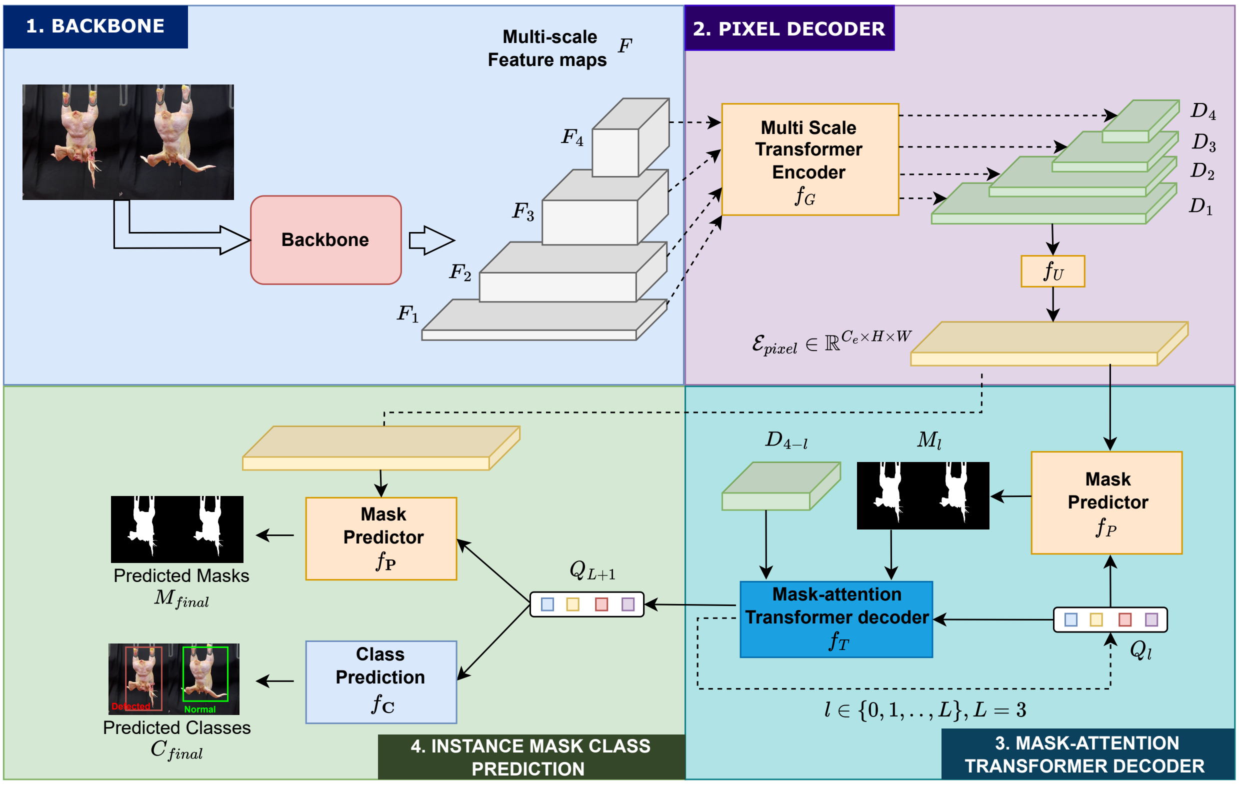
Graphical Abstract

CarcassFormer: An End-to-end Transformer-based Framework for Simultaneous Localization, Segmentation and Classification of Poultry Carcass Defect

Minh Tran†, Sang Truong† [1], Arthur F. A. Fernandes, Michael T. Kidd, Ngan Le ‡ [2]


1. BACKBONE
Multi-scale Feature maps F
Backbone
F_4
F_3
F_2
F_1
2. PIXEL DECODER
Multi Scale Transformer Encoder f_G
D_4
D_3
D_2
D_1
f_U
$\mathcal{E}_{pixel} \in \mathbb{R}^{C_e \times H \times W}$
D_{4-l}
M_l
Mask Predictor f_P
Mask-attention Transformer decoder f_T
Q_l
$l \in \{0, 1, .., L\}, L = 3$
3. MASK-ATTENTION TRANSFORMER DECODER
Q_{L+1}
Mask Predictor f_P
Class Prediction f_C
Predicted Masks M_{final}
Detected
Normal
Predicted Classes C_{final}
4. INSTANCE MASK CLASS PREDICTION

[1]† indicates the same contribution
[2]‡ indicates corresponding author

Highlights

CarcassFormer: An End-to-end Transformer-based Framework for Simultaneous Localization, Segmentation and Classification of Poultry Carcass Defect

Minh Tran†, Sang Truong† [3], Arthur F. A. Fernandes, Michael T. Kidd, Ngan Le ‡ [4]

- **Dataset**: A poultry carcass dataset was acquired, comprising a total of 7,321 images gathered from real-world environments and collected from diverse chicken ages, chicken size, and number of chickens per frame. The dataset has been carefully annotated by three experts.

- **Methodology**: CarcassFormer, an effective end-to-end Transformer-based framework, was proposed for simultaneously localizing poultry carcass regions, segmenting carcass areas, and determining carcasses with imperfections. CarcassFormer is based on Transformer-based Unet architecture.

  Our CarcassFormer is designed with four different components: Network Backbone to extract visual features, Pixel Decoder to utilize feature maps from various scales, Mask-Attention Transformer Decoder to predict the segmented masks of all instances, and Instance Mask and Class Prediction to provide segmentation mask and corresponding label of an individual instance. The extensive experiments showed that CarcassFormer outperforms both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019), and Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021) on different backbone networks of ResNet-34 and ResNet-50 on various metrics of AP, AP@50, AP@75.

- **Pre-trained models and Code**: The pre-trained model and source code of CarcassFormer is available for research purposes at: `https://github.com/UARK-AICV/CarcassFormer`.

[3] † indicates the same contribution

[4] ‡ indicates corresponding author

CarcassFormer: An End-to-end Transformer-based Framework for Simultaneous Localization, Segmentation and Classification of Poultry Carcass Defect

Minh Tran[†a], Sang Truong[† 1a], Arthur F. A. Fernandes[b], Michael T. Kidd[c], Ngan Le [‡ 2a]

[a]*Department of Computer Science and Computer Engineering, 1 University of Arkansas, Fayetteville, 72701, Arkansas, USA*
[b]*Cobb Vantress, Inc, 4703 US HWY 412 E, Siloam Springs, 72761, Arkansas, USA*
[c]*Department of Poultry Science, 1260 W. Maple, POSC O-114, Fayetteville, 72701, Arkansas, USA*

Abstract

In the food industry, assessing the quality of poultry carcasses during processing is a crucial step. This study proposes an effective approach for automating the assessment of carcass quality without requiring skilled labor or inspector involvement. The proposed system is based on machine learning (ML) and computer vision (CV) techniques, enabling automated defect detection and carcass quality assessment. To this end, an end-to-end framework called CarcassFormer is introduced. It is built upon a Transformer-based architecture designed to effectively extract visual representations while simultaneously detecting, segmenting, and classifying poultry carcass defects. Our proposed framework is capable of analyzing imperfections resulting from production and transport welfare issues, as well as processing plant stunner, scalder, picker, and other equipment malfunctions.

To benchmark the framework, a dataset of 7,321 images was initially acquired, which contained both single and multiple carcasses per image. In this study, the performance of the CarcassFormer system is compared with other state-of-the-art (SOTA) approaches for both classification, detection, and segmentation tasks. Through extensive quantitative experiments, our framework consistently outperforms existing methods, demonstrating remarkable improvements across various evaluation metrics such as AP, AP@50, and AP@75. Furthermore, the qualitative results highlight the strengths of CarcassFormer in captur-

[1†] indicates the same contribution
[2‡] indicates corresponding author

ing fine details, including feathers, and accurately localizing and segmenting carcasses with high precision. To facilitate further research and collaboration, the source code and trained models will be made publicly available upon acceptance.

Keywords: Carcass Defect, Detection, Segmentation, Classification, Defect Automation

1. Introduction

Increased consumption of poultry products will be a certainty for global food security achievement in the upcoming 30 years based on the efficiency of the utilization of poultry, as well as diverse consumer acceptance. The Food and Agriculture Organization of the United Nations 2005/2007 has projected that production of poultry will increase more than 100 percent by the year 2050 with an increased tonnage of poultry products, primarily broiler chickens, surpassing 180 million tons, with current projection estimated at just over 80 million tons Alexandratos and Bruinsma (2012). Numerous studies have demonstrated increasing annual poultry consummation rates, mainly due to relatively inexpensive price, nutritional value, and health benefits Elam (2022). In the U.S., broiler chicken efficiency of feed utilization has increased 7 percent from 2021 to the present at a similar slaughter age between 47 and 48 days across the decade Council (March 18, 2021). With annualized increases in broiler production, concomitant increases in labor are necessary for meat production supply chain efficiency. In addition to the costs of increased workforce labor and workforce development, many poultry companies are suffering from labor shortages Wu et al. (2022) Kaminski (2020). Another negative side of relying on people for the process of poultry processing represents the varying results of carcass evaluation consistency. Thus, many companies use assembly lines stationed by employees to inspect the quality of chicken carcasses, which leaves room for human error and can result in miscategorized carcass defections. As a result, numerous agriculture industries, including poultry production facilities and poultry processing plant factories, are researching and investing in automated robotic technologies to improve processing and labor wellbeing, as well as profit Ahlin (2022) Ren et al. (2020) Park et al. (2022). Further, there are numerous automation technologies offering noticeable economic benefits to agricultural production as of late Jin et al. (2021).

In the era of precision agriculture, Machine Learning (ML) and Computer Vision (CV) have emerged as high-performance computing technologies that are creating new opportunities to improve broiler management, production, and identification of processing defects with non-invasive low-cost techniques Aydin (2017) Caldas-Cueva et al. (2021). In this study, the focus was on utilizing modern ML&CV, i.e. Deep Learning, to analyze chicken carcasses after scalding, picking, and removal of head and feet in processing plants. Visual inspection is one of the most basic but also most important steps in controlling meat quality before the product is prepared, packaged, and distributed to the market. The image processing and classification within the poultry processing plants can optimize such systems, in addition to heightening food safety. Hence, our proposed intelligent and automated system will analyze and improve poultry processing concomitantly with increased data acquisition. Our computer vision system functions as an automated detection model capable of classifying defects and contaminated carcasses. While detection, segmentation, and classification are widespread tasks in computer vision Dong et al. (2021); Zhou et al. (2021); Le et al. (2022), they have focused on various tasks such as autonomous driving Le et al. (2017c,b); Janai et al. (2020); Tong et al. (2020); Truong et al. (2022); Nguyen et al. (2022), surveillance Wray et al. (2021); Gabeur et al. (2020); Yamazaki et al. (2022); Vo et al. (2022), biometrics Le and Savvides (2016); Le et al. (2017a); Duong et al. (2019b,a); Quach et al. (2022), and medical imaging Han et al. (2017); Le et al. (2018); Tran et al. (2022c,a); Le et al. (2023); Thang Pham et al. (2023); Nguyen et al. (2023), amodal understanding Tran et al. (2022b) which mainly target humans, car, objects, face, human organs. *None of them target analyzing poultry carcass condemnations defects.* One of the main reasons is the lack of publicly available data.

In the context of poultry carcass analysis, distinguishing between single and multiple carcasses in an image is a crucial step for accurate quality assessment. To achieve this, the problem was approached as an instance segmentation task, involving the localization of individual instances. Additionally, mask classification was performed to determine whether a single poultry carcass was defective or not. While per-pixel classification (e.g FCN Long et al. (2015), Unet-based approaches Ronneberger et al. (2015); Zhou et al. (2018); Ibtehaz and Rahman (2020); Le et al. (2021); Tran et al. (2022c)) applies a classification loss to each output pixel and partitions an image into regions of different classes, mask classification (e.g Mask-RCNN He et al. (2017), DETR Zhu et al. (2020)) predicts a set of binary masks, each associated with a single class prediction. In recent years, there has been a significant growth in the adoption of Transformer architecture Vaswani et al. (2017) for semantic segmentation tasks. This trend is underscored by numerous approaches that have leveraged Transformer models, demonstrating state-of-the-art performance in the field. Notable examples include DETR Carion et al. (2020), SegFormer Xie et al. (2021), Mask2Former Cheng et al. (2022), FASeg He et al. (2023a), and Mask DINO Li et al. (2023). In this paper, the question of how to *simultaneously handle both mask classification and pixel-level classification* is addressed.

To address the aforementioned question, we particularly leverage the recent Transformer technique Vaswani et al. (2017) and propose *CarcassFormer*, which aims to simultaneously localize poultry carcasses from moving shackles, segment the poultry carcass body, and classify defects or contaminated carcasses. To develop CarcassFormer, an experiment was set up at the University of Arkansas-Agricultural Experiment Station Pilot Processing Plant on the poultry research farm by placing a camera adjacent to the shackles of carcasses moving along a processing line. Each poultry carcass in the view of the camera will be analyzed by localizing with a bounding box, segmenting the boundary, and classifying to determine its imperfections. Any unapproved birds are then reworked. Notably, a bird is considered to be defective if it has one of the following issues: feathers, un-clean/dirty, skin peel, broken wings, or broken legs. The annotation requirement is following instructions provided by USAD USDA.

Our contribution is three-fold as follows:

- **Dataset**: A dataset containing a total of 7,321 images of poultry carcasses on a Pilot processing plant. The images in this diverse dataset contain real-world examples of chickens of a range of ages, sizes, and numbers of chickens per frame. The dataset has been carefully annotated by three experts.

- **Methodology**: We propose CarcassFormer, an effective end-to-end Transformer-based framework for simultaneously localizing poultry carcass regions, segmenting carcass areas, and determining carcasses with imperfections. CarcassFormer is based on Transformer-based Unet architecture.

- **Pre-trained models and Code**: We will release our pre-trained model and source code of CarcassFormer for research purposes.

1.1. Related Work

1.1.1. Image Segmentation

Image segmentation is a critical computer vision task that involves dividing an image into different regions based on visual features. This process can be accomplished through either *semantic segmentation* or *instance segmentation*. Semantic segmentation categorizes pixels into multiple classes, e.g. foreground and background, but does not differentiate between different object instances of the same class. Popular semantic segmentation models include the Fully Convolutional Network (FCN) Long et al. (2015) and its variants, such as the U-Net family Ronneberger et al. (2015); Zhou et al. (2018); Ibtehaz and Rahman (2020); Le et al. (2021), as well as the Pyramid Scene Parsing Network (PSPNet) Zhao et al. (2017) and DeepLabV3 Chen et al. (2018).

In contrast, instance segmentation aims to detect and segment individual objects by providing a unique segmentation mask for each object. There are two types of instance segmentation approaches: two-stage and one-stage methods. Two-stage approaches, such as top-down Cai and Vasconcelos (2018); Chen et al. (2019); Cheng et al. (2020) and bottom-up methods Arnab and Torr (2016); Chen et al. (2017); Newell et al. (2017), detect bounding boxes first and then perform segmentation within each region of interest. On the other hand, one-stage approaches, such as anchor-based methods Li et al. (2017); Bolya et al. (2019) and anchor-free methods Ying et al. (2019); Chen et al. (2020); Lee and Park (2020), perform both detection and segmentation simultaneously, resulting in less time consumption. Anchor-based one-stage approaches generate class-agnostic candidate masks on candidate regions and extract instances from a semantic branch. However, these approaches rely heavily on predefined anchors, which are sensitive to hyper-parameters. To address this issue, anchor-free one-stage methods eliminate anchor boxes and use corner/center points instead. Moreover, based on their feature backbone and learning mechanism, various approaches to instance segmentation can be categorized into either Convolution Neural Network (CNN)-based or Transformer-based approaches as follows.

1.1.2. CNN-based instance segmentation

The idea of "detect then segment" has dominated in instance segmentation task, which is a two-stage method. In particular, Mask R-CNN He et al. (2017) is the most representative work. Based on the priority of detection and segmentation, there are two groups in this category: top-down methods and bottom-up methods. The former first predicts a bounding box for each object and then generates an instance mask within each bounding box He et al. (2017); O Pinheiro et al. (2015). On the other hand, the latter associates pixel-level projection with each object instance and adopts a post-processing procedure to distinguish each instance Arnab and Torr (2016); Kong and Fowlkes (2018). While the top-down methods mainly rely on the detection results and are prone to systematic artifacts on an overlapping instance, the bottom-up methods depend on the performances of post-processing and tend to suffer from under-segment or over-segment problems Fathi et al. (2017). With a large amount of pixel-wise mask annotations, fully-supervised learning instance segmentation methods have achieved great performance. However, pixel-wise mask annotating is labor intensive (e.g., 22 hours to label 1000 segmentation masks Lin et al. (2014)). Thus, weakly-supervised Zhou (2018); Zhu et al. (2016) and semi-supervised Van Engelen and Hoos (2020) have been proposed. CNN-based image segmentation has been outreached in multiple Computer Vision tasks including amodal segmentation Li and Malik (2016), salient detection Fan et al. (2019), human segmentation Zhang et al. (2019), soft biometrics Luu et al. (2016). CNN-based instance segmentation survey can be found at Hafiz and Bhat (2020); Gu et al. (2022).

1.1.3. Transformer in Computer Vision

Transformer was first introduced by Vaswani et al. (2017) for language translation and obtained State-Of-The-Art (SOTA) results in many other language processing tasks. Recently, many models Carion et al. (2020), Liu et al. (2022), Li et al. (2022) successfully applied the Transformer concept to computer vision and achieved promising performance. The core idea behind transformer architecture Vaswani et al. (2017) is the self-attention mechanism to capture long-range relationships. It has obtained state-of-the-art in many Natural Language Processing (NLP) tasks. Besides, Transformers have worked well suited for parallelization, facilitating training on large datasets Transformer has been successfully applied to enrich global information in various tasks in Computer Vision such as image recognition Dosovitskiy et al. (2020); Touvron et al. (2021) object detection Carion et al. (2020); Zhu et al. (2020); Sun et al. (2021), image segmentation Ye et al. (2019); Zheng et al. (2021); Tran et al. (2022b), action localization Vo et al. (2021, 2022), video captioning Yamazaki et al. (2022, 2023). DETR Zhu et al. (2020) is the first model that uses Transformer as an end-to-end and query-based object detector, with bipartite-matching loss and set prediction objective. Inspired by Zhu et al. (2020); Cheng et al. (2021), which are end-to-end prediction objectives and successfully address multiple tasks without modifying the architecture, loss, or the training procedure, the merits of Transformer were inherited and CarcassFormer was proposed. Our network is an end-to-end Transformer-based framework and simultaneously tackles both segmentation and classification tasks.

Transformer-based networks have also found application in addressing detection and segmentation challenges within poultry science. Lin et al. (2022) proposes a vision transformer model to screen the breeding performance of roosters by analyzing correlations between cockscomb characteristics and semen quality, aiming to overcome the time-consuming and error-prone nature of human-based screening. Hu et al. (2023) improves pig segmentation in farming environments using a grouped transformer attention module with Mask R-CNN networks and data augmentation. Zhao et al. (2023) proposes a real-time mutton multi-part classification and detection method using Swin-Transformer. He et al. (2023b) presents Residual-Transformer-Fine-Grained (ResTFG), a model merging transformer and CNN for precise classification of seven chicken Eimeria species from microscopic images.

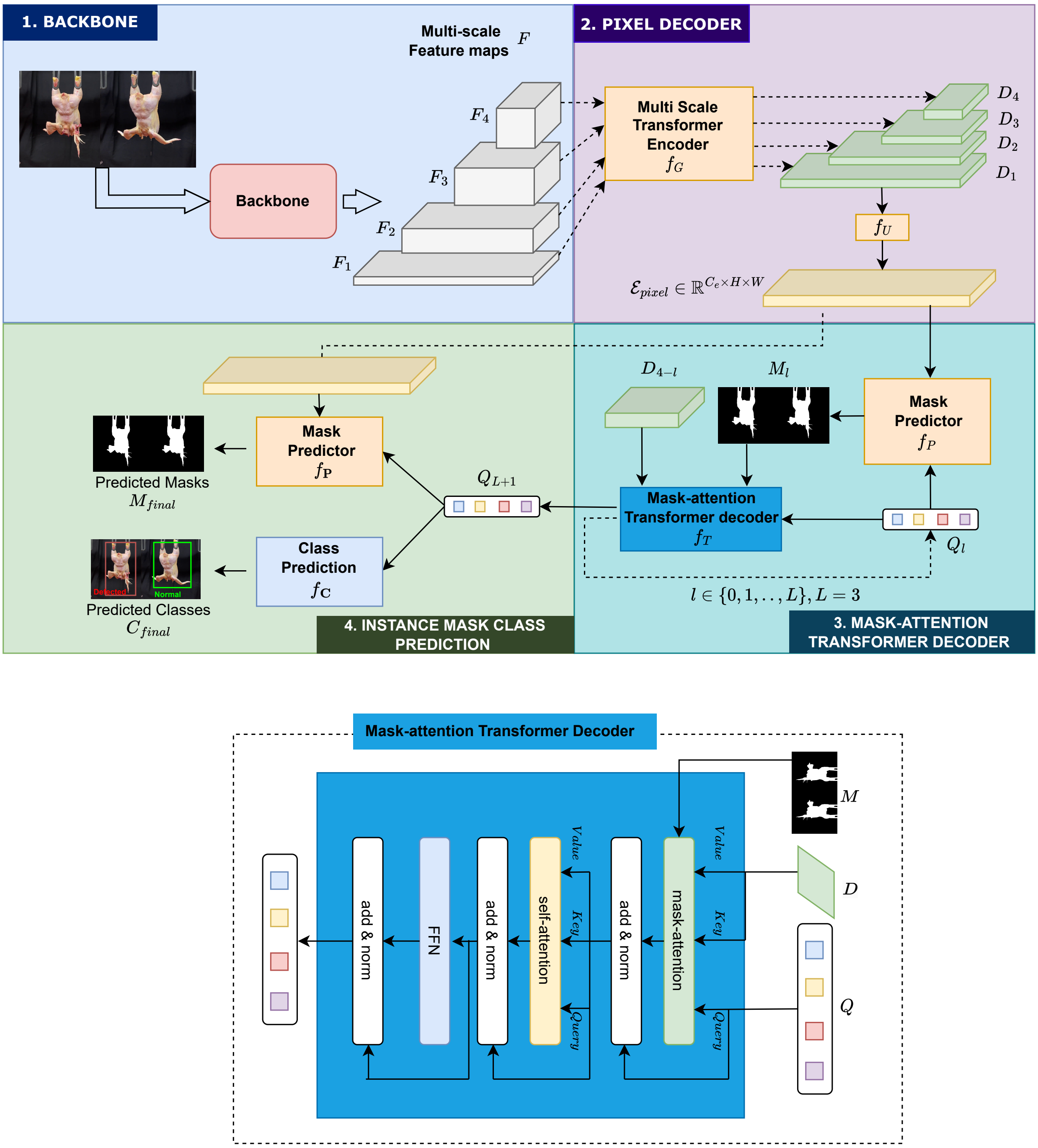

1. BACKBONE
Multi-scale Feature maps F
F_4
F_3
F_2
F_1
Backbone
2. PIXEL DECODER
Multi Scale Transformer Encoder f_G
D_4
D_3
D_2
D_1
f_U
$\mathcal{E}_{pixel} \in \mathbb{R}^{C_e \times H \times W}$
D_{4-l}
M_l
Mask Predictor f_P
Mask-attention Transformer decoder f_T
Q_l
$l \in \{0, 1, .., L\}, L = 3$
3. MASK-ATTENTION TRANSFORMER DECODER
Q_{L+1}
Mask Predictor f_P
Predicted Masks M_{final}
Class Prediction f_C
Predicted Classes C_{final}
Defected
Normal
4. INSTANCE MASK CLASS PREDICTION
Mask-attention Transformer Decoder
add & norm
FFN
add & norm
self-attention
Value
Key
Query
add & norm
mask-attention
Value
Key
Query
M
D
Q

Figure 1: Top: Overall flowchart of our proposed CarcassFormer consisting of four components: 1. Network Backbone; 2. Pixel Decoder; 3. Mask-Attention Transformer Decoder; 4. Instance Mask Class Prediction. Bottom: Details of third component Mask-Attention Transformer Decoder.

2. Materials and Methods

2.1. Data Collection

The data was collected at the University of Arkansas pilot processing plant (Fayetteville, AR). Multiple broiler chicken products at different ages were processed using standard commercial practices and following rigorous animal-handling procedures that are in compliance with federal and institutional regulations regarding proper animal care practices FASS (2010). The video-capturing system was set up in the area after feather picking and before chilling and evisceration. We decided on his system placement so that three common kinds of defects that can occur during normal processing could be evaluated, namely tearing of the skin, presence of feathers, and broken/disjointed bones.

To obtain the dataset named CarcassDefect, a camera was set up in front of the shackle line, whereas a black curtain was hung behind the shackle. Videos were recorded at 10 frames per second. The camera setup can be visualized in Fig. 2 and Fig. 3. In the end, a total of 7,321 images were collected, comprising 4,279 single carcass images and 3,042 multiple carcass images. Fig.4 illustrates some images from our CarcassDefect dataset, which comprised a large diversity of carcass quality such as resolution (small carcass, large carcass), the number of carcass per image (a single carcass per image, multiple carcasses per image), various defect (carcass defect can be the one with tearing of skin, feathers, broken/disjointed bones.), etc.

2.2. Data Annotation

Upon acquiring the video data, the next crucial step is to annotate the images extracted from the footage to generate training data for detection, segmentation, and classification tasks. Annotation involves labeling each frame with bounding boxes for detection, masks for segmentation, and labels for classification. This process enables the computer vision system to learn from the annotated data, which enhances its ability to perform these tasks accurately and efficiently. The data annotation process is illustrated in Fig.5. The annotated data is saved in a JSON file and follow the COCO format Lin et al. (2014) as demonstrated in Fig.6. In this COCO format, the data is described as follows:

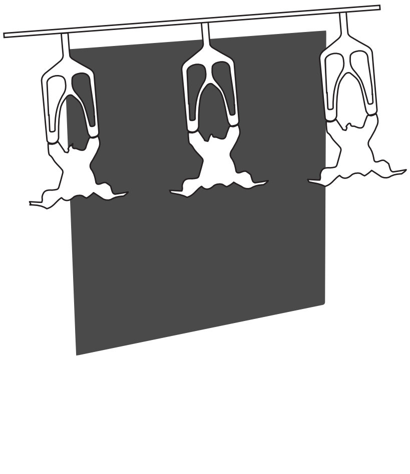


Figure 2: **Camera setup for data collection.** A black curtain is hung behind the shackle to provide a certain contrast to the carcasses. A camera is placed to capture the carcasses within the black curtain.

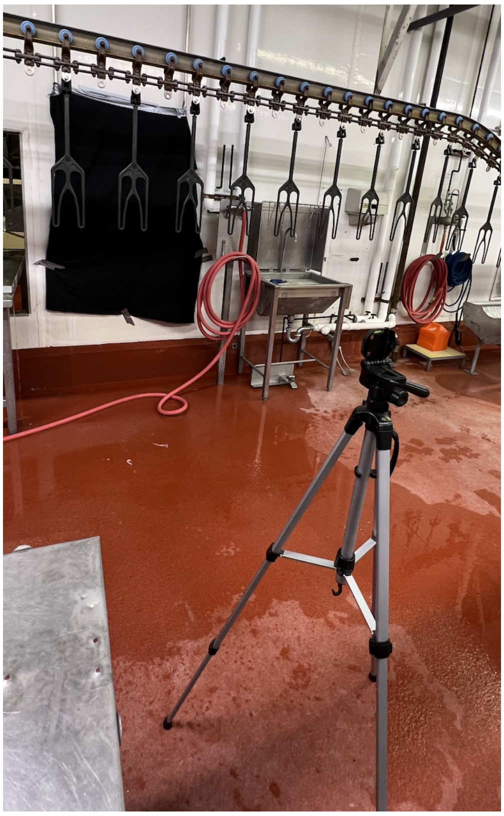


Figure 3: An overview image of the shooting location. The black curtain is hung on the wall behind the shackle.

Table 1: The distribution of the **images** in CarcassDefect Dataset in regard to the normal carcass and defective carcass at both a single carcass per frame and multiple carcasses per frame.

	Single carcass per image	**Multiple carcasses per image**	**Total**
Trainset	3,017	2,115	5,132
Valset	754	535	1,289
Testset	508	392	900
Total	4,279	3,042	7,321

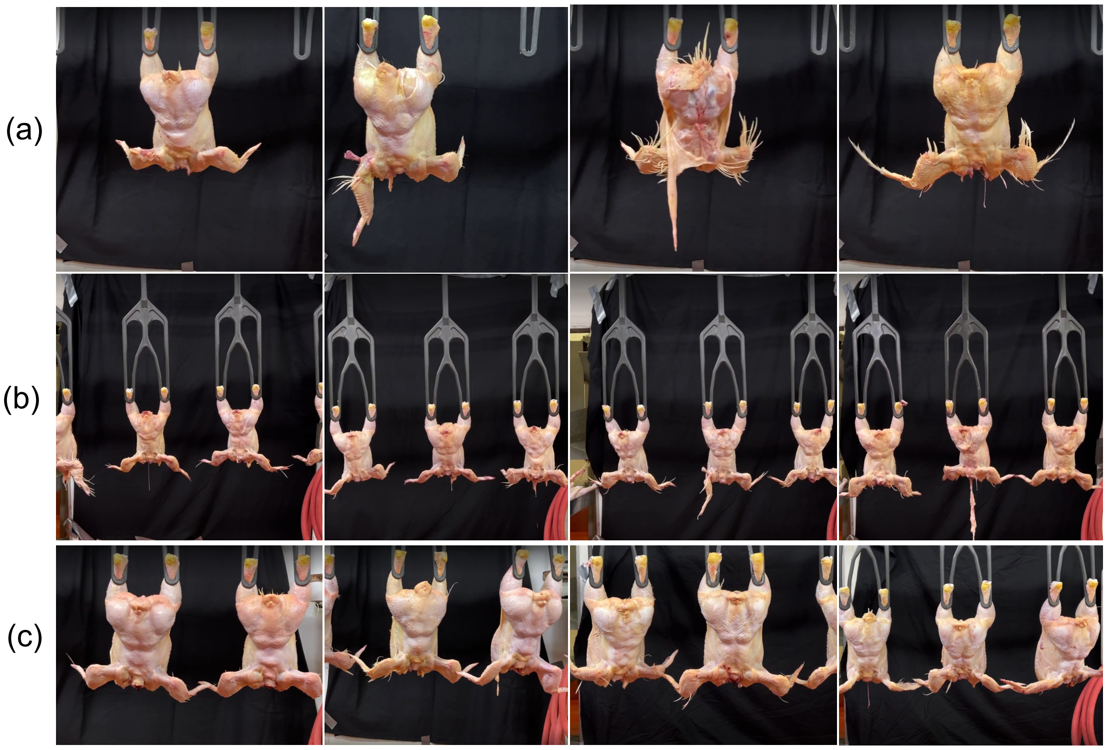


Figure 4: Illustrations of **data collected**, which comprises (a) single carcass/instance per image/frame; (b) multiple carcass/instance per image/frame; (c)carcass/instance at different scale/resolution. The carcass is processed with various defects such as tearing of skin, feathers, broken/disjointed bones.

- *categories*: defined as 'normal' and 'defect' presenting labeled in the dataset. The defect class is determined for a carcass that has either 'feather' or 'broken wings' 'broken legs' or 'peeled skin'.

- *images*: frames extracted from recorded videos. *images* is a list of objects with metadata information about the image. An object includes the following keys:

  - *id*: a unique identifier that differentiates each image within a list. It can be defined as the file name.
  - *file_name*: the name of the file. In the example (Fig.6).
  - *width*: the image height such as 950 pixels.
  - *height*: the image height such as 960 pixels.

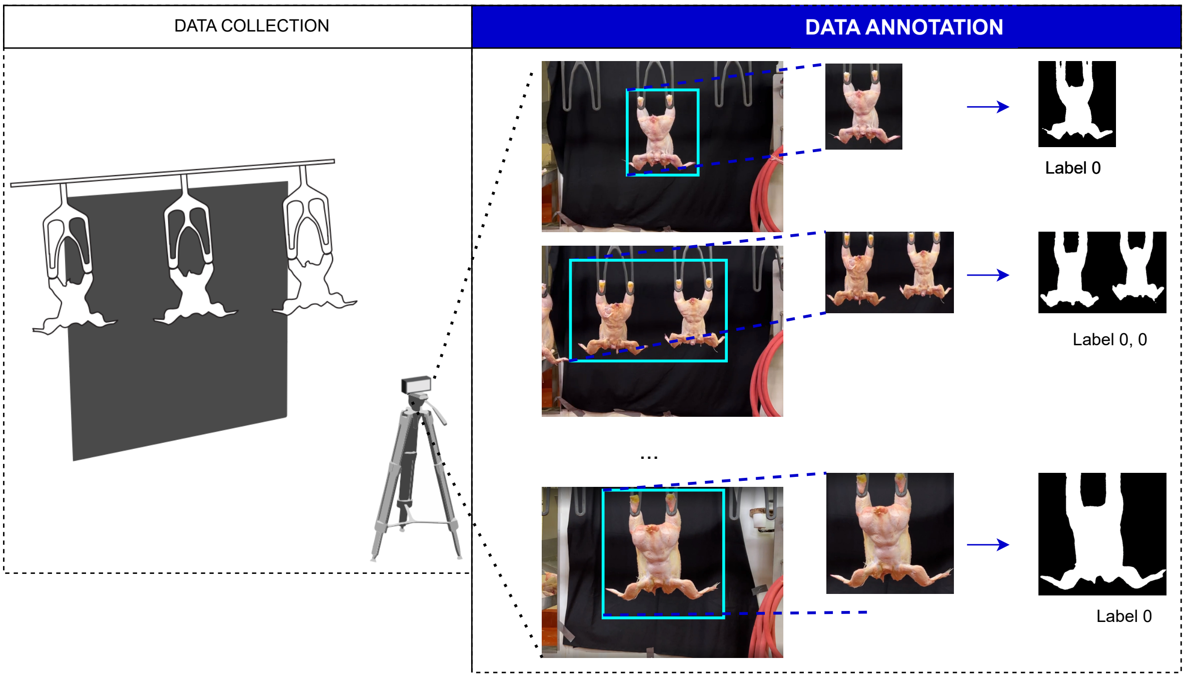


Figure 5: An illustration of **Data Annotation Process**. Each frame from the recorded video is annotated with bounding boxes for detection, masks for segmentation, and defect labels for classification.

  - *date_captured*: the date and time when the image was captured.

- *annotations*: contain all meta-data about the labels related to an object. They are a bounding box, segmentation mask and classification label.
  - *id*: The index of instance.
  - *image_id*: The index of the corresponding image. This *image_id* is corresponding to *id* in *images*.
  - *category_id*: This is category id which is defined in *categories*. In our case, *category_id* is either '1' - normal or '0' - defect.
  - *iscrowd*: if there are multiple instances/carcasses in the image, *iscrowd* is set as 1. Otherwise, *iscrowd* is set as 0 if there is a single instance/carcass in the image.
  - *area*: is the area of instance in the image.
  - *bbox*: The bounding box determines an object's location represented as [xmin,

ymin, width, height] where the (xmin, ymin) coordinates correspond to the top-left position of an object and (width, height) are width and height of the object. In the example shown in Fig. 6, xmin = 27, ymin = 0, width = 546, height = 731.

- *segmentation*: The segmentation mask is specified by Run-length encoded (RLE) values Golomb (1966).

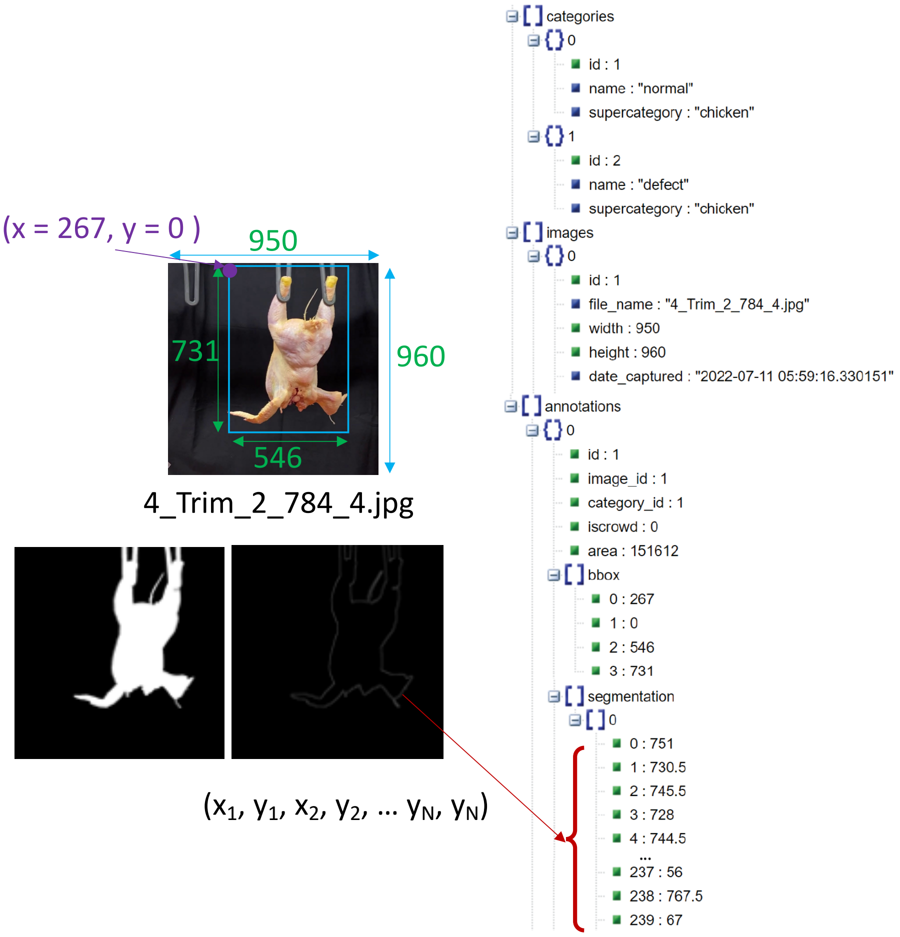
The data statistic of our CarcassDefect dataset is shown in Table 1 and Table 2. Table 1 shows the distribution of the image between a single carcass per image and multiple carcasses per image between two categories of normal and defect. Table 2 shows the distribution of the instance between a single carcass per image and multiple carcasses per image between two categories of normal and defect.

Table 2: The distribution of the **instances** in CarcassDefect Dataset regarded as normal and defective carcasses at both a single carcass per frame and multiple carcasses per frame.

	Single carcass per image		**Multiple carcasses per image**		**Total**
	Normal	**Defeat**	**Normal**	**Defeat**	
Trainset	1,302	1,715	1,571	1,842	6,430
Valset	355	399	422	466	1,642
Testset	320	188	359	267	1,134
Total	1,977	2,302	2,352	2,575	9,206

2.3. Proposed method

In the sections below, the proposed end-to-end transformer-based framework, termed CarcassFormer, is introduced for chicken carcass detection, segmentation, and carcass defect classification. Figure 1 illustrates the flowchart of our CarcassFormer network consisting of



```
[ ] categories
  { } 0
    id : 1
    name : "normal"
    supercategory : "chicken"
  { } 1
    id : 2
    name : "defect"
    supercategory : "chicken"
[ ] images
  { } 0
    id : 1
    file_name : "4_Trim_2_784_4.jpg"
    width : 950
    height : 960
    date_captured : "2022-07-11 05:59:16.330151"
[ ] annotations
  { } 0
    id : 1
    image_id : 1
    category_id : 1
    iscrowd : 0
    area : 151612
    [ ] bbox
      0 : 267
      1 : 0
      2 : 546
      3 : 731
    [ ] segmentation
      [ ] 0
        0 : 751
        1 : 730.5
        2 : 745.5
        3 : 728
        4 : 744.5
        ...
        237 : 56
        238 : 767.5
        239 : 67
```

Figure 6: A demonstration of data annotation in JSON file following COCO format.

four key parts: Backbone, Pixel Decoder, Multi-Scale Transformer Encoder, and Masked-attention Transformer Decoder. To train CarcassFormer, the stochastic gradient descent (SGD) optimizer was utilized with a learning rate of 0.0001 and a batch size of 4 over 100 epochs. The experiments were conducted using an Intel(R) Core(TM) i9-10980XE 3.00GHz CPU and a Quadro RTX 8000 GPU.

2.4. Backbone

A backbone network, a foundational architecture employed for feature extraction, is typically pre-trained on a variety of tasks and has demonstrated its effectiveness across various domains. AlexNet Krizhevsky et al. (2017) is regarded as the inaugural Deep Learning (DL) backbone. The VGG family, which includes VGG-16 and VGG-19 Simonyan and Zisserman

(2014), is one of the most prevalent backbones utilized in computer science endeavors. In contrast to AlexNet and VGG, ResNets He et al. (2016) are based on Convolutional Neural Networks (CNNs) and were developed concomitantly with the introduction of residual networks. ResNet variants, such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-151, are extensively employed for object detection and semantic segmentation tasks. Following the advent of ResNets, numerous other CNN-based backbones have been proposed, including Inception Szegedy et al. (2015), DenseNet Huang et al. (2017), DarkNet Lin et al. (2013), ShuffleNet Zhang et al. (2018), MobileNet Howard et al. (2017), and YOLO Huang et al. (2018). Recently, there has been a significant advancement in backbone architectures, incorporating transformer architecture Vaswani et al. (2017), along with leveraging the multi-scale features of ResNet. Prominent examples of these advancements include ViT Dosovitskiy et al. (2020), PvT Zhao et al. (2017), and Swin Liu et al. (2021). In the present work, ResNet (i.e., ResNet-18, 34, 50) He et al. (2016) and Swin (i.e., Swin-T) are employed as the backbone network.

By utilizing ResNets He et al. (2016) as the backbone network, an input image I with dimensions $H \times W$ is transformed into a multi-scale feature F, specifically a set of four feature maps $F_i i = 1^4$. These feature maps are represented as $F_1 \in \mathbb{R}^{CF_1 \times \frac{H}{4} \times \frac{W}{4}}$, $F_2 \in \mathbb{R}^{C_{F_2} \times \frac{H}{8} \times \frac{W}{8}}$, $F_3 \in \mathbb{R}^{C_{F_3} \times \frac{H}{16} \times \frac{W}{16}}$, and $F_4 \in \mathbb{R}^{C_{F_4} \times \frac{H}{32} \times \frac{W}{32}}$, where $C_{F_1}, C_{F_2}, C_{F_3}, C_{F_4}$ denote the number of channels.

2.5. Pixel Decoder

This module enhances the multi-scale features of an image by utilizing four feature maps from the backbone. It consists of two parts: the Multi-Scale Transformer Encoder (section 2.5.1) and the Per-pixel Embeddings Module (section 2.5.2). In general, the Multi-Scale Transformer Encoder uses an attention mechanism to learn the correlation between the multi-scale feature maps F_1, F_2, F_3, F_4. This results in corresponding, richer encoded feature maps D_1, D_2, D_3, D_4. Meanwhile, the per-pixel embeddings Module takes the encoded feature map (D_1) to compute the per-pixel embeddings $\mathcal{E}_{pixel}$ of the image.

2.5.1. Multi Scale Transformer Encoder

This module takes the last three features from the backbone, ordered from low to high resolution (i.e., F_4, F_3, F_2, and F_1), are processed in a hierarchical fashion. These three features first go through an embedding projection f_E to achieve the flattened embed feature S_i with a consistent channel size C_e. Note that the value of C_e is equal to C_{F_1}, and this is specifically intended for computing the per-pixel embeddings in section 2.5.2.

$$S_i = f_E(F_i) \tag{2.1}$$

where $i = \{4, 3, 2, 1\}$, $S_i \in \mathbb{R}^{H_i \cdot W_i \times C_e}$, f_E is a 1×1 convolution layer with the output dimension of C_e, followed by a flatten layer. The purpose of the flatten layer is to prepare S_i as input for a transformer layer, which requires a sequence of embedding features rather than spatial features.

To investigate the correlated feature embeddings between different levels, the flattened embedding features S_i from each multi-scale level were concatenated and passed through a transformer encoder. This involves merging the flattened embeddings from different levels into a single input sequence for the transformer encoder.

$$S = [S_i]_{i \in \{4,3,2,1\}} \tag{2.2}$$

where $S \in \mathbb{R}^{K \times C_e}$, K is the total number of embedding feature, $K = \sum_{i \in \{4,3,2,1\}} H_i \cdot W_i$.

However, since the current embedding features S are flattened out of their original spatial shapes and concatenated from multiple levels, they do not include information about the spatial location and scale level of each feature. To address this issue, each embedding feature in S is supplemented with two types of learnable encoding. The first is a positional encoding that provides spatial information about the original location of each feature within the image. The second is a level encoding that enables the transformer encoder to distinguish between features from different scales. By incorporating these encodings, spatial and scale level information is preserved during the calculation process. Let denote the learnable positional encodings as P and the learnable level encodings as L, where P and L share the same shape with S, $P, L \in \mathbb{R}^{K \times C_e}$.

Let denote the Multi Scale Transformer Encoder as f_G, this Transformer follows the architecture designed in Dosovitskiy et al. (2020). This transformer encoder produces learned features from the input sequence. It takes a sequence of embedded features and outputs encoded features that capture the relationships between the elements. These encoded features retain important information while removing redundancy. The encoder computes each encoded feature using a self-attention mechanism Vaswani et al. (2017), allowing it to selectively focus on the most relevant features and capture long-range dependencies, making it effective for enriching multiscale features. Formulary, the correlated feature embeddings between different levels $E \in \mathbb{R}^{K\times C_e}$ is computed by passing S, L, and P through f_G.

$$E = f_G(S, P, L) \tag{2.3}$$

The correlated feature embedding E is divided into groups based on the multi-scale level, denoted as E_i where $i \in \{4, 3, 2, 1\}$ and $E_i \in \mathbb{R}^{H_i\cdot W_i\times C_e}$. Next, each E_i is restored to its original spatial shape by unflattening, resulting in an output enriched multi-scale feature map D_i.

$$D_i = \text{unflatten}(E_i) \tag{2.4}$$

where $D_i \in R^{C_e\times H_i\times W_i}$.

In summary, this module takes the multi-scale feature map $F_4 \in \mathbb{R}^{C_{F_4}\times\frac{H}{32}\times\frac{W}{32}}$, $F_3 \in \mathbb{R}^{C_{F_3}\times\frac{H}{16}\times\frac{W}{16}}$, $F_2 \in \mathbb{R}^{C_{F_2}\times\frac{H}{8}\times\frac{W}{8}}$ and $F_1 \in \mathbb{R}^{CF_1\times\frac{H}{4}\times\frac{W}{4}}$ and outputs the enriched multi-scale feature map $D_4 \in \mathbb{R}^{C_e\times\frac{H}{32}\times\frac{W}{32}}$, $D_3 \in \mathbb{R}^{C_e\times\frac{H}{16}\times\frac{W}{16}}$, $D_2 \in \mathbb{R}^{C_e\times\frac{H}{8}\times\frac{W}{8}}$ and $D_1 \in \mathbb{R}^{C_e\times\frac{H}{4}\times\frac{W}{4}}$, which captures the correlation and important information while removing redundancy.

2.5.2. Per-pixel Embeddings Module

This section describes the second stage of Pixel Decoder, where the per-pixel embedding $\mathcal{E}_{pixel}$ is computed. This module takes the encoded feature map $D_1 \in \mathbb{R}^{C_e\times\frac{H}{4}\times\frac{W}{4}}$ from the Multi Scale Transformer Encoder module as input. The per-pixel embedding $\mathcal{E}_{pixel}$ is computed as follow:

$$\mathcal{E}_{pixel} = f_U(D_1) \tag{2.5}$$

The function f_U is a sequence of two 2×2 transposed convolutional layers with stride 2, which scales up the spatial shape of D_1 four times from $\frac{H}{4} \times \frac{W}{4}$ to the original image's spatial shape of $H \times W$. As a result, $\mathcal{E}_{pixel}$ has a dimension of $\mathbb{R}^{Ce \times H \times W}$.

Intuitively, each pixel feature of $\mathcal{E}_{pixel}$ represents both the semantic and the mask classification feature of the corresponding pixel on the original image.

2.6. Mask-attention Transformer Decoder

2.6.1. Mask Predictor

To predict the segmented masks of possible instances in an image, per-pixel embeddings $\mathcal{E}_{pixel} \in \mathbb{R}^{Ce \times H \times W}$ were utilized. These embeddings represent both the semantic and mask classification features of each corresponding pixel on the original image.

Then, the prediction process involves learning N per-segment query embeddings $Q \in \mathbb{R}^{N \times C_e}$, which represent the features of the maximum N possible instances in the image. Each instance query embedding correlates with every single pixel feature in $\mathcal{E}_{pixel}$ to determine whether the pixel belongs to the corresponding instance or not. Therefore, the predicted instance segmentation mask was derived as follows:

$$M = f_P(Q, \mathcal{E}_{pixel}) \tag{2.6}$$

where $M \in \mathbb{R}^{N \times H \times W}$, which are N masks of N possible instances in the image. The Mask Predictor f_P is a simple dot product on the feature channel C_e, followed by a sigmoid activation.

2.6.2. Mask-attention Transformer Decoder

The Mask-Attention Transformer Decoder f_T was employed to obtain effective per-segment query embeddings $Q \in \mathbb{R}^{N \times C_e}$ that represented instances in the image. This decoder applies attention to the image features, allowing it to decode the per-segment query embeddings and capture the instance mask feature.

The third blob in our overall flowchart (Figure 1) illustrates the procedure on applying the Mask-Attention Transformer Decoder. In general, this module decodes N per-segment query embeddings $Q \in \mathbb{R}^{N \times C_q}$ from the encoded feature maps D_1, D_2, D_3, and D_4. These query embeddings represent the features of the maximum N possible instances in the image.

The decoding procedure is performed recurrently, with each step treated as a layer (denoted as l) and beginning at $l = 0$. The encoded feature maps D have four levels, denoted as D_4, D_3, D_2, and D_1. Therefore, this recurrent process occurs four times, progressing from the lowest to highest resolution encoded feature maps. During each recurrent step, the encoded feature maps that are considered are D_{4-l}, where l represents the current layer. This means that during the first recurrent step ($l = 0$), the encoded feature maps that are used are D_4. During the second recurrent step ($l = 1$), the encoded feature maps that are used are D_3. During the third recurrent step ($l = 2$), the encoded feature maps that are used are D_2. Finally, during the fourth recurrent step ($l = 3$), the encoded feature maps that are used are D_1. At each layer, the queries Q_{l+1} are decoded from the previous layer's query Q_l and the corresponding encoded feature maps.

Additionally, a predicted mask, M_l is computed by using the current query embeddings Q_l and the per-pixel embeddings $\mathcal{E}_{pixel}$. The resulting mask is then interpolated to the same size as the current feature map D_{4-l}. This mask is used as an attention mechanism that helps the query embeddings to focus on the most salient parts of the feature maps. Specifically, during the decoding process, the attention mask is applied to the encoded feature maps D_{4-l}, allowing the query embeddings to selectively attend to certain regions of the feature maps that are most relevant to the instance being decoded. Formularly, at each recurrent step:

$$\begin{aligned} M_l &= f_P(Q_l, \mathcal{E}_{pixel}) \\ Q_{l+1} &= f_T(Q_l, D_{4-l}, M_l) \end{aligned} \tag{2.7}$$

2.7. Instance Mask and Class Prediction

The procedure can be visualized using the fourth component of the overall flowchart, as shown in Figure 1. In this step, the query encoder Q_L (where $L = 3$) and the per-pixel embeddings $\mathcal{E}pixel$ are utilized to compute the output instance segmentation masks, denoted as M_{final}. These masks, represented by a tensor $M_{final} \in \mathbb{R}^{N \times H \times W}$, correspond to

N possible instances within the image.

To generate the masks, the function f_P takes Q_L and $\mathcal{E}pixel$ as inputs:

$$M_{final} = f_P(Q_L, \mathcal{E}_{pixel}) \tag{2.8}$$

Moreover, alongside the masks, the semantic class of each instance is predicted using another function called f_C. This function is implemented as a Multi-Layer Perceptron (MLP) with two hidden layers. It takes the per-segment embeddings Q_L as input and produces N semantic classes, represented by $C_{final} \in \mathbb{R}^{N \times C}$. Here, C represents the number of semantic categories.

The prediction of semantic classes can be expressed as:

$$C_{final} = f_C(Q_L) \tag{2.9}$$

By combining these steps, this module is able to generate both instance segmentation masks (M_{final}) and predict the semantic class labels for each instance (C_{final}) using the decoded instance queries (Q_L) and per-pixel embeddings ($\mathcal{E}_{pixel}$).

2.8. Metrics

We adopt Average Precision (AP) to evaluate the method. AP quantifies how well the model is able to precisely locate and classify objects (e.g. defect or normal) within an image. The AP computation from MSCOCO Lin et al. (2014) was followed.

In the recognition task, each image is associated with a single prediction for classification. Evaluating the model became straightforward as the accuracy metric could be calculated, measuring the ratio of correct predictions. On the other hand, in the field of object detection and classification, a prediction comprises a bounding box or a segmentation mask that helps locate the object, along with the predicted category for that object. To determine a **correct prediction**, two criteria are considered. Firstly, the prediction must have an Intersection over Union (IoU) value greater than a threshold ϵ when compared to the actual box or mask of the object. Secondly, the prediction must accurately classify the category of the object. In addition, for each image, a method can output multiple predictions and the number of predictions can be higher or lower than the actual object within the image. Thus, precision and recall metrics are taken into account. Precision is the ratio of correct predictions to the total number of predictions (Equation 2.10). Precision can be considered as a measure of how precise the model's predictions were in terms of correctly detecting objects.

$$Pre = \frac{\text{number of correct predictions}}{\text{number of predictions}} \tag{2.10}$$

Meanwhile, recall is the ratio of correct predictions to the total number of actual objects within the image (Equation2.11). It can be thought of as a measure of how comprehensive the model's predictions are in terms of capturing all the objects present.

$$Rec = \frac{\text{number of correct predictions}}{\text{number of actual objects}} \tag{2.11}$$

Average Precision (AP) calculates the average precision across different recall values. Specifically, AP is computed at different IoU thresholds ϵ, which determine what is considered a correct prediction. For instance, when the threshold ϵ is set to 50%, it is denoted as AP@50. Let's consider an image with a list of actual ground truth objects denoted as $A = \{a_1, a_2, ..., a_n\}$, and a method that generates m predictions denoted as $B = \{b_1, b_2, ..., b_m\}$. The predictions are sorted in descending order based on their confidence scores. In the process, the sorted list B was iterated through, and at each step, the correctness of the prediction b_i (where $i \in \{1, 2, ..., m\}$) was determined. This is done by checking if the category is correctly matched and if the IoU is greater than the specified threshold ϵ. The number of correct predictions at this step was kept track of, denoted as C_i.

Using C_i, the precision Pre_i and recall Red_i could be computed at each step. The iteration stops when $Rec_i = 1$, indicating that all the objects have been captured, or when all the predictions were iterated through. The AP@ϵ is then computed as follows:

$$AP@\epsilon = \int_0^1 Pre(Rec), dRec \tag{2.12}$$

The reported AP in our table is the average of AP values ranging from AP@50 to AP@95, with a step size of 5% as depicted in Equation 2.13. This provides a comprehensive evaluation of the model's performance across different IoU thresholds and recall levels.

$$AP = \frac{1}{10} \sum_{\epsilon=0.5;\epsilon+=0.05}^{0.95} AP@\epsilon \tag{2.13}$$

We also quantify the model's complexity using three key metrics: the number of floating-point operations (FLOPs), the count of model parameters (Params), and the frames processed per second (FPS). FLOPs are computed as an average over 100 testing images. FPS is evaluated on a Quadro RTX 8000 GPU with a batch size of 1, calculated as the average runtime across the entire validation set, inclusive of post-processing time.

3. Results and Discussion

3.1. Implementation Details

The implementation of the pixel decoder in this study involves the use of an advanced multi-scale deformable attention Transformer (DERT) as described in Zhu et al. (2020). Specifically, the DERT is applied to feature maps with resolutions of 1/8, 1/16, and 1/32. A simple upsampling layer with a lateral connection is then employed on the final 1/8 feature map to generate the per-pixel embedding feature map of resolution 1/4. The Transformer encoder used in this study is configured with L=3 and a set of 100 queries.

3.2. Quantitative Performance and Comparison

In this section, our proposed CarcassFormer was evaluated on the Carcass dataset (Section 2.1), which consisted of two subsets corresponding to a single carcass per image and multiple carcasses per image. The performance of CarcassFormer on various metrics, as shown in Table 3 and Table 3 for different tasks, was detailedly reported. The authors then compared CarcassFormer with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019), as well as Transformer-based networks, namely Mask DINO Li et al. (2023), Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021). The comparison was conducted using three different backbone networks: ResNet-34, ResNet-50, and Swin-T. The performance for two subsets was reported: a single carcass per image (Table 5, Table 6) and multiple carcasses per image (Table 7, Table 8, Table 9). For each table, the metrics for detection, classification, segmentation, and model complexity were reported, as defined in Section 2.8.

Table 3: Detailed Performance of CarcassFormer on Single Carcass Per Image Dataset on both Detection and Segmentation, whereas AP_{normal} & AP_{defect} include classification results.

Backbone	**Task**	AP	AP_{50}	AP_{75}	AP_{95}	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
ResNet 34	Detection	97.70	98.23	98.23	92.89	98.02	97.38	41M	274G	5.1
	Segmentation	99.22	99.22	99.22	99.22	100.00	98.45			
ResNet 50	Detection	95.18	95.18	95.18	95.18	94.02	96.34	41M	274G	5.1
	Segmentation	98.43	98.43	98.43	98.43	99.79	97.06			
Swin-T	Detection	95.69	95.79	95.93	95.32	94.23	97.15	46M	281G	4.5
	Segmentation	97.77	98.65	98.93	98.15	99.11	96.42			

Table 4: Detailed Performance of CarcassFormer on Multiple Carcasses Per Image Dataset on both Detection and Segmentation, whereas AP_{normal} & AP_{defect} include classification results.

Backbone	**Task**	AP	AP_{50}	AP_{75}	AP_{95}	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
ResNet 34	Detection	89.72	91.45	91.45	78.51	93.48	85.96	41M	274G	5.1
	Segmentation	98.23	99.34	98.86	92.55	98.77	97.68			
ResNet 50	Detection	90.45	91.55	91.55	83.41	93.42	87.49	41M	274G	5.1
	Segmentation	98.96	99.98	99.48	94.36	99.15	98.76			
Swin-T	Detection	89.34	91.27	91.73	79.11	94.18	84.50	46M	281G	4.5
	Segmentation	98.70	99.29	98.32	93.10	99.47	97.92			

3.2.1. Detailed Quantitative Performance

Detailed performance conducted by our CarcassFormer is reported in Table 3 and Table 4 corresponding to two subsets: a single carcass per image and multiple carcasses per image. In each subset, our CarcassFormer network was examined on three different backbone networks: ResNet-34, ResNet-50, and Swin-T. For both tasks of detection and segmentation, Average Precision (AP) at different metrics of AP@50, AP@75, AP@95, and AP[50:95] (referred to as AP) were reported. Regarding detection and classification, AP_{normal} and AP_{defect} were evaluated for normal and defect classes. The results obtained from two tables (Table 3 and 4) underscores the remarkable performance of our model across various backbones and tasks, with every configuration achieving an AP of over 85 for all metrics. Additionally, it becomes evident that the Multiple Carcasses Per Image Dataset presents greater challenges compared to the Single Carcasses Per Image Dataset. This observation is substantiated by a noticeable decline in performance metrics when handling multiple overlapping carcasses per image, as opposed to the single carcass per image scenario.

Table 5: Performance comparison between CarcassFormer with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019) and Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021) on both Detection, Classification and Segmentation tasks. The comparison is conducted on **ResNet-34** backbone network and in the case of **single carcass per image**. Net. denotes Network architecture

Net.	**Method**	**Venue**	**Detection & Classification**			**Segmentation**			**Model Complexity**		
			AP	AP_{normal}	AP_{defect}	AP	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
CNN -based	Mask R-CNN He et al. (2017)	2017	79.73	87.82	71.65	81.36	85.41	77.31	41M	204G	5.8
	HTC Chen et al. (2019)	2019	89.00	95.30	82.60	82.30	86.30	78.40	109M	290G	4.3
Transformer -based	QueryInst Fang et al. (2021)	2021	90.40	98.00	82.90	82.20	88.10	76.30	45M	279G	4.8
	Mask2Former Cheng et al. (2022)	2022	58.33	58.42	58.24	75.32	92.08	58.57	41M	272G	5.2
	Mask DINO Li et al. (2023)	2023	88.12	95.11	81.12	77.67	83.45	71.89	49M	278G	5.1
	CarcassFormer (Ours)		**97.70**	**98.02**	**97.38**	**99.22**	**100.00**	**98.45**	41M	274G	5.1

3.2.2. Single Carcass Per Image

Table 5 and Table 6 present the performance on a single carcass per image using ResNet-34 and ResNet-50, respectively.

Table 6: Performance comparison between CarcassFormer with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019) and Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021) on both Detection, Classification and Segmentation tasks. The comparison is conducted on **ResNet-50** backbone network and in the case of **single carcass per image**. Net. denotes Network architecture. The best score in each table is highlighted in **bold**.

Net.	**Method**	**Venue**	**Detection & Classification**			**Segmentation**			**Model Complexity**		
			AP	AP_{normal}	AP_{defect}	AP	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
CNN -based	Mask R-CNN He et al. (2017)	2017	80.35	90.15	70.56	84.19	88.39	80.00	44M	207G	5.3
	HTC Chen et al. (2019)	2019	88.10	**96.20**	80.00	84.30	89.00	79.70	112M	294G	3.9
Transformer -based	QueryInst Fang et al. (2021)	2021	64.60	75.50	53.60	72.70	78.60	66.70	48M	281G	4.4
	Mask2Former Cheng et al. (2022)	2022	85.05	91.23	78.87	85.11	91.23	78.99	44M	276G	4.8
	Mask DINO Li et al. (2023)	2023	85.12	91.44	79.10	86.13	92.11	80.15	52M	280G	4.6
	CarcassFormer (Ours)		**95.18**	94.02	**96.34**	**98.43**	**99.79**	**97.06**	44M	278G	4.6

Table 5 compares the performance of CarcassFormer with existing methods on ResNet-34. In the first group, HTC Chen et al. (2019) obtains better performance than Mask R-CNN He et al. (2017) whereas our CarcassFormer gains significant performance gaps compared to both HTC Chen et al. (2019) and Mask R-CNN He et al. (2017). Take HTC Chen et al. (2019) as an example, CarcassFormer outperforms HTC with 8.70% higher AP for detection, 2.72% higher AP for normal carcass classification, 14.78 % higher AP for defect carcass classification, 16.92% higher AP for segmentation, 13.70% higher AP segmentation for normal carcass and 20.05% higher AP segmentation for defect carcass. In the second group, QueryInst Fang et al. (2021) obtains better performance than Mask2Former Cheng et al. (2022) and Mask DINO Li et al. (2023) while our CarcassFormer obtains the best performance. Compared to QueryInst Fang et al. (2021), CarcassFormer gains 7.30% higher AP for detection, 0.02% higher AP for normal carcass classification 14.48% higher AP for defect carcass classification, 17.02% higher AP for segmentation, 11.90% higher AP segmentation for normal carcass, 22.15% higher AP segmentation for defect carcass.

Table 6 compares the performance of CarcassFormer with existing methods on ResNet-50.

In the first group, HTC Chen et al. (2019) outperforms Mask R-CNN He et al. (2017) whereas CarcassFormer outperforms HTC Chen et al. (2019) with significant performance gaps, including 7.08% higher AP for detection, 16.34% higher AP for defect carcass classification, 14.13% higher AP for segmentation, 10.79% higher AP segmentation for normal carcass and 17.36% higher AP segmentation for defect carcass while it is compatible with HTC Chen et al. (2019) on normal classification. In the second group, while Mask2Former Cheng et al. (2022) and Mask DINO Li et al. (2023) obtains much better performance than QueryInst Fang et al. (2021), our CarcassFormer outperforms MaskDINO Li et al. (2023) 10.06% higher AP for detection, 2.58% higher AP for normal carcass classification, 17.24% higher AP for defect carcass classification, 13.30% higher AP for segmentation, 7.68% higher AP segmentation for normal carcass, 16.91% higher AP segmentation for defect carcass.

Table 7: Performance comparison between CarcassFormer with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019) and Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021) on both Detection, Classification and Segmentation tasks. The comparison is conducted on **ResNet-34** backbone network and in the case of **multiple carcasses per image**. Net. denotes Network architecture. The best score in each table is highlighted in **bold**.

Net.	**Method**	**Venue**	**Detection & Classification**			**Segmentation**			**Model Complexity**		
			AP	AP_{normal}	AP_{defect}	AP	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
CNN -based	Mask R-CNN He et al. (2017)	2017	77.08	84.33	69.83	74.81	79.00	70.63	41M	204G	5.8
	HTC Chen et al. (2019)	2019	77.80	89.70	65.90	74.00	79.10	68.90	109M	290G	4.3
Transformer -based	QueryInst Fang et al. (2021)	2021	84.10	89.60	78.70	83.20	87.70	78.70	45M	279G	4.8
	Mask2Former Cheng et al. (2022)	2022	53.86	54.00	53.72	71.69	85.39	58.00	41M	272G	5.2
	Mask DINO Li et al. (2023)	2023	68.44	74.55	62.33	78.80	88.26	69.33	49M	278G	5.1
	CarcassFormer (Ours)		**89.72**	**93.48**	**85.96**	**98.23**	**98.77**	**97.68**	41M	274G	5.1

3.2.3. Multiple Carcasses Per Image

Table 7 and Table 8 present the performance on multiple carcasses per image using ResNet-34 and ResNet-50, respectively.

Table 8: Performance comparison between CarcassFormer with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019) and Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021) on both Detection, Classification and Segmentation tasks. The comparison is conducted on **ResNet-50** backbone network and in the case of **multiple carcasses per image**. Net. denotes Network architecture. The best score in each table is highlighted in **bold**.

Net.	**Method**	**Venue**	**Detection & Classification**			**Segmentation**			**Model Complexity**		
			AP	AP_{normal}	AP_{defect}	AP	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
CNN -based	Mask R-CNN He et al. (2017)	2017	78.76	85.61	75.73	80.67	85.61	75.73	44M	207G	5.3
	HTC Chen et al. (2019)	2019	77.40	83.50	71.40	74.90	77.70	72.10	112M	294G	3.9
Transformer -based	QueryInst Fang et al. (2021)	2021	60.90	67.70	54.00	60.40	66.90	54.00	48M	281G	4.4
	Mask2Former Cheng et al. (2022)	2022	73.35	88.03	58.66	75.54	90.93	60.15	44M	276G	4.8
	Mask DINO Li et al. (2023)	2023	76.22	84.11	68.33	79.93	92.12	67.74	52M	280G	4.6
	CarcassFormer (Ours)		**90.45**	**93.42**	**87.49**	**98.96**	**99.15**	**98.76**	44M	278G	4.6

Table 9: Performance comparison between CarcassFormer with Mask2Former Cheng et al. (2022) on both Detection, Classification and Segmentation tasks. The comparison is conducted on **Swin-T** backbone network and in the case of **multiple carcasses per image**. Net. denotes Network architecture. The best score in each table is highlighted in **bold**.

	Method	**Venue**	**Detection & Classification**			**Segmentation**			**Model Complexity**		
			AP	AP_{normal}	AP_{defect}	AP	AP_{normal}	AP_{defect}	Params	FLOPs	FPS
	Mask R-CNN He et al. (2016)	2017	78.82	86.12	71.52	81.22	87.12	75.32	46M	230G	4.8
	Mask2Former Cheng et al. (2022)	2022	73.10	88.14	58.05	77.68	92.89	62.47	46M	280G	4.6
	CarcassFormer (Ours)		**89.34**	**94.18**	**84.50**	**98.70**	**99.47**	**97.92**	46M	281G	4.5

Table 7 compares the performance of CarcassFormer with existing methods on ResNet-34. In the first group, while Mask R-CNN He et al. (2017) and HTC Chen et al. (2019) are quite compatible on all tasks, our CarcassFormer gains big performance gaps. Take HTC Chen et al. (2019) as an example, CarcassFormer achieves 11.92% higher AP for detection, 3.78% higher AP for normal carcass classification, 20.06% higher AP for defect carcass classification, 24.23% higher AP for segmentation, 19.67% higher AP segmentation for normal carcass, 28.79% higher AP segmentation for defect carcass. In the second group, while QueryInst Fang et al. (2021) outperforms Mask2Former Cheng et al. (2022) and Mask DINO Li et al. (2023), our CarcassFormer obtains better performance than QueryInst Fang et al. (2021) with notable gaps, i.e., 5.62% higher AP for detection, 3.88% higher AP for normal carcass classification, 7.26% higher AP for defect carcass classification, 15.03% higher AP for segmentation, 11.07% higher AP segmentation for normal carcass, 18.98% higher AP segmentation for defect carcass.

Table 8 compares the performance of CarcassFormer with existing methods on ResNet-50. In the first group, while Mask R-CNN He et al. (2017) outperforms HTC Chen et al. (2019), our CarcassFormer achieves a best performance with 11.69% higher AP for detection, 7.81% higher AP for normal carcass classification, 11.76% higher AP for defect carcass classification, 18.29% higher AP for segmentation, 13.54% higher AP segmentation for normal carcass, 23.03% higher AP segmentation for defect carcass compared to Mask R-CNN He et al. (2017). In the second group, while Mask2Former Cheng et al. (2022) Mask DINO Li et al. (2023) obtain better performance than QueryInst Fang et al. (2021), our CarcassFormer achieves the best performance. It gains 14.23% higher AP for detection, 9.31% higher AP for normal carcass classification, 19.16% higher AP for defect carcass classification, 19.03% higher AP for segmentation, 7.03% higher AP segmentation for normal carcass, 31.02% higher AP segmentation for defect carcass compared to the second best method Mask DINO Li et al. (2023).

Table 9 compares the performance of CarcassFormer with Mask2Former Cheng et al. (2022) and Mask R-CNN He et al. (2017) on Swin-T. In comparison to the CNN-based method, Mask R-CNN He et al. (2017), our approach yields significant improvements across various performance metrics. Specifically, we observe a 10.52% increase in average precision (AP) for detection, an 8.06% enhancement for normal carcass classification, a 12.98% boost for defect carcass classification, a remarkable 17.48% rise for segmentation, as well as notable gains of 12.35% and 22.6% in AP segmentation for normal and defect carcasses, respectively. In terms of comparison with transformer based network, namely Mask2Former, our CarcassFormer achieves the significant better performance than both Mask2Former and

Mask R-CNN. Indeed, it gains 16.24% higher AP for detection, 6.04% higher AP for normal carcass classification, 26.45% higher AP for defect carcass classification, 21.02% higher AP for segmentation, 6.58% higher AP segmentation for normal carcass, 35.45% higher AP segmentation for defect carcass.

Model Complexity. Analysis of model complexity reveals that our method exhibits comparable complexity to the majority of existing methods. However, it consistently delivers notable performance enhancements across diverse tasks. Specifically, in the case of ResNet-34, as illustrated in Tables 5 and 7, our model has the smallest number of model parameters, equivalent to that of Mask R-CNN He et al. (2017) and Mask2Former Cheng et al. (2022), while maintaining comparable FLOPs and FPS with these models. However, our model exhibits a significant performance advantage over both. This trend is similarly observed for ResNet-50, as shown in Tables 6 and 8, and for Swin-T, as depicted in Table 9, where our model demonstrates comparable model complexity but yields substantial performance gains compared to other methods.

3.3. Qualitative Performance and Comparison

Based on the quantitative comparison in Section 3.2, Mask R-CNN He et al. (2017) was selected from the first group, and Mask2Former Cheng et al. (2022) was chosen from the second group to conduct the qualitative comparison. Specifically, the qualitative comparison was reported on both the detection and segmentation tasks, with a greater emphasis on the case of defect, namely feather and skin tearing.

3.3.1. Single Carcass Per Image

Figure 7 presents a qualitative performance comparison among three models: Mask R-CNN (a), Mask2Former (b), and our proposed CarcassFormer (c) on the defect of *single carcass with feathers* is present. While Mask R-CNN can segment the global content well, it fails to segment the details, such as feathers. On the other hand, Mask2Former performs better than Mask R-CNN in capturing details, but it still faces difficulties in capturing fine details, which can be seen at high resolution. Moreover, Mask R-CNN and Mask2Former

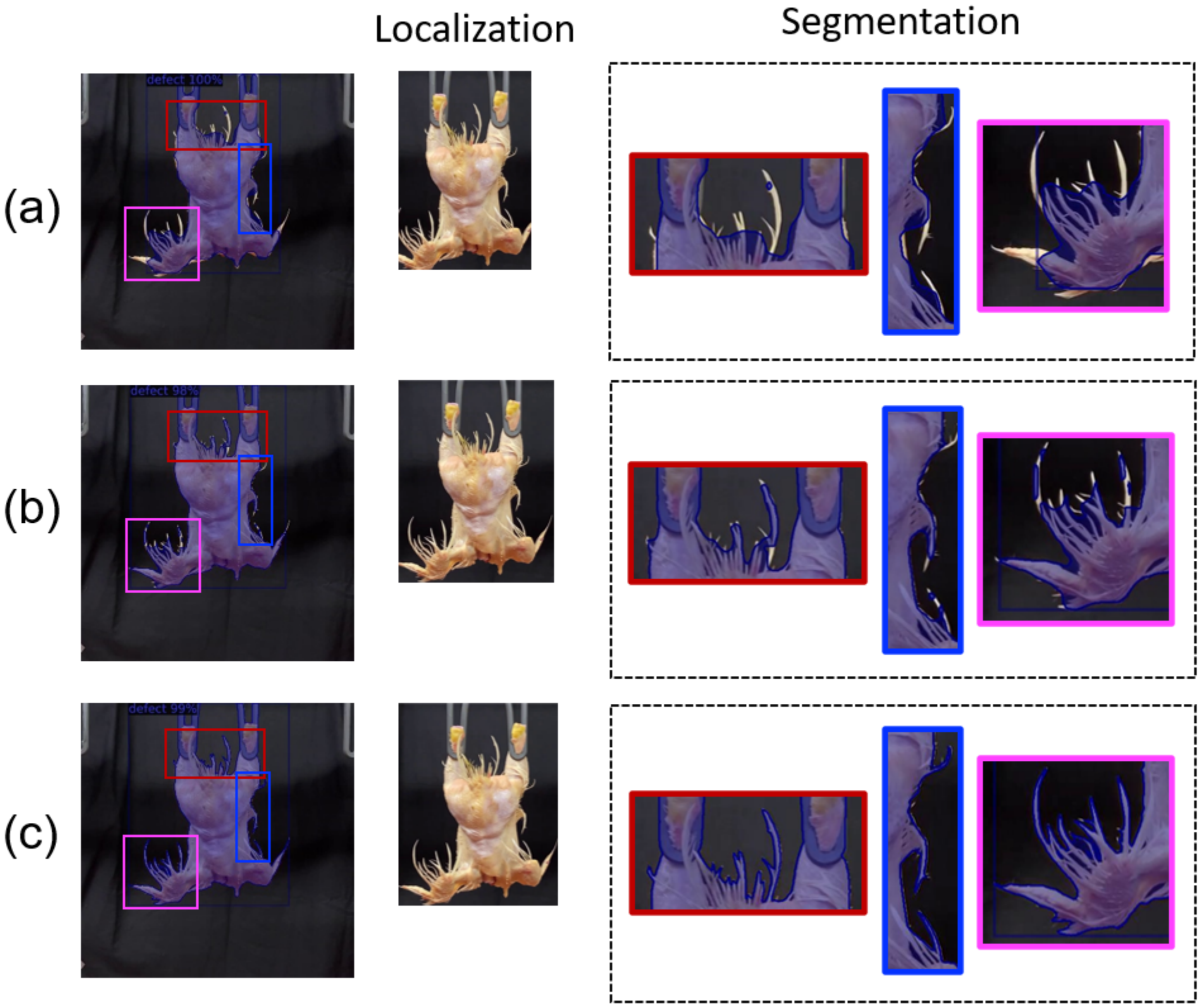


Figure 7: Performance comparison (a): Mask R-CNN He et al. (2016), (b): Mask2Former Cheng et al. (2022) and (c): our CarcassFormer on the defect where **single carcass with feathers**. In the Segmentation column, notable parts with feathers were highlighted. Compare with Mask R-CNN and Mask2Former, our CarcassFormer can localize carcass with more accurate bounding box and segment carcass with more details on feathers.

exhibited a tendency to under-localize the carcass, as observed from the detected bounding box that did not encompass the entire carcass with details on the boundary, such as wings and feathers. In contrast, our CarcassFormer not only accurately localizes the carcass with a fitting bounding box but is also capable of segmenting details at high resolution.

Figure 8 depicts a qualitative performance comparison among three models: Mask R-

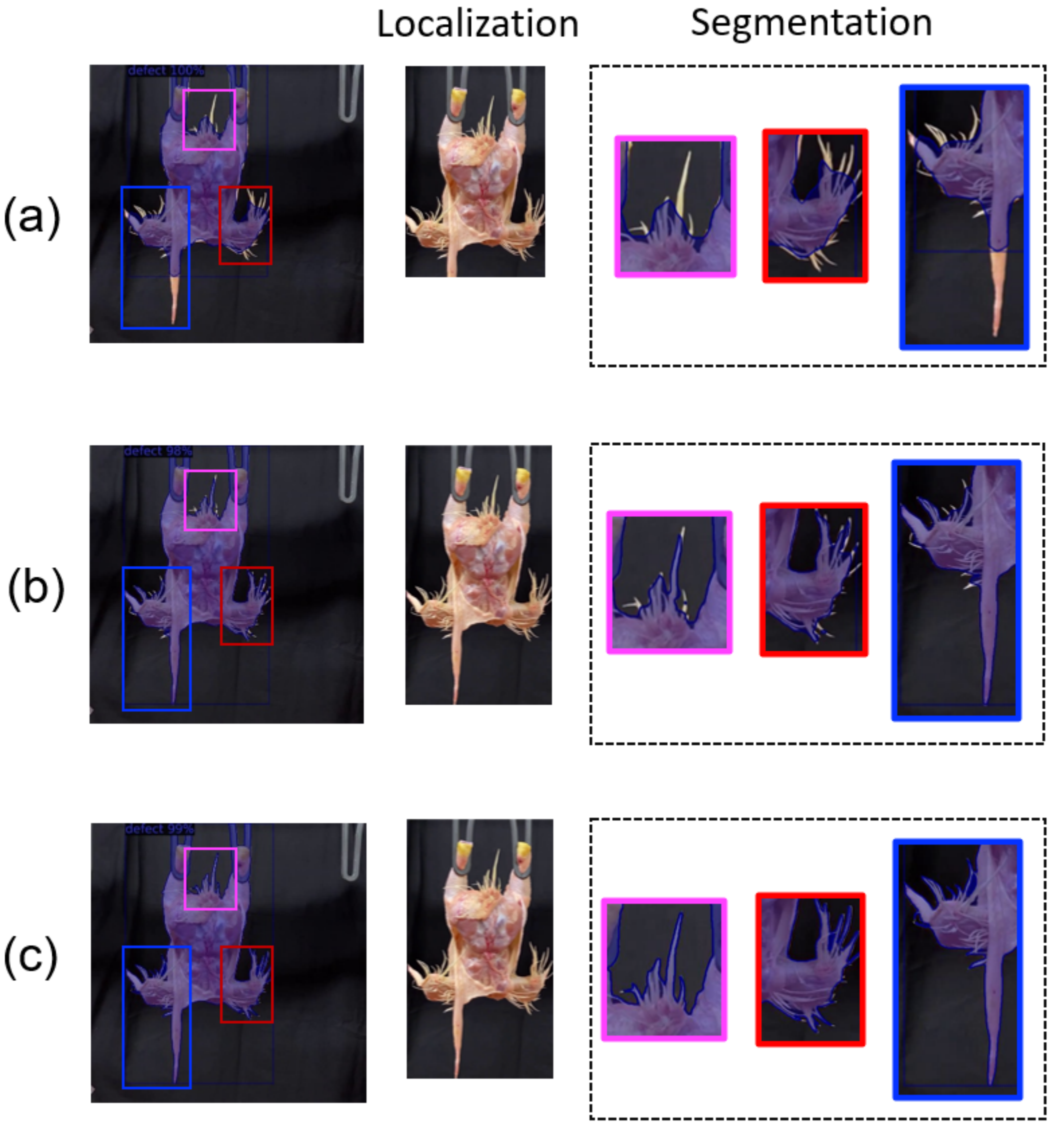


Figure 8: Performance comparison (a): Mask R-CNN He et al. (2016), (b): Mask2Former Cheng et al. (2022) and (c): our CarcassFormer on two defects **where single carcass with skins tearing on the back and feathers**. In the Segmentation column, notable parts with feathers and skins tearing occurred were highlighted. Compared with Mask R-CNN and Mask2Former, our CarcassFormer desnt not only detect the feathers well but also accurately localize carcass with its all skins tearing.

CNN (a), Mask2Former (b), and our proposed CarcassFormer (c), on *single carcass with both two defects of feather and skin tearing.* Although Mask2Former performs better than Mask R-CNN in localizing the carcass with skin tearing, it still faces difficulties in localizing

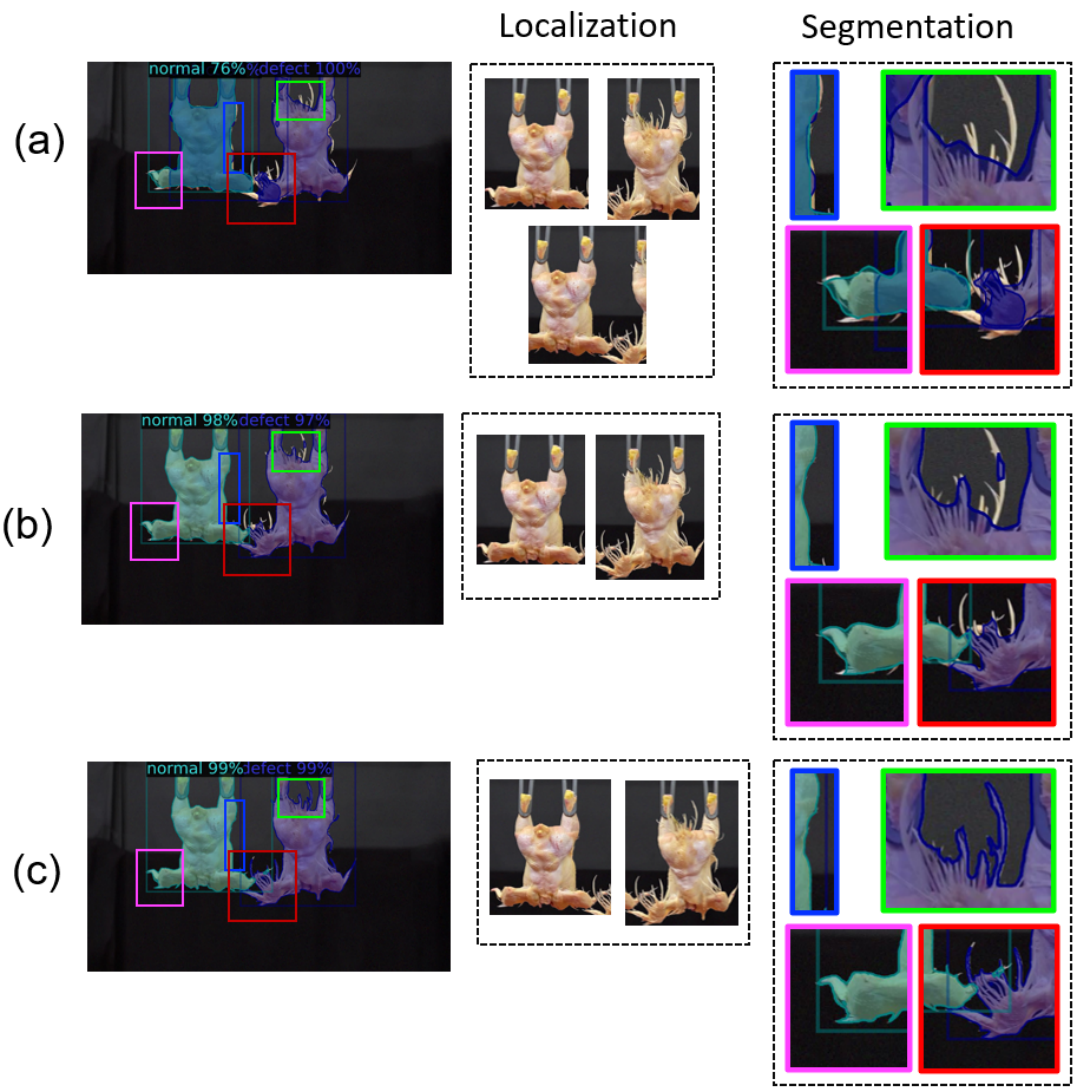


Figure 9: Performance comparison between Mask R-CNN (a), He et al. (2016) Mask2Former Cheng et al. (2022) (b) and our CarcassFormer (c) on **overlapping carcasses with feathers**. In the Segmentation column, notable parts with feathers were highlighted. Mask R-CNN not only lacks details in the segmentation results but also fails to localize individual carcasses. Although Mask2Former performs better than Mask R-CNN in localizing individual carcasses, it still struggles to accurately segment all details. In contrast, our CarcassFormer can simultaneously segment carcasses with details and accurately localize individual carcasses.

all details on feathers. Conversely, our CarcassFormer accurately localizes the carcass with a fitting bounding box and is also capable of segmenting details at high resolution.

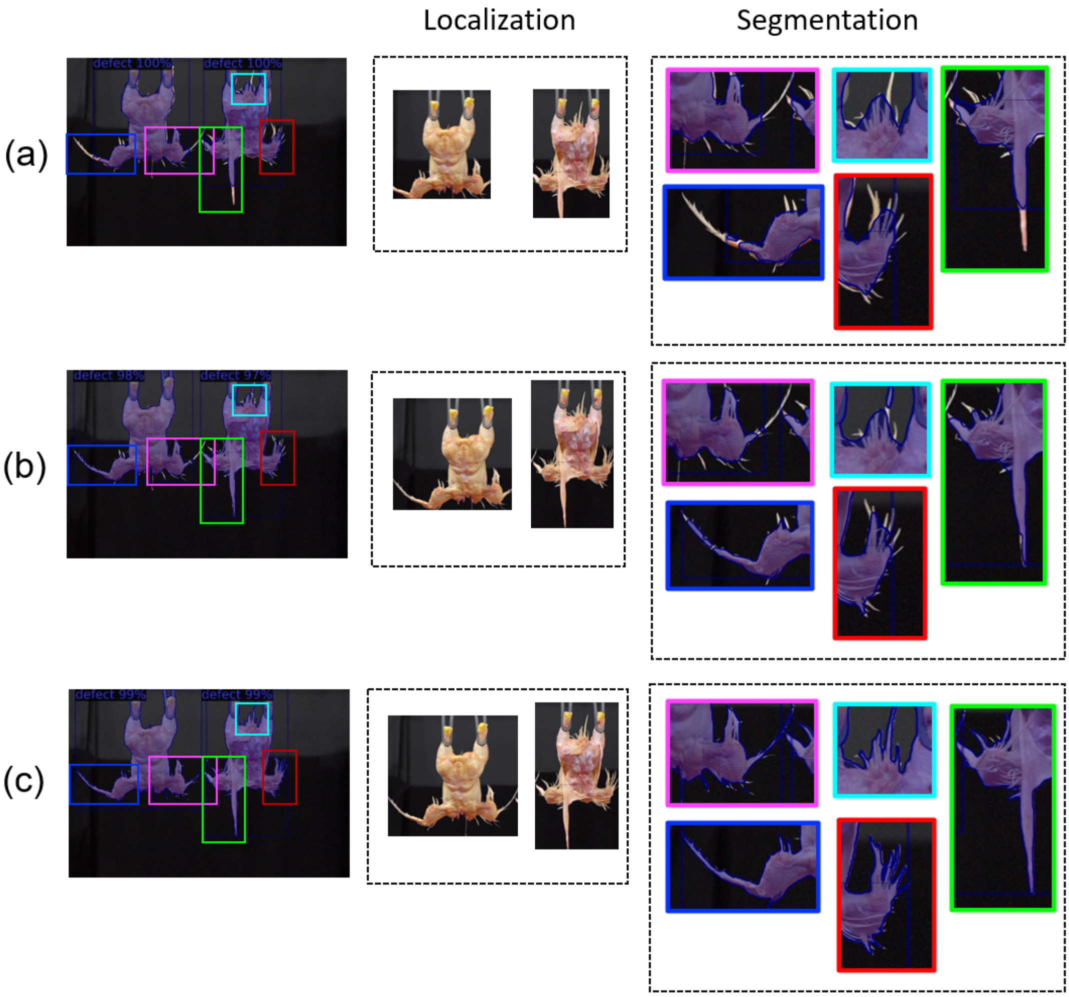


Figure 10: Performance comparison between (a): Mask R-CNN He et al. (2016), (b): Mask2Former Cheng et al. (2022) and (c): our CarcassFormer on **overlapping carcasses with feathers and skins tearing**. Segmentation highlights some notable parts with feathers occur. Both Mask R-CNN and Mask2Former struggle with accurately localizing individual carcasses and providing detailed segmentation, especially for fine details like feathers. In contrast, our CarcassFormer excels in simultaneously segmenting carcasses with fine details and accurately localizing individual carcasses.

3.3.2. Multiple Carcasses Per Image

In this section, the qualitative performance on images with multiple carcasses and their overlap was reported. Figure 9 illustrates the qualitative performance of three models: Mask R-CNN (a), Mask2Former (b), and our proposed CarcassFormer (c) *on multiple, overlapping carcasses with feathers*. Both Mask R-CNN and Mask2Former struggle to accurately segment and localize each individual carcass, especially in cases where the feathers of different carcasses overlap. Mask R-CNN lacks detail in the segmentation results, while Mask2Former, despite performing better in localizing individual carcasses, still fails to capture all the details accurately. On the contrary, our CarcassFormer excels at accurately segmenting each individual carcass and capturing the details of feathers, even in complex scenarios of overlap.

In Figure 10, the qualitative performance comparison of the three models *on multiple, overlapping carcasses with both feathers and skin tearing* presented. Here, it is evident that Mask R-CNN and Mask2Former both face significant challenges in accurately localizing individual carcasses and providing detailed segmentation, especially for tiny objects like feathers and areas of skin tearing. In stark contrast, our CarcassFormer performs outstandingly in these complex situations. It not only accurately localizes each carcass but also segments the fine details of feathers and skin tearing areas, thereby providing a comprehensive and detailed segmentation output.

4. Conclusions

In conclusion, an end-to-end Transformer-based network for checking carcass quality, CarcassFormer, has been described. Our CarcassFormer is designed with four different components: Network Backbone to extract visual features, Pixel Decoder to utilize feature maps from various scales, Mask-Attention Transformer Decoder to predict the segmented masks of all possible instances, and Instance Mask and Class Prediction to provide segmentation mask and corresponding label of an individual instance. To benchmark the proposed CarcassFormer network, a valuable realistic dataset was conducted at a poultry processing plant. The dataset acquired contained various defects including feathers, broken/disjointed bones, skins tearing, on different settings of a single carcass per image and multiple carcasses per image, and the carcass at various ages and sizes. The CarcassFormer was evaluated and compared with both CNN-based networks, namely Mask R-CNN He et al. (2017) and HTC Chen et al. (2019), as well as Transformer-based networks, namely Mask2Former Cheng et al. (2022) and QueryInst Fang et al. (2021), on both detection, classification, and segmentation tasks using two different backbone networks, ResNet-34 and ResNet-50. The extensive qualitative and quantitative experiment showed that our CarcassFormer outperforms the existing methods with remarkable gaps on various metrics of AP, AP@50, AP@75.

Our current CarcassFormer system operates solely on image-based inputs, limiting our ability to track carcasses across frames. While our model can currently determine whether a carcass is defective or not, it lacks the capability to identify specific types of defects, such as feathers around the carcass, feathers on the skin, flesh abnormalities, or broken wings. In our future endeavors, we aim to expand our research to include video analysis, enabling us to track carcasses across frames and thereby enhance the scalability of our system to process a larger volume of carcasses. Additionally, we intend to implement finer-grained defect detection to precisely identify the nature of defects present. This enhancement will provide more detailed insights into the types of defects observed, facilitating improved diagnosis and statistical analysis.

5. Acknowledgments

This material is based upon work supported by Cobb Vantress Inc., the National Science Foundation (NSF) under Award No OIA-1946391 RII Track-1, NSF 2119691 AI SUSTEIN.

References

Ahlin, K., 2022. The robotic workbench and poultry processing 2.0. Animal Frontiers 12, 49–55.

Alexandratos, N., Bruinsma, J., 2012. World agriculture towards 2030/2050: the 2012 revision. Agriculture Development Economics Division. Food and Agriculture Organization of the United Nations .

Arnab, A., Torr, P.H.S., 2016. Bottom-up instance segmentation using deep higher-order crfs, in: Wilson, R.C., Hancock, E.R., Smith, W.A.P. (Eds.), Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016, BMVA Press. URL: http://www.bmva.org/bmvc/2016/papers/paper019/index.html.

Aydin, A., 2017. Development of an early detection system for lameness of broilers using computer vision. Computers and Electronics in Agriculture 136, 140–146. doi:10.1016/j.compag.2017.02.019.

Bolya, D., Zhou, C., Xiao, F., Lee, Y.J., 2019. YOLACT: real-time instance segmentation, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE. pp. 9156–9165. URL: https://doi.org/10.1109/ICCV.2019.00925, doi:10.1109/ICCV.2019.00925.

Cai, Z., Vasconcelos, N., 2018. Cascade R-CNN: delving into high quality object detection, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, IEEE Computer Society. pp. 6154–6162. doi:10.1109/CVPR.2018.00644.

Caldas-Cueva, J.P., Mauromoustakos, A., Sun, X., Owens, C.M., 2021. Detection of woody breast condition in commercial broiler carcasses using image analysis. Poultry Science , 100977doi:10.1016/j.psj.2020.12.074.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16, Springer. pp. 213–229.

Chen, H., Sun, K., Tian, Z., Shen, C., Huang, Y., Yan, Y., 2020. Blendmask: Top-down meets bottom-up for instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, IEEE. pp. 8570–8578. URL: https://doi.org/10.1109/CVPR42600.2020.00860, doi:10.1109/CVPR42600.2020.00860.

Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., et al., 2019. Hybrid task cascade for instance segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 4974–4983.

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence 40, 834–848.

Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation, in: Proceedings of the European conference on computer vision (ECCV), pp. 801–818.

Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R., 2022. Masked-attention mask transformer for universal image segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1290–1299.

Cheng, B., Schwing, A., Kirillov, A., 2021. Per-pixel classification is not all you need for semantic segmentation. Advances in Neural Information Processing Systems 34, 17864–17875.

Cheng, T., Wang, X., Huang, L., Liu, W., 2020. Boundary-preserving mask r-cnn, in: European conference on computer vision, Springer. pp. 660–676.

Council, N.C., March 18, 2021. U.S. Broiler Performance. https://www.nationalchickencouncil.org/statistic/us-broiler-performance/. [Online; accessed 9-July-2021].

Dong, S., Wang, P., Abbas, K., 2021. A survey on deep learning and its applications. Computer Science Review 40, 100379.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. International Conference on Learning Representations (ICLR) .

Duong, C.N., Quach, K.G., Jalata, I., Le, N., Luu, K., 2019a. Mobiface: A lightweight deep learning face recognition on mobile devices, in: 2019 IEEE 10th international conference on biometrics theory, applications and systems (BTAS), IEEE. pp. 1–6.

Duong, C.N., Quach, K.G., Luu, K., Le, T., Savvides, M., Bui, T.D., 2019b. Learning from longitudinal face demonstration—where tractable deep modeling meets inverse reinforcement learning. International Journal of Computer Vision 127, 957–971.

Elam, T.E., 2022. Live chicken production trends.

Fan, R., Cheng, M.M., Hou, Q., Mu, T.J., Wang, J., Hu, S.M., 2019. S4net: Single stage salient-instance segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 6103–6112.

Fang, Y., Yang, S., Wang, X., Li, Y., Fang, C., Shan, Y., Feng, B., Liu, W., 2021. Instances as queries, in: Proceedings of the IEEE/CVF international conference on computer vision, pp. 6910–6919.

FASS, 2010. Guide for the Care and Use of Agricultural Animals in Research and Teaching. 3rd editio ed., Federation of Animal Science Societies. URL: `http://www.fass.orgorfromthe`.

Fathi, A., Wojna, Z., Rathod, V., Wang, P., Song, H.O., Guadarrama, S., Murphy, K.P., 2017. Semantic instance segmentation via deep metric learning. arXiv preprint arXiv:1703.10277 .

Gabeur, V., Sun, C., Alahari, K., Schmid, C., 2020. Multi-modal transformer for video retrieval, in: European Conference on Computer Vision, Springer. pp. 214–229.

Golomb, S., 1966. Run-length encodings (corresp.). IEEE transactions on information theory 12, 399–401.

Gu, W., Bai, S., Kong, L., 2022. A review on 2d instance segmentation based on deep neural networks. Image and Vision Computing , 104401.

Hafiz, A.M., Bhat, G.M., 2020. A survey on instance segmentation: state of the art. International journal of multimedia information retrieval 9, 171–189.

Han, L., Le, T.H.N., Savvides, M., 2017. An automatic cells detection and segmentation, in: Medical Imaging 2017: Biomedical Applications in Molecular, Structural, and Functional Imaging, SPIE. pp. 224–231.

He, H., Cai, J., Pan, Z., Liu, J., Zhang, J., Tao, D., Zhuang, B., 2023a. Dynamic focus-aware positional queries for semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11299–11308.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

He, P., Chen, Z., He, Y., Chen, J., Hayat, K., Pan, J., Lin, H., 2023b. A reliable and low-cost deep learning model integrating convolutional neural network and transformer structure for fine-grained classification of chicken eimeria species. Poultry Science 102, 102459.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 .

Hu, Z., Yang, H., Yan, H., 2023. Attention-guided instance segmentation for group-raised pigs. Animals 13, 2181.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708.

Huang, R., Pedoeem, J., Chen, C., 2018. Yolo-lite: a real-time object detection algorithm optimized for non-gpu computers, in: 2018 IEEE international conference on big data (big data), IEEE. pp. 2503–2510.

Ibtehaz, N., Rahman, M.S., 2020. Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation. Neural networks 121, 74–87.

Janai, J., Güney, F., Behl, A., Geiger, A., et al., 2020. Computer vision for autonomous vehicles: Problems, datasets and state of the art. Foundations and Trends® in Computer Graphics and Vision 12, 1–308.

Jin, Y., Liu, J., Xu, Z., Yuan, S., Li, P., Wang, J., 2021. Development status and trend of agricultural robot technology. International Journal of Agricultural and Biological Engineering 14, 1–19.

Kaminski, D.M., 2020. Re-Moo-Ving Barriers Within Labor: Exploring Current Events Related to Dairy and Poultry Labor Markets. Michigan State University.

Kong, S., Fowlkes, C.C., 2018. Recurrent pixel embedding for instance grouping, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9018–9028.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. Imagenet classification with deep convolutional neural networks. Communications of the ACM 60, 84–90.

Le, D., Truong, S., Brijesh, P., Adjeroh, D., Le, N., 2023. scl-st: Supervised contrastive learning with semantic transformations for multiple lead ecg arrhythmia classification. IEEE journal of biomedical and health informatics .

Le, N., Bui, T., Vo-Ho, V.K., Yamazaki, K., Luu, K., 2021. Narrow band active contour attention model for medical segmentation. Diagnostics 11, 1393.

Le, N., Rathour, V.S., Yamazaki, K., Luu, K., Savvides, M., 2022. Deep reinforcement learning in computer vision: a comprehensive survey. Artificial Intelligence Review , 1–87.

Le, T., Gummadi, R., Savvides, M., 2018. Deep recurrent level set for segmenting brain tumors, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer. pp.

646–653.

Le, T.H.N., Luu, K., Zhu, C., Savvides, M., 2017a. Semi self-training beard/moustache detection and segmentation simultaneously. Image and Vision Computing 58, 214–223.

Le, T.H.N., Quach, K.G., Zhu, C., Duong, C.N., Luu, K., Savvides, M., Center, C., 2017b. Robust hand detection and classification in vehicles and in the wild., in: CVPR Workshops, pp. 1203–1210.

Le, T.H.N., Savvides, M., 2016. A novel shape constrained feature-based active contour model for lips/mouth segmentation in the wild. Pattern Recognition 54, 23–33.

Le, T.H.N., Zhu, C., Zheng, Y., Luu, K., Savvides, M., 2017c. Deepsafedrive: A grammar-aware driver parsing approach to driver behavioral situational awareness (db-saw). Pattern Recognition 66, 229–238.

Lee, Y., Park, J., 2020. Centermask: Real-time anchor-free instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, IEEE. pp. 13903–13912. URL: https://doi.org/10.1109/CVPR42600.2020.01392, doi:10.1109/CVPR42600.2020.01392.

Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L., 2022. Dn-detr: Accelerate detr training by introducing query denoising, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13619–13627.

Li, F., Zhang, H., Xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y., 2023. Mask dino: Towards a unified transformer-based framework for object detection and segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3041–3050.

Li, K., Malik, J., 2016. Amodal instance segmentation, in: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, Springer. pp. 677–693.

Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y., 2017. Fully convolutional instance-aware semantic segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society. pp. 4438–4446. URL: https://doi.org/10.1109/CVPR.2017.472, doi:10.1109/CVPR.2017.472.

Lin, M., Chen, Q., Yan, S., 2013. Network in network. arXiv preprint arXiv:1312.4400 .

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, Springer. pp. 740–755.

Lin, X., Yan, Q., Wu, C., Chen, Y., 2022. Judgment model of cock reproductive performance based on vison transformer, in: Proceedings of the 2022 5th International Conference on Sensors, Signal and Image Processing, pp. 37–42.

Liu, S., Li, F., Zhang, H., Yang, X., Qi, X., Su, H., Zhu, J., Zhang, L., 2022. Dab-detr: Dynamic anchor boxes are better queries for detr. arXiv preprint arXiv:2201.12329 .

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B., 2021. Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF international conference on computer vision, pp. 10012–10022.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440.

Luu, K., Zhu, C., Bhagavatula, C., Le, T.H.N., Savvides, M., 2016. A deep learning approach to joint face detection and segmentation. Advances in face detection and facial image analysis , 1–12.

Newell, A., Huang, Z., Deng, J., 2017. Associative embedding: End-to-end learning for joint detection and grouping. Advances in neural information processing systems 30.

Nguyen, P., Quach, K.G., Duong, C.N., Le, N., Nguyen, X.B., Luu, K., 2022. Multi-camera multiple 3d object tracking on the move for autonomous vehicles, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2569–2578.

Nguyen, T.P., Pham, T.T., Nguyen, T., Le, H., Nguyen, D., Lam, H., Nguyen, P., Fowler, J., Tran, M.T., Le, N., 2023. Embryosformer: Deformable transformer and collaborative encoding-decoding for embryos stage development classification, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1981–1990.

O Pinheiro, P.O., Collobert, R., Dollár, P., 2015. Learning to segment object candidates. Advances in neural information processing systems 28.

Park, M., Britton, D., Daley, W., McMurray, G., Navaei, M., Samoylov, A., Usher, C., Xu, J., 2022. Artificial intelligence, sensors, robots, and transportation systems drive an innovative future for poultry broiler and breeder management. Animal Frontiers 12, 40–48.

Quach, K.G., Le, N., Duong, C.N., Jalata, I., Roy, K., Luu, K., 2022. Non-volume preserving-based fusion to group-level emotion recognition on crowd videos. Pattern Recognition 128, 108646.

Ren, G., Lin, T., Ying, Y., Chowdhary, G., Ting, K., 2020. Agricultural robotics research applicable to poultry production: A review. Computers and Electronics in Agriculture 169, 105216.

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

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 .

Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., et al., 2021. Sparse r-cnn: End-to-end object detection with learnable proposals, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 14454–14463.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–9.

Thang Pham, T., Brecheisen, J., Nguyen, A., Nguyen, H., Le, N., 2023. I-ai: A controllable & interpretable ai system for decoding radiologists' intense focus for accurate cxr diagnoses. arXiv e-prints , arXiv–2309.

Tong, K., Wu, Y., Zhou, F., 2020. Recent advances in small object detection based on deep learning: A review. Image and Vision Computing 97, 103910.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H., 2021. Training data-efficient image transformers & distillation through attention, in: International conference on machine learning, PMLR. pp. 10347–10357.

Tran, M., Ly, L., Hua, B.S., Le, N., 2022a. Ss-3dcapsnet: Self-supervised 3d capsule networks for medical segmentation on less labeled data, in: 2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI), IEEE. pp. 1–5.

Tran, M., Vo, K., Yamazaki, K., Fernandes, A., Kidd, M., Le, N., 2022b. Aisformer: Amodal instance segmentation with transformer. British Machine Vision Conference (BMVC) .

Tran, M., Vo-Ho, V.K., Le, N.T., 2022c. 3dconvcaps: 3dunet with convolutional capsule encoder for medical image segmentation, in: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 4392–4398. doi:10.1109/ICPR56361.2022.9956588.

Truong, T.D., Chappa, R.T.N., Nguyen, X.B., Le, N., Dowling, A.P., Luu, K., 2022. Otadapt: Optimal transport-based approach for unsupervised domain adaptation, in: 2022 26th International Conference on Pattern Recognition (ICPR), IEEE. pp. 2850–2856.

USDA, . Poultry-Grading Manual. United States Department of Agriculture.

Van Engelen, J.E., Hoos, H.H., 2020. A survey on semi-supervised learning. Machine learning 109, 373–440.

Vaswani, A., Shazeer, N., et al., 2017. Attention is all you need, in: NIPS, pp. 5998–6008.

Vo, K., Joo, H., Yamazaki, K., Truong, S., Kitani, K., Tran, M.T., Le, N., 2021. AEI: Actors-Environment Interaction with Adaptive Attention for Temporal Action Proposals Generation. BMVC .

Vo, K., Truong, S., Yamazaki, K., Raj, B., Tran, M.T., Le, N., 2022. Aoe-net: Entities interactions modeling with adaptive attention mechanism for temporal action proposals generation. International Journal of Computer Vision , 1–22.

Wray, M., Doughty, H., Damen, D., 2021. On semantic similarity in video retrieval, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3650–3660.

Wu, D., Cui, D., Zhou, M., Ying, Y., 2022. Information perception in modern poultry farming: A review. Computers and Electronics in Agriculture 199, 107131.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P., 2021. Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in neural information processing systems 34, 12077–12090.

Yamazaki, K., Truong, S., Vo, K., Kidd, M., Rainwater, C., Luu, K., Le, N., 2022. Vlcap: Vision-language with contrastive learning for coherent video paragraph captioning, in: 2022 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 3656–3661.

Yamazaki, K., Vo, K., Truong, S., Raj, B., Le, N., 2023. Vltint: Visual-linguistic transformer-in-transformer for coherent video paragraph captioning. AAAI .

Ye, L., Rochan, M., Liu, Z., Wang, Y., 2019. Cross-modal self-attention network for referring image segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10502–10511.

Ying, H., Huang, Z., Liu, S., Shao, T., Zhou, K., 2019. Embedmask: Embedding coupling for one-stage instance segmentation. ArXiv preprint abs/1912.01954. URL: https://arxiv.org/abs/1912.01954.

Zhang, S.H., Li, R., Dong, X., Rosin, P., Cai, Z., Han, X., Yang, D., Huang, H., Hu, S.M., 2019. Pose2seg: Detection free human instance segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 889–898.

Zhang, X., Zhou, X., Lin, M., Sun, J., 2018. Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6848–6856.

Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J., 2017. Pyramid scene parsing network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2881–2890.

Zhao, S., Bai, Z., Wang, S., Gu, Y., 2023. Research on automatic classification and detection of mutton multi-parts based on swin-transformer. Foods 12, 1642.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al., 2021. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 6881–6890.

Zhou, S.K., Le, H.N., Luu, K., Nguyen, H.V., Ayache, N., 2021. Deep reinforcement learning in medical imaging: A literature review. Medical image analysis 73, 102193.

Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J., 2018. Unet++: A nested u-net architecture for medical image segmentation, in: Deep learning in medical image analysis and multimodal learning for clinical decision support. Springer, pp. 3–11.

Zhou, Z.H., 2018. A brief introduction to weakly supervised learning. National science review 5, 44–53.

Zhu, C., Zheng, Y., Luu, K., Hoang Ngan Le, T., Bhagavatula, C., Savvides, M., 2016. Weakly supervised facial analysis with dense hyper-column features, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 25–33.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J., 2020. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 .