DISCO – U-Net based Autoencoder Architecture with Dual Input Streams for Skeleton Image Drawing

Soonyong Song, Heechul Bae, and Junhee Park
Industry and IoT Intelligence Research Department
Electronics and Telecommunications Research Institute (ETRI)
Daejeon, South Korea
{soony, hessed, juni}@etri.re.kr

Abstract

In this paper, we propose a DISCO, which is a manner of designing autoencoder architecture to process dual input streams for skeletal image generation. The DISCO was designed to be dealing with binary masks and skeletonized images concurrently at the input side. We expected the skeletonized images using traditional thinning algorithms could help to boost skeleton prediction performances. Inside the DISCO architecture, there exist two encoders and a single decoder. Each functional block is stacked with multiple logical layers. We designed that logical layer outputs of encoders transferred corresponding counterpart layers in a decoder referring to U-Net architecture. In addition, we proposed hybrid-type encoder models based on the DISCO architecture to capitalize on the effect of the model ensemble. We demonstrated performances of the DISCO-A and DISCO-B models derived from the proposed architecture in terms of f1-score and loss convergence per each epoch. We confirmed the DISCO-B had produced the best performance under symbolic label usage. In the development phase, our best score reached 0.7386 with 500 epochs.

1. Introduction

Skeletonization [1] is one of the object representation methods in images. It converts each target object from a shape to a set of joints and edges. Even though the skeletonized objects look simple, they can provide sufficient information in some situations with shapes of cartoons, emoticons, and road signs. Moreover, the simplified shapes help us perceive inherent object features intuitively. The skeletonization methods have been researched traditionally in the computer vision field by algorithmic approaches. Most of the algorithms tried to make them thin using geometric characteristics, for instance, medial axis calculation or morphological skeletonization. The legacy approaches work well for complex shapes having irregular curves. On the other hand, they possibly work poorly in objects having undistinguished features, such as symmetric and uniform shapes. So these approaches should make rules to applying proper algorithms depending on object features. Recently, some researchers proposed deep-learning based approaches to generate skeletal images for multiple objects, including human shapes. Many kinds of research focused on increasing skeletonization accuracy for the human body using joint-edge relationships since the OpenPose [2]. However, skeletonized datasets for general objects such as animals and insects have existed relatively rare.

In SkelNetOn 2019 Challenge [3], a Pixel SkelNetOn dataset containing binary mask and skeleton images for various objects was unveiled to understanding target shapes and their geometric characteristics. In terms of deep-learning methodologies, the goal of this challenge was to generate skeletonized results using the provided dataset. Applying autoencoder models that can transform source data to different information domains is one of the candidates for this task. Autoencoder tasks such as instance and semantic segmentation are examples of autoencoder applications. The autoencoder encodes input images to latent features, as shown in Figure 1. Following that, it decodes to generate images from the features. In these types of generative tasks, generative adversarial network models [4] may be useful in dealing with the given problem. In practice, a baseline model devised by SkelNetOn authors was trained based on the Pix2Pix method [5], and it was announced to perform around 0.6244 in terms of f1-score. Referred to the baseline, we trained our Pix2Pix model based on U-Net [6] backbone with configuring mask and skeleton images as input and output, respectively. We confirmed the model had a similar performance. We discovered that some pixels around joints did not decode well during visual inspection for the prediction results. Thus, we assumed that the decoding performance could be increased by utilizing sup-

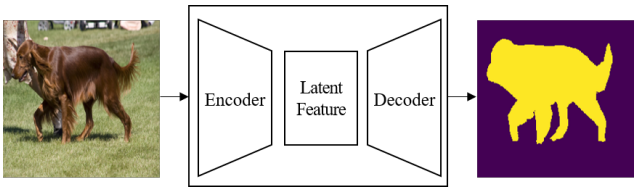


Figure 1. An example of semantic segmentation with the autoencoder: a sample in PASCAL VOC 2012 dataset [7]

plementary skeletal input images when training deep neural networks.

In this paper, we propose a DISCO architecture (Dual Input Streams autoenCOder) based on the U-Net models to dealing with additional skeleton image input. We consider U-Net, U-ResNet [8], and U-Transformer [9] as backbone neural network models to design our skeletonization models. At the beginning of neural network training, the proposed method generates reference skeleton images with traditional thinning methods. Here, we used Zhang's [10] and Guo's [11] thinning algorithms that had been already implemented in one of the extensions in the OpenCV for convenience. We designed the DISCO to the manner of modifying the backbone models to allocating dual encoders. The encoders were connected to a decoder via additional paths to transfer mask and skeleton features. We evaluated the performances of the proposed DISCO models in terms of f1-score and loss for training. In this paper, our contributions are as follows:

- In provided Pixel SkelNetOn dataset, we tried to utilize additional materials to training by guessing symbolic labels, and then we presented training results comparing with cases of using binary labels. Under the label settings, we trained baseline U-Net models and our models designed by the proposed DISCO architecture. The learning trends for those models will give us insights into optimization methodology researches for deep-learning based skeletonization.

- We presented our training pipeline to make reference skeleton images and corresponding model architecture. The proposed model architecture handled mask and reference images simultaneously. We showed an example of the DISCO model based on U-Net architecture, including inter-layer feature flows and their combinations. Moreover, the DISCO was designed to allocate heterogeneous encoders. Blending between the different types of encoders could enhance skeletonization performance by ensemble effects. We expect to help designing better architectures by referring to our methodologies.

2. Pixel SkelNetOn Dataset

The Pixel SkelNetOn dataset consists of pairs of source masks and target skeletons. The images are a single-channel gray color with 256 $\times$ 256 pixels resolution. Even though the images are gray-colored attributes, each pixel only contains a value which is one of 0 and 255. This dataset provides training, validation, and test samples with 1218, 241 pairs, and 266 masks, respectively. Here, we noticed that file names of masks for training and testing were available to guess true symbolic labels when analyzing the provided dataset. In the development phase, there were 90 labels in train shapes and 68 labels in test shapes. The test labels were a subset of train labels. We configured 91 labels, including a background tag. Logit vectors corresponding with the true labels were available, so cross-entropy [12] might be applicable for loss calculation and skeleton prediction. However, we hardly knew the effects of skeletonization performances when using the auxiliary labels. For this reason, we also tested under a condition of binary labels. The binary labels represented background and skeleton pixels. Likewise earlier, we used cross-entropy for loss calculation.

3. Proposed DISCO Model

3.1. Data Augmentation

As previously stated, the provided dataset contains 1218 pairs of images for training our models. It is insufficient to train objective models to improve performance. Thus, data augmentation is required to ensure an adequate quantity of training materials. In general, data augmentation involves manipulating source images through rotation, flipping, resizing, masking, and erasing. The Pixel SkelNetOn dataset contains images of both the source and the target. The images in a pair should have the same geometric properties when manipulating. In this challenge, we excluded resizing, masking, and erasing to preserve inherent skeletal characteristics for source masks. We applied random angle rotation, horizontal and vertical flips for data augmentation with 50% probability when loading the image samples.

3.2. Skeletonization

Even though the data augmentation increased the number of images, decoding performance was still insufficient. Some edge or joint pixels did not appear when examining the prediction results. To resurrect the vanished skeleton components, we will incorporate reference skeleton images into neural network models. Traditional thinning algorithms make it simple to obtain reference skeleton images. We consider Zhang's and Guo's thinning methods because they are already included in OpenCV extension packages. In the case of Python3, reference skeleton images are generated by *cv2.ximgproc.thinning* method.

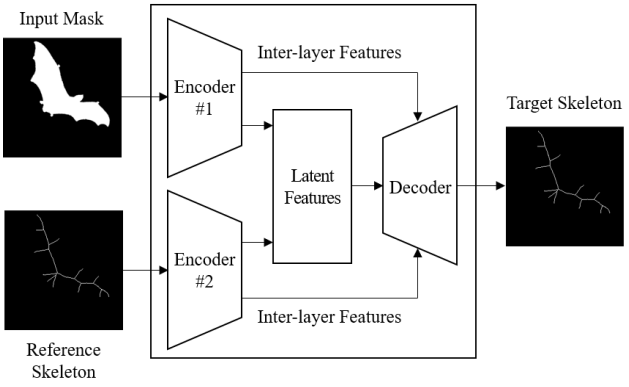


Figure 2. Simplified information flow for the proposed DISCO architecture

3.3. Model Architecture

U-Net models are useful to transform input images to other types of images. Usually, the U-Net models have been used to generate interpretable output results. For instance, the U-Net models can make mask images from the source in semantic segmentation tasks. Those results help to make decisions easily. The U-Net models are designed to have two main parts. The first one is an encoder, and the other one is a decoder. The encoder and decoder are built by stacking up logical layers. Each logical layer is structured with convolution, pooling, batch normalization blocks sequentially. Intermediate logical layers in the encoder and decoder are connected to equivalent-depth layers. The decoder possibly yields enhanced output results by referring to down-sampled features made from the encoder.

Similar to the U-Net architecture, we proposed the DISCO architecture with a dual encoder structure. The DISCO architecture intended to use additional features from reference skeleton images during the decoding process. As shown in Figure 2, the two images are fed into the encoders in turn. Convolutional filters fuse the two feature streams from the encoders' final logical layer at latent space. The decoder receives latent features corresponding to the convolutional filter outputs. Inter-layer features and latent features are merged at the same time. By reflecting from the convolutional features, the decoder restituted prediction images by passing through intermediate deconvolution layers.

In Figure 3, there is an example of a DISCO architecture that is modified based on the U-Net backbone. In the figure, there are two input paths. The first input path extracts mask features. Likewise, the second input path extracts reference skeleton features. The features go through logical layers, and they are transformed deeper and wider features. The features are downsampled with convolutional, maxpool, and batch-normalization blocks. The transformed features are delivered to the next logical layer and decoder's counterpart layer. These operations repeat until the fragmented features are reached the final layer. Final features by each encoder arrive at a latent space. After merging the features, they pass through convolutional blocks, then feed into a decoder's first layer. Here, the latent features are made by concatenating with two encoder's output features coming from mask and reference skeleton images. The decoder's logical layer yields upsampled features by transpose convolutional blocks. The upsampling process repeats until the skeleton prediction is finished.

In this paper, we proposed DISCO-A and DISCO-B models. The DISCO-A had the same architecture in Figure 3. The DISCO-B had a similar structure with the DISCO-A model, but the second path and decoder were replaced with U-Transformer components as well as changing the number of encoder layers. The DISCO-B had a similar structure to the DISCO-A model. But this model was modified to having 4 logical layers for the encoders and decoder by referring to the U-Transformer. To train the proposed models, we considered applying two loss functions which were known as CE (cross-entropy) and CE Dice (cross-entropy + dice [13]). The dice coefficient looks similar to the IoU (intersection over union) definition, and it leads predictions to make resembling targets. The dice coefficient is defined in the following Equation (1).

$$dice = \frac{2 \times |A \cap B|}{|A| + |B|}, \tag{1}$$

where A and B are the set of prediction and true pixels respectively.

4. Evaluation

4.1. Metric

In our task, pixels in prediction results have only two values. Thus, a goal of the given task is possible to regard as pixel classification. In the Pixel SkelNetOn baseline, models are evaluated by f1-score. Generally, the f1-score is defined by the following Equation (2).

$$f1 - score = \frac{2 \times precision \times recall}{precision + recall}, \tag{2}$$

where

$$precision = \frac{TP}{TP + FP} \tag{3}$$

$$recall = \frac{TP}{TP + FN}. \tag{4}$$

The TP, FP, and FN are the number of pixels for true positives, false negatives, and false positives, respectively.

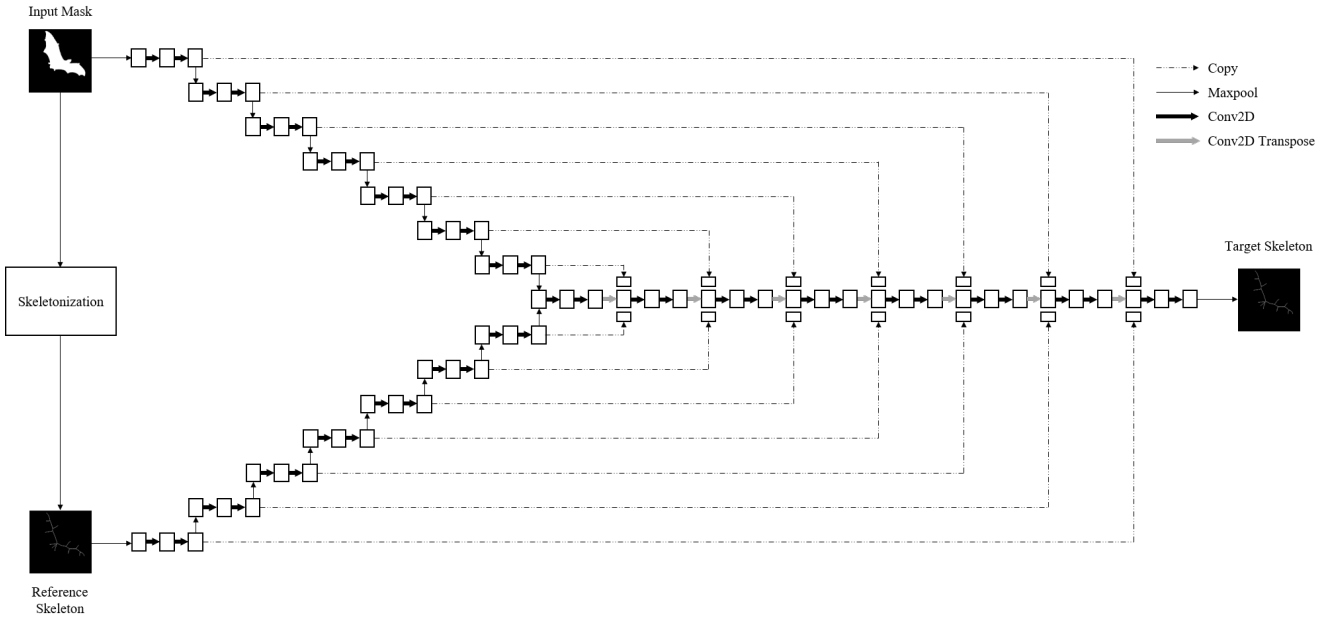


Figure 3. Example of specific DISCO architecture based on a U-Net backbone with 8-logical layers

4.2. Experiment Setup

Our evaluation software was written in Python 3.8 and Pytorch 1.8 environments. Other models, except U-ResNet, were created from scratch. Because the U-ResNet was built to use the ResNet18 backbone, it was trained using transfer learning. In this case, some pre-trained layers copied from the ResNet backbone had their weight, bias, and gradient terms froze. In addition, we calculated f1-score for the traditional thinning methods to compare the performance of deep-learning based models. We intended to train the proposed models on an NVIDIA GeForce RTX3090. We chose batch size 3 based on graphic memory capacity and U-Transformer model size. Because batch size can affect the convergence characteristics of performance curves, we use the same batch size for all training processes.

Also, the target skeleton images contain little positive pixels comparing with negative pixels. It means the pixels representing the skeleton target are sparse. Usually, the sparsity leads to overfitting when effective data samples are not sufficient. For this reason, we applied learning rate scheduling to alleviate bad effects such as falling into local minima. We used a *ReduceLROnPlateau* method with a default patience setting (=10). The rest of the hyperparameters are described in the following Table 1.

Table 1. Configuration of hyperparameters

parameters	properties
baseline models	U-Net, U-ResNet, U-Transformer
proposed models	DISCO-A (dual U-Net encoders), DISCO-B (U-Net & U-Transformer encoders)
thinning	Zhang (Z), Guo (G)
number of labels	2 (binary), 91 (symbol)
loss function	cross entropy (CE), cross entropy + dice (CE Dice)
optimizer	Adam
epoch	100
learning rate	$1.0 \times 10^{-3} \sim 1.0 \times 10^{-8}$
LR decay rate	0.5

4.3. Performance Results

In Table 2, we provided the performance results in terms of the f1-scores and the loss values. This table consisted of three groups. The first one was the performance of the traditional algorithms that decided skeleton pixels by calculation to geometric distance. We calculated the f1-score by averaging 10-times experiments for the augmented training dataset. We got 0.3130 and 0.3011 for Zhang's and Guo's methods, respectively. The second one was the performance of the U-Net based models that decided skeleton pixels with fully convolutional neural networks. In this case, the U-Transformer models showed good performance results. Even though the U-Transformer models were proposed quite recently, they still possessed lower f1-scores than the Pix2Pix baseline. The last one was the performance of the proposed DISCO models. The DISCO-B models showed higher performance results than the others. In the results, the best f1-score for the DISCO-B was 0.6668.

Table 2. The skeletonization performance for training dataset

model	f1-score	loss
Traditional Algorithm (Zhang)	0.3130	-
Traditional Algorithm (Guo)	0.3011	-
U-Net (CE, binary)	0.4072	0.0245
U-Net (CE, symbol)	0.4355	0.0240
U-Net (CE Dice, binary)	0.4128	0.0340
U-Net (CE Dice, symbol)	0.2688	0.0364
U-ResNet (CE, binary)	0.4968	0.0204
U-ResNet (CE, symbol)	0.4981	0.0204
U-ResNet (CE Dice, binary)	0.5269	0.0290
U-ResNet (CE Dice, symbol)	0.5168	0.0294
U-Transformer (CE, binary)	0.5526	0.0199
U-Transformer (CE, symbol)	0.5708	0.0190
U-Transformer (CE Dice, binary)	0.6095	0.0245
U-Transformer (CE Dice, symbol)	0.5984	0.0258
DISCO-A (Z, CE, binary)	0.5569	0.0209
DISCO-A (Z, CE, symbol)	0.5840	0.0206
DISCO-A (Z, CE Dice, binary)	0.5905	0.0283
DISCO-A (Z, CE Dice, symbol)	0.5943	0.0288
DISCO-A (G, CE, binary)	0.5457	0.0208
DISCO-A (G, CE, symbol)	0.5253	0.0222
DISCO-A (G, CE Dice, binary)	0.5669	0.0286
DISCO-A (G, CE Dice, symbol)	0.6082	0.0278
DISCO-B (Z, CE, binary)	0.5081	0.0214
DISCO-B (Z, CE, symbol)	0.6468	**0.0151**
DISCO-B (Z, CE Dice, binary)	0.6048	0.0242
DISCO-B (Z, CE Dice, symbol)	**0.6668**	0.0206
DISCO-B (G, CE, binary)	0.6111	0.0156
DISCO-B (G, CE, symbol)	0.6375	0.0206
DISCO-B (G, CE Dice, binary)	0.6182	0.0253
DISCO-B (G, CE Dice, symbol)	0.6167	0.0284

In Figure 4, parts of skeletonized results for our best DISCO-B model were presented to verify the benefits of deep-learning based approaches. In some situations, algorithmic approaches possibly lose detailed skeletons since they are hard to adapt target skeletons. However, the DISCO-B was able to recover branch skeletons as shown in Figure 4-(a). Also, the DISCO-B was able to alleviate noisy skeletons as shown in Figure 4-(b). However, the third image in Figure 4-(b) showed that the proposed model still had skeleton vanishing problems.

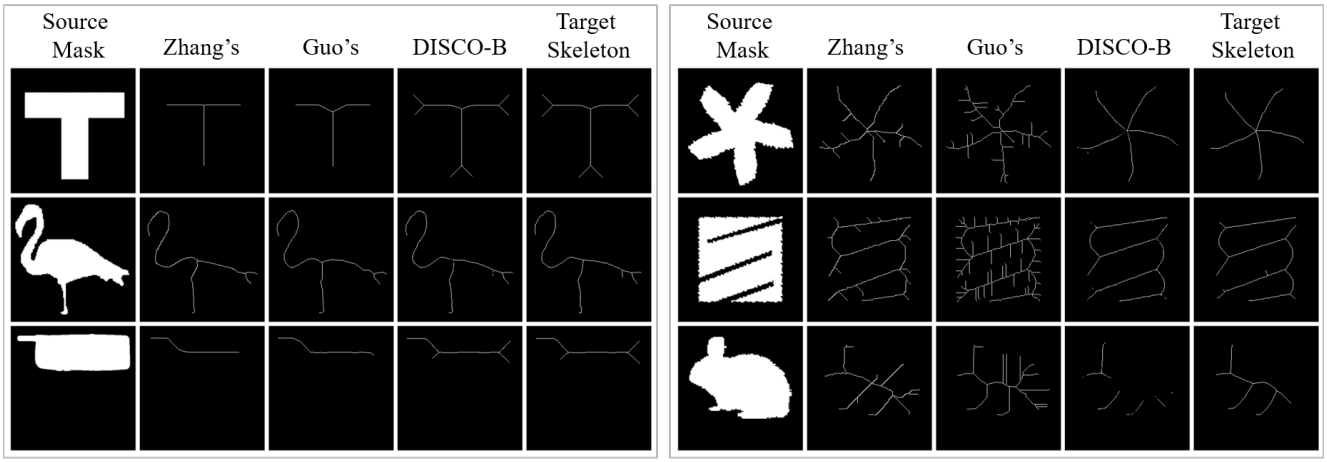
In Figures 5-12, training curves were presented. We controlled loss function and label type to drawing the performance curves to compare model performances purely. For all of the curves, the proposed DISCO models are better than other models. In the f1-score comparisons for the DISCO-B, we found out thinning methods affect model performances. In our experiment, Zhang's method showed better performances. Also, in the case of symbolic labels, we verified the proposed models took performance gains comparing with the other models. In the development phase, we reached 0.7386 with 500 epochs.

5. Conclusion and Future Work

We proposed the DISCO architecture for dual input streams in this paper. We provided DISCO-A and DISCO-B models based on the architecture to handle skeletonization tasks. The DISCO-B, in particular, was designed to be a hybrid type based on U-Net and U-Transformer, so we expected the model to improve skeletonization performance. As expected, we found that the DISCO-B had higher f1-scores than the other models in the evaluation results. However, skeleton outputs for the proposed models were still imperfect due to vanished pixels. In our architecture, the input skeletons affected the performance results positively even though skeletonization methods were quite old. We expect that better performance results will probably come when clean reference skeletons are available. Therefore, it is necessary to evaluate the proposed model with recent skeletonization algorithms in future work.

Acknowledgment

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government. [21ZR1100, A Study of Hyper-Connected Thinking Internet Technology by autonomous connecting, controlling and evolving ways]



(a) (b)

Figure 4. Benefits of the deep-learning based approaches: (a) Branch recovery, (b) Denoising

References

[1] P. K. Saha, G. Borgefors, and G. S. di Baja, "A survey on skeletonization algorithms and their applications," *Pattern recognition letters*, vol. 76, pp. 3–12, 2016.

[2] D. Osokin, "Real-time 2d multi-person pose estimation on cpu: Lightweight openpose," *arXiv preprint arXiv:1811.12004*, 2018.

[3] I. Demir, C. Hahn, K. Leonard, G. Morin, D. Rahbani, A. Panotopoulou, A. Fondevilla, E. Balashova, B. Durix, and A. Kortylewski, "Skelneton 2019: Dataset and challenge on deep learning for geometric shape understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, June 2010.

[8] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," in *Deep learning and data labeling for medical applications*, pp. 179–187, Springer, 2016.

[9] O. Petit, N. Thome, C. Rambour, and L. Soler, "U-net transformer: Self and cross attention for medical image segmentation," *arXiv preprint arXiv:2103.06104*, 2021.

[10] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[11] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Communications of the ACM*, vol. 32, no. 3, pp. 359–373, 1989.

[12] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[13] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571, IEEE, 2016.

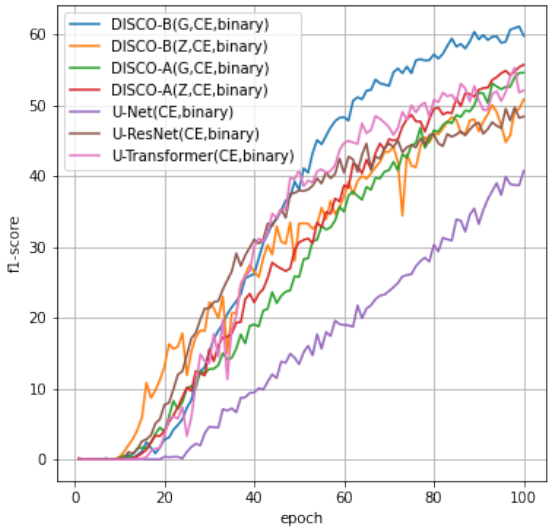


Figure 5. Training curve with respect to f1 score under CE criterion and binary labels

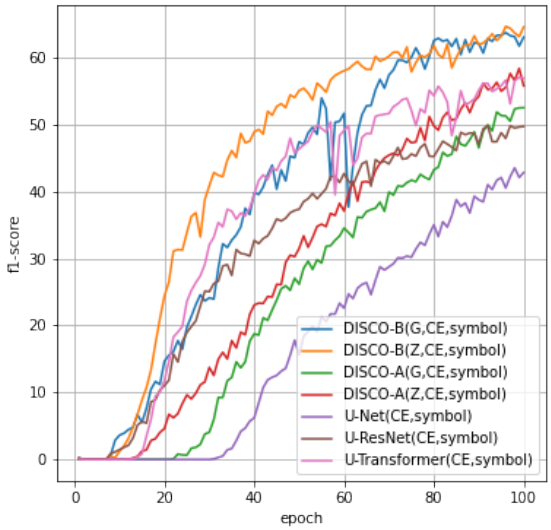


Figure 7. Training curve with respect to f1 score under CE criterion and symbolic labels

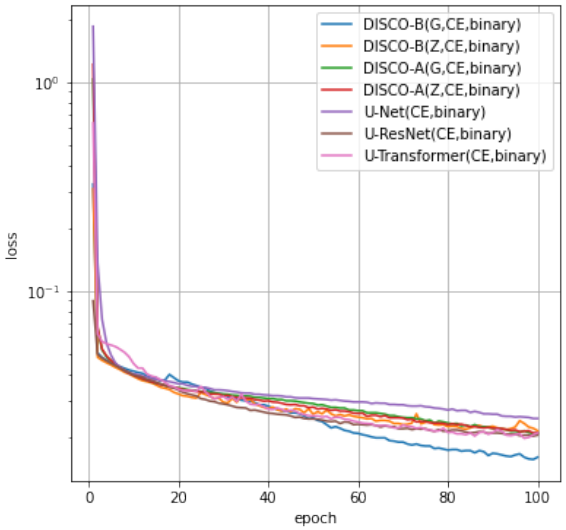


Figure 6. Training curve with respect to loss under CE criterion and binary labels

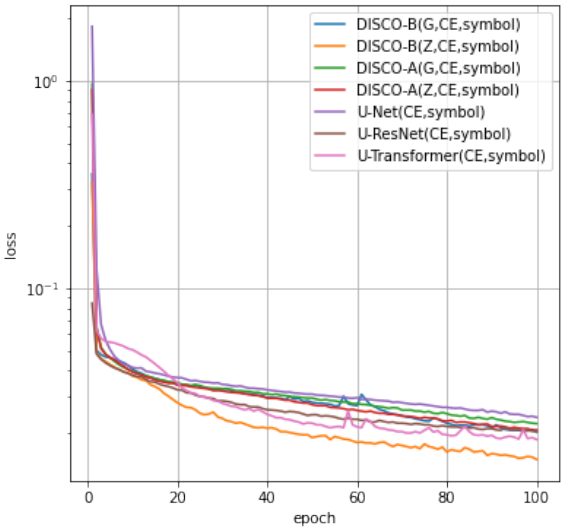


Figure 8. Training curve with respect to loss under CE criterion and symbolic labels

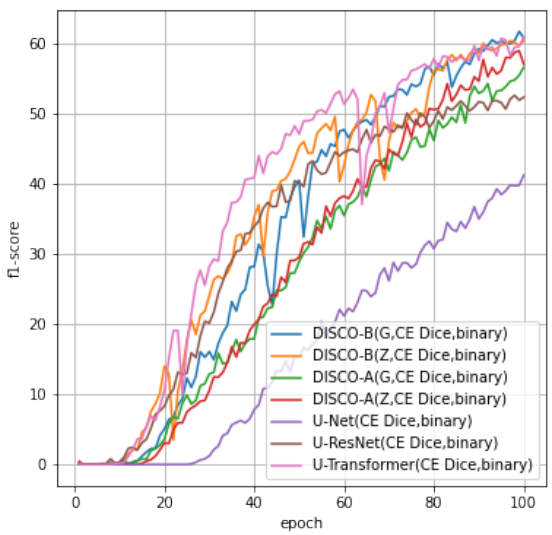


Figure 9. Training curve with respect to f1 score under CE Dice criterion and binary labels

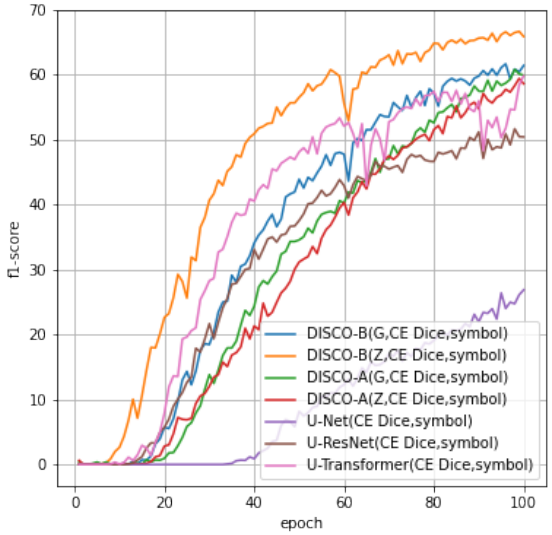


Figure 11. Training curve with respect to f1 score under CE Dice criterion and symbolic labels

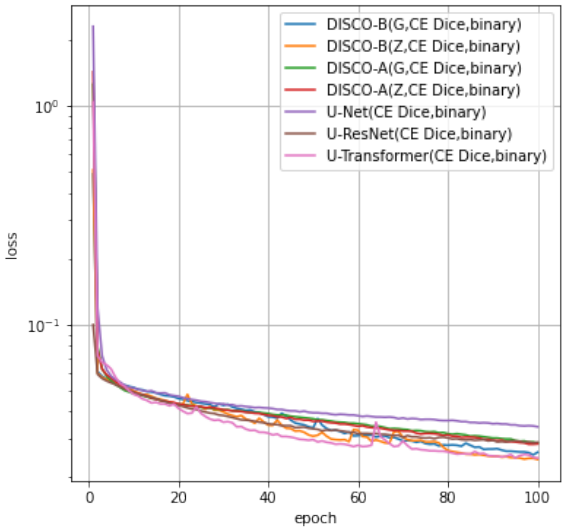


Figure 10. Training curve with respect to loss under CE Dice criterion and binary labels

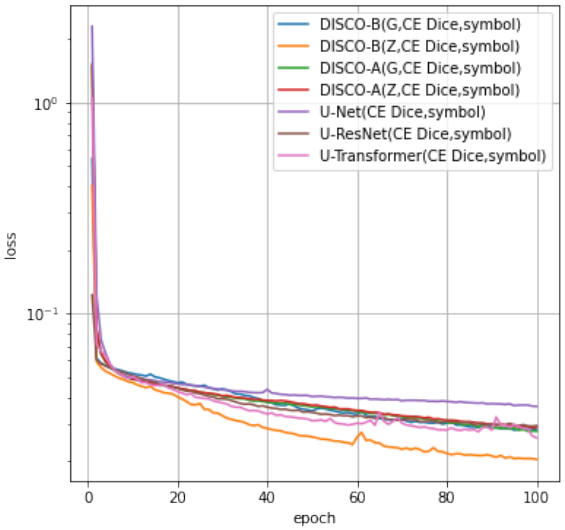


Figure 12. Training curve with respect to loss under CE Dice criterion and symbolic labels