXiv:1808.01340, 2018.

[13] O. M. Sincan, J. C. S. Jacques Junior, S. Escalera, and H. Y. Keles, "Chalearn LAP large scale signer independent isolated sign language recognition challenge: Design, results and future research," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2021.

[14] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," IEEE Trans. Pattern Anal. Mach. Intell. TPAMI, 2013, 35, 221–231, doi: 10.1109/TPAMI.2012.59.

[15] Z. Cao, G. Hidalgo, T. Simon, S.E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," IEEE Trans. Pattern Anal. Mach. Intell., 2019, arXiv:1812.08008v2, doi: 10.1109/TPAMI.2019.2929257.

[16] Google Research Team, "MediaPipe," 2020. [Online]. Available: https://google.github.io/mediapipe/solutions/hands.html.

[17] H. Wang and L. Wang, "Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017, pp. 499–508.

[18] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018, pp. 7444–7452.

[19] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with multi-stream adaptive graph convolutional networks," IEEE Trans. Image Processing, 2020, 29, 9532–9545, doi: 10.1109/TIP.2020.3028207.

[20] K. Cheng, Y. Zhang, C. Cao, L. Shi, J. Cheng, and H. Lu, "Decoupling GCN with DropGraph Module for Skeleton-Based Action Recognition," in Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020, Springer: Berlin/Heidelberg, Germany, 2020, pp. 536–553.

[21] S. Jin, L. Xu, J. Xu, C. Wang, W. Liu, C. Qian, W. Ouyang, and P. Luo, "Whole-body human pose estimation in the wild," in Proceedings of the European Conference on Computer Vision (ECCV 2020), Glasgow, UK, 23-28 August 2020, pp. 196-214.

[22] Q. Xiao, M. Qin, and Y. Yin, "Skeleton-based Chinese sign language recognition and generation for bidirectional communication between deaf and hearing people," Neural Netw., 2020, 125, 41–55.

[23] Y. F. Song, Z. Zhang, C. Shan, and L. Wang, "Stronger, Faster and More Explainable: A Graph Convolutional Baseline for Skeleton-Based Action Recognition," in Proceedings of the 28th ACM International Conference on Multimedia (ACMMM), Seattle, WA, USA, 12–16 October 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 1625–1633.

[24] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, "Disentangling and Unifying Graph Convolutions for Skeleton-Based Action Recognition," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 143–152.

[25] M. Vázquez-Enríquez, J.L. Alba-Castro, L.D. Fernández, and E.R. Banga, "Isolated Sign Language Recognition with Multi-Scale Spatial-Temporal Graph Convolutional Networks," in Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Virtual, 19–25 June 2021; pp. 3457–3466.

[26] M. Hrúz, I. Gruber, J. Kanis, M. Boháček, M. Hlaváč, and Z. Krňoul, "One Model is Not Enough: Ensembles for Isolated Sign Language Recognition," Sensors, 2022, 22, 5043, doi: 10.3390/s22135043.

[27] M. Al-Hammadi, M. A. Bencherif, M. Alsulaiman, G. Muhammad, M.A. Mekhtiche, W. Abdul, Y. A. Alohali, T. S. Alrayes, H. Mathkour, M. Faisal, M. Algabri, H. Altaheri, T. Alfakih, and H. Ghaleb, "Spatial Attention-Based 3D Graph Convolutional Neural Network for Sign Language Recognition," Sensors, 2022, 22, 4558, doi 10.3390/s22124558.

[28] K. M. Dafnis, E. Chroni, C. Neidle, and D. N. Metaxas, "Bidirectional Skeleton-Based Isolated Sign Recognition using Graph Convolution Networks," in Proceedings of the 13th Conference on Language Resources and Evaluation (LREC), Marseille, France, 20-25 June 2022; pp. 7328–7338.

[29] N. Hori and M. Yamamoto, "Sign Language Recognition using the reuse of estimate results by each epoch," in Proceedings of the 7th International Conference on Frontiers of Signal Processing (ICFSP), Paris, France, 7-9 September 2022; pp. 45-50, doi: 10.1109/ICFSP55781.2022.9924938.

[30] M. Novopoltsev, L. Verkhovtsev, R. Murtazin, D. Milevich, and I. Zemtsova, "Fine-tuning of sign language recognition models: a technical report," arXiv, 2023, arXiv:2302.07693v2, doi: 2302.07693v2.

[31] D. Ryumin, D. Ivanko, and E. Ryumina, "Audio-Visual Speech and Gesture Recognition by Sensors of Mobile Devices," Sensors, vol. 23, p. 2284, 2023, doi: 10.3390/s23042284.

[32] O. M. Sincan, A. O. Tur, and H. Y. Keles, "Isolated sign language recognition with multi-scale features using LSTM," in Proceedings of the 27th Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey, 24-26 April 2019; pp. 1-4.

[33] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," Neural Netw., vol. 18, pp. 602–610, 2005, doi: 10.1016/j.neunet.2005.06.042.

[34] M. D. Coster, M. V. Herreweghe, and J. Dambre, "Isolated Sign Recognition from RGB Video using Pose Flow and Self-Attention," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021, pp. 3436-3445.

[35] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu, "Video Swin transformer," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19-24 June 2022; pp. 3202-3211.

[36] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Online, 11-17 October 2021; pp. 6824-6835.

[37] D. Li, C. Rodriguez, X. Yu, and H. Li, "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 4–8 January 2020; pp. 1459–1469.

[38] S. Albanie, G. Varol, L. Momeni, T. Afouras, J.S. Chung, N. Fox, and A. Zisserman, "BSL-1K: Scaling up co-articulated sign language recognition using mouthing cues," in Proceedings of the 16th European Conference on Computer Vision (ECCV 2020), Glasgow, UK, 23-28 August 2020; pp. 35–53.

[39] N. Hori and M. Yamamoto, "Re-Evaluation Method by Index Finger Position in the Face Area Using Face Part Position Criterion for Sign Language Recognition," Sensors, vol. 23, p. 4321, 2023.

[40] NVIDIA, "Spec for GTX 1080Ti and RTX 2080Ti," https://developer.nvidia.com/blog/?p=11872, last accessed on Aug. 12, 2023.

[41] NVIDIA, "Spec for RTX 3090 FE," https://www.nvidia.com/content/PDF/nvidia-ampere-ga-102-gpu-architecture-whitepaper-v2.pdf, last accessed on Aug. 12, 2023.

[42] NVIDIA, "Spec for RTX 2080Ti and RTX 4090," https://images.nvidia.com/aem-dam/Solutions/Data-Center/l4/nvidia-ada-gpu-architecture-whitepaper-v2.1.pdf, last accessed on Aug. 12, 2023.

[43] S. Bahri, I. Zulaikha, S. Saon, K. A. Mahamad, K. Isa, U. Fadlilah, B. A. M. Ahmadon, and S. Yamaguchi, "Interpretation of Bahasa Isyarat Malaysia (BIM) Using SSD-MobileNet-V2 FPNLite and COCO mAP," Information, vol. 14, no. 6, p. 319, 2023.

Appendix

1. Results of recognition rate from Ten trials of Training Joint and Proposed Joint with Random Initial Conditions.

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
0	13.74	13.74	16.27	16.27	15.77	15.77	13.71	13.71	13.34	13.34
1	34.42	34.18	33.78	33.08	15.47	15.07	35.54	35.46	41.29	41.18
2	49.57	45.70	50.11	49.12	47.97	31.69	46.15	45.46	57.88	55.77
3	63.68	62.91	65.21	63.25	61.17	52.67	67.26	64.48	66.19	66.62
4	68.04	69.21	70.28	71.57	66.76	64.27	68.07	69.99	70.92	74.72
5	69.43	73.46	68.01	75.31	68.63	71.33	73.94	76.00	70.79	79.16
6	76.80	78.46	69.99	79.66	73.28	76.51	76.99	80.28	73.70	82.47
7	81.43	83.16	79.90	82.92	80.89	81.35	78.46	83.83	81.24	84.85
8	79.34	85.03	81.00	86.00	79.00	84.61	78.01	86.18	79.48	86.83
9	80.09	87.28	82.60	87.41	81.24	85.92	75.01	87.04	81.16	88.19
10	80.76	87.79	81.27	88.13	85.41	87.55	81.11	87.87	80.47	88.35
11	85.03	88.72	82.02	89.10	82.84	88.51	75.04	88.38	82.87	89.18
12	70.74	88.91	81.11	89.68	74.48	88.88	82.31	88.78	83.75	89.76
13	83.30	89.85	82.10	89.90	83.70	88.91	82.71	89.12	84.66	90.57
14	84.13	90.30	83.30	90.54	82.04	89.55	83.73	89.50	82.68	90.46
15	83.86	90.73	85.01	90.86	80.89	90.09	80.95	90.03	80.60	90.73
16	87.04	91.02	83.38	91.29	80.81	90.11	83.30	90.33	84.29	91.02
17	85.94	91.07	83.70	91.82	81.51	90.59	82.63	90.75	78.22	91.26
18	86.16	91.31	79.93	91.90	83.94	91.23	84.71	91.13	83.59	91.31
19	86.02	91.53	86.53	92.12	84.15	91.23	83.67	91.34	83.51	91.64
20	85.20	91.77	84.23	92.20	85.01	91.90	80.92	91.56	84.69	91.80
21	87.76	92.06	86.45	92.46	83.00	91.96	86.02	91.96	86.32	92.14
22	84.66	92.22	84.42	92.76	84.63	92.01	86.08	91.85	85.70	92.14
23	85.60	92.49	85.20	92.65	87.33	92.30	87.15	92.30	85.92	92.22
24	86.80	92.54	87.25	92.81	84.58	92.46	86.50	92.46	87.92	92.46
25	86.10	92.70	86.93	92.68	82.98	92.22	83.70	92.54	88.11	92.41
26	87.36	92.65	85.28	92.97	87.15	92.57	87.89	92.76	76.38	92.52
27	87.04	92.86	88.19	93.27	85.03	92.46	82.82	92.81	89.82	92.62
28	83.24	92.89	87.09	93.27	84.13	92.49	87.33	92.89	88.91	92.73
29	88.59	93.03	60.82	93.27	89.95	92.73	67.88	92.92	88.13	92.97
30	87.73	93.16	88.75	93.40	87.73	92.81	85.70	93.11	86.58	93.08
31	86.40	93.08	87.20	93.48	87.33	92.76	87.39	93.13	88.27	93.03
32	89.55	93.05	88.30	93.40	87.55	92.76	89.15	93.32	86.72	93.11
33	88.88	93.16	87.76	93.43	89.47	92.76	85.84	93.19	87.92	93.08
34	89.18	93.35	86.10	93.56	87.81	93.03	88.16	93.35	87.65	93.27
35	88.91	93.45	82.36	93.53	87.84	92.97	86.13	93.32	85.81	93.43
36	85.09	93.37	88.00	93.61	87.81	93.00	89.10	93.45	88.70	93.43
37	88.75	93.51	87.60	93.45	87.49	93.03	90.54	93.53	87.09	93.37
38	87.49	93.53	86.26	93.61	88.48	93.29	86.40	93.56	87.73	93.35
39	89.55	93.51	86.37	93.72	33.65	93.27	86.77	93.59	86.88	93.59

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
40	88.75	93.59	86.16	93.80	87.81	93.27	88.62	93.88	88.32	93.53
41	9.73	93.59	87.39	93.75	89.04	93.19	88.13	93.85	88.78	93.59
42	90.41	93.59	87.89	93.83	89.44	93.45	88.78	93.80	88.21	93.51
43	79.96	93.53	83.35	93.88	84.82	93.21	87.68	93.91	87.39	93.61
44	87.23	93.69	84.90	93.83	88.03	93.21	90.09	94.07	87.79	93.56
45	90.17	93.67	87.07	93.83	87.60	93.35	87.87	94.09	84.63	93.53
46	89.12	93.83	89.23	93.93	88.16	93.40	88.75	94.07	88.11	93.59
47	88.88	93.93	88.62	93.88	89.85	93.40	90.54	94.23	83.83	93.53
48	88.59	93.99	87.73	94.01	88.78	93.53	87.52	94.28	87.15	93.67
49	86.77	93.96	88.03	94.07	87.33	93.43	90.11	94.47	87.79	93.67
50	84.93	94.04	87.15	94.09	89.23	93.61	89.31	94.47	87.17	93.72
51	88.13	94.09	89.55	94.15	88.83	93.75	89.93	94.39	89.36	93.77
52	86.58	94.07	85.84	94.17	86.88	93.72	90.25	94.33	88.64	93.77
53	88.32	94.01	88.21	94.17	85.81	93.80	90.03	94.41	88.35	93.83
54	88.96	94.07	89.79	94.12	89.34	93.77	72.23	94.36	88.43	93.77
55	89.02	94.04	67.13	94.12	83.03	93.83	85.62	94.39	87.25	93.88
56	90.73	94.09	87.76	94.31	89.39	93.69	88.72	94.36	89.10	93.88
57	89.34	94.12	88.30	94.31	87.36	93.75	90.27	94.36	88.27	93.96
58	85.17	94.09	88.00	94.31	87.17	93.83	89.20	94.44	89.23	94.04
59	90.01	94.09	90.86	94.28	87.28	93.91	88.40	94.47	87.60	94.12
60	89.20	94.15	88.62	94.20	88.88	94.07	90.59	94.52	89.02	94.07
61	88.19	94.15	89.04	94.25	87.17	93.91	87.25	94.58	82.39	94.01
62	90.54	94.20	87.17	94.23	83.14	93.96	88.59	94.63	88.70	94.04
63	88.32	94.20	87.31	94.25	88.21	93.99	90.11	94.68	89.26	94.01
64	89.93	94.25	88.80	94.25	88.24	94.07	90.22	94.74	89.23	94.07
65	88.91	94.33	90.17	94.23	88.88	94.07	88.46	94.79	88.86	94.09
66	89.20	94.41	88.62	94.23	86.56	94.07	87.73	94.71	89.79	94.09
67	90.03	94.44	86.21	94.31	88.19	94.15	90.03	94.79	88.99	94.09
68	89.66	94.47	86.48	94.36	88.11	94.23	89.23	94.82	88.96	94.25
69	88.88	94.58	89.34	94.41	86.53	94.12	89.28	94.82	87.92	94.28
70	90.59	94.60	87.95	94.44	88.11	94.23	90.30	94.79	90.33	94.28
71	87.87	94.55	86.00	94.44	88.27	94.23	87.17	94.92	88.05	94.31
72	88.70	94.63	88.03	94.44	86.56	94.33	86.42	94.84	87.41	94.33
73	88.99	94.68	88.00	94.49	88.54	94.33	90.83	94.87	89.63	94.36
74	89.85	94.58	87.15	94.44	89.55	94.33	91.23	94.84	88.99	94.36
75	87.49	94.68	85.70	94.55	87.39	94.39	85.60	94.87	89.44	94.39
76	89.50	94.71	89.04	94.52	87.92	94.36	88.88	94.82	88.35	94.39
77	91.42	94.74	86.21	94.44	90.03	94.31	90.67	94.84	87.52	94.41
78	89.71	94.74	87.57	94.49	88.51	94.33	89.85	94.84	88.75	94.47
79	90.89	94.76	89.26	94.58	86.37	94.41	90.01	94.90	88.78	94.47
80	88.70	94.71	86.02	94.41	89.10	94.41	87.17	94.95	87.84	94.49
81	88.13	94.82	90.22	94.47	88.48	94.58	77.71	94.92	88.78	94.52
82	89.71	94.82	87.84	94.44	87.47	94.52	86.61	94.95	88.83	94.49
83	89.47	94.79	88.51	94.52	89.10	94.58	89.42	94.92	89.07	94.58
84	88.72	94.79	86.26	94.52	88.62	94.66	90.59	94.98	87.95	94.52
85	89.87	94.79	88.75	94.55	89.58	94.60	90.09	94.98	86.64	94.55
86	89.90	94.79	87.76	94.58	89.93	94.66	89.76	95.06	89.50	94.55
87	88.27	94.90	88.83	94.63	89.28	94.71	84.74	94.98	89.98	94.58
88	87.49	94.82	89.18	94.60	88.43	94.68	89.93	95.03	86.24	94.60
89	89.68	94.92	89.12	94.71	90.46	94.71	89.18	95.03	87.49	94.68
90	90.62	94.90	89.66	94.68	87.71	94.68	87.73	95.00	88.11	94.71

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
91	89.55	94.95	89.52	94.66	87.71	94.68	88.11	95.06	88.35	94.71
92	88.96	94.84	88.59	94.68	88.05	94.68	88.64	95.08	87.15	94.74
93	90.01	94.87	6.04	94.71	84.13	94.66	88.62	95.03	89.39	94.68
94	91.61	95.00	87.81	94.79	89.42	94.63	89.20	95.03	83.30	94.68
95	90.30	95.03	89.68	94.82	90.54	94.74	90.99	95.03	89.12	94.71
96	89.60	94.98	88.78	94.82	89.44	94.68	89.42	95.03	89.34	94.74
97	89.23	95.06	87.47	94.90	88.05	94.71	89.52	95.06	91.05	94.79
98	88.64	95.11	90.30	94.90	87.60	94.68	90.43	95.08	83.78	94.84
99	90.03	95.08	88.94	94.90	89.44	94.71	90.09	95.08	89.79	94.82
100	90.41	95.11	91.53	94.92	90.73	94.79	91.23	95.06	89.52	94.82
101	89.87	95.19	89.26	94.98	88.75	94.82	90.49	95.19	90.17	94.84
102	91.56	95.16	90.81	95.00	90.30	94.82	90.67	95.08	87.01	94.90
103	86.83	95.24	89.02	95.00	89.71	94.87	90.06	95.22	88.99	94.87
104	90.25	95.27	86.80	95.03	90.01	94.87	90.19	95.27	87.57	94.92
105	89.82	95.24	88.83	95.06	89.58	94.87	88.72	95.22	90.22	94.95
106	90.11	95.24	86.24	95.00	89.66	94.90	90.19	95.19	87.92	95.03
107	91.93	95.30	89.79	95.00	89.76	94.90	89.93	95.16	90.51	95.03
108	89.26	95.38	89.74	95.03	89.10	94.92	89.93	95.14	89.47	94.92
109	90.70	95.30	89.90	95.00	90.78	94.92	90.91	95.24	89.63	95.03
110	89.93	95.30	89.82	95.00	90.59	94.92	90.19	95.24	90.78	95.06
111	89.15	95.24	88.94	94.98	86.96	94.95	85.28	95.35	88.30	95.06
112	88.62	95.27	89.85	95.03	91.05	94.98	89.82	95.35	88.62	94.98
113	12.93	95.30	88.51	95.03	90.33	95.00	90.83	95.40	87.68	94.92
114	88.43	95.27	89.34	95.06	87.33	95.03	89.31	95.40	90.91	94.95
115	89.20	95.32	89.28	95.08	90.19	95.03	88.64	95.46	90.49	95.06
116	87.49	95.35	90.81	95.11	90.51	95.08	89.82	95.46	86.69	95.08
117	88.05	95.27	86.93	95.03	89.55	95.08	88.40	95.48	88.32	95.08
118	90.35	95.35	89.36	95.11	90.41	95.19	89.95	95.56	89.04	95.06
119	88.46	95.24	88.78	95.08	89.04	95.16	90.41	95.56	89.74	95.08
120	89.04	95.30	90.97	95.11	89.87	95.19	88.51	95.64	87.01	95.08
121	91.61	95.30	84.47	95.11	89.20	95.14	90.51	95.64	89.10	95.11
122	89.20	95.27	89.04	95.11	91.15	95.19	88.16	95.67	90.35	95.08
123	88.75	95.24	90.33	95.11	90.54	95.19	89.31	95.64	89.47	95.11
124	90.57	95.22	89.10	95.11	89.39	95.19	89.23	95.64	90.33	95.16
125	90.33	95.24	89.23	95.16	90.43	95.24	89.79	95.64	88.62	95.22
126	91.69	95.30	88.78	95.16	89.93	95.27	91.07	95.67	89.71	95.16
127	90.03	95.30	89.79	95.14	89.52	95.24	89.79	95.64	89.42	95.19
128	90.62	95.30	88.75	95.22	90.01	95.22	90.91	95.62	87.31	95.14
129	91.10	95.30	89.76	95.19	90.57	95.27	86.32	95.64	89.93	95.14
130	89.60	95.24	91.10	95.22	90.06	95.32	90.43	95.64	85.94	95.14
131	87.39	95.27	87.55	95.22	90.33	95.30	89.74	95.67	90.62	95.24
132	89.60	95.27	89.66	95.24	89.28	95.30	89.90	95.70	88.51	95.19
133	91.02	95.32	89.31	95.24	87.55	95.32	91.18	95.70	88.78	95.19
134	90.46	95.27	90.59	95.24	89.58	95.32	90.81	95.72	88.19	95.16
135	90.59	95.24	87.95	95.24	89.28	95.30	88.51	95.75	89.04	95.22
136	89.28	95.30	90.51	95.30	86.83	95.32	90.38	95.64	91.56	95.22
137	89.47	95.32	90.54	95.32	90.25	95.30	89.23	95.67	89.55	95.19
138	90.06	95.35	91.58	95.35	89.52	95.38	89.74	95.70	87.92	95.14
139	90.70	95.40	89.82	95.32	90.25	95.40	89.18	95.70	91.72	95.14
140	89.90	95.38	91.42	95.27	89.68	95.43	90.38	95.70	90.03	95.16
141	90.27	95.43	88.80	95.24	91.29	95.40	90.62	95.80	88.72	95.19

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
142	90.11	95.43	90.89	95.27	89.55	95.48	90.57	95.78	89.95	95.16
143	88.00	95.48	91.26	95.35	81.69	95.46	90.17	95.78	87.47	95.19
144	88.54	95.43	90.06	95.35	90.22	95.43	89.07	95.67	89.50	95.16
145	89.18	95.40	90.59	95.38	89.71	95.43	90.38	95.70	89.36	95.16
146	88.91	95.40	84.23	95.35	90.01	95.46	89.66	95.78	90.38	95.19
147	91.34	95.38	90.78	95.32	89.02	95.46	89.55	95.75	90.33	95.22
148	89.50	95.38	91.21	95.40	90.62	95.46	89.98	95.75	89.66	95.19
149	91.31	95.43	91.10	95.40	87.28	95.43	89.60	95.75	89.60	95.22
150	94.44	95.46	94.68	95.46	94.25	95.48	94.04	95.91	94.36	95.16
151	94.84	95.48	94.84	95.43	94.33	95.51	94.36	95.96	94.76	95.16
152	94.79	95.54	95.00	95.51	94.39	95.48	94.71	95.94	94.49	95.16
153	95.03	95.67	95.06	95.43	94.33	95.40	94.95	95.91	94.55	95.16
154	95.08	95.67	95.11	95.46	94.17	95.38	94.87	95.91	94.76	95.19
155	95.03	95.75	95.27	95.48	94.36	95.38	94.82	95.80	94.60	95.11
156	95.30	95.78	95.43	95.48	94.55	95.43	94.92	95.80	94.87	95.11
157	95.38	95.80	95.48	95.51	94.41	95.43	95.03	95.86	94.68	95.11
158	95.19	95.83	95.30	95.54	94.76	95.40	95.11	95.88	94.90	95.14
159	95.19	95.94	95.35	95.51	94.49	95.43	94.92	95.83	95.16	95.14
160	95.27	95.99	95.22	95.54	94.60	95.46	95.51	95.88	95.03	95.14
161	95.43	95.96	95.22	95.54	94.74	95.48	95.30	95.91	95.11	95.19
162	95.40	95.96	95.27	95.48	94.74	95.46	95.24	95.88	95.06	95.22
163	95.32	95.96	95.32	95.62	94.60	95.46	95.40	95.83	95.14	95.24
164	95.16	95.99	95.06	95.64	94.71	95.40	95.14	95.80	94.95	95.24
165	95.32	96.02	95.27	95.64	94.68	95.46	95.14	95.88	94.92	95.27
166	95.59	96.04	95.59	95.72	94.66	95.46	95.27	95.80	95.06	95.32
167	95.32	95.99	95.40	95.75	94.79	95.51	95.22	95.83	94.98	95.35
168	95.19	96.02	95.54	95.80	94.87	95.51	95.00	95.88	94.98	95.32
169	95.46	96.04	95.51	95.83	94.90	95.54	95.51	95.91	94.87	95.32
170	95.14	96.02	95.72	95.86	94.79	95.51	95.32	95.96	95.03	95.40
171	95.48	96.04	95.56	95.86	94.82	95.59	95.38	95.99	94.98	95.46
172	95.59	96.04	95.24	95.83	94.79	95.59	95.16	95.99	95.16	95.43
173	95.54	96.07	95.54	95.86	94.90	95.62	95.51	96.02	94.92	95.54
174	95.46	96.07	95.64	95.86	95.16	95.62	95.24	95.96	95.27	95.56
175	95.51	96.04	95.72	95.88	95.11	95.64	95.22	95.99	94.95	95.54
176	95.32	96.02	95.46	95.88	94.90	95.62	95.32	95.96	95.11	95.56
177	95.46	95.99	95.43	95.88	94.76	95.62	95.22	95.96	95.08	95.54
178	95.51	95.99	95.67	95.91	94.92	95.59	95.43	95.99	95.03	95.56
179	95.56	95.96	95.70	95.88	94.71	95.62	95.30	95.99	95.03	95.64
180	95.43	95.99	95.78	95.88	94.74	95.62	95.40	95.99	95.08	95.62
181	95.48	95.94	95.80	95.94	94.95	95.70	95.46	96.02	95.19	95.67
182	95.38	95.94	95.54	95.94	95.00	95.67	95.51	95.96	94.98	95.64
183	95.27	95.94	95.75	95.96	94.95	95.67	95.27	95.88	95.03	95.64
184	95.48	95.91	95.70	95.96	94.98	95.70	95.40	95.88	95.06	95.64
185	95.78	95.91	95.64	95.94	94.79	95.70	95.48	95.86	95.35	95.64
186	95.48	95.91	95.48	95.94	95.00	95.70	95.32	95.91	95.22	95.67
187	95.48	95.94	95.75	95.94	94.74	95.70	95.30	95.91	95.24	95.72
188	95.35	95.91	95.40	95.96	94.79	95.67	95.19	95.94	95.32	95.72
189	95.48	95.91	95.64	95.94	94.87	95.70	95.30	95.91	95.30	95.75
190	95.35	95.94	95.67	95.94	95.11	95.70	95.24	95.94	95.19	95.78
191	95.54	95.96	95.54	95.96	94.84	95.67	95.22	95.91	95.16	95.72
192	95.48	95.99	95.64	95.96	94.87	95.67	95.46	95.94	95.22	95.70

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
193	95.75	95.99	95.72	95.94	95.06	95.64	95.67	95.94	95.27	95.67
194	95.75	96.02	95.51	95.99	94.74	95.64	95.51	95.94	94.98	95.72
195	95.94	96.02	95.59	95.96	94.92	95.70	95.14	95.94	95.19	95.72
196	95.83	96.02	95.78	95.99	95.00	95.70	95.27	95.94	95.35	95.72
197	95.46	95.99	95.96	95.99	95.19	95.70	95.54	95.91	95.06	95.72
198	95.48	95.96	95.88	95.99	94.98	95.72	95.32	95.94	95.06	95.75
199	95.24	95.94	95.80	95.99	95.00	95.75	95.40	95.94	94.95	95.75
200	95.43	95.91	95.70	95.96	95.11	95.75	95.54	95.91	95.14	95.72
201	95.38	95.94	95.75	95.96	94.95	95.70	95.30	95.88	95.06	95.75
202	95.48	95.94	95.80	95.96	95.22	95.70	95.51	95.91	95.11	95.78
203	95.48	95.96	95.80	95.96	95.00	95.70	95.35	95.91	95.35	95.78
204	95.46	95.99	95.88	95.96	94.98	95.70	95.35	95.91	95.35	95.78
205	95.62	96.02	95.78	95.96	95.16	95.70	95.64	95.91	95.35	95.78
206	95.56	96.04	95.78	95.96	94.95	95.67	95.51	95.88	95.14	95.78
207	95.54	96.04	95.88	95.96	95.24	95.64	95.56	95.94	95.22	95.83
208	95.56	96.02	95.80	95.96	95.03	95.64	95.56	95.94	95.14	95.80
209	95.80	96.02	95.80	95.96	95.14	95.64	95.51	95.99	95.14	95.83
210	95.56	96.02	95.64	95.91	94.98	95.67	95.56	95.99	95.06	95.83
211	95.59	96.02	95.80	95.91	95.19	95.59	95.67	95.91	95.22	95.86
212	95.56	95.99	95.80	95.91	95.11	95.62	95.40	95.91	95.24	95.86
213	95.72	95.99	95.83	95.91	95.22	95.54	95.80	95.91	95.35	95.83
214	95.75	95.99	95.75	95.86	95.19	95.54	95.56	95.91	95.24	95.78
215	95.75	95.99	95.75	95.86	95.24	95.51	95.64	95.91	95.38	95.75
216	95.59	95.99	95.78	95.88	95.08	95.51	95.59	95.88	95.19	95.72
217	95.80	95.99	95.94	95.91	95.08	95.48	95.51	95.88	95.30	95.72
218	95.54	95.99	95.70	95.88	95.11	95.48	95.48	95.88	95.24	95.72
219	95.72	95.99	95.70	95.91	95.22	95.48	95.72	95.91	95.27	95.72
220	95.64	95.99	95.83	95.96	95.16	95.54	95.16	95.88	95.38	95.70
221	95.72	95.99	95.70	95.94	95.03	95.54	95.46	95.88	95.14	95.72
222	95.59	95.99	95.67	95.96	95.06	95.54	95.51	95.88	95.43	95.72
223	95.67	96.02	95.75	95.96	95.19	95.54	95.70	95.88	95.27	95.72
224	95.72	96.02	95.86	95.94	95.24	95.54	95.43	95.88	95.16	95.67
225	95.75	96.04	95.75	95.96	95.11	95.54	95.64	95.91	95.19	95.67
226	95.62	96.04	95.86	95.96	95.16	95.54	95.51	95.91	95.14	95.64
227	95.80	96.04	95.88	95.96	94.98	95.54	95.59	95.91	95.11	95.64
228	95.59	96.04	95.96	95.96	95.22	95.51	95.54	95.91	95.19	95.64
229	95.32	96.04	95.67	95.94	95.08	95.51	95.56	95.91	95.38	95.62
230	95.56	96.04	95.88	95.96	95.24	95.51	95.54	95.91	95.32	95.59
231	95.64	96.04	95.70	95.96	95.06	95.48	95.56	95.91	95.43	95.59
232	95.59	96.04	96.02	95.96	95.19	95.48	95.56	95.91	95.11	95.59
233	95.59	96.04	95.86	95.99	95.08	95.48	95.67	95.88	95.27	95.59
234	95.54	96.04	95.70	95.96	95.11	95.48	95.48	95.88	95.19	95.59
235	95.64	96.04	95.86	95.96	95.16	95.48	95.59	95.86	95.27	95.56
236	95.54	96.02	95.86	95.94	95.24	95.46	95.72	95.86	95.19	95.56
237	95.59	96.04	95.86	95.94	95.03	95.43	95.54	95.86	95.24	95.56
238	95.70	96.04	95.62	95.94	95.03	95.46	95.46	95.86	95.27	95.56
239	95.80	96.04	95.75	95.96	94.98	95.46	95.62	95.86	95.30	95.56
240	95.64	96.04	95.83	95.96	95.08	95.46	95.46	95.88	95.16	95.56
241	95.59	96.04	95.75	95.96	95.16	95.43	95.51	95.88	95.19	95.56
242	95.70	96.07	95.75	95.96	95.06	95.43	95.46	95.88	95.40	95.56
243	95.86	96.10	95.67	95.99	94.90	95.40	95.67	95.88	95.14	95.56

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
244	95.72	96.10	95.80	95.96	95.16	95.38	95.43	95.88	95.22	95.56
245	95.83	96.10	95.75	95.99	95.32	95.38	95.67	95.88	95.38	95.56
246	95.54	96.10	95.67	95.99	95.27	95.38	95.40	95.88	95.51	95.56
247	95.48	96.10	95.86	95.96	95.22	95.38	95.48	95.88	95.24	95.56
248	95.51	96.10	95.64	95.96	95.35	95.38	95.67	95.88	95.35	95.56
249	95.48	96.07	95.78	95.96	95.03	95.38	95.59	95.86	95.32	95.56

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
0	15.50	15.50	14.48	14.48	13.20	13.20	13.04	13.04	11.49	11.49
1	35.11	34.87	38.19	38.16	36.53	35.97	35.70	35.62	34.90	33.54
2	25.76	33.81	59.41	56.87	49.09	48.48	58.50	56.89	47.97	47.46
3	66.92	64.08	64.91	66.03	59.51	58.26	68.09	67.74	49.47	53.71
4	72.18	71.06	70.31	73.94	74.53	73.54	69.56	73.73	71.51	69.48
5	73.17	77.71	70.87	79.08	75.65	77.93	70.07	78.14	72.31	75.07
6	75.17	81.53	78.35	82.76	78.11	82.10	77.31	81.61	79.08	80.41
7	76.06	84.42	80.52	85.49	80.49	84.50	76.51	83.91	75.12	82.95
8	78.73	85.44	82.15	86.96	78.83	86.10	80.33	85.49	77.36	84.71
9	78.78	87.01	80.28	87.49	81.91	86.96	82.36	86.13	78.59	86.13
10	80.89	88.11	83.32	88.70	83.00	88.38	83.46	87.17	84.69	87.73
11	81.61	88.64	85.22	89.39	86.83	89.12	84.26	88.16	79.10	88.32
12	82.63	89.68	81.69	89.76	86.21	90.17	79.82	88.43	83.81	89.10
13	81.35	89.90	84.58	90.25	83.51	90.38	82.90	89.36	81.00	89.68
14	77.85	90.49	85.36	90.81	86.48	91.07	82.92	89.76	82.04	89.76
15	81.69	90.67	83.16	91.23	81.67	91.18	81.16	89.90	83.94	90.54
16	83.06	90.97	74.00	91.07	81.21	91.58	80.87	90.38	83.06	90.73
17	60.96	91.15	78.41	91.37	83.97	91.66	80.73	90.43	87.57	91.31
18	80.38	91.29	84.61	91.50	84.58	91.77	78.75	90.51	82.90	91.26
19	83.11	91.56	81.56	91.74	84.63	92.06	82.55	90.91	83.54	91.77
20	84.29	91.72	85.70	91.77	84.18	92.17	84.61	91.31	83.22	91.98
21	87.76	91.96	85.44	91.96	86.56	92.41	84.34	91.58	74.53	92.04
22	84.90	91.93	86.53	91.80	51.50	92.46	85.97	91.72	86.29	92.52
23	83.30	92.04	84.05	92.09	86.08	92.68	76.59	91.82	86.32	92.52
24	85.09	92.12	86.96	92.28	87.55	92.89	86.48	91.88	87.09	92.81
25	86.13	92.41	86.69	92.68	87.73	93.08	84.15	92.38	86.34	92.94
26	87.63	92.70	85.62	92.60	86.53	93.19	86.99	92.41	88.00	93.11
27	87.52	92.65	87.33	92.78	86.56	93.27	85.97	92.49	87.68	93.24
28	86.72	92.73	81.51	92.86	87.87	93.48	86.40	92.52	88.94	93.45
29	86.58	93.00	87.31	93.05	84.79	93.48	89.15	92.57	88.78	93.53
30	83.91	93.16	87.09	93.21	88.54	93.45	88.56	92.76	87.28	93.64
31	85.44	93.29	85.17	93.27	89.36	93.51	88.99	92.84	85.54	93.61
32	87.57	93.37	89.18	93.56	88.30	93.56	85.30	92.76	85.14	93.59
33	85.22	93.32	87.68	93.53	86.64	93.51	87.95	92.92	86.45	93.59
34	88.64	93.51	89.26	93.59	87.89	93.56	87.33	92.89	85.20	93.77
35	87.41	93.51	85.94	93.56	87.71	93.56	84.93	93.00	86.53	93.61
36	89.44	93.48	88.00	93.69	89.44	93.67	88.08	93.16	84.18	93.69
37	86.96	93.53	86.21	93.72	86.40	93.59	88.27	93.19	87.63	93.80
38	88.96	93.48	88.16	93.88	88.80	93.64	88.00	93.08	89.47	93.83
39	88.08	93.37	86.93	93.96	86.93	93.75	87.49	93.19	89.39	93.83

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
40	87.76	93.45	86.58	93.91	88.13	93.67	86.00	93.21	85.41	93.96
41	88.67	93.67	89.31	93.88	86.99	93.72	87.20	93.45	88.99	93.91
42	86.42	93.64	89.10	93.99	88.96	93.69	89.76	93.64	86.80	94.01
43	85.73	93.77	83.99	94.07	87.25	93.72	89.07	93.69	85.92	93.91
44	84.58	93.83	87.33	94.07	87.95	93.67	85.76	93.72	88.19	93.93
45	86.64	93.85	87.04	93.85	86.21	93.80	87.39	93.72	90.35	93.91
46	87.41	93.80	87.87	94.07	86.66	93.80	86.53	93.83	86.24	94.04
47	88.05	93.85	88.19	94.07	88.38	93.91	87.33	93.93	88.30	93.99
48	81.96	94.01	87.97	94.20	89.98	94.01	88.16	93.99	88.46	94.04
49	88.78	94.04	85.81	94.17	86.26	94.12	86.18	94.04	86.48	94.17
50	70.92	94.07	89.12	94.31	87.44	94.23	86.93	94.07	89.34	94.17
51	88.67	94.07	88.08	94.36	87.49	94.20	87.47	94.09	87.81	94.28
52	86.69	94.23	87.63	94.39	88.38	94.15	88.91	94.12	90.62	94.20
53	89.10	94.09	86.75	94.39	87.20	94.23	89.85	94.12	87.60	94.33
54	89.52	94.12	86.61	94.39	86.66	94.31	89.12	94.28	85.28	94.33
55	89.18	94.12	87.41	94.39	89.60	94.25	88.91	94.17	87.81	94.33
56	88.96	94.17	81.77	94.44	89.31	94.41	88.32	94.25	89.74	94.41
57	89.52	94.17	86.53	94.52	86.69	94.33	89.50	94.33	87.17	94.52
58	89.23	94.15	88.80	94.60	86.69	94.36	87.87	94.28	87.65	94.41
59	86.93	94.23	87.09	94.60	87.65	94.33	87.52	94.31	88.35	94.49
60	87.97	94.23	88.96	94.47	85.28	94.36	90.78	94.31	87.92	94.52
61	89.82	94.33	86.00	94.44	88.27	94.36	86.56	94.36	88.32	94.49
62	88.35	94.31	88.05	94.63	88.56	94.25	88.72	94.41	85.73	94.49
63	88.24	94.31	88.62	94.47	88.78	94.47	91.21	94.47	87.65	94.47
64	88.32	94.47	84.98	94.52	87.44	94.52	89.52	94.44	89.12	94.49
65	88.75	94.36	89.18	94.55	84.29	94.52	88.43	94.36	89.52	94.44
66	89.31	94.41	87.17	94.60	88.00	94.55	90.30	94.47	89.10	94.58
67	87.57	94.44	88.48	94.60	89.90	94.52	89.36	94.49	89.18	94.52
68	88.94	94.44	89.71	94.66	87.84	94.47	91.07	94.63	88.56	94.58
69	88.86	94.47	89.15	94.60	90.01	94.52	88.99	94.71	89.63	94.55
70	86.29	94.44	87.81	94.60	87.81	94.47	89.07	94.68	88.03	94.55
71	83.27	94.41	86.16	94.60	86.16	94.55	89.63	94.76	88.70	94.58
72	87.76	94.49	89.63	94.68	87.87	94.58	88.40	94.71	88.75	94.60
73	87.60	94.44	90.03	94.68	88.80	94.58	89.60	94.76	89.12	94.71
74	90.22	94.52	88.43	94.66	89.66	94.66	86.10	94.68	90.35	94.76
75	89.93	94.44	89.82	94.58	88.94	94.66	87.84	94.63	88.88	94.76
76	90.57	94.44	87.15	94.58	88.48	94.58	89.36	94.66	89.76	94.76
77	87.31	94.55	87.79	94.58	88.03	94.60	88.03	94.68	88.38	94.82
78	90.65	94.58	89.20	94.71	88.72	94.55	90.30	94.68	85.20	94.84
79	90.22	94.55	83.62	94.74	88.08	94.55	88.51	94.74	88.35	94.82
80	90.67	94.58	90.43	94.76	86.91	94.55	88.78	94.84	88.30	94.84
81	85.06	94.58	89.63	94.68	90.83	94.60	90.25	94.76	86.26	94.90
82	86.37	94.55	89.31	94.74	90.27	94.60	92.25	94.79	89.20	94.90
83	87.20	94.52	88.30	94.76	89.12	94.66	90.38	94.90	86.88	94.92
84	90.86	94.52	87.44	94.82	87.71	94.66	86.77	94.92	86.08	95.03
85	88.70	94.58	87.68	94.74	89.66	94.74	89.47	94.95	87.57	95.14
86	89.23	94.49	89.95	94.71	89.79	94.71	87.68	94.98	90.89	95.11
87	89.20	94.58	89.36	94.76	90.33	94.74	89.50	95.00	89.71	95.14
88	88.99	94.63	85.49	94.63	89.58	94.74	91.58	95.00	89.44	95.06
89	89.52	94.74	87.31	94.68	76.59	94.68	88.86	94.98	88.88	95.16
90	87.63	94.76	87.52	94.71	90.17	94.76	89.36	95.00	90.25	95.19

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
91	91.40	94.74	89.34	94.74	87.87	94.82	89.68	95.00	88.19	95.24
92	90.03	94.90	88.43	94.82	90.14	94.71	87.57	95.00	85.86	95.22
93	89.02	94.82	88.21	94.79	87.87	94.82	88.78	95.03	88.38	95.30
94	88.83	94.87	88.38	94.76	89.02	94.76	89.04	95.08	89.28	95.27
95	87.52	94.95	88.91	94.74	87.57	94.79	90.27	95.08	90.25	95.24
96	90.33	94.95	89.39	94.74	88.35	94.76	88.59	95.11	88.99	95.27
97	88.67	94.95	88.24	94.74	89.42	94.87	89.58	95.08	89.93	95.27
98	89.85	94.95	88.16	94.76	87.17	94.84	87.12	95.11	89.82	95.24
99	90.01	94.92	89.31	94.74	88.46	94.82	86.66	95.11	89.39	95.27
100	89.58	95.00	89.42	94.76	89.58	94.87	86.77	95.08	91.23	95.27
101	91.05	94.98	89.74	94.84	88.75	94.98	90.81	95.16	87.95	95.30
102	87.71	94.98	89.52	94.79	86.66	94.98	87.63	95.11	88.78	95.32
103	88.94	95.03	87.87	94.84	89.63	94.95	89.10	95.19	88.35	95.35
104	89.58	95.03	88.48	94.87	88.05	94.92	88.83	95.11	89.34	95.38
105	90.01	95.03	89.52	94.90	88.43	94.92	90.03	95.22	89.28	95.35
106	87.12	95.11	90.75	94.82	91.02	95.03	90.25	95.16	90.59	95.32
107	90.51	95.08	90.09	94.79	89.47	95.00	90.65	95.16	88.59	95.35
108	88.70	95.16	87.04	94.84	91.05	95.11	89.74	95.16	90.83	95.40
109	90.19	95.14	90.22	94.90	87.55	95.11	90.09	95.16	83.30	95.40
110	87.87	95.16	88.70	94.87	89.36	95.11	90.30	95.22	88.96	95.30
111	91.61	95.14	90.75	94.87	88.27	95.16	89.85	95.19	88.80	95.32
112	91.26	95.22	88.64	94.90	89.12	95.16	88.70	95.19	88.35	95.27
113	87.12	95.30	89.79	94.87	90.54	95.22	89.95	95.14	88.62	95.30
114	87.15	95.22	88.99	94.87	91.50	95.19	91.29	95.16	91.07	95.32
115	90.38	95.24	89.66	94.90	88.67	95.22	87.20	95.19	87.92	95.35
116	89.87	95.22	90.43	94.92	89.12	95.24	89.42	95.22	88.86	95.35
117	87.71	95.22	87.23	94.87	90.73	95.27	90.70	95.22	90.57	95.38
118	89.26	95.22	87.07	94.95	87.76	95.24	89.39	95.16	88.13	95.40
119	87.12	95.16	90.30	94.90	88.70	95.32	91.82	95.14	89.20	95.35
120	88.88	95.16	90.43	94.90	89.76	95.38	89.60	95.22	88.56	95.32
121	90.78	95.19	89.12	94.98	90.19	95.38	89.18	95.30	91.15	95.35
122	90.46	95.22	89.93	94.98	88.21	95.30	90.70	95.30	91.29	95.35
123	91.02	95.27	89.63	94.92	87.20	95.38	89.07	95.27	90.54	95.43
124	89.26	95.22	90.54	94.90	90.78	95.30	89.28	95.24	89.68	95.40
125	90.43	95.22	90.41	94.90	89.87	95.35	91.15	95.30	88.13	95.43
126	90.01	95.14	88.94	94.98	89.20	95.38	89.74	95.30	88.86	95.46
127	87.97	95.11	90.49	94.95	89.18	95.40	90.86	95.32	91.07	95.40
128	87.65	95.14	91.50	94.98	88.80	95.40	89.31	95.27	87.33	95.46
129	88.13	95.14	90.67	94.95	90.46	95.35	90.25	95.38	90.97	95.48
130	87.23	95.14	89.95	94.98	87.81	95.40	91.15	95.38	90.38	95.51
131	89.68	95.14	90.67	95.00	89.95	95.43	90.89	95.43	90.73	95.46
132	89.79	95.11	89.87	95.00	90.27	95.43	89.15	95.43	89.18	95.46
133	90.33	95.14	91.42	95.00	89.10	95.43	88.83	95.51	88.62	95.43
134	90.01	95.11	90.62	95.03	87.33	95.48	90.38	95.51	90.35	95.46
135	89.04	95.19	90.51	95.03	87.73	95.48	87.97	95.54	90.35	95.40
136	89.71	95.14	86.32	95.08	89.55	95.56	88.78	95.54	91.74	95.40
137	90.70	95.16	90.75	95.08	90.41	95.48	90.78	95.56	89.58	95.46
138	90.38	95.19	90.59	95.06	90.14	95.54	89.26	95.51	88.96	95.48
139	88.16	95.19	91.10	95.00	88.96	95.51	90.91	95.54	89.23	95.51
140	89.74	95.16	89.74	95.06	88.96	95.54	90.89	95.54	86.64	95.43
141	89.07	95.16	88.83	95.14	90.46	95.51	87.87	95.59	89.02	95.51

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
142	88.96	95.16	89.79	95.11	88.48	95.59	89.20	95.59	91.40	95.51
143	89.55	95.14	89.82	95.14	91.23	95.51	89.98	95.54	86.99	95.51
144	89.74	95.14	88.24	95.11	88.38	95.51	87.68	95.56	88.91	95.51
145	88.86	95.14	90.09	95.11	89.60	95.46	91.05	95.54	90.51	95.51
146	87.60	95.19	89.85	95.08	90.70	95.51	89.90	95.56	89.39	95.48
147	89.82	95.19	90.70	95.11	90.49	95.46	91.15	95.54	90.91	95.48
148	87.44	95.22	89.10	95.11	90.57	95.43	90.65	95.56	89.60	95.48
149	89.55	95.19	89.87	95.14	89.60	95.40	88.86	95.56	90.83	95.48
150	94.44	95.27	94.20	95.16	93.96	95.54	94.17	95.59	94.20	95.48
151	94.47	95.32	93.88	95.11	94.25	95.54	94.47	95.59	94.49	95.51
152	94.84	95.30	94.17	95.11	94.36	95.59	94.63	95.59	94.71	95.56
153	95.16	95.40	94.44	95.19	94.58	95.62	94.98	95.62	94.63	95.59
154	94.92	95.40	94.58	95.30	94.58	95.64	94.92	95.59	94.66	95.67
155	94.92	95.35	94.33	95.32	94.47	95.67	94.98	95.62	95.08	95.72
156	95.08	95.32	94.63	95.35	94.63	95.62	95.00	95.59	94.66	95.72
157	94.95	95.35	94.66	95.43	94.52	95.64	95.14	95.62	95.27	95.72
158	95.14	95.38	94.66	95.48	94.74	95.64	95.11	95.64	95.19	95.70
159	95.00	95.40	94.79	95.54	94.74	95.56	95.03	95.62	95.03	95.75
160	95.14	95.43	94.84	95.51	94.63	95.51	95.16	95.64	95.14	95.72
161	95.14	95.56	94.82	95.56	94.92	95.51	95.30	95.70	95.00	95.72
162	95.35	95.59	94.82	95.62	94.74	95.56	95.00	95.70	94.87	95.72
163	95.19	95.62	94.60	95.59	95.03	95.62	95.16	95.64	95.16	95.72
164	94.98	95.64	95.06	95.64	94.68	95.56	95.08	95.64	94.98	95.78
165	95.11	95.62	94.90	95.59	94.84	95.54	95.03	95.62	95.19	95.86
166	95.11	95.59	95.00	95.56	94.90	95.56	95.11	95.70	94.74	95.86
167	95.22	95.59	95.00	95.54	95.08	95.59	95.06	95.75	95.03	95.88
168	95.06	95.56	95.06	95.54	94.87	95.54	95.06	95.75	95.08	95.96
169	95.24	95.62	95.06	95.54	95.24	95.56	94.92	95.62	94.84	95.96
170	95.38	95.62	94.95	95.56	94.84	95.56	95.06	95.62	95.27	95.96
171	95.14	95.67	95.00	95.56	94.95	95.56	94.95	95.54	95.16	95.96
172	94.87	95.70	95.22	95.62	94.66	95.62	95.22	95.59	95.08	96.02
173	95.27	95.67	95.16	95.56	94.84	95.56	95.08	95.62	95.08	95.96
174	95.46	95.62	95.00	95.51	94.66	95.56	94.95	95.62	95.24	95.96
175	95.48	95.59	95.32	95.51	94.79	95.56	95.08	95.64	95.32	95.96
176	95.40	95.59	95.19	95.54	95.06	95.54	95.19	95.62	95.03	95.96
177	95.48	95.59	95.08	95.56	95.24	95.51	95.35	95.70	94.98	95.96
178	95.22	95.56	94.92	95.48	94.84	95.54	95.19	95.72	95.22	95.99
179	95.46	95.54	95.27	95.46	94.76	95.54	95.11	95.72	94.90	95.99
180	95.30	95.54	95.43	95.46	94.98	95.54	95.24	95.72	95.16	95.96
181	95.30	95.56	95.32	95.46	94.90	95.54	95.30	95.75	95.27	95.99
182	95.40	95.59	95.14	95.43	95.06	95.59	95.30	95.80	95.22	96.02
183	95.46	95.59	95.27	95.48	95.06	95.54	95.27	95.78	95.19	96.04
184	95.19	95.59	95.22	95.48	95.16	95.54	95.19	95.75	95.35	96.04
185	95.40	95.64	95.32	95.51	95.06	95.56	95.38	95.75	95.19	96.04
186	95.27	95.59	95.16	95.54	95.35	95.56	95.40	95.78	95.35	96.04
187	95.24	95.64	95.22	95.54	95.03	95.56	95.35	95.78	95.00	96.04
188	95.30	95.59	95.14	95.48	95.19	95.56	95.27	95.78	94.95	96.04
189	95.22	95.67	95.16	95.56	95.22	95.56	95.64	95.78	95.11	96.04
190	95.46	95.64	94.87	95.51	95.06	95.56	95.38	95.75	95.24	96.04
191	95.27	95.67	95.30	95.54	95.22	95.54	95.40	95.70	95.22	96.04
192	95.38	95.67	95.16	95.54	95.11	95.48	95.24	95.70	95.03	96.04

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
193	95.27	95.67	95.16	95.56	95.14	95.51	95.43	95.70	95.00	96.04
194	95.59	95.72	95.06	95.59	95.03	95.51	95.51	95.70	95.11	96.04
195	95.35	95.75	95.22	95.54	95.00	95.51	95.46	95.72	95.38	96.04
196	95.40	95.80	95.03	95.54	94.84	95.51	95.46	95.70	95.46	96.04
197	95.32	95.78	95.27	95.54	95.11	95.51	95.35	95.70	95.14	96.04
198	95.30	95.75	95.27	95.56	95.40	95.51	95.40	95.67	95.11	96.04
199	95.32	95.75	95.14	95.56	94.98	95.54	95.32	95.67	94.90	96.04
200	95.32	95.70	95.30	95.51	95.14	95.54	95.27	95.67	95.16	96.04
201	95.32	95.70	95.03	95.54	95.08	95.54	95.32	95.70	95.06	96.02
202	95.43	95.70	95.27	95.51	95.22	95.51	95.24	95.70	95.24	96.04
203	95.27	95.64	95.35	95.54	95.11	95.51	95.40	95.70	95.22	96.02
204	95.43	95.64	95.19	95.59	95.30	95.51	95.22	95.72	95.03	96.02
205	95.38	95.62	95.06	95.56	95.32	95.51	95.46	95.70	95.43	96.02
206	95.32	95.62	95.38	95.56	95.24	95.48	95.38	95.70	95.19	96.02
207	95.27	95.59	95.30	95.56	95.16	95.48	95.30	95.70	95.40	96.02
208	95.51	95.56	95.35	95.56	95.14	95.46	95.43	95.70	95.11	95.99
209	95.38	95.62	95.27	95.54	95.30	95.46	95.48	95.64	95.16	95.96
210	95.27	95.62	95.14	95.56	95.30	95.46	95.27	95.64	95.38	95.94
211	95.43	95.56	95.27	95.54	95.32	95.43	95.38	95.62	95.38	95.91
212	95.32	95.59	95.16	95.54	95.35	95.46	95.51	95.59	95.32	95.91
213	95.32	95.62	95.51	95.59	95.14	95.46	95.46	95.56	94.98	95.91
214	95.22	95.64	95.38	95.59	95.24	95.46	95.35	95.54	95.51	95.88
215	95.27	95.64	95.27	95.62	95.30	95.46	95.40	95.56	95.16	95.88
216	95.30	95.64	95.46	95.62	95.30	95.46	95.48	95.56	95.16	95.83
217	95.35	95.62	95.32	95.62	95.16	95.46	95.48	95.56	95.32	95.86
218	95.24	95.62	95.46	95.64	95.14	95.46	95.62	95.56	95.46	95.80
219	95.30	95.62	95.67	95.64	95.24	95.46	95.46	95.56	95.27	95.78
220	95.30	95.59	95.67	95.62	95.22	95.46	95.43	95.59	95.40	95.80
221	95.32	95.59	95.40	95.62	95.30	95.46	95.43	95.56	95.08	95.78
222	95.30	95.59	95.56	95.59	94.98	95.46	95.51	95.54	95.35	95.80
223	95.32	95.62	95.46	95.62	95.40	95.48	95.62	95.54	95.51	95.80
224	95.43	95.62	95.43	95.59	95.27	95.48	95.40	95.54	95.22	95.80
225	95.35	95.62	95.64	95.59	95.27	95.51	95.56	95.56	95.19	95.75
226	95.27	95.62	95.38	95.56	95.30	95.51	95.62	95.56	95.40	95.75
227	95.32	95.64	95.54	95.62	95.32	95.51	95.38	95.56	95.19	95.72
228	95.27	95.62	95.59	95.67	95.30	95.51	95.54	95.54	95.38	95.75
229	95.27	95.59	95.59	95.67	95.24	95.51	95.38	95.56	95.24	95.72
230	95.30	95.62	95.56	95.64	95.27	95.51	95.56	95.56	95.48	95.72
231	95.54	95.56	95.40	95.64	95.30	95.51	95.48	95.56	95.35	95.72
232	95.56	95.59	95.56	95.62	95.38	95.51	95.46	95.56	95.40	95.75
233	95.22	95.56	95.51	95.64	95.35	95.51	95.54	95.56	95.32	95.75
234	95.54	95.59	95.67	95.64	95.24	95.51	95.56	95.56	95.11	95.75
235	95.40	95.56	95.64	95.64	95.32	95.51	95.62	95.56	95.30	95.72
236	95.62	95.59	95.46	95.64	95.03	95.51	95.59	95.56	95.24	95.70
237	95.43	95.56	95.59	95.64	95.19	95.54	95.56	95.59	95.19	95.70
238	95.40	95.54	95.46	95.67	95.35	95.54	95.56	95.59	95.24	95.72
239	95.43	95.54	95.32	95.62	95.38	95.54	95.46	95.56	95.32	95.72
240	95.40	95.56	95.46	95.62	95.24	95.54	95.32	95.62	95.27	95.72
241	95.46	95.51	95.27	95.62	95.32	95.56	95.51	95.62	95.19	95.72
242	95.32	95.48	95.56	95.62	95.30	95.56	95.54	95.64	95.56	95.72
243	95.35	95.48	95.54	95.62	95.40	95.56	95.40	95.67	95.22	95.72

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
244	95.32	95.48	95.48	95.62	95.11	95.54	95.62	95.64	95.35	95.72
245	95.30	95.48	95.62	95.62	95.48	95.54	95.59	95.64	95.22	95.72
246	95.51	95.48	95.48	95.62	95.38	95.54	95.51	95.64	95.32	95.72
247	95.22	95.51	95.46	95.59	95.22	95.51	95.51	95.64	95.38	95.72
248	95.38	95.51	95.43	95.59	95.32	95.51	95.51	95.64	95.14	95.72
249	95.67	95.51	95.56	95.59	95.38	95.51	95.62	95.64	95.43	95.72

2. Results of recognition rate from Ten trials of Training Bone and Proposed Bone with Random Initial Conditions.

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
0	11.20	11.20	12.00	12.00	11.38	11.38	11.38	11.38	9.78	9.78
1	23.22	23.22	31.67	31.61	28.14	28.06	27.39	27.34	30.71	30.25
2	45.24	44.92	52.78	51.52	47.92	47.22	49.04	46.26	54.06	51.68
3	64.83	62.13	63.55	62.88	64.11	63.44	63.74	63.44	62.03	62.85
4	71.54	70.66	72.45	72.74	73.17	72.82	74.35	72.02	72.29	72.34
5	74.48	75.87	70.50	76.14	74.00	75.41	73.04	76.16	75.49	77.02
6	78.92	80.76	74.96	79.58	78.01	80.38	77.79	80.47	73.81	80.44
7	82.68	84.29	76.62	81.85	79.45	82.84	77.63	82.47	79.77	83.27
8	79.69	86.10	76.96	83.73	80.63	84.47	81.93	84.42	73.68	85.09
9	78.33	87.39	81.72	85.68	77.28	84.87	83.46	86.45	82.12	86.40
10	82.71	88.56	83.54	87.04	83.56	86.13	82.71	87.31	83.06	87.63
11	84.87	89.90	82.76	87.60	83.14	87.57	83.59	87.95	78.11	88.32
12	86.48	90.43	81.64	87.97	81.83	88.13	83.75	89.04	79.93	88.70
13	86.24	90.89	81.53	88.96	83.86	88.64	83.11	89.34	84.31	89.52
14	85.33	91.31	86.02	89.68	82.84	88.78	82.18	89.82	77.53	89.87
15	83.14	91.58	79.82	89.63	80.09	89.34	85.81	90.22	85.36	90.57
16	82.87	91.82	84.71	90.22	86.02	89.74	81.88	90.30	84.90	91.07
17	83.67	92.09	84.77	90.86	82.28	90.35	82.34	90.73	85.78	91.23
18	84.42	92.06	85.49	91.26	83.24	90.38	86.13	91.02	80.49	91.58
19	84.63	92.25	84.37	91.37	83.46	90.65	83.75	91.29	85.06	91.82
20	84.34	92.33	86.80	91.85	86.53	90.99	86.93	91.64	84.47	92.22
21	88.48	92.65	84.15	92.04	86.50	91.42	81.51	91.80	82.58	92.28
22	84.82	92.57	85.11	92.17	88.08	91.72	87.65	92.25	83.35	92.30
23	86.80	92.81	85.57	92.44	87.09	91.82	85.94	92.12	85.52	92.62
24	85.49	93.00	86.56	92.54	87.41	92.12	86.53	92.30	85.11	92.81
25	85.52	92.73	87.28	92.57	84.85	92.28	82.26	92.36	85.57	92.78
26	86.83	92.94	88.72	92.70	87.76	92.44	87.57	92.68	83.30	92.73
27	86.56	93.05	85.30	92.92	87.15	92.46	21.38	92.65	82.79	92.92
28	85.62	93.00	83.86	92.65	87.20	92.57	86.18	92.49	87.63	93.03
29	86.42	93.11	82.44	92.73	87.39	92.54	86.58	92.84	84.13	93.11
30	88.43	93.24	86.02	92.76	87.52	92.89	85.92	92.92	86.80	93.24
31	88.91	93.37	86.85	93.05	85.92	93.08	88.80	93.13	83.65	93.27
32	88.51	93.37	86.72	92.92	87.65	93.13	87.31	93.32	88.56	93.35
33	89.36	93.51	84.82	93.03	88.94	93.27	88.05	93.24	87.41	93.53
34	87.15	93.67	88.56	93.11	88.16	93.43	89.04	93.21	88.67	93.59
35	88.46	93.72	85.86	93.27	85.14	93.56	87.28	93.37	89.07	93.51
36	89.55	93.85	88.27	93.29	88.30	93.56	82.63	93.61	85.70	93.83
37	89.34	93.85	86.93	93.24	65.63	93.59	89.55	93.64	87.81	93.72
38	85.20	94.15	87.41	93.32	88.08	93.53	88.54	93.72	88.03	93.88
39	87.49	94.01	83.24	93.43	86.72	93.59	87.92	93.59	88.05	93.85
40	83.83	94.07	88.75	93.43	88.48	93.56	87.36	93.75	89.66	93.88
41	88.99	94.12	77.15	93.40	88.56	93.69	88.72	93.91	86.75	93.85
42	88.88	94.25	86.99	93.43	89.07	93.80	88.40	93.99	85.52	93.93
43	87.79	94.17	87.71	93.59	88.54	94.07	88.62	94.04	88.00	93.91
44	86.13	94.25	87.65	93.61	88.35	94.15	87.39	94.15	86.26	93.91
45	84.85	94.15	86.93	93.80	89.34	94.23	88.30	94.23	86.88	93.99

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
46	88.05	94.12	88.59	93.93	88.08	94.23	89.98	94.28	83.11	93.99
47	90.27	94.12	89.39	93.99	85.33	94.33	89.90	94.28	87.09	94.07
48	84.37	94.20	32.60	93.96	88.51	94.20	88.11	94.33	87.28	94.07
49	88.13	94.20	86.48	93.99	88.35	94.25	89.76	94.44	88.19	94.15
50	88.03	94.20	89.52	94.07	89.95	94.23	88.46	94.36	88.88	94.25
51	82.26	94.20	88.13	94.15	90.27	94.36	86.75	94.49	87.12	94.20
52	90.06	94.23	86.80	94.17	85.11	94.36	87.17	94.49	87.33	94.20
53	89.42	94.31	87.25	94.15	87.60	94.44	87.60	94.47	89.26	94.20
54	91.40	94.39	89.42	94.20	90.70	94.47	88.96	94.44	88.11	94.25
55	88.03	94.39	89.34	94.23	88.86	94.47	89.36	94.52	87.52	94.33
56	86.34	94.36	89.58	94.28	88.99	94.60	88.46	94.55	86.02	94.36
57	89.28	94.47	89.36	94.33	89.23	94.60	90.51	94.68	89.82	94.41
58	89.68	94.47	86.80	94.47	87.07	94.60	87.87	94.79	89.36	94.31
59	90.35	94.58	89.93	94.47	87.01	94.63	90.11	94.79	87.76	94.31
60	88.91	94.63	88.05	94.60	87.65	94.63	89.18	94.74	88.70	94.39
61	88.13	94.60	83.59	94.63	85.20	94.71	88.56	94.71	84.87	94.47
62	88.94	94.58	88.72	94.60	87.44	94.71	87.81	94.68	85.97	94.49
63	87.97	94.58	88.54	94.60	88.83	94.74	88.59	94.74	88.16	94.52
64	88.38	94.60	86.40	94.58	90.09	94.71	87.71	94.68	88.96	94.49
65	89.50	94.66	88.46	94.55	89.47	94.79	86.29	94.71	86.56	94.66
66	82.36	94.66	87.68	94.71	85.73	94.66	90.09	94.74	86.66	94.63
67	86.69	94.66	89.02	94.63	88.43	94.68	89.07	94.82	88.08	94.63
68	86.21	94.71	88.00	94.68	88.80	94.74	86.50	94.74	89.60	94.60
69	91.82	94.74	88.19	94.63	88.72	94.71	87.23	94.82	85.46	94.58
70	90.06	94.82	89.90	94.60	86.69	94.76	89.50	94.79	87.97	94.63
71	86.32	94.82	89.28	94.58	88.94	94.79	89.39	94.90	88.40	94.60
72	88.16	94.82	90.14	94.68	88.21	94.79	89.26	94.90	87.63	94.58
73	89.55	94.87	88.27	94.82	84.93	94.87	90.70	94.95	88.43	94.60
74	90.83	94.95	81.85	94.82	88.54	94.84	88.91	95.03	87.73	94.60
75	88.80	94.98	88.05	94.82	90.22	94.95	89.20	95.03	87.95	94.58
76	85.94	94.98	88.64	94.79	87.63	94.90	90.51	95.14	86.72	94.55
77	89.39	94.95	89.07	94.71	86.99	94.95	89.50	95.08	87.47	94.58
78	89.36	95.03	88.94	94.71	88.51	94.90	88.75	95.08	88.64	94.60
79	89.68	95.08	89.74	94.76	90.38	94.92	89.74	95.11	85.20	94.49
80	88.21	95.16	59.51	94.79	88.67	94.95	89.44	95.19	81.64	94.58
81	88.86	95.19	87.44	94.76	89.90	95.00	88.13	95.24	86.29	94.49
82	89.23	95.24	89.93	94.66	86.58	94.95	89.63	95.32	85.38	94.55
83	90.86	95.19	88.43	94.76	81.45	94.98	89.74	95.32	89.39	94.68
84	43.93	95.24	87.60	94.84	87.79	95.06	87.36	95.24	85.73	94.68
85	90.22	95.22	89.63	94.87	89.31	95.06	86.50	95.30	89.34	94.71
86	88.56	95.19	89.85	94.84	89.90	95.03	89.18	95.27	89.76	94.63
87	85.36	95.24	87.17	94.92	91.05	95.08	88.32	95.30	85.09	94.66
88	89.52	95.14	88.46	94.98	88.78	95.14	89.71	95.32	89.15	94.68
89	89.74	95.14	90.41	95.00	87.89	95.22	90.70	95.35	85.20	94.68
90	90.22	95.22	88.88	95.00	90.94	95.30	87.73	95.38	85.73	94.63
91	88.30	95.19	86.48	95.00	86.21	95.27	88.00	95.30	86.18	94.63
92	89.36	95.22	90.70	95.06	90.75	95.22	88.48	95.40	88.91	94.76
93	89.07	95.32	90.06	95.06	87.68	95.19	89.98	95.51	88.54	94.68
94	88.46	95.27	90.67	95.03	89.04	95.16	90.97	95.46	87.65	94.74
95	91.18	95.38	91.66	95.08	88.24	95.22	89.42	95.56	87.73	94.76
96	90.33	95.32	90.54	95.14	88.32	95.16	90.57	95.54	86.85	94.71

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
97	91.10	95.40	87.95	95.11	88.16	95.27	90.51	95.54	88.40	94.71
98	88.91	95.32	90.30	95.14	89.42	95.30	88.38	95.54	87.63	94.74
99	88.96	95.40	88.80	95.14	87.17	95.27	88.99	95.62	86.45	94.84
100	88.40	95.46	88.67	95.14	86.05	95.24	90.38	95.64	88.75	94.79
101	89.28	95.46	88.48	95.16	88.27	95.35	91.37	95.59	87.07	94.76
102	89.90	95.43	91.64	95.16	88.48	95.24	88.05	95.56	88.96	94.84
103	86.80	95.46	89.47	95.19	89.60	95.27	90.25	95.64	87.57	94.82
104	90.94	95.54	88.88	95.24	88.56	95.30	88.46	95.64	88.03	94.84
105	88.27	95.56	85.86	95.22	0.64	95.30	89.52	95.62	86.61	94.82
106	88.40	95.51	91.07	95.27	89.76	95.27	89.34	95.67	87.52	94.82
107	90.57	95.48	88.46	95.24	89.26	95.30	89.58	95.64	88.24	94.79
108	87.55	95.46	89.74	95.24	89.95	95.38	90.25	95.67	87.52	94.84
109	88.27	95.46	89.34	95.27	88.70	95.40	90.43	95.62	84.71	94.87
110	90.57	95.48	88.83	95.35	90.75	95.48	90.30	95.67	88.38	94.79
111	90.43	95.51	90.59	95.38	89.04	95.54	89.34	95.72	88.32	94.90
112	89.23	95.51	91.18	95.38	90.41	95.59	89.71	95.67	89.36	95.00
113	88.27	95.54	89.63	95.32	88.11	95.59	90.33	95.80	84.26	94.92
114	87.36	95.56	87.84	95.35	90.57	95.64	89.68	95.70	86.61	94.95
115	83.59	95.59	89.50	95.30	88.56	95.56	89.68	95.72	88.72	95.03
116	91.69	95.59	89.12	95.30	90.78	95.59	88.67	95.70	89.36	95.08
117	89.76	95.64	91.18	95.30	86.80	95.62	89.47	95.70	88.94	95.08
118	90.59	95.67	87.01	95.40	89.66	95.64	90.54	95.72	89.66	95.19
119	89.60	95.62	90.03	95.40	88.27	95.75	90.41	95.72	90.14	95.19
120	4.97	95.62	90.22	95.32	88.38	95.67	89.82	95.67	89.02	95.16
121	90.75	95.67	89.39	95.35	88.43	95.70	89.93	95.67	89.93	95.22
122	88.40	95.67	89.82	95.32	89.71	95.67	88.94	95.70	89.79	95.22
123	90.67	95.67	91.48	95.40	79.05	95.67	88.83	95.78	86.88	95.24
124	91.13	95.64	89.74	95.40	91.07	95.67	89.58	95.80	88.24	95.19
125	90.49	95.80	88.88	95.51	90.25	95.70	81.40	95.78	89.23	95.16
126	89.90	95.75	88.43	95.43	90.51	95.70	89.87	95.80	90.59	95.22
127	89.36	95.75	90.30	95.43	91.23	95.75	89.55	95.78	89.10	95.27
128	90.65	95.80	90.67	95.43	88.27	95.67	90.22	95.75	88.08	95.24
129	89.26	95.75	90.30	95.46	89.02	95.67	89.87	95.80	89.15	95.24
130	89.12	95.78	88.75	95.43	90.73	95.72	90.43	95.75	88.83	95.24
131	89.63	95.80	89.60	95.43	91.42	95.70	91.07	95.72	90.46	95.24
132	90.06	95.78	87.49	95.48	89.42	95.64	90.49	95.75	89.42	95.24
133	92.22	95.75	86.99	95.46	91.29	95.70	90.46	95.78	90.59	95.24
134	90.62	95.86	91.21	95.40	88.88	95.72	89.31	95.78	89.95	95.24
135	88.91	95.80	88.21	95.43	90.75	95.78	91.07	95.78	84.85	95.24
136	88.91	95.78	87.76	95.46	90.06	95.75	90.03	95.78	89.68	95.32
137	88.21	95.78	90.33	95.48	90.41	95.75	91.72	95.78	89.39	95.38
138	89.66	95.80	89.44	95.46	88.08	95.83	89.93	95.75	86.61	95.40
139	90.41	95.75	90.51	95.51	89.79	95.80	92.04	95.75	88.67	95.38
140	90.67	95.78	88.32	95.48	91.37	95.83	90.73	95.78	88.16	95.35
141	87.57	95.75	85.60	95.48	91.42	95.83	89.44	95.78	87.89	95.38
142	90.14	95.72	89.82	95.51	91.58	95.86	90.22	95.86	90.57	95.35
143	90.57	95.78	90.38	95.54	91.05	95.80	91.13	95.88	85.60	95.35
144	90.54	95.70	89.93	95.48	91.05	95.91	90.86	95.88	89.66	95.32
145	88.03	95.80	89.26	95.46	91.69	95.88	87.31	95.96	89.87	95.30
146	91.02	95.80	90.97	95.46	89.52	95.88	88.32	95.86	88.67	95.35
147	90.06	95.83	90.65	95.51	89.50	95.91	90.22	95.91	88.19	95.32

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
148	87.55	95.80	90.62	95.48	88.38	95.96	89.31	95.94	91.45	95.32
149	89.90	95.78	90.25	95.48	88.30	95.96	90.94	95.94	90.11	95.38
150	94.23	95.83	94.33	95.48	93.83	95.94	94.17	96.04	94.49	95.35
151	94.60	95.75	94.71	95.54	94.25	95.94	94.55	96.02	94.41	95.38
152	94.82	95.78	94.71	95.59	94.15	95.99	94.58	95.99	94.82	95.38
153	94.79	95.75	94.95	95.56	94.41	95.91	94.76	96.02	94.87	95.38
154	95.03	95.80	94.95	95.59	94.66	95.91	94.82	96.07	95.03	95.38
155	94.84	95.75	95.03	95.59	94.60	95.91	94.58	96.07	95.03	95.46
156	95.19	95.75	95.32	95.59	94.87	95.94	95.00	96.04	94.95	95.46
157	95.14	95.83	95.22	95.62	94.74	95.91	95.00	96.04	95.08	95.43
158	95.14	95.83	95.06	95.56	94.49	95.96	94.84	96.02	94.74	95.46
159	94.76	95.80	94.84	95.54	94.60	95.94	94.68	95.94	94.66	95.43
160	95.14	95.80	94.98	95.56	94.84	95.91	94.82	95.91	94.79	95.43
161	95.35	95.83	95.14	95.59	94.74	95.91	94.92	95.94	94.92	95.48
162	95.06	95.83	95.24	95.64	94.76	95.91	95.27	95.99	94.90	95.56
163	95.22	95.83	95.27	95.64	95.03	95.91	95.11	96.02	95.00	95.59
164	95.14	95.83	95.24	95.64	94.90	95.91	95.00	96.04	94.92	95.59
165	95.06	95.86	95.19	95.59	94.68	95.96	95.32	95.96	94.92	95.56
166	95.27	95.86	95.32	95.62	95.08	95.99	95.32	95.96	94.84	95.51
167	95.19	95.86	95.14	95.64	94.68	95.96	95.08	95.96	94.87	95.48
168	95.32	95.91	95.48	95.67	94.92	95.96	95.14	95.96	95.00	95.51
169	95.67	95.91	95.56	95.59	95.03	96.02	95.14	95.96	94.76	95.54
170	95.70	95.94	95.56	95.59	95.14	96.07	95.22	95.96	94.92	95.54
171	95.22	95.94	95.54	95.62	94.95	96.02	95.16	96.02	95.00	95.56
172	95.51	95.94	95.11	95.59	95.06	95.99	95.32	96.10	95.24	95.54
173	95.40	95.94	95.30	95.59	95.03	95.96	95.30	96.04	95.16	95.56
174	95.19	95.94	95.46	95.70	95.03	95.94	95.19	96.10	95.03	95.62
175	95.35	95.94	95.27	95.64	94.98	95.96	95.03	96.04	94.98	95.62
176	94.95	95.91	95.27	95.70	95.24	95.96	94.92	96.07	94.98	95.54
177	95.11	95.94	95.38	95.70	95.16	95.96	95.06	96.04	95.06	95.59
178	95.32	95.88	95.27	95.70	95.00	95.94	95.24	96.02	94.92	95.56
179	95.11	95.96	95.43	95.70	95.08	95.94	95.03	95.99	95.08	95.59
180	95.40	95.99	95.48	95.70	95.19	95.94	95.40	95.91	95.08	95.64
181	95.43	95.94	95.46	95.72	95.27	95.91	95.35	95.91	95.00	95.62
182	95.35	95.94	95.27	95.75	95.22	95.91	95.43	95.96	94.90	95.67
183	95.35	95.94	95.59	95.72	95.16	95.88	95.08	95.94	95.27	95.64
184	95.54	95.96	95.48	95.75	94.98	95.88	95.35	95.96	94.90	95.67
185	95.72	95.96	95.51	95.72	95.22	95.88	95.22	95.94	94.95	95.59
186	95.30	95.96	95.48	95.78	95.11	95.86	95.32	95.88	95.22	95.56
187	95.35	96.02	95.16	95.80	95.00	95.86	95.40	95.88	95.00	95.51
188	95.43	96.02	95.19	95.80	95.38	95.86	95.38	95.94	94.90	95.51
189	95.48	96.02	95.40	95.80	95.14	95.83	95.22	95.91	95.11	95.51
190	95.56	96.04	95.35	95.83	95.30	95.91	95.40	95.96	95.00	95.51
191	95.72	96.04	95.22	95.86	95.11	95.88	95.40	95.96	95.08	95.48
192	95.24	96.10	95.32	95.83	95.06	95.83	95.27	95.96	95.11	95.54
193	95.30	96.10	95.24	95.80	95.32	95.83	95.43	95.91	95.06	95.54
194	95.19	96.10	95.48	95.83	95.22	95.80	95.83	95.91	95.19	95.56
195	95.30	96.07	95.51	95.78	95.00	95.80	95.51	95.91	94.98	95.62
196	95.54	96.15	95.38	95.78	95.19	95.80	95.56	95.94	95.16	95.62
197	95.40	96.15	95.62	95.80	95.11	95.80	95.38	95.91	95.03	95.56
198	95.48	96.15	95.14	95.80	95.35	95.83	95.43	95.88	95.40	95.56

	Example 1		Example 2		Example 3		Example 4		Example 5	
Reuse		✓		✓		✓		✓		✓
199	95.27	96.15	95.27	95.78	95.48	95.83	95.48	95.91	95.30	95.56
200	95.51	96.18	95.19	95.80	95.51	95.83	95.14	95.83	95.54	95.54
201	95.56	96.18	95.35	95.78	95.22	95.80	95.38	95.80	95.30	95.56
202	95.40	96.15	95.51	95.72	95.51	95.86	95.35	95.80	95.24	95.56
203	95.62	96.18	95.43	95.72	95.62	95.86	95.30	95.80	95.30	95.67
204	95.35	96.15	95.16	95.70	95.46	95.86	95.48	95.75	95.38	95.67
205	95.64	96.15	95.54	95.70	95.56	95.83	95.30	95.80	95.51	95.67
206	95.51	96.13	95.43	95.70	95.51	95.83	95.40	95.75	95.30	95.64
207	95.56	96.13	95.51	95.72	95.43	95.78	95.46	95.72	95.32	95.67
208	95.56	96.13	95.46	95.67	95.30	95.78	95.40	95.72	95.46	95.70
209	95.56	96.13	95.46	95.67	95.43	95.75	95.48	95.83	95.30	95.70
210	95.80	96.10	95.59	95.70	95.32	95.80	95.51	95.83	95.00	95.70
211	95.40	96.10	95.59	95.70	95.35	95.80	95.24	95.83	95.35	95.70
212	95.62	96.13	95.43	95.72	95.35	95.80	95.38	95.80	95.43	95.72
213	95.56	96.13	95.54	95.75	95.30	95.78	95.56	95.80	95.27	95.72
214	95.59	96.13	95.46	95.72	95.38	95.80	95.46	95.78	95.32	95.70
215	95.62	96.13	95.43	95.72	95.48	95.78	95.40	95.83	95.38	95.70
216	95.46	96.07	95.30	95.75	95.08	95.78	95.27	95.83	95.54	95.70
217	95.70	96.02	95.48	95.78	95.32	95.78	95.43	95.83	95.40	95.70
218	95.70	96.02	95.30	95.78	95.08	95.75	95.24	95.86	95.59	95.70
219	95.32	95.99	95.22	95.78	95.06	95.75	95.46	95.86	95.43	95.70
220	95.59	96.02	95.62	95.78	95.32	95.78	95.54	95.83	95.35	95.67
221	95.70	96.02	95.38	95.78	95.32	95.75	95.51	95.80	95.40	95.70
222	95.72	96.02	95.38	95.75	95.38	95.80	95.48	95.80	95.54	95.70
223	95.62	96.02	95.48	95.75	95.40	95.78	95.56	95.78	95.48	95.64
224	95.62	96.02	95.64	95.72	95.48	95.78	95.56	95.78	95.32	95.67
225	95.83	96.04	95.64	95.72	95.32	95.78	95.40	95.78	95.40	95.64
226	95.75	96.07	95.48	95.72	95.24	95.75	95.54	95.78	95.56	95.64
227	95.83	96.07	95.56	95.72	95.48	95.75	95.56	95.80	95.35	95.62
228	95.72	96.07	95.70	95.72	95.35	95.75	95.51	95.80	95.51	95.62
229	95.62	96.02	95.67	95.70	95.32	95.72	95.35	95.80	95.62	95.62
230	95.56	96.02	95.54	95.70	95.48	95.72	95.35	95.80	95.46	95.59
231	95.48	96.02	95.64	95.70	95.30	95.72	95.35	95.80	95.43	95.59
232	95.80	96.02	95.75	95.72	95.51	95.72	95.56	95.80	95.51	95.59
233	95.67	96.02	95.56	95.70	95.40	95.72	95.54	95.83	95.32	95.59
234	95.88	95.99	95.54	95.67	95.54	95.72	95.51	95.83	95.22	95.59
235	95.75	96.02	95.16	95.64	95.27	95.72	95.51	95.80	95.40	95.59
236	95.67	95.99	95.46	95.64	95.35	95.72	95.43	95.78	95.22	95.59
237	95.72	95.99	95.43	95.64	95.43	95.72	95.40	95.78	95.22	95.59
238	95.62	95.99	95.51	95.62	95.22	95.72	95.22	95.75	95.40	95.59
239	96.02	96.02	95.59	95.62	95.51	95.70	95.59	95.78	95.35	95.59
240	95.24	96.02	95.40	95.62	95.00	95.64	95.56	95.78	95.30	95.59
241	95.80	96.04	95.59	95.62	95.27	95.62	95.43	95.78	95.19	95.59
242	95.59	96.02	95.54	95.59	95.22	95.62	95.54	95.78	95.43	95.59
243	95.56	95.99	95.48	95.62	95.14	95.62	95.38	95.78	95.30	95.59
244	95.64	95.99	95.48	95.59	95.35	95.62	95.43	95.78	95.27	95.59
245	95.80	95.99	95.48	95.56	95.32	95.64	95.62	95.78	95.22	95.56
246	95.75	95.99	95.40	95.59	95.40	95.64	95.43	95.78	95.30	95.56
247	95.75	95.99	95.78	95.59	95.43	95.64	95.46	95.78	95.38	95.59
248	95.64	95.96	95.43	95.59	95.35	95.64	95.35	95.78	95.22	95.59
249	95.94	95.96	95.67	95.59	95.16	95.62	95.59	95.78	95.40	95.59

	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
0	13.47	13.47	9.30	9.30	9.51	9.51	10.02	10.02	7.91	7.91
1	26.27	26.59	29.58	28.27	34.10	31.67	30.30	29.88	34.66	32.63
2	52.86	52.19	49.63	48.90	48.64	47.94	50.40	48.53	51.34	50.35
3	59.70	62.19	58.20	58.44	60.40	59.57	60.26	60.77	65.74	62.75
4	68.17	71.03	65.37	67.88	66.92	68.79	72.39	71.73	71.99	72.85
5	75.36	77.36	74.61	75.92	76.32	76.75	73.22	77.20	78.54	78.11
6	75.47	80.60	62.21	77.50	77.58	80.84	79.08	81.56	76.35	81.69
7	77.58	82.98	73.68	81.51	77.95	82.98	81.19	84.02	76.75	84.02
8	74.51	84.42	78.25	84.55	75.33	85.06	78.17	85.94	73.70	85.30
9	82.84	86.80	82.76	86.58	79.72	86.66	81.05	86.77	79.13	86.21
10	82.52	87.68	82.82	87.65	82.28	88.00	80.60	88.03	81.32	87.68
11	79.24	88.38	84.47	88.67	84.37	89.15	84.47	88.59	85.97	88.75
12	81.93	89.60	85.03	89.50	79.66	89.55	82.18	89.68	82.02	89.39
13	79.48	90.25	77.63	89.74	77.28	89.74	83.08	89.76	74.08	90.03
14	86.96	90.97	83.73	90.06	86.61	90.35	81.88	90.25	83.89	90.62
15	83.11	91.23	82.26	90.43	85.68	90.67	87.09	90.65	85.49	90.89
16	86.72	91.34	83.11	91.02	85.14	91.45	84.02	91.15	84.63	91.45
17	80.47	91.45	83.54	91.07	83.22	91.50	81.67	91.56	85.33	91.85
18	81.61	91.74	84.37	91.58	80.30	91.82	84.93	91.74	83.73	91.82
19	88.16	91.88	85.01	91.69	83.70	92.09	83.56	91.66	83.73	91.93
20	82.20	92.22	81.80	92.01	86.16	92.30	85.28	92.01	83.16	92.12
21	87.07	92.57	81.24	91.85	86.58	92.49	84.37	92.04	82.39	92.12
22	85.46	92.60	85.36	92.14	81.83	92.54	86.69	92.12	84.95	92.28
23	85.68	92.92	83.86	92.38	86.48	92.62	85.49	92.52	83.51	92.52
24	84.21	93.08	88.88	92.73	84.66	92.81	87.79	92.49	85.49	92.73
25	85.97	93.27	80.87	92.97	88.51	92.73	84.47	92.65	85.33	92.84
26	84.66	93.32	85.41	92.84	87.44	93.05	87.47	92.73	87.28	93.08
27	87.15	93.27	85.25	92.92	88.51	93.08	88.35	92.86	83.59	93.16
28	83.75	93.48	88.19	93.05	88.27	92.97	88.96	93.08	89.55	93.08
29	82.55	93.48	86.96	93.27	87.12	93.29	86.75	93.19	87.76	93.32
30	83.14	93.64	88.40	93.21	88.21	93.45	89.90	93.19	88.62	93.37
31	82.18	93.80	86.48	93.43	87.01	93.51	88.99	93.32	89.50	93.67
32	88.35	93.77	85.89	93.48	87.09	93.61	86.24	93.37	89.39	93.85
33	84.13	93.83	88.32	93.61	88.32	93.91	86.64	93.45	87.84	93.91
34	89.20	93.96	84.10	93.43	86.96	93.88	88.91	93.56	85.54	94.01
35	85.54	93.99	87.57	93.53	87.33	93.83	84.95	93.64	85.78	94.04
36	87.36	94.07	87.68	93.67	89.18	93.99	88.51	93.59	87.63	94.12
37	88.72	94.01	85.41	93.77	86.96	93.88	85.20	93.64	85.70	94.04
38	87.28	93.99	88.43	93.91	87.79	93.96	86.05	93.64	89.36	94.01
39	84.29	94.12	87.09	94.01	88.83	94.01	87.20	93.61	88.80	93.99
40	85.65	94.12	88.51	94.01	88.03	94.01	88.51	93.61	89.60	94.07
41	85.41	94.17	87.81	94.15	90.35	94.12	88.67	93.72	89.15	94.25
42	87.12	94.12	87.81	94.12	86.50	94.15	87.20	93.72	87.39	94.20
43	86.34	94.17	88.91	94.17	89.44	94.12	87.20	93.75	88.40	94.58
44	85.97	94.23	88.72	94.41	88.05	94.07	87.31	93.77	87.81	94.63
45	90.30	94.23	47.49	94.41	89.52	94.15	87.95	93.83	89.31	94.68
46	88.24	94.39	85.60	94.44	85.30	94.09	84.66	93.88	87.92	94.63
47	86.32	94.49	89.52	94.49	87.95	94.04	89.71	93.83	85.57	94.58
48	87.65	94.60	87.92	94.55	87.81	94.09	84.02	93.91	89.82	94.63
49	87.23	94.60	84.37	94.58	88.56	94.20	88.11	93.91	87.71	94.71

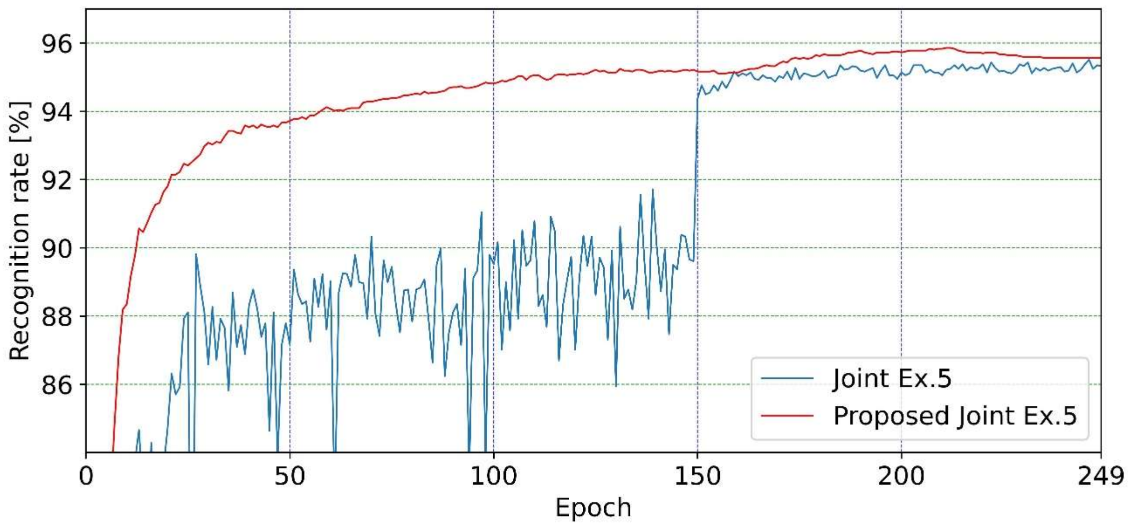
	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
50	87.71	94.58	86.00	94.60	85.65	94.33	87.20	93.96	84.42	94.66
51	85.57	94.60	86.80	94.71	86.50	94.47	88.00	94.01	86.50	94.66
52	89.04	94.68	89.50	94.63	87.84	94.47	89.26	94.07	87.39	94.71
53	86.10	94.79	88.43	94.63	88.19	94.52	85.81	94.04	87.73	94.84
54	85.65	94.76	88.48	94.71	84.55	94.55	87.55	94.12	88.75	94.84
55	89.68	94.82	86.75	94.55	87.25	94.55	86.93	94.23	87.23	94.87
56	87.15	94.76	86.93	94.55	87.89	94.58	89.23	94.31	86.26	94.90
57	87.87	94.79	87.41	94.68	82.50	94.63	85.76	94.25	83.40	94.92
58	86.02	94.71	86.21	94.74	85.03	94.58	88.96	94.28	88.64	95.08
59	87.60	94.71	87.01	94.87	87.20	94.52	87.73	94.28	89.07	95.08
60	86.61	94.76	86.56	94.87	83.65	94.58	88.00	94.31	87.52	94.98
61	89.76	94.76	87.33	95.00	89.85	94.60	85.60	94.36	88.05	95.03
62	86.56	94.82	89.60	94.95	88.56	94.66	88.13	94.41	85.94	95.08
63	84.66	94.82	89.71	94.95	88.75	94.74	88.40	94.36	89.02	95.06
64	87.12	94.74	90.14	94.95	88.75	94.76	89.66	94.47	89.12	95.14
65	74.56	94.76	86.91	95.00	88.56	94.84	89.23	94.52	86.96	95.06
66	88.46	94.82	87.04	94.92	88.08	94.79	87.23	94.55	90.22	95.08
67	85.36	94.92	86.85	94.98	87.15	94.76	88.86	94.49	14.83	95.11
68	88.11	95.00	85.86	95.03	87.25	94.76	87.97	94.49	88.19	95.08
69	88.16	94.92	88.32	94.92	88.00	94.84	90.67	94.55	89.36	95.06
70	88.64	95.00	90.81	95.06	87.33	94.84	88.94	94.49	87.31	95.00
71	88.27	94.95	90.14	95.11	89.31	94.87	88.21	94.55	88.21	95.14
72	87.36	94.92	89.23	95.14	88.96	94.79	90.51	94.60	85.89	95.16
73	89.42	94.90	88.94	95.08	85.70	94.79	86.37	94.66	87.97	95.22
74	87.68	94.95	88.46	95.11	88.30	94.82	89.15	94.66	87.52	95.16
75	87.79	94.90	76.88	95.11	84.82	94.82	89.02	94.71	84.90	95.19
76	89.28	94.82	85.73	95.08	89.18	94.87	87.73	94.58	83.83	95.24
77	87.41	94.87	92.28	95.08	89.79	94.84	87.76	94.58	88.54	95.24
78	87.92	94.87	87.20	95.19	88.96	94.92	88.62	94.60	87.39	95.22
79	88.75	94.90	87.79	95.14	89.52	94.92	87.20	94.52	90.01	95.35
80	89.20	94.87	87.33	95.11	89.12	95.00	89.39	94.60	88.35	95.30
81	88.46	94.90	89.10	95.16	90.83	95.00	88.75	94.68	88.05	95.35
82	89.42	94.87	88.11	95.03	89.04	94.98	90.11	94.68	89.20	95.32
83	89.50	94.98	86.16	95.11	88.21	95.00	89.93	94.66	89.52	95.27
84	86.69	94.98	86.72	95.06	86.02	94.98	89.18	94.66	85.86	95.30
85	64.64	94.98	87.81	95.08	88.56	94.92	88.67	94.66	86.77	95.32
86	84.45	94.95	90.01	94.98	89.44	94.98	89.93	94.63	88.78	95.30
87	89.07	94.92	88.24	95.00	89.42	94.98	87.65	94.68	85.25	95.38
88	87.71	94.95	89.58	95.03	87.63	95.03	89.55	94.76	88.91	95.38
89	87.63	94.95	88.38	95.03	89.52	94.98	90.01	94.74	87.81	95.35
90	89.74	95.00	89.10	95.08	87.52	95.03	88.21	94.79	89.79	95.48
91	88.72	95.03	88.32	95.03	89.60	95.00	89.10	94.79	87.81	95.35
92	89.95	95.06	89.12	95.06	87.60	95.06	89.98	94.87	89.31	95.43
93	84.18	95.16	88.94	95.11	89.15	95.06	88.72	94.87	90.25	95.43
94	85.70	95.16	86.08	95.11	87.60	95.06	90.38	94.92	87.07	95.48
95	87.76	95.16	86.77	95.06	87.97	95.11	88.88	94.87	88.13	95.46
96	89.66	95.08	89.23	95.14	89.50	95.06	88.05	94.87	88.94	95.46
97	86.77	95.11	89.07	95.16	87.31	95.11	88.67	94.90	89.50	95.51
98	87.68	95.06	88.56	95.22	89.50	95.06	85.68	94.92	89.55	95.56
99	90.25	95.03	88.00	95.19	90.67	95.06	88.38	94.95	88.99	95.62
100	89.63	95.11	88.75	95.24	90.25	95.08	86.80	94.87	88.21	95.62

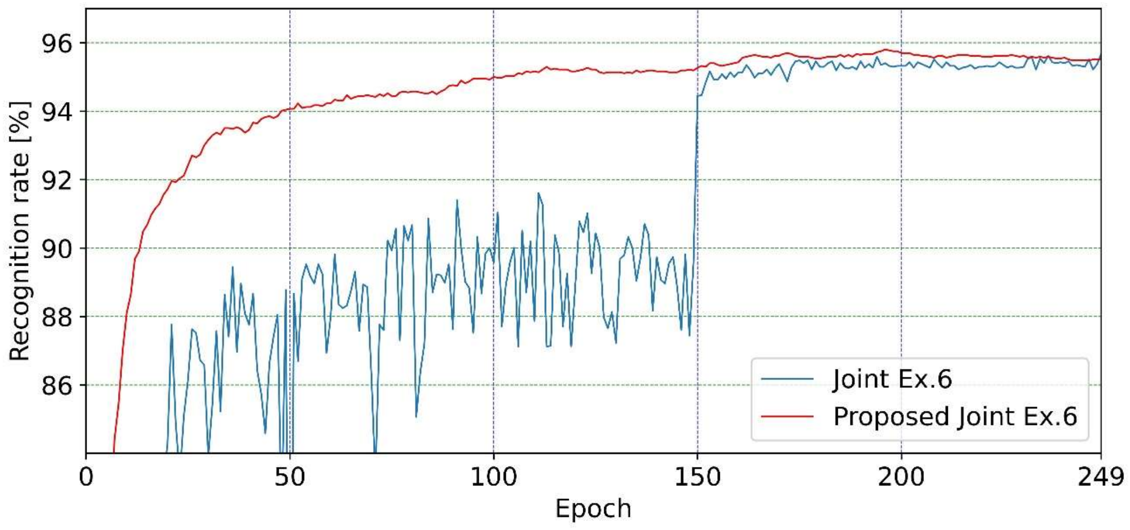
	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
101	89.12	95.19	90.57	95.27	90.73	95.16	89.39	94.95	88.54	95.62
102	88.75	95.19	88.56	95.24	86.13	95.14	88.78	95.00	88.43	95.62
103	88.46	95.19	87.92	95.24	89.58	95.14	87.76	94.98	86.58	95.62
104	89.52	95.22	88.05	95.27	89.76	95.08	89.79	95.06	88.48	95.62
105	90.35	95.27	89.02	95.24	89.42	95.11	90.25	95.06	89.63	95.70
106	87.71	95.24	89.36	95.14	90.70	95.06	89.04	95.03	89.23	95.67
107	88.62	95.30	90.09	95.22	89.28	95.11	89.52	95.03	89.39	95.64
108	86.21	95.24	89.20	95.16	88.24	95.11	89.31	95.06	87.07	95.67
109	88.51	95.27	88.72	95.19	88.40	95.08	89.93	95.03	87.73	95.72
110	88.43	95.24	89.18	95.16	89.52	95.19	86.21	95.03	90.83	95.75
111	88.80	95.32	89.82	95.16	87.52	95.27	89.68	94.98	90.17	95.80
112	89.31	95.30	86.83	95.16	87.89	95.27	89.76	94.98	90.62	95.75
113	87.73	95.30	91.50	95.27	89.95	95.32	87.76	94.98	90.65	95.72
114	84.50	95.30	89.60	95.30	89.44	95.32	89.39	95.08	87.39	95.75
115	90.99	95.27	90.22	95.27	89.39	95.27	91.31	95.14	87.25	95.72
116	88.64	95.27	88.72	95.24	90.70	95.32	89.85	95.19	88.67	95.75
117	89.52	95.32	90.59	95.35	89.18	95.32	89.26	95.19	88.70	95.78
118	89.12	95.38	89.76	95.35	90.14	95.38	90.49	95.19	89.20	95.75
119	89.85	95.38	89.63	95.35	89.39	95.38	88.91	95.11	88.21	95.86
120	88.91	95.32	90.46	95.35	89.85	95.40	88.70	95.14	90.35	95.80
121	90.11	95.35	89.95	95.38	89.95	95.38	40.54	95.11	88.99	95.88
122	87.97	95.32	88.08	95.35	90.51	95.35	89.36	95.16	90.03	95.75
123	89.85	95.30	91.56	95.30	90.30	95.38	90.59	95.14	90.67	95.83
124	87.25	95.24	88.27	95.35	89.58	95.35	90.62	95.14	90.62	95.75
125	88.83	95.24	87.81	95.35	88.13	95.40	90.14	95.16	88.80	95.78
126	88.40	95.30	90.43	95.35	89.52	95.43	88.24	95.19	81.67	95.75
127	85.11	95.27	88.46	95.43	91.10	95.35	88.86	95.22	88.13	95.72
128	90.67	95.27	90.78	95.38	88.80	95.38	87.49	95.30	89.58	95.70
129	85.41	95.27	89.85	95.43	87.47	95.30	89.26	95.24	90.19	95.72
130	90.11	95.30	86.75	95.38	89.79	95.30	88.51	95.24	90.41	95.75
131	89.58	95.30	90.99	95.51	89.71	95.32	89.82	95.22	91.80	95.80
132	89.10	95.30	88.80	95.54	88.72	95.32	90.03	95.16	90.89	95.83
133	88.30	95.27	90.41	95.56	89.12	95.35	89.74	95.11	89.23	95.78
134	88.03	95.30	91.21	95.56	90.33	95.32	90.27	95.16	89.98	95.78
135	89.68	95.27	90.91	95.56	89.95	95.30	88.96	95.22	91.05	95.80
136	91.88	95.30	87.89	95.59	88.16	95.27	89.76	95.22	88.30	95.91
137	88.21	95.40	90.35	95.56	90.01	95.32	89.79	95.19	88.83	95.88
138	87.81	95.38	86.48	95.62	89.07	95.30	89.07	95.19	88.94	95.88
139	90.27	95.40	90.67	95.59	87.92	95.35	90.35	95.16	90.65	95.86
140	90.57	95.43	90.17	95.62	89.93	95.32	83.56	95.16	90.81	95.86
141	88.75	95.46	89.44	95.67	88.94	95.32	87.76	95.14	88.75	95.88
142	88.54	95.43	88.03	95.64	89.82	95.35	91.05	95.11	89.74	95.86
143	91.48	95.46	89.34	95.67	90.57	95.32	88.99	95.08	90.43	95.83
144	90.19	95.38	93.03	95.56	86.96	95.32	89.18	95.14	87.65	95.80
145	90.11	95.43	87.76	95.59	90.14	95.38	90.41	95.14	87.49	95.86
146	87.89	95.43	90.51	95.56	91.58	95.32	88.99	95.08	90.49	95.86
147	89.87	95.43	91.58	95.56	91.74	95.40	89.85	95.11	89.95	95.88
148	86.05	95.43	89.02	95.56	90.49	95.51	88.64	95.16	90.38	95.88
149	90.38	95.43	90.06	95.56	89.79	95.54	89.58	95.08	89.60	95.88
150	94.33	95.43	94.68	95.70	94.25	95.62	93.99	95.14	94.12	95.88
151	94.44	95.43	95.00	95.70	94.58	95.62	94.60	95.19	94.47	95.86

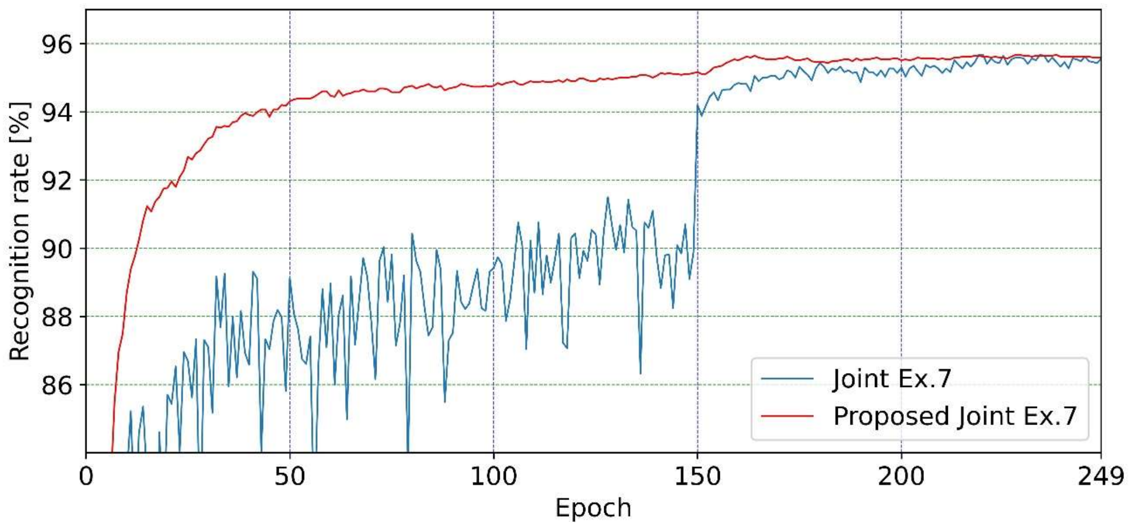
	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
152	94.79	95.43	95.22	95.70	94.84	95.64	94.44	95.14	94.84	95.83
153	95.00	95.43	95.48	95.70	94.74	95.62	94.79	95.16	94.55	95.83
154	95.22	95.54	95.24	95.75	94.87	95.59	94.60	95.19	94.60	95.86
155	94.98	95.51	95.62	95.75	94.84	95.64	94.79	95.27	94.68	95.88
156	94.92	95.51	95.59	95.78	94.95	95.70	94.76	95.35	94.52	95.80
157	94.90	95.54	95.67	95.83	95.19	95.64	95.00	95.38	94.98	95.88
158	95.08	95.59	95.54	95.83	95.14	95.56	94.92	95.46	94.79	95.83
159	94.98	95.54	95.62	95.83	95.06	95.59	94.68	95.46	94.55	95.80
160	94.98	95.54	95.40	95.88	95.22	95.59	95.00	95.51	94.82	95.80
161	95.11	95.48	95.51	95.91	95.43	95.67	94.82	95.51	94.68	95.80
162	95.03	95.48	95.56	95.86	95.24	95.67	94.95	95.48	94.95	95.80
163	94.95	95.51	95.51	95.94	95.40	95.64	94.95	95.62	94.76	95.83
164	95.11	95.48	95.54	95.91	95.32	95.72	94.92	95.67	94.71	95.83
165	95.08	95.54	95.59	95.96	95.38	95.75	94.98	95.64	95.14	95.83
166	95.08	95.56	95.75	95.99	95.19	95.80	94.92	95.70	94.95	95.83
167	95.14	95.59	95.96	95.96	95.30	95.78	94.95	95.72	95.08	95.88
168	94.98	95.56	95.91	96.02	95.19	95.83	95.00	95.70	95.19	95.83
169	95.51	95.54	95.72	96.04	95.32	95.86	94.95	95.70	94.84	95.83
170	95.06	95.54	95.67	96.04	95.22	95.80	95.16	95.70	95.00	95.83
171	95.14	95.56	95.43	96.07	95.32	95.78	95.32	95.75	95.32	95.83
172	95.22	95.62	95.80	96.07	95.54	95.80	95.19	95.70	95.03	95.83
173	95.19	95.59	95.80	96.07	95.32	95.86	94.98	95.75	94.87	95.70
174	95.35	95.56	95.86	96.04	95.16	95.86	94.84	95.70	95.03	95.72
175	95.32	95.59	96.02	96.10	95.30	95.86	94.90	95.72	95.32	95.75
176	95.35	95.59	95.72	96.13	95.14	95.86	95.16	95.70	95.00	95.75
177	95.32	95.64	95.99	96.15	94.98	95.86	95.06	95.70	94.98	95.70
178	95.24	95.70	95.75	96.18	95.11	95.91	94.98	95.72	94.98	95.70
179	95.03	95.72	95.80	96.18	95.32	95.94	94.66	95.72	94.84	95.62
180	95.08	95.70	95.78	96.23	95.16	95.94	94.76	95.67	94.84	95.62
181	95.19	95.70	95.94	96.31	95.35	95.96	94.87	95.67	95.14	95.59
182	94.98	95.72	95.96	96.23	95.46	95.96	95.16	95.67	95.11	95.56
183	95.27	95.78	95.67	96.18	95.43	95.99	94.74	95.67	95.08	95.56
184	95.35	95.78	95.67	96.18	95.30	95.94	94.71	95.67	95.16	95.56
185	95.22	95.80	96.04	96.18	95.32	95.94	94.84	95.64	95.00	95.56
186	95.16	95.78	95.62	96.18	95.56	95.94	95.08	95.64	95.19	95.59
187	95.35	95.83	95.80	96.15	95.48	96.02	94.90	95.70	94.92	95.59
188	95.40	95.83	95.72	96.13	95.43	95.99	95.08	95.59	95.22	95.59
189	95.35	95.83	95.83	96.15	95.48	95.91	95.03	95.62	95.22	95.59
190	95.27	95.83	95.62	96.13	95.48	95.91	94.74	95.59	95.08	95.59
191	95.51	95.83	95.59	96.15	95.40	95.91	94.79	95.64	95.16	95.62
192	95.40	95.86	95.54	96.13	95.27	95.91	94.84	95.67	94.98	95.70
193	95.30	95.86	95.99	96.15	95.38	95.91	94.74	95.64	95.19	95.70
194	95.46	95.88	95.67	96.13	95.48	95.91	94.92	95.62	95.16	95.70
195	95.32	95.86	95.72	96.13	95.19	95.86	94.87	95.62	95.16	95.70
196	95.27	95.88	95.86	96.13	95.22	95.86	94.87	95.62	95.30	95.70
197	95.11	95.88	95.78	96.13	95.27	95.91	94.71	95.62	95.14	95.72
198	95.24	95.86	95.62	96.13	95.24	95.86	94.79	95.64	95.16	95.75
199	95.19	95.86	95.46	96.13	95.22	95.91	94.95	95.64	94.87	95.70
200	95.14	95.83	95.59	96.15	95.27	95.88	95.00	95.62	95.32	95.75
201	95.06	95.83	95.54	96.18	95.11	95.86	94.98	95.56	94.95	95.72
202	95.16	95.83	95.56	96.18	95.32	95.86	94.76	95.56	94.95	95.72

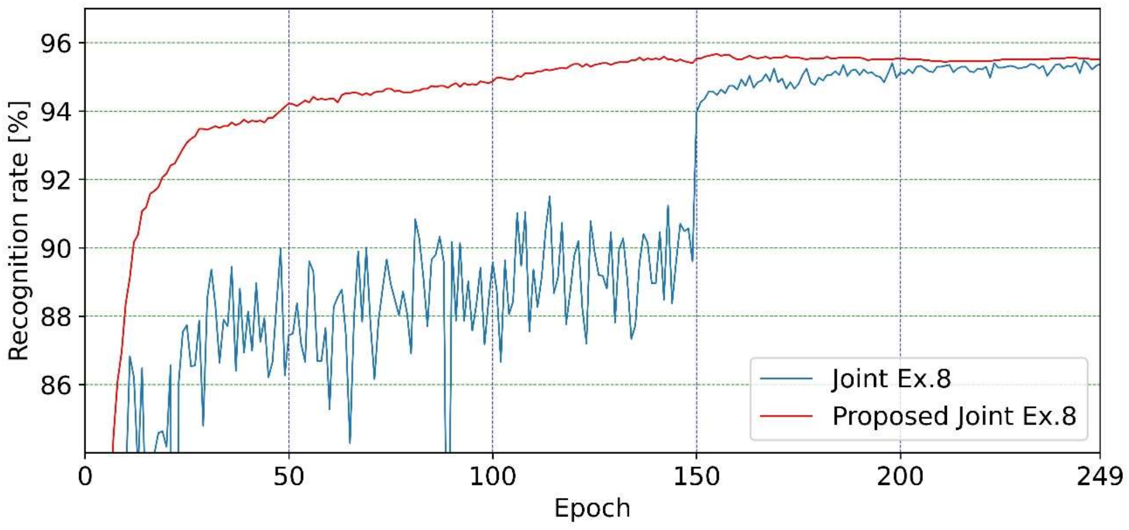
	Example 6		Example 7		Example 8		Example 9		Example 10	
Reuse		✓		✓		✓		✓		✓
203	95.03	95.83	95.56	96.18	95.27	95.86	94.92	95.51	94.90	95.72
204	95.38	95.83	95.72	96.15	95.51	95.86	94.82	95.51	95.22	95.78
205	95.24	95.83	95.80	96.18	95.38	95.88	94.76	95.48	95.38	95.80
206	95.22	95.80	95.72	96.10	95.40	95.86	94.63	95.51	95.35	95.80
207	95.27	95.78	95.72	96.10	95.35	95.88	94.60	95.48	95.16	95.80
208	95.38	95.78	95.78	96.10	95.35	95.94	94.74	95.51	95.19	95.80
209	95.32	95.78	95.62	96.10	95.35	95.91	94.92	95.51	94.87	95.78
210	95.35	95.78	95.54	96.10	95.24	95.96	94.68	95.51	95.06	95.80
211	95.03	95.75	95.64	96.10	95.27	95.96	95.43	95.51	95.16	95.83
212	95.11	95.75	95.56	96.07	95.32	95.91	94.95	95.51	95.14	95.83
213	95.43	95.75	95.83	96.07	95.46	95.88	95.08	95.51	94.98	95.83
214	95.35	95.75	95.83	96.02	95.38	95.86	94.95	95.51	95.00	95.80
215	95.32	95.70	95.70	96.04	95.16	95.86	95.19	95.51	95.14	95.83
216	95.35	95.70	95.78	96.04	95.38	95.83	95.16	95.51	95.06	95.80
217	95.40	95.72	95.88	96.02	95.35	95.80	94.87	95.46	95.30	95.78
218	95.40	95.67	95.67	95.99	95.24	95.78	95.08	95.46	95.32	95.78
219	95.30	95.67	95.99	96.02	95.54	95.78	94.98	95.46	95.32	95.78
220	95.24	95.62	95.72	95.99	95.56	95.78	95.06	95.46	94.90	95.80
221	95.24	95.62	95.86	95.99	95.51	95.75	94.79	95.43	95.14	95.80
222	95.35	95.64	95.83	96.02	95.30	95.70	94.90	95.46	95.24	95.80
223	95.32	95.62	95.80	96.02	95.43	95.67	94.90	95.46	95.27	95.80
224	95.22	95.64	96.07	96.04	95.59	95.67	94.76	95.43	95.19	95.86
225	95.56	95.62	95.83	96.04	95.27	95.64	94.84	95.43	95.08	95.86
226	95.24	95.62	95.72	96.04	95.27	95.64	95.11	95.38	95.32	95.86
227	95.32	95.62	95.56	96.04	95.27	95.67	94.87	95.40	95.00	95.83
228	95.35	95.62	95.83	96.04	95.43	95.67	95.06	95.35	95.06	95.83
229	95.27	95.62	95.80	96.02	95.32	95.67	94.92	95.35	95.27	95.86
230	95.22	95.62	95.75	96.02	95.40	95.72	94.92	95.35	95.06	95.88
231	95.56	95.59	95.78	96.04	95.22	95.72	94.82	95.38	95.32	95.88
232	95.51	95.59	96.02	96.07	95.54	95.70	94.98	95.40	95.35	95.88
233	95.30	95.62	95.96	96.07	95.62	95.70	94.68	95.38	95.35	95.91
234	95.35	95.59	96.07	96.07	95.40	95.70	95.16	95.40	95.30	95.91
235	95.35	95.62	95.88	96.10	95.46	95.70	95.11	95.40	95.22	95.91
236	95.48	95.64	96.02	96.10	95.32	95.70	94.95	95.40	95.03	95.91
237	95.62	95.64	95.88	96.10	95.24	95.70	94.71	95.40	95.08	95.91
238	95.38	95.67	95.80	96.10	95.38	95.75	94.84	95.40	95.22	95.91
239	95.16	95.67	95.91	96.10	95.40	95.75	95.14	95.40	95.11	95.88
240	95.46	95.67	95.88	96.10	95.38	95.75	94.87	95.40	95.27	95.88
241	95.48	95.70	95.94	96.10	95.70	95.78	94.95	95.35	95.16	95.86
242	95.43	95.70	95.88	96.10	95.35	95.75	94.71	95.35	95.24	95.86
243	95.38	95.70	95.94	96.10	95.56	95.75	95.06	95.35	95.24	95.86
244	95.38	95.70	95.80	96.10	95.38	95.75	94.92	95.35	94.74	95.86
245	95.51	95.70	96.07	96.10	95.46	95.75	94.92	95.35	95.35	95.86
246	95.38	95.70	95.94	96.10	95.27	95.70	95.08	95.35	95.08	95.86
247	95.32	95.70	95.99	96.10	95.59	95.70	94.76	95.30	95.08	95.86
248	95.40	95.70	95.99	96.10	95.35	95.67	94.98	95.30	95.08	95.86
249	95.48	95.70	95.83	96.10	95.38	95.72	95.08	95.30	95.30	95.78

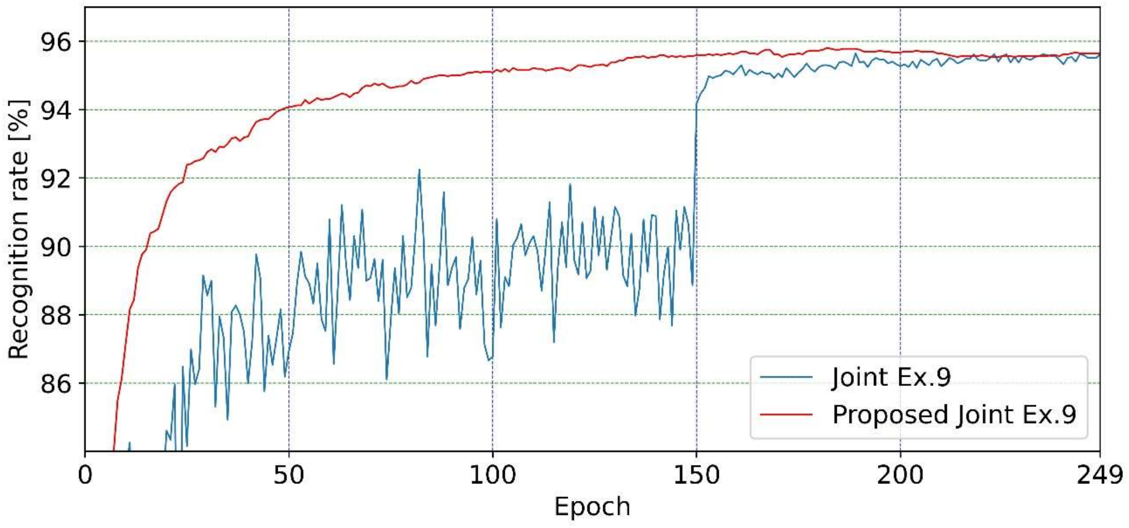
3. The remaining six comparison graphs of recognition rate for Joint and Proposed Joint across each training sessions with random initial conditions.

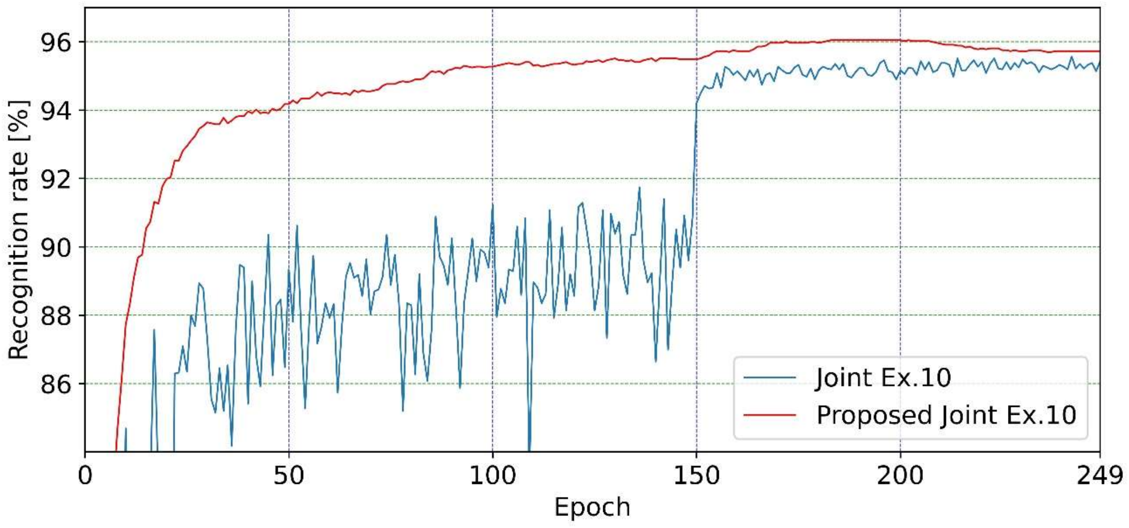




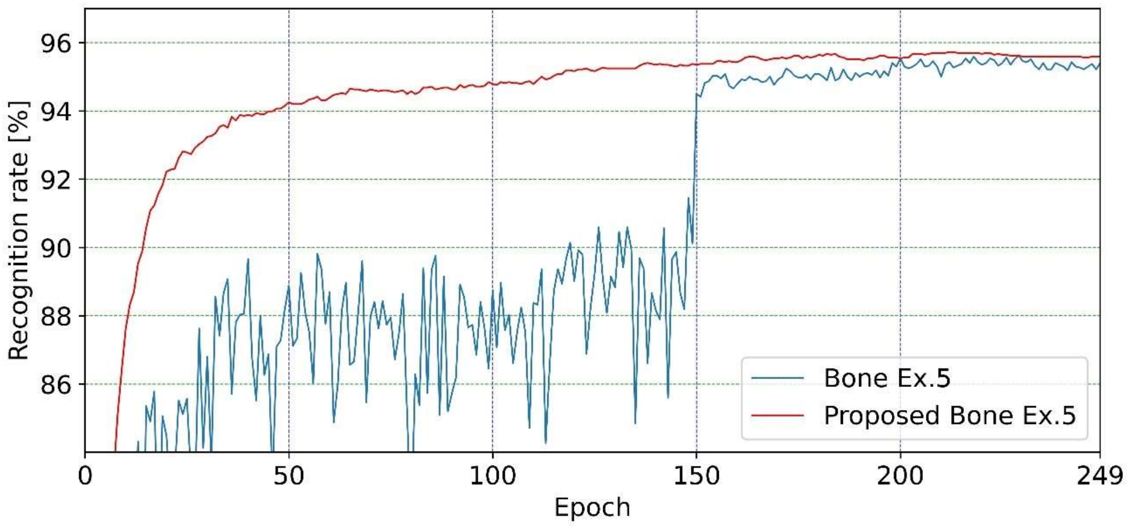


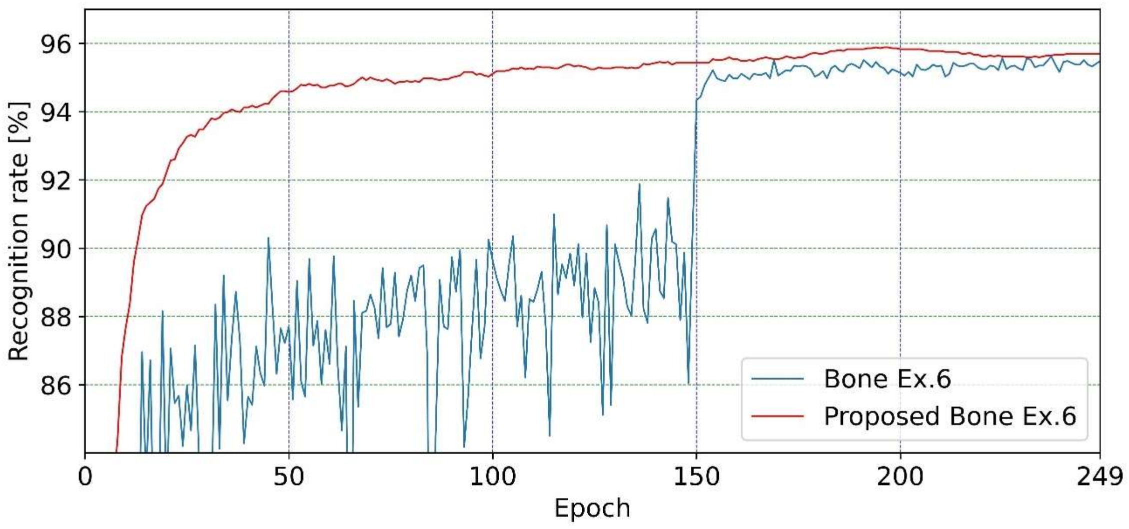

Recognition rate [%]
96
94
92
90
88
86
0
50
100
150
200
249
Epoch
Joint Ex.8
Proposed Joint Ex.8

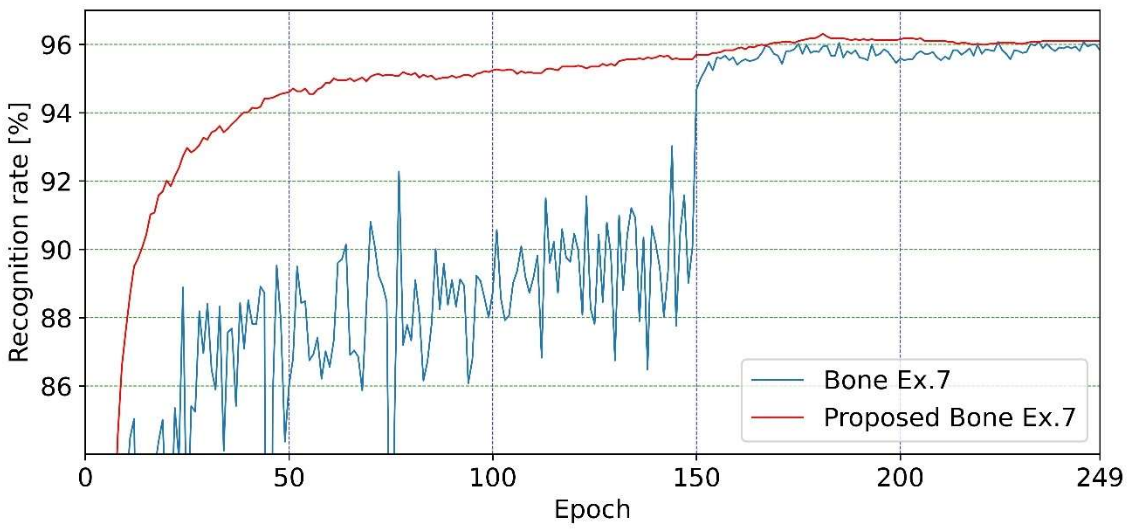

Recognition rate [%]
96
94
92
90
88
86
0
50
100
150
200
249
Epoch
Joint Ex.9
Proposed Joint Ex.9

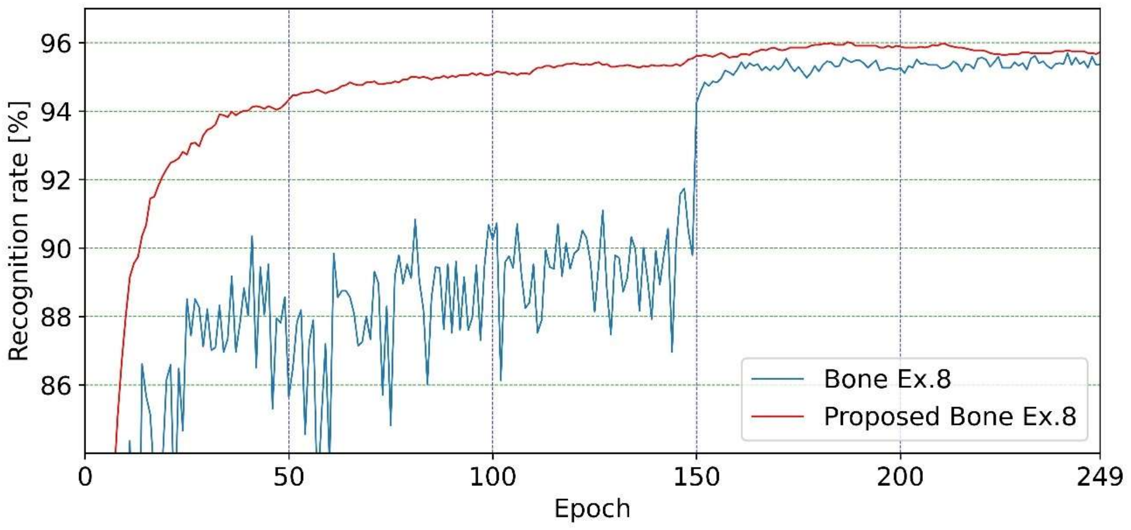

Recognition rate [%]
96
94
92
90
88
86
0
50
100
150
200
249
Epoch
Joint Ex.10
Proposed Joint Ex.10

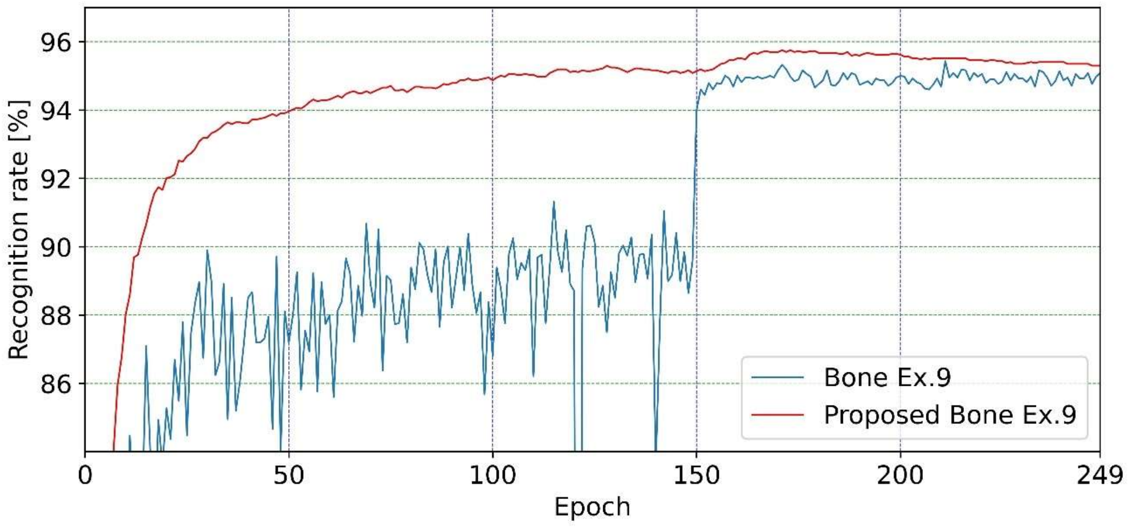
4. The remaining six comparison graphs of recognition rate for Bone and Proposed Bone across each training sessions with random initial conditions.

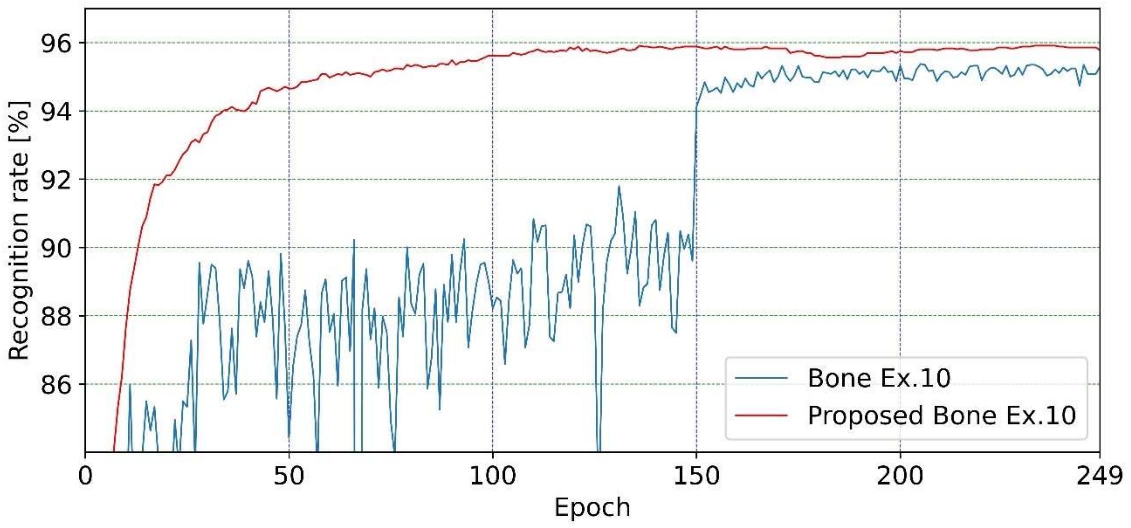




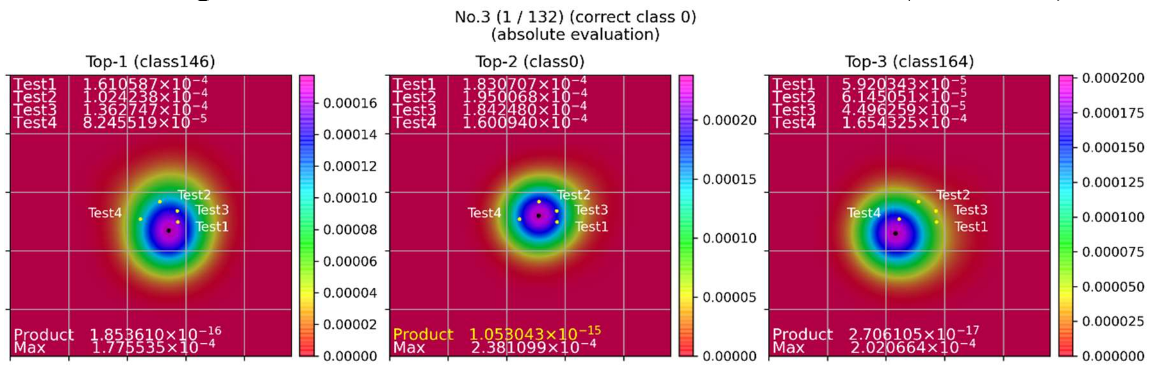


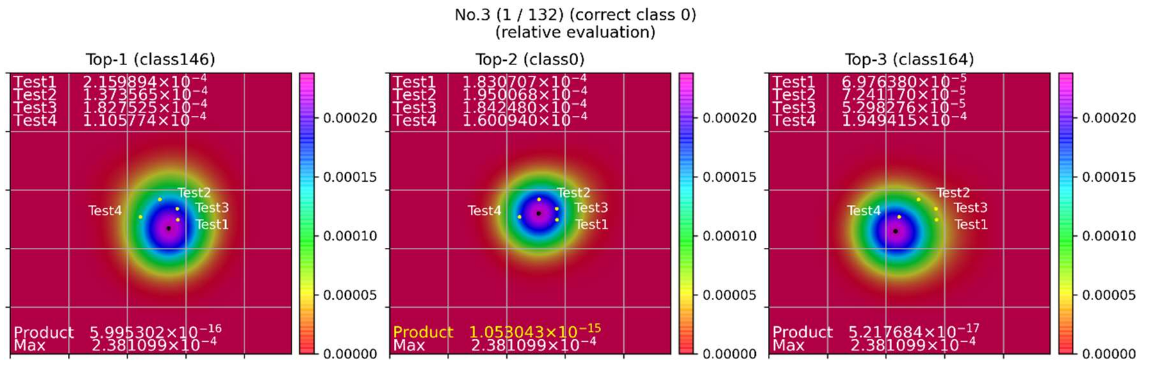

Recognition rate [%]
Epoch
Bone Ex.8
Proposed Bone Ex.8

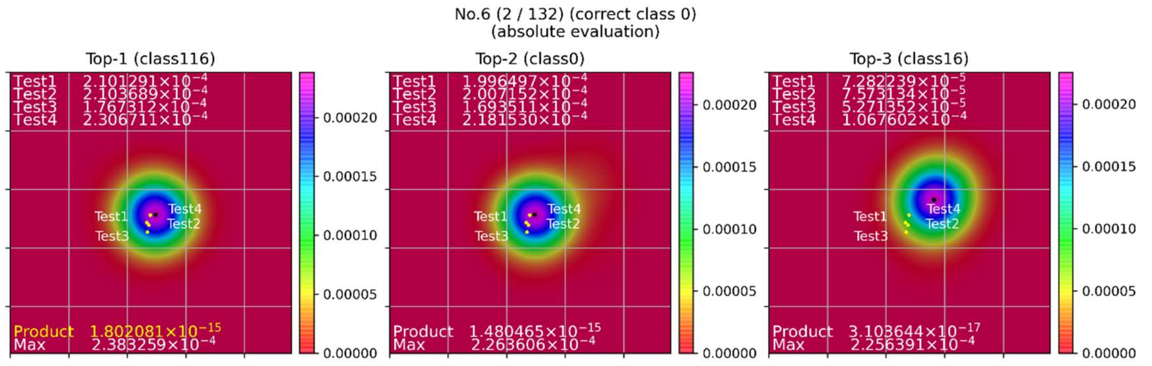

Recognition rate [%]
Epoch
Bone Ex.9
Proposed Bone Ex.9

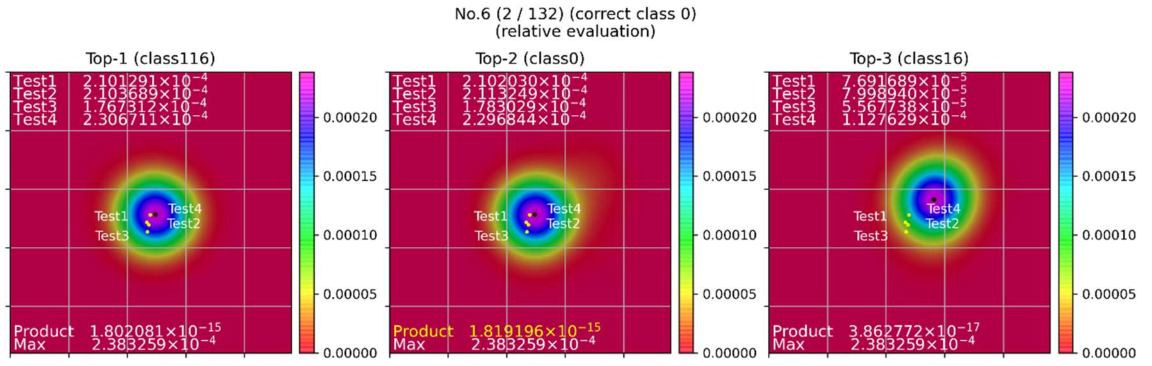

Recognition rate [%]
Epoch
Bone Ex.10
Proposed Bone Ex.10

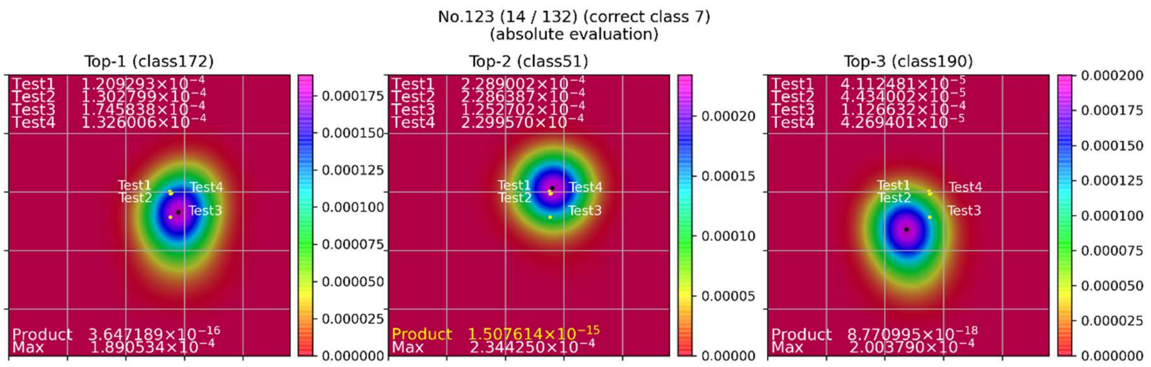
5. Heatmap's results on re-evaluation on default (35 cases)

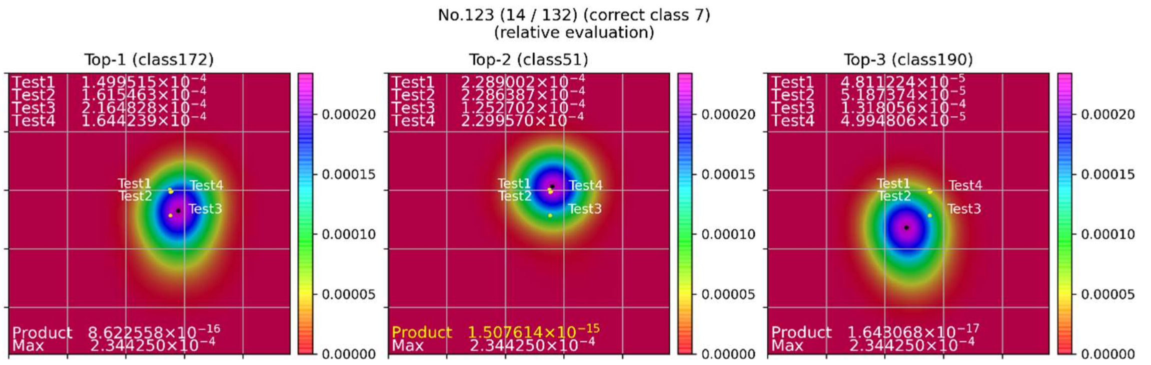


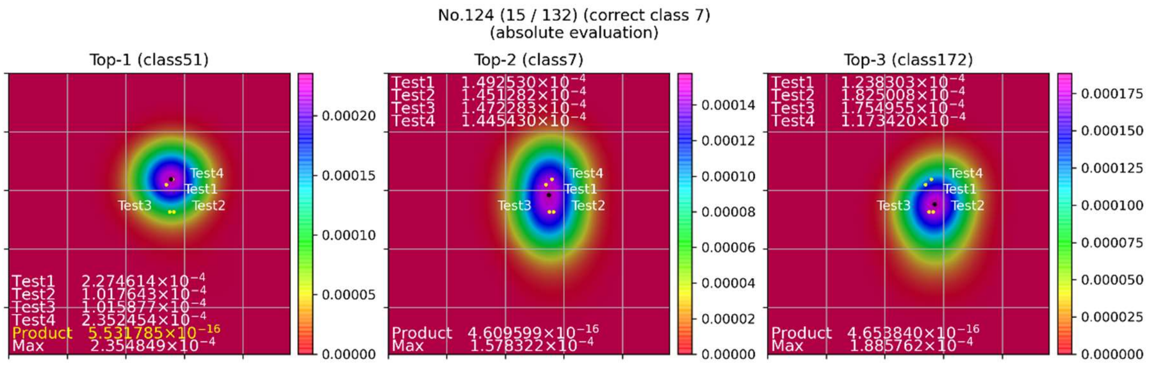


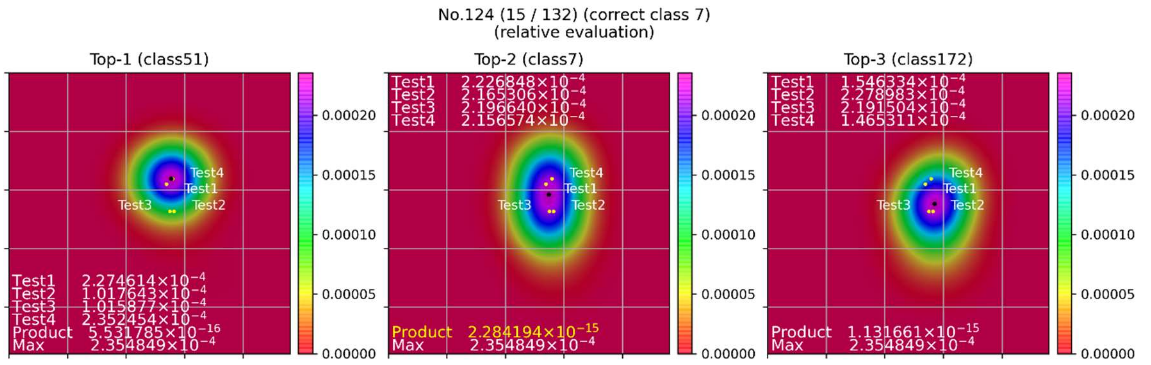


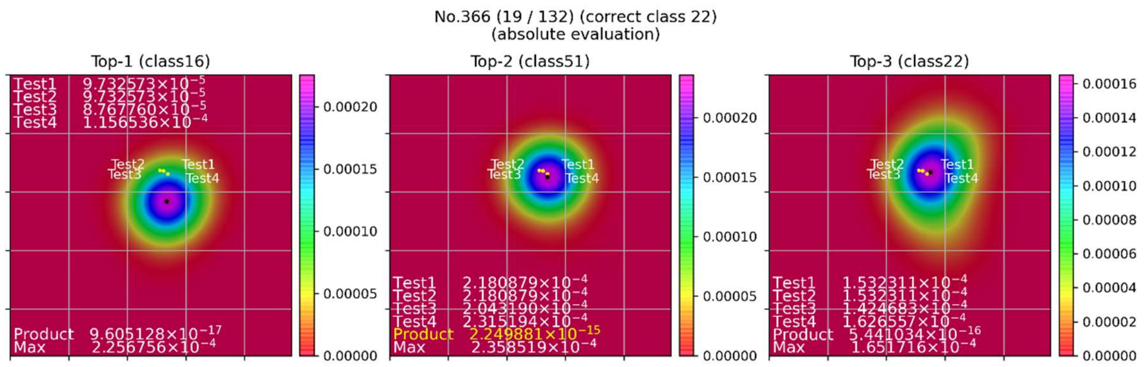


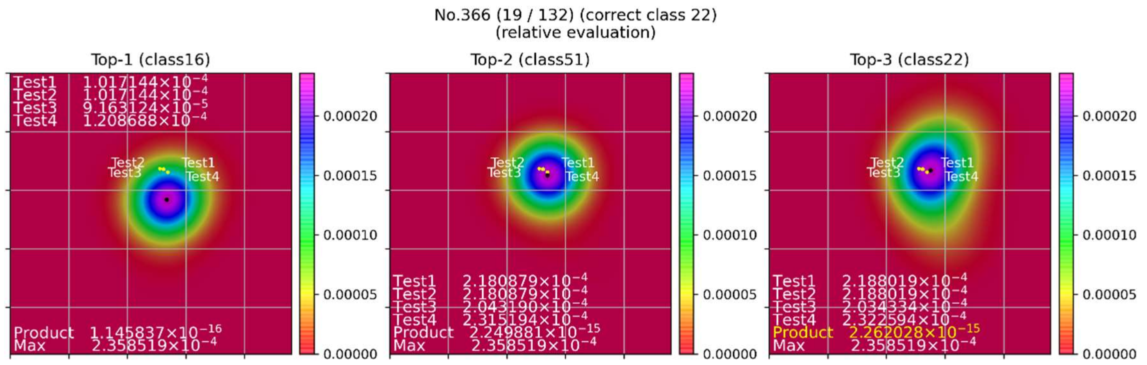

No.123 (14 / 132) (correct class 7)
(absolute evaluation)
Top-1 (class172)
Test1 1.209293×10⁻⁴
Test2 1.302799×10⁻⁴
Test3 1.745838×10⁻⁴
Test4 1.326006×10⁻⁴
Product 3.647189×10⁻¹⁶
Max 1.890534×10⁻⁴
Top-2 (class51)
Test1 2.289002×10⁻⁴
Test2 2.286387×10⁻⁴
Test3 1.252702×10⁻⁴
Test4 2.299570×10⁻⁴
Product 1.507614×10⁻¹⁵
Max 2.344250×10⁻⁴
Top-3 (class190)
Test1 4.112481×10⁻⁵
Test2 4.434002×10⁻⁵
Test3 1.126632×10⁻⁴
Test4 4.269401×10⁻⁵
Product 8.770995×10⁻¹⁸
Max 2.003790×10⁻⁴

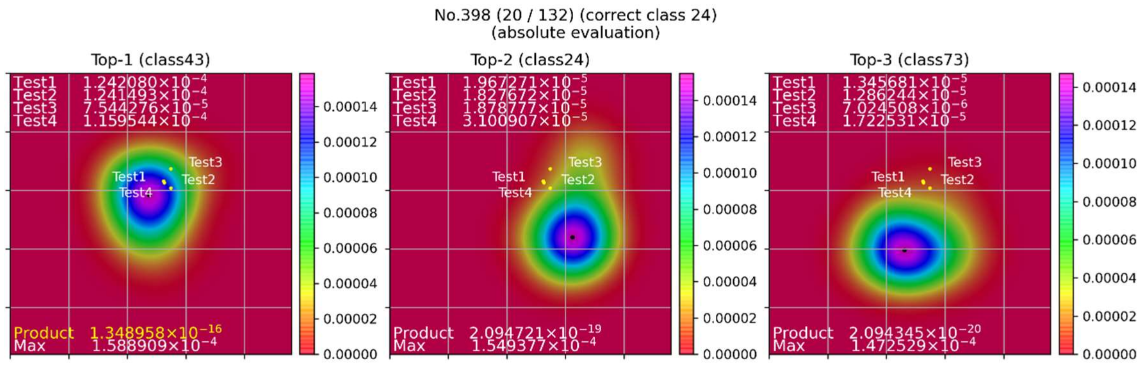

No.123 (14 / 132) (correct class 7)
(relative evaluation)
Top-1 (class172)
Test1 1.499515×10⁻⁴
Test2 1.615463×10⁻⁴
Test3 2.164828×10⁻⁴
Test4 1.644239×10⁻⁴
Product 8.622558×10⁻¹⁶
Max 2.344250×10⁻⁴
Top-2 (class51)
Test1 2.289002×10⁻⁴
Test2 2.286387×10⁻⁴
Test3 1.252702×10⁻⁴
Test4 2.299570×10⁻⁴
Product 1.507614×10⁻¹⁵
Max 2.344250×10⁻⁴
Top-3 (class190)
Test1 4.811224×10⁻⁵
Test2 5.187374×10⁻⁵
Test3 1.318056×10⁻⁴
Test4 4.994806×10⁻⁵
Product 1.643068×10⁻¹⁷
Max 2.344250×10⁻⁴

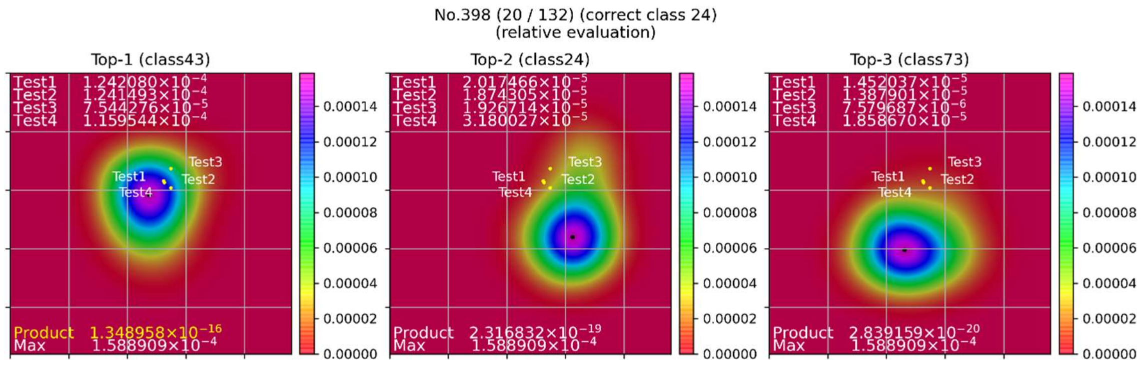

No.124 (15 / 132) (correct class 7)
(absolute evaluation)
Top-1 (class51)
Test1 2.274614×10⁻⁴
Test2 1.017643×10⁻⁴
Test3 1.015877×10⁻⁴
Test4 2.352454×10⁻⁴
Product 5.531785×10⁻¹⁶
Max 2.354849×10⁻⁴
Top-2 (class7)
Test1 1.492530×10⁻⁴
Test2 1.451282×10⁻⁴
Test3 1.472283×10⁻⁴
Test4 1.445430×10⁻⁴
Product 4.609599×10⁻¹⁶
Max 1.578322×10⁻⁴
Top-3 (class172)
Test1 1.238303×10⁻⁴
Test2 1.825008×10⁻⁴
Test3 1.754955×10⁻⁴
Test4 1.173420×10⁻⁴
Product 4.653840×10⁻¹⁶
Max 1.885762×10⁻⁴

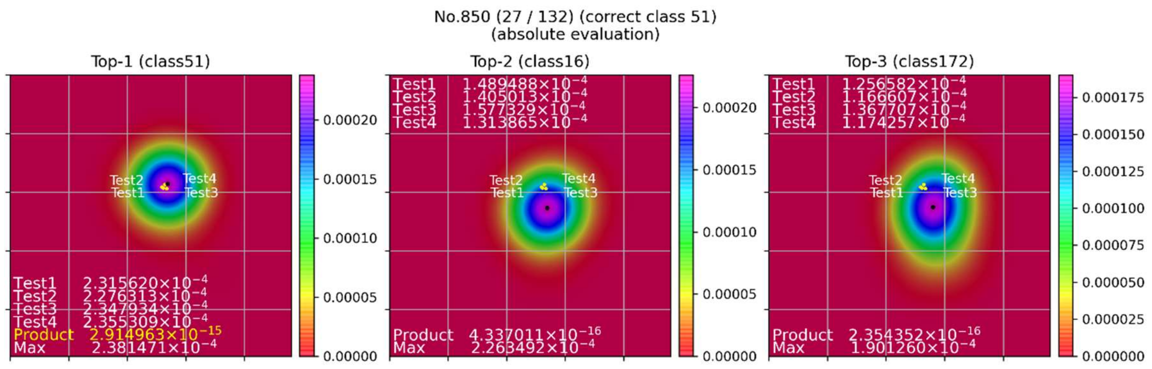

No.124 (15 / 132) (correct class 7)
(relative evaluation)
Top-1 (class51)
Test1 2.274614×10⁻⁴
Test2 1.017643×10⁻⁴
Test3 1.015877×10⁻⁴
Test4 2.352454×10⁻⁴
Product 5.531785×10⁻¹⁶
Max 2.354849×10⁻⁴
Top-2 (class7)
Test1 2.226848×10⁻⁴
Test2 2.165306×10⁻⁴
Test3 2.196640×10⁻⁴
Test4 2.156574×10⁻⁴
Product 2.284194×10⁻¹⁵
Max 2.354849×10⁻⁴
Top-3 (class172)
Test1 1.546334×10⁻⁴
Test2 2.278983×10⁻⁴
Test3 2.191504×10⁻⁴
Test4 1.465311×10⁻⁴
Product 1.131661×10⁻¹⁵
Max 2.354849×10⁻⁴

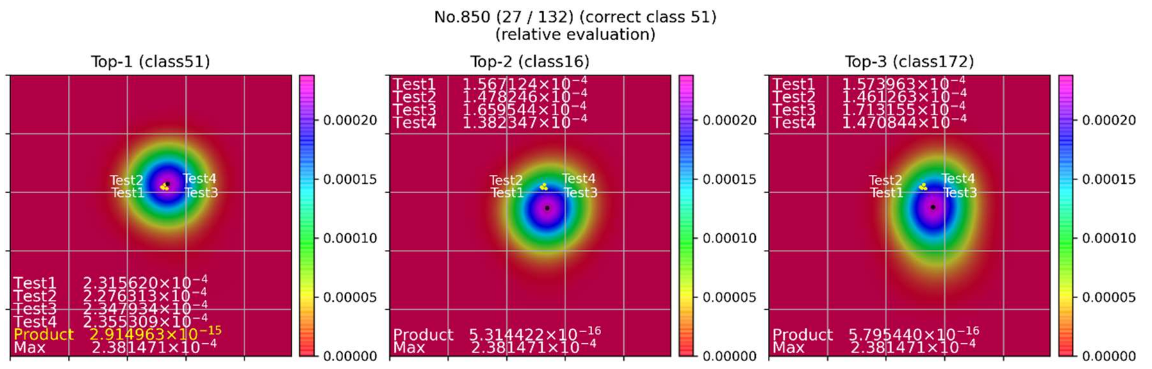

No.366 (19 / 132) (correct class 22)
(absolute evaluation)
Top-1 (class16)
Test1 9.732573×10^-5
Test2 9.732573×10^-5
Test3 8.767760×10^-5
Test4 1.156536×10^-4
Product 9.605128×10^-17
Max 2.256756×10^-4
Top-2 (class51)
Test1 2.180879×10^-4
Test2 2.180879×10^-4
Test3 2.043190×10^-4
Test4 2.315194×10^-4
Product 2.249881×10^-15
Max 2.358519×10^-4
Top-3 (class22)
Test1 1.532311×10^-4
Test2 1.532311×10^-4
Test3 1.424683×10^-4
Test4 1.626557×10^-4
Product 5.441034×10^-16
Max 1.651716×10^-4

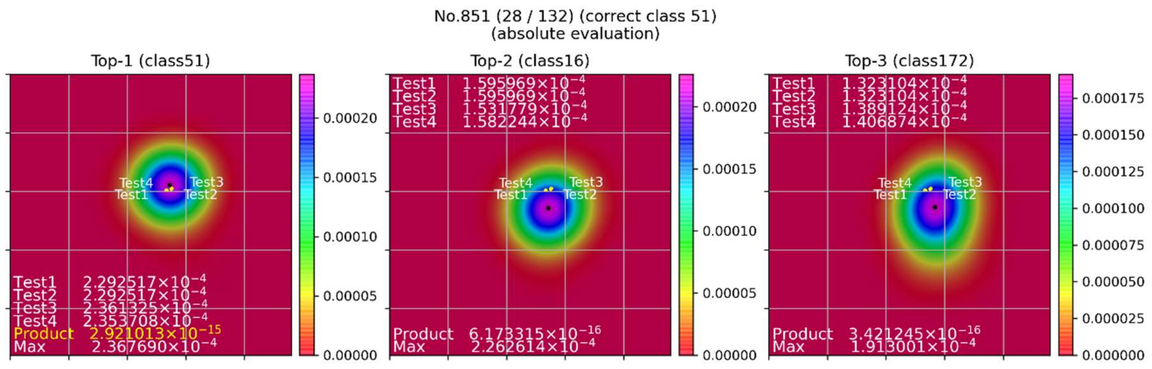

No.366 (19 / 132) (correct class 22)
(relative evaluation)
Top-1 (class16)
Test1 1.017144×10^-4
Test2 1.017144×10^-4
Test3 9.163124×10^-5
Test4 1.208688×10^-4
Product 1.145837×10^-16
Max 2.358519×10^-4
Top-2 (class51)
Test1 2.180879×10^-4
Test2 2.180879×10^-4
Test3 2.043190×10^-4
Test4 2.315194×10^-4
Product 2.249881×10^-15
Max 2.358519×10^-4
Top-3 (class22)
Test1 2.188019×10^-4
Test2 2.188019×10^-4
Test3 2.034334×10^-4
Test4 2.322594×10^-4
Product 2.262028×10^-15
Max 2.358519×10^-4

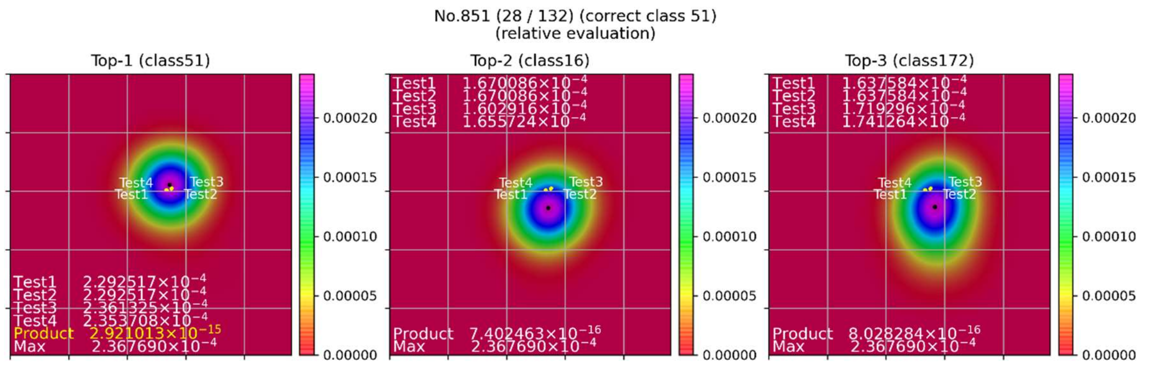

No.398 (20 / 132) (correct class 24)
(absolute evaluation)
Top-1 (class43)
Test1 1.242080×10^-4
Test2 1.241493×10^-4
Test3 7.544276×10^-5
Test4 1.159544×10^-4
Product 1.348958×10^-16
Max 1.588909×10^-4
Top-2 (class24)
Test1 1.967271×10^-5
Test2 1.827672×10^-5
Test3 1.878777×10^-5
Test4 3.100907×10^-5
Product 2.094721×10^-19
Max 1.549377×10^-4
Top-3 (class73)
Test1 1.345681×10^-5
Test2 1.286244×10^-5
Test3 7.024508×10^-6
Test4 1.722531×10^-5
Product 2.094345×10^-20
Max 1.472529×10^-4

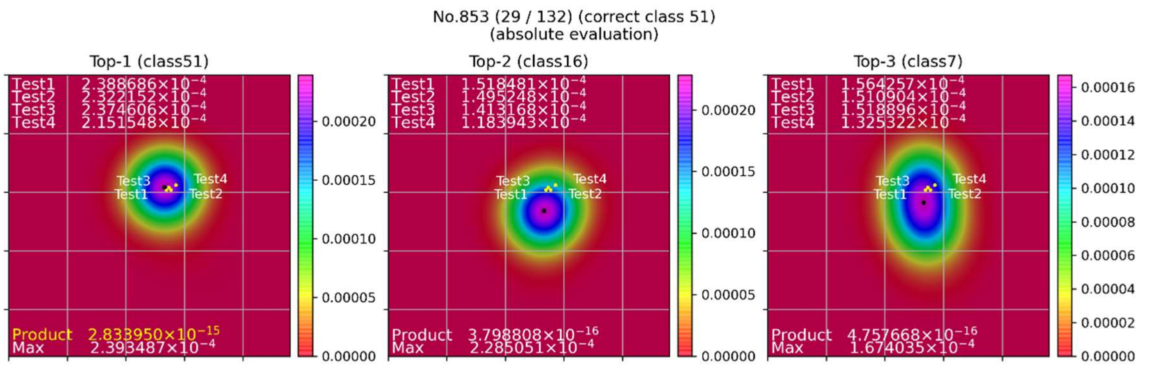

No.398 (20 / 132) (correct class 24)
(relative evaluation)
Top-1 (class43)
Test1 1.242080×10^-4
Test2 1.241493×10^-4
Test3 7.544276×10^-5
Test4 1.159544×10^-4
Product 1.348958×10^-16
Max 1.588909×10^-4
Top-2 (class24)
Test1 2.017466×10^-5
Test2 1.874305×10^-5
Test3 1.926714×10^-5
Test4 3.180027×10^-5
Product 2.316832×10^-19
Max 1.588909×10^-4
Top-3 (class73)
Test1 1.452037×10^-5
Test2 1.387901×10^-5
Test3 7.579687×10^-6
Test4 1.858670×10^-5
Product 2.839159×10^-20
Max 1.588909×10^-4

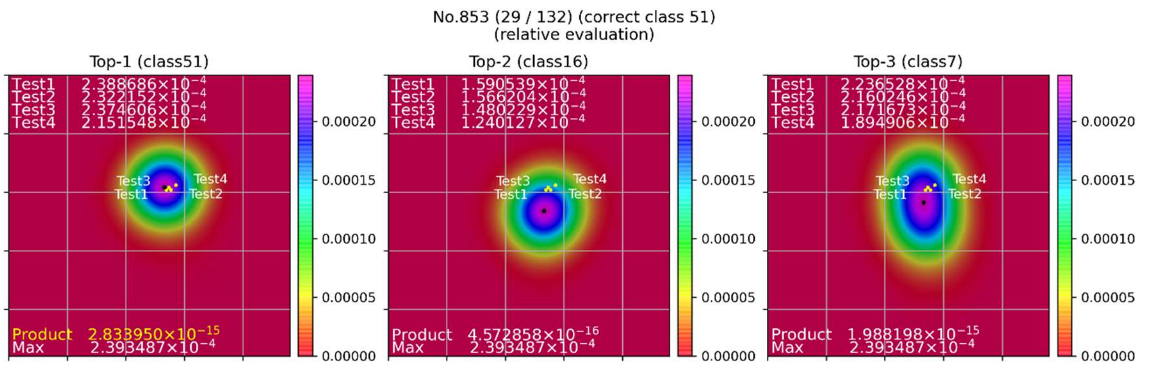

No.850 (27 / 132) (correct class 51)
(absolute evaluation)
Top-1 (class51)
Test1 2.315620×10⁻⁴
Test2 2.276313×10⁻⁴
Test3 2.347934×10⁻⁴
Test4 2.355309×10⁻⁴
Product 2.914963×10⁻¹⁵
Max 2.381471×10⁻⁴
Top-2 (class16)
Test1 1.489488×10⁻⁴
Test2 1.405013×10⁻⁴
Test3 1.577329×10⁻⁴
Test4 1.313865×10⁻⁴
Product 4.337011×10⁻¹⁶
Max 2.263492×10⁻⁴
Top-3 (class172)
Test1 1.256582×10⁻⁴
Test2 1.166607×10⁻⁴
Test3 1.367707×10⁻⁴
Test4 1.174257×10⁻⁴
Product 2.354352×10⁻¹⁶
Max 1.901260×10⁻⁴

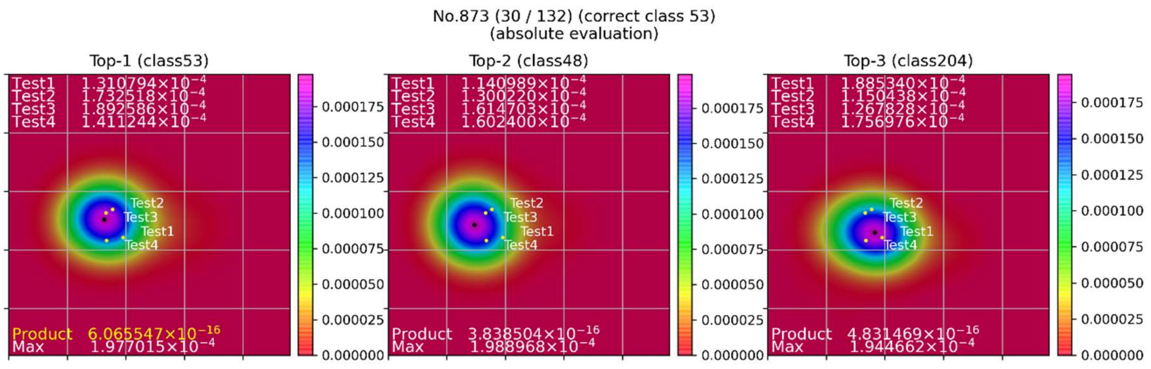

No.850 (27 / 132) (correct class 51)
(relative evaluation)
Top-1 (class51)
Test1 2.315620×10⁻⁴
Test2 2.276313×10⁻⁴
Test3 2.347934×10⁻⁴
Test4 2.355309×10⁻⁴
Product 2.914963×10⁻¹⁵
Max 2.381471×10⁻⁴
Top-2 (class16)
Test1 1.567124×10⁻⁴
Test2 1.478246×10⁻⁴
Test3 1.659544×10⁻⁴
Test4 1.382347×10⁻⁴
Product 5.314422×10⁻¹⁶
Max 2.381471×10⁻⁴
Top-3 (class172)
Test1 1.573963×10⁻⁴
Test2 1.461263×10⁻⁴
Test3 1.713155×10⁻⁴
Test4 1.470844×10⁻⁴
Product 5.795440×10⁻¹⁶
Max 2.381471×10⁻⁴

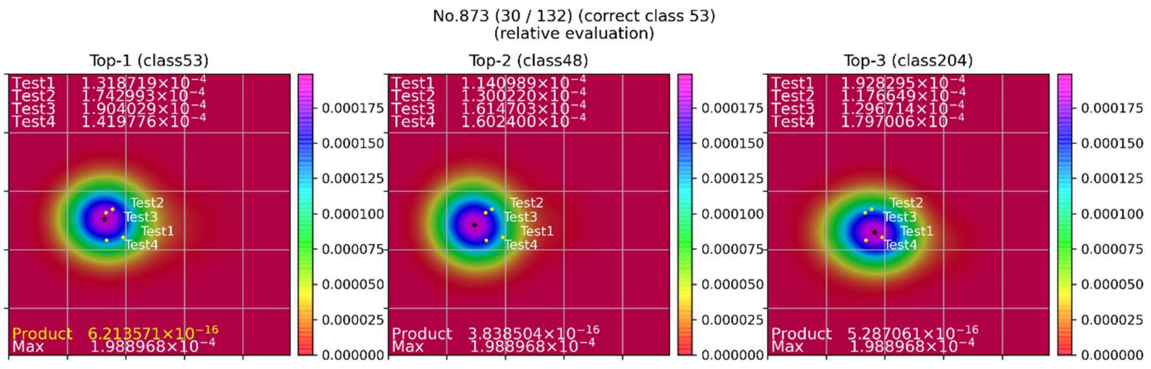

No.851 (28 / 132) (correct class 51)
(absolute evaluation)
Top-1 (class51)
Test1 2.292517×10⁻⁴
Test2 2.292517×10⁻⁴
Test3 2.361325×10⁻⁴
Test4 2.353708×10⁻⁴
Product 2.921013×10⁻¹⁵
Max 2.367690×10⁻⁴
Top-2 (class16)
Test1 1.595969×10⁻⁴
Test2 1.595969×10⁻⁴
Test3 1.531779×10⁻⁴
Test4 1.582244×10⁻⁴
Product 6.173315×10⁻¹⁶
Max 2.262614×10⁻⁴
Top-3 (class172)
Test1 1.323104×10⁻⁴
Test2 1.323104×10⁻⁴
Test3 1.389124×10⁻⁴
Test4 1.406874×10⁻⁴
Product 3.421245×10⁻¹⁶
Max 1.913001×10⁻⁴

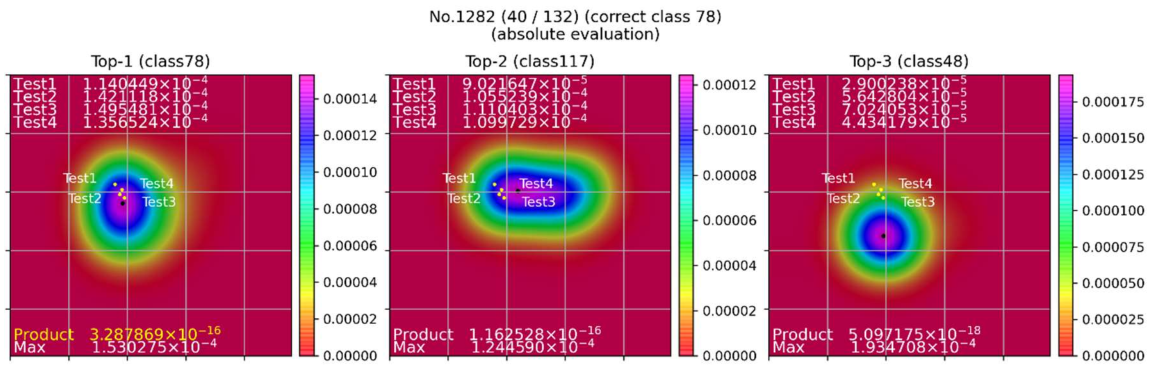

No.851 (28 / 132) (correct class 51)
(relative evaluation)
Top-1 (class51)
Test1 2.292517×10⁻⁴
Test2 2.292517×10⁻⁴
Test3 2.361325×10⁻⁴
Test4 2.353708×10⁻⁴
Product 2.921013×10⁻¹⁵
Max 2.367690×10⁻⁴
Top-2 (class16)
Test1 1.670086×10⁻⁴
Test2 1.670086×10⁻⁴
Test3 1.602916×10⁻⁴
Test4 1.655724×10⁻⁴
Product 7.402463×10⁻¹⁶
Max 2.367690×10⁻⁴
Top-3 (class172)
Test1 1.637584×10⁻⁴
Test2 1.637584×10⁻⁴
Test3 1.719296×10⁻⁴
Test4 1.741264×10⁻⁴
Product 8.028284×10⁻¹⁶
Max 2.367690×10⁻⁴

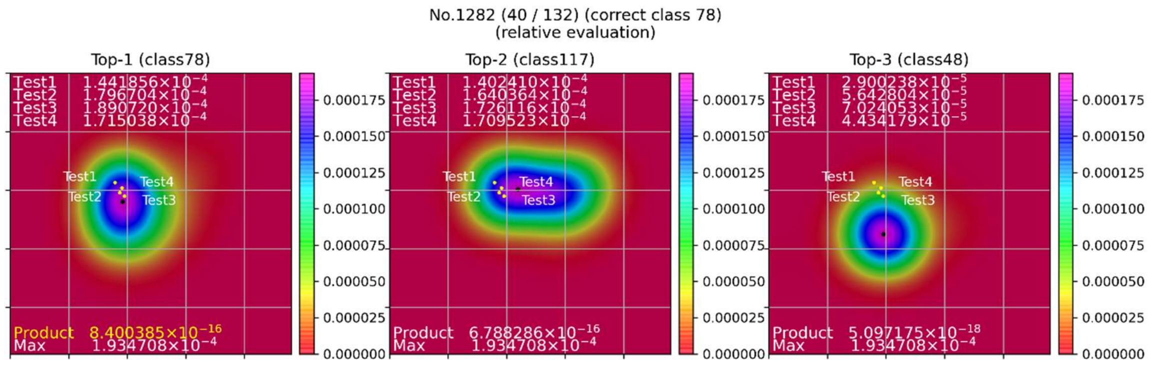

No.853 (29 / 132) (correct class 51)
(absolute evaluation)
Top-1 (class51)
Test1 2.388686×10^{-4}
Test2 2.322152×10^{-4}
Test3 2.374606×10^{-4}
Test4 2.151548×10^{-4}
Product 2.833950×10^{-15}
Max 2.393487×10^{-4}
Top-2 (class16)
Test1 1.518481×10^{-4}
Test2 1.495248×10^{-4}
Test3 1.413168×10^{-4}
Test4 1.183943×10^{-4}
Product 3.798808×10^{-16}
Max 2.285051×10^{-4}
Top-3 (class7)
Test1 1.564257×10^{-4}
Test2 1.510904×10^{-4}
Test3 1.518896×10^{-4}
Test4 1.325322×10^{-4}
Product 4.757668×10^{-16}
Max 1.674035×10^{-4}

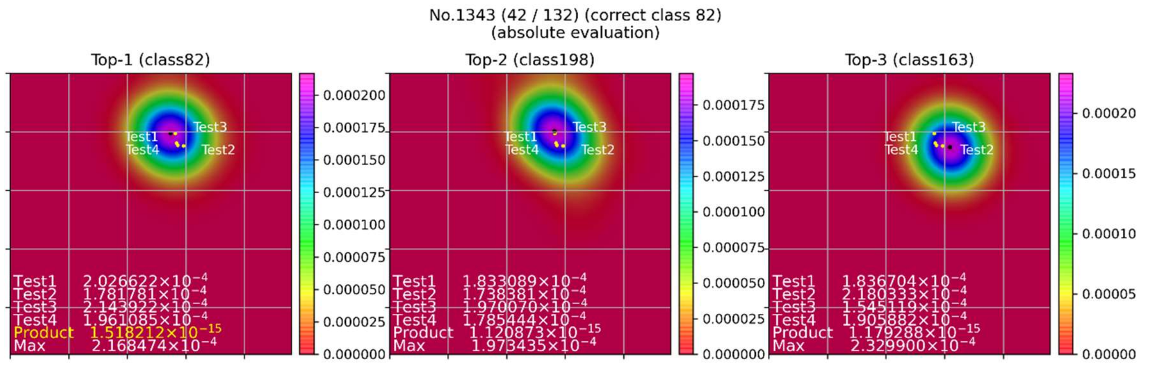

No.853 (29 / 132) (correct class 51)
(relative evaluation)
Top-1 (class51)
Test1 2.388686×10^{-4}
Test2 2.322152×10^{-4}
Test3 2.374606×10^{-4}
Test4 2.151548×10^{-4}
Product 2.833950×10^{-15}
Max 2.393487×10^{-4}
Top-2 (class16)
Test1 1.590539×10^{-4}
Test2 1.566204×10^{-4}
Test3 1.480229×10^{-4}
Test4 1.240127×10^{-4}
Product 4.572858×10^{-16}
Max 2.393487×10^{-4}
Top-3 (class7)
Test1 2.236528×10^{-4}
Test2 2.160246×10^{-4}
Test3 2.171673×10^{-4}
Test4 1.894906×10^{-4}
Product 1.988198×10^{-15}
Max 2.393487×10^{-4}

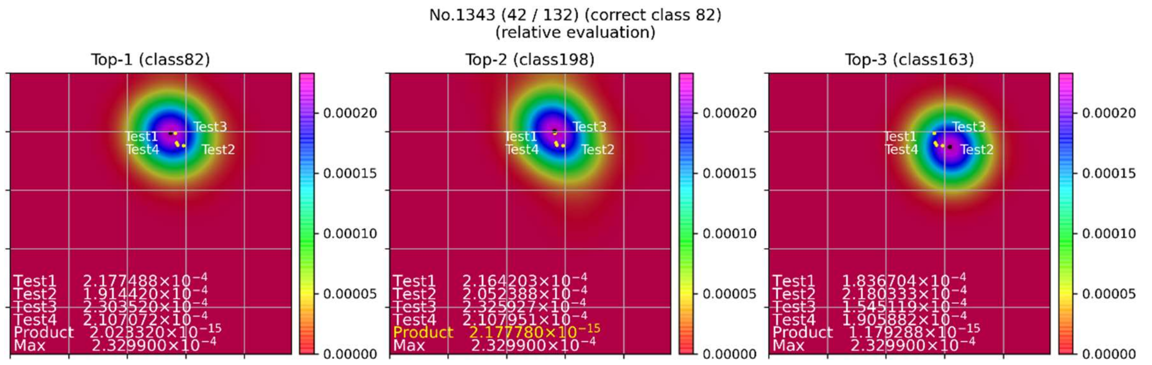

No.873 (30 / 132) (correct class 53)
(absolute evaluation)
Top-1 (class53)
Test1 1.310794×10^{-4}
Test2 1.732518×10^{-4}
Test3 1.892586×10^{-4}
Test4 1.411244×10^{-4}
Product 6.065547×10^{-16}
Max 1.977015×10^{-4}
Top-2 (class48)
Test1 1.140989×10^{-4}
Test2 1.300220×10^{-4}
Test3 1.614703×10^{-4}
Test4 1.602400×10^{-4}
Product 3.838504×10^{-16}
Max 1.988968×10^{-4}
Top-3 (class204)
Test1 1.885340×10^{-4}
Test2 1.150438×10^{-4}
Test3 1.267828×10^{-4}
Test4 1.756976×10^{-4}
Product 4.831469×10^{-16}
Max 1.944662×10^{-4}

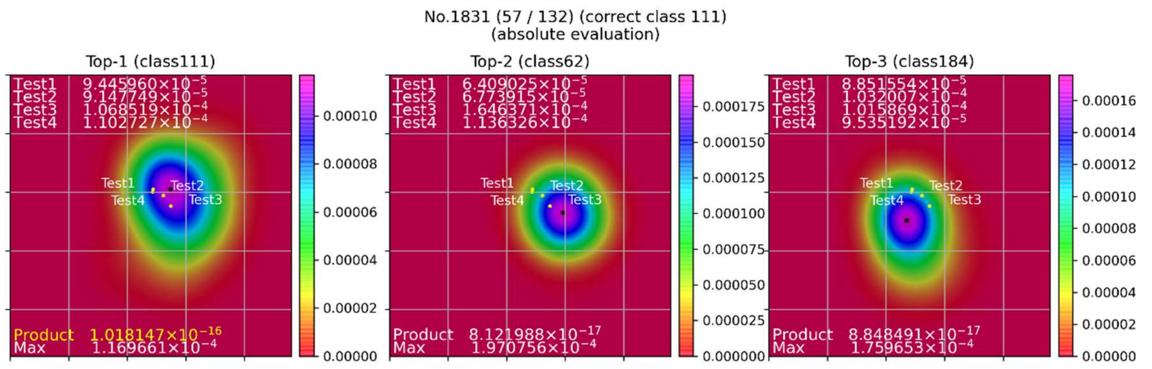

No.873 (30 / 132) (correct class 53)
(relative evaluation)
Top-1 (class53)
Test1 1.318719×10^{-4}
Test2 1.742993×10^{-4}
Test3 1.904029×10^{-4}
Test4 1.419776×10^{-4}
Product 6.213571×10^{-16}
Max 1.988968×10^{-4}
Top-2 (class48)
Test1 1.140989×10^{-4}
Test2 1.300220×10^{-4}
Test3 1.614703×10^{-4}
Test4 1.602400×10^{-4}
Product 3.838504×10^{-16}
Max 1.988968×10^{-4}
Top-3 (class204)
Test1 1.928295×10^{-4}
Test2 1.176649×10^{-4}
Test3 1.296714×10^{-4}
Test4 1.797006×10^{-4}
Product 5.287061×10^{-16}
Max 1.988968×10^{-4}

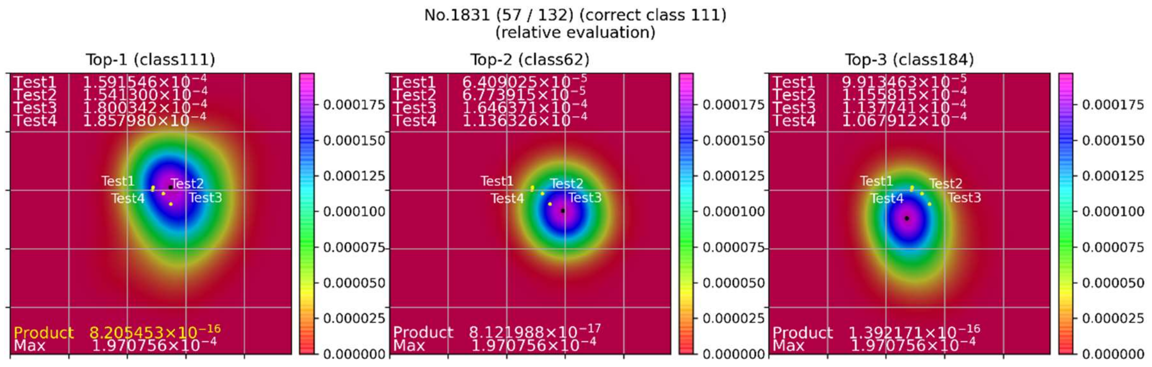

No.1282 (40 / 132) (correct class 78)
(absolute evaluation)
Top-1 (class78)
Test1 1.140449×10⁻⁴
Test2 1.421118×10⁻⁴
Test3 1.495481×10⁻⁴
Test4 1.356524×10⁻⁴
Product 3.287869×10⁻¹⁶
Max 1.530275×10⁻⁴
Top-2 (class117)
Test1 9.021647×10⁻⁵
Test2 1.055239×10⁻⁴
Test3 1.110403×10⁻⁴
Test4 1.099729×10⁻⁴
Product 1.162528×10⁻¹⁶
Max 1.244590×10⁻⁴
Top-3 (class48)
Test1 2.900238×10⁻⁵
Test2 5.642804×10⁻⁵
Test3 7.024053×10⁻⁵
Test4 4.434179×10⁻⁵
Product 5.097175×10⁻¹⁸
Max 1.934708×10⁻⁴

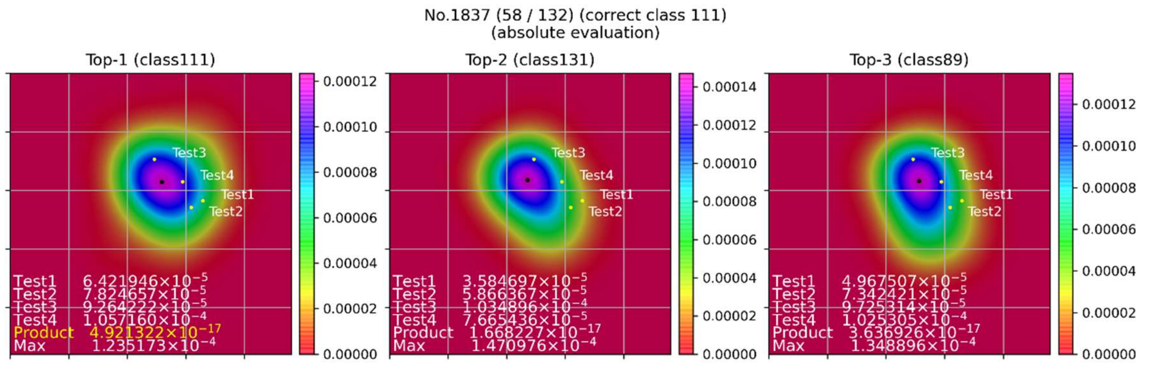

No.1282 (40 / 132) (correct class 78)
(relative evaluation)
Top-1 (class78)
Test1 1.441856×10⁻⁴
Test2 1.796704×10⁻⁴
Test3 1.890720×10⁻⁴
Test4 1.715038×10⁻⁴
Product 8.400385×10⁻¹⁶
Max 1.934708×10⁻⁴
Top-2 (class117)
Test1 1.402410×10⁻⁴
Test2 1.640364×10⁻⁴
Test3 1.726116×10⁻⁴
Test4 1.709523×10⁻⁴
Product 6.788286×10⁻¹⁶
Max 1.934708×10⁻⁴
Top-3 (class48)
Test1 2.900238×10⁻⁵
Test2 5.642804×10⁻⁵
Test3 7.024053×10⁻⁵
Test4 4.434179×10⁻⁵
Product 5.097175×10⁻¹⁸
Max 1.934708×10⁻⁴

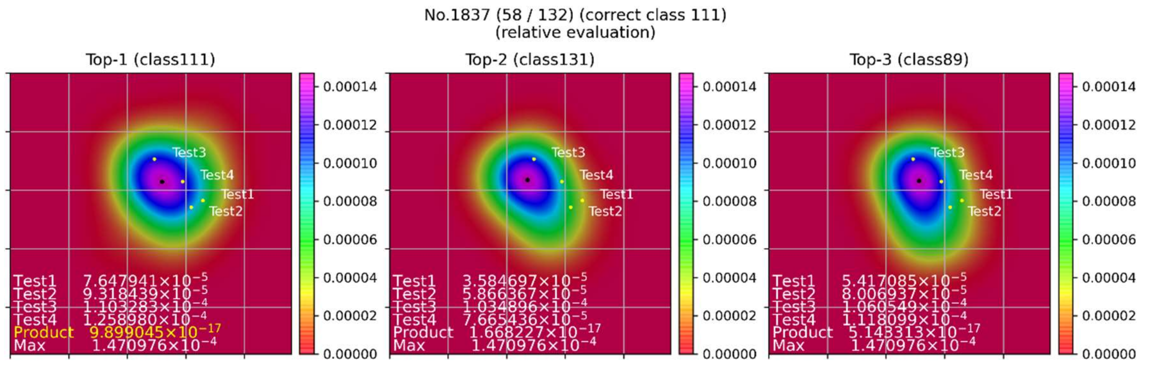

No.1343 (42 / 132) (correct class 82)
(absolute evaluation)
Top-1 (class82)
Test1 2.026622×10⁻⁴
Test2 1.781781×10⁻⁴
Test3 2.143922×10⁻⁴
Test4 1.961085×10⁻⁴
Product 1.518212×10⁻¹⁵
Max 2.168474×10⁻⁴
Top-2 (class198)
Test1 1.833089×10⁻⁴
Test2 1.738381×10⁻⁴
Test3 1.970070×10⁻⁴
Test4 1.785444×10⁻⁴
Product 1.120873×10⁻¹⁵
Max 1.973435×10⁻⁴
Top-3 (class163)
Test1 1.836704×10⁻⁴
Test2 2.180333×10⁻⁴
Test3 1.545119×10⁻⁴
Test4 1.905882×10⁻⁴
Product 1.179288×10⁻¹⁵
Max 2.329900×10⁻⁴

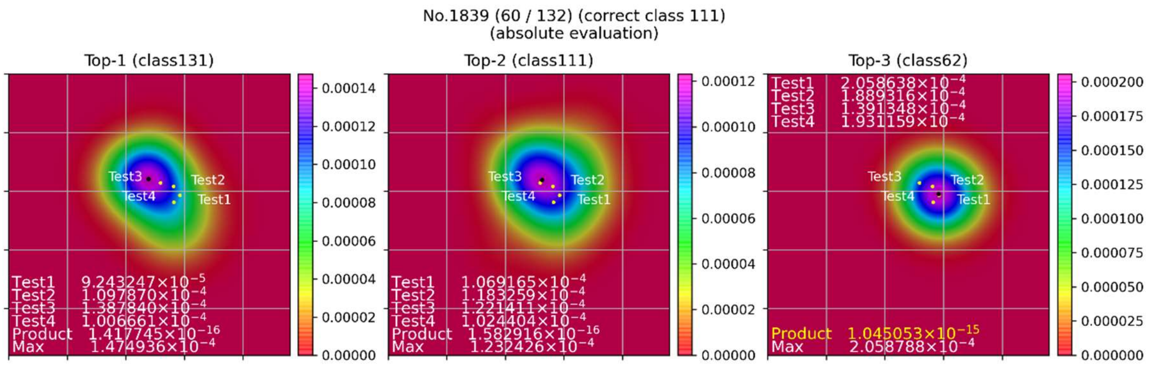

No.1343 (42 / 132) (correct class 82)
(relative evaluation)
Top-1 (class82)
Test1 2.177488×10⁻⁴
Test2 1.914420×10⁻⁴
Test3 2.303520×10⁻⁴
Test4 2.107072×10⁻⁴
Product 2.023320×10⁻¹⁵
Max 2.329900×10⁻⁴
Top-2 (class198)
Test1 2.164203×10⁻⁴
Test2 2.052388×10⁻⁴
Test3 2.325927×10⁻⁴
Test4 2.107951×10⁻⁴
Product 2.177780×10⁻¹⁵
Max 2.329900×10⁻⁴
Top-3 (class163)
Test1 1.836704×10⁻⁴
Test2 2.180333×10⁻⁴
Test3 1.545119×10⁻⁴
Test4 1.905882×10⁻⁴
Product 1.179288×10⁻¹⁵
Max 2.329900×10⁻⁴

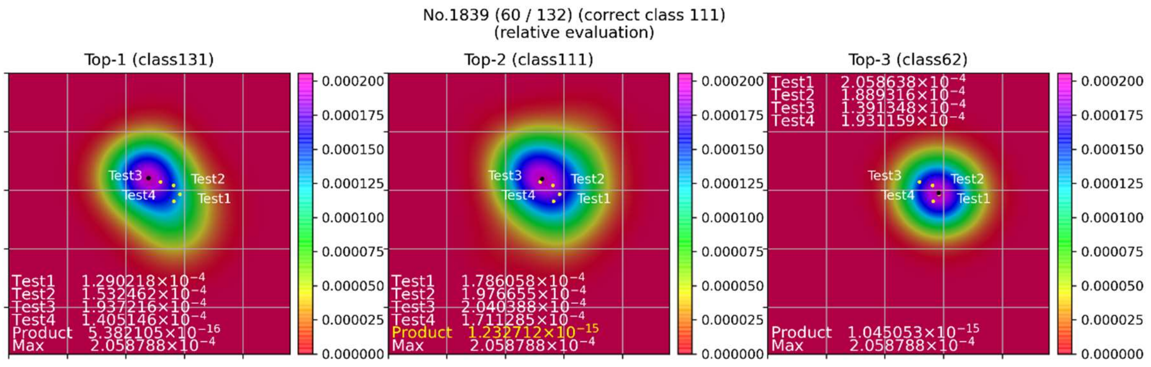

No.1831 (57 / 132) (correct class 111)
(absolute evaluation)
Top-1 (class111)
Test1 9.445960×10^{-5}
Test2 9.147749×10^{-5}
Test3 1.068519×10^{-4}
Test4 1.102727×10^{-4}
Product 1.018147×10^{-16}
Max 1.169661×10^{-4}
Top-2 (class62)
Test1 6.409025×10^{-5}
Test2 6.773915×10^{-5}
Test3 1.646371×10^{-4}
Test4 1.136326×10^{-4}
Product 8.121988×10^{-17}
Max 1.970756×10^{-4}
Top-3 (class184)
Test1 8.851554×10^{-5}
Test2 1.032007×10^{-4}
Test3 1.015869×10^{-4}
Test4 9.535192×10^{-5}
Product 8.848491×10^{-17}
Max 1.759653×10^{-4}

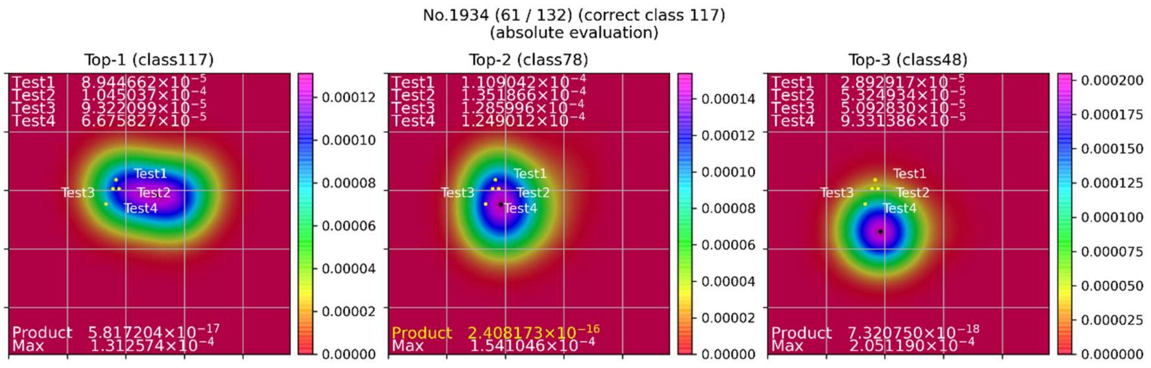

No.1831 (57 / 132) (correct class 111)
(relative evaluation)
Top-1 (class111)
Test1 1.591546×10^{-4}
Test2 1.541300×10^{-4}
Test3 1.800342×10^{-4}
Test4 1.857980×10^{-4}
Product 8.205453×10^{-16}
Max 1.970756×10^{-4}
Top-2 (class62)
Test1 6.409025×10^{-5}
Test2 6.773915×10^{-5}
Test3 1.646371×10^{-4}
Test4 1.136326×10^{-4}
Product 8.121988×10^{-17}
Max 1.970756×10^{-4}
Top-3 (class184)
Test1 9.913463×10^{-5}
Test2 1.155815×10^{-4}
Test3 1.137741×10^{-4}
Test4 1.067912×10^{-4}
Product 1.392171×10^{-16}
Max 1.970756×10^{-4}

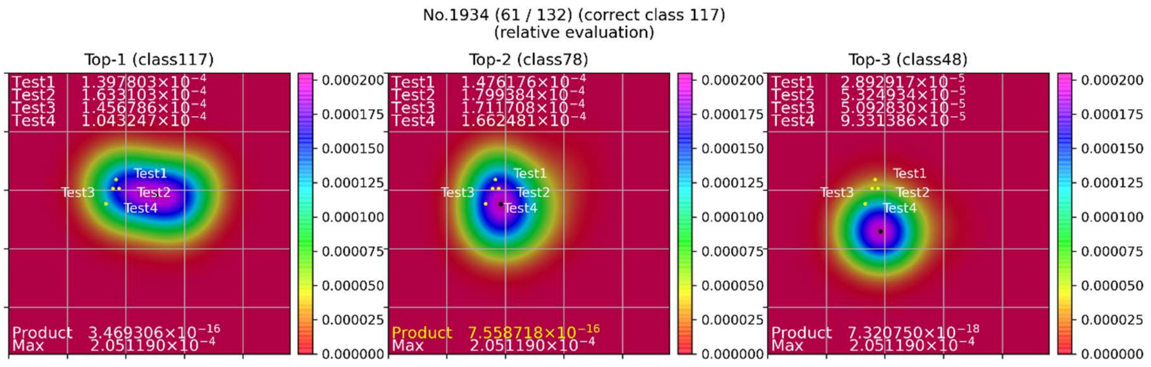

No.1837 (58 / 132) (correct class 111)
(absolute evaluation)
Top-1 (class111)
Test1 6.421946×10^{-5}
Test2 7.824657×10^{-5}
Test3 9.264222×10^{-5}
Test4 1.057160×10^{-4}
Product 4.921322×10^{-17}
Max 1.235173×10^{-4}
Top-2 (class131)
Test1 3.584697×10^{-5}
Test2 5.866367×10^{-5}
Test3 1.034896×10^{-4}
Test4 7.665436×10^{-5}
Product 1.668227×10^{-17}
Max 1.470976×10^{-4}
Top-3 (class89)
Test1 4.967507×10^{-5}
Test2 7.342421×10^{-5}
Test3 9.725314×10^{-5}
Test4 1.025305×10^{-4}
Product 3.636926×10^{-17}
Max 1.348896×10^{-4}

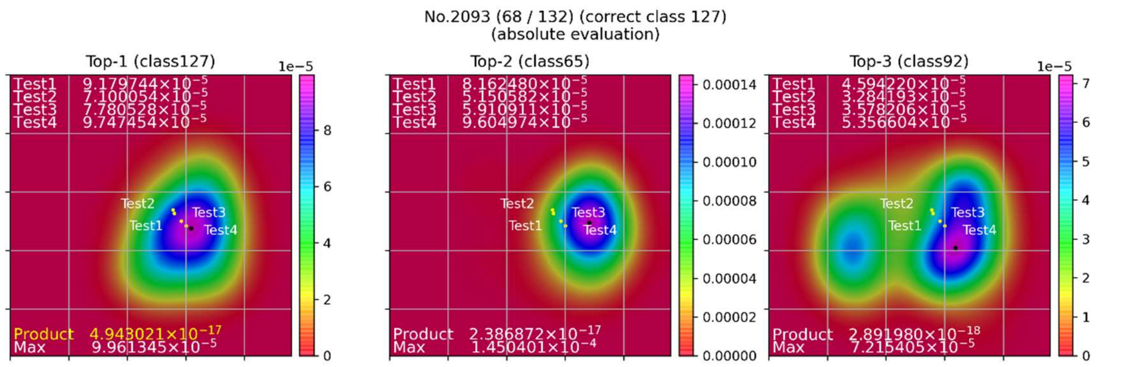

No.1837 (58 / 132) (correct class 111)
(relative evaluation)
Top-1 (class111)
Test1 7.647941×10^{-5}
Test2 9.318439×10^{-5}
Test3 1.103283×10^{-4}
Test4 1.258980×10^{-4}
Product 9.899045×10^{-17}
Max 1.470976×10^{-4}
Top-2 (class131)
Test1 3.584697×10^{-5}
Test2 5.866367×10^{-5}
Test3 1.034896×10^{-4}
Test4 7.665436×10^{-5}
Product 1.668227×10^{-17}
Max 1.470976×10^{-4}
Top-3 (class89)
Test1 5.417085×10^{-5}
Test2 8.006937×10^{-5}
Test3 1.060549×10^{-4}
Test4 1.118099×10^{-4}
Product 5.143313×10^{-17}
Max 1.470976×10^{-4}

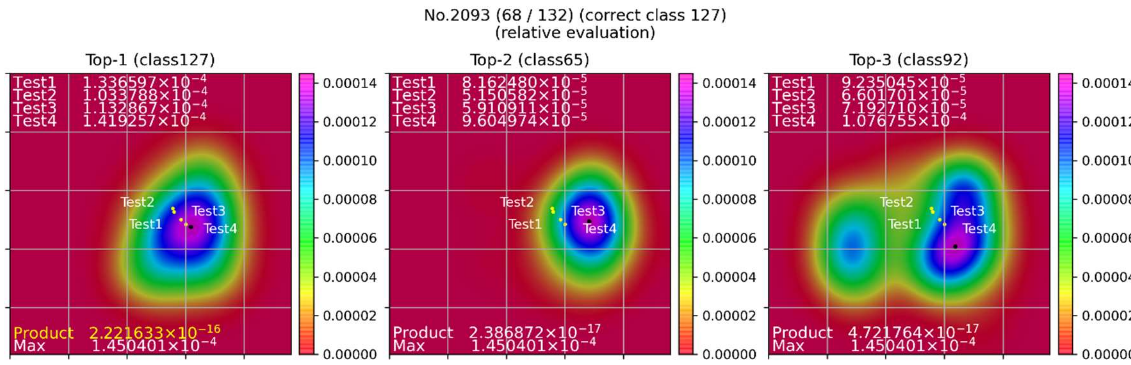

No.1839 (60 / 132) (correct class 111)
(absolute evaluation)
Top-1 (class131)
Test1 9.243247×10⁻⁵
Test2 1.097870×10⁻⁴
Test3 1.387840×10⁻⁴
Test4 1.006661×10⁻⁴
Product 1.417745×10⁻¹⁶
Max 1.474936×10⁻⁴
Top-2 (class111)
Test1 1.069165×10⁻⁴
Test2 1.183259×10⁻⁴
Test3 1.221411×10⁻⁴
Test4 1.024404×10⁻⁴
Product 1.582916×10⁻¹⁶
Max 1.232426×10⁻⁴
Top-3 (class62)
Test1 2.058638×10⁻⁴
Test2 1.889316×10⁻⁴
Test3 1.391348×10⁻⁴
Test4 1.931159×10⁻⁴
Product 1.045053×10⁻¹⁵
Max 2.058788×10⁻⁴

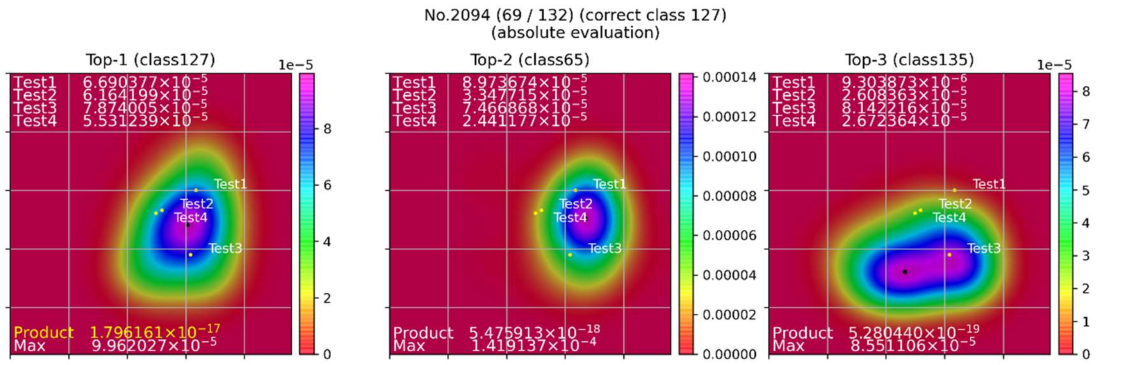

No.1839 (60 / 132) (correct class 111)
(relative evaluation)
Top-1 (class131)
Test1 1.290218×10⁻⁴
Test2 1.532462×10⁻⁴
Test3 1.937216×10⁻⁴
Test4 1.405146×10⁻⁴
Product 5.382105×10⁻¹⁶
Max 2.058788×10⁻⁴
Top-2 (class111)
Test1 1.786058×10⁻⁴
Test2 1.976655×10⁻⁴
Test3 2.040388×10⁻⁴
Test4 1.711285×10⁻⁴
Product 1.232712×10⁻¹⁵
Max 2.058788×10⁻⁴
Top-3 (class62)
Test1 2.058638×10⁻⁴
Test2 1.889316×10⁻⁴
Test3 1.391348×10⁻⁴
Test4 1.931159×10⁻⁴
Product 1.045053×10⁻¹⁵
Max 2.058788×10⁻⁴

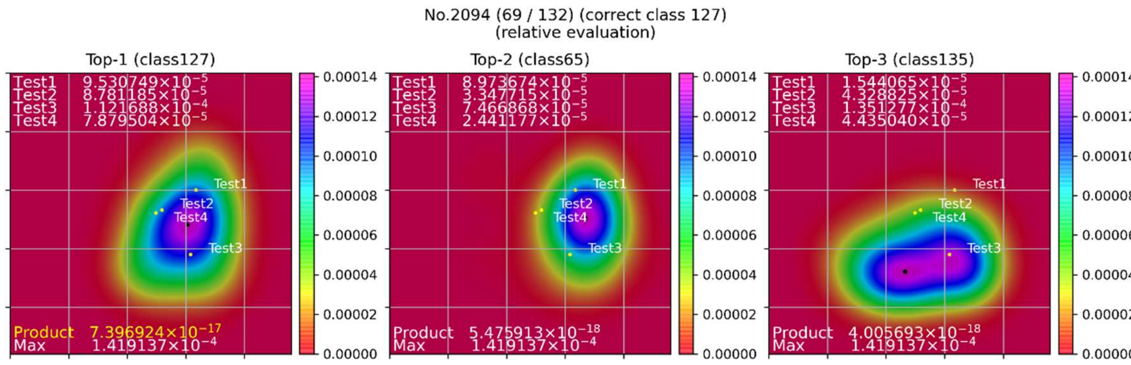

No.1934 (61 / 132) (correct class 117)
(absolute evaluation)
Top-1 (class117)
Test1 8.944662×10⁻⁵
Test2 1.045037×10⁻⁴
Test3 9.322099×10⁻⁵
Test4 6.675827×10⁻⁵
Product 5.817204×10⁻¹⁷
Max 1.312574×10⁻⁴
Top-2 (class78)
Test1 1.109042×10⁻⁴
Test2 1.351866×10⁻⁴
Test3 1.285996×10⁻⁴
Test4 1.249012×10⁻⁴
Product 2.408173×10⁻¹⁶
Max 1.541046×10⁻⁴
Top-3 (class48)
Test1 2.892917×10⁻⁵
Test2 5.324934×10⁻⁵
Test3 5.092830×10⁻⁵
Test4 9.331386×10⁻⁵
Product 7.320750×10⁻¹⁸
Max 2.051190×10⁻⁴

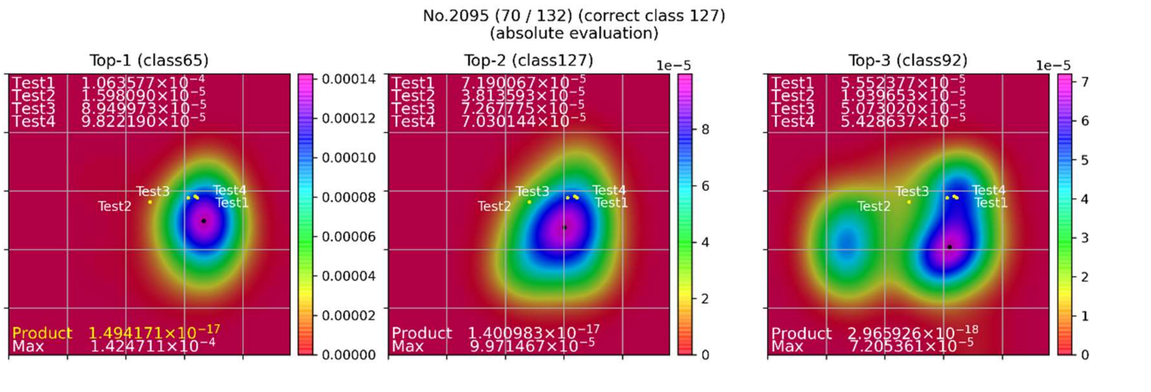

No.1934 (61 / 132) (correct class 117)
(relative evaluation)
Top-1 (class117)
Test1 1.397803×10⁻⁴
Test2 1.633103×10⁻⁴
Test3 1.456786×10⁻⁴
Test4 1.043247×10⁻⁴
Product 3.469306×10⁻¹⁶
Max 2.051190×10⁻⁴
Top-2 (class78)
Test1 1.476176×10⁻⁴
Test2 1.799384×10⁻⁴
Test3 1.711708×10⁻⁴
Test4 1.662481×10⁻⁴
Product 7.558718×10⁻¹⁶
Max 2.051190×10⁻⁴
Top-3 (class48)
Test1 2.892917×10⁻⁵
Test2 5.324934×10⁻⁵
Test3 5.092830×10⁻⁵
Test4 9.331386×10⁻⁵
Product 7.320750×10⁻¹⁸
Max 2.051190×10⁻⁴

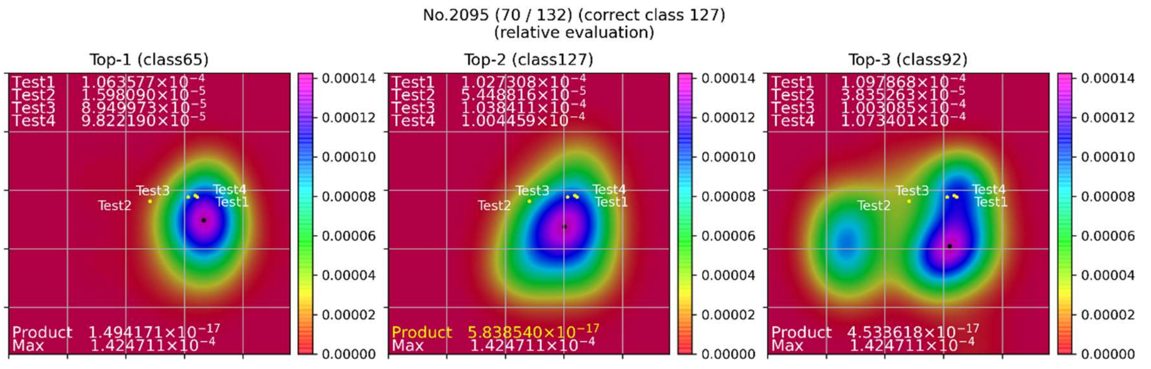

No.2093 (68 / 132) (correct class 127)
(absolute evaluation)
Top-1 (class127)
1e−5
Test1 9.179744×10⁻⁵
Test2 7.100054×10⁻⁵
Test3 7.780528×10⁻⁵
Test4 9.747454×10⁻⁵
Product 4.943021×10⁻¹⁷
Max 9.961345×10⁻⁵
Top-2 (class65)
Test1 8.162480×10⁻⁵
Test2 5.150582×10⁻⁵
Test3 5.910911×10⁻⁵
Test4 9.604974×10⁻⁵
Product 2.386872×10⁻¹⁷
Max 1.450401×10⁻⁴
Top-3 (class92)
1e−5
Test1 4.594220×10⁻⁵
Test2 3.284193×10⁻⁵
Test3 3.578206×10⁻⁵
Test4 5.356604×10⁻⁵
Product 2.891980×10⁻¹⁸
Max 7.215405×10⁻⁵

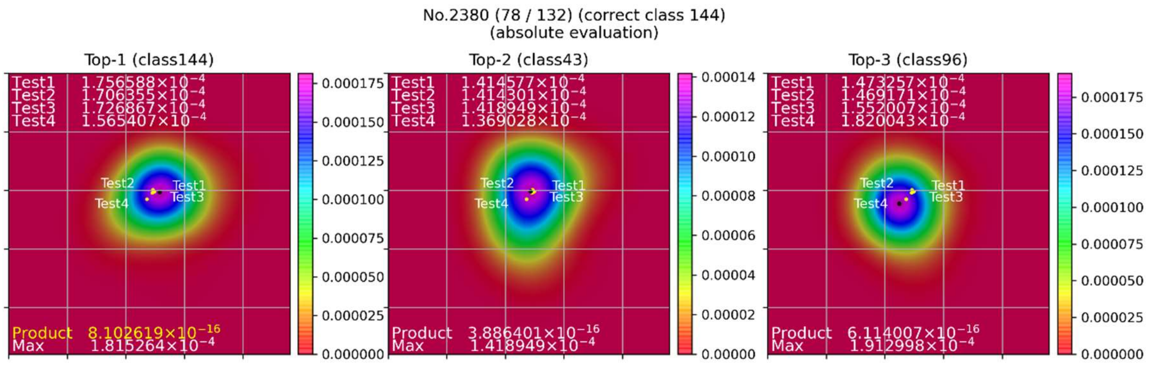

No.2093 (68 / 132) (correct class 127)
(relative evaluation)
Top-1 (class127)
Test1 1.336597×10⁻⁴
Test2 1.033788×10⁻⁴
Test3 1.132867×10⁻⁴
Test4 1.419257×10⁻⁴
Product 2.221633×10⁻¹⁶
Max 1.450401×10⁻⁴
Top-2 (class65)
Test1 8.162480×10⁻⁵
Test2 5.150582×10⁻⁵
Test3 5.910911×10⁻⁵
Test4 9.604974×10⁻⁵
Product 2.386872×10⁻¹⁷
Max 1.450401×10⁻⁴
Top-3 (class92)
Test1 9.235045×10⁻⁵
Test2 6.601701×10⁻⁵
Test3 7.192710×10⁻⁵
Test4 1.076755×10⁻⁴
Product 4.721764×10⁻¹⁷
Max 1.450401×10⁻⁴

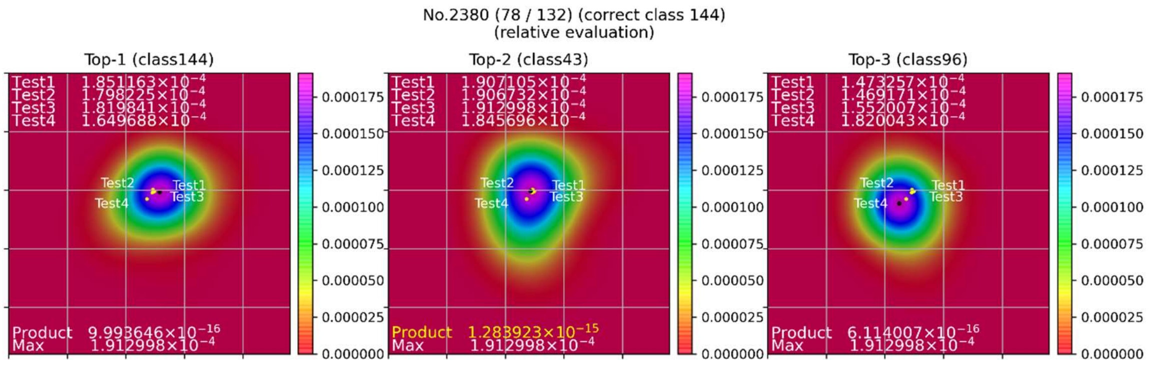

No.2094 (69 / 132) (correct class 127)
(absolute evaluation)
Top-1 (class127)
1e−5
Test1 6.690377×10⁻⁵
Test2 6.164199×10⁻⁵
Test3 7.874005×10⁻⁵
Test4 5.531239×10⁻⁵
Product 1.796161×10⁻¹⁷
Max 9.962027×10⁻⁵
Top-2 (class65)
Test1 8.973674×10⁻⁵
Test2 3.347715×10⁻⁵
Test3 7.466868×10⁻⁵
Test4 2.441177×10⁻⁵
Product 5.475913×10⁻¹⁸
Max 1.419137×10⁻⁴
Top-3 (class135)
1e−5
Test1 9.303873×10⁻⁶
Test2 2.608363×10⁻⁵
Test3 8.142216×10⁻⁵
Test4 2.672364×10⁻⁵
Product 5.280440×10⁻¹⁹
Max 8.551106×10⁻⁵

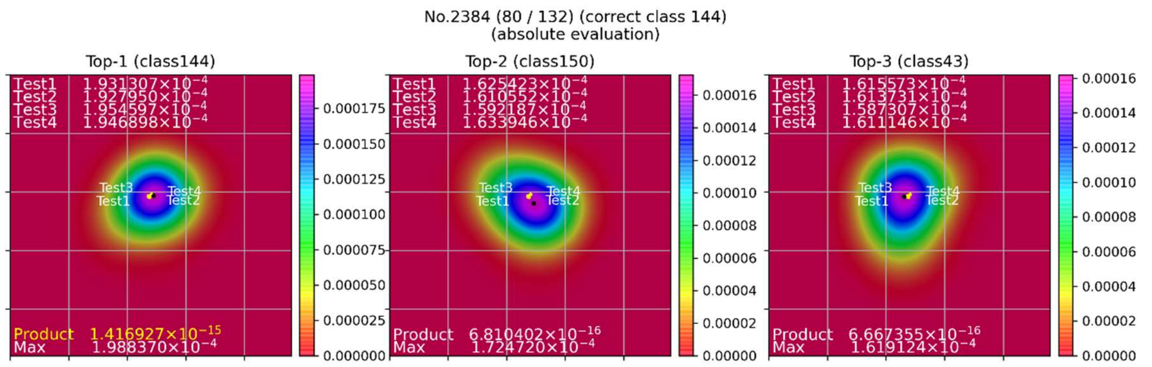

No.2094 (69 / 132) (correct class 127)
(relative evaluation)
Top-1 (class127)
Test1 9.530749×10⁻⁵
Test2 8.781185×10⁻⁵
Test3 1.121688×10⁻⁴
Test4 7.879504×10⁻⁵
Product 7.396924×10⁻¹⁷
Max 1.419137×10⁻⁴
Top-2 (class65)
Test1 8.973674×10⁻⁵
Test2 3.347715×10⁻⁵
Test3 7.466868×10⁻⁵
Test4 2.441177×10⁻⁵
Product 5.475913×10⁻¹⁸
Max 1.419137×10⁻⁴
Top-3 (class135)
Test1 1.544065×10⁻⁵
Test2 4.328825×10⁻⁵
Test3 1.351277×10⁻⁴
Test4 4.435040×10⁻⁵
Product 4.005693×10⁻¹⁸
Max 1.419137×10⁻⁴

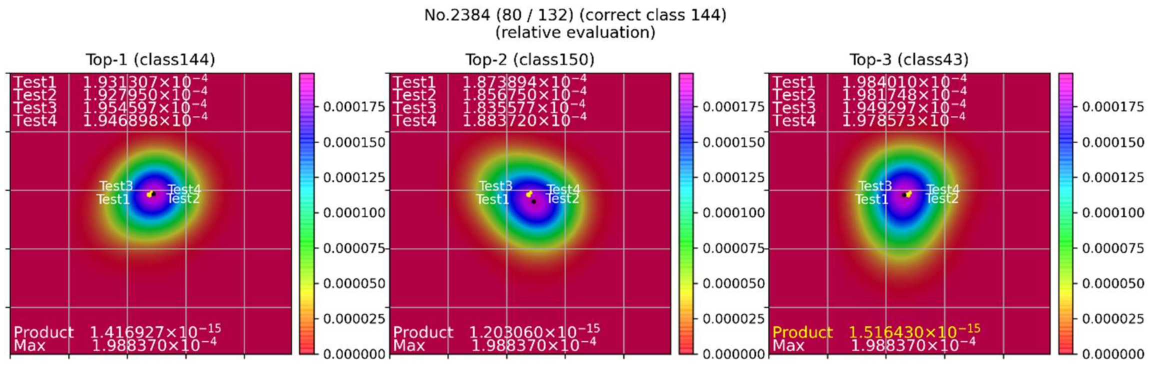

No.2095 (70 / 132) (correct class 127)
(absolute evaluation)
Top-1 (class65)
Test1 1.063577×10⁻⁴
Test2 1.598090×10⁻⁵
Test3 8.949973×10⁻⁵
Test4 9.822190×10⁻⁵
Product 1.494171×10⁻¹⁷
Max 1.424711×10⁻⁴
Top-2 (class127)
Test1 7.190067×10⁻⁵
Test2 3.813593×10⁻⁵
Test3 7.267775×10⁻⁵
Test4 7.030144×10⁻⁵
Product 1.400983×10⁻¹⁷
Max 9.971467×10⁻⁵
Top-3 (class92)
Test1 5.552377×10⁻⁵
Test2 1.939653×10⁻⁵
Test3 5.073020×10⁻⁵
Test4 5.428637×10⁻⁵
Product 2.965926×10⁻¹⁸
Max 7.205361×10⁻⁵

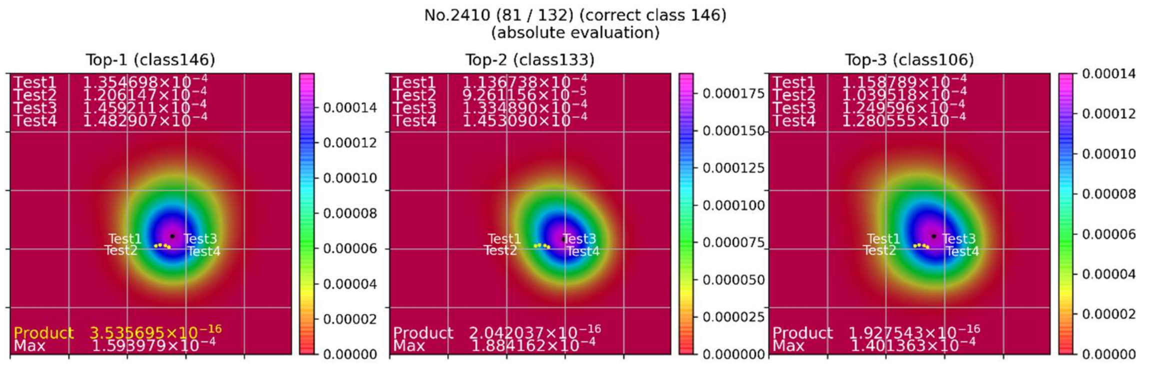

No.2095 (70 / 132) (correct class 127)
(relative evaluation)
Top-1 (class65)
Test1 1.063577×10⁻⁴
Test2 1.598090×10⁻⁵
Test3 8.949973×10⁻⁵
Test4 9.822190×10⁻⁵
Product 1.494171×10⁻¹⁷
Max 1.424711×10⁻⁴
Top-2 (class127)
Test1 1.027308×10⁻⁴
Test2 5.448816×10⁻⁵
Test3 1.038411×10⁻⁴
Test4 1.004459×10⁻⁴
Product 5.838540×10⁻¹⁷
Max 1.424711×10⁻⁴
Top-3 (class92)
Test1 1.097868×10⁻⁴
Test2 3.835263×10⁻⁵
Test3 1.003085×10⁻⁴
Test4 1.073401×10⁻⁴
Product 4.533618×10⁻¹⁷
Max 1.424711×10⁻⁴

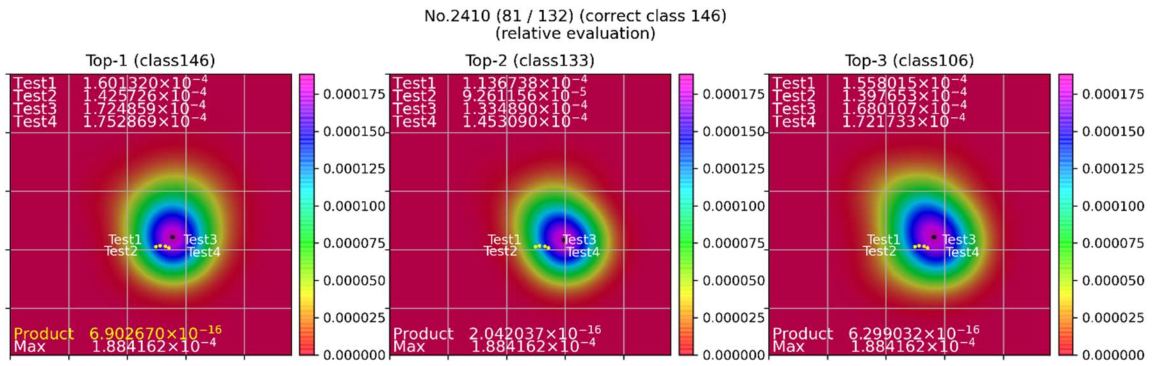

No.2380 (78 / 132) (correct class 144)
(absolute evaluation)
Top-1 (class144)
Test1 1.756588×10⁻⁴
Test2 1.706355×10⁻⁴
Test3 1.726867×10⁻⁴
Test4 1.565407×10⁻⁴
Product 8.102619×10⁻¹⁶
Max 1.815264×10⁻⁴
Top-2 (class43)
Test1 1.414577×10⁻⁴
Test2 1.414301×10⁻⁴
Test3 1.418949×10⁻⁴
Test4 1.369028×10⁻⁴
Product 3.886401×10⁻¹⁶
Max 1.418949×10⁻⁴
Top-3 (class96)
Test1 1.473257×10⁻⁴
Test2 1.469171×10⁻⁴
Test3 1.552007×10⁻⁴
Test4 1.820043×10⁻⁴
Product 6.114007×10⁻¹⁶
Max 1.912998×10⁻⁴

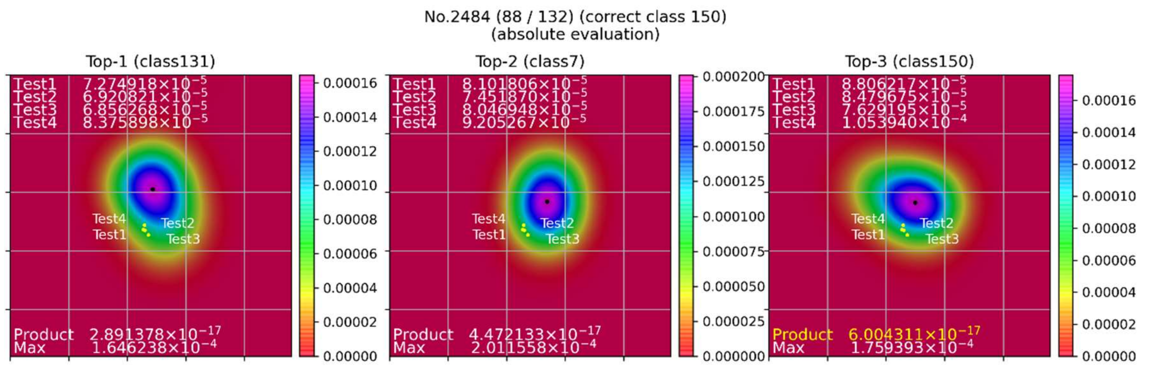

No.2380 (78 / 132) (correct class 144)
(relative evaluation)
Top-1 (class144)
Test1 1.851163×10⁻⁴
Test2 1.798225×10⁻⁴
Test3 1.819841×10⁻⁴
Test4 1.649688×10⁻⁴
Product 9.993646×10⁻¹⁶
Max 1.912998×10⁻⁴
Top-2 (class43)
Test1 1.907105×10⁻⁴
Test2 1.906732×10⁻⁴
Test3 1.912998×10⁻⁴
Test4 1.845696×10⁻⁴
Product 1.283923×10⁻¹⁵
Max 1.912998×10⁻⁴
Top-3 (class96)
Test1 1.473257×10⁻⁴
Test2 1.469171×10⁻⁴
Test3 1.552007×10⁻⁴
Test4 1.820043×10⁻⁴
Product 6.114007×10⁻¹⁶
Max 1.912998×10⁻⁴

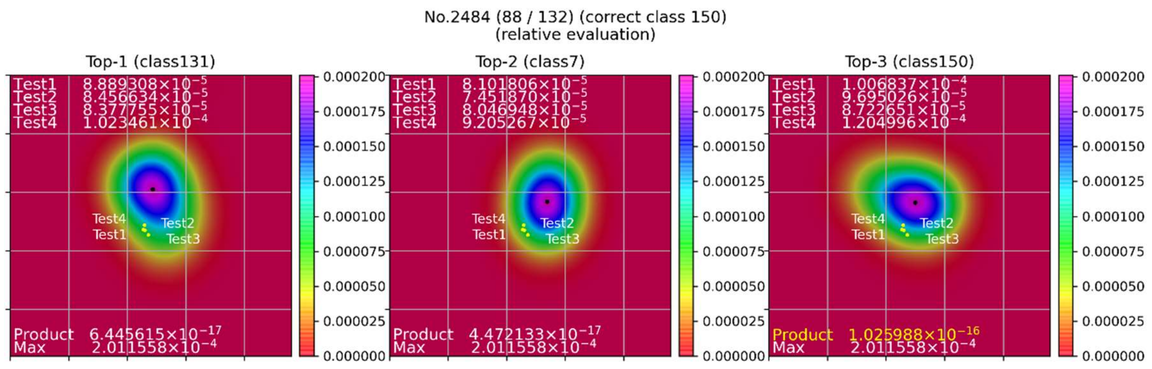

No.2384 (80 / 132) (correct class 144)
(absolute evaluation)
Top-1 (class144)
Test1 1.931307×10⁻⁴
Test2 1.927950×10⁻⁴
Test3 1.954597×10⁻⁴
Test4 1.946898×10⁻⁴
Product 1.416927×10⁻¹⁵
Max 1.988370×10⁻⁴
Top-2 (class150)
Test1 1.625423×10⁻⁴
Test2 1.610552×10⁻⁴
Test3 1.592187×10⁻⁴
Test4 1.633946×10⁻⁴
Product 6.810402×10⁻¹⁶
Max 1.724720×10⁻⁴
Top-3 (class43)
Test1 1.615573×10⁻⁴
Test2 1.613731×10⁻⁴
Test3 1.587307×10⁻⁴
Test4 1.611146×10⁻⁴
Product 6.667355×10⁻¹⁶
Max 1.619124×10⁻⁴

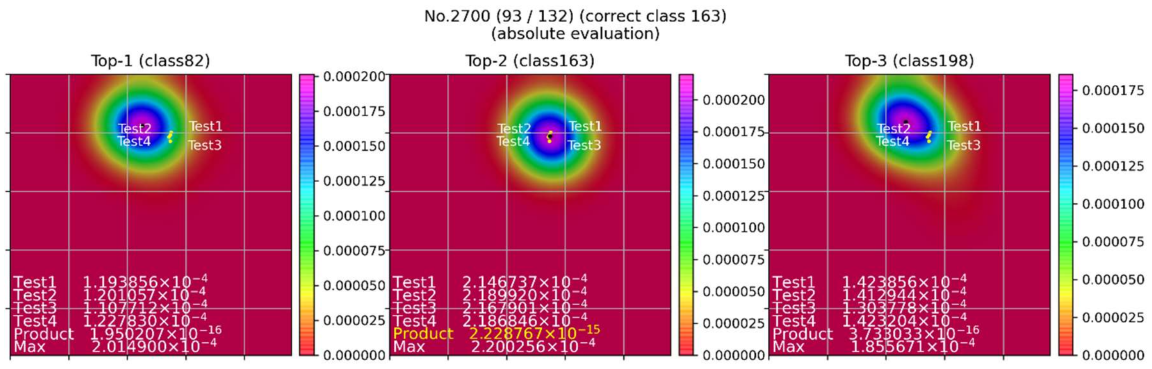

No.2384 (80 / 132) (correct class 144)
(relative evaluation)
Top-1 (class144)
Test1 1.931307×10⁻⁴
Test2 1.927950×10⁻⁴
Test3 1.954597×10⁻⁴
Test4 1.946898×10⁻⁴
Product 1.416927×10⁻¹⁵
Max 1.988370×10⁻⁴
Top-2 (class150)
Test1 1.873894×10⁻⁴
Test2 1.856750×10⁻⁴
Test3 1.835577×10⁻⁴
Test4 1.883720×10⁻⁴
Product 1.203060×10⁻¹⁵
Max 1.988370×10⁻⁴
Top-3 (class43)
Test1 1.984010×10⁻⁴
Test2 1.981748×10⁻⁴
Test3 1.949297×10⁻⁴
Test4 1.978573×10⁻⁴
Product 1.516430×10⁻¹⁵
Max 1.988370×10⁻⁴

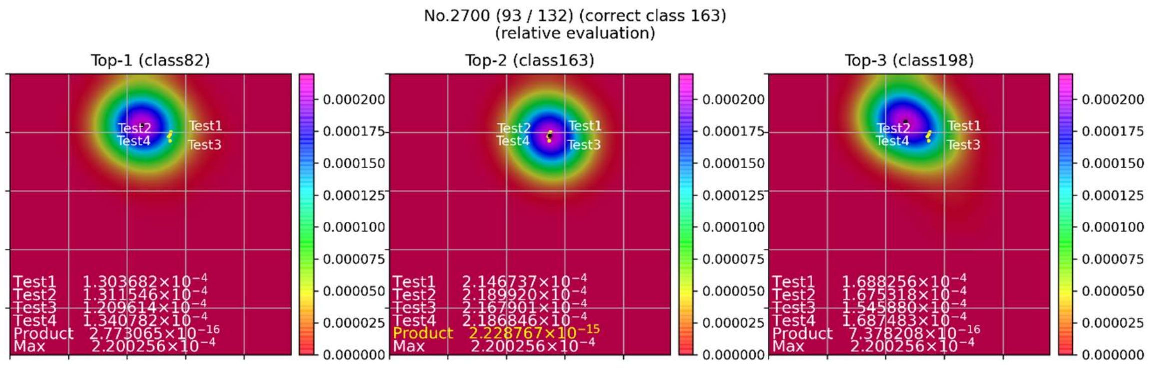

No.2410 (81 / 132) (correct class 146)
(absolute evaluation)
Top-1 (class146)
Test1 1.354698×10⁻⁴
Test2 1.206147×10⁻⁴
Test3 1.459211×10⁻⁴
Test4 1.482907×10⁻⁴
Product 3.535695×10⁻¹⁶
Max 1.593979×10⁻⁴
Top-2 (class133)
Test1 1.136738×10⁻⁴
Test2 9.261156×10⁻⁵
Test3 1.334890×10⁻⁴
Test4 1.453090×10⁻⁴
Product 2.042037×10⁻¹⁶
Max 1.884162×10⁻⁴
Top-3 (class106)
Test1 1.158789×10⁻⁴
Test2 1.039518×10⁻⁴
Test3 1.249596×10⁻⁴
Test4 1.280555×10⁻⁴
Product 1.927543×10⁻¹⁶
Max 1.401363×10⁻⁴

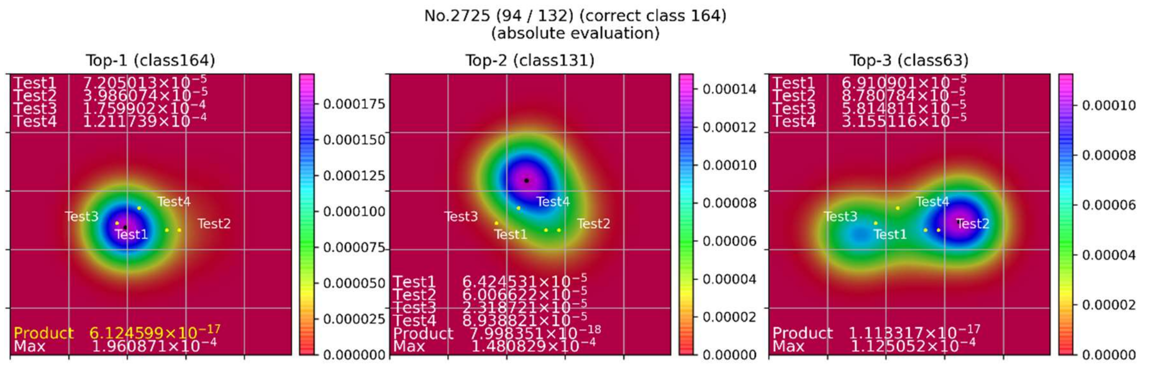

No.2410 (81 / 132) (correct class 146)
(relative evaluation)
Top-1 (class146)
Test1 1.601320×10⁻⁴
Test2 1.425726×10⁻⁴
Test3 1.724859×10⁻⁴
Test4 1.752869×10⁻⁴
Product 6.902670×10⁻¹⁶
Max 1.884162×10⁻⁴
Top-2 (class133)
Test1 1.136738×10⁻⁴
Test2 9.261156×10⁻⁵
Test3 1.334890×10⁻⁴
Test4 1.453090×10⁻⁴
Product 2.042037×10⁻¹⁶
Max 1.884162×10⁻⁴
Top-3 (class106)
Test1 1.558015×10⁻⁴
Test2 1.397653×10⁻⁴
Test3 1.680107×10⁻⁴
Test4 1.721733×10⁻⁴
Product 6.299032×10⁻¹⁶
Max 1.884162×10⁻⁴

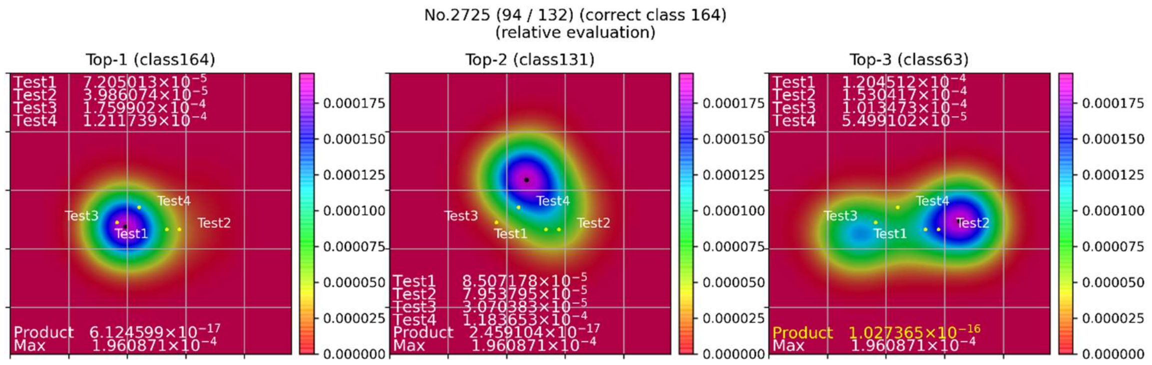

No.2484 (88 / 132) (correct class 150)
(absolute evaluation)
Top-1 (class131)
Test1 7.274918×10⁻⁵
Test2 6.920821×10⁻⁵
Test3 6.856268×10⁻⁵
Test4 8.375898×10⁻⁵
Product 2.891378×10⁻¹⁷
Max 1.646238×10⁻⁴
Top-2 (class7)
Test1 8.101806×10⁻⁵
Test2 7.451870×10⁻⁵
Test3 8.046948×10⁻⁵
Test4 9.205267×10⁻⁵
Product 4.472133×10⁻¹⁷
Max 2.011558×10⁻⁴
Top-3 (class150)
Test1 8.806217×10⁻⁵
Test2 8.479675×10⁻⁵
Test3 7.629195×10⁻⁵
Test4 1.053940×10⁻⁴
Product 6.004311×10⁻¹⁷
Max 1.759393×10⁻⁴

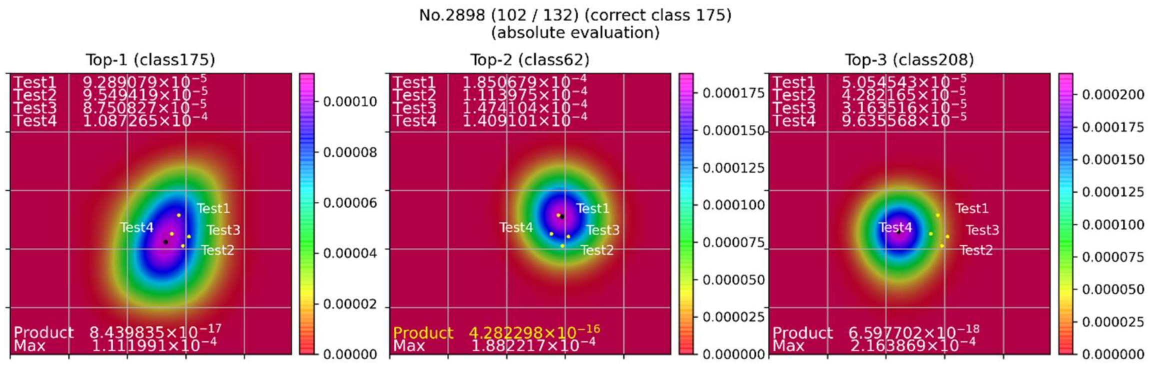

No.2484 (88 / 132) (correct class 150)
(relative evaluation)
Top-1 (class131)
Test1 8.889308×10⁻⁵
Test2 8.456634×10⁻⁵
Test3 8.377755×10⁻⁵
Test4 1.023461×10⁻⁴
Product 6.445615×10⁻¹⁷
Max 2.011558×10⁻⁴
Top-2 (class7)
Test1 8.101806×10⁻⁵
Test2 7.451870×10⁻⁵
Test3 8.046948×10⁻⁵
Test4 9.205267×10⁻⁵
Product 4.472133×10⁻¹⁷
Max 2.011558×10⁻⁴
Top-3 (class150)
Test1 1.006837×10⁻⁴
Test2 9.695026×10⁻⁵
Test3 8.722651×10⁻⁵
Test4 1.204996×10⁻⁴
Product 1.025988×10⁻¹⁶
Max 2.011558×10⁻⁴

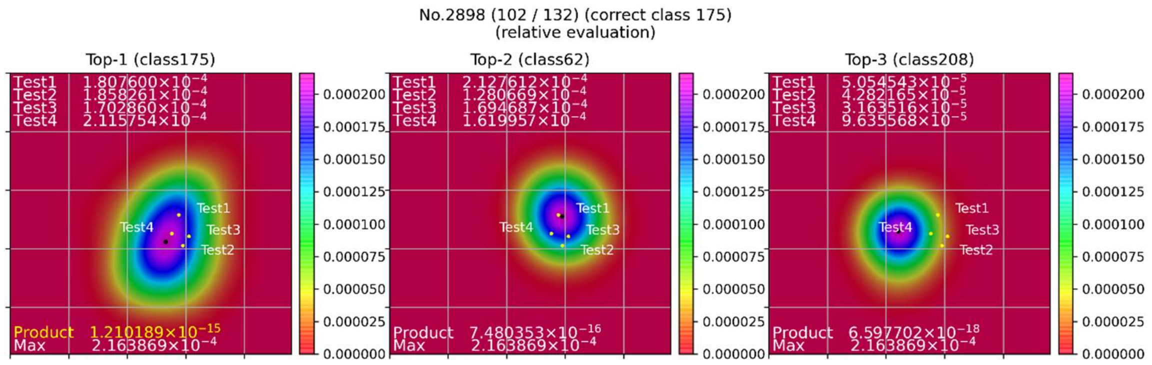

No.2700 (93 / 132) (correct class 163)
(absolute evaluation)
Top-1 (class82)
Test1 1.193856×10⁻⁴
Test2 1.201057×10⁻⁴
Test3 1.107712×10⁻⁴
Test4 1.227830×10⁻⁴
Product 1.950207×10⁻¹⁶
Max 2.014900×10⁻⁴
Top-2 (class163)
Test1 2.146737×10⁻⁴
Test2 2.189920×10⁻⁴
Test3 2.167901×10⁻⁴
Test4 2.186846×10⁻⁴
Product 2.228767×10⁻¹⁵
Max 2.200256×10⁻⁴
Top-3 (class198)
Test1 1.423856×10⁻⁴
Test2 1.412944×10⁻⁴
Test3 1.303778×10⁻⁴
Test4 1.423204×10⁻⁴
Product 3.733033×10⁻¹⁶
Max 1.855671×10⁻⁴

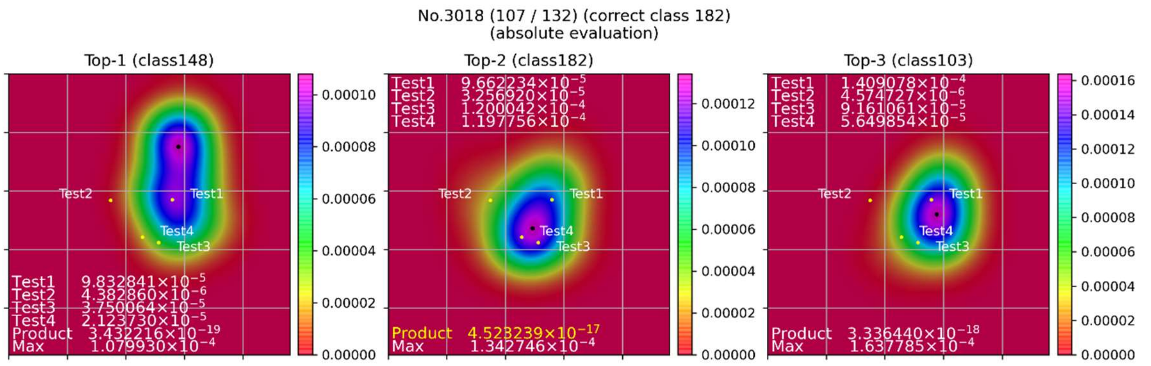

No.2700 (93 / 132) (correct class 163)
(relative evaluation)
Top-1 (class82)
Test1 1.303682×10⁻⁴
Test2 1.311546×10⁻⁴
Test3 1.209614×10⁻⁴
Test4 1.340782×10⁻⁴
Product 2.773065×10⁻¹⁶
Max 2.200256×10⁻⁴
Top-2 (class163)
Test1 2.146737×10⁻⁴
Test2 2.189920×10⁻⁴
Test3 2.167901×10⁻⁴
Test4 2.186846×10⁻⁴
Product 2.228767×10⁻¹⁵
Max 2.200256×10⁻⁴
Top-3 (class198)
Test1 1.688256×10⁻⁴
Test2 1.675318×10⁻⁴
Test3 1.545880×10⁻⁴
Test4 1.687483×10⁻⁴
Product 7.378208×10⁻¹⁶
Max 2.200256×10⁻⁴

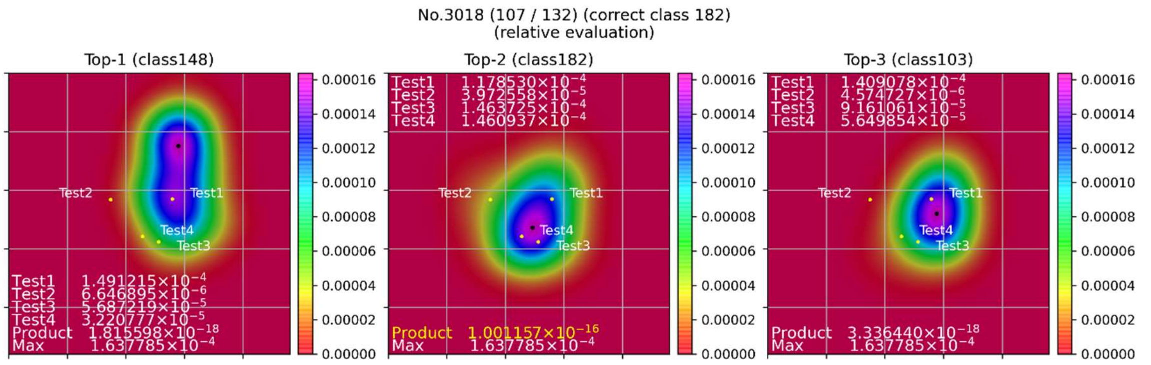

No.2725 (94 / 132) (correct class 164)
(absolute evaluation)
Top-1 (class164)
Test1 7.205013×10⁻⁵
Test2 3.986074×10⁻⁵
Test3 1.759902×10⁻⁴
Test4 1.211739×10⁻⁴
Product 6.124599×10⁻¹⁷
Max 1.960871×10⁻⁴
Top-2 (class131)
Test1 6.424531×10⁻⁵
Test2 6.006622×10⁻⁵
Test3 2.318721×10⁻⁵
Test4 8.938821×10⁻⁵
Product 7.998351×10⁻¹⁸
Max 1.480829×10⁻⁴
Top-3 (class63)
Test1 6.910901×10⁻⁵
Test2 8.780784×10⁻⁵
Test3 5.814811×10⁻⁵
Test4 3.155116×10⁻⁵
Product 1.113317×10⁻¹⁷
Max 1.125052×10⁻⁴

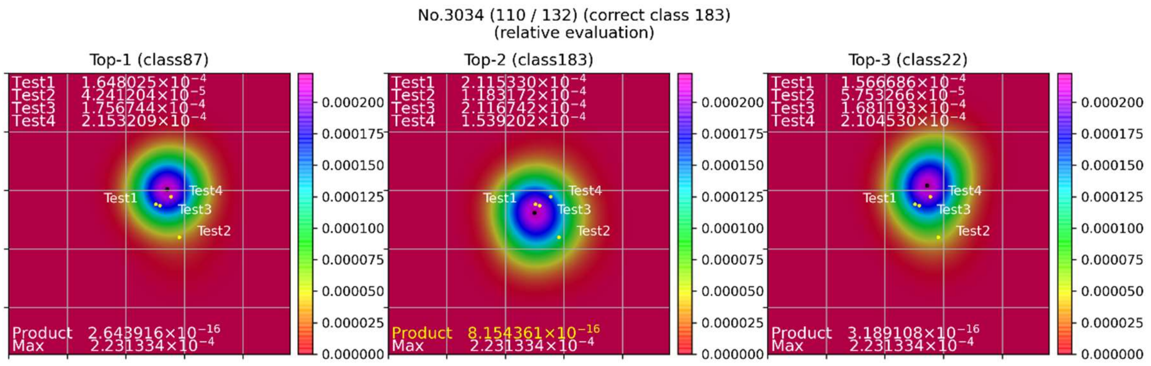

No.2725 (94 / 132) (correct class 164)
(relative evaluation)
Top-1 (class164)
Test1 7.205013×10⁻⁵
Test2 3.986074×10⁻⁵
Test3 1.759902×10⁻⁴
Test4 1.211739×10⁻⁴
Product 6.124599×10⁻¹⁷
Max 1.960871×10⁻⁴
Top-2 (class131)
Test1 8.507178×10⁻⁵
Test2 7.953795×10⁻⁵
Test3 3.070383×10⁻⁵
Test4 1.183653×10⁻⁴
Product 2.459104×10⁻¹⁷
Max 1.960871×10⁻⁴
Top-3 (class63)
Test1 1.204512×10⁻⁴
Test2 1.530417×10⁻⁴
Test3 1.013473×10⁻⁴
Test4 5.499102×10⁻⁵
Product 1.027365×10⁻¹⁶
Max 1.960871×10⁻⁴

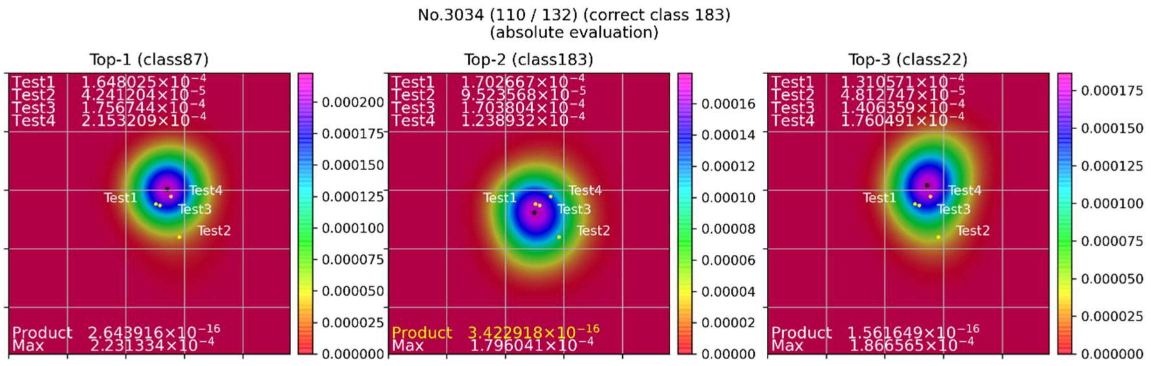

No.2898 (102 / 132) (correct class 175)
(absolute evaluation)
Top-1 (class175)
Test1 9.289079×10⁻⁵
Test2 9.549419×10⁻⁵
Test3 8.750827×10⁻⁵
Test4 1.087265×10⁻⁴
Product 8.439835×10⁻¹⁷
Max 1.111991×10⁻⁴
Top-2 (class62)
Test1 1.850679×10⁻⁴
Test2 1.113975×10⁻⁴
Test3 1.474104×10⁻⁴
Test4 1.409101×10⁻⁴
Product 4.282298×10⁻¹⁶
Max 1.882217×10⁻⁴
Top-3 (class208)
Test1 5.054543×10⁻⁵
Test2 4.282165×10⁻⁵
Test3 3.163516×10⁻⁵
Test4 9.635568×10⁻⁵
Product 6.597702×10⁻¹⁸
Max 2.163869×10⁻⁴

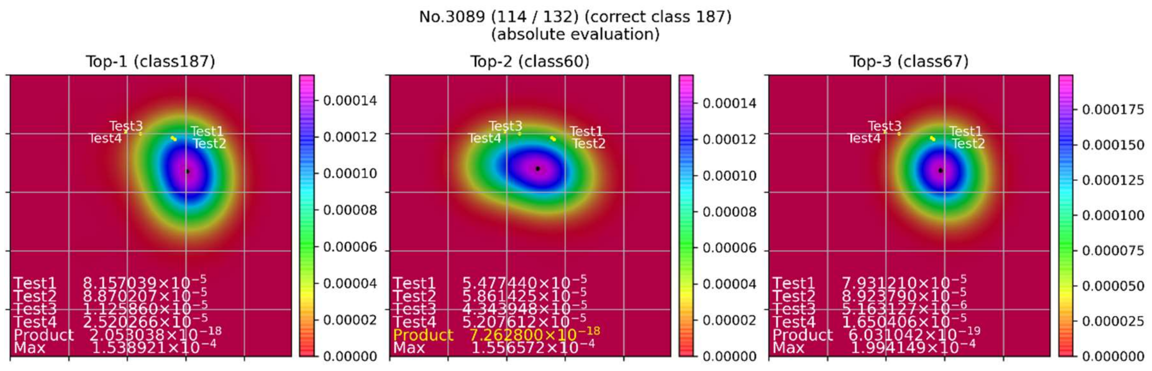

No.2898 (102 / 132) (correct class 175)
(relative evaluation)
Top-1 (class175)
Test1 1.807600×10⁻⁴
Test2 1.858261×10⁻⁴
Test3 1.702860×10⁻⁴
Test4 2.115754×10⁻⁴
Product 1.210189×10⁻¹⁵
Max 2.163869×10⁻⁴
Top-2 (class62)
Test1 2.127612×10⁻⁴
Test2 1.280669×10⁻⁴
Test3 1.694687×10⁻⁴
Test4 1.619957×10⁻⁴
Product 7.480353×10⁻¹⁶
Max 2.163869×10⁻⁴
Top-3 (class208)
Test1 5.054543×10⁻⁵
Test2 4.282165×10⁻⁵
Test3 3.163516×10⁻⁵
Test4 9.635568×10⁻⁵
Product 6.597702×10⁻¹⁸
Max 2.163869×10⁻⁴

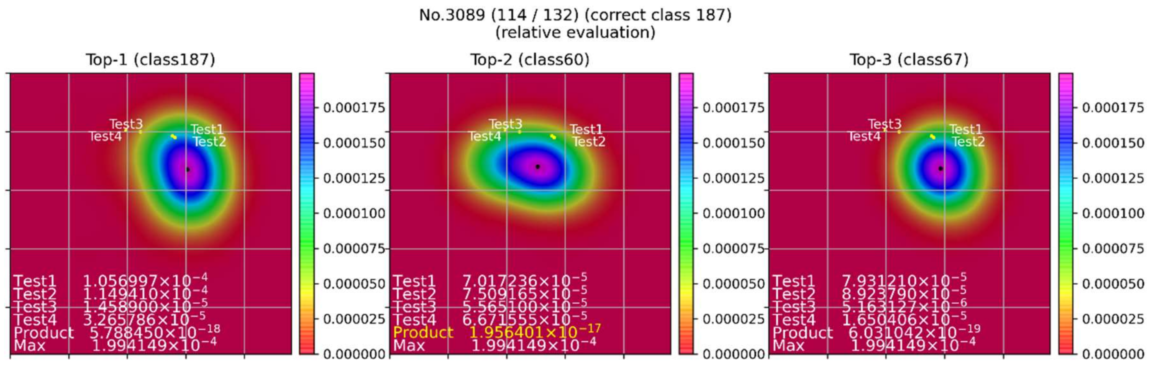

No.3018 (107 / 132) (correct class 182)
(absolute evaluation)
Top-1 (class148)
Test2
Test1
Test4
Test3
Test1 9.832841×10^-5
Test2 4.382860×10^-6
Test3 3.750064×10^-5
Test4 2.123730×10^-5
Product 3.432216×10^-19
Max 1.079930×10^-4
Top-2 (class182)
Test1 9.662234×10^-5
Test2 3.256920×10^-5
Test3 1.200042×10^-4
Test4 1.197756×10^-4
Product 4.523239×10^-17
Max 1.342746×10^-4
Top-3 (class103)
Test1 1.409078×10^-4
Test2 4.574727×10^-6
Test3 9.161061×10^-5
Test4 5.649854×10^-5
Product 3.336440×10^-18
Max 1.637785×10^-4

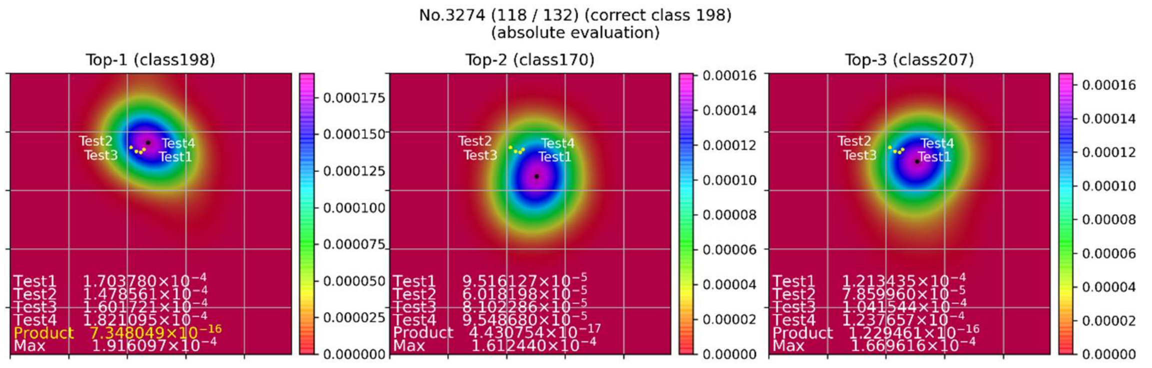

No.3018 (107 / 132) (correct class 182)
(relative evaluation)
Top-1 (class148)
Test1 1.491215×10^-4
Test2 6.646895×10^-6
Test3 5.687219×10^-5
Test4 3.220777×10^-5
Product 1.815598×10^-18
Max 1.637785×10^-4
Top-2 (class182)
Test1 1.178530×10^-4
Test2 3.972558×10^-5
Test3 1.463725×10^-4
Test4 1.460937×10^-4
Product 1.001157×10^-16
Max 1.637785×10^-4
Top-3 (class103)
Test1 1.409078×10^-4
Test2 4.574727×10^-6
Test3 9.161061×10^-5
Test4 5.649854×10^-5
Product 3.336440×10^-18
Max 1.637785×10^-4

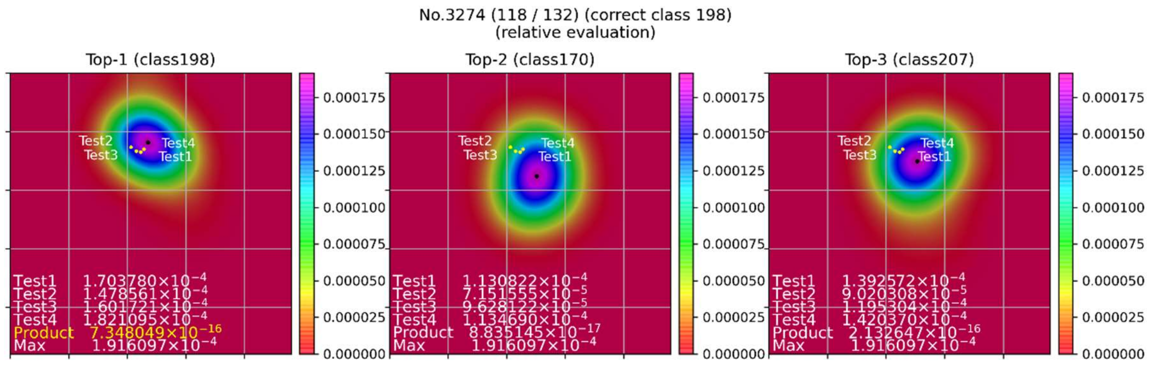

No.3034 (110 / 132) (correct class 183)
(relative evaluation)
Top-1 (class87)
Test1 1.648025×10^-4
Test2 4.241204×10^-5
Test3 1.756744×10^-4
Test4 2.153209×10^-4
Product 2.643916×10^-16
Max 2.231334×10^-4
Top-2 (class183)
Test1 2.115330×10^-4
Test2 1.183172×10^-4
Test3 2.116742×10^-4
Test4 1.539202×10^-4
Product 8.154361×10^-16
Max 2.231334×10^-4
Top-3 (class22)
Test1 1.566686×10^-4
Test2 5.753266×10^-5
Test3 1.681193×10^-4
Test4 2.104530×10^-4
Product 3.189108×10^-16
Max 2.231334×10^-4

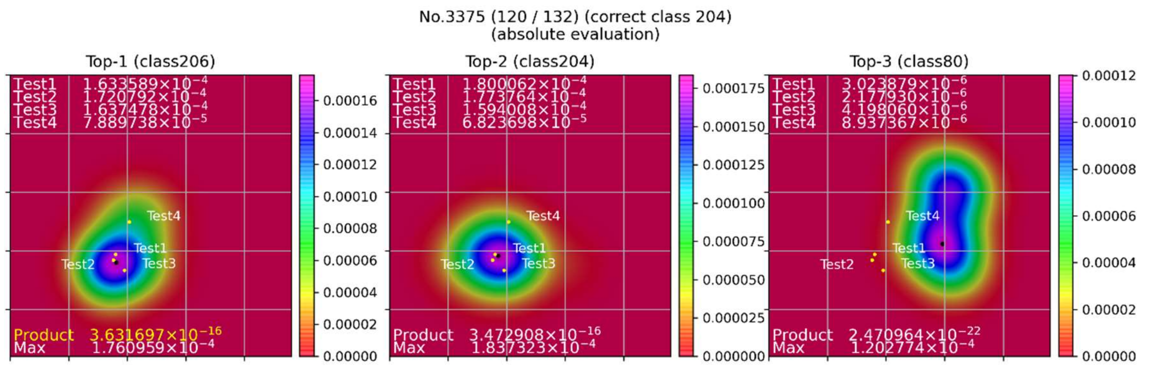

No.3034 (110 / 132) (correct class 183)
(absolute evaluation)
Top-1 (class87)
Test1 1.648025×10^-4
Test2 4.241204×10^-5
Test3 1.756744×10^-4
Test4 2.153209×10^-4
Product 2.643916×10^-16
Max 2.231334×10^-4
Top-2 (class183)
Test1 1.702667×10^-4
Test2 9.523568×10^-5
Test3 1.703804×10^-4
Test4 1.238932×10^-4
Product 3.422918×10^-16
Max 1.796041×10^-4
Top-3 (class22)
Test1 1.310571×10^-4
Test2 4.812747×10^-5
Test3 1.406359×10^-4
Test4 1.760491×10^-4
Product 1.561649×10^-16
Max 1.866565×10^-4

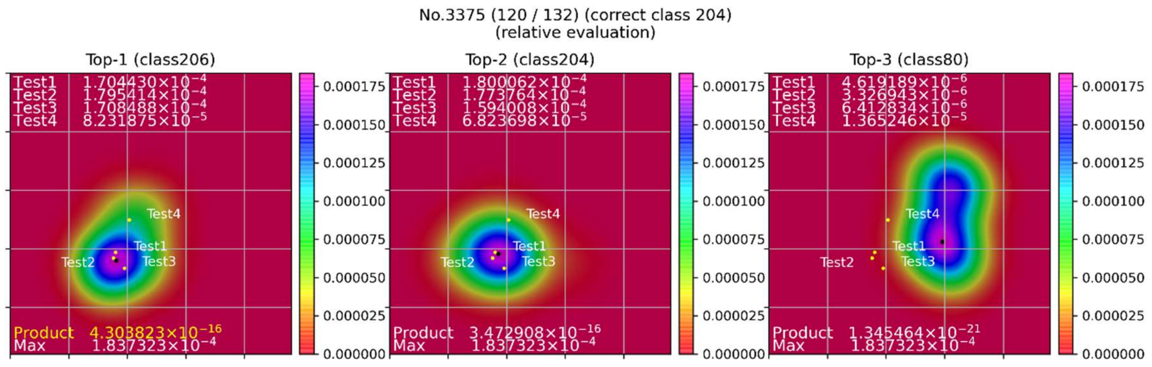

No.3089 (114 / 132) (correct class 187)
(absolute evaluation)
Top-1 (class187)
Test3
Test4
Test1
Test2
Test1 8.157039×10^-5
Test2 8.870207×10^-5
Test3 1.125860×10^-5
Test4 2.520266×10^-5
Product 2.053038×10^-18
Max 1.538921×10^-4
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
Top-2 (class60)
Test3
Test4
Test1
Test2
Test1 5.477440×10^-5
Test2 5.861425×10^-5
Test3 4.343948×10^-5
Test4 5.207612×10^-5
Product 7.262800×10^-18
Max 1.556572×10^-4
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
Top-3 (class67)
Test3
Test4
Test1
Test2
Test1 7.931210×10^-5
Test2 8.923790×10^-5
Test3 5.163127×10^-6
Test4 1.650406×10^-5
Product 6.031042×10^-19
Max 1.994149×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000

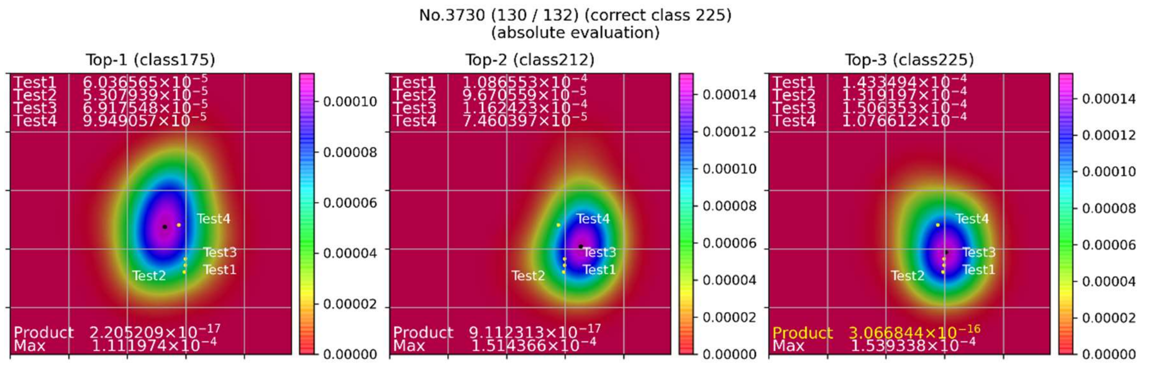

No.3089 (114 / 132) (correct class 187)
(relative evaluation)
Top-1 (class187)
Test3
Test4
Test1
Test2
Test1 1.056997×10^-4
Test2 1.149410×10^-4
Test3 1.458900×10^-5
Test4 3.265786×10^-5
Product 5.788450×10^-18
Max 1.994149×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
Top-2 (class60)
Test3
Test4
Test1
Test2
Test1 7.017236×10^-5
Test2 7.509165×10^-5
Test3 5.565100×10^-5
Test4 6.671555×10^-5
Product 1.956401×10^-17
Max 1.994149×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
Top-3 (class67)
Test3
Test4
Test1
Test2
Test1 7.931210×10^-5
Test2 8.923790×10^-5
Test3 5.163127×10^-6
Test4 1.650406×10^-5
Product 6.031042×10^-19
Max 1.994149×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000

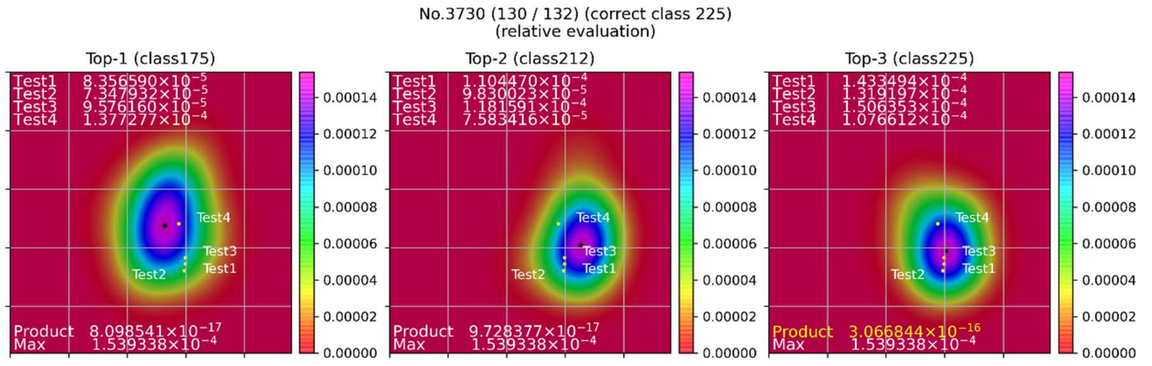

No.3274 (118 / 132) (correct class 198)
(absolute evaluation)
Top-1 (class198)
Test2
Test3
Test4
Test1
Test1 1.703780×10^-4
Test2 1.478561×10^-4
Test3 1.601721×10^-4
Test4 1.821095×10^-4
Product 7.348049×10^-16
Max 1.916097×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
Top-2 (class170)
Test2
Test3
Test4
Test1
Test1 9.516127×10^-5
Test2 6.018198×10^-5
Test3 8.102286×10^-5
Test4 9.548680×10^-5
Product 4.430754×10^-17
Max 1.612440×10^-4
0.00016
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
Top-3 (class207)
Test2
Test3
Test4
Test1
Test1 1.213435×10^-4
Test2 7.859960×10^-5
Test3 1.041544×10^-4
Test4 1.237657×10^-4
Product 1.229461×10^-16
Max 1.669616×10^-4
0.00016
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000

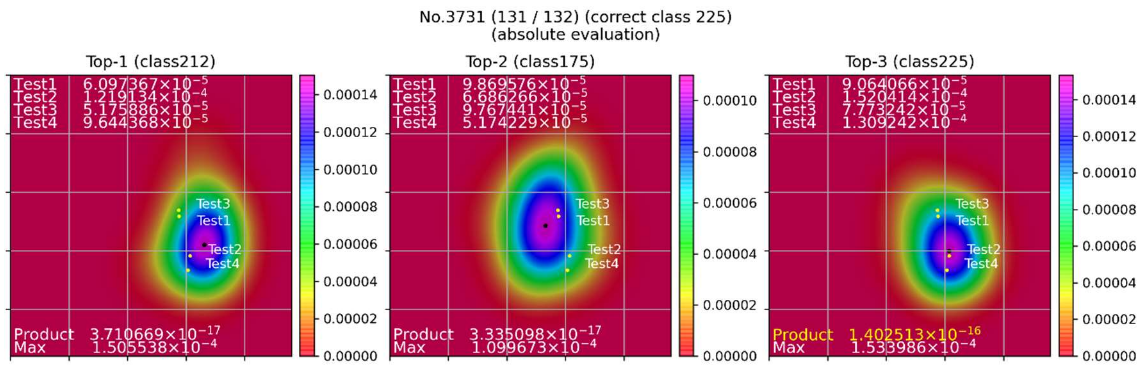

No.3274 (118 / 132) (correct class 198)
(relative evaluation)
Top-1 (class198)
Test2
Test3
Test4
Test1
Test1 1.703780×10^-4
Test2 1.478561×10^-4
Test3 1.601721×10^-4
Test4 1.821095×10^-4
Product 7.348049×10^-16
Max 1.916097×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
Top-2 (class170)
Test2
Test3
Test4
Test1
Test1 1.130822×10^-4
Test2 7.151555×10^-5
Test3 9.628122×10^-5
Test4 1.134690×10^-4
Product 8.835145×10^-17
Max 1.916097×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
Top-3 (class207)
Test2
Test3
Test4
Test1
Test1 1.392572×10^-4
Test2 9.020308×10^-5
Test3 1.195304×10^-4
Test4 1.420370×10^-4
Product 2.132647×10^-16
Max 1.916097×10^-4
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000

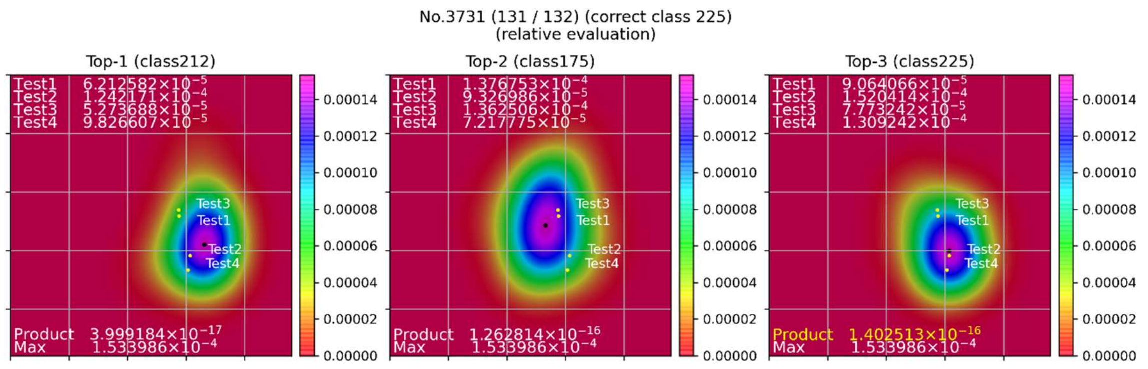

No.3375 (120 / 132) (correct class 204)
(absolute evaluation)
Top-1 (class206)
Test1 1.633589×10⁻⁴
Test2 1.720792×10⁻⁴
Test3 1.637478×10⁻⁴
Test4 7.889738×10⁻⁵
Product 3.631697×10⁻¹⁶
Max 1.760959×10⁻⁴
Top-2 (class204)
Test1 1.800062×10⁻⁴
Test2 1.773764×10⁻⁴
Test3 1.594008×10⁻⁴
Test4 6.823698×10⁻⁵
Product 3.472908×10⁻¹⁶
Max 1.837323×10⁻⁴
Top-3 (class80)
Test1 3.023879×10⁻⁶
Test2 2.177930×10⁻⁶
Test3 4.198060×10⁻⁶
Test4 8.937367×10⁻⁶
Product 2.470964×10⁻²²
Max 1.202774×10⁻⁴


No.3375 (120 / 132) (correct class 204)
(relative evaluation)
Top-1 (class206)
Test1 1.704430×10⁻⁴
Test2 1.795414×10⁻⁴
Test3 1.708488×10⁻⁴
Test4 8.231875×10⁻⁵
Product 4.303823×10⁻¹⁶
Max 1.837323×10⁻⁴
Top-2 (class204)
Test1 1.800062×10⁻⁴
Test2 1.773764×10⁻⁴
Test3 1.594008×10⁻⁴
Test4 6.823698×10⁻⁵
Product 3.472908×10⁻¹⁶
Max 1.837323×10⁻⁴
Top-3 (class80)
Test1 4.619189×10⁻⁶
Test2 3.326943×10⁻⁶
Test3 6.412834×10⁻⁶
Test4 1.365246×10⁻⁵
Product 1.345464×10⁻²¹
Max 1.837323×10⁻⁴


No.3730 (130 / 132) (correct class 225)
(absolute evaluation)
Top-1 (class175)
Test1 6.036565×10⁻⁵
Test2 5.307939×10⁻⁵
Test3 6.917548×10⁻⁵
Test4 9.949057×10⁻⁵
Product 2.205209×10⁻¹⁷
Max 1.111974×10⁻⁴
Top-2 (class212)
Test1 1.086553×10⁻⁴
Test2 9.670559×10⁻⁵
Test3 1.162423×10⁻⁴
Test4 7.460397×10⁻⁵
Product 9.112313×10⁻¹⁷
Max 1.514366×10⁻⁴
Top-3 (class225)
Test1 1.433494×10⁻⁴
Test2 1.319197×10⁻⁴
Test3 1.506353×10⁻⁴
Test4 1.076612×10⁻⁴
Product 3.066844×10⁻¹⁶
Max 1.539338×10⁻⁴


No.3730 (130 / 132) (correct class 225)
(relative evaluation)
Top-1 (class175)
Test1 8.356590×10⁻⁵
Test2 7.347932×10⁻⁵
Test3 9.576160×10⁻⁵
Test4 1.377277×10⁻⁴
Product 8.098541×10⁻¹⁷
Max 1.539338×10⁻⁴
Top-2 (class212)
Test1 1.104470×10⁻⁴
Test2 9.830023×10⁻⁵
Test3 1.181591×10⁻⁴
Test4 7.583416×10⁻⁵
Product 9.728377×10⁻¹⁷
Max 1.539338×10⁻⁴
Top-3 (class225)
Test1 1.433494×10⁻⁴
Test2 1.319197×10⁻⁴
Test3 1.506353×10⁻⁴
Test4 1.076612×10⁻⁴
Product 3.066844×10⁻¹⁶
Max 1.539338×10⁻⁴


No.3731 (131 / 132) (correct class 225)
(absolute evaluation)
Top-1 (class212)
Test1 6.097367×10⁻⁵
Test2 1.219134×10⁻⁴
Test3 5.175886×10⁻⁵
Test4 9.644368×10⁻⁵
Product 3.710669×10⁻¹⁷
Max 1.505538×10⁻⁴
Top-2 (class175)
Test1 9.869576×10⁻⁵
Test2 6.686266×10⁻⁵
Test3 9.767441×10⁻⁵
Test4 5.174229×10⁻⁵
Product 3.335098×10⁻¹⁷
Max 1.099673×10⁻⁴
Top-3 (class225)
Test1 9.064066×10⁻⁵
Test2 1.520414×10⁻⁴
Test3 7.773242×10⁻⁵
Test4 1.309242×10⁻⁴
Product 1.402513×10⁻¹⁶
Max 1.533986×10⁻⁴


No.3731 (131 / 132) (correct class 225)
(relative evaluation)
Top-1 (class212)
Test1 6.212582×10⁻⁵
Test2 1.242171×10⁻⁴
Test3 5.273688×10⁻⁵
Test4 9.826607×10⁻⁵
Product 3.999184×10⁻¹⁷
Max 1.533986×10⁻⁴
Top-2 (class175)
Test1 1.376753×10⁻⁴
Test2 9.326986×10⁻⁵
Test3 1.362506×10⁻⁴
Test4 7.217775×10⁻⁵
Product 1.262814×10⁻¹⁶
Max 1.533986×10⁻⁴
Top-3 (class225)
Test1 9.064066×10⁻⁵
Test2 1.520414×10⁻⁴
Test3 7.773242×10⁻⁵
Test4 1.309242×10⁻⁴
Product 1.402513×10⁻¹⁶
Max 1.533986×10⁻⁴

Acknowledgements

I extend my heartfelt gratitude to Professor Masahito Yamamoto for his unwavering support and guidance throughout my research journey. His mentorship during my doctoral course has been invaluable, providing me with opportunities to contribute to journals and present at international conferences. I am also thankful for the insightful guidance from Prof. Itsuki Noda, Prof. Tetsuo Ono, and Prof. Hidenori Kawamura, my secondary supervisors.

I extend my sincere thanks to Mr. Koshida for his proficient handling of various procedures, including the procurement of computer equipment and coordination of travel for international conferences. Additionally, I express my appreciation to Ms. Jin, my laboratory assistant, for her invaluable assistance in submitting documents and managing administrative correspondence throughout my student life.

Through the collective efforts of everyone involved, this research has successfully culminated in a comprehensive paper. I express my heartfelt thanks to each one of you for your unwavering support and collaboration, which played a pivotal role in bringing this project to fruition.

List of Publication of the Author

Journal Papers

[1] N. Hori and M. Yamamoto, "Re-Evaluation Method by Index Finger Position in the Face Area Using Face Part Position Criterion for Sign Language Recognition," Sensors, vol. 23, p. 4321, 2023

Reviewed Conference Papers

[1] N. Hori and M. Yamamoto, "Sign Language Recognition using the reuse of estimate results by each epoch," in Proceedings of the 7th International Conference on Frontiers of Signal Processing (ICFSP), Paris, France, 7-9 September 2022; pp. 45-50, doi: 10.1109/ICFSP55781.2022.9924938.

[2] N. Hori and M. Yamamoto, "Real-time Isolated sign language recognition," in Proceedings of the 1st International Conference on Frontiers of Artificial Intelligence, Ethics, and Multidisciplinary Applications (FAIEMA), Athens, Greece, 25-26 September 2023, in printing.

Reviewed Conference Papers (others)

[1] N. Hori and K. Hara, "Real Time Finger Typing Recognition on iPhone's RGB Camera," in Proceedings of the 4th International Conference on Frontiers of Signal Processing (ICFSP), Paris, France, 7-9 September 2018; pp. 125-129.