Enhancing Image Segmentation: A Comparative Study

Aaryan, Aayush Kumar, Darpan Gaur, Varun Gupta
Indian Institute of Technology Hyderabad
{co21btech11001, co21btech11002, co21btech11004, cs21btech11060}@iith.ac.in

Abstract

Image segmentation is a crucial task in computer vision with significant applications in various fields such as autonomous driving, medical imaging, and object detection. Accurate segmentation allows for precise identification and localization of objects within an image, which is essential for tasks that require detailed scene understanding. In this project, we aim to compare the performance of different image segmentation models, including Convolutional Neural Networks (CNNs), Graph Convolutional Networks (GraphCNNs), and Transformers. Our goal is to evaluate their effectiveness and identify the strengths and weaknesses of each approach in the context of image segmentation. This literature survey was greatly inspired by [14].

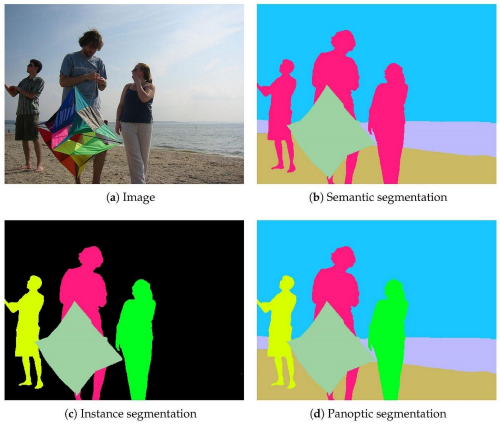

Figure 1. An Example showing the three types of segmentation tasks: Semantic Segmentation, Instance Segmentation, and Panoptic Segmentation. From [8]

.

1. Introduction

In Image Segmentation, there are three common types of segmentation tasks: *Semantic Segmentation*, *Instance Segmentation*, and *Panoptic Segmentation*. In Semantic Segmentation, the goal is to classify each pixel in the image into a predefined set of categories (such as ”car”, ”road”, ”tree”, ”person”, etc.). All pixels that belong to the same category are assigned the same label. Instance Segmentation identifies individual objects within the image and assigns a unique label to each object i.e., it distinguishes between different instances of the same category (such as two cars will be assigned different labels). Panoptic Segmentation combines both semantic and instance segmentation. It aims to provide a complete understanding of the scene by identifying both object instances and background areas. In this work, we focus on the task of Semantic Segmentation.

2. Literature Review

2.1. Convolutional Neural Network Architectures

Convolutional Neural Networks (CNNs) have been a fundamental approach in the field of image processing, particularly in the domain of image segmentation. This section reviews three prominent CNN architectures: Fully Convolutional Networks (FCN), UNet++, and DeepLabV3+, highlighting their innovations, strengths, weaknesses, and typical use cases.

2.1.1 Fully Convolutional Networks (FCN)

Introduced by *Long et al.* in 2015, the key innovation of FCN is its fully convolutional nature, which allows it to operate on input images of any size [11]. This architecture is simple and flexible, attributed to its use of convolutional layers that replace fully connected layers, enabling the network to efficiently learn spatial hierarchies for pixel-wise segmentation.

Strengths:

- Simple and flexible architecture.
- Efficient in learning spatial hierarchies.

Weaknesses:

- Produces coarse segmentation maps, lacking in fine detail.

Best Use Case:

- General segmentation tasks where detailed precision is less critical.

2.1.2 UNet++

UNet++ is a more advanced variant that introduces nested dense skip connections to improve the fine-grained segmentation capability, particularly beneficial in medical imaging [15]. The architecture includes several enhancements such as deep supervision and feature fusion, which allow for more precise and detailed segmentation outputs.

Key Features:
- Nested skip connections facilitate feature refinement across different network layers.
- Deep supervision at multiple decoder levels enhances the training dynamics.
- Feature fusion through decoder nodes integrates information from multiple preceding layers.

Advantages:
- Achieves fine-grained segmentation, crucial for medical applications.
- Manages computational costs effectively compared to more complex models.

Weakness:
- Slightly higher computational overhead than basic UNet due to additional connections.

Best Use Case:
- High-resolution medical image segmentation where detail is paramount.

2.1.3 DeepLabV3+

DeepLabV3+, developed by *Chen et al.*, incorporates atrous convolution and an atrous spatial pyramid pooling (ASPP) module to effectively capture multi-scale contextual information [5]. This design allows the network to handle complex scenes with varying object scales efficiently.

Strengths:
- Handles multiple scales effectively using ASPP.
- Suitable for complex scene segmentation with high variability in object size and structure.

Weaknesses:
- Computationally intensive, requiring significant resources.

Best Use Case:
- Complex scene segmentation such as urban landscapes or densely populated images.

2.2. UNet++

UNet++ is an enhanced structure of the traditional UNet model designed to achieve more accurate segmentation, especially in medical imaging applications. It introduces several innovative concepts:

Nested Skip Connections These connections are comprised of multiple intermediate nodes that refine and combine features from various network paths, enhancing the network's ability to propagate context and capture detailed structures within the image.

Deep Supervision UNet++ incorporates output layers attached at multiple decoder levels, facilitating the direct influence of the loss function across different depths of the network. This approach helps in the faster convergence of the network and improves the gradient flow, leading to enhanced learning capabilities.

Feature Fusion The decoder nodes in UNet++ integrate information from both preceding layers and skip connections. This fusion occurs via up-sampling and convolutions, allowing the network to preserve essential details and improve the quality of the segmentation output.

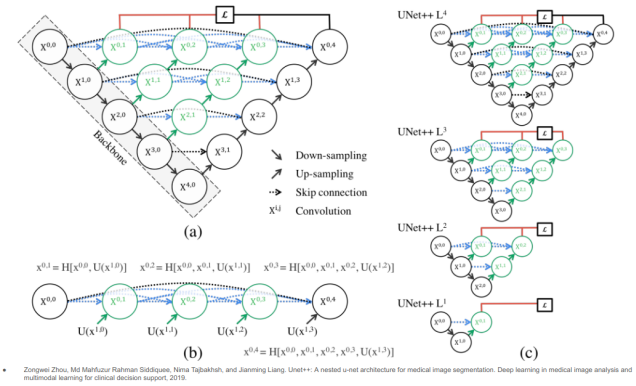

Figure 2. The architecture of UNet++ showing nested skip connections, deep supervision, and feature fusion mechanisms.

Mathematical Formulation The operations within the UNet++ can be described by the following equation:

$$F_{i,j} = H_{i,j}\left(F_{i-1,j} + \sum_{k=0}^{j-1} H_{i,k}\right)$$

where $F_{i,j}$ represents the feature map at the j^{th} node of the i^{th} layer, $H_{i,j}$ is the convolution operation at node j of layer i, and the summation aggregates the feature maps from all previous nodes at the same layer, enhancing the feature integration across the network.

Advantages
- UNet++ is capable of capturing detailed features such as water boundaries, which is critical for applications like flood area segmentation.

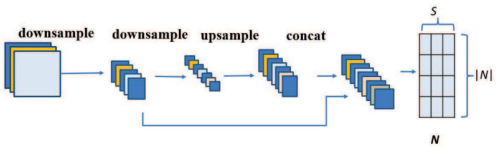

Figure 3. Graph-FCN node initialization. From [12].

- It performs well on smaller datasets, a common limitation in specialized fields such as medical imaging, where large annotated datasets may not be available.
- The computation cost, while higher than simpler architectures like FCN, remains manageable and less intensive compared to more complex models like DeepLabV3+.

Applications UNet++ is particularly suited for high-resolution image tasks where detailed feature delineation is critical. It has shown promising results in medical imaging, including tasks like tumor segmentation and organ delineation.

2.3. Graph Neural Networks: Graph-FCN and CNN-G

Deep learning has made significant advancements in semantic segmentation by classifying pixels in images. However, during high-level feature extraction, it often ignores the local spatial information, which is important for accurate segmentation. Graph-based models address this limitation by including the missing local context.

2.3.1 Graph-FCN

Graph-FCN combines graph convolutional networks (GCNs) with fully convolutional networks (FCNs) to capture local spatial relationships in images. Initially, a convolutional network converts the image grid data into a graph structure, transforming the semantic segmentation task into a graph node classification problem. The graph convolutional network is then applied to classify the nodes in this graph, effectively solving the segmentation challenge.

FCN-16s generates feature maps, and node annotations are initialized by combining upsampled feature vectors and node locations. Labels for nodes are obtained by pooling the raw label image during training. The process is shown in Figure 3.

In the graph model, edges are represented by an adjacency matrix, where each node is connected to its nearest l nodes. These connections allow node annotations to transfer through the edges in the graph neural network.

FCN-16s handles node classification and graph model initialization on a small feature map, while a 2-layer GCN classifies the nodes within the graph. Cross-entropy loss is computed for both outputs, and like FCN-16s, Graph-FCN is trained end-to-end. The output of the prediction process is the output of the convolutional network. The graph model is only used during the training.

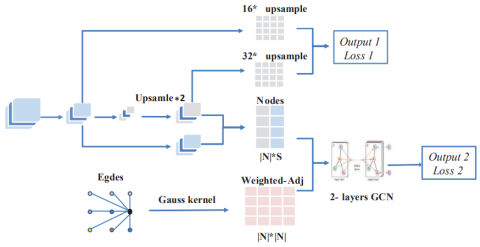

Figure 4. Graph-FCN model structure. From [12].

The structure of the Graph-FCN model is shown in Figure 4.

2.3.2 CNN-G

CNN-G builds on the Graph-FCN model by incorporating a graph-based approach with a convolutional neural network. Two types of structure models are used in CNN-G: distance-based and semantic-based, solved by GCN and GAT, respectively. The distance-based model captures the spatial relationships between nodes, while the semantic-based model focuses on the semantic relationships.

Using a graph attention network (GAT) enables flexible feature extraction across various receptive fields. This approach allows the model to integrate both structure learning and feature extraction.

Distance-Based Structure: Based on the assumption that the closer nodes are more correlated, the Gauss kernel function is used to generate the weighted edges.

The adjacent matrix $A = [e_{ij}]_{|N|\times|N|}$ is used to represent the edge set:

$$e_{ij} = \begin{cases} \exp\left(-\frac{\|p_i - p_j\|^2}{\sigma^2}\right), & \text{an edge between } n_i \text{ and } n_j \\ 0, & \text{otherwise} \end{cases}.$$

To simplify the calculation, we make the nodes connect to the l closest nodes.

Semantic-based model: The initial attention coefficient c_{ij} is used to measure the correlation between two nodes. The features of nodes with the same category will be more similar, the attention coefficient between each other will be larger, and the similarity is gradually strengthened in the iteration. A linear transformation is taken to map the concatenation of two nodes' features into a real number:

$$c_{ij} = a\left(n_i' || n_j'\right)$$

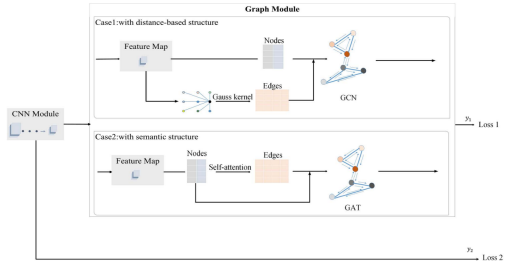

Figure 5. CNN-G model structure. From [13].

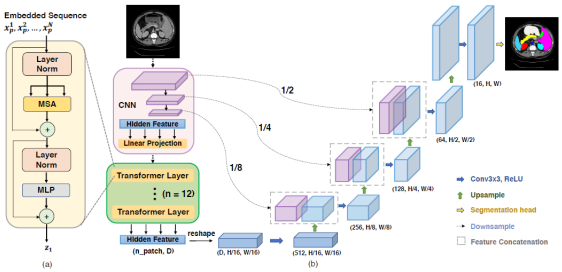

Figure 6. TransUnet architecture. From [4].

The final attention coefficient is obtained through the normalization of all neighborhood nodes:

$$e_{ij} = \frac{\exp(c_{ij})}{\sum_{k \in neib_i} \exp(c_{ik})}.$$

When the graph structure is unknown, the matrix $A_{\text{att}} = [e_{ij}]_{|N| \times |N|}$ can be taken as the adjacency matrix. In this case, the edge set is generated by the calculation of the attention coefficients.

In each case, the model generates two outputs, $y1$ and $y2$, which corresponds to two losses, $loss1$ and $loss2$. Both share the convolutional layer's extracted features. The final prediction is based on $y1$. Similar to Graph-FCN, the graph model is only used during training.

The structure of the CNN-G model is shown in Figure 5.

2.4. Transformer

2.4.1 TransUnet

TransUNet [4] has emerged as a robust framework for medical image segmentation by combining the strengths of U-Net and Transformer architectures. Refer the figure 6 for the architecture diagram.

TransUNet addresses these limitations by introducing a hybrid architecture that leverages both CNNs and Transformers. The encoder consists of a CNN followed by a Transformer module. The CNN is used to extract feature maps from the input images, which are then tokenized into image patches and fed into the Transformer. This CNN-Transformer hybrid allows TransUNet to capture detailed spatial features (via CNN) and global context (via the Transformer) simultaneously.

In the decoder, a cascaded upsampling module (CUP) is employed, consisting of multiple upsampling blocks with convolutional layers and ReLU activation. These upsampled features are combined with the high-resolution CNN feature maps from the encoder through skip connections, similar to the U-Net structure, enabling precise localization. This U-Net-like design ensures that TransUNet can recover lost spatial detail while preserving the high-level semantic understanding captured by the Transformer.

2.4.2 MaskFormer

Image segmentation models which can solve all 3 tasks (instance, semantic and panoptic segmentation) with a unified architecture started to appear in the literature in the last few years. This started with DETR [3] and later improved by MaskFormer [6] for semantic segmentation.

Mask2Former [7], which adopts the same meta architecture as MaskFormer, with a backbone, a pixel decoder and a Transformer decoder, extends this to instance segmentation by further improving the neural network architecture in the following ways:

- **Masked Attention**: They use *masked attention* (rather than the standard cross-attention) in the Transformer decoder which restricts the attention to localized features centered around predicted segments, which can be either objects or regions depending on the specific semantic for grouping.
- **Multi-scale high-resolution features**: High-resolution features improve model performance, especially for small objects. However, this is computationally demanding. Therefore, instead of always using the high-resolution feature map, they utilize a feature pyramid which consists of both low and high-resolution features and feed one resolution of the multi-scale feature to one Transformer decoder layer at a time.

It also made improvements in optimization and training strategies. The architecture diagram is given in Figure 7.

3. Experiments

We conducted an experiment to compare MaskFormer and Mask2Former by fine-tuning them on the Flood-Segmentation dataset [9] **for semantic segmentation**. This dataset consists of 280 RGB images of flood-affected areas, with corresponding ground-truth binary masks (1 stands for flood, 0 for non-flood). Out of the 280 images, we used 200 for training, 20 for validation, and 60 for testing. Out of the 280 images, we used 200 for training, 20 for validation, and 60 for testing.

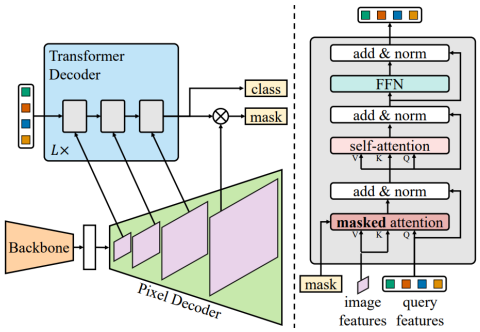

Figure 7. Mask2Former architecture. From [7].

3.1. MaskFormer

To make the fine-tuning more robust, we used the following data augmentation techniques on the training set:

- `Resize`: Randomly resized the images to a size 768×768 (ignoring the channel dimension, which is 3).
- `RandomCrop`: Randomly cropped the images to a size 512×512.
- `HorizontalFlip`: Randomly flipped the images horizontally with a probability of 0.5.
- `Normalize`: Normalized each channel of the images with the mean and standard deviation of the pixel values of training set.

Note that the spatial transformations (all except `Normalize`) are applied to the corresponding binary mask.

We used the *facebook/maskformer-swin-base-ade* and *facebook/mask2former-swin-base-ade-semantic* models from Hugging Face's model hub for MaskFormer and Mask2Former, respectively.

Fine-tuning

Note: Since the number of classes were different in the pre-trained model, the classification head was randomly initialized for both the models.

We fine-tuned both of the models for 10 epochs with a batch size of 4. We used the Adam optimizer [10] with a learning rate of 5×10^{-5} for MaskFormer and 9×10^{-5} for Mask2Former. MaskFormer took around 45 minutes while Mask2Former took around 1 hour to fine-tune on a single Tesla P100-PCIE-16GB GPU.

3.2. TransUnet

Transforms perfomed on the training data:

- Normalization: Normalized each channel of the images with mean and standard deviation of the training set.
- Resize: Resized the images to 256×256, filling the rest with zeros, keeping the aspect ratio intact.

Training

- **Optimizer**: SGD optimizer with a learning rate of 0.01, momentum of 0.9, and weight decay of 0.0001.
- **Loss function**: A combination of Dice loss and cross-entropy loss, with a weight of 0.5 for each.
- **Epochs**: 150
- **Batch size**: 4

On training the TransUnet model, we achieved a mean IoU of 0.9671 and dice score of 0.98367 on the validation set, and a mean IoU of 0.96789 and dice score of 0.98326 on the test set. The model was trained on a single single Tesla P100-16GB GPU. The code is referred from the official implementation of TransUnet [2].

3.3. Graph-FCN & CNN-G

Uses DeepLabV3 [1] model as the FCN backbone

Training Details

- **Epochs**: 60
- **Batch Size**: 4
- **Learning Rate**:
  - Graph-FCN: 1e-4
  - CNNG: 2e-5
- **Optimizer**: Adam
- **Loss**: Cross Entropy

MIoU Results

- Graph-FCN: 83.45
- CNNG: 84.33

3.4. Experimental Setup for UNet++

Training Details The UNet++ model was configured and trained with the following parameters:

- **Epochs:** 10
- **Learning Rate:** 0.0001
- **Batch Size:** 4
- **Optimizer:** Adam with a learning rate of 1×10^{-4}

Loss Function The loss function used was a combination of Binary Cross-Entropy and Dice Coefficient, formulated as:

$$\mathcal{L}(Y, \hat{Y}) = -\frac{1}{N} \sum_{b=1}^{N} \left(\frac{1}{2} Y_b \cdot \log \hat{Y}_b + \frac{2 \cdot Y_b \cdot \hat{Y}_b}{Y_b + \hat{Y}_b} \right)$$

This composite loss function is designed to optimize the model by addressing both class imbalance and the need for precise boundary delineation.

Results The UNet++ model achieved the following performance metrics:

- **Best Validation Intersection over Union (IoU):** 0.8291
- **Test IoU:** 0.8032

The results demonstrate the effectiveness of UNet++ in segmenting images with high precision, particularly under the constraint of relatively small datasets.

4. Results

4.1. Evaluation Metrics

IoU Score

Intersection over Union (IoU) is defined as the ratio of the area of intersection between the predicted segmentation map A and the ground truth map B to the area of their union.

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|}.$$

4.2. Quantitative Comparision

Model	mIoU(Val Set)	mIoU(Test Set)
UNet++	80.32	82.91
Graph-FCN	83.94	83.45
CNNG	82.93	84.33
MaskFormer	88.85	86.11
Mask2Former	88.70	86.16
TransUnet	96.71	96.89

Table 1. Comparison of mIoU scores on validation and test sets

Model	Number of Parameters
UNet++	260.8M
Graph-FCN	609.91M
CNNG	609.92M
TransUnet	105.3M
MaskFormer	101.8M
Mask2Former	106.8M

Table 2. Comparison of number of parameters

4.3. Qualitative Comparision

The qualitative comparison of segmentation results is shown in Figures 8 and 9.

5. Conclusion

- TransUnet performs the best in terms of `mIoU` on the validation and test sets.
- Even though Mask2Former claim to be better than MaskFormer, it shows only marginal improvement in terms of `mIoU` on this dataset.

Figure 8. Qualitative comparison of segmentation results.

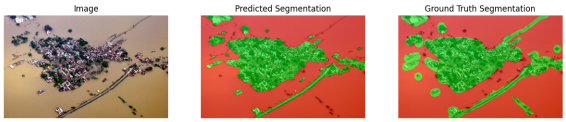

Figure 9. Qualitative comparison of segmentation results.

- Mask2Former (occupies 412M space on disk) has more parameters than MaskFormer (occupies 393M space on disk) which makes it slightly slower for fine-tuning and inference.
- CNNG performs better than Graph-FCN which shows the effectiveness of using the GAT network.

6. Code Availability

The code for the experiments conducted in this study is available at `https://github.com/darpan-gaur/dlProject`.

References

[1] Deeplabv3. Link. 5

[2] Transunet: Transformers make strong encoders for medical image segmentation. Github repository Link. 5

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. 4

[4] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation, 2021. 4

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 801–818, 2018. 2

[6] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation, 2021. 4

[7] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation, 2022. 4, 5

[8] Abhishek Jain. Semantic vs instance vs panoptic segmentation on medium Link. 1

[9] Md Faizal Karim, Krish Sharma, and Niyar R Barman. Flood area segmentation, 2022. 4

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 1

[12] Yi Lu, Yaran Chen, Dongbin Zhao, and Jianxin Chen. Graph-fcn for image semantic segmentation, 2020. 3

[13] Yi Lu, Yaran Chen, Dongbin Zhao, Bao Liu, Zhichao Lai, and Jianxin Chen. Cnn-g: Convolutional neural network combined with graph for image segmentation with theoretical analysis. *IEEE Transactions on Cognitive and Developmental Systems*, 13(3):631–644, 2021. 4

[14] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):3523–3542, 2022. 1

[15] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. *Deep learning in medical image analysis and multimodal learning for clinical decision support*, 2019. 2