How do different semantic segmentation models perform at recognizing faces with different cultural backgrounds?

Estefania S. Cunha

***Abstract*—This report is written for the project in the Master Course "Bild 1" at the Frankfurt University of Applied Sciences in the Summer term of 2024. The project's objective was to develop and evaluate semantic segmentation models to enhance the understanding of semantic segmentation in computer vision. Using the DeepLabV3 and PSPNet models, I created and trained semantic segmentation models. These models were applied to the Mut1ny dataset, a labeled face/head segmentation dataset, and tested on custom images of individuals from different cultural backgrounds, specifically Asian, Black, and Caucasian ethnicities. Evaluation metrics such as F1 score, recall, precision, accuracy, and IoU were calculated to compare the performance of the models. The results provide insights into how these models perform in recognizing faces from diverse cultural backgrounds.**

***Index Terms*—Semantic Segmentation, DeepLabV3, PSPNet, Mut1ny Dataset**

I. Introduction

Semantic segmentation is a crucial technique in computer vision that involves classifying each pixel in an image into a predefined category [1]. The process entails partitioning an image into non-overlapping regions based on features such as gray-scale, color, texture, and shape, enabling machines to automatically segment and recognize objects within [2]. In the context of face recognition, semantic segmentation helps accurately identify and distinguish facial features, which is essential for applications such as identity verification, emotion detection, and demographic analysis [3].

Given the still-needed research on the performance of different semantic segmentation models on facial structures of various ethnicities, as highlighted by the paper *Racial Identity-Aware Facial Expression Recognition Using Deep Convolutional Neural Networks* [4], and to gain a deeper understanding of how different model architectures perform, my research question for this project is: **"How do different semantic segmentation models perform at recognizing faces with different cultural backgrounds?"**.

This study aims to evaluate the **DeepLabV3** and **PSPNet** models using the Face/Head Segmentation dataset Mut1ny. The performance of these models will be assessed using metrics such as Accurancy, F1 score, IoU, Loss, Precision and Recall. By developing and training these models on the Mut1ny dataset [5], the objective is to understand their effectiveness in recognizing faces across various ethnic groups. This evaluation will provide valuable insights into improving the fairness and accuracy of facial recognition systems.

II. Related Work

During my research, I found several models used in semantic segmentation for face segmentation, such as Fully Convolutional Networks [6], U-Net [7], and SegNet [8]. Additionally, I discovered various datasets for face and head segmentation. These include JAFFE [9], which consists of 213 images of different facial expressions from various Japanese female subjects; the Extended Cohn-Kanade [10] dataset, which contains 593 video sequences from 123 different subjects, ranging in age from 18 to 50 years with a variety of genders and heritage; and the Radboud Faces Dataset [11], which comprises pictures of 67 models (both adults and children, male and female) displaying 8 emotional expressions.

In search of models and datasets that fit within my project scope, I decided to focus on developing the DeepLabV3 and PSPNet models and using the Mut1ny dataset [5]. This decision was made due to the specific requirements and time constraints of my project, ensuring a focused and feasible approach to achieving the research objectives.

III. Methodology

A. Model's Description

1) DeepLabV3: As mentioned previously, one of the models I used was DeepLabV3 (Fig. 1), an advanced semantic segmentation architecture. DeepLabV3 addresses multi-scale object segmentation using atrous convolution and improves the Atrous Spatial Pyramid Pooling (ASPP) module. This includes global average pooling, 1×1 convolution with 256 filters and batch normalization, and bilinearly upsampling the features. The refined ASPP comprises one 1×1 convolution and three 3×3 convolutions with atrous rates of 6, 12, and 18, all with 256 filters and batch normalization, plus the image-level features [12].

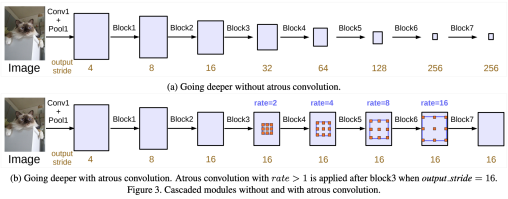


Fig. 1: DeepLabV3 Architecture

2) PSPNet: Another model I used is PSPNet, or Pyramid Scene Parsing Network, a semantic segmentation model that utilizes a pyramid parsing module to exploit global context information through different-region-based context aggregation. This combination of local and global clues enhances prediction reliability. Given an input image, PSPNet uses a pre-trained CNN with a dilated network strategy to extract the feature map. The final feature map is the same size as the input image. The pyramid pooling module then gathers context information using a 4-level pyramid with pooling kernels that cover the entire image, half of it, and smaller portions. These pooled features are fused as the global prior, concatenated with the original feature map, and followed by a convolution layer to generate the final prediction map [13]. In summary, while DeepLabV3 focuses on capturing multi-scale details with atrous convolution, PSPNet emphasizes combining local and global context through pyramid pooling. Using both models allows for a comprehensive evaluation of face segmentation across diverse datasets.

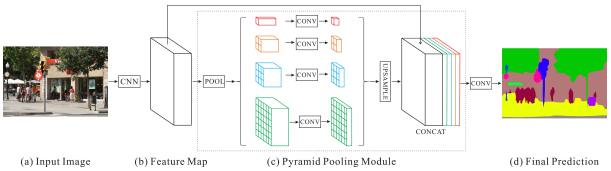


Fig. 2: PSPNet Architecture

B. Dataset

The dataset I used for this project is Mut1ny, a head/face labeled dataset that contains over 16,500 (16,557) fully pixel-level labeled segmentation images. It includes facial images from different ethnicities, ages, and genders, making it a well-balanced dataset (Fig. 3). Additionally, it offers a wide variety of facial poses and different camera angles, providing good coverage from -90 to 90 degrees facing [5].

Fig. 3: Example images and masks from Mut1ny face/head segmentation dataset source from [5].

For each real image in the dataset, there exists a corresponding PNG RGB label image pair. This label image encodes 11 different labeled areas of the face using specific RGB values. The RGB labels for these areas are as follows:

Red value	Green value	Blue value	Label description
0	0	0	Background
255	0	0	Lips/mouth
0	255	0	Eyes
0	0	255	Nose
128	128	128	General face/head
255	255	0	Hair
0	255	255	Eyebrows
255	0	255	Ears
0	128	255	Teeth
255	192	192	Facial hair/beard
0	128	128	Specs/sunglasses

TABLE I: Label descriptions with corresponding RGB values

C. Data Preparation

To prepare the data for training both models, the following steps were followed:

1) **Download Dataset:** The Face/Head Segmentation dataset from Mut1ny, which contains over 16.5k labeled segmentation images, was downloaded.

2) **Organize Dataset:** The dataset was organized into multiple folders based on types such as real images, multiple faces, gender, and names of persons/models. For example, a folder named femalealison1 contains images of ”Alison” belonging to the ”female” class.

3) **Group into Categories:** These folders were grouped into four broad categories: male, female, real images, and multiple faces. Each folder was iterated through to determine its category, and images were copied to the relevant folder.

4) **Clean Dataset:** The dataset was cleaned by removing duplicate and near-duplicate images. This was achieved by hashing the images to obtain a numerical representation and removing images with identical hashes.

5) **Create Subsamples:** After cleaning, approximately 14k samples remained. Subsamples were then created for faster experimentation.

6) **Subsample Selection:** For subsampling, all samples from the ”real images” category were used, 50 images were randomly sampled from the ”multi-person images” category, and 10 images were randomly sampled from other categories (e.g., images of the same person like female: Alison, male: Gabriel).

7) **Final Subsamples:** After subsampling, 2096 samples were ready for model training.

D. Setup

1) DeepLabV3 Setup: To train the DeepLabv3 model for semantic segmentation, I first created a data loader by splitting the dataset of 2096 subsampled images into training and testing sets. I then applied several transformations to the images, including resizing them to 256x256, converting them from NumPy arrays to PyTorch tensors, and normalizing them. I used PyTorch's **'DataLoader'** to load the dataset with a specific batch size.

Next, I built the model using the **'deeplabv3_resnet_101'** backbone from the **'torchvision'** library, which is pre-trained on the COCO dataset. I defined the training components, including the loss function, optimizer, and model evaluation metrics. I set up a training pipeline to fine-tune the model for **25 epochs**.

During training, I fed batches of the subsampled data into the model, calculated the loss, and backpropagated to update the model weights. I evaluated the model's performance on the validation set and saved the best model weights based on the validation loss. Training the model on my MacBook Pro 13" with an M1 chip, using PyCharm, took around **31 hours and 17 minutes**.

Additionally, I evaluated the model using metrics such F1 score, recall, precision, accuracy, and IoU. I saved the metrics from each epoch in an Excel file for further analysis.

2) PSPNet Setup: To train the PSPNet model for semantic segmentation, I used a similar approach as with the DeepLabv3 model. I first created a data loader by splitting the dataset of 2096 subsampled images into training and testing sets. I applied several transformations to the images, including resizing them to 256x256, converting them from NumPy arrays to PyTorch tensors, and normalizing them. I used PyTorch's **'DataLoader'** to load the dataset with a specific batch size. Next, I built the model using the PSPNet architecture with a ResNet101 encoder pre-trained on ImageNet. The model was configured to handle RGB images and had three output channels for segmentation classes. I trained the model for 25 epochs, using the same data loaders for training and testing. During training, I computed the metrics F1 score, IoU, recall, precision, and accuracy for each epoch and saved the metrics in a CSV file. I saved the best model weights based on the lowest validation loss.

The training took place on my MacBook, using PyCharm, which took around **7 hours and 22 minutes**.

IV. Results

A. Evaluation Metrics Results

After training the models, I received a CSV file with evaluation metrics for each of the 25 epochs, including train loss, test loss, train F1 score, train recall, train precision, train IoU, test F1 score, test accuracy, test recall, test precision, and test IoU. I chose these metrics for their comprehensive insights into model performance:

- F1 Score: Balance between precision and recall, provides a balanced measure of the model's accuracy.
- IoU: Measures the overlap between the predicted segmentation and the ground truth, divided by their union.
- Recall: The ratio of correctly predicted positive observations to all actual positives, indicating the model's ability to detect relevant instances.
- Precision: Accuracy of positive predictions.
- Loss: Difference between predictions and actual outcomes for training (train loss) and testing (test loss).
- Accurancy: Correct predictions over total predictions.

Training metrics assess the model's performance on the training data, indicating how well it learns from data it has seen. Test metrics evaluate performance on unseen test data, demonstrating the model's ability to generalize. By monitoring both metrics across 25 epochs, the learning progress could be tracked, and any signs of overfitting could be identified. This provided a clear understanding of the models' strengths and informed further experimentation.

B. DeepLabV3 Evaluation Metric Results

Given that the DeepLabv3 model was trained on only 2096 images, here's what I observed from the Plots I generated found in figure 4:

- Limited Data: Training on 2096 images is quite small for deep learning models, especially for segmentation tasks that need large and diverse datasets.
- Performance: The model shows promising trends in accuracy, F1 score, and IoU, but the fluctuations in test metrics suggest it might not generalize well to new data.
  - Accurancy: Training accuracy steadily improves from 73.66% to 88.05%, but test accuracy fluctuates from 81.33% to 83.69%.
  - F1 Score: Both training and test F1 scores increase. Training F1 score starts at 65.98% and improves to 81.64%, while test F1 score starts at 73.28% and fluctuates, ending at 75.55%.
  - IoU: IoU shows improvement starting from 40.70% and increasing to 52.21%, though test IoU is inconsistent, ending at 48.18%.
  - Loss: Training loss decreases consistently from 0.0586 to 0.0260, while test loss fluctuates, starting at 0.0377 and ending at 0.0355.
  - Precision and Recall: Training precision improves from 53.12% to 72.12%, and recall improves from 90.50% to 95.59%. Test precision improves from 62.01% to 66.57%, and recall fluctuates, starting at 91.40% and ending at 89.73%.

Overview:

- Strenghts: Shows learning capability with improved training metrics.
- Weakness: Fluctuations in test metrics indicate inconsistent performance and possible overfitting.

- Conclusion: While promising, the small dataset size limits its ability to segment faces accurately. Increasing the dataset could potentially increase the performance.

C. PSPNet Metric Results

The PSPNet model was also trained on only 2096 images. As I stated before in Section IV-B, 2096 images is a small dataset for models to effectively learn, especially in regards to segmentation tasks.

Like the trained DeepLabV3 model, it shows promising trends in accuracy, F1 Score, and IoU, but the fluctuations in test metrics suggest that it might not generalize well to new data.

The following observations could be retrieved from Figure 5:

- Accuracy: Training accuracy steadily improves from 54.38% to 62.10%, but test accuracy fluctuates from 54.98% to 61.15%.
- F1 Score: Both training and test F1 scores increase. Training F1 score starts at 46.25% and improves to 57.17%, while test F1 score starts at 49.81% and fluctuates, ending at 56.59%.
- IoU: IoU shows improvement, starting from 24.88% and increasing to 31.67%, though test IoU is inconsistent, ending at 33.46%.
- Loss: Training loss decreases consistently from 0.3717 to 0.0797, while test loss fluctuates, starting at 0.1623 and ending at 0.0649.
- Precision and Recall: Training precision improves from 35.56% to 43.13%, and recall improves from 69.53% to 88.94%. Test precision improves from 36.70% to 41.68%, and recall fluctuates, starting at 80.91% and ending at 91.58%.

Overview:

- Strengths: Shows learning capability with improved training metrics.
- Weaknesses: Fluctuations in test metrics indicate inconsistent performance and possible overfitting.
- Conclusion: The model is promising like the DeepLabV3 model, but the small dataset limits its ability to segment faces accurately.

D. Summary of Differences

After analyzing and comparing both models based on their evaluation metrics output, DeepLabv3 shows significantly higher training and test accuracy compared to PSPNet, suggesting it learns better from the training data and generalizes slightly better.

Both models improve their F1 scores, but DeepLabv3 achieves higher values and shows slightly more stability. DeepLabv3 also has higher IoU scores for both training and test datasets, indicating better segmentation performance.

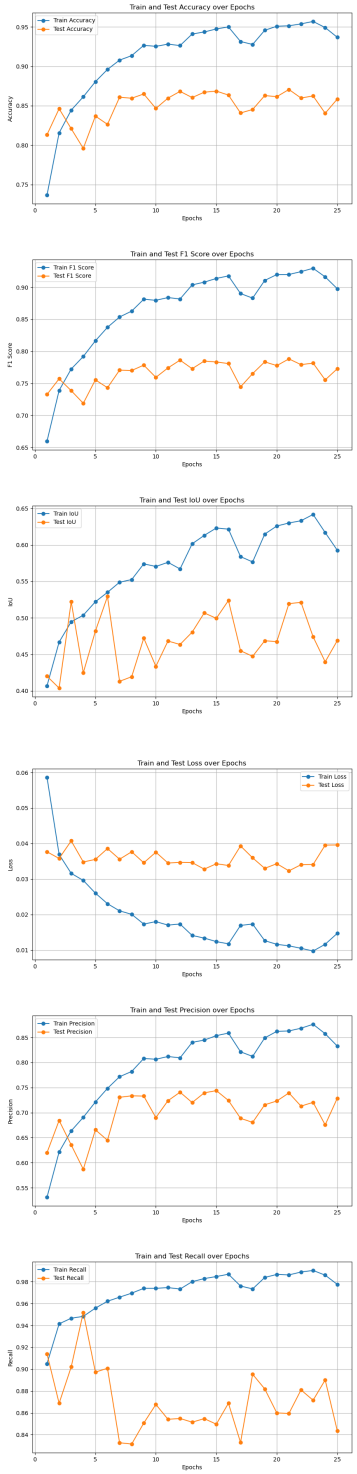


Fig. 4: Performance metrics of the DeepLabv3 model: (a) Accuracy, (b) F1 Score, (c) IoU, (d) Loss, (e) Precision, (f) Recall

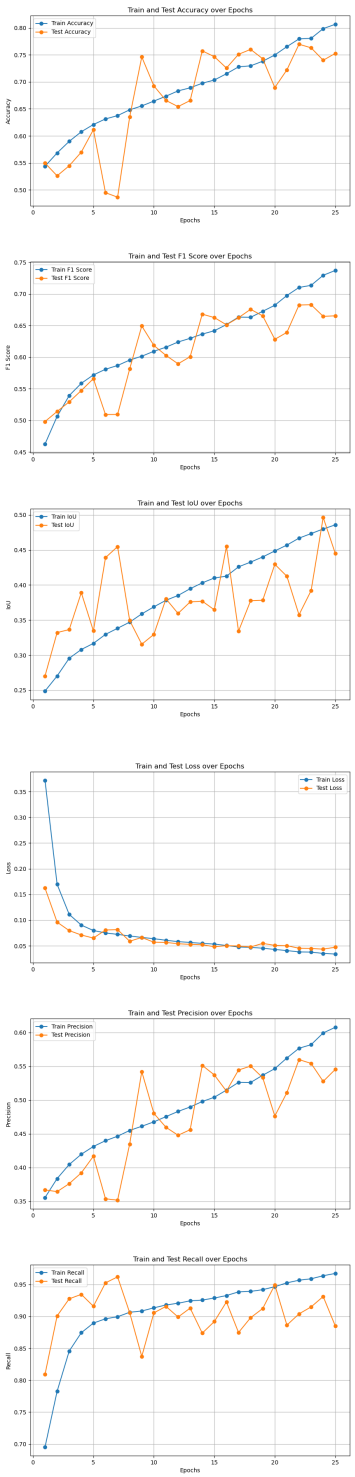


Fig. 5: Performance metrics of the PSPNet model: (a) Accuracy, (b) F1 Score, (c) IoU, (d) Loss, (e) Precision, (f) Recall

Training loss decreases consistently for both models, but DeepLabv3 starts with a much lower loss value. Test loss is more stable in DeepLabv3.

DeepLabv3 shows higher precision and recall values for both training and test sets, indicating better performance in identifying and capturing relevant instances.

While both models show promise, the DeepLabv3 model generally outperforms the PSPNet model in terms of accuracy, F1 score, IoU, precision, and recall. However, both models exhibit fluctuations in test metrics, indicating inconsistent performance and possible overfitting. Like previously stated before, I believe that if I increasing the dataset size both models could potentially achieve better and more stable performance in segmenting faces.

V. Experiment

After training my models, the next step was to apply them to my chosen images to evaluate their segmentation performance. I aimed to select images featuring individuals of different ages, ethnicities, and emotional expressions. Additionally, I wanted these images to display varying body positions and lighting conditions to thoroughly analyze how both models perform under diverse scenarios.

I focused on three ethnic groups: Asian, Black/African American, and Caucasian. Within each group, I selected images of individuals displaying three distinct emotions: sadness, happiness, and anger. Each ethnic group included three different people of varying ages, each portraying a different emotion. This resulted in a total of nine images.

The images were sourced from iStock under a purchased license. The selected images can be seen in the Table II.

	Sad	Happy	Anger
Asian			
Black/ African American			
Caucasian			

TABLE II: Images showing different emotions across ethnicities

A. Segmentation Results - DeepLabV3

The segmentation results from DeepLabv3 (Table III) shows that the model is able to identify and segment the

general shape of the face and some facial features. However, the segmentation is only partially accurate.

The model correctly identifies the general face/head regions but struggles with precise segmentation, particularly with finer details such as individual facial features.

The segmentation labels are not fully accurate, leading to some misclassification of regions. For instance, the model sometimes mislabels areas that should correspond to specific facial features. Eye-catching is the use of turquoise blue, which is intended to label ears but is instead often used to label hair.

The accuracy of segmentation varies across different emotions and ethnic groups. This suggests that the model's performance is inconsistent and might not generalize well to different types of faces.

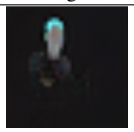
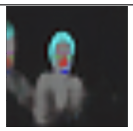
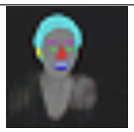
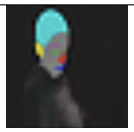
	Sad	Happy	Anger
Asian			
Black/ African American			
Caucasian			

TABLE III: Images showing Segmentation Results of DeepLabV3

B. Segmentation Results - PSPNet

The PSPNet model (Table IV)also shows an ability to segment the general face and some facial features, but like DeepLabv3, it is only partially accurate. Similar to DeepLabv3, PSPNet can identify the general face/head regions but struggles with finer details and precise segmentation.

PSPNet also mislabels some regions. The turquoise blue, intended for ears, is frequently used to label hair instead, leading to inaccuracies.

VI. Discussion

It is challenging to fully answer the question, "How do different semantic segmentation models perform at recognizing faces with different cultural backgrounds?" More training on a larger dataset is needed to provide a definitive answer. However, from what I could observe, both DeepLabv3 and PSPNet can segment faces of people with different ethnic backgrounds but their performance is only partially accurate, based on the small dataset I used.

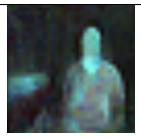
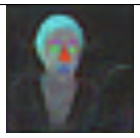
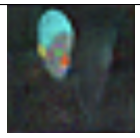
	Sad	Happy	Anger
Asian			
Black/ African American			
Caucasian			

TABLE IV: Images showing Segmentation Results of PSPNet

A significant issue is the mislabeling of regions. For example, both models often used turquoise blue to label hair instead of ears. This mislabeling likely stems from the models' inability to generalize well due to the limited training data. The segmentation accuracy varies across different emotions and ethnic groups, indicating inconsistent performance.

DeepLabv3 showed some promise but had inconsistencies in labeling and struggled with finer details. PSPNet had similar issues, with frequent mislabeling and inconsistent performance across different emotions and ethnic groups.

The models'were trained on only 2096 images out of 16,500 available in the MUTINY dataset. This limited data likely contributed to the inaccuracies observed. Training on a slightly smaller dataset took around 30 hours for DeepLabV3 and 7 hours for PSPNet, indicating that significantly more computational resources and time would be required to train on the full dataset.

For better segmentation and labeling, I believe that using a bigger dataset, refining the labeling process to reduce errors, and investing in more computational resources for longer training sessions could improve the models.

VII. Conclusion

In conclusion, while DeepLabv3 and PSPNet show potential in segmenting faces across different cultural backgrounds, their current limitations highlight the need for further training and refinement. By addressing generalization issues, limited data, and labeling inaccuracies, more accurate face segmentation can be achieved.

REFERENCES

[1] A. Garcia-Garcia, S. Orts, S. Oprea, V. Villena Martinez, P. Martinez-Gonzalez, and J. Rodríguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, May 2018. DOI: 10.1016/j.asoc.2018.05.018.

[2] W. Song, N. Zheng, R. Zheng, X. Zhao, and A. Wang, "Digital image semantic segmentation algorithms: A survey," *J. Inf. Hiding Multim. Signal Process.*, vol. 10, pp. 196–211, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:198355884.

[3] S. K. Khare, V. Blanes-Vidal, E. S. Nadimi, and U. R. Acharya, "Emotion recognition and artificial intelligence: A systematic review (2014–2023) and research recommendations," *Information Fusion*, vol. 102, p. 102 019, 2024, ISSN: 1566-2535. DOI: 10.1016/j.inffus.2023.102019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253523003354.

[4] M. Sohail, G. Ali, J. Rashid, *et al.*, "Racial identity-aware facial expression recognition using deep convolutional neural networks," *Applied Sciences*, vol. 12, no. 1, p. 88, 2022. DOI: 10.3390/app12010088. [Online]. Available: https://doi.org/10.3390/app12010088.

[5] Mut1ny, *Face/head segmentation dataset community edition*, https://store.mut1ny.com/product/face-head-segmentation-dataset-community-edition?v=3a52f3c22ed6, Accessed: 2024-07-17, 2023. [Online]. Available: https://store.mut1ny.com/product/face-head-segmentation-dataset-community-edition?v=3a52f3c22ed6.

[6] P. with Code, *Fully convolutional network (fcn)*, https://paperswithcode.com/method/fcn, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/method/fcn.

[7] P. with Code, *U-net segmentation*, https://paperswithcode.com/task/unet-segmentation, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/task/unet-segmentation.

[8] P. with Code, *Segnet*, https://paperswithcode.com/method/segnet, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/method/segnet.

[9] P. with Code, *Jaffe dataset (japanese female facial expression)*, https://paperswithcode.com/dataset/jaffe, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/dataset/jaffe.

[10] P. with Code, *Ck+ dataset (extended cohn-kanade dataset)*, https://paperswithcode.com/dataset/ck, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/dataset/ck.

[11] R. U. Nijmegen, *Radboud faces database (rafd)*, https://rafd.socsci.ru.nl/?p=main, Accessed: 2024-07-17, 2023. [Online]. Available: https://rafd.socsci.ru.nl/?p=main.

[12] P. with Code, *Deeplabv3*, https://paperswithcode.com/method/deeplabv3, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/method/deeplabv3.

[13] P. with Code, *Pspnet*, https://paperswithcode.com/method/pspnet, Accessed: 2024-07-17, 2023. [Online]. Available: https://paperswithcode.com/method/pspnet.