Republic of Iraq

Ministry of Higher Education and Scientific Research

University of Babylon

College of Information Technology

Software Department

Human Pose Transfer using Multi-Level Attention Adversarial Generative Networks

A Thesis

Submitted to the Council of the College of Information Technology for Postgraduate Studies of University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology - Software

by

Mohammad Baqer Haleem Mohammad

Supervised by

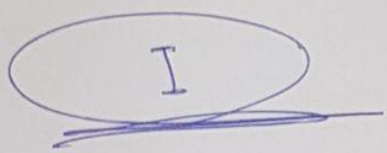
Prof. Dr. Israa Hadi Ali

2023 A.D. **1444 A.H**

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ ﴿1﴾ خَلَقَ الْإِنْسَانَ مِنْ عَلَقٍ
﴿2﴾ اقْرَأْ وَرَبُّكَ الْأَكْرَمُ ﴿3﴾ الَّذِي عَلَّمَ بِالْقَلَمِ ﴿4﴾ عَلَّمَ
الْإِنْسَانَ مَا لَمْ يَعْلَمْ ﴿5﴾

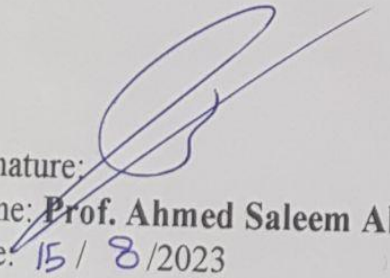
Supervisor Certification

I certify that this thesis was prepared under my supervision at the Department of Software \ Collage of Information Technology \ University of Babylon, by **Mohammad Baqer Haleem Mohammad** as a partial fulfillment of the requirements for the degree of **Master in Information Technology**.



Signature:
Name: **Prof. Dr. Israa Hadi Ali.**
Date: 13 / 8 / 2023

The Head of the Department Certification
In view of the available recommendation, we forward this thesis for debate by the examining committee.



Signature:
Name: **Prof. Ahmed Saleem Abbas.**
Date: 15 / 8 /2023

Certification of the Examination Committee

We, the undersigned, certify that (**Mohammad Baqer Haleem**) candidate for the degree of Master in Information Technology - Software, has presented his thesis of the following title (**Human Pose Transfer using Multi-Level Attention Adversarial Generative Networks**) as it appears on the title page and front cover of the thesis that the said thesis is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on: 20/7/2023.

Signature:

Name: Dr. Nidhal K. El Abbadi

Title: Professor

Date:13/8/2023

(Chairman)

Signature:

Name: Dr. Mehdi Ebady Manaa

Title: Assistant Professor

Date:13/8/2023

(Member)

Signature:

Name: Dr. Wadhah Razooqi Abbood Baiee

Title: Lecturer

Date:13/8/2023

(Member)

Signature:

Name: Dr. Israa Hadi Ali

Title: Professor

Date:13/8/2023

(Member and Supervisor)

Signature:

Name: Dr. Hussein Attia Lafta

Title: Professor

Date: 5/8/2023

(Dean of Collage of Information Technology)

Declaration

I hereby declare that this thesis, submitted to University of Babylon in partial fulfillment of requirement for the degree of Master in Information Technology - Software department, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose source are appropriately cited in the references.

Signature: 

Name: Mohammad Baqer Haleem Mohammad

Date: 13/ 8 / 2023

Dedication

This work is dedicated to…

The martyrs of Iraq of all sects and nationalities.

The one who guides me to the way of God and God gave me guidance on his hands,

my Shafi'i on the Day of Deen,

the Beloved Prophet Muhammad (PBUH) and his successor Imam Ali (AS).

My father and My mother,

I ask Allah to protect you from all evil and I will not disappoint you.

To my beloved brothers and sisters,

who stood by me when things look very difficult.

Acknowledgements

All praise is for Allah. We praise Him, we seek His aid and we ask for His forgiveness. We seek Allah's refuge from the evils of ourselves and our evil actions. Whosoever Allah guides, no one can misguide him; and whosoever Allah misguides, there is no one who can guide him.

As humans, Allah has bestowed on us the nature to be grateful and we should thus express that gratitude not just to Allah but to the people whom we deal with as well. In many places in the Qur'an, Allah divides people as being grateful and as ungrateful to motivate us to join the camp of those who are grateful.

There are no proper words to convey my deep gratitude and respect for my supervisor, **Prof. Dr. Israa Hadi Ali**: I am forever grateful for her patience, guidance, wise counsel and most importantly, she has provided positive encouragement and warm spirit to finish this thesis.

I also want to extend my thanks to my friends, and my second family in the maintenance of irrigation and drainage projects in Babylon: especially, the employees of the Machinery and Equipment Division for their constant support, cooperation, and encouragement at all times.

My beloved mother, thank you for being amazing in so many ways as always. You stood by me through thick and thin and I don't know how to repay you for all the sacrifices you've made.

My deepest gratitude goes to all of my family members. my brothers and sisters for the endless support. It would not be possible to write this thesis without their support.

Last but not least, I would like to thank those who helped me with a word of encouragement or scientific information that benefited me in this thesis: **Ali Saleem Haleem, Ali Saadi Abbas, Mustafa Saleem Al-Saffar, Qusay Odeh Jafar, Ali Majid Najim, Ibrahim Sami Kareem** and all my teachers who have been so supportive along the way of doing my thesis. I thank them wholeheartedly.

Mohammad Baqer Haleem Mohammad

List of Publications

Some of the works presented in this thesis have been published or accepted as listed below.

Paper No.1: Human Pose Transfer Based on Deep Learning Models: A Survey

Abstract

Human Pose Transfer is the process of transferring the appearance of the source image into the pose of the target image. This topic has a diverse variety of applications, including video generation, image-based animation, data augmentation to train human pose estimation, person re-identification, and human parsing models. The concept of human pose transfer can assist in the creation of multi-view images of the same individual. Along with human parsing, human pose transfer tries to generate a new image of a person using the source image and the target pose. While retaining the appearance of the source image.

Keypoints and human parsing were utilized together to generate the target image. The proposed system consists of four phases. The first phase is dataset preparation which includes extracting the keypoints and human parsing for all images. Second phase is preprocessing the images and the features extracted from phase one. Third phase is generating a human parsing map matched with the target pose by the Parse Generator to depict the style of the clothes. Fourth phase is using the Image Generator to transfer the source image to be aligned with the target pose with the help of the generated parsing from the third stage. The proposed Multi-Level Attention Generative Adversarial Network includes two levels of attention the first is pixel-wise represented by Gated-Conv and the second attention level is channel-wise attention represented by Squeeze and Excitation Block. Also, we have used the Wasserstein GAN Gradient Penalty loss function instead of the vanilla loss function.

DeepFashion was used with (101967) training pair of images and (8570) testing pair of images. Research results show that the proposed model outperforms other state-of-the-art approaches. Four metrics were used Frechet Inception Distance (FID), Learned Perceptual Image Patch Similarity (LPIPS), Inception Score (IS) and

Structural Similarity Index Measure (SSIM). The first two metrics are more significant than the rest metrics because they are closer to human judgment. The model has achieved first in FID (10.938**)** and third in the rest while the other metrics (0.2173, 3.475, 0.775) respectively.

Table Of Content

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

Abbreviation	Meaning
2D	Two Dimensional
3D	Three Dimensional
AI	Artificial Intelligence
CGAN	Conditional GAN
CSV	Comma Separated Values
CNN	Convolution Neural Network
FID	Frechet Inception Distance
CV	Computer Vision
GAN	Generative Adversarial Networks
HPE	Human Pose Estimation
IS	Inception Score
IN	Instance Normalization
LPIPS	Learned Perceptual Image Patch Similarity
Leaky ReLU	Leaky Rectified Linear Units
MLA-GAN	Multi-Level Attention Generative Adversarial Networks
MSE	Mean Square Error
MAE	Mean Absolute Error
ML	Machine Learning
PSNR	Peak Signal-to-Noise Ratio
PGN	Part Grouping Network

ReLU	Rectified Linear Units
RGB	Red Green Blue
ResNet	Residual Network
SSIM	Structural Similarity Index Measure
SGD	Stochastic Gradient Descent
SiLU	Sigmoid Linear Units
std	Standard Deviation
SENets	Squeeze-and-Excitation Networks
Tanh	Tangent Hyperbolic
VAE	Variational Auto Encoders
WGAN	Wasserstein GAN
WGAN-GP	Wasserstein GAN Gradient Penalty

Chapter One
Introduction

Chapter One: Introduction

1.1 Overview

Computer vision (CV) is a popular study topic that investigates at how machines and computers may copy human visual systems and gain, extract, understand, and analyze data from videos and images [1]. Computer vision has vital applications to a wide range of industries, including remote sensing, security, biology, engineering, and medicine [2]. Image analysis is a critical activity in this field since it seeks to obtain and interpret useful data from images. For decades, Researchers have explored a variety of image analysis problems, including image segmentation, image feature extraction, image classification, face recognition, object identification, and object tracking [3]. However, Because of a variety of factors such as data difficulty, cost of computing, limited interpretability, an absence of adequate labelled data, data with a large dimension, and substantial variance among images, these tasks remain extremely difficult. As a result, CV is still a fast-expanding field, with many new studies and strategies being offered to efficiently tackle various jobs.

Human Pose Transfer attempts to rebuild a human appearance in different pose given an image of the person. While it is quite easy for humans to visualize what a human might seem in an another body pose, with a single 2D image of a human person, it has been a tough task in computer vision to produce realistic photos based just on the pose. The concept of human pose transfer can assist in the creation of multi-view images of the same individual.

This topic has several applications in video generation, image-based animation, data augmentation for training human pose estimation systems or person re-identification or human parsing models. The production of pose-guided human images is a difficult

task due to the distributions of clothing, body features, backgrounds, and positions in human images vary substantially and one person in various poses might have drastically different visual attributes and the generator network should usually infer the appearance regions of body parts that are not visible in the source image.

The first paper that proposed the idea of human pose transfer was written by Ma et al [4]. Ma used a two stages model. The first stage focuses on pose integration and produces an initial output that captures the human's global structure. While the second stage refines the initial result through adversarial training and creates clearer images.

Ma used 2D representation which is a set of joints and some face-landmarks, previous papers used either 25 keypoints like Karmakar et al [5] and Esser et al [6] or 18 keypoint as the rest of the papers. The methods that used 25 keypoints achieved a better SSIM than the 18 keypoints methods but in trade of additional computation cost. Most of the papers used 18 keypoints representations because it’s simpler, faster and covers the entire body well enough.

Fine-grained spatial deformations cannot be captured by key-point-based models. As a result, distortions or unrealistic characteristics are commonly produced, especially when significant position changes are present. Latest developments used a more comprehensive pose representation rather than the key-point-based one. The latter is to allow for the computation of 'Human parsing' which defines how to shift pixels from the input pose more effectively. As a result, surface-based pose representation is a preferable option.

In this thesis, taking into account the drawbacks of using only the keypoints as the pose representation as mentioned above. Human parsing was used along with the keypoints to help the generator produce better and sharper images.

Although recent advances in deep generative models like Generative Adversarial Networks (GAN) [7] and Variational Auto Encoders (VAE) [8], as well as per-trained model architecture like U-Net [9], human image generation between poses, remains very difficult. In general, effective human pose transfer requires a good representation of human pose and appearance. Most previous papers used one or a mix of the models above like Lathuilière et al [10] used Attention-based U-Net and Li et al [11] adopted U-Net like architecture while Esser et al [6] used a mix of VAE and U-net, but most of the papers used conditional GAN (CGAN). Although Esser et al [6] with the mixed VAE and U-Net model achieved a better result in terms of Structural Similarity Index Measure (SSIM) and Inception Score (IS) than GAN in the papers that only used the 2D pose (Key-Points), the GANs overcame it when the representation is the keypoints along with the human parsing.

A multi-level attention generative adversarial network (MLA-GAN) is presented in this thesis, with the generator being an encoder-decoder architecture that largely relies on the attention mechanism. The generator features two levels of attention: pixel-wise attention, represented by the Gated Conv block, and channel-wise attention, represented by the Squeeze and Excitation blocks.

1.2 Research Problem

The quality, quantity, and relevance of training data determine the performance of most ML models, particularly deep learning models. Unfortunately, one of the most common barriers to implementing machine learning models is a shortage of data. This is because acquiring such data may be expensive and time-consuming in many cases. Data augmentation can help researchers reduce their reliance on training data collection and preparation, allowing them to develop more accurate machine learning models faster, such as re-identification, human pose estimation, and human parsing datasets.

1.3 Challenges

Much research has dealt with the topic of human pose transfer, and the main challenges of this problem are as follows:

a- The distributions of clothing, body appearance, backgrounds, and positions in human images vary greatly.

b- A person in different stances may have significantly diverse visual characteristics.

c- In most cases, the generative model must infer the appearance features of body components that are not visible in the input images.

1.4 Related Works

The related work is illustrated in the following review a number of the studies on literature approaches and table 1.1 shows a summary of the papers:

Siarohin et al [12] in 2021 proposed Deformable GANs where in the generator deformable skip connections are used to "shuttle" local information from the generator's encoder to the decoder. Where they take the feature maps of the encoder and transfer them to match the target by affine transformation and then concatenate the result with the decoder as in a normal skip connection. They also advocated replacing typical pixel-to-pixel losses (e.g., L1 or L2 losses) in conditional generative techniques with a nearest-neighbor loss.

According to Khatun et al [13] in 2021, the condition image and condition pose are initially supplied into the appearance encoder and pose encoder, respectively, to build the appearance map and the pose map. The pose attention-guided appearance network receives both the appearance and the pose map. The output, along with the desired pose, is sent into the pose attention-guided generation network to generate the final image.

Jiang et al [14] proposed BPA-GAN in 2021 which contains three generators, two of which are part of the generation modules in charge of dealing with the head and other body parts individually; both are encoder-decoder networks with skip links but employ distinct parametric values. The encoders may extract features from multiple positions since the training data is obtained from a video that typically contains frames of varied viewpoints and poses of the target person. Furthermore, the target's appearance is encoded in the network while being monitored by real images. Another generator is programmed to combine the body sections generated by the previous two generators and scaled by the mid-output module into the final outcome. Zhang et al [15]'s technique in 2021 comprises two generators: a parser generator and an image generator. A human parse image aligned with the target pose is estimated by the parsing generator. This enables controllable image manipulation of the final image's shape. Based on the human parsing map, the image generator creates a high-quality image of the reposed person. They presented combining global and local per-region encoding and normalizing to decouple the style and shape to offer detailed control of the styles in particular regions. They detached the shape and style of clothing using the created parsing map and per-region style control to make image editing tasks easier.

In 2020, Lathuilière et al [10] proposed a generalization for the person-image generation challenge. A human image is generated from a target position and a collection of many source photographs. This allows for the use of various potentially complementary images. They developed an attention-based decoder, which extends the U-Net architecture to a multiple-input situation by incorporating an attention mechanism that picks meaningful information from numerous sources and image areas.

Men et al [16] proposed in 2020 pose encoding and deconstructed part encoding are two approaches for embedding the desired pose and input image in a latent space.

They extract component attributes using a human parser and encode them with a global texture encoder for the latter. A succession of style blocks equipped with a fusion module is presented to inject the texture style of the source person into the pose code by altering the affine transform variables in adaptive instance normalization (AdaIN) layers. Ultimately, a decoder is utilized to rebuild the desired image.

Zhang et al [17] in 2020 presented a framework for adaptive hierarchical deformation for person's pose transfer. First deformation step creates human semantic parsing that is aligned with the target pose, and the second deformation level creates the final textured person image with semantic guiding in the target pose. Furthermore, while vanilla convolution is unsuitable for unaligned generation tasks, they used gated convolution to dynamically choose essential features and modify the image layer by layer.

Song et al [18] in 2019 introduced an unsupervised method in which the training set consists of a source image together with its pose and segmentation in the absence of a corresponding ground-truth image. The main concept is to use human parsing to break it down into two models: semantic parsing transformation and appearance generation. The semantic parsing transformation module seeks to produce a semantic image under the goal pose first, which offers an important prior for the person shape and garment properties. The appearance creation module then generates textures for the desired image, guided by the expected semantic map and the reference image. To achieve cycle consistency, they employ the final output, together with its pose and segmentations, to construct the source image, as in Cycle GAN.

In 2018, Zhao et al [19] proposed VariGAN. It comprises three modules: a coarse image generator, a fine image generator, and a conditional discriminator. The coarse image generator creates a low-resolution (LR) image conditioned by the desired

person, conditioned image, and desired view during training. The fine image generator with skip connections creates a high-resolution (HR) image. Lastly, the HR image and the conditioned image are joined as fake pairs and sent into the conditional discriminator alongside real pairs (target image and condition image) to identify between real and fake.

Soft-Gated Warping-GAN was proposed by Dong et al [20] in 2018. The model creates target parsing using a pose-guided parser first, given a condition image and a target pose. Then, using a geometric matcher and a soft-gated warping-block to warp the image features, estimate the transformations between the condition and target parsing. The warped feature maps, embedded pose, and synthetic parser were then concatenated to form a realistic-looking image.

Ma et al [4] want to separate the appearance and structure components in person photographs so that they may change the foreground, backdrop, and pose independently. They proposed a two-stage pipeline in 2017 to do this. In Stage I, they use a reconstruction network to divide and conquer the foreground, backdrop, and position elements. They recreate human images, in particular, by first disentangling them into intermediate embedding characteristics of the three variables, and then recovering the original image by decoding these features. In step II, they take these features as actual to develop mapping functions for adversarially translating a Gaussian distribution to the embedding feature distribution.

Table 1.1: Related Works Summary

References	**Pose Representation**	**Dataset**	**Number of Human Parsing Classes**	**Methodology**
[12]	KeyPoints	DeepFashion, Market-1501	-	Deformable GANS

[13]	KeyPoints	Market-1501, DukeMTMC-reID	-	conditional DCGAN and U-Net
[14]	KeyPoints + Human Parsing	They gathered the training data from Internet videos	23	Body-Parts-Aware Generative Adversarial Network
[15]	KeyPoints + Human Parsing	DeepFashion	8	Decoupled GAN
[10]	KeyPoints	DeepFashion, Market-1501	-	Attention-based U-Net
[16]	KeyPoints + Human Parsing	DeepFashion	8	Attribute-Decomposed GAN
[17]	KeyPoints + Human Parsing	DeepFashion	12	GAN with Gated Conv
[18]	KeyPoints + Human Parsing	DeepFashion Market-1501	12	Model like Cycle-GAN
[19]	KeyPoints	DeepFashion, MVC 1	-	Variational GAN
[20]	KeyPoints + Human Parsing	DeepFashion, Market-1501	20	Soft-Gated Warping-GAN
[4]	KeyPoints	DeepFashion, Market-1501	-	GAN

1.5 Aim of The Thesis

The main aim of this thesis is to generate a person image based on two separate factors: the appearance of a particular person in a certain image and the same or different person's pose in another image. The main aim is divided into three objectives:

The first objective is to extract keypoints and human parsing maps for all the images in the dataset. The second objective is to build a model that is capable of generating

a human parsing for the given image aligned with the conditioned pose. Third objective is to construct another model to transfer the appearance of the input image to align with the target pose.

1.6 Thesis Contributions

The framework of this thesis was presented by a few previous papers but the contribution is to build a new network architecture for the GAN generator also the WGAN-GP loss function will be used instead of the traditional loss function of GAN.

1.7 Thesis Organization

This thesis includes five chapters and each chapter begins with an overview and then some details about the problem that this thesis addresses, as follows:

The first chapter explains a general introduction to the problem and its history, in addition to the reasons and Aim of the thesis. The content of the other chapters is as follows:

Chapter Two: Presents a comprehensive description of the main principles of "theoretical background", which were used later in this thesis.

Chapter three: Under the title of “The proposed system”. Explains the proposed system and the algorithms used in it.

Chapter Four: This chapter clarifies the results of the proposed system and research experiments. It also discusses the evaluation of the system's performance.

Chapter Five: mentioned precisely the conclusions, it includes what was discovered in this thesis, and the potential future research directions to improve this work.

Chapter Two
Theoretical Background

Chapter Two: Theoretical Background

2.1 Overview

Deep learning is the most interesting field in machine learning, and neural networks and now deep learning presents solutions which are the most efficient for a broad range of issues in image, audio, and text recognition, time series issues, video analysis, and natural language processing. Because deep learning is a smarter way than shallow algorithms of machine learning. However, it is not a simple task. Deep neural networks are more difficult in training, and for better and faster results, they require graphics processing support and large datasets [21]. The purpose of this chapter is to introduce the fundamental information and concepts that are employed throughout this thesis to achieve the Human Pose Transfer task. In section 2.2 human pose will explained how to extract it from images furthermore the approach used in this thesis will explained. In Section 2.3 human parsing will be introduced and the approach have been used in this thesis will be shown. Section 2.4 Deep learning will be introduced along with its types, activation functions, loss functions, optimizers and a gentle overview of CNNs, data augmentation and GANs. Squeeze and excitation will be explained in section 2.5. Finally, section 2.6 will be about the evaluation metrics that have been used to evaluate the results in this thesis.

2.2 Human Pose Estimation (HPE)

Pose estimation is a computer vision job that allows robots to recognize and analyze human figures in films and photographs. It assists machines in determining where the human knee is in an image, for example. Pose estimation is concerned with predicting the placement of important body joints and is incapable of recognizing an individual's identity in a video or image [22].

Pose estimation models can aid in the tracking of an object or person (including several persons) in real-world environments. They are superior than object detection models in some circumstances, which may find things inside an image but only give coarse-grained localization with a bounding box framing the item. In comparison, pose estimation models estimate the precise position of the key points linked with a specific object [22].

A pose estimation model's input is often a processed camera image, and its output is key point information. A keypoint ID and a confidence score from 0 to 1 are assigned to the discovered key points. The confidence score's job is to reflect the likelihood that a crucial point exists in that given place.

Because the position of persons in an image is unclear, multi-person pose estimation presents a substantial issue. Here are two methods for resolving this issue [22]:

a- Top-Down Approach

Top-down pose estimation operates by initially recognizing candidates for people in the image, after which the section within the bounding box is evaluated of every identified human to anticipate human joints. For instance, an algorithm that can serve as a human detector.

The top-down method has various disadvantages:

• The accuracy of the pose estimator is heavily dependent on human detection results; the pose estimator is usually particularly reactive to the person boundary boxes found in the image.

• The time needed to execute the algorithm rises in proportion to the number of people detected in the image, making it time-consuming to run.

b- Bottom-Up Approach

Bottom-up pose estimation recognizes all human joints in a image and then assembles them into a pose for each individual. Zhe Cao et al. [23] suggested OpenPose in 2019.

It is a bottom-up approach in which the algorithm first detects body parts or keypoints in the image and then maps related keypoints to produce pairings.

CNN is the fundamental architecture of OpenPose, as seen in Figure (2.1). A VGG-19 network is used to extract patterns and representations from the input data. The VGG-19 output is routed into a pair of convolutional networks. The first network predicts a collection of confidence maps for each body component, whereas the second predicts a set of component Affinity Fields (PAFs), which provides a degree of linkage between parts. To obtain the final skeleton shape, they prune the weak linkages in the bipartite graphs [23].

Essentially, the predictions from the two branches, together with the attributes, are concatenated for the subsequent step to form a human skeleton, depending on the number of persons in the input. CNNs are used in stages to improve prediction [23].

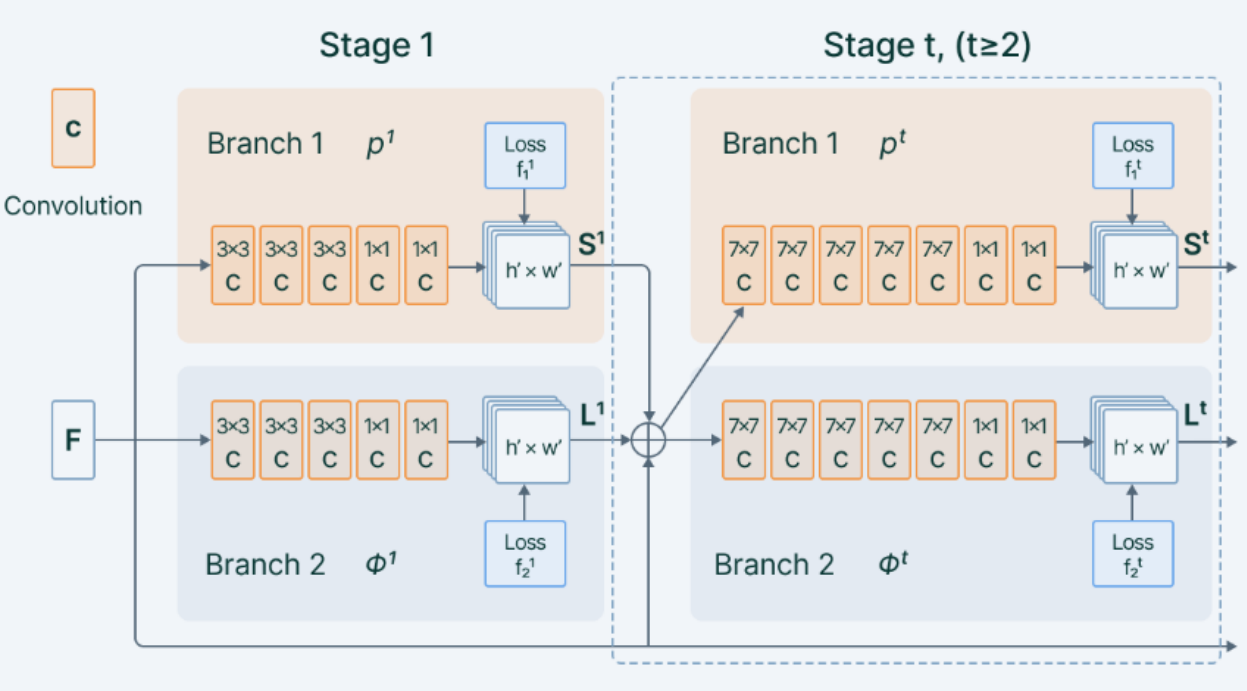


Figure 2-1 OpenPose Architecture [23].

2.3 Human Parsing

Complete person visual comprehension of situations in the real world, is considered one among the primary challenges in CV, it has the potential to make a huge influence. On a wide range of application, including human behavior analysis, automatic recommendation of products, person re-identification, and video surveillance. Human parsing is a semantic segmentation challenge that seeks to detect person images pixel by pixel, assigning every single Pixel to a relevant class, such as arms, hair, clothes, and so on. Finally, as seen in Figure 2.2, these pixels combine to generate a human parsing outcome. Human parsing aids in interpreting the semantic data of different sections of the person body in the image and gives a more complete comprehension of image contents, whereas human pose estimation focuses on finding the precise placements of essential body joints. Human parsing and position estimation are two crucial and associated activities in interpreting

human images by giving pixel-wise knowledge as well as high-level joint structures [24] [25].

Figure 2-2 Human Parsing Example [26].

Gong et al [26] suggested the Part Grouping Network (PGN) Model in 2020. In a unified model, PGN trains and refines semantic part segmentation and instance-aware edge detection. Both of these subtasks are pixel-by-pixel classification issues that Fully Convolutional Networks (FCNs) excel at. PGN is defined as FCNs. It first acquires a similar representation utilizing common intermediate layers before adding a pair of concurrent routes for semantic component segmentation and edge detection. A refinement branch is also provided to make two targets mutually helpful for each other by leveraging complementary contextual information to examine and exploit the semantic linkage of these two activities. Finally, an efficient partition technique with a heuristic grouping strategy may be used to provide instance-level human parsing results by performing a breadth-first search over line segments obtained by jointly scanning the constructed semantic component segmentation maps and instance-aware edge maps.

2.4 Deep Learning

Deep learning is a field of artificial intelligence and both are elements of machine learning [27]. It consists of hierarchical architecture, where each higher layer builds on its previous lower layer, which makes it popular for the first time as hierarchical learning [28]. The beginning of deep learning was in 2007. Deep learning works well with a massive amount of data to solve a complicated problem. Many applications improved with deep learning such as Object Recognition, Object detection, Automatic Machine Translation, Investment Modeling, and drug discovery.

Deep learning can extract high-level features from the input data. For example, in image processing, deep learning can be used in several applications such as edge extraction, image segmentation, shape recognition, and face verification.

Deep learning algorithms such as the Convolution Neural Network (CNN) boost their efficiency on the conventional algorithms due to the automated function learned without the need for human computational effort as it does in conventional machine learning techniques. The conventional machine learning technique requires a feature vector corresponding to the raw data (intensity pixel values in the case of an image) as an input to detect and process it so the features must be extracted and later fed into a classifier. Deep learning algorithms can learn these features on their own [29].

2.4.1 Deep Learning Types

There are three main types of deep learning models, each has its specific network architectures inside and has specific use.

a- Supervised Deep Learning

In Supervised Deep Learning algorithms, the algorithm is trained on pre-labelled data. Then the model is used the learning algorithm to adjust itself, and during the testing phase, the model should determine the correct answer without relying on any label. Supervised deep learning has two main fields: classification problems and regression problems. One of the most frequently supervised models used is a convolution neural network (CNN). Some of the CNN-based supervised deep learning architectures are AlexNet, LeNet-5, VGGNet, GoogleNet, ResNet, InceptionNet, and others [30] [31].

b- Unsupervised Deep Learning

In the unsupervised Deep Learning algorithms, training the model based on unlabeled data, and the model tries for extracting patterns and features on its own. Some of the unsupervised deep learning architectures are Deep Belief_Network (DBN), Restricted_Boltzmann_Machine (RBM) and Generated Adversarial Networks (GANs) [28].

c- Semi-Supervised Deep Learning

Semi-Supervised Deep Learning takes on an intermediate stage between supervised and unsupervised learning. It takes a lot of categorized data to support a larger collection of unlabeled data. This approach is especially useful when it is difficult to extract relevant data features, and when labelling the samples takes a long time [32].

2.4.2 Activation Functions

An Activation Function determines if a neuron is activated or not. This means that during the prediction phase, it will employ simpler mathematical operations to decide if the neuron's input to the network is necessary or not. The function of the

Activation Function is to create output from a set of input values presented to a node (or a layer) [33]. The most five activation functions that are important for use in hidden layers:

a. Rectified Linear Activation (ReLU)

Rectified Linear Units (ReLU). Widely used in Neural networks especially Deep learning models, it works by setting the threshold at 0, Simply stated, it produces 0 when x is zero or a negative value and returns the same value in another state [34] the function represented by equation (2.1). Figure (2.3) represents the ReLU Activation Function plot.

$$ReLU(x) = \max(0, x) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

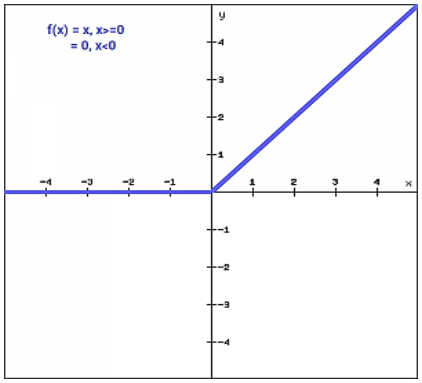


Figure 2-3 ReLU Activation Function [35].

b. Leaky Rectified Linear Unit

Leaky ReLU is a sort of activation function built around the ReLU activation function that instead of a level slope, features a slight slope for negative values. The slope factor is specified prior to training rather than during training. This type of activation function is commonly used in sparse gradient applications, such as training generative adversarial networks [36] the function represented by the equation (2.2). Figure (2.4) represents the LeakyReLU Activation Function plot.

$$\boldsymbol{LeakyReLU}(\boldsymbol{y}) = \begin{cases} \boldsymbol{ay} \ \ \boldsymbol{for\ x < 0} \\ \boldsymbol{y} \ \ \boldsymbol{for\ x \geq 0} \end{cases} \quad \ldots\ldots \ (2.2)$$

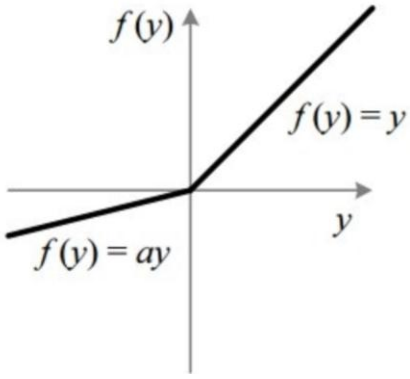


Figure 2-4 Leaky ReLU Activation Function [36].

c. Sigmoid

Since sigmoid is a non-linear function, it is the most commonly employed especially in binary classification. The Sigmoid activation function transforms the value in the [0 – 1] range [33]. As shown in Figure (2.5), it can be summed up as equation (2.3): $f(x) = 1/(1+e^{-x})$ ……. (2.3)

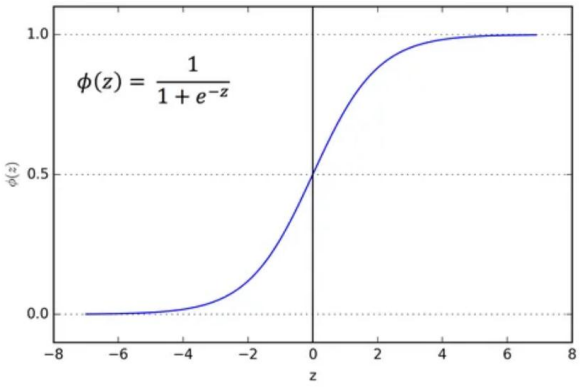


Figure 2-5 Logistic (Sigmoid) Activation Function [33].

d. Hyperbolic Tangent (Tanh)

It's a Tangent Hyperbolic function. The Tanh function resembles the sigmoid activation function, However, it is symmetric across the origin. As a

consequence, the outputs from previous layers would have different signs when supplied as input to the following layer [33]. It's written as equation (2.4):

$$f(x) = 2 * \text{Sigmoid}(2x) - 1 \quad \text{.....(2.4)}$$

The Tanh activation function is a continuous and differentiable function with values ranging from -1 to 1 as shown in Figure (2.6). The Tanh function has a steeper gradient than the Sigmoid function.

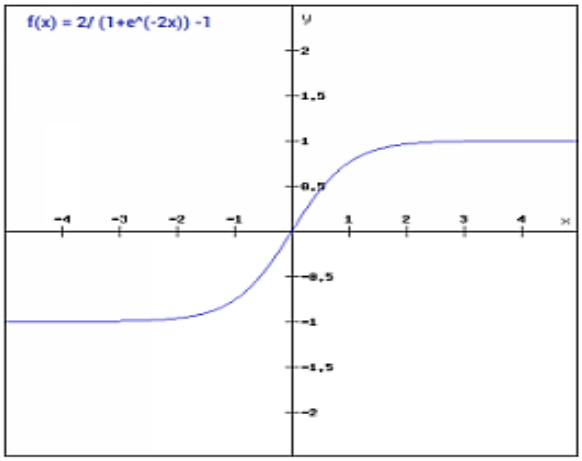


Figure 2-6 Hyperbolic Tangent (Tanh) Activation Function [33].

e. Softmax

The softmax function is one of the activation function types, it is widely used in neural computing at the last layer to calculate the multiple probability distributions of multi classes of more than two classes by using a vector of real numbers. The output of Softmax function values in the range between 0 and 1, where the summation of the probabilities is equal to 1, and the target class which have the highest value, the equation (2.5) represent the Softmax function [37].

$$f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \text{..... (2.5)}$$

2.4.3 Loss Function

In early models of neural networks, the error is measured by calculating the difference between the real output and the expected output. At the moment, several formulas have appeared for calculating errors in neural networks, these formulas are called Loss Functions [38]. When using various loss functions can lead to a different error value for the same prediction, therefore the type of loss function has a major impact on the output of the network. There are three main loss function types: Classification Loss Functions, Regression Loss, and Embedding Loss Functions [39]. The classification loss functions use with classification problems. The Regression Loss functions are used in regression problems. While the Embedding loss functions use with tasks that need to measure the similarity between two inputs [40].

a- Binary Cross Entropy

The binary classification issues in deep neural networks, the objective function that is the binary cross entropy equation, as in logistic regression shown in equation (2.6), where y_i and $\hat{y}_i$ are the true labels and the estimated output probability from the neural network respectively [41].

$$l = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \quad \text{..... (2.6)}$$

b- Categorical Cross-Entropy

This loss function uses in Multi-class classification problems. In these problems, the target can only belong to one out of many possible categories. This function is designed to calculate the difference between two probability distributions [41]. The following equation (2.7) illustrates the categorical cross-entropy.

$$L(X_i, Y_i) = -\sum_{j=1}^{c} y_{ij} * \log(p_{ij}) \quad \text{..... (2.7)}$$

$$where\ Y_i\ is\ one-hot\ encoded\ target\ vector\ (y_{i1}, y_{i2}, \ldots, y_{ic})$$

$$y_{ij} = \begin{cases} 1, & if\ the\ i_{th} element\ is\ in\ class\ j \\ 0, & otherwise \end{cases}$$

$$p_{ij} = f(X_i) = Probability\ that\ i_{th} element\ is\ in\ class\ j$$

c- Mean Square Error Loss Function

MSE is one of the important regression loss functions, which measures the square difference between the real and expected values [41]. Equation (2.8) illustrates the mean square error loss.

$$\boldsymbol{MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2} \quad \text{..... (2.8)}$$

Where $\boldsymbol{y_i}$ actual target vector, $\boldsymbol{p_i}$ the output predicted vector,n vector length.

d- Euclidean Distance Loss

This loss function is an embedding loss usually used in problems that need to measure the similarity between two inputs, not for classification problems. It measures the distance between two distinct points or two vectors [41]. The following equation (2.9) illustrates the Euclidean Distance Loss.

$$\textbf{Euclidean loss} = \sqrt{\sum_{i=1}^{n}(\ f(y_i) - p_i\ )^2} \quad \text{..... (2.9)}$$

Where $\boldsymbol{p_i}$ is the predicted vector

$\boldsymbol{f}(\mathbf{y_i})$ is the actual input vector

n is the length of the vector.

e- Mean Absolute Error (MAE) / L1 Loss Function

L1 is calculating the absolute difference between the real and the generated image for one image while the MAE is the average of L1 values of all the dataset [42].

For Image Enhancement, MAE will most likely produce an image that looks to be of greater quality to a human viewer. The L1 Loss function is shown in the following equation (2.10):

$$L1 - LossFunction = \sum_{i=1}^{n} | y_{true} - y_{predicted} | \quad \ldots\ldots\ldots\ldots (2.10)$$

Where y_{true} is the real image and the $y_{predicted}$ is the generated image.

f- Perceptual Loss

Perceptual loss functions are used when evaluating two similar photos, such as the same photo but shifted by one pixel. The function is utilized to evaluate high-level variations across images, such as differences in content and style. A perceptual loss function is extremely similar to a per-pixel loss function since both are used to train feed-forward neural networks for image modification tasks. The perceptual loss function is a more extensively used component because it delivers more precise results for style transfer [43]. The equation of perceptual loss can be defined as equation (2.11)

$$L_{percep} = \sum_{i=1}^{N} \alpha_i \; || \phi_i (I_g) - \phi_i (I_t) ||_1 \; \ldots (2.11)$$

where ϕi (Ig) indicates the feature map of the i-th (i = 0,1,2,3,4) layer VGG-19 for the generated image Ig. And ϕi (I_t) indicates the feature map of the i-th (i = 0,1,2,3,4) layer in VGG-19 for the real image I_t.

g- Gradient Penalty Wasserstein GAN (GP-WGAN) Loss Function

GAN training is difficult. Mode collapses are common, and models may never converge. To proceed, either incremental improvements are made or embark on a new road for a new cost function.

The Wasserstein distance (Earth Mover's distance) is a metric for measuring the distance between two probability distributions on a particular metric space. It may be viewed intuitively as the least amount of effort required to change one distribution to another, where work is defined as the product of the mass of the distribution to be shifted and the distance to be transported [44].

The original GAN objective is shown to be the minimization of the Jensen-Shannon (JS) Divergence [4]. Compared to JS, Wasserstein distance has the following advantages:

- The Wasserstein Distance is continuous and almost differentiable everywhere, allowing the model to train to optimality.
- JS Divergence locally saturates when the discriminator improves, so the gradients become zero and disappear.
- Wasserstein distance is a meaningful metric in the sense that it converges to 0 as the distributions grow closer together and diverges as they move further apart.
- The Wasserstein Distance objective function is more stable than the JS divergence goal function. When Wasserstein distance is used as the goal function, the mode collapse problem is also minimized.

WGAN introduces a critic instead of the discriminator usually seen with GANs. The critic network is similar in design to a discriminator network but predicts the Wasserstein distance which can be written as in equation (2,11).

$$min_G\ max_{D\ ||f||_l \leq 1}\ E[f(x)] - E[f(\tilde{x})] \ldots\ldots\ldots\ldots\ldots\ldots (2.11)$$

Where X is the real data while x̃ is the generated data. Here, to enforce Lipschitz continuity on the function f, the authors resort to restricting the weights w to a compact space. This is done by clamping the weights to a small range ([-1e-2, 1e-2] in the WGAN paper [44]).

The difference between the discriminator and the critic is that the discriminator is trained to correctly identify samples of the ground truth distribution from samples of the fake distribution, the critic estimates the Wasserstein distance between the real distribution and the fake distribution.

The weight clipping technique employed to force Lipschitz continuity on the critic is at the root of WGAN's difficulties. WGAN-GP swaps weight clipping with a constraint on the critic's gradient norm to ensure Lipschitz continuity. This makes it possible for greater stability in network training than WGAN and necessitates less hyperparameter tuning. Gradient Penalty aims to establish a constraint in such a way that the gradients of the critic's outcome with regard to the input have a unit norm. The authors of WGAN-GP proposed a soft form of this constraint in which the gradient norm on the samples x Px is penalized. The new aim is depicted in Equation (2.12).

$$\ell = E_{\tilde{x}\sim P_g}\,[f(\tilde{x})] - E_{x\sim P_r}\,[f(x)] + \lambda\, E_{\hat{x}\sim P_{\hat{x}}}\,[(||\nabla_{\hat{x}} f(\hat{x})||_2 - 1)^2]\ \ldots (2{,}12)$$

The component to the left of the total in the preceding calculation is the initial critic loss, while the term to the right of the sum is the gradient penalty [44].

Px is the distribution derived by sampling uniformly along a straight line between the real and generated distributions Pr and Pg. Because the optimum critic has straight lines with unit gradient norms between the samples linked from Pr and Pg,

this is done. , on the other hand, is the penalty coefficient that is used to weight the gradient penalty term. The authors set =10 for all of their experiments in the publication.

Batch Normalization is no longer utilized in the critic since it translates a batch of inputs into a batch of outputs. In this scenario, calculating the gradients of each output in relation to its corresponding inputs is needed [44].

2.4.4 The Optimization Algorithms

One of the most essential phases is choosing the algorithm to use to optimize a neural network. The Main types of optimization methods in machine learning are batch or deterministic gradient methods, which handle all training instances in a big batch at the same time, and stochastic methods, which work with only one instance at a time and mini-batch which takes a specific number of the training instances at a time [39].

There are two types of optimization algorithms: algorithms of adaptive learning and algorithms with constant learning rates, such as SGD [45]. In the first group, the learning rate η is chosen manually. The task of choosing the learning rate in this type of algorithm is somewhat difficult. When choosing a relatively small learning rate, the learning process is slowed, and the training time becomes too large. While choosing a relatively large learning rate, can lead to fluctuation in the loss value around the minimum value, and this hinders the convergence process. While the algorithms of the second group, they do not need to set the learning rate manually, they have a heuristic approach that takes care of adjusting the learning rate value, several algorithms appeared that belong to these two categories, the most important of which are illustrates as follows:

a- Stochastic Gradient Descent (SGD)

SGD is one of the Constant Learning Rate Algorithms that try to update the network parameters more repeatedly. In this algorithm, the network parameters are modified on each training sample after the loss calculation. Therefore, if the dataset contains 900 samples, the network parameters will be modified 900 times in one cycle of the dataset. The SGD is used to update parameters in a neural network [45].

***b*- Adam**

Adam is considered one of the adaptive learning rate optimization algorithms, it measures individual learning rates for various parameters [45]. Adam sets the learning parameter automatically, this was done by using the first and second-moment estimation. The moment is the expectation of a random variable at the power of n [46]. The moment can illustrate in equation (2.13).

$$m_n = E[X^n] \quad \text{..... (2.13)}$$

Where: m is the_moment, and X is a random_variable.

The following equations are used to estimate, the first and second moment of Adam.

$$\widehat{\boldsymbol{m}}_{\boldsymbol{t}} = \frac{\boldsymbol{m_t}}{\mathbf{1-\beta_1}} \quad \text{.....(2.14)}$$

$$\widehat{\boldsymbol{v}}_{\boldsymbol{t}} = \frac{\boldsymbol{v_t}}{\mathbf{1-\beta_2}} \quad \text{..... (2.15)}$$

Where $\boldsymbol{m_t}$ is the previous first moment and $\boldsymbol{v_t}$, is the previous second moment and they are initialized with 0 in the first step.

β1 and β2 are new parameters inserted into the algorithm. They have default values of 0.9 and 0.999 respectively.

After calculating the value of the first and second moments, it is used to update the network weights according to the following equation (2.16) [46].

$$w_t = w_{t-1} - \eta \frac{\widehat{m}}{\sqrt{\widehat{v_t}}+\epsilon} \quad \ldots\ldots\ldots\ldots (2.16)$$

Where W is network weights, η is Stepsize, $\boldsymbol{\epsilon = 10^{-8}}$

2.4.5 Convolution Neural Network (CNN)

CNN is a type of deep discriminative architecture that has been demonstrated to be effective in processing two-dimensional data such as images and videos, with a grid-like layout. The architecture of CNNs is based on brain cortical architecture in animals. During the 1960s, CNNs is the first successful architecture for deep learning Because of the excellent training of the hierarchical layers. The CNN architecture takes advantage of spatial relationships to reduce the parameter number of the network, which improves performance when typical BP algorithms are used. The growth of computation techniques and GPUs-accelerated, have been used to train CNNs more effectively. The CNN performs two functions, the extraction function then the prediction function. Each function used certain layers to achieve a particular purpose [47].

a- Convolution Layer

A convolution layer is a fundamental component of the CNN architecture that performs feature extraction, which typically consists of a combination of linear and nonlinear operations, i.e., convolution operation and activation function.

Convolution is a specialized type of linear operation used for feature extraction, where a small array of numbers, called a kernel, is applied across the input, which is an array of numbers, called a tensor. An element-wise product between each element of the kernel and the input tensor is calculated at each location of the tensor and summed to obtain the output value in the corresponding position of the output tensor, called a feature map (Figure 2.7 a–c). This procedure is repeated applying multiple kernels to form an arbitrary number of feature maps, which represent different characteristics of the input tensors; different kernels can, thus, be considered as different feature extractors. Two key hyper parameters that define the convolution operation are size and number of kernels. The former is typically 3 × 3, but sometimes 5 × 5 or 7 × 7. The latter is arbitrary, and determines the depth of output feature maps [48].

b- Gated Convolution

Vanilla convolutions are commonly employed in convolution neural networks, which have made significant advances in object identification, image segmentation, and image-to-image translation.

The vanilla convolution layer output values are computed using the same filter in all spatial locations. It accepts all pixels as acceptable values and uses a sliding window to extract local features, which is useful for object identification, image segmentation, and alignment generation tasks [49].

However, for misalignment tasks, such as human pose transfer, vanilla convolution features do not necessarily have a good influence on the result. As a result, learning a dynamic feature selection process to modify the image is increasingly critical [49].

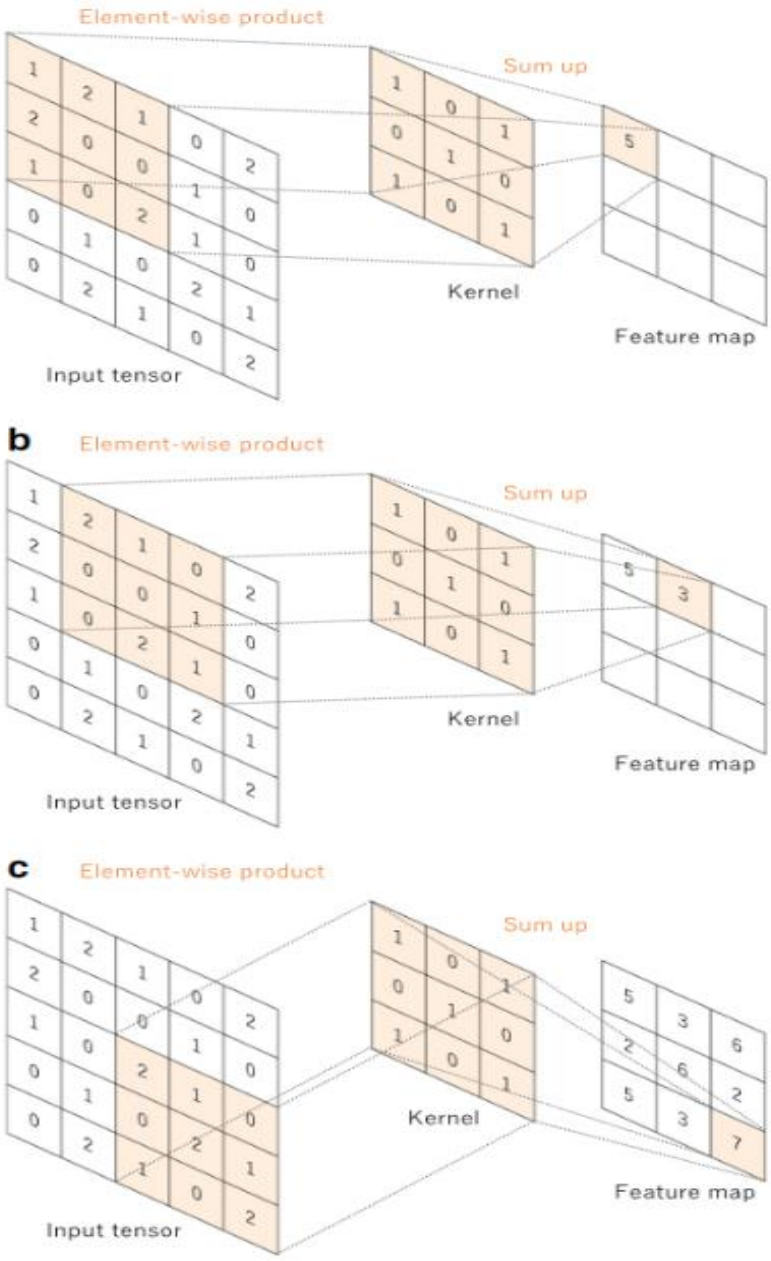


Figure 2-7 An example of convolution operation [48]

While the Gated Conv is dynamically selecting the important pixels by passing the input into two streams the first is the vanilla convolution and the other stream will be the score for each corresponding pixel in the first stream as illustrated in Figure (2.8).

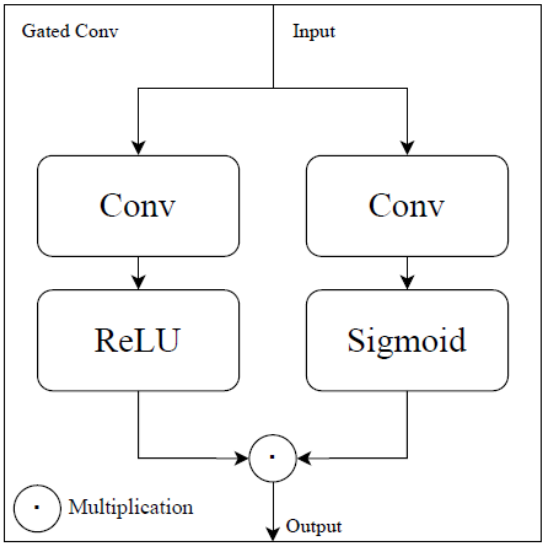


Figure 2-8 Gated Convolution

c- Regularization Layers

Overfitting is one of the important issues that face the network model, this issue occurs in both following cases: when a model is complex, and when a dataset is poor. The overfitting causes difficulty in generalizing the model on the unseen samples, therefore, the accuracy of the model will be high during the training phase and low during the testing phase. The regularization layers are one of the important methods to avoid the overfitting problem. The most two used regularization layers can be illustrated as follows:

- Dropout Layer

Dropout is a technique that addresses the overfitting issue. The word "dropout" refers to removing (both hidden and visible) units. Where dropping a unit out in a neural network means logically removing it or in other words disconnecting its incoming and outgoing connections in the network. These units are chosen randomly based on value [49].

- Batch Normalization Layer Activation

Deep networks need layers of normalization and activation functions as building blocks for stable generalization improvement and optimization. Batch normalization is one of the techniques that regularize the neural network and avoid overfitting. It makes to solve the Internal Covariate Shift problem which is defined as the changes in the distributions of inputs to the current layer due to the changes in the parameters of the preceding layer. Readjusting the current layer to new distribution constantly is required accordingly, in order to increase training performance. Normalizing the output of the previous layers by fixing it to be zero mean and one standard deviation (std) [50]. Suppose input data x, batch normalization of it is as the following equation (2.17):

$$X_1 = \frac{x - mean}{std} \quad \ldots\ldots\ldots\ldots (2.17)$$

By continuing training, the network, layer by layer, it becomes more intelligent to recognize shapes difficult than those recognized by the previous layer. Where the beginner layers are capable to recognize colors, lines, corners, etc. The later layers are capable to recognize components of shapes and the overall shapes.

- Instance Normalization

Instance normalization (IN) is a word that was initially used in the StyleNet paper to refer to contrast normalization. Both names provide some details about this method. The term "instance normalization" indicates that it acts on a single sample. Contrast normalization, on the other hand, claims to equalize the contrast between a sample's spatial parts. IN conducts intensity normalization over the width and height of a single feature map of a single sample when given a Convolution Neural Network (CNN). To execute

instance normalization on a single instance, the mean and variance must be computed. The calculated mean and variance are then used to normalize each spatial dimension [51].

2.4.6 Data Augmentation

Data augmentation is the process of intentionally increasing the quantity of data by creating more data points from current data. Making minor modifications to data or using machine learning models to generate additional data points in the original data's latent space to boost the dataset are examples of this. The distinction between data augmentation and synthetic data is that synthetic data is created artificially and does not employ real-world images. Generative Adversarial Networks (GAN) are widely used to produce synthetic data, whereas augmented data originates from original images with slight geometric adjustments (such as flipping, translation, rotation, or noise addition) to improve the variety of the training set [52] [53].

The performance of a Deep Learning model is determined by two factors:

• Neural Network Model

• Data Quality and Quantity

Even when a suitable model is used, satisfying outcomes are not always achieved. The issue then becomes the data utilized to train the network. A huge dataset is critical for the deep learning model's success. However, there may be a lack of the quantity and diversity of data needed to train the model for a specific demand, resulting in poor performance.

Data augmentation has several advantages, including:

• Improving model prediction accuracy.

• Contributing extra training data to models.

• Reducing data shortages for improved models.

• Minimizing data overfitting and introducing unpredictability in data.

• Enhancing model generalization capacity.

• Assisting in resolving class imbalance concerns in the classification.

• Lowering data collection and labelling expenses.

• Enabling rare event prediction.

2.4.7 Generative Adversarial Networks (GANs)

Generative adversarial networks (GAN) are algorithmic frameworks that put two neural networks competing with one another (Therefore, the expression "adversarial") to generate fresh, synthetic data samples that may be passed as real data. They are often used in the creation of photos, films, and music.

GAN works by simultaneously training a discriminator and a generator in two networks, as shown in Figure (2.9). The discriminator is trained to differentiate among real dataset samples and fake samples created by the generator. The generator is trained to create false samples from an easy-to-sample random source so that the discriminator cannot distinguish between actual and fake data samples [7].

Because GANs may mimic any data distribution method, they have immense potential for both good and harm. GANs may be trained to create worlds that are uncannily similar to every domain: images, audio, and speech. In other words, GANs may be used to generate phony media content, and this is the technique behind Deep Fakes [54].

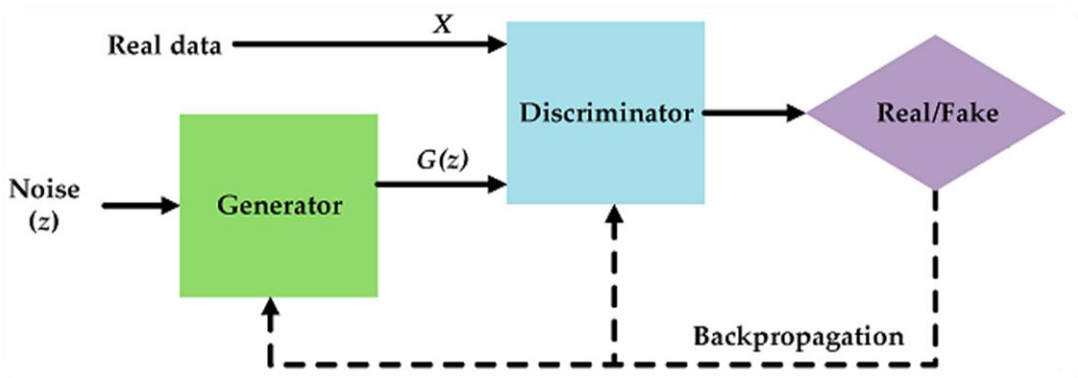


Figure 2-9 Simple Architecture of GAN network [55].

There is another type of GAN Conditional generative adversarial networks in which both the generator and the discriminator are conditioned by some extra information y.

The condition might be any type of additional information, such as class labels or data from other modalities. Conditioning may be accomplished by including y as an extra layer of input of each of the generator and the discriminator. In the generator, the prior input noise Pz(z) and y are paired to generate a joint hidden representation, and the adversarial training framework makes it possible to tremendous flexibility in how this hidden representation is formed. x and y are presented as inputs to a discriminative function in the discriminator [56].

The objective function of a two-player minimax game would be as in the formula (2,18):

$$min_G\, max_D\, V(D,G) = E_{x\sim P_{data}(x)}\,[\log D(x|y)] + E_{z\sim P_z(z)}\,[\log\left(1 - D\left(G(z|y)\right)\right)] \quad \ldots. (2{,}18)$$

Where x is the real image and z is a latent vector.

D is the discriminator and G is the generator.

$D(x|y)$ is the discriminator output of the real image conditioned on y.

$D(G(x|y))$ is the discriminator output of the generated image conditioned on y.

Figure (2.10) illustrates the structure of a simple conditional adversarial net.

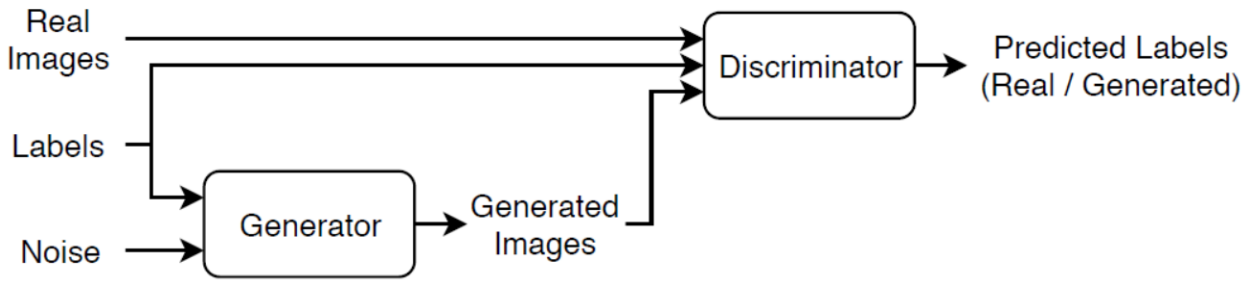


Figure 2-10 structure of a simple conditional GAN [57].

2.5 Squeeze and Excitation

Squeeze and Excitation networks [58] present a constructing component for convolutional neural networks that increases layer interdependencies at nearly no extra expense of calculation. They provide significant performance gains and are simply integrated into current designs. The basic concept is to add variables to every channel of a CNN block so that the model may adaptively change each feature map's weight.

Convolutional filters are used by CNNs to gain hierarchical data from images. Upper layers may distinguish persons, characters, or other sophisticated geometrical structures while lower levels identify unimportant background features like as edges or high frequencies. They obtain anything is essential to efficiently perform an objective. Everything of this occurs because merging spatial and channel information from a image. Before merging the data's overall potential output channels, the various filters first will check for spatial components for every input channel [58].

When constructing the output feature maps, the model equally weights all its channels. SENets aim to change that with including a content-aware method to adaptively weight every channel. In its simplest form, this may imply assigning a

single variable to every channel and assigning a linear scalar to how important each of them is. The next stages describe SE blocks, as shown in Figure (2.11):

1- It averages pools every channel to one numeric value provided the input block and the present the amount of available channels.

2- Nonlinearity is added by a completely connected layer followed by a ReLU activation function. It also decreases the outcome channel's complexity by a specific factor.

3- A smooth gating function for each channel is provided by a second totally connected layer, then a Sigmoid function.

4- Finally, depending on the results of the side network, it weights each convolutional block feature map.

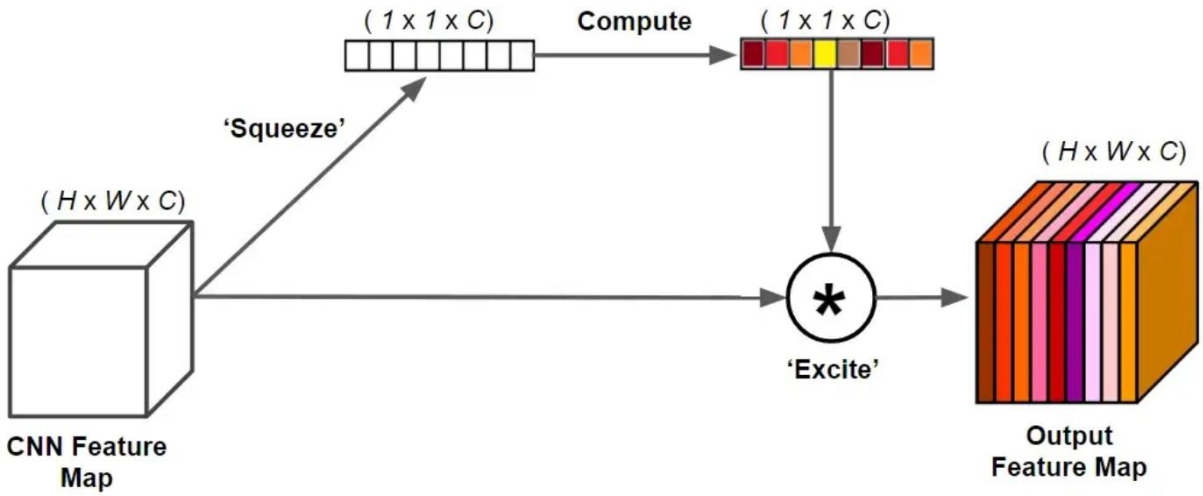


Figure 2-11 Squeeze and Excitation Mechanism [58]

These four procedures add nearly no processing cost (less than 1% more) and may be applied to any model.

The authors of SE demonstrate that by including SE-blocks into ResNet-50, the results will be nearly identical quality as ResNet-101. This is amazing for a network that just requires fifty percent the computing expenditures.

2.6 Evaluation Measures

When developing a machine learning model, it is critical to evaluate performance and efficiency. In order for the machine learning model to be dependable, an assessment tool that is appropriate for the nature of the model's work must be chosen. When evaluating machine learning models, it is common to utilize more than one scale to verify that the model is correctly evaluated. The primary machine learning assessment metrics for GANs are:

2.6.1 Inception Score (IS)

The Inception score (IS) is a prominent statistic for evaluating Generative Adversarial Networks (GANs) image outputs. A GAN is a network that learns how to create fresh unique images that are comparable to the training data. The IS accepts an image list and returns a single floating point value. The score indicates how realistic the output of a GAN is. It is an automated alternative to having people judge image quality. The score considers two factors at the same time: the diversity of the images and the distinct appearance of each image. The score is considered high if both of these requirements are satisfied. The score will be low if any or both of these claims are false. It is ideal to get a higher score. This means that your GAN may generate a large variety of images. Zero is the lowest possible score. The greatest possible score in mathematics is infinite.

Tim Salimans et al. [59] suggested the Inception score. In the research, the authors of IS analyzed a huge number of GAN-produced photos using a crowd-sourcing platform (Amazon Mechanical Turk). They created the inception score to eliminate the subjective human appraisal of visuals. The authors found that their scores were highly associated with subjective judgment.

2.6.2 Structural Similarity Index Measure (SSIM)

The structural similarity index measure (SSIM) [60] estimates the perceived quality of digital television, cinema, and other forms of digital images and videos. To compare the similarities between the two images, the SSIM technique is utilized. The SSIM index is a full baseline metric, which implies that image quality is assessed or projected using a reference image that is uncompressed or distortion-free. The SSIM index originated from prior methodologies that were demonstrated to be incompatible with human perceptual physiology, such as PSNR (peak signal-to-noise ratio) and the MSE method. The difference between MSE and PSNR is that these methods evaluate absolute errors.

The premise behind structural information is that pixels have a lot of interdependence, especially when they're close together. These dependencies communicate important information about the organization of the components in the visual scene [54]. The SSIM score is depicted in equation (2,19).

$$\boldsymbol{SSIM(x.y)} = \frac{(2\mu_x\mu_y + \boldsymbol{c_1})(2\sigma_{xy} + \boldsymbol{c_2})}{(\mu_x^2 + \mu_y^2 + \boldsymbol{c_1})(\sigma_x^2 + \sigma_y^2 + \boldsymbol{c_2})} \quad \ldots\ldots (2,19)$$

Where

- μ_x is the pixel sample mean of x.
- μ_y is the pixel sample mean of y.
- σ^2_x is the variance of x.
- σ^2_y is the variance of y.
- σ_{xy} the covariance of x and y.
- $c_1=(k_1 L)^2$, $c_2=(k_2 L)^2$ are two variables to stabilize the division with a weak denominator.
- L is the dynamic range of the pixel values (typically this is $2^{\#\text{bits per pixel}}$ -1).
- $k_1 = 0.01$ and $k_2 = 0.03$ by default.

2.6.3 Learned Perceptual Image Patch Similarity (LPIPS)

LPIPS approach is utilized to compare the perceptual likeness of two images. LPIPS calculate the likeness of two image patch activations for an identified model. This metric was discovered to correctly represent human perception. Perceptually, image patches with low LPIPS scores are comparable [61].

The LPIPS metric is based on the use of so-called deep features in the construction of a loss function. The measure has several conceivable versions, including lin, tune, and scratch. Pre-trained network weights (F) are fixed in lin, but linear weights (w) are learned. A pre-trained categorization model is employed for tuning, and all network (F) weights are adjusted. For the last variation, scratch, a network is trained using judgment from connected research and initialized with random Gaussian weights [66]. The distance between a reference patch x and a distorted patch x0 determined with a given network (F) is provided by equation (2,20).

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||w_l \cdot (\hat{y}^l_{hw} - \hat{y}^l_{0hw})||_2^2 \quad \ldots\ldots (2{,}20)$$

where $\hat{y}^l$ and $\hat{y}^l_0$ are extracted deep embeddings from layer l, w_l is a scaling vector and (H, W) the spatial components [61].

2.6.4 Frechet Inception Distance (FID)

FID score [62] computes the distance of feature vectors obtained from real and generated images.

The score sums up how comparable both of the groups are in terms of statistics on CV aspects of raw images extracted from the inception v3 image categorization network. The lower scores imply that the two groups of images are more comparable, or that their statistics are more similar, whereas an ideal score of 0.0 indicates that the two groups of images are equivalent.

Lower FID values have been demonstrated to correlate better with better-quality photos when utilized for assessing the quality of images created by generative adversarial networks.

The inception value does not account for how generated and actual visuals compare. The FID score was created with the objective of evaluating synthetic images by considering the statistics of a collection of generated photos vs the statistics of a collection of actual image from the desired domain [62].

Chapter Three
The Proposed System

Chapter Three: The Proposed System

3.1 Overview

This chapter describes the main stages of the practical aspect of the thesis and the methodology used to construct the proposed system. The proposed system includes several phases to be able to perform a task that is capable of transferring a person's image into a specific given pose. By first aligning the human parse of the source person with the target pose and then using it to transfer the appearance of the input image to align with the target pose with another generator.

3.2 System Framework

The proposed system includes four phases as shown in Figure 3.1. The first phase is preparing the dataset, two features are extracted from each image by pretrained models the features are Keypoints Pose and Human Parsing images. The second phase applies the necessary pre-processing on the image and the two features that have been extracted from phase one. The third phase takes the source parse image and transfers it to align with the target pose with a specific generator called Parse Generator. The fourth and last phase takes the source image and transfers the appearance of it to align with the target pose with help from the result of phase three.

3.2.1 Phase 1: Dataset Preparation

In this phase, the dataset is prepared to make it suitable for Human Pose Transfer task by extracting two features from each image by pre-trained models.

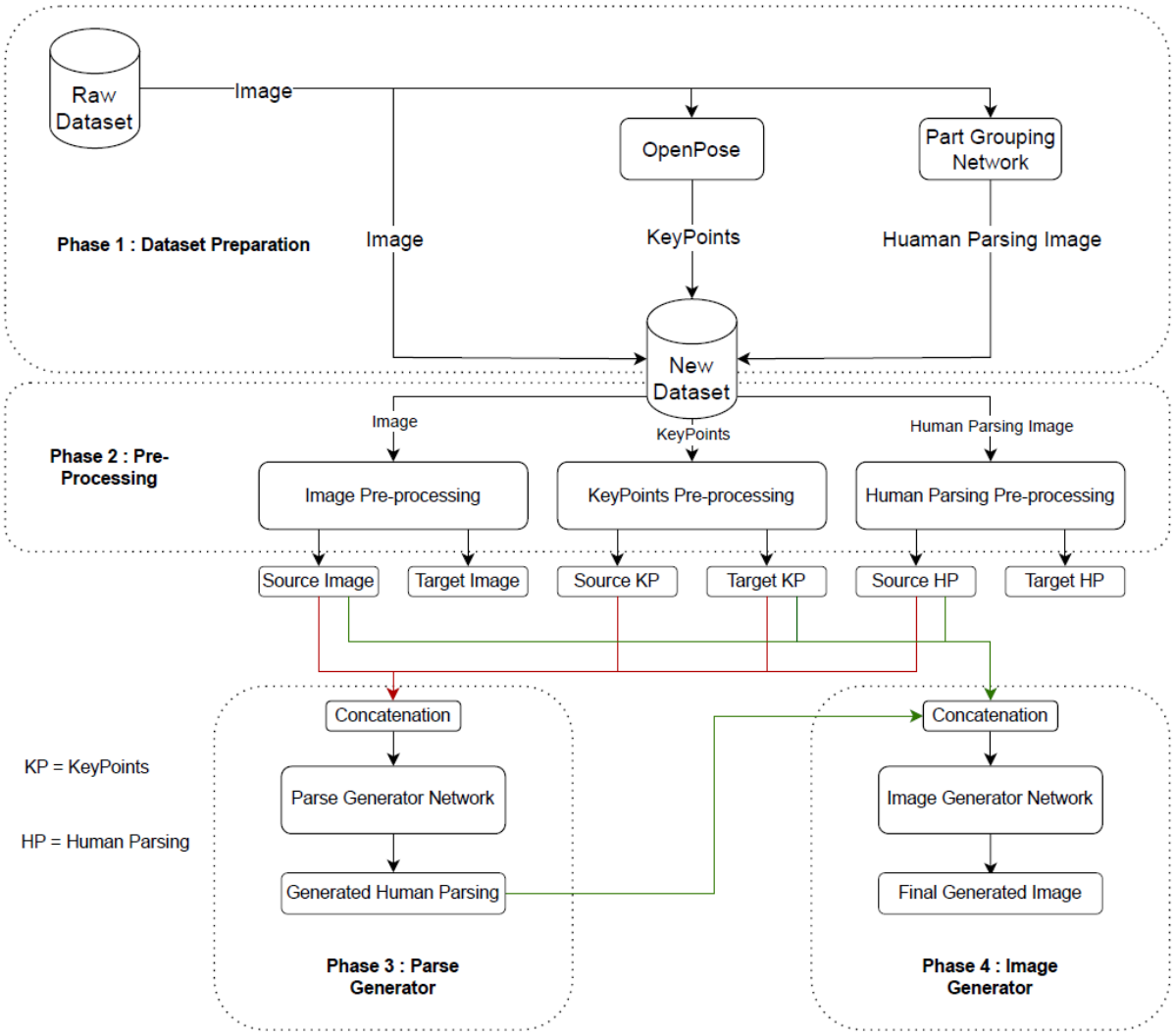


Figure 3-1 The Proposed System Framework

3.2.1.1 Dataset Organization

The images should be set as pairs where two images of the same person are taken and paired so one of them will be the input image and the other one will be the target image. The pairs are saved as an Excel file with two columns the first column is the paths of the source images and the second column is the paths of the target images. The final files are a collection of (101967) training pairs and (8570) testing pairs.

3.2.1.2 Feature Extraction

The proposed system requires KeyPoints and Human parsing for each image and since the dataset doesn't provide these features thus this features will have to be

extracted. First, the pose (Keypoints) is extracted using a pretrained Human Pose Estimation (OpenPose) model. Second, the human parse (Semantic segmentation for humans) of each person is extracted also using a pretrained Part Grouping Network PGN model as illustrated in Figure 3.2.

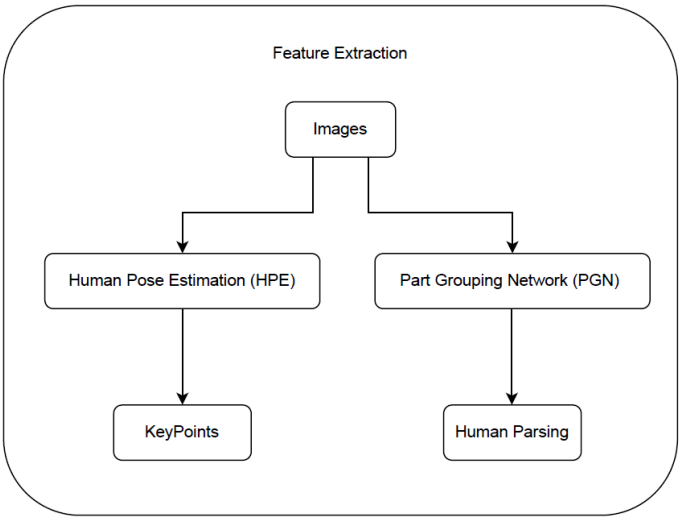


Figure 3-2 Feature Extraction

a- Human Pose Extraction

Since the Human Pose Transfer task is transferring a person's image conditioned on the pose and the dataset doesn't provide the pose for each image the pose will have to be extracted for each image using a pre-trained model. Human Pose Estimator (OpenPose) will be applied to extract 18 key-point as pose representation these key-points are (nose, neck, right shoulder, right elbow, right wrist, left shoulder, left elbow, left wrist, right hip, right knee, right ankle, left hip, left knee, left ankle, left eye, right eye, left ear, right ear) as shown in Figure 3.3 and Algorithm 3.1.

(a) (b)

Figure 3-3 Poses detected using Human Pose Estimator (a) Dataset Images (b) poses extracted using HPE

Algorithm 3.1: Human Pose Extraction
Input: Dataset Images.
Output: a CSV file contains 18 (x, y) points for each image in the dataset. Each point represents a joint or a face landmark and the missing joint will be (-1).
Variables: KeyPointsList: a list will be holding all the images KeyPoints. KeyPoints: KeyPoints is a set of sets each one of the sets represents a point and contains the x coordinate and the y coordinate. Subset: a set shows the index of the available keypoints and -1for the missing ones. xList: a list will be containing all the x coordinates of all the keypoints. yList: a list will be containing all the y coordinates of all the keypoints.

```
Begin
    For image in dataset:
        xList ← -1 // initializing a list with all its values as -1
        yList ← -1 // initializing a list with all its values as -1
        KeyPoints, Subset = OpenPose(Image) // Sending the image into the OpenPose Model
                                            // returning the keypoints and the Subset
        For ID in Subset:
          If (ID = -1):   continue
          else
              x, y = KeyPoints[ID] // KeyPoints are (x, y) coordinates
              add x to xList[ID]
              add y to yList[ID]

        KeyPointsList.append([ImageName, yList, xList])
        Header = [“ImageName”, “KeyPoint_y”, “KeyPoint_x”]
        with open ('ImageAnnotations.csv', 'w') as file: // make a csv file to add the keypoints
              write the header row only once
              write the KeyPointsList into the csv with each image taking row
End
```

b- Human Parsing Extraction

Most of the published papers about Human Pose Transfer are only using key-points as a condition to transfer the person’s image into the target pose although it’s working very well but key-points only provide so little information about the pose and don’t cover the shape of the body this may lead to unsharp images. To solve this issue, another pose representation will be added along with the key-points that will be Human Parsing. The process of segmenting a human image into fine-grained semantic elements such as the head, chest, arms, and legs is known as human parsing as in Figure 3.4. PGN will be used to produce human segmentation maps with 20

labels (Background, Hat, Hair, Glove, Sunglasses, Upper-clothe, Dress, Coat, Socks, Pants, torso-skin, Scarf, Skirt, Face, Left-arm, Right-arm, Left-leg, Right-leg, Left-shoe and Right-shoe).

Because the human parser's projected 20 segmentation maps contain some unclear regions like (left arm and right arm), the classes will be rearranged into 12 classes:

1. background (sunglasses)
2. hat
3. hair
4. upper clothes (scarf, coat)
5. dress
6. pants
7. neck (torso-skin)
8. skirt
9. face
10. hands (left arm, right arm, gloves)
11. legs (left leg, right leg)
12. shoes (left-shoe, right-shoe, socks).

(a) (b)

Figure 3-4 Parsing maps extracted using Part Grouping Network (a) Dataset Images (b) segmentation images extracted by PGN

The process of extracting the human parsing using PGN and the reorganization of the classes is illustrated in algorithm 3.2.

Algorithm 3.2: Human Parsing Extraction
Input: Dataset Images.
Output: Human Parsing map for the input image, in the image each pixel is classified into specific classes and Reorganize them into 12 classes.
Variables: from_label: list of the original classes. to_list: list of the reorganized classes. Parse: the output of the PGN where each class is classified into a specific class.
Begin

```
from_label = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
to_label = [0, 33, 1, 8, 0, 2, 3, 2, 10, 4, 5, 2, 6, 7, 8, 8, 9, 9, 10, 10]
For image in dataset:
    Parse = PGN(Image) // pass the image to PGN model to get the labeled image
    img = cv2.imread(Parse) // read the image
    For label in range(len(from_label)):
        Parse [Parse == from_label[label]] = to_label[label] // reorganize the labels
    Parse [Parse == 33] = 11 // swap every pixel with value 33 to 11
    cv2.save(Parse) // save the reorganized image
```

End

3.2.2 Phase 2: Pre-processing

Pre-processing is an essential part of every system, the image and the two features that have been extracted in phase one the key-points and the human parsing will be prepared to be ready to enter the generator and the discriminator models.

3.2.2.1 Image Pre-processing

Image pre-processing includes all the images in the dataset. First, the images will be cropped and then normalized and finally filtered.

a- Images Cropping

For the image pre-processing first the unnecessary parts of the images are cropped, which will not only reduce the computation time due to fewer pixels being used but also affects the quality of the generated image also due to removing the unnecessary parts of image. The original image is a size of (256, 256) as shown in Figure 3.5.

Figure 3-5 The original image from the dataset before cropping

The left and right grey regions will be removed from the image by cropping them. Figure 3.6 shows the image after the image is cropped it resulting in shape (256, 176).

Figure 3-6 The Image after Cropping

b- Normalization

After cropping the image, the images should be converted to tensors so they can be used in training also the images will be scaled from the [0, 255] range to the [0, 1] range since the computation of large numeric values may become more difficult when the images are used as they are and send them through a Deep Neural Network.

Thus scaling the images is done therefore the numbers will be small and the computation becomes easier and faster.

The next step is to normalize the images because data normalization is a critical process that guarantees each input pixel gets the same distribution of data. This speeds up convergence while training the network. Data normalization involves taking away the mean of each of the pixels and divide the outcome by the standard deviation. Such data would have a distribution resembling a Gaussian curve centered at zero. Also since GAN is used in this thesis and the GAN models for images always end with a Tanh activation function which ranges the output pixels between -1 and 1. The input images need to be normalized so it has the same range as the output images to be more accurate when calculating the loss functions. After scaling down the image's pixel value from [0, 255] to [0, 1] and normalizing it with 0.5 mean and 0.5 std the result range will be [-1, 1] which is the optimal range when training GANs.

c- Dataset Filtering

Another step in pre-processing is the excluding of the "non-person" images in the dataset there are some images where it only shows clothes as in Figure 3.7 and no human in the image so these images will be removed since they are useless for Human Pose Transfer task, this process will be done after extracting the key-points for each image where it will be filtered on a simple condition. The condition is the image has to at least have these 4 key-points to be a valid image, these keypoints are (Left Shoulder, Right Shoulder, Left Hip, Right Hip).

Figure 3-7 Only cloth images in the dataset

3.2.2.2 Key-points Pre-processing

In terms of the key-points, they will also have to be prepared to be ready to enter the generator and the discriminator models. The output from the pre-trained HPE model is just 18 (x, y) points each coordinate indicate a certain key-points thus every key-point will be projected in a separate channel resulting in a total of 18 channel each channel will be a same shape as the cropped image (256, 176) as shown in Figure 3.8. Dividing each key-point into a different channel gives us more control over the pose and easier training since each key-point will have its own channel not just a small region in one channel.

3.2.2.3 Human Parsing Pre-processing

Human parsing (semantic segmentation for humans) also has to be prepared to be ready to enter the generator and the discriminator models. The output gotten from applying the pretrained Part Grouping Network model on the images is an image where each pixel is classified into one of the 12 classes (background, hat, hair, upper clothes, dress, pants, neck, skirt, face, hands, legs, shoes). For more control over the given human parse image it will be divided where each class has its own channel same as the key-points resulting in 12 channels and each channel have the same

shape as the cropped images (256, 176). Dividing the images into separate channels based on the classes not only gives us more control and easier training but also giving us the ability to transfer each region independently for future generator architecture.

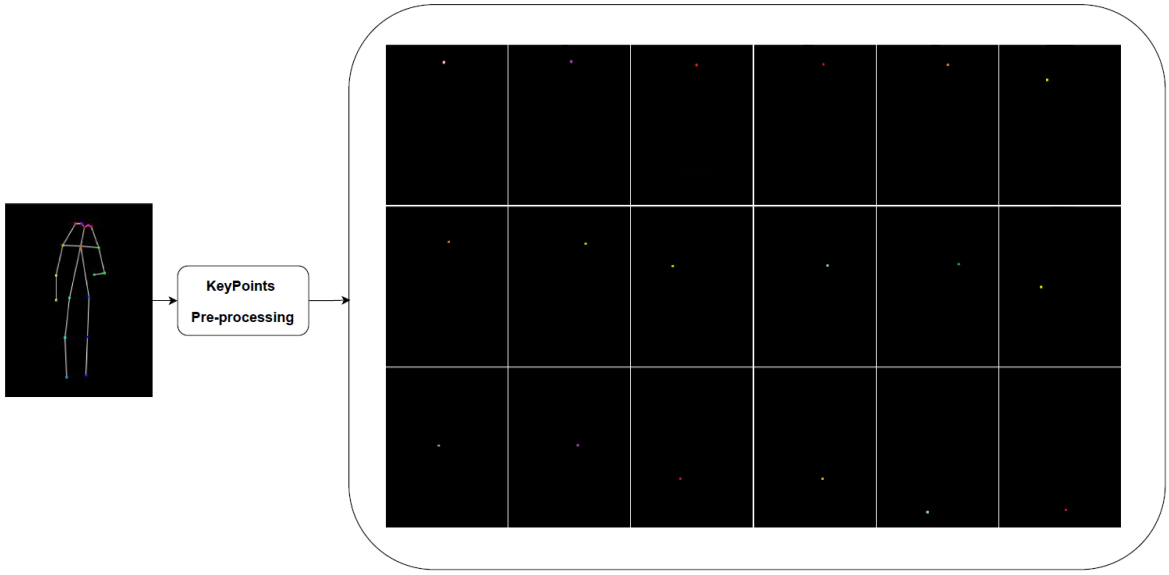


Figure 3-8 Keypoints Pre-processing

3.2.3 Phase 3: Parse Generator

Human Pose Transfer is transferring the appearance of the source image to align with the target image conditioned on the pose and the human parsing, in human parsing there might be a big difference between the source and target images like the source person is tall, wearing different cloths, fat or slim, it may differ significantly so source human parsing should also be transferred to align with the desired pose also to maintain the original image's appearance and shape. An encoder-decoder architecture was designed and called the Parsing Generator to perform this task. Table 3.1 shows the architecture of the generator (C_{out} is 12 in the parsing generator one channel for each part). The inputs for the parse generator are the input image, source pose, source parse image and the target pose as shown in Figure 3.9, with these inputs the generator learns to generate the target parsing map.

Two loss functions have been used the first one is the reconstruction loss represented as the L1 distance loss between the original parse image and the desired parse image, the equation of the L1 loss function is in equation (2.10).

The second is loss is categorical cross-entropy, which is used to motivate the parse generator to generate better-quality parse images. The equation for the categorical cross-entropy loss function is in equation (2.7).

where M_t is the ground truth parsing map and M_g is the generated parsing map and N is the number of label classes (in this research, N = 12) and S is softmax activation function.

The total loss function of the parse generator is the addition of both of the loss functions with equal weights and it can be formulated as equation (3.1)

$$L_{Parsing} = L_{log} + L_{\ell 1} \ldots\ldots (3.1)$$

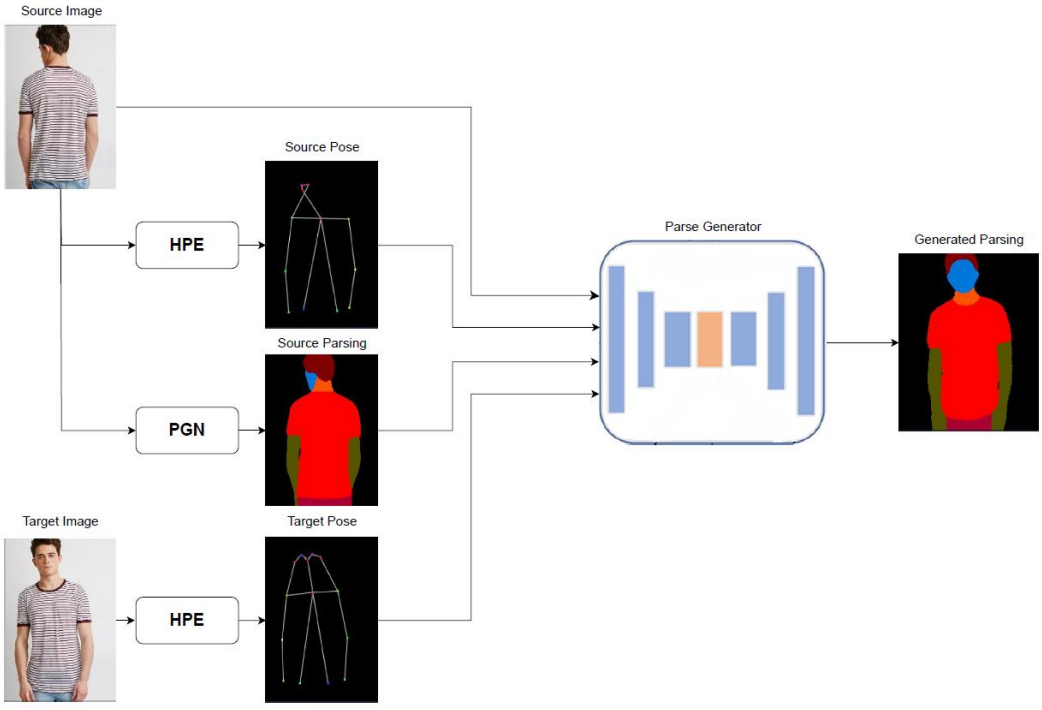


Figure 3-9 Parse Generator

Table 3.1: The network architecture (C_{out} is the number of output channels).

Operation	Stride	Output Shape
Input	-	(256,176,47)
GatedConv	2	(128,88,64)
GatedConv	1	(128,88,128)
GatedConv	2	(64,44,256)
GatedConv	1	(64,44,256)
ResBlock	-	(64,44,256)
ResBlock	-	(64,44,256)
ResBlock	-	(64,44,256)
GatedDeconv	1	(128,88,256)
GatedConv	1	(128,88,128)
GatedDeconv	1	(256,176,64)
GatedConv	1	(256,176,32)
Conv	1	(256,176, C_{out})

3.2.4 Phase 4: Image Generator

The Last and final phase in the proposed system is generating the target image which should be aligned along the conditioned pose and the generated human parse which generated from phase 3 the Parse Generator. The same encoder-decoder architecture has been adopted as the parsing generator as shown in Table 3.1 (C_{out} is 3 in this generator because it outputs a colored RGB image), but obviously with different weights because it's trained to do a different task. The input image, source parse image, target pose and the output of the parse generator (generated human parse image) are concatenated then fed to the image generator outputting an image with the input image appearance and the target image pose as shown in Figure 3.10.

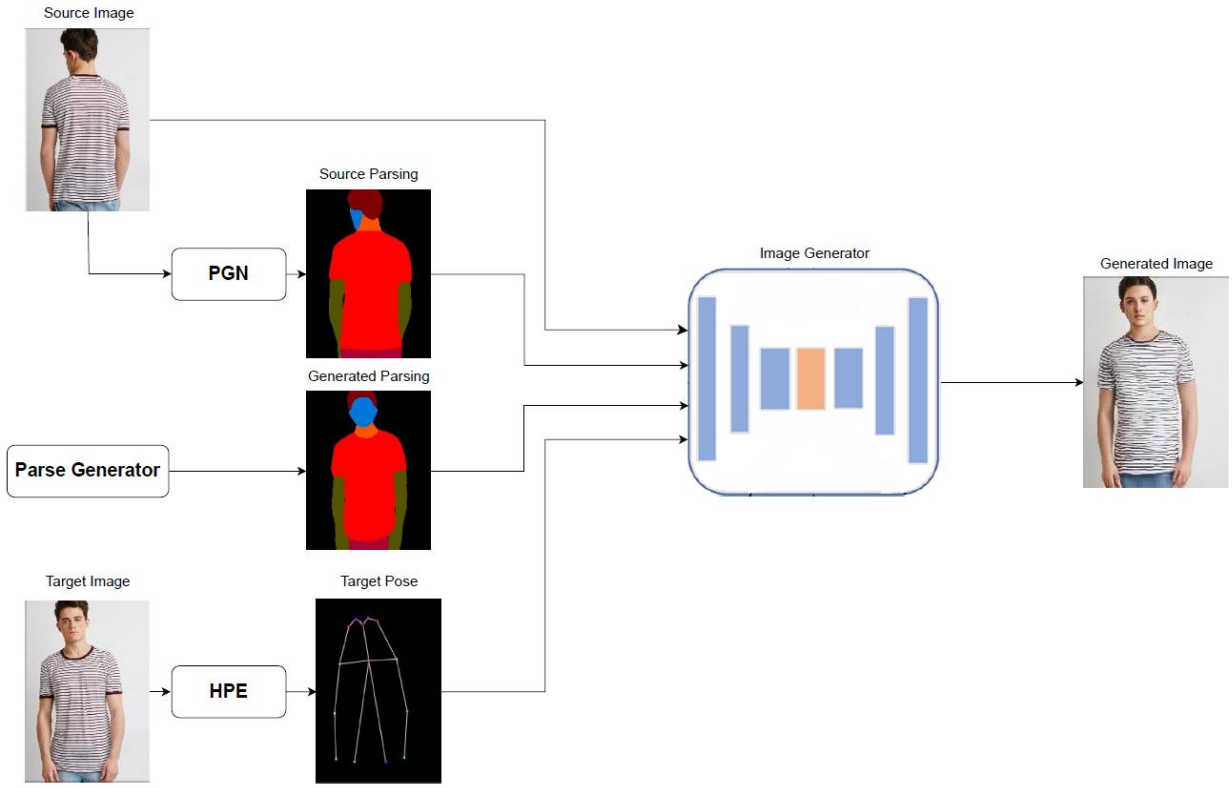


Figure 3-10 Image Generator

The generator is trained with three loss functions. The first loss function is the conditional adversarial loss with two discriminators. Appearance discriminator to make sure to have the right texture and pose discriminator to ensure that the generated image is aligned with the desired pose. The loss functions are defined as equation (2.12)

another loss function is also used called L1 (Manhattan Distance) distance to calculate the distance among the synthesized image I_g and the real image I_t, as shown in equation (2.10).

Since perceptual loss has been so successful in image generation it has been added as a third loss function for the Image generator. Perceptual loss is applied to calculate the distances between high-level features from a pretrained model among the

generated image I_g and the real image I_t. The formula of the perceptual loss is in equation (2.11)

Finally, the total loss function of the image generator is the weighted addition of all of the three loss functions and this loss function will be used to train the image generator. It can be formulated as equation (3.2).

$$L_{Image} = \lambda_1 L_{\ell 1} + \lambda_2 L_{percep} + \lambda_3 L_{CGAN} \ldots . (3.2)$$

where λ_1 is the weight of the L1 loss function, and λ_2 is the weight of the perceptual loss function, and lastly λ_3 represents the weight of the adversarial loss function. Figure 3.11 illustrate how the loss functions are calculated and Algorithm 3.3 shows the entire training process.

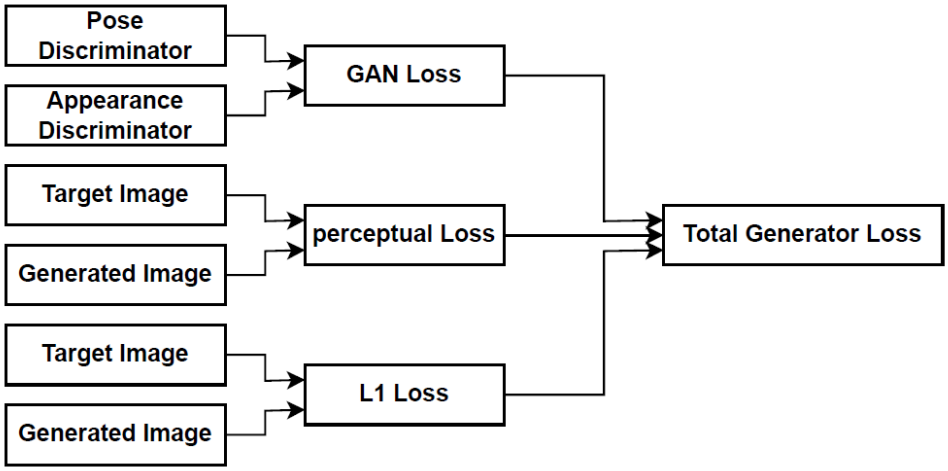


Figure 3-11 Generator Loss Function

Algorithm 3.3: Image Generator
Input: SourceImage, TargetImage.
Output: an image has the appearance of the source image and is aligned with the target pose.
Begin
Dataset Preparing
SourceKeyPoints = HumanPoseExtraction (SourceImage) TargetKeyPoints = HumanPoseExtraction (TargetImage) SourceParse = HumanParseExtraction (Source Image)

Data Preprocessing

```
SourceImage = ImagePreprocessing (SourceImage)
TargetImage = ImagePreprocessing (TargetImage)
SourcePose = KeyPointsPreprocessing (SourceKeyPoints)
TargetPose = KeyPointsPreprocessing (TargetKeyPoints)
SourceParse = HumanParsingPreprocessing (SourceParse)
```

Image Generator

```
ParseGenerator = GeneratorModel () // initializing the Parse Generator
ImageGenerator = GeneratorModel () // initializing the Image Generator
AppearanceDisc = ResNetDiscriminator () // initializing the Appearance Discriminator
PoseDisc = ResNetDiscriminator () // initializing the Pose Discriminator

// Extracting the Generated target parse from the parse generator
GeneratedTargetParse = ParseGenerator (SourceImage, SourceParse, SourcePose, TargetPose)

//Training The Discriminators
// DiscGenRatio is the ratio of training the discriminator to the generator in this case the
// discriminator will be trained twice as much as the generator (1:2)
For from 0 to DiscGenRatio
    GeneratedImage = ImageGenerator (SourceImage, SourceParse, TargetPose,
                                     GeneratedTargetParse)
    // Appearance Discriminator Training
    RealAppearance = AppearanceDisc (SourceImage, TargetImage) // Real pair
    FakeAppearance = AppearanceDisc (SourceImage, GeneratedImage) // Fake pair
    // calculate the loss function of the appearance discriminator using Wasserstein GAN loss
    AppearanceLoss = WassersteinGAN (RealAppearance, FakeAppearance)
    AppearanceDisc.UpdateWeights (AppearanceLoss) // update the weights

    // Pose Discriminator Training
    RealPose = PoseDisc (TargetImage, TargetPose) // Real pair
    FakePose = PoseDisc (GeneratedImage, TargetPose) // Fake pair
```

```
    // calculate the loss function of the pose discriminator using Wasserstein GAN loss
    PoseLoss = WassersteinGAN (RealPose, FakePose)
    AppearanceDisc.UpdateWeights (AppearanceLoss) // update the weights

// Generator Training
 FakeAppearance = AppearanceDisc (SourceImage, GeneratedImage)
 FakePose = PoseDisc (GeneratedImage, TargetPose)
 GANLoss = WassersteinGAN (FakeAppearance, FakePose)
 L1_Loss = L1 (TargetImage, GeneratedImage)
 Perceptual_Loss = Perceptual (TargetImage, GeneratedImage)
 // Calculate the total loss function for the image generator
 Total_Generator_Loss = λ1 * L1_Loss + λ2 * Perceptual_Loss + λ3 GAN_Loss
 // updating the weights of the image generator
 ImageGenerator.UpdateWeights (Total_Generator_Loss)
// This algorithm is for one image only, it will be repeated for each image in the dataset
End
```

3.3 Generative Adversarial Networks Models

There are two components of Generative Adversarial Networks the main component is the Generator and the second component is the Discriminator. This thesis will propose its own Generator and Discriminator architectures.

3.3.1 Generator Model

The Generator is a neural network that is responsible for generating images, video, sounds etc. An encoder-decoder architecture was used as shown in Figure 3.12 where the encoder down size the input image to (64, 44) shape, and after the ResBlocks it is reversed with the decoder to get back to the original shape. Various techniques were used in the generator because they were suitable for Human Pose Transfer task. Because this task is defined as an “unaligned” task the generator will heavily depend on attention mechanisms thus the generator has two levels of

attention, the first is pixel-wise attention which is represented by using gated convolution instead of vanilla convolution layer. The second attention level is channel-wise attention which is represented in squeeze and excitation blocks (SE Block). Also, skip connections used the same as Residual Networks, these skip connections were used in ResBlocks, three ResBlocks were used due to the limited computation power while the intended number of ResBlocks was five.

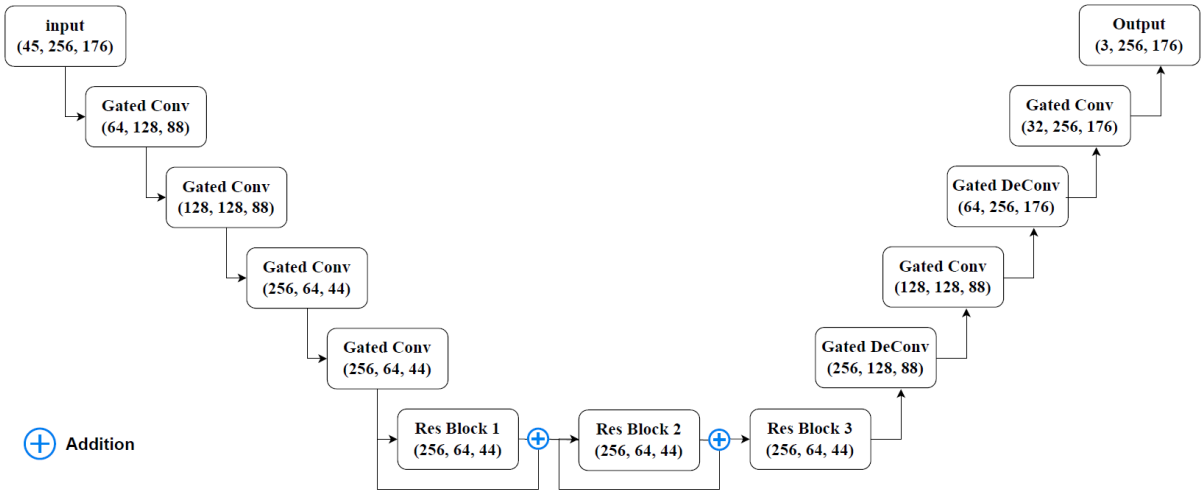


Figure 3-12 Generator Architecture

3.3.1.1 Gated Convolution

Gated Convolution is giving a weight for each pixel selecting only the important regions and these weights and learnable parameters that will learn to pick the desired areas to deform throughout the training stage.

Gated convolution may gather important information at each spatial position for the Parse Generator, which is excellent for maintaining semantic components of the person (e.g., clothing style). To produce a textured human image, gated convolution can maintain critical characteristics while deforming vital areas.

As a result, all normal convolutions are replaced with gated convolutions in order to acquire and distort features adaptively. Gated Convolution sends the input in two different streams the first one is a normal convolution layer with a ReLU activation

function while the second stream is also passing the input through a convolution layer but with a Sigmoid activation function to scale the output values between (0, 1) because they will be the attentions score for each corresponding pixel in the first stream. After that the output of both the streams is multiplied to get the final output from the GatedConv. Figure 2.8 shows the architecture of the GatedConv block.

3.3.1.2 Squeeze and Excitation

Squeeze and Excitation (SE), along with Gated Convolution, is the second level of attention in the generator model. Squeeze and Excitation enhance channel inter-dependencies while consuming essentially little computing resources. It provides a significant performance improvement. The basic concept is to add variables to every layer of a convolutional blocks in order that the network itself might automatically change how it weights each of its feature maps. As a result, only the most significant channels are considered.

The SE Block that is used in the generator model consist of an adaptive average pooling layer that squeezes the entire channel down to a single value and then passes it through a conv layer for further computations with the SiLU activation function continuing with another conv layer but this time with sigmoid activation functions to scale the output between (0, 1) and give the final attention score which will be multiplied with the input of the Squeeze and Excitation Block so each attentions score excite its corresponding channel as shown in Figure 3.13.

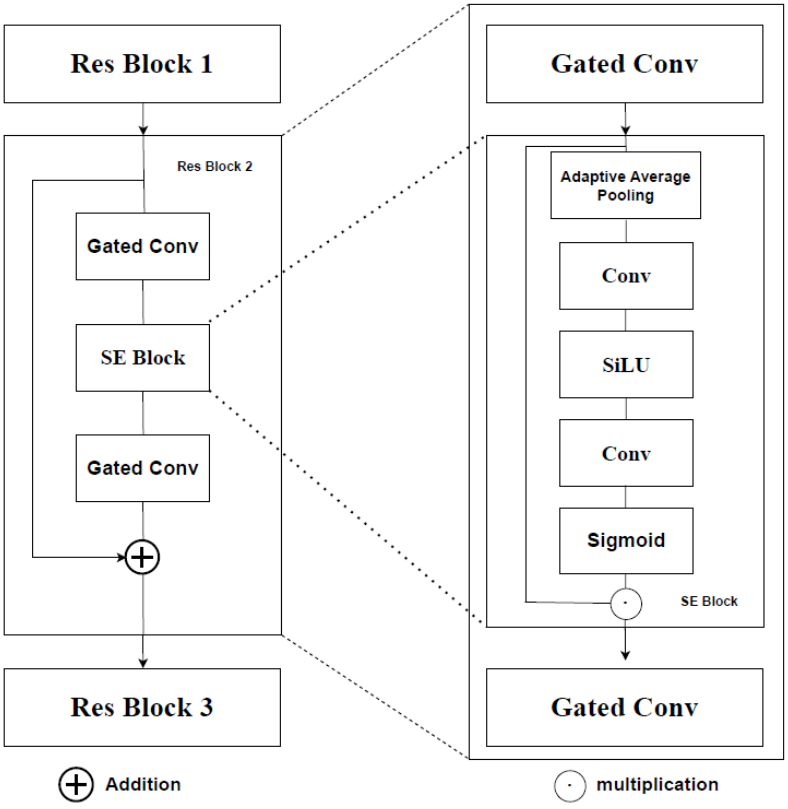


Figure 3-13 ResBlock and SE Block details

3.3.2 Discriminator Model

By replicating the ground truth distributions, GANs are utilized to create realistic looking images. In conventional generative adversarial networks, the discriminator is utilized to identify if the generated image is real or fake in order to motivate the generators to generate realistic images. In addition to providing realistic images, it is critical for conditional image generation activities to guarantee that the created images fulfil the condition image criteria (for example, in human pose transfer, the pose of the generated image needs to be matched with the target pose). As a result, two discriminators have been used: a pose discriminator and an appearance discriminator, to motivate the generators to produce images that are aligned with the target pose while keeping the appearance that was present in the source image. The CGAN loss function is shown in equation (2.12).

3.3.2.1 Appearance Discriminator

The appearance discriminator aims to keep the appearance of the generated image as close as to the source image, the inputs to the appearance discriminator are two pairs one of them is real represented by (source image, target image). The second pair are the fake pairs represented by (source image, generated image) as in Figure 3.14.

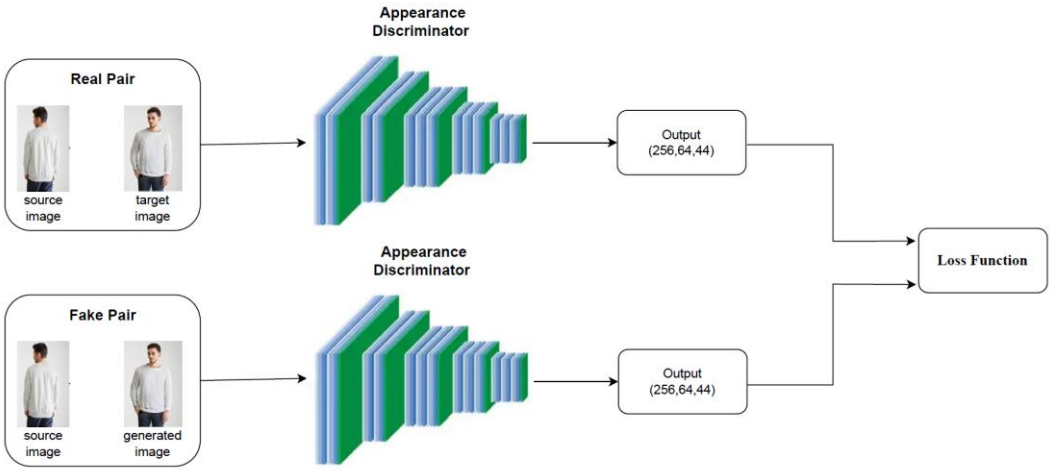


Figure 3-14 Appearance Discriminator

3.3.2.2 Pose Discriminator

The pose discriminator aims to keep the pose of the generated image aligned with the target pose, the inputs of the pose discriminator are two pairs real and fake. The real pair is represented by (target image, target pose), while the fake pair is represented by (generated image, target pose) as shown in Figure 3.15.

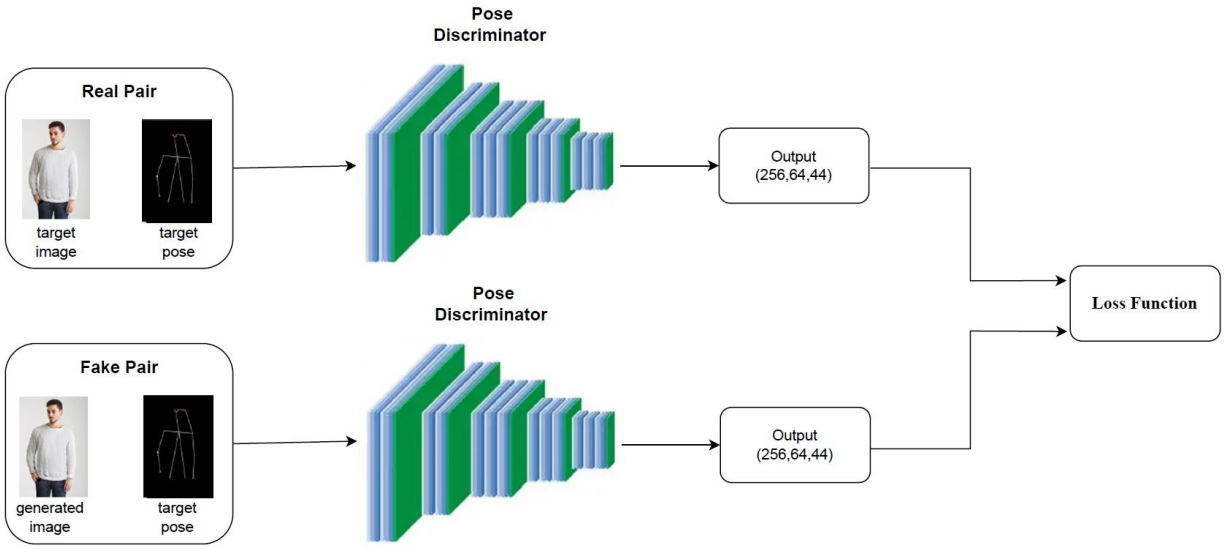


Figure 3-15 Pose Discriminator

3.3.2.3 Discriminators Architecture

The discriminator's architecture shown in Figure 3.16 down-samples the input images to (64,44) and then passes it through four ResBlocks which outputs a (256,64,44) output shape where the first number represents the number of channels while the second and third represent the resolution of each channel. The output goes directly to the loss function to get the GAN loss value in order to add it to the other loss functions (perceptual and L1) to get the final loss value.

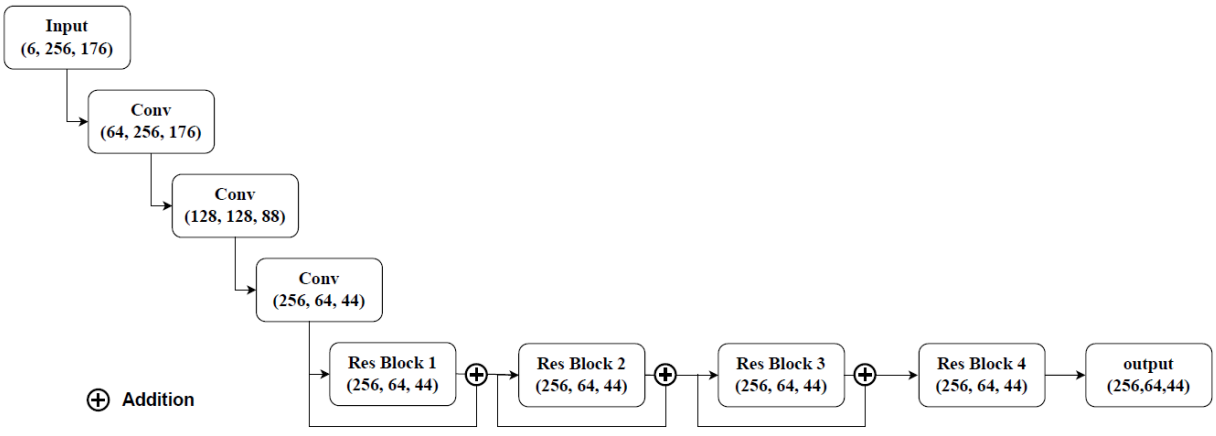


Figure 3-16 Discriminator architecture

3.3.3 GAN Loss Function

There are two models in GAN implementation: discriminator and generator. After being trained directly on actual and generated photos, the discriminator is responsible for classifying photographs as real or not. Rather than directly training the generator, the discriminator model is employed to teach it. The discriminator is particularly trained to generate the generator's loss function.

The Wasserstein loss is based on observing that typical GANs are driven to reduce the distance between the true and predicted probability distributions, which is commonly referred to as the Jensen-Shannon divergence or Kullback-Leibler.

Rather they propose modeling the issue using the Earth-Mover distance, often known as the Wasserstein-1 distance. The Earth-Mover's distance is the distance between two probability distributions measured in terms of the cost of transforming one distribution (pile of earth) into another.

The GAN with Wasserstein loss involves altering the discriminator into a critic that is updated more frequently than the generator model (e.g., five times as frequently). The score is derived in such a way that the difference between the scores for true and fake images is as little as possible.

Because of the benefits of the Wasserstein mentioned in Chapter 2 section 2.6.3 Wasserstein loss is used instead of the original GAN loss function.

Chapter Four
Experimental Results and Discussion

Chapter Four: Experimental Results and Discussion

4.1 Overview

This chapter includes an explanation of the experiments and results for each step of the proposed model. In addition to explaining the used dataset in the proposed system, the hardware, and software used to implement the proposed system. The results of the stages are sorted by their appearance in Chapter Three.

4.2 Hardware Requirements

Implementing image processing systems using machine learning with a deep learning approach requires high computer resources to apply flexibly, especially with huge datasets. Therefore, the proposed model was implemented using two main sources:

A. Local Hardware

- Central Processing Unit (CPU): Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz.
- RAM: 16 GB.
- Graphics Processing Unit (GPU): NVIDIA GeForce GTX 1060 6GB.
- Hard Disk: 256 GB SSD.
- Operating system: Windows10, 64 bit.
- Programing language: Python 3.9 Anaconda with PyCharm 2022 IDE.

B. Cloud GPU

- Google Colab Pro subscription for 1 month.
- Google Colab Pro+ subscription for 1 month.

4.3 The Experimented Dataset

DeepFashion database is a large-scale clothing database [63] with various enticing features:

• DeepFashion, the biggest visual fashion research database, has over 800,000 varied fashion images ranging from well-posed store shots to unrestricted consumer photos.

• Second, DeepFashion contains detailed information on clothing goods. This dataset labels each image with 50 categories, 1,000 descriptive features, a bounding box, and clothing landmarks.

• DeepFashion also includes approximately 300,000 cross-pose/cross-domain image pairings.

The DeepFashion database is used for developing four benchmarks: attribute prediction, consumer-to-shop clothes retrieval, in-store clothes retrieval, and landmark detection.

4.4 Dataset and Implementation Details

DeepFashion Inshop Clothes Retrieval Benchmark [63] is used for the experiment, which has 52712 photos with a resolution of (256x256). The attitudes and looks of the images vary.

First, the parsing generator is trained for 50 epochs, after obtaining a decent result from the parse generator, the parse generator stop training. The input of the parse

generator is the target pose, source pose, source image and source parsing image, the result is a generated parse image aligned with the target pose.

The discriminators and the image generator are alternatively trained for 50 epochs and the discriminator is trained twice as much as the generator. The inputs of the image generator are the source image, source parse image, generated parse image and the target pose, the result is the final generated image.

While the appearance discriminator takes the (source image, target image) as the real pair and (source image, generated image) as the fake pair, and the pose discriminator takes (target pose, target image) as the real pair and (target pose, generated image) as the fake pair.

The ratio of generator-discriminator training is (1:2) the discriminator is trained more than the generator to obtain a better discriminative power in order to get better results from the image generator. Adam optimizer is used with b1=0.5 and b2=0.999 and learning rate=1e-4. Also, the values of the $\lambda 1$, $\lambda 2$, and $\lambda 3$ in the generator loss are (1, 1, 5) respectively.

4.5 Features Extraction

This section will be about the results of the feature extraction stage in the data preparation phase.

4.5.1 Pose Extraction

The Human Pose Estimation system that has been used in this thesis to extract the pose is called “OpenPose” after downloading the project and the weights the images are fed to the model to get 18 (x, y) coordinates for each point representing a joint or a face landmark. After getting the coordinates it will be saved in a csv file as (ImageName, Keypoints_y, Keypoints_x).

If a full-person image is fed to OpenPose the output should be 18 (x, y) points as shown in Figure 4.1.

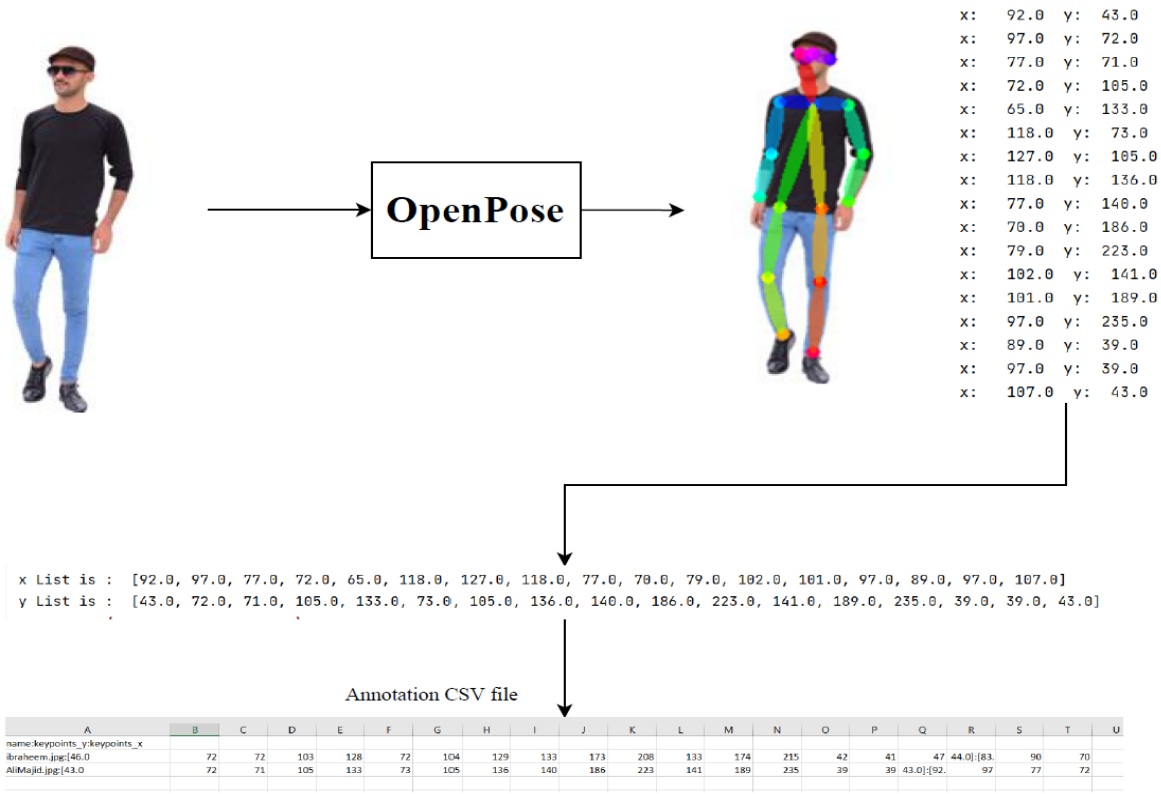


Figure 4-1 Pose Extraction

4.5.2 Human Parsing Extraction

The human parsing system that has been used in this thesis is Part Grouping Network (PGN). After downloading the project and the weights now human parsing maps will be extracted for each image in the dataset. When the images are fed to the PGN model the output should be an image where the person in the input image is segmented into 20 classes. After receiving the segmented image, the classes will be reorganized into 12 as shown in Figure 4.2.

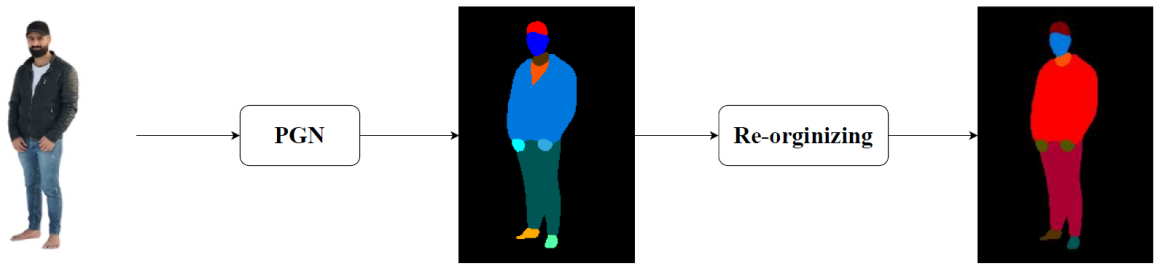


Figure 4-2 Human Parsing Extracting

4.6 GAN Model Experiments

In this section, the development of the MLA-GAN will be described as well as the stages of and all the tests that have been done.

All the tests will be on a sample of the dataset. This sample will be 200 pairs with 50 epochs for each test.

4.6.1 Generator Architecture

The Generator is the main part of GANs, since it's the most important part more tests will be done with the generator than the discriminator, after having a look at serval pre-trained models architectures like (LeNet, VGG, InceptionNet, ResNet, EfficientNet, Xception, SE-Net, U-Net) and serval types of GAN architectures like (vanilla GAN, C-GAN, DC-GAN, Pix2Pix-GAN, Cycle-GAN, Pro-GAN, SR-GAN, Style-GAN), basic knowledge is gained about how to build a deep neural network for GAN from scratch.

4.6.1.1 Generator Architecture Version 1 (V1)

An encoder-decoder architecture network with convolution and residual blocks was the baseline of the network as shown in Figure 4.3. The convolution block consists of a convolution layer, instance normalization and ReLU respectively.

The convolution layer parameters are (kernel-size = 3x3, stride = 2, padding = 1, padding-mode = "reflection"):

a- kernel size of 3 was used because of the following:
   - first odd numbers are very preferable over even numbers when it comes to picking a kernel size due to the odd kernel size having a center position, unlike the even kernel size.
   - Second, the smaller the kernel size the smaller the computations needed due to having fewer parameters to update while back-propagation.

b- The stride used was 2 to downsize the feature maps to half.

c- While the padding was used to make sure the downsizing is perfectly the half of the input feature map else it would be minus one dimension because the kernel size used is a (3x3), while the padding mode was used is "replicate" because the borders of the images are unimportant in our case so it just replicates the nearest pixel to it which is 99% are just white background pixels in the dataset.

About the normalization layer, instance normalization was chosen which normalizes each channel of each sample individually because every single channel should be independent of the other channels due to the nature of human pose transfer.

ReLU was utilized as the activation function, which stands for "Rectified Linear Unit". Main advantage of using this activation versus the others is that it won't simultaneously triggers all of the neurons. This indicates that neurons are only going to be silenced if the result is lower than zero. Negative input values result in zero, indicating that the neuron is not activated. The ReLU function is significantly faster in terms of computation than the other function since it just engages certain groups of neurons.

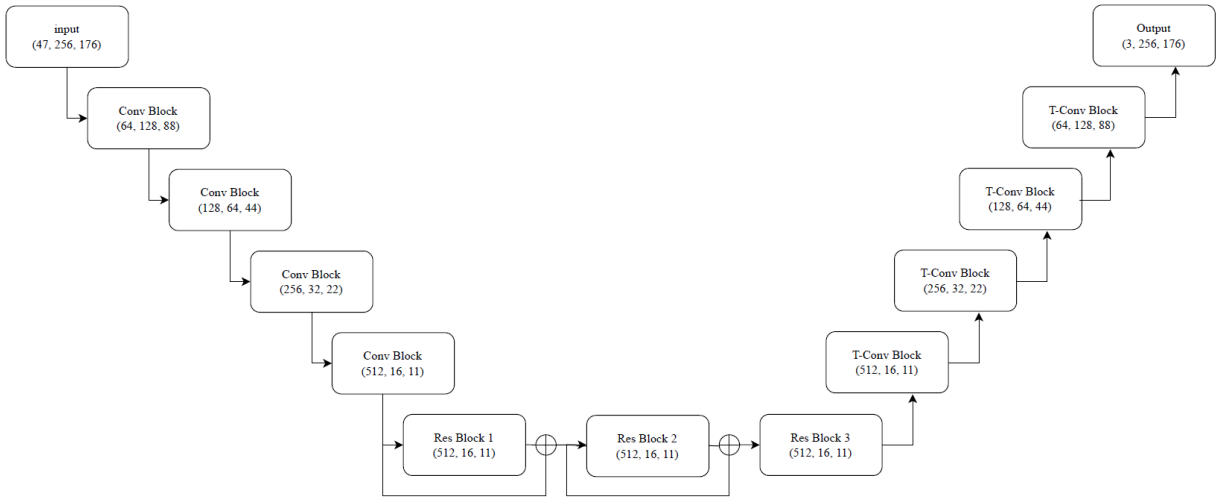


Figure 4-3 Illustration of the early steps in developing the Generator

The Res Block consist of three Conv Blocks with a "ResNet "skip connection as shown in Figure 4.4.

Skip connections take the input features of the res block and add them with the output of the third conv block features at the end of the res block right before the output. This procedure takes no additional parameters since the result from the layer before it is added to the layer ahead. The skip connection of ResNet allowed us to build a deeper network to solve even more complex tasks.

4.6.1.2 Generator Architecture Version 2 (V2)

In this version of the generator, skip connections have been added between the encoder and the decoder to transfer the feature maps of the encoder directly to the decoder just like U-net skip connections as shown in Figure 4.5. The layers in the encoder part are concatenated with layers in the decoder part. Long skip connections are used to transfer features from the encoder side to the decoder side, permitting the recovery of spatial information left while down-sampling. These skip connections are different from the ones used in the Res Block, the skip connections between the encoder and the decoder are called long skip connections to reuse the features of the

encoder in the decoder by concatenating them, while the skip connections in the Res Block are short connections and it adds the input to the output of the res block.

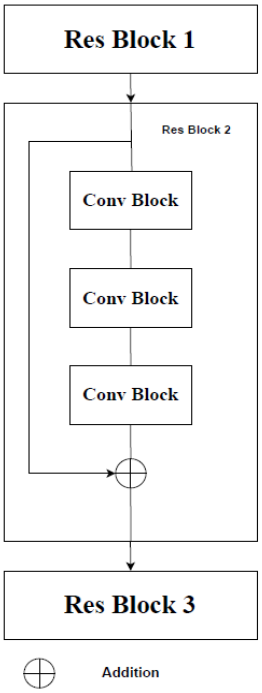


Figure 4-4 Res Block details

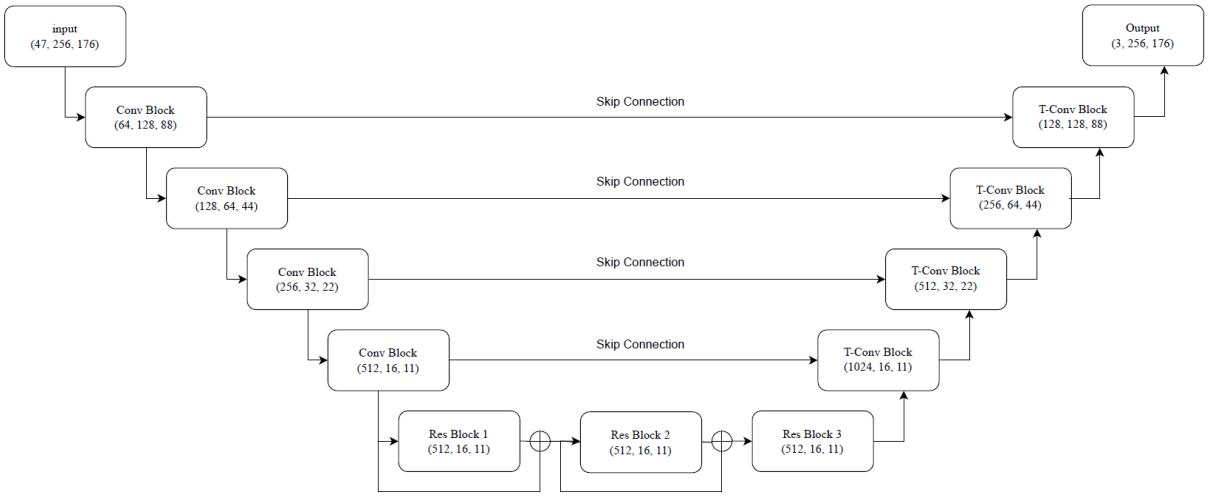


Figure 4-5 The Generator Architecture after adding the skip connections

4.6.1.3 Generator Architecture Version 3 (V3)

In generator V3 vanilla convolution was replaced with gated convolution as shown in Figure 4.6 as well as adding squeeze and excitation blocks.

a- Gated Conv: normal convolution layers show that the result values are generated using the same filter in all spatial locations. It accepts every pixel as acceptable values and uses a sliding window for obtaining local features. which is useful for applications like as object identification, image segmentation, and aligned tasks. However, for misaligned tasks, such as human pose transfer, vanilla convolution features do not necessarily have a good influence on the result. As a result, learning a dynamic feature selection process to modify the image is increasingly critical. Thus using gated convolution is more suitable for this task by adding attention to each pixel.

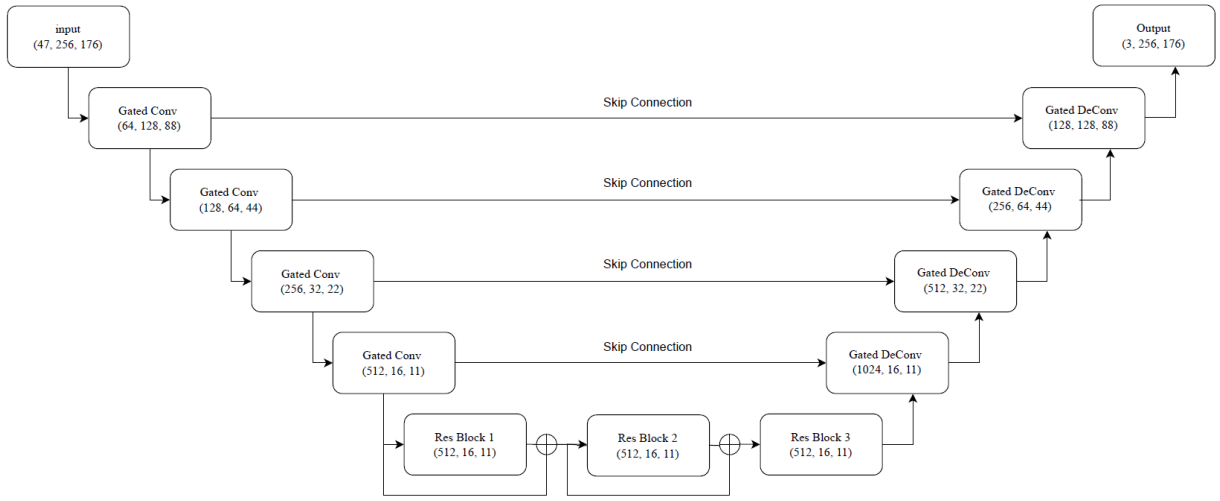


Figure 4-6 The Generator Architecture After adding the Gated Conv

b- Squeeze-and-Excitation Networks (SENets): SENets presented a CNN building block that increases channel interdependencies at nearly no computational cost. They provide significant performance gains and are simply integrated into current designs. The core concept is to add variables to every channel within a convolutional block in order that the algorithm may modify the importance of every feature map adaptively. So the second gated conv block is replaced with a squeeze and excitation block instead as shown in Figure 3.15.

SE blocks have been used because in our task not all channels are equally important thus an attention mechanism was needed over the feature maps to pick the most useful features that directly help to transfer either the human parsing in the Parse Generator or transferring the appearance in the Image Generator.

SE blocks use a third type of skip connection, unlike the ResNet skip connections which are added to the output, and the U-net skip connection are concatenated, SE skip connections are the multiplication of the input with the output because the output of SE block will be only the attention scores.

4.6.1.4 Generator Architecture Version 4 (V4)

Long skip connections have been removed between the encoder and the decoder due to the nature of this task which is unaligned which means the input image is not aligned with the generated image, unlike semantic segmentation which is an aligned task (U-net major usage is in semantic segmentation tasks). The tests after removing the long skip connection yielded better generated images. But this issue can be fixed by adding a transformation block before concatenating the encoder features with the decoder features. This transformation block should be trained to make the encoder features aligned with the target image. In this case, the long skip connection would be useful in this task, but this will be a future work due to the limited resources during the development of this generator architecture. Figure 4.7 shows the architecture after removing the skip connections.

4.6.1.5 Generator Architecture Version 5 (V5)

The number of down samples has been reduced from 4 down samples which made the feature map size to be (16, 11) to only 2 making the feature map size to be (64, 4), thus getting the feature maps bigger in the Res Blocks. The new feature maps did affect the generated images for the better. We are not exactly sure why it yielded better results but it may be due to bigger feature maps that can learn more complex features and with the channel-wise attention this paired very well.

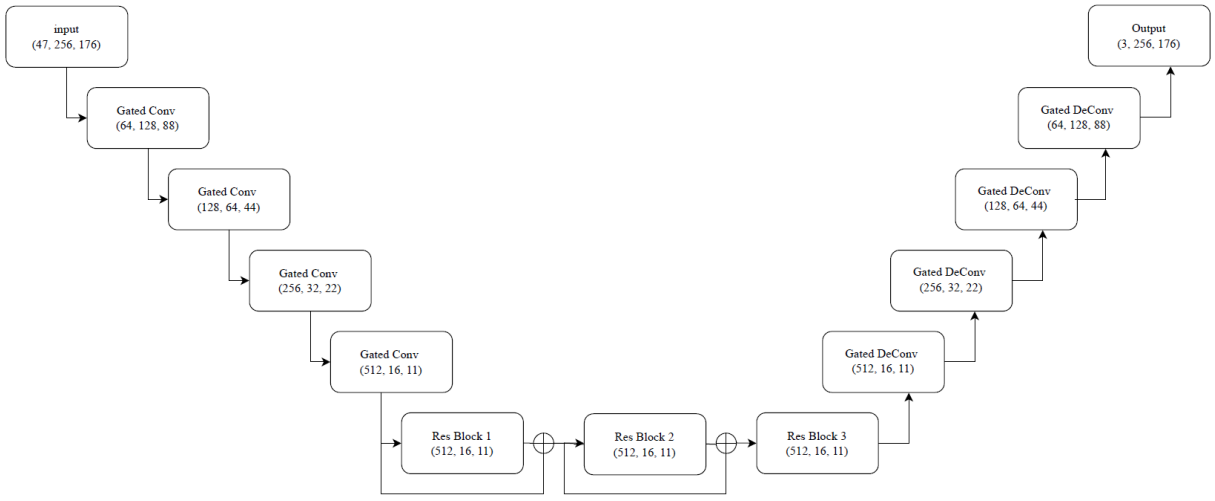


Figure 4-7 Generator Architecture after removing the skip connections

But getting the feature maps bigger brought another problem during the training which is the computation. Thus the number of channels has been limited to (256) instead of (512). This did affect the generated image but not significantly while it did reduce the training time quite well. The final form of the generator is shown in Figure 3.13.

With this architecture, we are satisfied enough with the result but we would like to keep developing and testing more techniques and tweaking the architecture unfortunately due to the limited time the development stopped with the current architecture.

4.6.1.6 Generators Comparison Results

Comparing the different versions of the generators according to four metrics (IS, SSIM, LPIPS, FID) in Table 4.1. The table shows the evolution of the generator during the development process and its noticeable that adding the attention to the generator enhances the result.

Table 4.1 Comparison of different versions of the generator

Generator Version	IS↑	SSIM↑	LPIPS↓	FID↓	Number of Parameters↓
V1	2.3877	0.3561	0.3731	37.7260	15.6M
V2	2.5082	0.6278	0.3684	24.64329	30.8M
V3a	2.6344	0.7064	0.2492	15.5304	39.6M
V3b	2.7391	0.7638	0.2147	12.7649	35.4M
V4	2.7709	0.8900	0.1471	9.6994	20.8M
V5	2.6641	0.8497	0.1751	10.3568	7.5M
Real Data	2.7894	1.0000	0.0000	0.0000	-

The aforementioned table presentation has undergone experimentation involving a dataset comprising a mere two hundred pairs. Each testing iteration, spanning fifty epochs, was executed. In the initial generator iteration denoted as V1, a convolutional neural network (CNN) employing an encoder-decoder architecture was exclusively utilized. Subsequently, in the second generator iteration (V2), an augmentation was introduced in the form of skip connections extending from the encoder to the decoder. This augmentation is discernible in the substantial increase in the count of trainable parameters. Consequently, the augmented parameter load contributes to an extended training duration.

The third version (V3) introduces an attention mechanism into the generator, applied across two distinct levels. In the first tier (V3a), conventional convolutional layers were replaced with gated convolutional layers. While this substitution led to improvements in results, it simultaneously engendered a notable expansion in the parameter space, thereby potentially complicating the training process. Upon the

introduction of the second tier of attention (V3b), characterized by the incorporation of squeeze and excitation blocks within the residual blocks, a twofold effect was observed: a reduction in the parameter count and an amelioration in results.

Generator iteration four (V4) departs from the approach undertaken in V2 by omitting the integration of skip connections. This deliberate omission yielded a substantial reduction in parameter count, nearly fifty percent, consequently expediting the training duration and concurrently enhancing output quality. The developmental constraints imposed by limited temporal and resource provisions necessitated a further reduction in training time was applied in generator version five (V5), even if accomplished at the expense of the resultant image quality.

4.6.2 Discriminator Architecture

The discriminator is the second model of GAN and it is only used during training. The discriminator is nothing more than a classifier. It tries to distinguish between real and fake data. It can employ any kind of network layout appropriate for the kind of data getting classified.

The discriminator's training data is taken from two sources. The first is real data examples, such as actual images of people. The discriminator views these images as positive examples during training. The second is the generator's fake data images. The discriminator uses these images as negative examples during training.

4.6.2.1 DC-GAN Discriminator

DC-GAN paper proposed a simple discriminator with a single-digit output. With the output being as close to (1) is an indication that the input image is a real image (ground truth) while if the output is closer to (0) this represents that the input image is fake (generated). This discriminator can be shown in Figure 4.8. But the issue with this discriminator it’s too simple for this task, the discrimination power of this discriminator is not powerful enough.

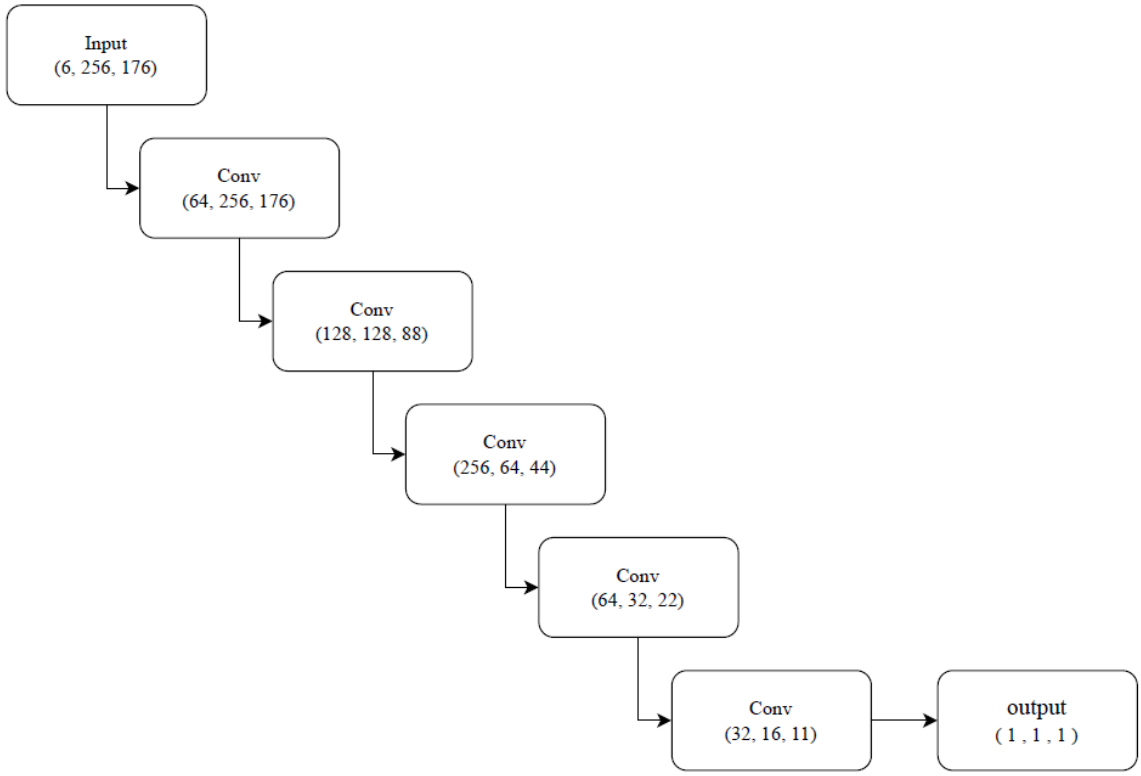


Figure 4-8 Single Digit Discriminator

4.6.2.2 Patch Discriminator

After the single digit discriminator, another type of discriminator has been tried called patch discriminator. Patch discriminator output is a single channel but it has height and width (matrix) the idea behind it is that each pixel tells if the corresponding area to it in the input image is fake or real. For example, if the output of the patch discriminator is (2x2) that the original image is divided into 4 quarters and each pixel of the output is corresponding to one quarter. This discriminator can be shown in Figure 4.9. After using this discriminator, it improved the quality of the images due to the fact that a better discriminator can in turn leads to a better generator. Because the better the discriminator can get in distinguishing between the real and fake images the better the chance of training the generator to optimality.

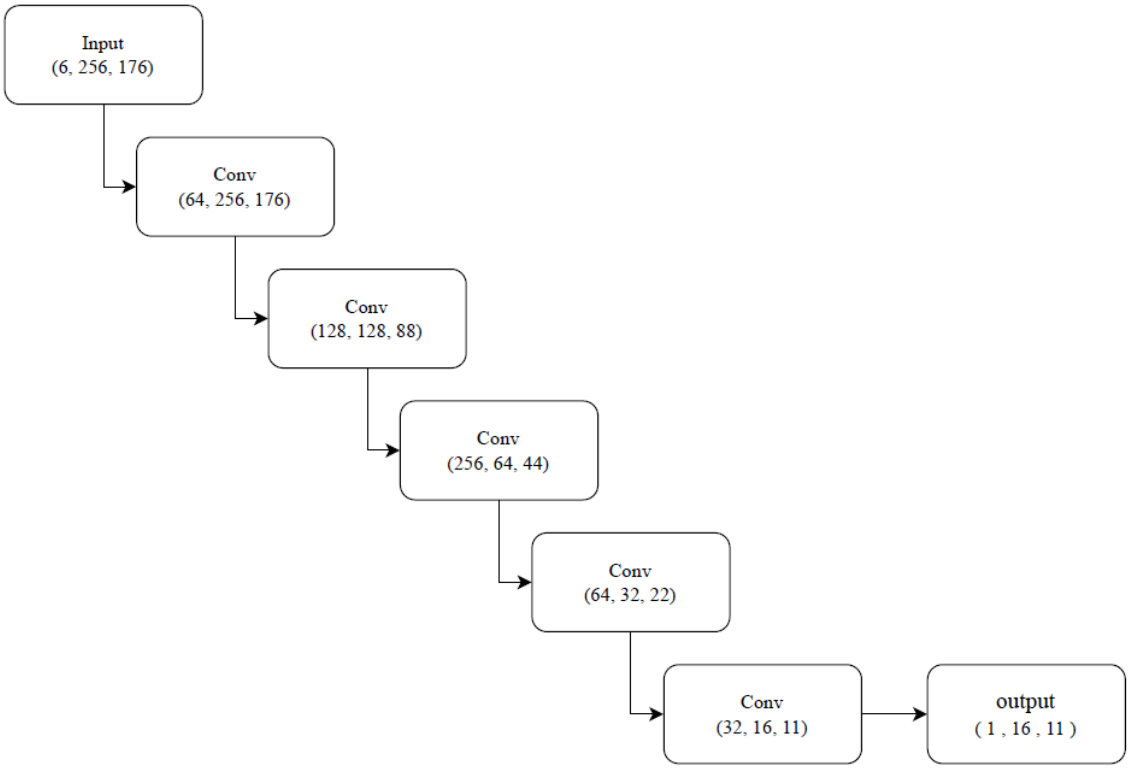


Figure 4-9 Patch Discriminator

4.6.2.3 ResNet Discriminator

Another discriminator was tested and that is the ResNet Discriminator. It is called the ResNet discriminator due to it having few residual blocks in its architecture. The idea behind using this discriminator is to use the internal features instead of a few parameters thus resulting in a better discriminating power. The ResNet discriminator can be shown in Figure 3.18.

4.7 Visual Results

In this section, visual results will be shown for the proposed system including both the parse and the image generators along with the inputs. The results are shown in Figure 4.10. We can note the importance and the effect the parse generator has on the image generator; the image generator just fills the texture depending on the parse generator output. Also from the below figure we can notice some mistakes with the parse generator if the target image is wearing shorts but this isn't clear in the input

image the parse generator assumed the input image was wearing pants instead due to lack of information in the input image so the parse generator just predicted pants because it is the dominant class over the shorts in the dataset.

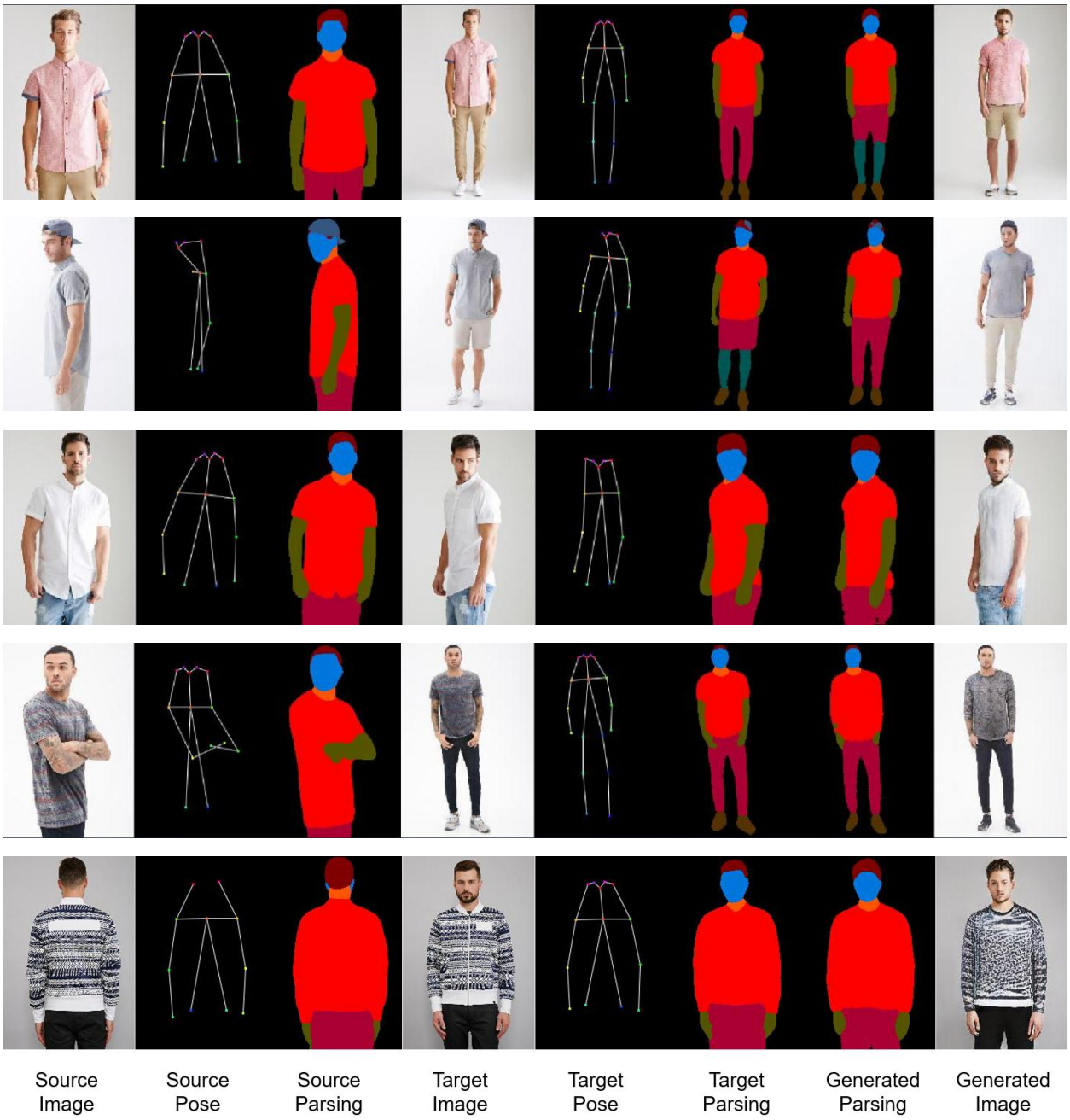


Figure 4-10 Visual Results

4.7.1 Parse Generator

The results of the parse generator are shown in Figures 4.11 and 4.12. In the Figures, the input image is on the left and the condition pose and the output of it is on the right.

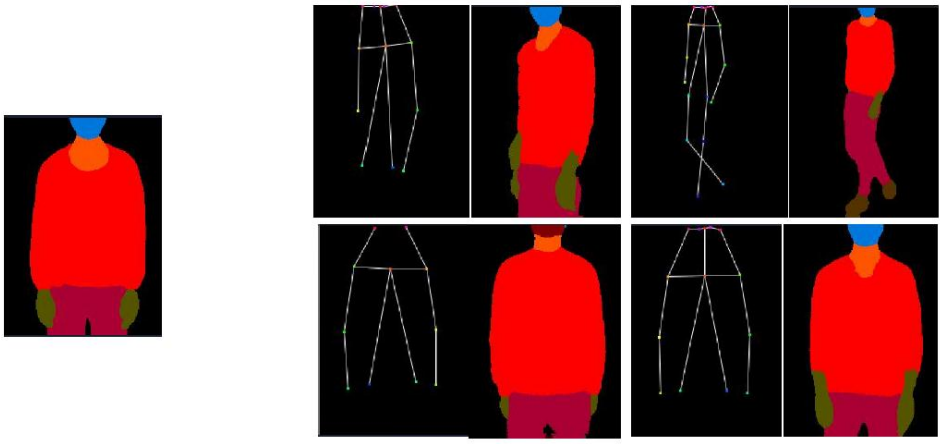


Figure 4-11 Parsing Generator results

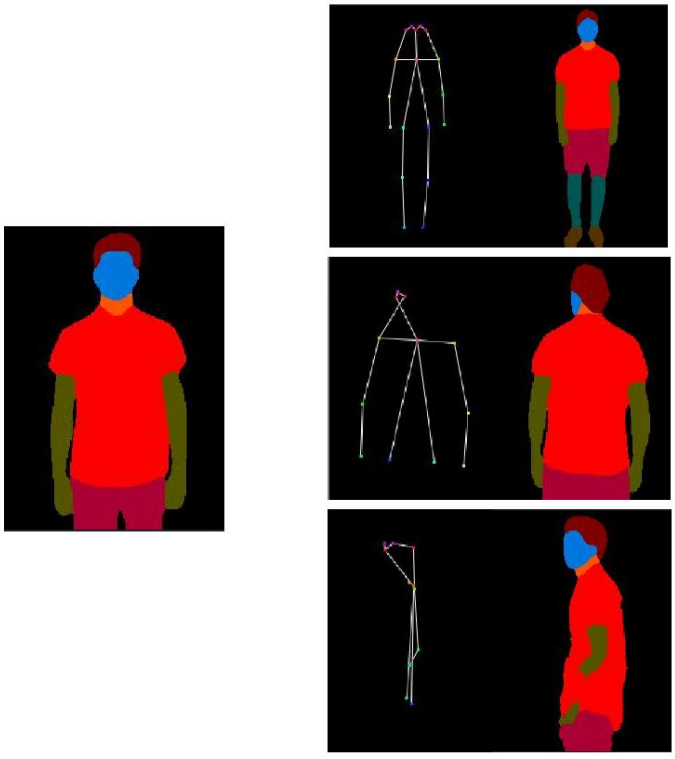


Figure 4-12 Parsing Generator results

4.7.2 Image Generator

The results of the image generator are shown in Figures 4.13 and 4.14. In the figures, the input image is on the left and the condition pose and the output of it is on the right.

Figure 4-13 Image Generator results

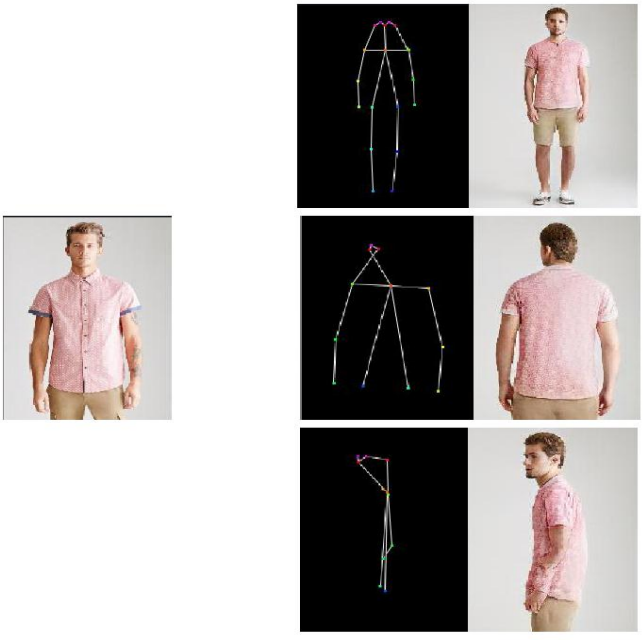


Figure 4-14 Image Generator results

4.8 Quantitative Comparison

Four metrics were employed as evaluation metrics to assess the efficacy of human pose transfer. The qualitative results are applied on DeepFashion dataset. Table 4.2 compares the findings of 8750 test images to the state-of-the-art approach.

The Inception Score (IS) and Structural Similarity (SSIM) are two popular assessment measures for quantifying perceptual performance and image quality. SSIM directly compares the reconstructions to the original images. Because the technique varies from both by creating images based on pose and appearance, it highlights the value of conditional representation for image production. Unlike SSIM, the inception score is calculated on the collection of reconstructed images apart from the original images.

Learned Perceptual Image Patch Similarity (LPIPS) is utilized to determine the error of reconstruction among the created image and the real in order to match human judgment. The Frechet Inception Distance (FID) is a metric used to assess the realism of produced images. LPIPS computes the perceptual distance between produced images and references images. It denotes the perceived distinction between the inputs. Meanwhile, FID computes the Wasserstein-2 distance between the produced image distributions and the ground-truth image distributions.

the results obtain the best FID performance, which is more compatible with human judgment, indicating that the model not only creates realistic images but also maintains shape and texture consistency.

In terms of IS, the model gained the 3rd place in the table with only a 0.001 difference from the 2nd place. The same with SSIM the model took the 3rd spot with a 0.003 difference behind the 2nd spot. While in LPIPS the model again took the 3rd place. But this research took 1st place in the FID score. Besides, this model has the fewest parameters.

We need to mention that this score isn't the full potential of the generator model because it wasn't fully trained because it requires very high computation power. Unlike the models we compared which had access to high-end GPUs and were able to train their models for (200-800) epochs while we only been able to train the model for 50 epochs.

Table 4.2: Results comparison with state of art papers. **Bold** indicates the highest score and underline indicates the second highest score.

Method	IS↑	SSIM↑	LPIPS↓	FID↓	Number of Parameters↓
Deformable GANs for Pose-based Human Image Generation	3.439	0.756	0.2330	18.457	82.08M
A Variational U-Net for Conditional Appearance and Shape Generation	3.087	**0.786**	0.2637	23.667	139.36M
Progressive Pose Attention Transfer for Person Image Generation	3.209	0.773	0.2533	20.739	41.36M
Dense Intrinsic Appearance Flow for Human Pose Transfer	3.338	0.778	**0.2131**	16.314	49.58M
Pose Guided Person Image Generation	3.090	0.762	0.2901	47.917	437.09M
Guided Image-to-Image Translation with Bi-Directional Feature Transformation	3.220	0.767	-	12.266	-
Human Pose Transfer by Adaptive Hierarchical Deformation	3.419	-	0.2159	12.635	20.41M
Controllable Person Image Synthesis with Attribute-Decomposed GAN	3.364	0.772	0.2383	18.395	-
MUST-GAN: Multi-level Statistics Transfer for Self-driven Person Image Generation	**3.692**	0.742	0.2412	15.902	-
Bipartite Graph Reasoning GANs for Person Image Generation	3.430	0.778	0.2428	24.360	-
XingGAN for Person Image Generation	3.476	0.778	0.2914	41.790	-
Region-Adaptive Texture Enhancement For Detailed Person Image Synthesis	3.125	0.774	0.2180	14.611	-
Towards Fine-grained Human Pose Transfer with Detail Replenishing Network	3.125	0.774	0.2180	14.611	-
Our Model	3.475	0.775	0.2173	**10.938**	**7.5M**

Chapter Five
Conclusion and Future Works

Chapter Five: Conclusion and Future Work

5.1 Conclusions

Several conclusions throughout the design and implementation of the proposed system have been drawn. Following the results of this thesis:

1. KeyPoints and Human Parsing are the chosen pose representation for our proposed model because the key-points don't provide enough information about the pose so we used human parsing which is semantic segmentation for humans along with the keypoints to boost the pose representation, furthermore the human parsing gives us control over each individual part of the human in the input image.
2. Attention mechanism is to take a huge role in our generator architecture due to its efficiency in selecting the important features. We have used two levels of attention: pixel-wise represented by the gated convolution and channels-wise represented by the squeeze and excitation blocks:
    a. In the generator architecture vanilla convolution was replaced with gated convolution because gated convolution is more suitable for the unaligned task like human pose transfer. Gated convolution is giving an attention score for each pixel in each feature map thus taking the important parts of each feature only.
    b. squeeze and excitation technique is added to the the generator architecture to select the most important feature channels by giving an attention score for each channel in the residual blocks.
3. GP-WGAN was used instead of the traditional GAN loss function because it

allows us to train the model to optimality and Wasserstein Distance as an objective function is more stable than using JS divergence also we have trained the discriminator twice as much as the generator as recommended when using this loss function. The recommended ratio is (5:1) discriminator to the generator but due to our limited resources, we have used a (2:1) ratio.

4. Human pose transfer is a complex task so one loss function is not enough to cover all the aspects. So we have used three loss functions in the image generator, the first loss is the L1 loss function as a reconstruction loss and the Perceptual loss function to keep the texture of the image along with the GAN loss function.
5. In terms of the GAN loss function we have used two discriminators instead of one due to the complexity of the task we have assigned each discriminator a task the first one makes sure the output is aligned with the target pose thus it is called pose discriminator and the other one is used to ensure that the output has the same appearance as the input image thus it called the appearance discriminator.
6. Three different types of discriminators were tested and we used the Res discriminator over the single digit and the patch discriminator because the Res discriminator compares the features not only outputting if this image was real or fake.
7. Although in GANs the best metric is human judgment still we have to use image metrics to evaluate the proposed system like IS, SSIM, LPIPS and FID. We have achieved the best in FID and the 3rd in the other metrics. It's important to note that LPIPS and FID are closer to human judgment than the IS and SSIM thus they are more important.
8. Unseen parts are still difficult to predict. The dominant class (in human parsing) is usually what is predicted in the missing parts. Although the face is

impossible to predict no matter what.

5.2 Future Works

1. Try to get the generator model bigger by simply adding more channels and more Residual blocks like doubling the channels and adding 2 more residual blocks so in total it would be 5. Because bigger models usually do better.
2. Long skip connection were removed the from the generator because it was unsuitable for the unaligned tasks but we suggested that adding a few layers to transform the encoder features before the concatenation might solve the problem. We haven't applied it due to the limited computation power we had.
3. Transferring each part of the human individually with each part having its own simple generator and then merging them back into one image but due to the limited time it has been added to the future work.
4. key-points and the human parsing were used to represent the pose and there few more representations could be tried like DensePose and 3D meshes or might be there are other representations

References

[1] D. Forsyth and J. Ponce, Computer vision: A modern approach. Prentice hall, 2011.

[2] R. Szeliski, Computer vision: algorithms and applications. Springer Science & Business Media, 2010

[3] M. K. Bhuyan, Computer vision and image processing: Fundamentals and applications. CRC Press, 2019.

[4] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," Adv. Neural Inf. Process. Syst., vol. 2017-Decem, pp. 406–416, 2017.

[5] Karmakar and D. Mishra, "A robust pose transformational GAN for pose guided person image synthesis," Communications in Computer and Information Science, pp. 89–99, 2020.

[6] P. Esser and E. Sutter, "A variational u-net for conditional appearance and shape generation," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018.

[7] I. Goodfellow et al., "Generative Adversarial Networks," Cambridge University Press eBooks, pp. 153–173, Jun. 2014, doi: 10.1017/9781108891530.013.

[8] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," arXiv (Cornell University), Dec. 2013,

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," Lecture Notes in Computer Science, pp. 234–241, Oct. 2015, doi: 10.1007/978-3-319-24574-4_28.

[10] S. Lathuiliere, E. Sangineto, A. Siarohin, and N. Sebe, "Attention-based fusion for multi-source human image generation," 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.

[11] Y. Li, C. Huang, and C. C. Loy, "Dense intrinsic appearance flow for human pose transfer," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[12] A. Siarohin, S. Lathuiliere, E. Sangineto, and N. Sebe, “Appearance and pose-conditioned human image generation using deformable GANs,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 4, pp. 1156–1171, 2021.

[13] A. Khatun, S. Denman, S. Sridharan, and C. Fookes, “Pose-driven Attention-guided Image Generation for Person Re-Identification,” Apr. 2021,

[14] J. Jiang, G. Li, S. Wu, H. Zhang, and Y. Nie, “BPA-Gan: Human Motion transfer using body-part-aware generative adversarial networks,” Graphical Models, vol. 115, p. 101107, 2021.

[15] J. Zhang, K. Li, Y.-K. Lai, and J. Yang, “Pise: Person image synthesis and editing with decoupled gan,” 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[16] Y. Men, Y. Mao, Y. Jiang, W.-Y. Ma, and Z. Lian, “Controllable person image synthesis with attribute-decomposed gan,” 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[17] J. Zhang, X. Liu, and K. Li, “Human pose transfer by adaptive hierarchical deformation,” Computer Graphics Forum, vol. 39, no. 7, pp. 325–337, 2020.

[18] S. Song, W. Zhang, J. Liu, and T. Mei, “Unsupervised person image generation with semantic parsing transformation,” 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[19] B. Zhao, X. Wu, Z.-Q. Cheng, H. Liu, Z. Jie, and J. Feng, “Multi-view image generation from a single-view,” Proceedings of the 26th ACM international conference on Multimedia, 2018.

[20] H. Dong, X. Liang, K. Gong, H. Lai, J. Zhu, and J. Yin, “Soft-gated warping-GAN for pose-guided person image synthesis,” Adv. Neural Inf. Process. Syst., vol. 2018-December, no. NeurIPS, pp. 474–484, 2018.

[21] K. Obaid, S. Zeebaree, and O. Ahmed, “Deep Learning Models Based on Image Classification: A Review,” Int. J. Soc. Sci. Bus., vol. 4, pp. 75–81, 2020, doi: 10.5281/zenodo.4108433.

[22] N. Barla, “A Comprehensive Guide to Human Pose Estimation,” V7, Feb. 06, 2023.

[23] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 172–186, 2021.

[24] L. Yang, W. Jia, S. Li, and Q. Song, "Deep Learning Technique for Human Parsing: A Survey and Outlook," pp. 1–19, 2023,

[25] X. Zhang, X. Zhu, M. Tang, and Z. Lei, "Deep Learning for Human Parsing: A Survey," pp. 1–16, 2023, [26] K. Gong, L. Lin, Z. Hu, Y. Chen, and M. Yang, "Instance-Level Human Parsing via Part Grouping Network," Springer eBooks, pp. 805–822, Sep. 2018, doi: 10.1007/978-3-030-01225-0_47.

[27] S. Gollapudi, "Deep Learning for Computer Vision," Learn Comput. Vis. Using OpenCV, pp. 51–69, 2019, doi: 10.1007/978-1-4842-4261-2_3.

[28] Dr. Juliansyah Noor, Encyclopedia of Computational Biology Bioinformatics and, vol. 53, no. 9. 2019.

[29] A. Yaguchi, T. Suzuki, S. Nitta, Y. Sakata, and A. Tanizawa, "S Calable D Eep N Eural N Etworks," pp. 1–14, 2019.

[30] S. Theodoridis, "Neural Networks and Deep Learning," in Machine Learning, 2015.

[31] R. W. Pettit, R. Fullem, C. Cheng, and C. I. Amos, "Artificial intelligence, machine learning, and deep learning for clinical outcome prediction," Emerg. Top. Life Sci., vol. 5, no. 6, pp. 729–745, 2021, doi: 10.1042/ETLS20210246.

[32] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep Models under the GAN: Information leakage from collaborative deep learning," 2017, doi: 10.1145/3133956.3134012.

[33] S. Sharma, S. Sharma, and A. Anidhya, "Understanding Activation Functions in Neural Networks," Int. J. Eng. Appl. Sci. Technol., vol. 4, no. 12, pp. 310–316, 2020.

[34] R. H. R. Hahnioser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex- inspired silicon circuit," Nature, vol. 405, no. 6789, pp. 947–951, 2000, doi: 10.1038/35016072.

[35] A. F. M. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” arXiv, no. 1, pp. 2–8, 2018.

[36] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical Evaluation of Rectified Activations in Convolutional Network.,” arXiv (Cornell University), May 2015.

[37] J. Feng and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks,” J. Phys. Conf. Ser., vol. 1237, no. 2, 2019, doi: 10.1088/1742-6596/1237/2/022030.

[38] S. I. Granshaw, Neural networks and neurodiversity, vol. 36, no. 175. 2021.

[39] S. Theodoridis, “Neural Networks and Deep Learning,” in Machine Learning, 2015.

[40] S. P. Siregar and A. Wanto, “Analysis of Artificial Neural Network Accuracy Using Backpropagation Algorithm In Predicting Process (Forecasting),” IJISTECH (International J. Inf. Syst. Technol., vol. 1, no. 1, p. 34, 2017, doi: 10.30645/ijistech.v1i1.4.

[41] K. Janocha and W. M. Czarnecki, “On loss functions for deep neural networks in classification,” Schedae Informaticae, vol. 25, pp. 49–59, 2016, doi: 10.4467/20838476SI.16.004.6185.

[42] J. Qi, J. Du, S. M. Siniscalchi, X. Ma, and C.-H. Lee, “On Mean Absolute Error for Deep Neural Network Based Vector-to-Vector Regression,” IEEE Signal Processing Letters, vol. 27, pp. 1485–1489, Aug. 2020, doi: 10.1109/lsp.2020.3016837.

[43] J. C. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual Losses for Real-Time Style Transfer and Super-Resolution,” Springer eBooks, pp. 694–711, Oct. 2016, doi: 10.1007/978-3-319-46475-6_43.

[44] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of wasserstein GANs,” Neural Information Processing Systems, vol. 30, pp. 5769–5779, Dec. 2017,

[45] S. Ruder, “An overview of gradient descent optimization algorithms,” pp. 1–14, 2016,

[46] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–15, 2015.

[47] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “NNs Archtectures review,” Elsevier, pp. 1–31, 2017.

[48] Yamashita, R. et al. (2018) ‘Convolutional Neural Networks: An overview and application in Radiology’, Insights into Imaging, 9(4), pp. 611–629. doi:10.1007/s13244-018-0639-9.

[49] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, “Free-Form Image Inpainting With Gated Convolution,” International Conference on Computer Vision, Oct. 2019, doi: 10.1109/iccv.2019.00457.

[50] B. Mele and G. Altarelli, “Lepton spectra as a measure of b quark polarization at LEP,” Phys. Lett. B, vol. 299, no. 3–4, pp. 345–350, 1993, doi: 10.1016/0370-2693(93)90272-J.

[51] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, “How does batch normalization help optimization?,” 2018.

[52] S. Yang, W. Xiao, M.-C. Zhang, S. Guo, J. Zhao, and F. Shen, “Image Data Augmentation for Deep Learning: A Survey,” arXiv (Cornell University), Apr. 2022, doi: 10.48550/arxiv.2204.08610.

[53] C. Shorten and T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” Journal of Big Data, vol. 6, no. 1, Jul. 2019, doi: 10.1186/s40537-019-0197-0.

[54] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Improved Texture Networks: Maximizing Quality and Diversity in Feed-Forward Stylization and Texture Synthesis,” Computer Vision and Pattern Recognition, Jan. 2017, doi: 10.1109/cvpr.2017.437.

[55] X. Yi, E. Walia, and P. Babyn, “Generative adversarial network in medical imaging: A review,” Med. Image Anal., vol. 58, 2019, doi: 10.1016/j.media.2019.101552.

[56] J. Feng et al., “Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification,” Remote Sens., vol. 12, no. 7, 2020, doi: 10.3390/rs12071149.

[57] M. Mirza and S. Osindero, “Conditional Generative Adversarial Nets,” arXiv (Cornell University), Nov. 2014.

[58] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-Excitation Networks,” arXiv (Cornell University), Jun. 2018, doi: 10.1109/cvpr.2018.00745.

[59] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training GANs,” Neural Information Processing Systems, vol. 29, pp. 2234–2242, Dec. 2016,

[60] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessment: From Error Visibility to Structural Similarity,” IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, Apr. 2004, doi: 10.1109/tip.2003.819861.

[61] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018).

[62] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium,” Neural Information Processing Systems, vol. 30, pp. 6626–6637, Jan. 2017,

[63] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. 2016. doi: 10.1109/cvpr.2016.124.

Appendix The Published Paper

Formal Acceptance

3rd International Conference of Information Technology to enhance E-learning and other Application-2022

[IT-ELA 2022]

To/ Mohammad Baqer Haleem; Israa Hadi Ali

Dear respected author(s):

With heartiest congratulations. We are pleased to inform you that based on the recommendations of the reviewers and the Technical Committees, your paper entitled:

"Survey on Human Pose Transfer Based on Deep Learning"

It has been accepted for oral presentation at the 3rd International Conference of Information Technology to enhance E-learning and other application-2022[IT-ELA 2022], technically sponsored by IEEE, to be held on December 27-28, 2022, Baghdad, Iraq.

Your paper will be submitted (**within the conference proceeding**) to the IEEE Xplore digital library for (**final acceptance of uploading to the Digital library**).

IT-ELA 2022 Baghdad – Iraq

Email:it-ela2022@baghdadcollege.edu.iq

Website: https://baghdadcollege.edu.iq/it-ela2022

Baghdad College of Economic Sciences University

Computer Sciences Department

ID: 1570869468

Baghdad college of Economic Sciences University
Computer Science Department

Certificate of Participation

This Certificate is Proudly Presented to

Mohammad Baqer Haleem

for presenting your paper entitled " Survey on Human Pose Transfer Based on Deep Learning " in the "3rd International Conference on Information Technology to enhance E-Learning and other Application - 2022", [3rd IT-ELA 2022]. This conference organized by the Computer Science Department / Baghdad College of Economic Sciences, University, with scientific sponsorship from IEEE, Iraq section and held from 27 -28 December 2022, in Baghdad, Iraq.

Prof. Dr. Qasim Naif Alwan
Dean of the Baghdad College of Economic Sciences University

Prof.Dr. Eng. Sattar B. Sadkhan
Representative of IEEE IRAQ Section

Assist Prof. Dr. Kawther Abbood Neamah
Head of Computer Sciences Department Baghdad College of Economic Sciences University

الخلاصة

تحويل هيئة الإنسان هو عملية نقل مظهر الصورة المصدر إلى وضعية الصورة الوجهة. يحتوي هذا الموضوع على مجموعة واسعة من التطبيقات، بما في ذلك إنشاء الفيديو والرسوم المتحركة القائمة على الصور وزيادة البيانات لتدريب أنظمة تقدير الوضع البشري وإعادة تعريف الشخص ونماذج التحليل البشري. باستخدام صورة واحدة ثنائية الأبعاد لشخص بشري، كانت مهمة صعبة في Computer Vision لإنتاج صور واقعية بناءً على الوضع فقط. يمكن لمفهوم نقل الوضع البشري أن يساعد في إنشاء صور متعددة الوضعيات لنفس الفرد. يحاول نقل وضع الإنسان، جنبًا إلى جنب مع Human Parsing، إنشاء صورة جديدة لشخص من صورة لهذا الشخص ووضع مستهدف. مع الاحتفاظ بمظهر الصورة الأصلية.

تم استخدام Human Parsing Key-Points معًا لإنشاء الصورة المستهدفة. يتكون النظام المقترح من أربع مراحل. المرحلة الأولى هي إعداد مجموعة البيانات التي تتضمن استخراج الميزات المطلوبة وتخزينها. المرحلة الثانية هي المعالجة المسبقة للصور والميزات المستخرجة من المرحلة الأولى لتكون مناسبة لدخول النموذج. المرحلة الثالثة هي توليد Parsing map تتماشى مع الوضع المستهدف لتمثيل شكل الملابس بواسطة Parsing Generator. تستخدم المرحلة الرابعة Image Generator لنقل الصورة المصدر لمحاذاة الوضع المستهدف بمساعدة Parsing الناتج من المرحلة الأولى. تتضمن شبكة الخصومة التوليدية متعددة المستويات المقترحة (MLA-GAN) مستويين من ال Attention، الأول هو على مستوى البكسل الذي يمثله Gated-Conv والثاني هو على مستوى القناة الذي يمثله Squeeze and Excitation Block. أيضًا، استخدمنا GP-WGAN بدلاً من ال Loss function الاصلية.

مجموعة البيانات المستخدمة هي DeepFashion مع 48 ألف صورة تدريبية وصور اختبار 8ألف. توضح النتائج التجريبية أن نموذجنا يحقق أداءً لائقًا مقارنةً بالأساليب الحديثة الأخرى مع عدد أقل من ال Parameters. لقد استخدمنا أربعة مقاييس (SSIM ،IS ، FID،LPIPS) أول مقياسين أكثر أهمية من باقي المقاييس لأنهما أقرب إلى الحكم البشري. لقد حققنا المركز الأول في FID والثالث في البقية. على الرغم من أننا لم ندرب نموذجنا بشكل كامل بسبب نقص قوة الحساب (computation power)، إلا أننا قمنا فقط بتدريب النموذج على 50 دورة بينما كان الهدف المقصود 150 دورة.

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعــة بابـل

كلية تكنولوجيا المعلومات

قسم البرمجيات

تحويل هيئة الإنسان باستخدام شبكات توليدية تخاصمية متعددة المستويات للانتباه

رسالة مقدمه الى

مجلس كلية تكنولوجيا المعلومات – جامعة بابل كجزء من متطلبات نيل درجة

الماجستير في تكنولوجيا المعلومات / البرمجيات

من قبل

محمـد باقـر حليم محمـد هلال

بإشراف

أ. د. اسـراء هـادي علـي

1444هـ

2023 م