Universidade do Minho
Escola de Engenharia

João Emanuel Lavinas Gomes Lopes

Segmentation of surgical tools from laparoscopy images

outubro de 2023

Segmentation of surgical tools from laparoscopy images
João Emanuel Lavinas Gomes Lopes
UMinho | 2023

Universidade do Minho
Escola de Engenharia

João Emanuel Lavinas Gomes Lopes

Segmentation of surgical tools from laparoscopy images

Relatório de Projeto Individual [Dissertação]
Mestrado em Engenharia Biomédica

Trabalho efetuado sob a orientação do
Professor Doutor Carlos Manuel Gregório Santos Lima
e do
Doutor Nuno Renato Azevedo de Freitas

outubro de 2023

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Acknowledgements

Em primeiro lugar agradeço ao Professor Doutor Carlos Lima pela sua orientação durante este ano e pelo auxílio no planeamento das metodologias adotadas. Agradeço também ao Doutor Nuno Renato Freitas por me ter aturado durante este tempo todo, nem sei como é que ele aguentou.

Obrigado a um grupo de amigos especial, Bioquartas, composto pela Cati, Cocky, Horst, Rúben, Sotaquesss, Turquia, Velma e Zener, por me terem aturado durante este ano e em muitas ocasiões que não vale a pena numerar porque assim não se saia daqui. Agradeço também ao grupo dos Bimbos pelo espírito animado que me levantava em dias difíceis.

Um forte agradecimento à minha família, os meus pais, Isildo e Fernanda, e ao meu avô, João. Vocês foram espetaculares nestes dias todos, obrigado pelo amor e pelo aconchego que me deram ao longo da minha vida.

Declaração de Integridade

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio, nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Resumo

Cirurgias roboticamente assistidas têm vindo a substituir as cirurgias abertas com enorme impacto no tempo de convalescença do paciente e consequentemente em tudo o que isso implica, economia de recursos no sector da saúde e a retoma antecipada das atividades laborais do paciente. Este tipo de cirurgia auxiliada por um sistema robótico é guiado por uma câmara laparoscópica, facultando ao médico uma visão das partes anatómicas do paciente. A fim do cirurgião se encontrar apto para operar este equipamento tem de passar por inúmeras horas de formação, tornando o processo desgastante e dispendioso. Para além do referido, a manipulação dos instrumentos cirúrgicos em concordância com a câmara laparoscópica não é de todo um processo intuitivo, ou seja, os erros de natureza subjetiva não são erradicados. A diretiva desta tese é o desenvolvimento de um sistema automático capaz de segmentar instrumentos cirúrgicos, possibilitando desta forma a monitorização constante da posição dos instrumentos. Para tal foram explorados diferentes modelos de aprendizagem automática. Numa segunda fase, foram considerados métodos que pudessem ser incorporados no modelo base. Tendo-se encontrado uma resposta, partiu-se para a comparação dos modelos previamente selecionados, com o modelo base e ainda com o otimizado. Numa terceira abordagem, de forma a melhorar as métricas que serviram de comparação, procurou-se por soluções alternativas, nomeadamente a geração de dados artificiais. Neste ponto, deparou-se com duas possibilidades, uma baseada em sistemas de aprendizagem autónoma por competição e outra em sistemas de aprendizagem de síntese de imagens a partir de ruido com densidade espectral sucessivamente incrementada. Ambas as abordagens permitiram o aumento da base de dados tendo-se aferido a sua eficácia por comparação do efeito do aumento de dados nos sistemas de segmentação. O sistema proposto pode vir a ser implementado em cirurgias roboticamente assistidas, necessitando apenas de mínimas alterações.

Palavras-chave: câmara laparoscópica**,** cirurgia robótica assistida, geração, segmentação semântica.

Abstract

Robotic-assisted surgeries have been replacing open surgeries with a significant impact on patient recovery time, and consequently, on various aspects such as healthcare resource savings and the early resumption of the patient's work activities. This type of surgery, assisted by a robotic system, is guided by a laparoscopic camera, providing the surgeon with a view of the patient's anatomical structures. To operate this equipment, surgeons must undergo numerous hours of training, making the process exhaustive and costly. In addition, manipulating surgical instruments in coordination with the laparoscopic camera is not an intuitive process, meaning errors of a subjective nature are not eliminated. The objective of this thesis is the development of an automated system capable of segmenting surgical instruments, thereby enabling constant monitoring of their positions. Various *machine learning* models were explored to address this issue. In a second phase, methods that could be incorporated into the base model were considered. Once a solution was found, a comparison was made between the previously selected models, the base model, and the optimized model. In a third approach, with the aim of improving the comparison metrics, alternative solutions were sought, including the generation of synthetic data. At this point, two possibilities were encountered, one based on autonomous learning systems through competition and the other on image synthesis learning systems from progressively increasing noise spectral density. Both approaches expanded the available database, and their effectiveness was evaluated by comparing the impact of data augmentation on segmentation systems. The proposed system can potentially be implemented in robotic-assisted surgeries with minimal modifications.

Keywords: generation, laparoscopic camera, segmentation, robotic-assisted surgery

Index

Figures Index

Table Index

List of Abbreviations

Adam	Adaptative Moment Estimation
AESOP	Automated Endoscopic System for Optimal Positioning
AI	Artificial Intelligent
C	Clipper
CD	Clipping Device
CDF	Dissector Forceps
CI	Coagulation Instrument
CNN	Convolutional Neural Network
ConvGRU	Convolutional Gated Recurrent Unit
DDPM	Denoising Diffusion Probabilistic Model
E	Exhauster
FCN	Fully Connected Network
FID	Fréchet Inception Distance
G	Grasper
GAN	Generative Adversarial Networks
GRU	Gated Recurrent Unit
HI	Holding Instrument
HOG	Histogram of Oriented Gradients
HSL	hue-saturation-lightness
HSV	hue-saturation-value
IoU	Intersection over Union
LBP	Local Binary Patterns
LG	Long Grasper
LPIPS	Learned Perceptual Image Patch Similarity
LSTM	Long Short Term Memory
mAP	mean Average Precision

Mask R-CNN	Mask Region-based Convolutional Neural Network
ML	Machine Learning
MLP	Multi Layer Perception
MS	Monopolar Spatula
NCSN	Noise-Conditioned Score Networks
ND	Needle Driver
NHD	Normalized Hausdorff Distance
NN	Neural Network
ORB	Oriented FAST
PSNR	Peak Signal-to-Noise Ratio
R-CNN	Region Based Convolutional Neural Network
ReLU	Rectified Linear Unit
ResNet	Residual Neural Network
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RPN	Region Proposal Network
S	Scissor
SB	Specimen Bag
Sc	Scissors
SDE	Stochastic Differential Equations
SDM	Semantic Diffusion Model
SGD	Stochastic Gradient Descendent
SI	Suction/Irrigation
SIFT	Scale-Invariant Feature Transform
SSIM	Structural Similarity Index
SURF	Speeded-Up Robust Feature
St	Stapler
SVM	Support Vector Machine
T	Trocar

TanH	Hyperbolic Tangent
UI	Undefined Instrument
VAE	Variational Autoencoder
VGG	Visual Geometry Group
YOLO	You Only Look Once

Chapter 1 Introduction

In robotically assisted surgery one of the fundamental problems is merely mechanical in origin. Given the constitution of a mechanical arm, what is the estimated movement of each part of the arm to reach a target position? The answer to this question requires knowledge of the structure of the arm and the transmission of movements between the various parts of it. This knowledge can only be obtained by the structural representation of the arm and the instantaneous position of its parts which in turn can be obtained by using semantic segmentation procedures.

The main objective of this thesis is the development of a *machine learning* algorithm to automatically segment parts of surgical instruments and the type in laparoscopic images. This chapter provides information of the motivation behind this dissertation describing the problem and the proposed objectives.

1.1 Motivation

Robotic-assisted surgery is revolutionizing surgical practices by improving conventional procedures and reducing the risk of postoperative complications. Unlike industrial robots designed for repetitive tasks, surgical robots are tailored for dynamic and shifting environments [1], hence a much more challenging problem requiring the best artificial intelligence has to offer. Despite its very high cost of approximately 2.25 million euros per unit robotic surgical systems will be essential in modern hospitals because they are essential for patients' well-being (physically and mentally), allow a relatively quick return on investment for the hospital by reducing length of stay and post-surgical complications, for the national health system by improving the overall performance of hospitals and for the community by reducing work complications. With all this panoply of advantages, assisted surgery will certainly be the most used in the near future for an increasingly extensive range of interventions if it proves to be reliable

and safe, which is a huge challenge not so much for medicine but for contemporary engineering.

Prostatectomy, the removal of the prostate gland, is a common surgery often performed with robotic assistance. Between 2005 and 2008, the number of discharges for prostatectomy increased by more than 60%, according to the Nationwide Inpatient Sample. This surgery is frequently associated with prostate cancer, encouraging patients to undergo the procedure. In Portugal, the use of robotic-assisted surgery led to a 31% increase in prostatectomy procedures in 2011 [2],[3].

Beyond patient benefits, medical professionals require a clear view of the anatomical body and precise tactile feedback during surgery. Robotic-assisted surgery involves the use of laparoscopic images captured throughout the procedure. Surgeons may encounter challenges such as loss of visual field or obscured images due to inexperience or issues like tissue burning. To minimize such constraints, surgeons need extensive training, often requiring at least 150 procedures [2]. While robotic-assisted surgery incurs higher training costs than open surgery, these expenses can be mitigated through the implementation of computer-assisted techniques.

Computer vision has made significant advancements in various scientific fields, including engineering, mathematics, and medicine. In medical image analysis, robots have been utilized for pre-operative planning of procedures like aortic valve surgeries (e.g., 3mensio Structural Heart [4]). The next logical step is integrating robotic-assisted surgery with these technologies. To achieve comprehensive solutions, several applications must be developed, including full-scene segmentation, instrument segmentation and monitoring, and workflow recognition. Instrument segmentation, a focus of this work, is fundamental for future all-encompassing solutions. Identifying entire instruments and their components within a surgeon's field of view prevents instrument misplacement and enables real-time monitoring. Emphasis should be placed on developing models that can robustly handle segmentations. However, due to patient variability, encoding diverse clinical knowledge into AI systems presents challenges. These solutions require extensive and diverse datasets, which are often not readily

available. As a result, alternative approaches to overcome these limitations must also be developed.

1.2 Objectives

The main objective of this thesis is to study methods capable of segment surgical instruments used throughout a robotic-assisted surgery. These methods are based in *deep learning* strategies using laparoscopic images, and can be sorted into three parts:

The first line of thinking for this work was the search for different *deep learning* models capable of doing semantic segmentation. This study focused more in models that could outperform models already explored for this task.

The second phase was the study of different ways of increasing the scores achieved by the elected model. In this point techniques that could be implemented in the model without increasing exponentially the number of its parameters were employed. It was also witnessed that methodologies like transfer learning enhance models' capabilities.

The last point was trying to solve the problem of lack of data. The model without a high number of images with enough diversity, can not extract features that can fulfil the requirement of this thesis. Therefore, methods that can add more information about the data through image generation were searched. Synthetic data generation is a valid technique for increasing the dataset size and provides better scoring results [5], [6].

Chapter 2 Robotic-assisted surgery: overview

The laparoscopic surgery, which is a surgery guided by a laparoscope, has been assisted by different types of robotic forms, during the last decades. The first commercially available robot was the Automated Endoscopic System for Optimal Positioning (AESOP), a robot arm with a camera. The robot had the ability to imitate a human arm, allowing the surgeon to have access to a flexible operative field. The arm control could be made by voice commands, that were previously recorded, or by a foot pedal [7]. Over the years, all the developed robotic arms were passive systems, which means that the robots only increase the window vision of the surgeon, through a laparoscope, and do not directly intervene on the surgery. Until the early ’90s, the main developments in this area were a robotic camera synchronized with the surgeon head movement, which was made by Armstrong Healthcare [7] . The technology progressed over the years, but dexterity remained a serious problem. The first master-slave manipulators were developed during the 1990s. These manipulators obligated the surgeon to control the system from a remote cabinet. A computer was located between the surgeon and the surgical instruments, where the hands’ actions of the surgeon were reflected in movements of instruments inside the patient’s body. The main issue of this technique was the lack of freedom because until 1992 the robot’s arms only had 4 degrees of freedom. During the mentioned year, in Germany, the first system with 6 degrees of freedom was made, where the arms were controlled by a joystick. Although it’s a revolutionary technology, the system never had a clinical application [8]. At the same time, two companies from the US worked on telemanipulation systems, to make the surgery as minimally invasive as possible. The first one, mentioned as the ZEUS robotic arm surgical system, was composed of a control system that allowed the manipulation of three robotic arms at once on the operation table. Two of the arms had only 4 degrees of freedom, having a variety of surgical tools, while the remaining arm was an endoscope holder, that could be controlled by voice. The controller, which was manipulated by the surgeon, has the ability to filter the tremors making scale-down

movements by a factor of 2 to 10. The visual field was a screen located on the console, so it was a two-dimensional image [9], [10]. Although the ZEUS robot was capable of passing through the test and used for surgery, it was discontinued after the company was purchased by Intuitive Surgical. Intuitive Surgical remade the robotic surgical system, adapting the arms to incorporate 7 degrees of freedom and integrating force feedback with a 3D vision. This innovate robot model was called by da Vinci and is still referred to by this name up to the present[7], [9]. The most recent technology searches for a robotic-assisted surgery with a minimally invasive as possible. The surgical procedure looks after to mitigate the trauma on the surrounding tissues. During the surgery, the surgeon does not touch or directly see the patient's body, but he visualizes the surgical field and manipulates the tools through a robot, instead [11]. This type of surgical procedure has three components: the surgeon console, the articulated arms, and the imaging system. The surgeon console is where the surgeon can control the different available surgical instruments and can see the operation field through a binocular stereoscopic camera. The control is composed by two joysticks, where one permits the instruments' movements and the other adjusts the lens. The binocular stereoscopic camera contains two 5mm diameter cameras, each one responsible for transmitting information for one eye, allowing a 3D vision. The trolley, that supports the robotic arms, in the more recent da Vinci models, has four arms, where one holds the laparoscopic camera, and the other three hold the different surgical instruments. The last part, which is the imaging system, is composed of an insufflator, a light source, and a dual camera. This structure is allocated in the operating room, so the image could be transmitted on the surgical field[10].

2.1 Clinical context

Robotic-assisted surgery could be applied in different type of diseases. Diseases can be split in two different sets: benign and malignant. Bening diseases are characterized for being tumors that are not cancerous, i.e., they do not spread to the nearby tissues or to the other parts of the body. This type of disease is armful, however

it can cause pain and other type of problems when they make pressure in nerves or bloodstreams. Usually, when they are removed do not return. In contrast to this condition, malignant diseases are cancerous tumors that grow rapidly and can spread to other organs via the bloodstream. Malignant diseases pose a significant threat to the individual's life. In these types of illnesses, complete removal of the tumor is necessary because any remaining cells have the potential to grow and form a new tumor hence the need of a surgery of high precision to access all the cancerous tissue.

The next sub-sections expose the different types of benign and malignant diseases that can be treated by using robotic-assisted surgery.

2.1.1 Benign diseases

Gallbladder is an organ responsible for creating the bile, which helps to digest fat. In this organ is very common the occurrence of benign diseases, such as gallstones, cholesterol polyps or even adenomas. When this illness is detected, is necessary the organ removal.

Robotic-assisted surgery could be used for cholecystectomy, which is a procedure to remove the gallbladder. The technical approaches for the robotic cholecystectomy are 4: multiport technique, single port technique, robotic common bile duct exploration, and Robotic Hepaticojejunostomy. **The multiport technique**, as the name indicates, uses 4 entry points, where three are for the instruments and the last one for the camera. At the beginning of the surgery, each trocar is placed 10cm away from the patient body, and the distance between each other should be 6cm, to avoid interference. Figure 1 illustrates a schematic of the robotic surgery by multiport:

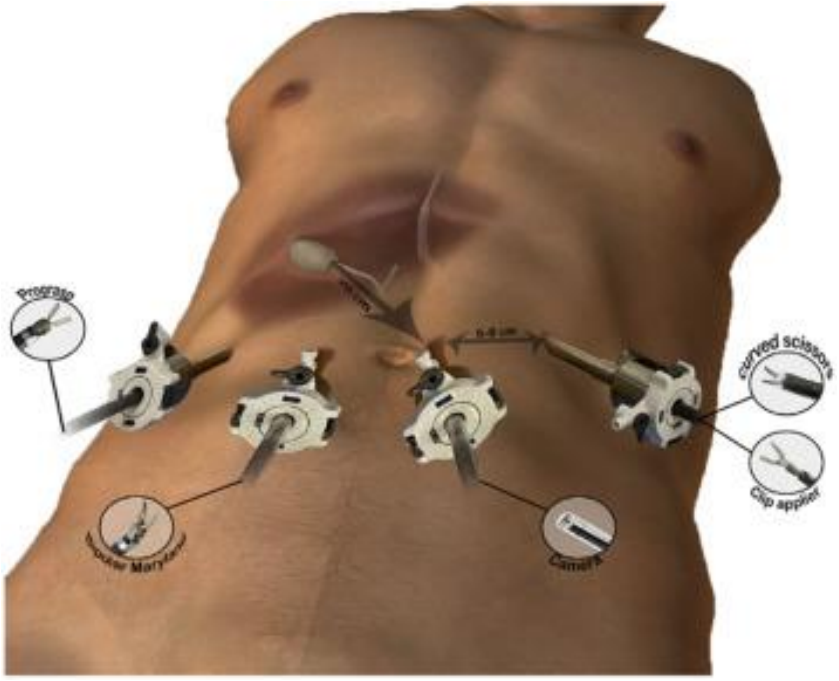

Figure 1. Instrumentation for robotic multiport cholecystectomy [12].

The dissection begins by grasping the Hartmann's pouch, which is located on top of the gallbladder, and then to have a clear view of the cystic duct (the connection between the gallbladder and the bile duct), the cystic duct is pulled to a 90º angle. To identify the cystic duct, to achieve a critical view safety, the Calot's triangle (small anatomical space in the abdomen that is located at the porta hepatis of liver) must be free of fatty tissue, and only two structures should be seen entering the gallbladder. Once the critical view safety is achieved, two clips are placed on the cystic duct and then the gallbladder is removed from its bed. After this process, gallbladder is detached following an absorbable suture to avoid hernias[12].

In the **single port technique**, unlike the previous one, there is only one entry point for the anatomy body. The entry point is made by a 3cm vertical incision in the umbilicus and next a transverse incision is performed. The tools used are more flexible, which allows their insertion into curved cannulas, that are placed in the entry point [12]. The remaining process is identical to the previous one. Figure 2 illustrates a single-port technique for a cholecystectomy surgery:

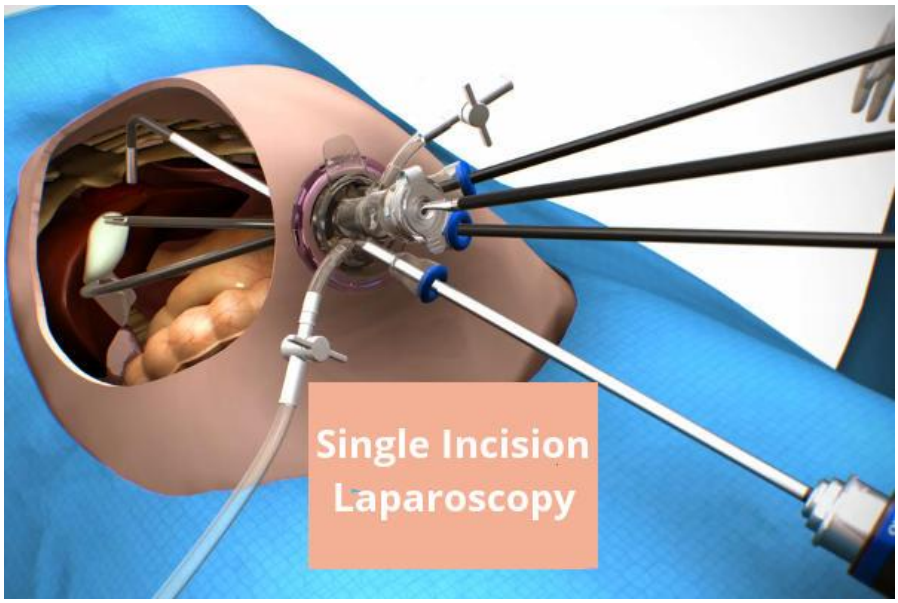

Figure 2. Single port technique instrumentation [12].

The next technique, the **robotic Common Bile Duct Exploration**, is a common procedure to remove stones when they cannot be cleared via transcystic. So, this procedure requires an extra instrument that is placed in the left lateral abdomen [12], [13].

The last one, the **robotic Hepaticojejunostomy**, is a surgery that seeks to create communication between the hepatic duct and the jejunum [14].

The robotic-assisted surgery is a revolutionary technique that minimizes the post-operatory traumas and increases the recovery speed. Patients report less pain after the procedure, due to the lack of extensive incisions, because the incisions on the abdominal body are not more than 20 mm in diameter [8], [15]. Although the equipment is very expensive and not affordable in all countries, in a long-term perspective the investment could yield returns due to the absence of a long-term need nursing care. Robotic surgery, due to small incisions, reports less blood loss and lower risks of infection [11].

2.1.2 Malignant diseases

Beyond of mentioned, robotic-assisted surgery is often performed in **cancer surgery**. The cancer kills millions of persons every year. The surgical intervention is the primary resource to cure from solid organ cancers. According to the surgeon policy, the surgery should remove the tumor completely and minimize the surgical post-trauma of the patient. This principle guarantees quality of life and decrease the morbidity. Robotic-assisted surgery is often the chosen approach because allows the surgeons to perform complex surgeries with more precision than open surgery. This surgical procedure allows the physician to effectively treat the cancer from the patient, facilitating a significantly quicker and smoother recovery. Depending on the cancer stage, the surgeon has three possibilities: remove the tumor completely, reduce the tumor size so the patient could undergo to other treatment or relieve the pain [16].

There are many cancers that are treated with robotic surgery, like colorectal, pancreatic, thoracic, and thyroid. **Colorectal** cancer is the third most common cancer diagnosed in both men and women, causing 53000 deaths per year. This type of cancer appears in the colon and rectum. The cancer starts when the cells' body grow out of control. The colorectal cancer can be seen in Figure 3:

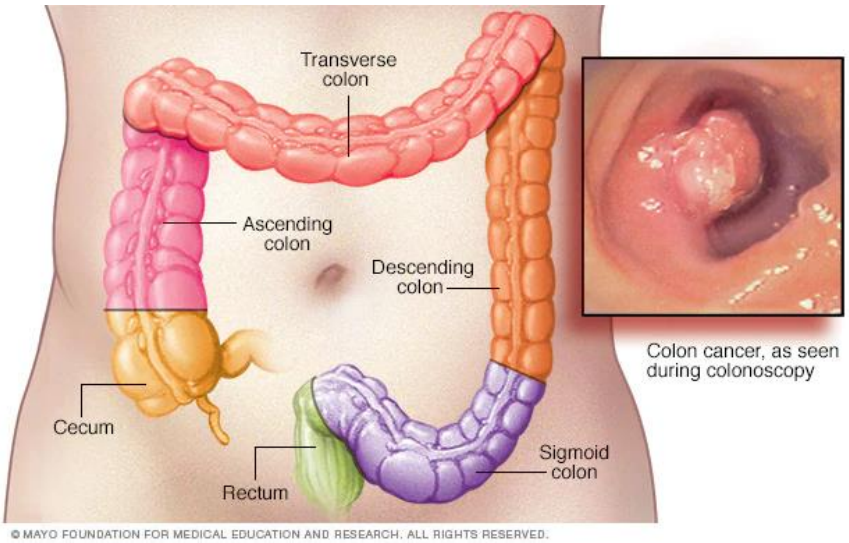

Figure 3. Schematic of a colorectal cancer [17].

Robotic surgery for colorectal cancer allows the surgeon to perform the procedure with less blood loss, due to small incisions. Furthermore, it provides to the

physician a 3D visual field, helping the robotic arms manipulation, eliminated surgical tools vibrations, permitting a more precision surgery. Moreover, due to the robotic arms ergonomic, with a wrist with 180º degrees of freedom, allows meticulous dissections. The meticulous dissection is also useful to preserve nerves from the pelvic area. In Addition, the surgeon can perform the surgery seated. With robotic-assisted surgery, the recovery time only take 2 to 3 weeks, unlike the conventional that takes 4 to 6 week [17], [18].

Pancreas is located behind the lower part of the stomach, providing enzymes that help to digest food. There are many types of **pancreatic cancer**, being the most common the pancreatic ductal adenocarcinoma. This type of cancer starts in the ducts that carry the enzymes away from the pancreas. For this oncology problem, the surgeon has two options: excise the entire mass of the pancreas or decrease the tumor size, relieving the symptoms. The main problem of this type of intervention is the pos-operative recovery because in the open surgeries it is needed an exploration approach, unlike robotic-assisted surgery, in order to achieve the pancreas. Another problem in the conventional techniques is the blood loss because the pancreas removal, named as pancreatomy. In robotic-assisted surgeries, this is not the case because of the small incisions made in the patient's body and the precision of the dissections [19]. Moreover, robotic-assisted surgery can reduce the recovery period to as little as 2 weeks, whereas the traditional surgery typically requires a month. For a better understanding of pancreatomy, Figure 4 represents a procedure assisted by robotic arms.

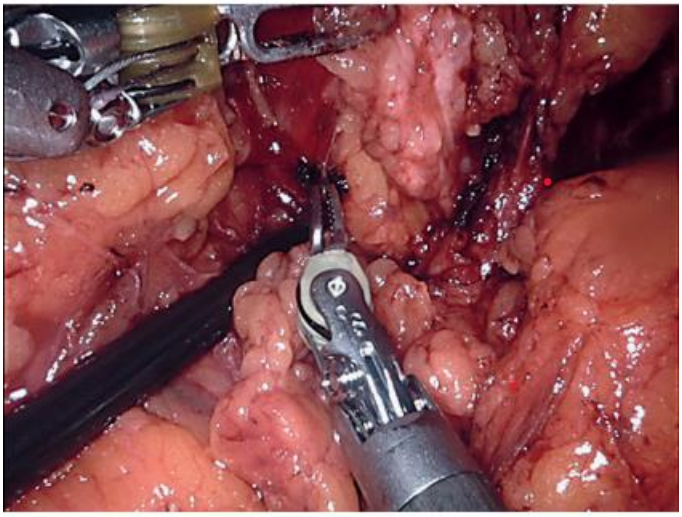

Figure 4. Pancreatectomy procedure by robotic-assited surgery [19].

The **thoracic cancer** is the second most common cancer in both men and women. According to de data, each year there are roughly 2,000,000 cases of lung cancer and 1,800,000 deaths worldwide. In order to remove the tumor from the lungs a lobectomy is needed, which is a surgical intervention that removes an entire lube from the lungs. Figure 5 shows a traditional lobectomy intervention.

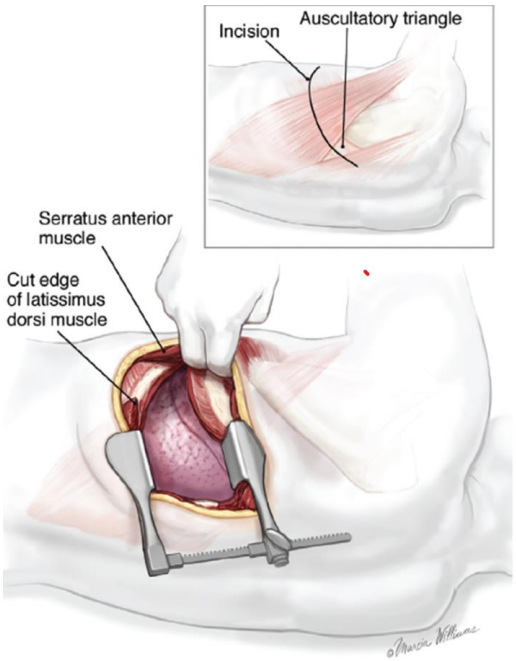

Figure 5. Traditional lobectomy procedure [20].

In the traditional way, it required the opening of a large incision, breaking of chest bones and even the spreading the chest. This procedure requires a lot of recovery time, which could be mitigate by the implementation of robotic-assisted surgery. The surgeries assisted by robotic arms avoid the breaking bones and the spreading of the chest because the incisions can pass through the intercostal spaces, without damaging the bones. This surgery must be performed with the patient in a lateral position, where the robot accesses to entire length of the patient back. With thoracic robotic-assisted surgery the patient's recovery time is 2 to 4 weeks, while in the traditional surgery takes 6 to 8 weeks.

The thoracic cancer is very related to lifestyle factors, like smoke and diet, which are the main problems of the society today [21].

The **thyroid cancer** is a disease that can be treated by a robotic-assisted intervention. The thyroid is a gland located at the throat with a shape of a butterfly. This type of cancer is more common in women than men. Like all cancer there are different types of thyroid cancer, being the most frequent the papillary thyroid cancer. This cancer requires the removal of the thyroid glands by a surgical intervention, named thyroidectomy. The traditional thyroidectomy involves the cut of the whole throat, that may damage the muscles and nerves, as it can see in Figure 6.

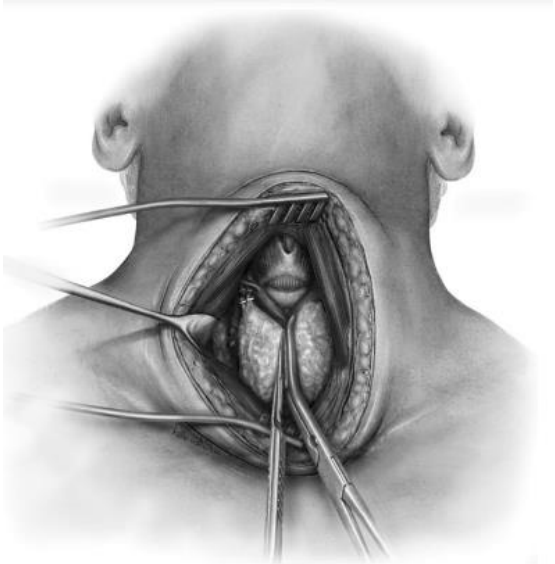

Figure 6. Representation of a conventional thyroidectomy surgery [22].

The thyroidectomy surgery with robotic arms avoids this issue because the organ access is made by the armpit. The robot is positioned so that the arms of the robot would be perpendicular to the patient throat. After the patient recovery, which only takes 1 week, the individual only has a scar below his arm. Also, the patient does not require to be in bed all the time just 24 hours after the surgery. The conventional procedure, due to its invasive intervention as mentioned before, the patient needs 3 weeks to be fully recovered.

Like all techniques that possess both advantages and disadvantages, this one is no exception. While performing the surgery, surgeon has lacks needs to remain in an extremely static position, which does not happen in an open surgery. Managing long instruments that go through the fixed entry points while looking at a screen reduces the feedback. Another issue is the learning curve, which is a slow process because surgeons during their academic phase learn the open operations and its associated tactile feedback. They then need to translate this tactile feedback into robotic feedback, which is not an intuitive process [11]. Furthermore, robotic-assisted surgery depends on the skillfulness of the surgeon, because, as mentioned before, he needs to coordinate the movement of several surgical tools with a small visible field. Therefore, *machine learning* systems play a crucial role because they can help the surgeon to identify the instrument location and avoid wrong organs dissection in such environments. From a futuristic perspective, *machine learning* algorithms have the potential to become an assistant during surgery, helping with more robust decisions.

Chapter 3 Machine Learning Overview

Intelligence is vague to define being usually related with the capacity for an entity to reason, learn, and adapt to new situations which are the main ingredients to create new knowledge. Human intelligence in particular is highly complex and multifaceted, involving skills like language, abstract thinking, creativity, and emotional intelligence which allow humans to solve complex problems. Artificial intelligence (AI) is a field of computer science that involves the development of computer systems that can perform tasks that would typically require human intelligence to complete. These systems are designed to learn, reason, and make decisions based on data, with the goal of achieving human-like or superhuman performance in specific areas. *Machine learning* is just a subfield of AI that uses algorithms trained on data to produce adaptable models that can perform a variety of complex tasks.

This chapter approaches general concepts of *machine learning*, such as the types of learning techniques and the main obstacles within. Then it is presented a classification problem and how the learning process could be addressed, including an overview of different objective functions what the *machine learning* model searches to minimize/maximize. Afterwards, the main process behind the model optimization and the proceeding optimization are both introduced. A brief overview of some conventional learning models and some basics of *deep learning* models are quoted.

3.1 General concepts

Machine learning (ML) is a sub-field of AI, consisting of a learnable system capable of detecting patterns in a dataset and providing an answer to a specific task, such as classification, object segmentation and detection, regression, and clustering. Overall, ML enables computers to learn from historical data using probabilistic theories, resulting in a model that can be used to make accurate predictions about past, current, and future events.

3.1.1 Training strategies

ML models can be divided into three main categories depending on the task and the training data, namely *supervised learning*, *unsupervised learning* and *reinforcement learning*. ***Supervised learning*** is an approach whose primary goal is to map the input x to the output labels y. To train these models, the dataset consists of input-output pairs, $D = \{(x_i, y_i)\}_{i=1}^{N}$, where D is the training set and N the number of training samples. Each input, x_i, is a D-dimensional vector composed of features, attributes or covariates that describe the characteristics of a complex object, such as an image or a graph. Similarly, y_i denotes the target variable which in most cases is a categorical (classification or patterns recognition) or nominal variable (regression to predict real-valued outputs). The second type is ***unsupervised learning***, where D is an unlabeled set composed just of x_i, $D = \{(x_i)\}_{i=1}^{N}$. The main task is to find implicit patterns in data without any guidance from a target variable. In these problems, the decision boundaries are not well delineated, as the system lacks awareness of the patterns it aims to discern. Consequently, there is an absence of a clearly defined metric to quantify errors. The last type of ML is known as ***reinforcement learning***. This type of learning involves training models, also known as agents, to make decisions in a trial-and-error approach to maximize cumulative reward over time. The system learns by interacting with an environment and acting according to rewards or punishments [23].

3.1.2 Training control

Controlling the training process in supervised learning-based systems is fundamental to obtaining an accurate model and requires a tradeoff between accuracy and generalizability. Systems with very high accuracy in the training set usually degrade significantly when used in unseen data because excessive details of data, such as noise and stochasticity, have been learned as essential characteristics for data modelling. While accuracy can be measured and controlled in the training set generalizability requires the use of the test set. Therefore, a third independent and usually small data set, the validation set, must be used to simultaneously measure accuracy and

generalizability allowing to establish the required trade-off. When this trade-off is not accomplished, we fall in one of the two obstacles in the training of *machine learning* systems: *underfitting* and *overfitting*. ***Underfitting*** happens when the model doesn't acquire an error low enough during the training phase, while ***overfitting*** is when the model learns noise and stochastic variations from data and propagates them, developing a disparity between the training error and the testing error [24], [25].

3.2 Conventional machine learning models

Conventional *machine learning* models are nowadays considered all those that do not include *deep learning*, with the exception of transformers. In fact, the use of Convolutional Neural Networks (CNNs) established a border point in the classification of ML models since handcrafted feature extraction modules are no longer necessary being replaced by optimal features obtained from CNNs training.

Regarding training strategies, the most usual is the *supervised training* except when labeling is difficult or even not possible. Supervised training strategy allows a better use of the training set for learning purposes, especially when compared to unsupervised learning where no training data is used. Nowadays, there is a panoply of *machine learning* models, such as: *Nearest Neighbor*, *Naïve Bayes*, Decision Trees, Linear Regression, Support Vector Machines (SVM), Neural Networks (NN) [26]. In this section the *Multilayer Perceptron* (MLP) is used as an example of *machine learning* model. The explanation of how it works and what is behind its operating system is provided, as well as the backpropagation algorithm, the optimization and objective functions.

3.2.1 MLP

This type of NN is inspired by biological neurons, which is a set of processing elements, named nodes, interconnected with each other by weighted connections (weights). These connections are arranged in a specific architecture, complemented by a learning algorithm. MLP has an input and output layer, with one or more paired neurons as hidden layers. Each neuron is seen as a regression model or logistic

regression, depending on whether it is a regression or a classification problem, respectively [27]. Figure 7 shows a schematic representation of an MLP structure:

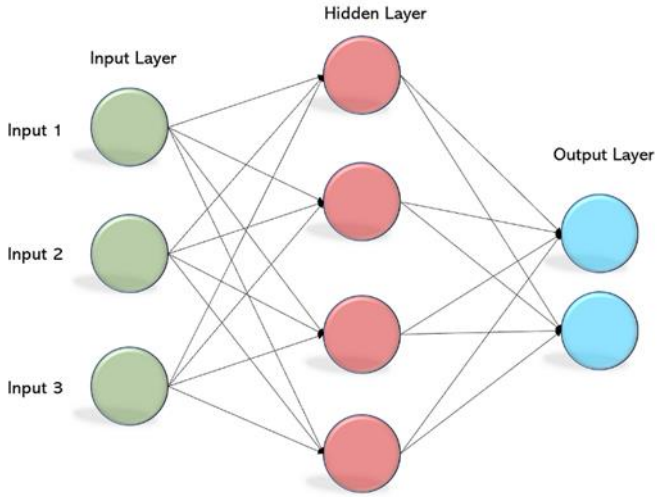

Figure 7. Schematic representation of a MLP structure [27].

In this structure, each node of the hidden layer receives a signal from previous neurons and apply to the signal an affine transformation. Subsequently, an activation function is employed with the aim of converting the problem into a non-linear one. The MLP could use **non-linear activation functions**, such as *Sigmoid*, *Hyperbolic Tangent* (*TanH*), *Rectified Linear Unit* (*ReLU*) and *Leaky (ReLU)*, being the last the most resorted [26]. Figure 8 shows the mentioned activation functions:

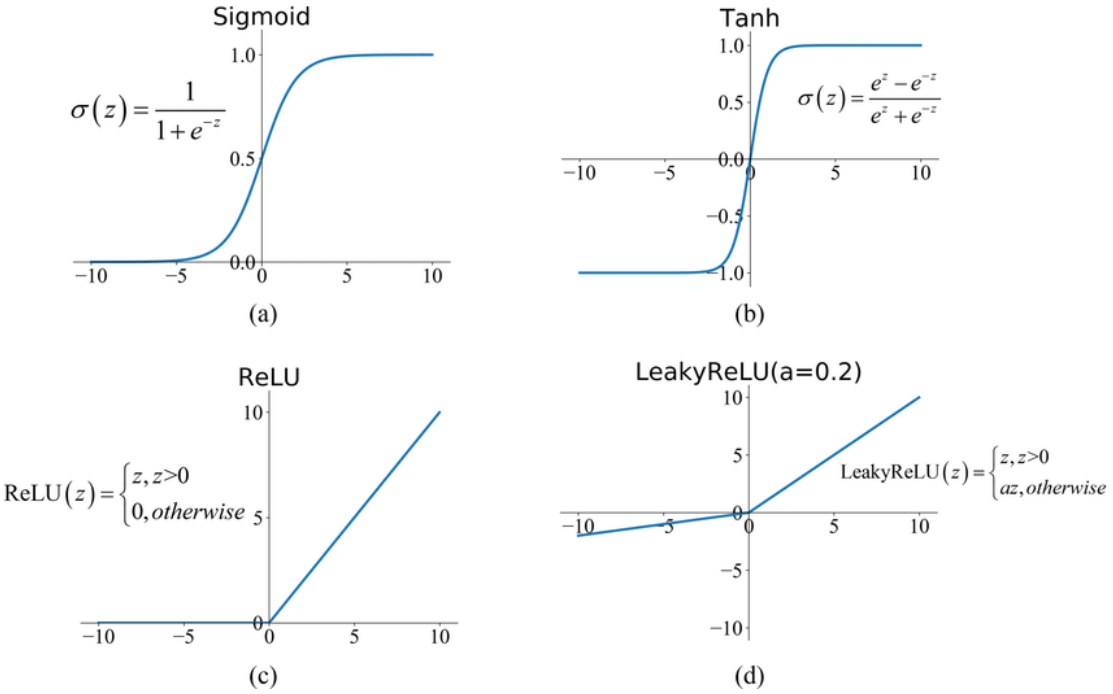

Figure 8. Non-linear activation functions used by MLP [26].

MLP training with non-linear activation functions allows the model to generalize to unseen data. The learning process is composed by two phases: *forward* and *backward*. The forward phase is when the input layers receive the data, propagating it through the model until reaching the last layer of the MLP. But before the backward phase, the MLP searches for minimizing/maximizing an **objective function**, depending on the type of problem. The objective function, also called loss function, is a measure that denotes how well the model has adjusted to a specific dataset. To accomplish a good train, the model should converge to the **global minimum** (the lowest value in a function) and avoid being trapped in l**ocal minima** (the lowest value in some vicinity). The more commonly employed approach to ascertain the minimal point is through the descent gradient. To choose the loss function, it is imperative to take into account various factors, such as the outliers or the selected *machine learning* algorithm. Due to these factors, the loss function will play a pivotal role in determining the model's performance. There is a diversity of functions, so these are categorized into two groups: classification and regression. The main difference is the annotation format, where these are finite discrete values and continuous values, respectively. Table 1 shows the different types of objective functions, grouped according to their respective categories:

Table 1. Type of objective functions, grouped by their categories [24]

Application	Loss function
Classification	0–1 loss, Perceptron loss, Logarithmic loss, Exponential loss, Sigmoid cross-entropy loss, SoftMax cross-entropy loss, Hinge loss, Ramp loss, Pinball loss, Truncated pinball loss, Rescaled hinge loss
Regression	Square loss, Absolute loss, Huber loss, Log-cosh loss, Quantile loss

Classification is a process that maps different inputs, x_i, into outputs, where each output $y \in \{1, \dots, C\}$, belongs to a predefined number of classes C. When $C = 2$ the classification is binary, while when $C > 2$ the classification is named as multi-class. In many classification problems, the objective of MLP is to find the function $\hat{f}$ that can effectively map the input data to an estimated value of $\hat{y} \in \{1, \dots, C\}$. However, in determined methods, instead of predicting a class, learning the probabilistic distribution of the input is the objective. In this case, the conditional probability density function of the output is:

$$\hat{y} = \hat{f}(x) = \underset{c \in \{1,.,C\}}{argmax}\, p(y = c|x) \tag{3.1}$$

The probabilistic classification does not predict directly and unequivocally the class, i.e., offers a probability magnitude that enables one to ascertain the level of trust in the model [24].

The *0-1 loss* function delineates that if the projected value from the sample aligns with the annotation, the loss function assumes a value of 0, otherwise, it assumes a value of 1. As would be expected, this function does not consider the error associated with the prediction, so the function graph is not convex, where the models take many shortcuts to improve the performance [24]. Unlike this one, the *logarithmic loss* function makes a probabilistic prediction, i.e., the probabilistic value is obtained by the

distribution of conditional probability. As such, the higher the match between the prediction and the label lower the measured value by the loss function [24], [28]. One of the most known objective functions for classification purposes, is the *sigmoid cross entropy*, which is very similar to the previous one. The *sigmoid cross entropy* is an objective function that computes the value through the transformation of the probability value, influenced by an activation function called *sigmoid*, which characterizes the actual output of the *cross entropy*. The cross-entropy describes the distance between the current output value and the expected output, which is lower for closer probabilistic distributions of both entities [24], [28]. Another type of *cross-entropy*; SoftMax *cross-entropy*, uses *SoftMax* as the activation function. This change allows to solve multi-class classification problems, which are the majority of the problems in *machine learning* systems [24], [28]. Regarding regression functions, the *square loss* function is one of the most applied, measuring the mean square error between the label and the predicted value. The next objective function is the *absolute loss error* and is similar to the previous one, just differing in error measure. In this context, it is obtained by the absolute difference between the current output and the expected output divided by the number of outputs. This type of function is used for linear regressions, however when the error is close to 0 the function does not smooth in 0, therefore it is not so used as *square loss function* [23], [28]. *Huber loss function* is another noticeable function which is an amalgam between the above-mentioned regression loss functions. *Hubber loss* uses δ as a boundary parameter to control which loss function should be applied. The samples that are inside of the defined border by the parameter, are subject to the *square loss function*, while in the remaining is used the *absolute loss error*. This loss function minimizes the outliers' influence, avoiding *overfitting* [23], [24].

The error is calculated based on the current weights, which were all updated during the *backward* stage with the main goal of minimizing the objective function.

Unfortunately, due to non-linearity presence, the objective function becomes non-convex, weakening the gradient as it walks deeper into the network. This disadvantage could make the model be trapped in local minima.

It should be noted that the inclusion of **regularization parameters** in loss functions aims to prevent *overfitting*. These parameters enhance the performance of the models, as well as mitigate *overfitting*. There are many regularization methods, such as L1 and L2, that incorporate a penalization term at the end of the objective functions. L1 represents the cumulative sum of the absolute values of the weights from the model:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |w_i| \tag{3.2}$$

The effect made by L1 is controlled by the λ value known as *weight decay*. This parameter intends to decrease the error as much as possible. The λ can influence the model's learning, so the values must be carefully selected, as a high value can lead the model towards *underfitting*. Usually, L1 is used when there is a high value of features. L2 parameter is one of the most popular which is the sum of squared magnitude coefficients:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2 \tag{3.3}$$

L2 can influence the prediction system because penalizes insignificant predictions. These regularization methods aren't the only methods of *overfitting* attenuation. The *dropout*, unlike the regularization, does not add penalization. Instead, it randomly interrupts the operation of certain neurons during the forward phase. Through *dropout*, the model is forced to have diversity [26],[27].

MLP is a non-convex function of its parameters, so to reach a local optimal it resorts to a standard gradient-based optimization process. It is usual to apply first-order online methods, such as Stochastic Gradient Decent (SGD), or second-order offline methods, like Hessian. So, before the optimization process, the gradients are calculated,

leading to the backpropagation. The **backpropagation** begins by receiving the loss, which is a scalar value, measures the partial derivative of the objective function regarding the network output. The gradient applied during the backpropagation phase is a partial derivative. The function that describes the backpropagation is (3.4):

$$\nabla(\mathcal{L}, \hat{y}) = \frac{\partial \mathcal{L}}{\partial \hat{y}} = \frac{\partial \mathcal{L}}{\partial o} . \frac{\partial o}{\partial \hat{y}} \tag{3.4}$$

Where $\mathcal{L}$ is the loss function, $\hat{y}$ is the label and *o* are the predicted value by the model. The gradient computation could be described in 3 essential steps:

- It first performs a forward path, where the different parameters are computerized, such as the pre-synaptic, a_n (input signal), and post-synaptic, z_n (input signal translated through transfer functions, like *sigmoid*, threshold, and *Gaussian*) of the hidden layers, the pre-synaptic, b_n (Weight application on the post-synaptic hidden layer), and post-synaptic, $\hat{y}_n$ (non-linear application on the b_n) of the output layers;
- It is computed the error for the output layer, $\delta_n^{(1)}$, which is passed backwards through the weights, so it can be calculated the error from the hidden layers, $\delta_n^{(2)}$;
- Finally, the overall gradient is calculated by Equation (3.5) [23], [24].

$$\nabla_\theta \mathcal{J}(\theta) = \sum_n [\delta_n^v x_n, \delta_n^w z_n] \tag{3.5}$$

As a quote, the first point noted is referent to the forward process, but due to its importance for the gradient it is again highlighted.

3.3 Optimization process

Optimization is a critical element for the correct performance of the different *machine learning* systems. The main directive of the networks is to find a subjacent function, with its foundation in the data that encapsulates the problem. To do so, it is necessary to discover a significant number of functions capable of describing the final function and then find the ideal parameters, using the objective function. The third step involves utilizing the discovered function from the preceding stage and performing predictions on unseen data. These three steps are also known as the representation, optimization and generalization phases where, usually, each of these is analyzed individually [29].

The optimizer to achieve the global minima, will update these parameters in the opposite direction of the gradient of the objective function, taking into account the *learning rate*, η, which determines the step size to reach the minimum. The optimizer will go after the valleys made by the objective function until reaches the *plateau*. There are three variants of the gradient descendent, that differ in the quantity of computerized data in the objective function, being necessary to make a *trade-off* between the parameters' *accuracy* and the time taken to update. The one that has the higher impact is the SGD, due to its agility to deal with noisy data and outliers, where the parameters update is made in each sample $x^{(i)}$ and annotation $y^{(i)}$. **SGD** eliminates redundancy making the update one at a time, providing much faster training, and making it possible to do it online. Since the SGD updates the parameters with high variance, it triggers a fluctuation in the objective function. Due to this point, the fluctuation made by the SGD allows it to jump to better optimal local minima. Despite its various advantages, also have some challenges, such as selecting the correct *learning rate*, to avoid slow convergence in small values or lead to the algorithm divergence in extreme situations with high learning rate. Usually, in the *deep learning* context, *learning rate* adjusted over training, i.e., during the training the η is reduced, allowing to take smaller strides towards the attainment of the local minima. This is done by setting an a priori *learning rate* decay factor. One of the problems associated with SGD

is being stuck in numerous optimal local minima. To overcome such adversities, some techniques have been proposed. The SGD gradient has orientation difficulties when faced with multiple minimal values, a circumstance that frequently arises in proximity to the optimal point. In these situations, it relies in *momentum,* which is a method that can accelerate the progress of the SGD (faster convergence), as well as dampen the oscillations. In a simplified way, the *momentum* is a criterion taking into account for parameters updates that should be maximized as much as possible. To do so, the *momentum* increases for dimensions that have gradients pointing in the same direction, while reduces updates for dimensions whose gradients change directions. Although the *momentum* gives these advantages, it cannot provide knowledge about which direction to go because the *learning rate* is not adaptative, it is previously set [24], [23], [30]. Figure 9 illustrates the SGD technique with and without *momentum*:

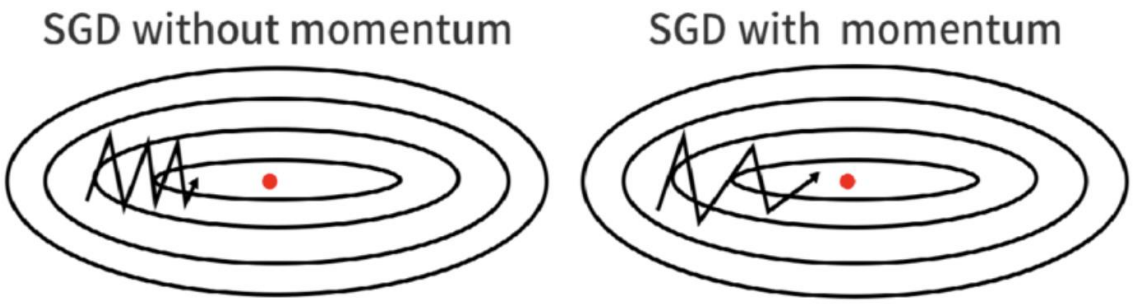

Figure 9. Schematic of SGD with and without momentum [31].

As such, it resorts to methods that can adapt the *learning rate*, such as the ***Adaptive Moment Estimation*** (Adam). This method can compute adaptative *learning rates* for each parameter. The Adam stores the exponentially decaying average of previous squared gradients, which is the uncentered variance (quantifies how much the gradients vary around their means) and the exponentially decaying average of antecedent gradients. The iterative Adam estimate is given by Equations (3.6) and (3.7):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{3.6}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{3.7}$$

where m_t and v_t are estimates for the mean and uncentered variance respectively, and g_t is the square root value of the error estimate. The Adam authors realized that at the beginning of the training the m_t and v_t were initialized as vectors of 0's and were biased towards 0. To avoid this situation, they defined default values β_1 and β_2 of 0.9 and 0.999, respectively. At this point we have a bias estimator, so in order to correct this first and second order estimates are computed.

$$\widehat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{3.8}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{3.9}$$

The authors of *Adam* realized that the optimizer was biased towards zero [23], [24], [31].

3.4 Deep Learning

Before the *deep learning* systems appeared, the feature extraction was made through *feature engineering* techniques, which are *feature* extractors that provide information from the data to *machine learning* systems. The advantage of *deep learning* is that the features are obtained in the training process in the form of filter coefficients which minimize the error output hence they are optimal in the error sense. For being effective more complex architectures employing novel learning mechanisms are required in the deep learning approach. Following sub-sections introduce the basic concept of *deep learning* and the *Convolutional Neural Network* (CNN). *Deep learning* allows the models

to learn the data representation through multiple extraction layers. Due to this property, it was possible to improve speech and object recognition [30].

3.4.1 Convolutional Neural Network (CNN)

This type of neural network has the role of performing data processing, such as 1D data, signals that vary across time, or 2D, which are images made by pixels, or even 3D, which are images with a third representation made by voxels. CNNs have been more applied in image context, where the input data are an array with three dimensions, such as number of channels, height, and width. The height and the width could extract information about the spatial dimensions of the data, while the number of channels refers to the independent properties across the spatial space. This last one, in image context, confers color to data, namely red, green, and blue (RGB channels). Each convolutional layer is organized in the following form $k_H \times k_W \times k_{ch}$, where k_H denotes the vertical dimension, k_W signifies the horizontal dimension and k_{ch} the number of input channels of kernel k. Usually, the kernel has a smaller dimension than the data. The kernel is responsible for applying a convolutional operation, which involves placing it at every possible position on the data. The kernel, for each spatial overlap with data calculates the dot product between it and the corresponding spatial domain of the input data as shown by Figure 10.

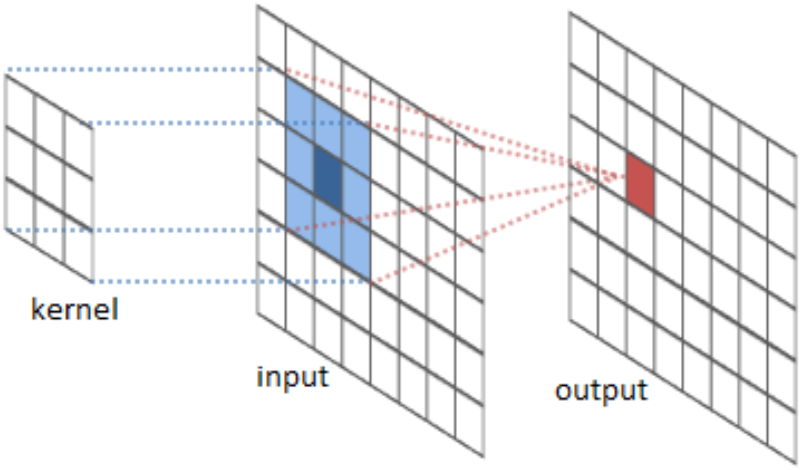

Figure 10. Convolution operation between the input data and the kernel [32].

The output will be a matrix, where each value will be the result from the dot product operation. As expected, the output dimensions will be smaller than the input, because only the regions with total overlap with the kernel undergo the scalar product. One of the challenges is the loss of information at the periphery of the image and to avoid this issue padding with the same dimensions as the kernel is usually applied. There are numerous kernels in each convolutional layer, where each one makes its own spatial output, referred to as a *feature map*. Each filter has as its main goal to identify patterns, so the model will have a higher ability to detect a large number of patterns, however, an exacerbated number of kernels brings in redundant *features*, guiding the model to *overfitting*. Typically, a convolutional network has three stages. In the first one, the convolutional layers make a set of linear activations where, in the second phase, each one of these crosses a non-linear function, being so named as the detection stage. Finally, with the directive to change the output for the next layers, it resorts to a *pooling* operation. The *pooling* layer is applied in each *feature map*, keeping up the number of *feature maps*. This operation doesn't increase the number of training parameters, nor decrease the computational cost. The *pooling* is a square network ($p_h \times p_w$) that will affect the dimension reduction of the input, being the factor reduction of p_h. The spatial dimension reduction allows the next convolutional layers to find patterns in distant regions, increasing the receptive field. Like the convolutional operations, the *pooling* is translation invariant because a slight change in input does not trigger a significative alteration in the output. The most popular *pooling* layer is the *Max-Pooling* which returns the maximum value of the network, analogous to the Figure 11 representation:

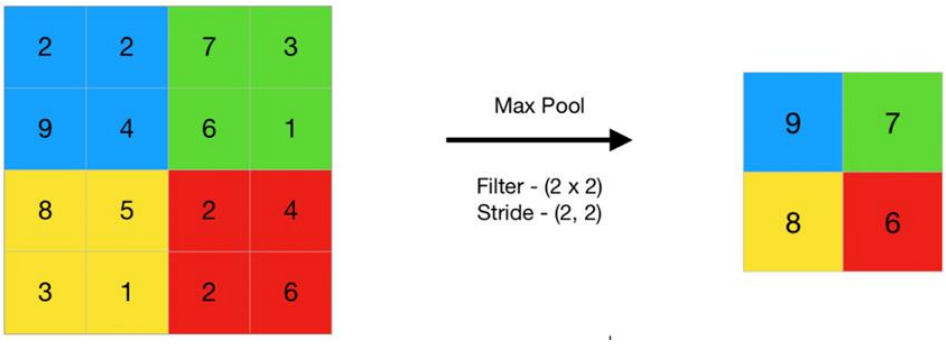

Figure 11. *Max-pooling* operation [26].

Max-Pooling is used in partnership with a convolutional layer, but there are also other spatial aggregation operations, like *Average-Pooling* and *Global-Pooling*. These aren't as used as *Max-Pooling* because this originates high non-linearity [23], [24].

3.4.2 Advantages of CNN

CNNs possess three main properties that make them a high standard to computer vision: translation invariant, parameters share, and a smaller quantity of *sparse connectivity* (each neuron is connected to a limited number of neurons). In a convolutional layer, the same kernel convolves with the input data, so the found patterns will be independently of spatial location within an image. This property is extremely important when dealing with imaging data. However, transformations like rotations could make different *feature maps*. To generate the feature map, the kernel parameters are used for the whole image. Subsequently, each feature map point is operated on by the rest of the parameters. This property avoids the necessity of many parameters to do *features* extraction. Finally, the *sparse connectivity* fights the scaling of the parameter, which is inconvenient for the fully connected layers, where all nodes are connected to each other. The CNN does not have total connectivity with the input data, just establishes connectivity with a small region, controlled by the kernel size [24], [26].

The advantages highlighted indicate that it is beneficial to apply *deep learning* models based on CNN because CNN had proven to be exceptionally effective in various tasks, particularly in image applications. Their ability to automatically learn and extract features from the data makes them a powerful choice.

Chapter 4 State-of-the-Art of ML Based Methods for Assisted Surgery

Digital image processing has become a crucial tool for doctors because it allows them to filter and enhance relevant information. It is essential to study all existing techniques in this area, with a focus on systems that can improve relevant characteristics in laparoscopic images. This chapter discusses three types of tasks that can be enhanced using ML: classification, detection, and segmentation. Classification involves categorizing objects in an image, detection involves identifying the presence of an object, and segmentation involves separating objects from the background. Understanding these techniques can help doctors getting deep insights from medical images and improving patient outcomes.

4.1 Classification Methods

The classification approaches are mainly focused on surgical tools. This is a challenging task in surgical endoscopic videos but has added relevance assisting doctors during robotic surgery. The classification process in *machine learning* systems consists of predicting the present class in an image. Classification approaches can be of two types: binary classification where the purpose is to infer the presence or absence of an object, or multi-class classification, which is the best approach when multiple surgical instruments are present.

The first approaches to surgical instrument classification involved the use of classical feature-engineering *machine learning* algorithms. Liu et. Al [33] proposed a method to monitor surgical tools during surgery or robotics' applications. This system consists of extracting features that are used as input of a classifier. The first phase allows the classifier to recognize the surgical tools present on an input image. The authors proposed an analysis based on color images, through *hue-saturation-value* (HSV) color space, which is robust to luminosity intensity. In HSV space, *hue* refers to the color

property, varying between 0º to 360º, *saturation* is the color purity, fitted 0 to 1, and *value* describes the pixel intensity. From H and S histogram distributions, a threshold was defined to segment the image. Then, Fourier descriptors are extracted, describing the form and contour characteristics and presenting some transformations (rotation and translation) and noise invariance. Later, all the extracted characteristics were utilized as the input for the Naïve Bayes classifier, which is based on the Bayes' theorem of maximum a posteriori estimation. The data presented in this paper was gathered under static conditions, where the surgical instruments were positioned at a fixed distance on the operating table with the camera.

In 2015, Primus et. Al [34] performed a task for the classification of six surgical instruments, namely Long grasper (LG), Dissector forceps (CDF), Scissor (S), Clipping device (CD), Monopolar spatula (MS) and Exhauster (E). The performance of the classifiers was evaluated following the BoW-SVM architecture. This structure comprises a frameset bifurcated into distinct training and test sets. The visual features within each frame are grouped by k-means *clustering* method, where the centroids (mean for each group) construct a vocabulary. Features of each image are mapped to the vocabulary and the histogram of visual words based on the vocabulary, are used as an image representation. These histograms are used to train the SVM classifier. The developed work compares three extraction features techniques: *Oriented FAST* (ORB), *scale-invariant feature transform* (SIFT) and *speeded-up robust feature* (SURF). The primal of these three is to find in real time the key points with correspondence to visual features. For the same architecture, the three techniques were evaluated in terms of *precision* (true positives divided by all positive predictions), *recall* (true positives divided by all positive cases in the dataset), and *accuracy* (all correct identified cases divided by all prediction realized) for each surgical laparoscopy tool. The reported values are shown in Table 2:

Table 2. Metric values of precision, recall and accuracy for each one of features extractors to the six instruments

Classes	Models	Precision	Recall	Accuracy
LG	ORB	0.65	0.31	0.75
	SURF	0.48	0.47	0.73
	SIFT	0.32	-	0.71
CDF	ORB	0.45	0.23	0.73
	SURF	-	-	0.82
	SIFT	-	-	0.83
S	ORB	0.29	0.05	0.93
	SURF	0.1	0.03	0.91
	SIFT	0.18	0.02	0.9
CD	ORB	0.7	0.25	0.89
	SURF	0.2	0.15	0.78
	SIFT	0.38	0.2	0.67
MS	ORB	0.7	0.39	0.83
	SURF	0.36	0.15	0.8
	SIFT	0.1	0.02	0.88
E	ORB	0.61	0.09	0.9
	SURF	0.3	0.2	0.91
	SIFT	0.22	0.53	0.61

Based on these values, the authors have deduced that the amalgamation of ORB keypoints and SVM classifier with BoW exhibits high accuracy across all six surgical instruments.

Later in 2017, Petscharnig et. al [35] studied CNN behavior for gynecology organs classification and surgical workflow using laparoscopic videos. However, the second task is out of the scope of this study and will therefore not be addressed. Three possibilities were proposed for classification: the first approach applied AlexNet. The additional suggestion was GoogleNet, which introduces a module known as *inception*, that avoids *overfitting* through dimension reduction. Like AlexNet, this network ends in a fully

connected layer. Both models were trained from scratch starting from random weights in the first iteration. The third and last concept applied was an SVM. In this model features were from an AlexNet network that was pretrained on the large ImageNet dataset [36]. This concept of getting knowledge from one dataset to another is known as *transfer learning* and is useful in applications with scarce data. The authors studied extracting features from three different depths of fully connected layers as inputs of SVM: SVM connected to the third layer of AlexNet (SVM_Class), SVM associated with *fc7* (SVM_fc7), and SVM linkage to *fc6* (SVM_fc6). Each network was trained on an annotated dataset, composed of five organs: colon, liver, ovaries, oviduct, and utero. After the training, they were tested to annotate the metrics of *precision*, *recall,* and *F-value* (provides better insights under imbalanced datasets). In Table 3, the recorded metrics values attained by every model:

Table 3. Metric values, recall, precision and F-value obtained by the 5 networks for each organ

Metrics	**Networks**	Colon	Liver	Ovaries	Oviduct	Utero
Recall	AlexNet	0.652	0.596	0.858	0.442	0.528
	GoogLeNet	**0.795**	0.862	**0.888**	**0.623**	0.743
	SVM_Class	0.554	0.601	0.484	0.562	0.581
	SVM_fc7	0.663	**0.891**	0.755	0.374	**0.801**
	SVM_fc6	0.572	0.854	0.712	0.412	0.697
Precision	AlexNet	0.595	0.765	0.546	0.8	0.613
	GoogLeNet	**0.805**	0.896	**0.747**	0.839	**0.619**
	SVM_Class	0.461	0.659	0.591	0.86	0.273
	SVM_fc7	0.792	**0.927**	0.561	0.862	0.475
	SVM_fc6	0.751	0.882	0.535	**0.874**	0.408
F-value	AlexNet	0.622	0.67	0.667	0.569	0.568
	GoogLeNet	**0.8**	0.879	**0.811**	**0.715**	**0.676**
	SVM_Class	0.503	0.629	0.532	0.68	0.372
	SVM_fc7	0.722	**0.909**	0.644	0.522	0.596
	SVM_fc6	0.649	0.868	0.611	0.56	0.514

Overall, the GoogleNet model exhibits superior metrics in the classification of gynecology images.

More recent works have focused on predicting surgical tools by using CNNs. A research group Jaafari et. al [37], selected three CNNs, VGG19, Inception v-4, and NASNet-A to study the impact of *ensemble learning* on the classification of surgical tools. To accomplish this, they have trained all three models as well as an additional one, with the final model being a fusion of the three, *ensemble learning* method. The VGG19 is a neural network that demonstrates that augmenting the layers of a CNN enhances the accuracy of classification. The Inception v-4 owns similar blocks to GoogleNet, with a new block that permits a boost in training, *batch normalization*. Contrarily to the other models, NASNet-A applies a distinctive learning method, *Reinforcement Learning*. This learning system consists in the search for the best parameters of filters dimensions, strides sizes, number of layers, among others. The model doesn't have a fixed number of blocks or cellules, i.e. the quantity of iterations for the convolution layers and the quantity of filters are parameters that can be freely adjusted. In this network, what raises more attention is the presence of *Normal Cell* and *Reduction Cell*, which are convolution blocks that return features maps with the same dimensions and produces the same map with a size reduction by a factor of 2, respectively. In an ensemble learning system voting is the most used approach to produce an answer and consists of selecting the highest prediction value among all models. The networks were trained in a dataset with annotated surgical tools: grasper, bipolar, hook, clipper, irrigator, specimen bag and scissors. It is relevant to mention that all models undergo a fine-tuning process through transfer learning with the ImageNet dataset [36]. All models are subject an analyze of their *precision* and *Receiver Operating Characteristic* (ROC), which is the performance of the model between the sensibility (ability to correctly identify a class, true positives) and the specificity (capability to no identify a class when is not represented, true negatives). Table 4 shows that the ensemble learning has the best performance to classification task, achieving more than 94% in all classes for both metrics.

Table 4. Precision and ROC metrics of all models

Metrics	Classes	*VGG19*	*Inception v-4*	*NasNet-A*	*Ensemble Learning*
Precision	Grasper	0.977	0.9367	0.9854	**0.9951**
	Bipolar	**0.979**	0.9654	0.9732	0.977
	Hook	0.967	0.9433	0.9711	**0.9814**
	Clipper	0.998	0.997	0.9989	**0.9991**
	Irrigator	0.961	0.9208	0.9591	**0.9779**
	Specimen Bag	0.952	0.9394	0.9635	**0.9729**
	Scissors	0.876	0.8084	0.9006	**0.9454**
ROC	Grasper	0.998	0.9937	0.9989	**0.9995**
	Bipolar	0.965	0.9516	0.9631	**0.9688**
	Hook	0.997	0.9917	0.9953	**0.9981**
	Clipper	0.999	0.9979	0.9991	**0.9993**
	Irrigator	0.994	0.9874	0.9941	**0.9985**
	Specimen Bag	0.996	0.9939	0.9956	**0.9981**
	Scissors	0.988	0.9811	0.9918	**0.9964**

The classification accomplishment, without the localization of the object, does not add much information on a surgical process.

4.2 Detection Methods

Object detection in a laparoscopic image permit locating a surgical instrument or an organ and know the approximated occupied area, represented by a *bounding box*. This knowledge may augment the proficiency of surgeons in their manipulation of the equipment, as well as to develop a fully automated surgery, without human intervention. The pioneers in this section were Lee et. al [38] that searched for an algorithm with the ability to detect surgical tools. For that, this research group developed an image processing system divided into 4 steps. In the initial stage, named as the classification phase, different color signatures of the organs and surgical instruments with the goal of individual pixel classification are used. They implemented binary Bayesian classifier (instrument VS organ), which maximizes the predicted a

posteriori probability. Then, a unique *label* in each region with an instrument is assigned. The pixel identity is determined in two stages: the temporal phase where the previous frames' annotations are propagated to the next frame, and the other phase which is annotating the surgical tools pixels not *labeled* through pixel adjacency. In this third stage, each tool was allocated inside a *bounding box*, a rectangular or trapeze form. The laparoscopy images used by the authors do not possess any type of *ground truth*, so none algorithm analyze had been performed. The only criteria used by the authors was visual. According to them, their qualitative results were good and promising.

Subsequent endeavors were fashioned to assist the surgeon in discerning the apex of a surgical instrument. Due to camera instability, this could result in a wrong interpretation by the surgeon, so Climent et. al [39] developed an algorithm where they used the *Hough* transform as a mean to highlight the surgical tool from the background. In this case, to extract the best properties of the Hough transform, it is necessary, according to the authors, work with a filtered image, so they proposed to apply a Gaussian filter to attenuate the noise. After, as a requirement for such transform, the orientations of borders were extracted through a gradient operator. Lastly, the application of the *Hough* transform. The advantage lies in locating pixels that need not necessarily be contiguous to one another, however, the *Hough* transform could be misleading when the object is lined up with the boundary of the image. A solution to this preceding issue is suggested, wherein the ultimate instrument would be the discontinuity in the loss direction of the gradient along the linear trajectory. The pixels across the straight line are tracked until these orientations present an expressive change relative to the axis of the straight line. Each line pixel was calculated by the difference between the gradient direction and the straight line. In numerous instances, the elongated lines do not serve as surgical instruments, thus it became imperative to extract additional information using heuristic filters and movement data. The position prediction was performed knowing the previous and next frame position, where a first-order model was implemented, that follows Equations (4.1) and (4.2):

$$v_k = x_k - x_{k-1} \tag{4.1}$$

$$\hat{x}_{k+1} = x_k + v_k T \tag{4.2}$$

Where v_k stands for speed of the instrument, x_k and x_{k-1} are two consecutive positions, $\hat{x}_{k+1}$ is the predicted position, and T is the time gap between the actual and the last frame.

This adopted technique allowed the system to work without any color restriction or special marks present on instruments. Despite these advantages, the performance was not very good when the instrument does not possess any contrast with the background or when the tool vanishes from the visible area.

In an effort to ascertain the whereabouts of surgical instruments, Haase et. al [40] proposed a combination of color information and a score system. With the goal of decreasing the computational cost and increasing the system robustness, the authors offered the regions of interest, i.e., regions that are more likely to contain a surgical tool. As a precondition, the insertion of the surgical instrument is carried out along the periphery of the image. Therefore, to be considered a region of interest, the proximity between the instrument and the sensor must be minimal. Consequently, the point values are diminished. Another possibility was saturation occurrences, low intensities mean colorless pixels which indicate the surgical tool's presence. Once they found all points of interest, false positives were discarded with these hypotheses: the neighbor point should have similar intensities and an intensity value smaller than the intensity saturation value from the total image. The *Sobel* filter was applied to highlight the borders. After, the image gradients from the region were transformed to *Hough* space in order to detect marked lines in polar coordinates. Hough transform application in surgical instruments induced two peaks in his space, which were the two lines of localization that delineate the axel borders. As expected, this transformation could generate erroneous contenders. Therefore, an analysis of the angle between the candidates and the line was proposed. Finally, in terms of detection, it was assumed that the instrument apex extends beyond the center line, particularly within the expansive gradient zone. This zone serves as the intermediary region between the tool and the

background. The reported scores were promising however, carry some limitations namely if the field vision has not any value all candidate points are rejected. Another constraint lies in the presence of blood, as it augments the saturation, thereby hindering the algorithm's ability to discern the initial valid points.

In 2017 the first *deep learning* models appeared, as well as annotated datasets that could be used for metric analysis or model comparison. For surgical tools detection, Choi et. al [41] proposed a CNN, based on *You Only Look Once* (YOLO), that can be a model detection candidate. This network differs from YOLO due to the presence of two fully connected layers instead of one, the *dropout* layers between those layers and batch normalization. These changes have the main goal of avoiding *overfitting* and augmenting performance. Another research team, Jin et. Al [42] proposed another neural network based on *Region Based Convolutional Neural Network* (R-CNN), which is the Faster R-CNN. The network was pre-trained on the ImageNet dataset, to learn some visual generic features. During the model training, the search of regions of interest is done in *Region Proposal Network* (RPN). The positive notations, indicating the presence of an object were established when the Intersection over Union (IoU) value exceed or equal 0.8. Contrarily, the negative label required a value lower than or equal to 0.3. The YOLO adaptation and the Faster R-CNN underwent training using the same dataset, *m2cai16-tool-location*. The expected output is drawing a *bounding box* over the objects, so the best metric for this kind of situation is the *mean Average Precision* (mAP) that relies on the IoU between the predicted and the ground-truth. Table 5 shows the obtained values for each model for the seven classes.

Table 5. AP values for each class from the dataset

Autor	Grasper	Bipolar	Hook	Scissors	Clipper	Irrigator	Specimen -bag	mAP
Choi[41]	**0.893**	0.324	0.93	0.666	**0.903**	0.424	**0.914**	0.7226
Jin[42]	0.872	**0.751**	**0.95**	**0.708**	0.884	**0.735**	0.821	**0.818**

Another *deep learning* application for the detection of surgical instruments, used a fully connected network (FCN). In medical imaging, such networks are expert at detecting malignant regions within images. An example is the EndoNet, which is a *deep learning* model comprising convolutional layers to extract features in a highly proficient manner. One of the issues with such networks is their incapacity of learning well the dataset which can be alleviated by using a pre-trained network. The *weakly supervised learning* networks, which is the network in question, were the first capable of detecting surgical instruments. These types of models are compounds by convolutional layers that generate in total 512 features maps. In order to acquire the semantic map for every class and preserve all spatial information within the network, EndoNet implemented a *Spatial pooling* mechanism in conjunction with FCN. Through this combination, it could transform a feature map into a vector with confidence values, providing a binary classification for each tool. Although it has achieved a high accuracy, with predictions around 90% in each category, it can only detect objects. Therefore, searching for other approaches, preferentially with the ability to perform the three tasks, segmentation, detection and classification seems to make some sense [43].

4.3 Segmentation Methods

Image segmentation entails the process of grouping pixels according to their respective classes. There are three types of segmentation: semantic, instance, and panoptic. Semantic segmentation entails the process of associating pixels with a specific class. Instance segmentation is identical to semantic segmentation, however, can distinguish types of entities within each class. Finally, panoptic segmentation is the combination of the other types of segmentation, semantic and instance. This type of segmentation discerns *stuff* from *things*, i.e., pixels from the background are semantically segmented, while the pixels classified as *things* are segmented as instances since they are entities with more interest.

The first works tried using pre-existing or during surgery knowledge, from different sources, such as sensors and models of images. The first target was the

extraction of information of color, as it represents an inherent characteristic of an image. Since all images could possess brightness or shadow points, researchers looked for different techniques. The RGB was initially used for surgical tools detection, however, due to a lack of information present in these three channels, it was only used as a complement of HSV and *hue-saturation-lightness* (HSL) scales. These two are alternative representations of RGB that approximate to human visual perception, allowing discern chromatics intensities from luminosity intensities. The dispersion of luminosity among the other components could potentially enable access to a greater wealth of information. As expected, using all these channels as input results of more than 8 bits per channel, increasing the processing time. To understand which color space could give more discrimination between the surgical tools and the anatomic parts was imperative to apply *Bhattacharyya* distance, which is a metric that measures the similarity of two probability distributions. Another alternative is the importance metric coming from the *Random Forest*. This consists of training the *machine learning* model so the model can select the bests *features*, always considering a variable importance [44]. Despite color is one of the dominant strategies for *feature* extraction, the light used in surgical environments, combined with soft tissues, could cause a large spectral reflection, corrupting the white and the gray present in metallic instruments. This constraint led authors to abandon this option and move on to another [45].

The second possibility was based on gradients. Usually, gradients are generated from intensity values of images. They can be acquired from image derivation in both axis, x, and y, using a *Sobel* filter or *Canny* filter. However, as one would anticipate, extracting pertinent information proves to be quite arduous, thereby needing supplementary techniques. The other helpful techniques could be *Hough* transform or *Histogram of Oriented Gradients* (HOG). Although it is a method that allows border detection and instrument orientations, it is not very applied because the number of orientation positions necessary for object detection in instances matches a 6, while the HOG resorts an extra channel for magnitude, turning the real-time processing not possible [45].

The HOG approach is not doable, due to high latency, so a research team proposed an alternative linked to gradients, such as texture. The texture of an image is characterized by the periodic recurrence of local patterns, which enable the acquisition of more resilient *features*. Initially, the image texture was extracted applying a filter, such as *Gabor* filter or through texture descriptors, like *Local Binary Patterns* (LBP). Another solution that was widely used in extracting points of interest for object detection was SIFT. The SIFT technique is an adopted approach for extracting *features* that remain invariant to changes in dimensions or orientation, displaying remarkable resilience to variations in lighting, noise, and partial occlusion [46], [45]. *Haralick features* were proposed to improve texture description, however processing in grayscale and applying 14 equations in 4 angular levels, resulting in 4 values for each equation, becoming undoable for real-time surgical tools detection [47].

Due to difficulties found, another research team looked for new alternatives based on shape. Most of instrument detectors have as input parameters the *features* dimension, which is commonly denoted by a series of numerical values to define a dimension. In this area, different approaches were followed: *region moment*, *Wavelets transforms,* and *transform-domain shape*. The first is used to disjoint the instruments from the background, applying the *Otsu Threshold*. Although it presents visible results, it is not robust to noise, occlusion, and non-rigid deformations. The *Wavelets transforms*, allow the decomposition of the information based on the gradient direction, with borders associated with higher frequency content. However, such as the previous, it is not robust to noise. Finally, the *transform-domain shape*, associated with Fourier descriptors enables the depiction of the border through the computation of Fourier components for every pixel along its perimeter. An advantage of these descriptors is being invariant to rotation and translation and, its combination with the color classifiers could obtain an optimal border region [48]. As expected, the transform-domain shape possesses disadvantages like the computation cost, due to the calculation of Euclidian distance for each pixel present in interesting region [45].

All these different methods of *features* extraction would be used to train different kinds of *machine learning* models, such as SVM [45].

Posteriors works applied *machine learning* systems, such as Bouget et. al [49]. The authors proposed a binary semantic segmentation, named *SquaresChnFtrs*. This approach is a decision tree classifier where the focus is on *features* channels. The input channels for the classifier should have the gradient, color, texture, and position information. The dataset applied was the '*The NeuroSurgicalTool Dataset*'. In this case, research team analyzed precision for semantic classification pixel by pixel, obtaining 0.858. Despite the high precision value, the semantic map is highly distorted, particularly along the borders.

García-Peraza-Herrera et. al [50], proposed an automatic segmentation method based on FCN-8s of non-rigid surgical tools. They simply changed the amount of scoring layers to two, as their primary objective was binary segmentation. In addition to FCN, they implemented an optical flow system that utilized the predictions generated by FCN-8s to enhance the efficacy of tool segmentation. The model was pre-trained in 'PASCAL-context 59-class' and the train and test were made in '*MICCAI 2015 Endoscopic Vision Challen–e - Instrument Segmentation and Tracking Sub-challenge'* dataset. Once again the adopted metric was *accuracy*, having achieved 0.883, in binary segmentation. The authors also did multi-class semantic segmentation, being composed by background, shaft and manipulator instrument parts, and scored 0.837.

In laparoscopy images, a research team from the Institute for Intelligent Systems Research and Innovation (IISRI), Attia et al. [51] proposed a model that combines CNN with Recurrent Neural Network (RNN) to do semantic segmentation. The RNN, which is the *Long Short Term Memory* (LSTM), was applied to preserve the local and global contextual dependencies and improve the quality of the generated masks by CNN. LSTM can learn the special dependencies between the neighbor pixels and as expected, could possess four distinct frames to analyze an image (up to bottom, bottom to up, left to right, and right to left). The proposed architecture had an *encoder* phase to extract *features*, containing 7 convolutional layers, two MaxPooling, and 4 LSTMs. The second stage, which was the *decoder* phase, had a deconvolution layer to produce the semantic mask. The training dataset is from "*MICCAI 2016 challenge "EndoVis"*, getting an average IoU of 0.827.

Another system, only with CNN, was developed by Ni et. al [52]. The research group suggested the RASNet for semantic segmentation of surgical tools. This model was structurally similar to the previous one, with an *encoder* and a *decoder*. RASNet had the ResNet-50 as an *encoder*, which was pre-trained in ImageNet dataset. In the decoder stage, the authors introduced the concept of attention blocks, where the directive was to extract global *features* from high levels that had information about the studied object. The dataset was the "*MICCAI 2017 challenge EndoVis*", where the achieved mean IoU was 0.9033. The value is ambiguous because criterion for the dataset split into train and test sets did not avoid the presence of similar frames in both sets since it was not assured that the training and test sets have different patients.

Another team applied a network based on U-Net, named PaI-Net. The research team, Wang et. Al [53] made a model composed by three stages: *encoder-decoder*, attention blocks parallels, and fusion block as output. The system *encoder-decoder* shared the same ideology as a U-Net incorporating *skip-connections*. The attention blocks consist of consecutive convolutions with distinct kernels, interconnected in a parallel fashion to each convolutional layer. The fusion models in the output were made to obtain different *features* from the various levels of the *decoder*, that have many degrees of semantic information. The application of these techniques increased the *precision* metric. The applied dataset was the same from the RASNet, which was "*MICCAI 2017 challenge EndoVis*". The performance accomplished by the PAI-Net was 0.33 IoU, doing a semantic segmentation of three instruments parts, wrist, shaft and cappers.

In theory, a better solution should do the three main tasks at once, classification, detection, and segmentation. Two models emerge with the ability to segment and classify, namely TernausNet and ToolNet Holistically-Nested, being both based on U-Net [54]. The TernausNet is a U-Net with a pre-trained VGG-11 as the *encoder* and the remaining architecture was the same as the U-Net [55]. The ToolNet Holistically-Nested embraces the *encoder-decoder* paradigm, however, in contrast to TernausNet, it underwent certain modifications in the *decoder* domain. In the *decoder*, each convolutional layer merged into a unified whole, and a deconvolution block was

employed to ensure that the input and output shared identical resolutions [56]. To know which one had the best performance, the average IoU was obtained for each model on the same dataset. TernausNet achieved 0.542 for instrument type segmentation, while ToolNet Holistically-Nested had only 0.337 [54].

In the field of computer vision, multi-task learning refers to a method where a model can perform multiple tasks simultaneously. This approach allows the model to handle tasks such as classification, detection, and segmentation in a single framework. One notable example of a multi-task method is Mask R-CNN, which stands for Mask Region-based Convolutional Neural Network [57]. Mask R-CNN is a network architecture that consists of three output heads: one for classification, one for object detection, and one for instance segmentation. The classification head is responsible for categorizing objects into different classes, such as identifying whether an object is a car, a person, or a tree. The detection head focuses on localizing and identifying objects within an image, providing information about their bounding boxes. Lastly, the instance segmentation head aims to segment each instance of an object, assigning a unique mask to distinguish it from other objects.

To train the Mask R-CNN model, a multi-task loss function is employed. This loss function considers the contributions from each of the three heads, allowing the model to learn and improve its performance across all tasks simultaneously.

While Mask R-CNN has been widely used for instance segmentation tasks, it may not always be necessary to employ this level of segmentation. In cases where there is no overlap of objects or instruments of interest, instance segmentation might be unnecessary and can be disregarded. This decision is often made based on the specific problem being studied and the requirements of the task at hand.

4.4 Conclusion

Over the years, computer vision in robotic-assisted surgery has undergone significant exploration. Initially, researchers attempted to classify the various types of surgical tools that appeared in the physicians' vision field. However, it was observed that

this classification approach did not significantly enhance the surgeons' knowledge or aid in their performance. Consequently, the scientific community began exploring alternative methods to detect these instruments, as providing a *bounding box* around them could offer a better understanding of their location. While instrument detection is crucial, the subsequent focus shifted towards instrument segmentation, which presents the crux of the problem.

In the early stages of development, algorithms without learnable parameters were employed for classification, detection and segmentation. This was followed by a combination of image processing techniques with *machine learning* models. Eventually, the field progressed to utilizing deep learning models exclusively.

The remarkable results achieved by deep learning models have led to an increased adoption of their approaches in this domain. By leveraging the power of *deep learning*, significant advancements have been made in accurately detecting and segmenting surgical instruments, thereby enhancing the overall effectiveness of robotic-assisted surgery.

Chapter 5 Proposed models

Some models mentioned in chapter 4 are based on U-Net, where the researchers looked for different strategies to improve the model efficiency. As such, the initial approach was starting with the U-Net as a baseline and exploring its several variants. Notably, some of these variants had not been previously applied for this specific task.

The focus of this chapter is to describe the models that were used for semantic segmentation. For each model, the architecture, and the innovation within were explained. This chapter focuses on the description of the U-Net, Nested U-Net and Transformer meets U-Net models.

5.1 U-Net

The U-Net is one of most employed networks for biomedical image segmentation due to its ability to generate a semantic map with more details [58]. The network is well known for its U shape, which coheres to the encoder and decoder. When it was presented to the scientific community, the original architecture had image patches as input, which are pixels grouped into windows of a local region. The model had to predict the class label of the central pixel, taking into account the surrounding region. The U-Net has the ability to propagate context information through the network, due to its large number of *feature* channels. The encoder block has 4 layers, where each one is made by two convolutions with a *kernel* size 3×3, without padding, each followed by an activation function, ReLU, and a Max-Pooling 2×2 with a stride of 2. For each downsampling, the *feature* maps increase twice from the previous one. On the other hand, the decoder also consists of 4 layers. Each layer involves a transposed convolution to reduce the feature count by half, followed by concatenation with the corresponding feature map from the encoder block. Following this, two convolutions with a 3×3 kernel size are applied, each followed by a ReLU activation function. In the end, it is employed a convolution 1×1 to decrease the feature maps for the desired number of classes. Figure 12 shows the U-Net architecture:

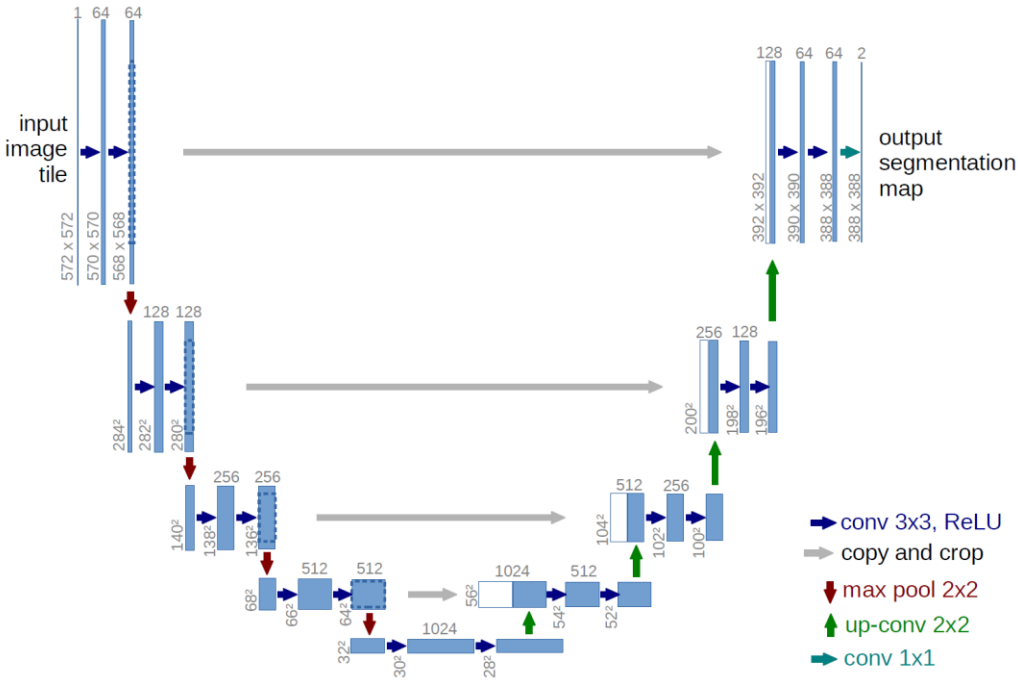

Figure 12. U-Net architecture [58].

Due to the lack of data, it is a good practice to resort to transfer learning techniques, that could leverage the learning of the model on big data. The transfer learning allows the models' weights to not start the training with random values. The adopted strategy was to search for a network that was trained in a large dataset and then transfer its knowledge to the U-Net. The major models, that are open source, are Visual Geometry Group (VGG) [59] and Residual Neural Network (ResNet) [60]. These two networks were trained on a large dataset, such as ImageNet [36], which contains data for classification proposes. VGG is a convolutional neural network with different kinds of depths. The network is composed by 6 convolutional blocks. The first two blocks consist of two convolutions separated by batch normalization and a ReLU function, while the remaining blocks have three convolutions. Between each pair of blocks, it is applied a Max-Pooling. In this context the backbone was used as feature extractor therefore, the last layers of the original VGG network responsible to provide the classification answer will not be considered. A residual block comprises two convolutional layers, with batch normalization and ReLU activation functions applied between them. Afterward, the input vector undergoes a shortcut identity, where all characteristics are preserved,

and it adds to the output of the last convolutional layer [60]. Figure 13 represents a residual block:

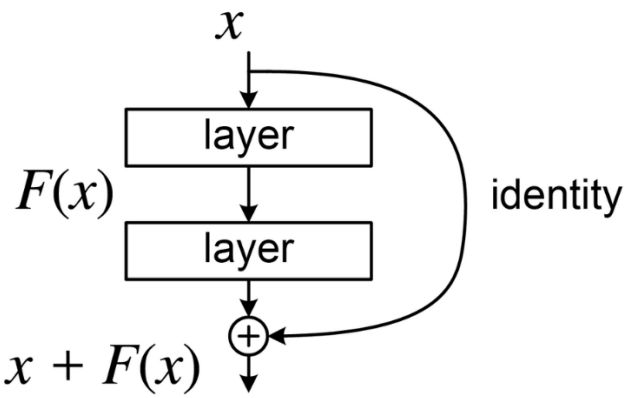

Figure 13. Residual block representation, where F(x) is the set of convolutionals, batch normalization, and activation functions; x is the input vector [60].

5.2 Nested U-Net

The Nested U-Net, also called U-Net++, links the backbone (encoder) with the decoder through a series of nested convolutional blocks through the skip connections [61]. The convolution blocks, which are called “skip pathways” by the authors, consist of three convolutional layers, where the input *feature* of each convolution layer comes from the concatenation between the output of the previous convolution layer and the up-sample output originated by a lower dense block. With the implementation of convolutional blocks, the network could bring the semantic level within the decoder to the decoder *features* maps. For example, the data to arrive from $X^{0,0}$ to $X^{0,1}$, it passes through a convolution block in $X^{0,0}$. After that, a downsample followed by a convolution block is applied, obtaining the $X^{1,0}$. The resulting features from the $X^{1,0}$ are upsampled and then concatenated with the features from $X^{0,0}$. Finaly, $X^{0,1}$ is created, by applying a convolution block in the concatenated features. In addition to the aforementioned, the model applies another item, which is deep supervision. It is a new approach to calculate the loss function, where the loss is the average of the above semantic levels, which are $X^{0,1}$, $X^{0,2}$, $X^{0,3}$, and $X^{0,4}$. The architecture is illustrated in Figure 14:

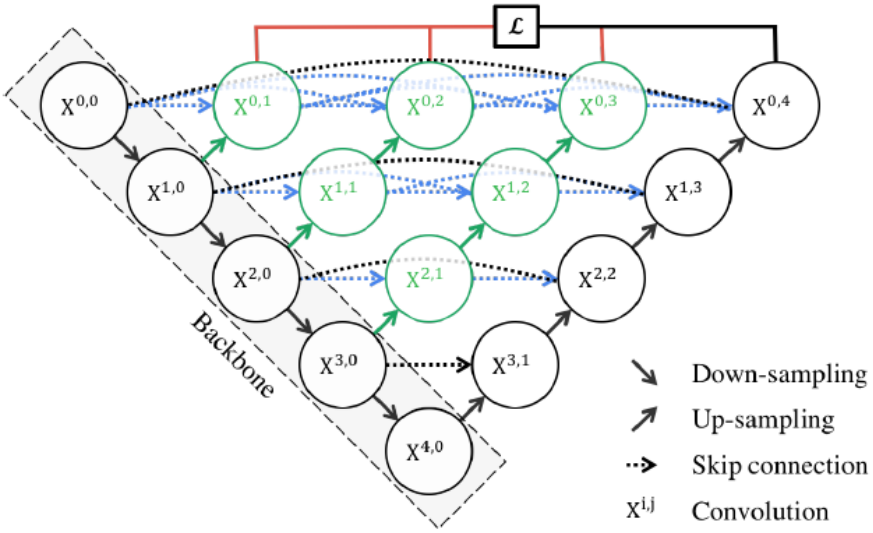

Figure 14. Nested U-Net representation, being the backbone the encoder [61].

5.3 Transformer meets U-Net

The proposed model introduced a recurrent neural network (a transformer) cooperating with a convolutional neural network [62]. The transformer has an encoder-decoder architecture, where the encoder maps an input sequence into a continuous representation. The decoder, taking the continuous representation, generates an input sequence one element at a time. As it is current in recurrent networks, the transformer is also recursive, where takes the previous generated element as an additional input. The encoder is composed by a stack of 6 layers, where each layer has two sub-layers. The first one is multi-head self-attention and the other is a feed-forward network. Each sub-layer applies a residual connection followed by an add and normalization layer. The decoder is also a stack of 6 layers, but unlike the encoder, it introduces a new sub-layer, masked multi-head self-attention. To ensure that the predictions for the position *t* only depend on the known outputs before *t*, both the inputs and the outputs pass through an embedding and a positional encoding. The first adopted step is the embedding, which consists of converting the input and output tokens into vectors that could be interpreted by the following applied sub-layers. To ensure that the order is preserved, the positional encoder gives context to each input embedding according to its position. The embedded

input with positional encoding feeds the multi-head self-attention. The multi-head could be seen as a set of Scaled Dot-Product Attention, that has three input vectors: query, key, and value. This works as a retrieval process, where the query is mapped to a set of keys (representing context or reference). The similarity between queries and keys determines how much attention should be given to each key with respect to the query. Values represent the content that is being searched, retrieved and combined based on the attention scores. The self-attention is computed using the dot product of queries and keys, and these scores are then used to weight the values generated from the input, creating the final output. This mechanism is often extended to include multiple heads to capture different relationships and dependencies within the data or focus on different parts of the input sequence, each with its set of vectors. Having already the three multi-head inputs, it is computed the Scaled Dot-Product Attention, which consists in the following equation(5.1):

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.1}$$

where Q, K, and V are the queries, keys, and values, respectively, d_k is the key dimension and K^T is the transposed vector key. The authors found it beneficial to project linearly the queries, keys, and values. The previous process is replied many times as pretended, whereas the final result of the multi-head self-attention is the concatenation of all Scaled Dot-Product Attention. The next step is the feed-forward network, which is a multi-layer perceptron. Another important block, present in the decoder, is the masked multi-head self-attention that has the same mechanism as multi-head self-attention, however, the main goal is to prevent positions from taking attention to subsequent positions. At the end of the decoder a linear transformation is applied, followed by a SoftMax [62], [63]. The architecture of a transformer is represented in the Figure 15:

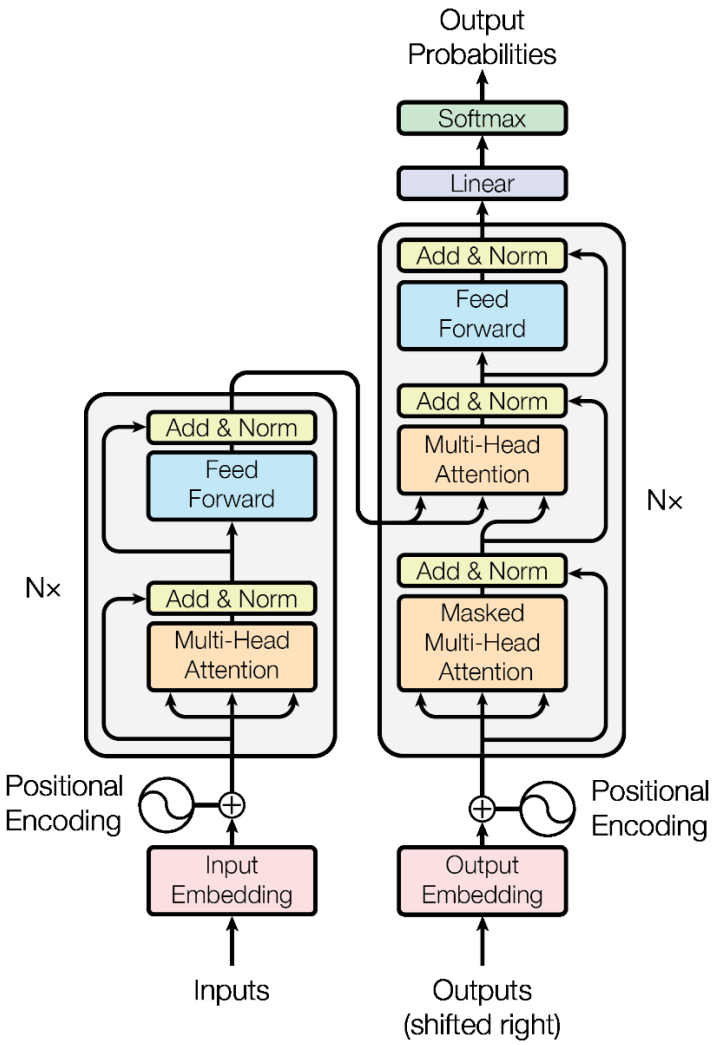

Figure 15. Model architecture of a transformer [63].

The previously explained transformer describes a generic application. In this context, the Vision Transformer was the chosen model that only has the decoder block repeated 4 times, for each patch image as input. The number of multi-heads in the decoder is 16, so the patch size was 16×16. The Vision Transformer cooperates with a U-Net with ResNet as the backbone, where the segmentation map is the combination of these two through a spatial normalization. The spatial normalization repeats the elements from the vector of the transformer until they have the same dimension as the U-Net output. Then concatenate the resultant matrix with the U-Net output.

5.4 Attention gate

One of the major problems in semantic segmentation is the lack of attention that the models give to specific features. To avoid this issue, attention mechanisms that enhance relevant features to return a better segmentation are suggested. Many had already been proposed, based on convolutional networks, such as the convolutional block attention module [64], or on recurrent networks [63]. However, these attention modules exponentially increase the number of parameters in a network, restricting the train conditions, like the input patches, batch size, and so on. A research team presented an attention module that could be easily inserted into a network, without a high memory cost. The attention block, named as attention gate, can increase sensibility and prediction accuracy by learning to focus on target structures, without any additional supervision [65]. The attention gate has two input arguments, where each argument passes through a convolutional layer with a kernel size of 1×1. Next, the attention gate concatenates them followed by an activation function, ReLU. The flatten array undergoes through a convolution layer accompanied by a batch normalization and an activation function, *sigmoid*. Finally, the flattened array is multiplied by one of the input arguments [65], [66]. The attention gate is illustrated in Figure 16:

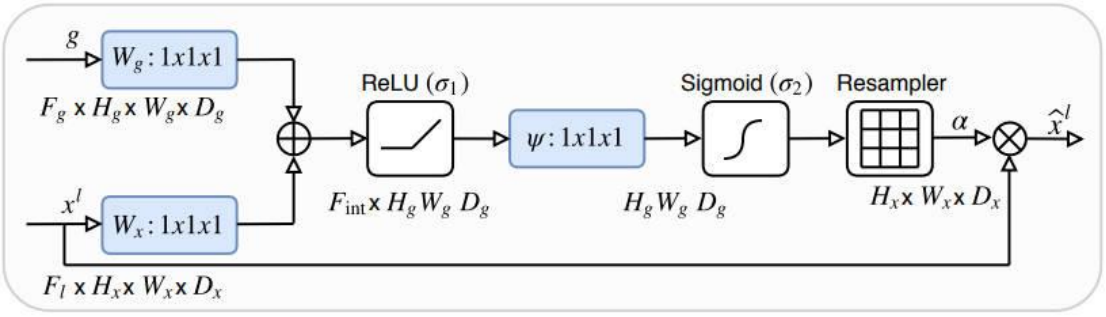

Figure 16. Illustration of the proposed attention gate [65].

The attention gate was implemented on the U-Net, more specifically in the skip-connections. One of the arguments is from the encoder block, while the other is from the decoder block. As mentioned before, one of the input arguments multiplies with the flattened array, which in this case, would be the encoder argument.

5.5 Conclusion

U-Net is a widely used deep learning model for performing semantic segmentation on biomedical images. Its encoder-decoder structure enables it to extract crucial features from the input image. Over the years, numerous approaches have been proposed to enhance this baseline model. Some approaches involve replacing the backbone of the U-Net with different architectures and incorporating pre-trained backbones. This allows the model to leverage the knowledge learned from large-scale datasets and transfer it to the task of biomedical image segmentation. Other approaches focus on increasing the number of layers between the skip-connections in order to introduce deep supervision. This helps in capturing more detailed information and improving the overall segmentation performance. More recently, researchers have explored the integration of U-Net with RNN models or incorporating attention blocks. These additions aim to further refine the segmentation results by incorporating temporal or spatial dependencies and highlighting specific regions of interest.

By enriching the baseline U-Net model with these various approaches, it is expected that the segmentation performance can be significantly improved, leading to better scoring metrics and more accurate segmentation results in the field of biomedical imaging.

Chapter 6 Data augmentation

Deep learning models have remarkable performance in discriminative tasks. This phenomenon is explained by the appearance of advanced networks architectures, powerful computation, and access to big data. These networks are very reliant on big data to avoid overfitting, but many application domains do not have access to big data. The *deep learning* system, to be useful, must presents a decreasing of the validation error with the training error. Data augmentation is a very powerful technique to guarantee this. This approach increases the information that can be extracted from the original dataset. The augmentation can be based on geometric transformations, such as flipping, cropping, rotation or even translation, or can be based on sampling from data distributions that can be learned by *deep learning* approaches. The main problem of geometric transformations is not having the ability to create shapes or appearances of the same surgical instrument. The generation of synthetic data could unlock additional information from a dataset if the generated data can span the entire distribution [67].

An effective data augmentation depends on 3 fundamental factors:

1- The existing database does not need to be very extensive, but it needs to cover the entire sample space of the application. This factor doesn't depend on the effectiveness of the generator machine but in the insufficiency of the dataset.

2- The training of the generative system must adequately capture the distribution of the training data.

3- As a generator, the generative system must span the entire extension of the distribution, even the areas of lower likelihood that tend to be discarded given that in estimation theory, the most likely events give better performance to the systems. Therefore, in practice it becomes difficult to deal with this assumption and complements are normally required for more efficient generation. Joining other types of information into conditional probability density functions, thus increasing the likelihood of the data, making a conditional generation on an existing sample (image), or controlling the

diversity of the generated image (for example through FID) are some of the approaches used to include unlikely samples in the generation process [67].

6.1 Generative Adversarial Network (GAN)

Despite shown discriminative power in different areas of application, the deployment of current ML approaches in medical environments remains constrained. This limitation is due to numerous factors, such as the lack of properly annotated and adequately diverse datasets that would enable neural networks to comprehensively learn about the given problem. In other words, the first problem described above interferes with the following ones, without sufficient data it is not possible to estimate an adequate model and without it is not possible to generate data coherent with the application. In order to mitigate such issues, *Generative Adversarial Networks* (GAN) were proposed to generate artificial data with similar characteristics of real data. GAN is a convolutional network with *unsupervised learning*, where their major objective is generating images, taking into account the problem. Its operation is based on a competitive system, where it could discern two models: generative and discriminative. The generative model is responsible for image synthetization starting from the latent space or noise. It can create a realistic image with the aim of misleading the discriminative model. The discriminator has the role of discerning between a real image and an image made by the generative model. Its classification task, which is true or false, is made in percentual values, so knowing that the generative model will create an image the most realistic as possible until the discriminator is unable to differentiate between a genuine and a synthesized image. The discriminator model, to bypass this issue, will try to increase the percentual disparity between a real image and a false one. In a simplified way, this process could be represented by Equation (6.1):

$$\min_{G} \max_{D} [E_{x \sim p_{data}} \log D(x) + E_Z \log(1 - D(G(z))] \tag{6.1}$$

Where *G* and *D* are the generator and discriminator respectively, having as input the latent space z and parameters x . The loss function of this network is produced by the accumulation of the individual loss functions, E, from the discriminator and generator. For being an *unsupervised learning*, the model will be trained without any knowledge of its purpose, which usually leads to the generation of blurred images. This phenomenon occurs in the generative model because to be capable of deceiving the discriminative model, the generator will decrease the Euclidian distance (distance between the pixel and its neighbor). However, the generative model learns the data distribution, without separating it by classes. So, the generator loses the ability to drop the Euclidian distance. To avoid this outcome, in addition to the latent space, a condition is introduced. The condition could be a semantic map that is fed into the generative and discriminative models, so the GAN can learn the labels and their differences [68], [69].

Due to the diversity of condition formats, some authors classified two major dataset groups, for this type of convolutional network: paired dataset and unpaired dataset. The paired dataset is split into two sets, maintaining a correspondence between each respective set, so the number of elements in each group must be equal. The other type of dataset has also two groups, but there is not correspondence between an image from one set to another. The unpaired dataset is the most common because usually it is intended to transfer an image to another domain. However, there will not be any similarity between the real and the generated images since only the original image and a representation from the other domain are supplied. The GAN could proceed to different processes of image generation, but taking into account the type of dataset available, it is intended a GAN capable to do a supervised *image-to-image translation* (I2I). In I2I, the generative network can model the distribution of each class label, producing a fake image like the target domain. The generative model presumes that the synthesized data follow a Gaussian distribution and tries to approach this distribution into a particular algorithm. With this methodology, the generative model has the capacity to create new data due to its modulation of the target domain. The I2I translation could be split into two categories: unimodal output and multimodal output.

The first one, the model translates an input image to another domain, while the multimodal translates a single input image to a distribution of possible outputs. Considering the previous statements and the target objectives, the following content looked for a conditional model that can generate images with high resolution. Since the goal was to generate identical samples from real data increasing diversity without changing domains, the selected model should be able to train in a paired dataset [70], [71].

Like in other *deep learning* approaches, several metrics have also been proposed to evaluate the image quality. The most common are the following: Peak signal-to-noise ratio (PSNR), Structural similarity index (SSIM), Fréchet inception distance (FID), and Learned Perceptual Image Patch Similarity (LPIPS). The PSNR measures the intensity distortions between the translated image and the ground truth. This metric is expressed in a logarithmic scale due to the very wide dynamic range of many signals, and higher PSNR value means a close intensity between the two images. SSIM is a metric that analyzes three characteristics, namely luminance, contrast, and structure between two computed images. Like the previous one, a high value indicates a strong similarity in these parameters. The FID is a more complex evaluation system that needs to extract *features* representations from the real and generated images, like edges and higher-order shapes, from an Inception-v3 model. Then calculates the mean and covariance matrix of the features in each image. Finally, computes the Fréchet distance between the real and generated images and unlike the priors, a lower FID means a better performance. The LPIPS evaluates the diversity of the generated images by calculating the average LPIPS distance between pairs of randomly selected translated outputs derived from the same input. This metric can measure diversity and similarity, where a high value indicates diversity in the generated image and a low value shows the similarity, depending on what it is looking for [72], [73].

The conditional GANs that fulfill the requirements are the following: pix2pix [74], DRPAN [75], pix2pixHD [76], 'Semantic Image Synthesis with Spatially-Adaptive Normalization' (SPADE) [77], 'Image Synthesis with Semantic Region-Adaptive

Normalization' (SEAN) [78] and 'Full-Resolution Correspondence Learning for Image Translation' (CoCosNet-v2) [79], [71].

Work in image synthetization for laparoscopy surgery is still relatively low explored. Marzullo et. al [80], proposed an application of a conditional GAN as a data augmentation method. They used the pix2pix GAN to I2I translation, producing an image similar to the real ones. The generator in pix2pix follows a U-Net architecture, meaning it is a convolutional network following an encoder-decoder structure with skip-connections. This sub-network receives as input the latent space and a semantic map. The discriminator, unlike other conditional GANs, has the real and generated images as input without semantic map. Therefore, it lacks the capability to classify the data distribution by class. In this work, the researchers used metrics that are more common for image segmentation purposes, such as Dice, precision, and F1-score, and did not apply the usual FID.

Considering the lack of research conducted on laparoscopy image synthesis, the selection of the best model will fall on the quality of image generation by the model. To best models were selected relying on a state-of-the-art dataset, like '*ADE20k*' [81], [82] or '*Microsoft Common Objects in Context (COCO)*' [83], which are widely recognized by the scientific community. These data consist of a real image along with its corresponding semantic map. The choice was conducted by analyzing the FID metric because, at our best knowledge, it is the most used to deliberate the image quality.

6.1.1Pix2pixHD

Wang et. al [76] proposed a conditional GAN capable of generating images with high resolution. In this work, the authors declared that can synthesize 2048×1024 images with photo realism. To accomplish this claim, they applied a multi-scale generator as well as discriminator architectures. The generator was broken into two, being classified as the global generator and local enhancer. The global generator works

at a resolution of 1024×512, while the local enhancer outputs an image with a resolution of 512×256. Figure 17Figure 18 shows pix2pixHD architecture:

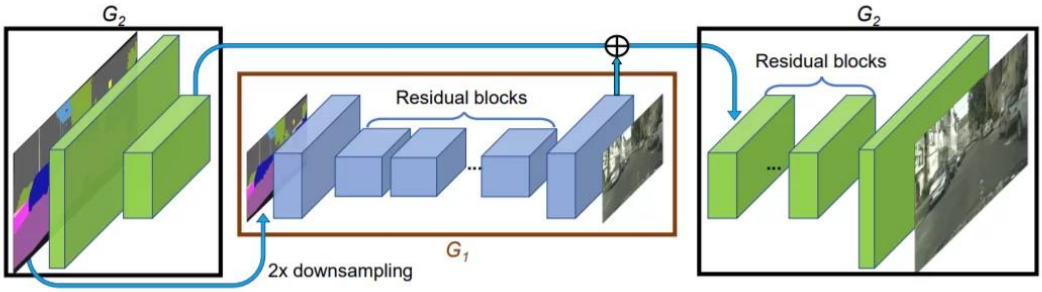

Figure 17. Pix2pixHD architecture, where G_1 is the global generator and G_2 is the local enhancer generator [76].

The global generator, represented by G_1, is composed of three elements: a convolutional block in the beginning, followed by a series of residual blocks, and concluding with a transposed convolutional block.

The transpose convolutional block, also known as deconvolutional, reverses the operation of a standard convolutional layer.

The local enhancer generator has the same three elements as the previous one, but, unlike the other, the input feature map of the residual blocks is obtained by the element-wise sum of the output from the convolutional element of the local enhancer and the output from the global generator. Through this procedure, they sustain that global information is incorporated into the local enhancement network. During the training, the authors proposed first training the global network and then the local enhancer. After the separated train, they suggested joining the networks to do fine-tuning.

The discriminator requires an extensive receptive field, in order to differentiate high-resolution real and generated images. To avoid increasing memory usage, they applied three discriminators that discern only in terms of the image scales they analyze. Both real and synthesized images are downsampled by a factor of 2 and 4 to create a pyramid of three scales. This pyramidal structure provides a global view complemented with finer details. The discriminator architecture is named Patch Discriminator, which is

composed by 5 convolutional layers, with an activation function and a batch normalization between. This Patch Discriminator classifies each $N \times N$ patch in an image. This discriminator undergoes convolutionally across the image, being the ultimate output the average of all responses. The patch classification permits the parameters reduction [74]. In Table 6 the metrics achieved by Pix2pixHD in *'ADE20k'* dataset, are shown:

Table 6. FID achieved scores by Pix2pixHD in 'ADE20k' dataset

GAN	FID
Pix2pixHD [76]	81.8

Although the GAN could generate images with high resolution, it had an extremely high cost on GPU memory, and it can not preserve the semantic information across the network.

6.1.2 SPADE

Park et. al [77], proposed a spatially-adaptative normalization to synthetize photorealistic images. They defended that the previous methods degrade the semantic information. The SPADE block is represented in Figure 18:

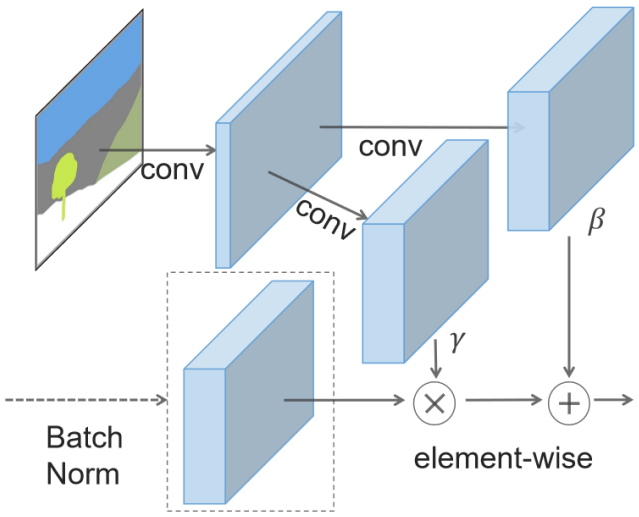

Figure 18. SPADE block, where γ and β represent modulation parameters [77].

As shown, the semantic map is projected to an embedded space and then it is convolved to engender new modulation learnable parameters, such as γ and β. After this process, the γ and β are multiplied and added to the batch norm. The activation value is given by Equation (6.2):

$$\gamma^i_{c,y,x}(m)\frac{h^i_{n,c,y,x}-\mu^i_c}{\sigma^i_c}+\beta^i_{c,y,x}(m) \tag{6.2}$$

Where $h^i_{n,c,y,x}$ represents the activation before the normalization, being i the number of layers of a deep network, c the channel, y the height, x the width, n the batch sample, and m the semantic segmentation map. Instead of feeding the network with the semantic map, this information was injected into all SPADE blocks that were placed within the generator network. Consequently, the authors discarded away the encoder, remaining the decoder in the generator. The final product was a generator, where the encoder was composed by residual blocks with SPADE blocks and up-sampling layers. The discriminator is a Patch Discriminator that also applies the three scale levels.

The last point discussed by the authors was the random vector as the input of the generator. They applied an image encoder, which is responsible for extracting a 256-feature vector containing the mean and the variance from the real image. The image encoder was built up by a set of convolutional layers, where between each one there was an activation function and batch normalization. In the last convolutional layer a two linear featurization was applied to obtain the mean and the variance. The final architecture is given in Figure 19:

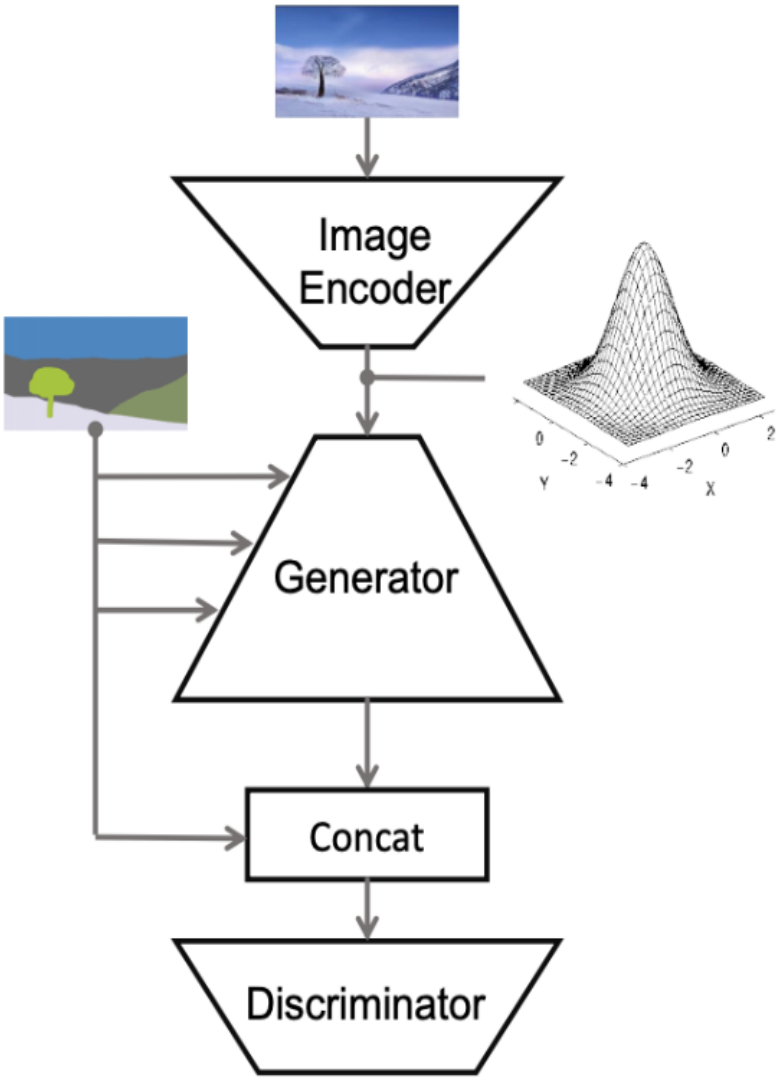

Figure 19. SPADE conditional GAN architecture [77].

The image encoder implementation requires the addition of a new loss function, namely Kullback–Leibler divergence. The Kullback-Leibler divergence measures how much one probability distribution differs from another, over the same variable. If two distributions match, this measure will be 0, otherwise, the value goes from 0 to ∞ [77], [84]. Table 7 Shows the metrics achieved by SPADE in '*ADE20k*' dataset:

Table 7. FID achieved scores by SPADE in 'ADE20k' dataset

GAN	FID
SPADE [77]	33.9

One of the problems in the SPADE is the vanishing of texture patterns. The texture is a crucial characteristic on medical images because it is a function of color variation that forms repeated patterns [85].

6.1.3 CoCosNet-v2

Zhou et. al [79] referred that one of the main issues of the GAN was exploiting the information represented in an exemplar image. This network resorts to an exemplar image, where it transfers the information within the exemplar to the synthesized image. To accomplish this, the authors proposed a multi-level domain alignment, which was responsible for extracting *features* through a U-Net. The multi-level domain alignment involved extracting 4 levels of features from both the exemplar image and the semantic map. The chosen resolutions for these *features* were 64 × 64, 128 × 128, 256 × 256, and 512 × 512. After the *features* acquisition, the authors suggested a coarse-to-fine strategy, which consists of correspondences between features of both exemplar images and semantic map. This followed a hierarchical process, meaning that the correspondence matching started at the lowest resolution level. In the successive levels, the matching results from the previous layer were used as the initial guidance. The research team split this phase into two parts: GRU-assisted Patch Match and differentiable warping function. The first part was propagation with a Patch Match. This network was responsible to do the matching between the *features* of the exemplar image and the semantic map, knowing the matching results of its neighbors. Patch Match typically examines spatially adjacent patches and can become stuck in local optima. To enhance propagation, the team incorporated a Convolutional Gated Recurrent Unit (ConvGRU) in addition to Patch Match. The Gated Recurrent Unit (GRU) is a type of RNN designed to mitigate the vanishing gradient problem often encountered

in standard RNNs. The GRU uses two types of gates: update and reset gates. The first is the update gate and is given by the following equation:

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}) \tag{6.3}$$

When x_t is plugged into the network, it is multiplied by its own weight, denoted as $W^{(z)}$, and then this result is added to the product of the previous information t-1 and its own weight, $U^{(z)}$. A *sigmoid* activation function is employed afterward, making the values vary between 0 and 1. This gate is responsible for determining how much past information should be passed along to the future. The next gate, the reset gate, chooses the past information to forget. The reset gate follows a similar equation to the update gate, with the distinction lying in the weights, which are specific to the reset operation:

$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right) \tag{6.4}$$

After the gate resolution, the current memory content is computed, which has the ability to preserve important information from the past. The current memory is obtained through this equation:

$$h'_t = \tanh\left(Wx_t + r_t \odot Uh_{t-1}\right) \tag{6.5}$$

Where W and U represent weights associated with the current memory content. In this equation, a Hadamard product was computed between the reset gate and the previous information, which helped determine what information to discard from past time steps. The Hadamard product makes an element-wise product, resulting in a matrix composed of the corresponding elements multiplied from two matrices. After the operations, a *tanh* activation function is applied. In the final stage, the network calculates the vector that stores the information. To accomplish this, Equation (6.6) is employed:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t \tag{6.6}$$

As can be seen, if z_t is set to 1, the h_t will preserve the majority of the previous information and discard the current content [86], [87]. Figure 20 shows a GRU architecture:

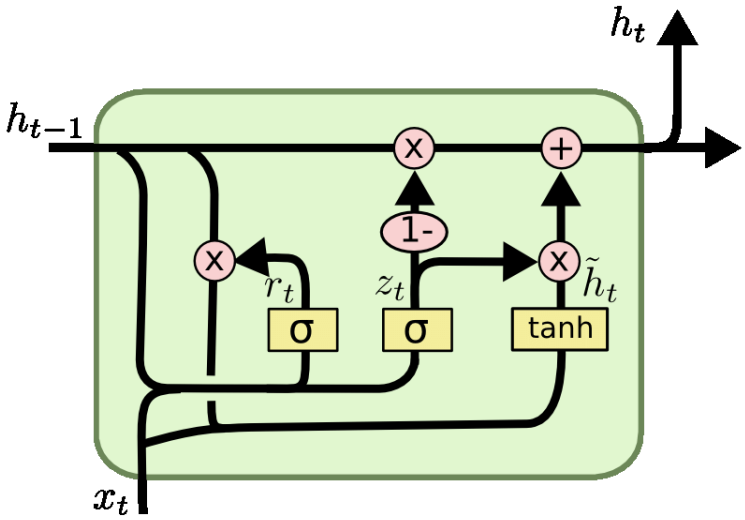

Figure 20. Gated Recurrent Unit architecture [87].

The last result of the propagation, after the data passed through the Patch Match and the GRU, was the addiction between the output of the Patch Match and the output of the GRU. Lastly, following the propagation, the subsequent step involved translation is applied. In this step, the warped image, created by the Patch Match correspondence field, which transformed the exemplar image, was passed through two convolutional layers to generate modulation parameters. The parameters were plugged into the SPADE blocks located within the decoder network. This whole process, CoCosNet-v2 network, is shown in Figure 21:

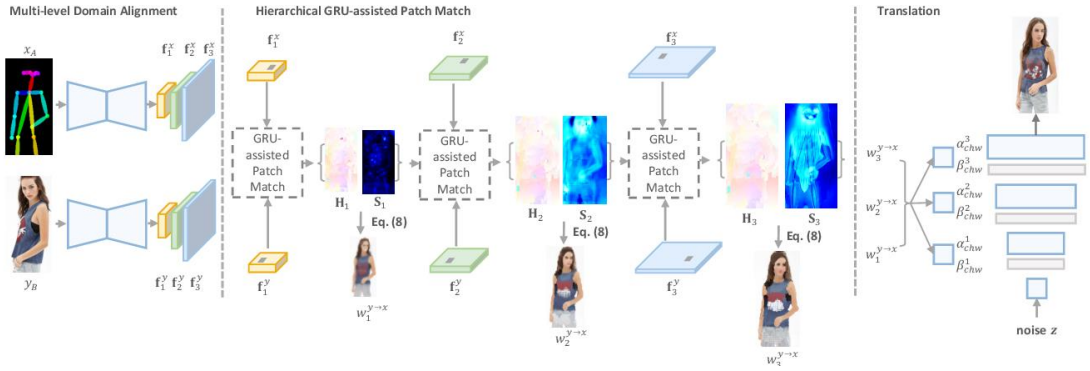

Figure 21. CoCosNet-v2 architecture, where x_A is the semantic map and y_B the exemplar image [79].

Like the previous GANs, it was also calculated the FID in *'ADE20k'* dataset, as it can see in Table 8:

Table 8. FID achieved scores by CoCosNet-v2 in 'ADE20k' dataset

GAN	FID
CoCosNet-v2 [79]	25.2

6.1.4 GANs drawbacks

This type of generative models has some limitations. GANs suffer from collapse and instability during training. The mode collapse refers to the state that the generator only produces similar images. This shows that the GAN has difficulty achieving the Nash equilibrium (the optimal outcome is accomplished when there is no incentive for the generator or discriminator to deviate from their initial strategy). To address this issue, one can explore various strategies, which may involve adjusting the architecture of the generator. These strategies could include adding or removing layers from the generator network, extending the training duration to allow for more learning, or incorporating techniques like dropouts and batch normalization within the generator's layers [88].

Another problem that could emerge is the vanishing gradient during the backward step. As it flows, the gradient gets increasingly smaller, and sometimes it stops learning if the gradient is extremely small prohibiting initial layers from learning by not changing the weight values. This issue gets worse when training bigger networks, leading

to non-convergence situations. Some researchers proposed balancing the discriminator and the generator during the training phase, but this approach is not optimal because a good discriminator gives good feedback. Another potential solution is to adapt the objective function by introducing a new probability distance metric or a different loss function [72], [73], [88].

6.2 Diffusion models

A limitation of GANs is their lack of ability to insert variability/diversity in the generated image. The diffusion models are one possibility to generate realistic images with diversity very different from the ground truth, maintaining the detail. Furthermore, diffusion models can also be applied in classification, detection, and segmentation tasks, but they are categorized as generative models, due to their ability to learn the data distribution.

This section explains the diffusion model mechanism. It is also approached the different types of diffusion models existing until now. Finally, the model applied to generate laparoscopic images is described.

6.2.1 Description

Diffusion model is a *deep learning* generative network with two stages. The first stage, also called by forward stage, is composed by successive steps to inject Gaussian noise in the ground truth image. The second phase, named by backward stage, walks through the steps in the opposite direction so that it can recover the original image [89]. All diffusion models apply this concept, being based on Markov chains. The Markov chain is a stochastic process with numerable states. The stochastic process foretold that future states depend only on the instant state and not on the previous ones. The time parameter, also called by step, must be a non-negative integer number. Through this approach, the Markov chain will converge to an equilibrium point, when the network trains during a certain time [90].

This type of diffusion model could be separated into three subcategories: denoising diffusion probabilistic model (DDPM) [89], noise-conditioned score networks (NCSN) [91], and stochastic differential equations (SDE) [92]. For the purpose of this work, the generative model must be conditioned by the semantic map and must be for image-to-image translation. Therefore, only DDPM is explained in the next sub-section, as it is the unique type that fulfills such requirements.

6.2.2 DDPM

DDPM has the main function of estimate the probabilistic distribution of the latent space variations. To achieve this, DDPM induces variations within the latent space. DDPM slowly injects Gaussian noise into the training data, following Equation (6.7):

$$p(x_t|x_{t-1}) = N\left(x_t; \sqrt{1-\beta_t}.\, x_t, \beta_t.\, \boldsymbol{I}\right), \forall t \in \{1, \dots, T\} \tag{6.7}$$

Where $p(x_t)$ describes the data density. T represents the number of steps for noise injection, so when t is equal to 0, the data will be the original data. $\beta \in [0, 1]$ are parameters that represent the variance across the diffusion steps. $\boldsymbol{I}$ is the identity matrix with the same dimensions of the data. $N(x; \mu, \sigma)$ is the normal distribution, with the mean represented by μ, and variance by σ. One property of this formulation is obtaining any version x_t only knowing the original image and $\hat{\beta}$. This premise is described in Equation (6.8):

$$p(x_t|x_0) = N\left(x_t; \sqrt{\hat{\beta}}.\, x_0, \left(1-\hat{\beta}_t\right).\boldsymbol{I}\right) \tag{6.8}$$

Where $\hat{\beta} = \prod_{i=1}^{t} \alpha_i$, being $\alpha = 1 - \beta_t$. The process to achieve x_t is made by a reparameterization method. The method refers that the standard value from a sample x that follows a normal distribution, $x \sim \mathcal{N}(\mu, \sigma^2.\boldsymbol{I})$, can be obtained by subtracting

the mean, and divide it by the variance, $z = \frac{x-\mu}{\sigma}$. For being a reparameterization method, it is possible to achieve any x_t value following Equation (6.9):

$$x_t = \sqrt{\hat{\beta}_t}.x_0 + \sqrt{(1-\hat{\beta}_t)}.z_t \tag{6.9}$$

As could be ascertained, the selection of $(\beta_t)_{t=1}^{T}$ that induce the convergence of $\hat{\beta}_t$ to 0, makes x_T to follow a normal distribution. As a default, the selected values for β_1 and β_T are 10^{-4} and 2.10^{-2}, respectively. It is possible to generate other samples from $p(x_0)$ if the network starts from $x_T \sim \mathcal{N}(0, \boldsymbol{I})$ [93], [89]. The network needs also to follow the reverse steps:

$$p(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu(x_t, t), \sum(x_t, t)\right) \tag{6.10}$$

To approximate the steps, the network is trained to receive the noisy image and embedding time steps as input, where it could learn to predict the mean and the covariance. The covariance is a constant value, so it could be represented by Equation (6.11):

$$\mu_\theta = \frac{1}{\sqrt{\alpha_t}} \cdot \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\widehat{\beta}_t}} \cdot z_\theta(x_t, t) \right) \tag{6.11}$$

With this simplification, the objective function can measure the distance between the noise z_t and the estimated noise $z_\theta(x_t, t)$, for any time step. The objective function could be simplified by Equation:

$$\mathcal{L} = \mathbb{E}_{t\sim[1,T]} \mathbb{E}_{x_0 \sim p(x_0)} \mathbb{E}_{z_0 \sim \mathcal{N}(0,\boldsymbol{I})} \|z_t - z_\theta(x_t, t)\|^2 \tag{6.12}$$

Where $\mathbb{E}$ is the estimated value, $z_\theta(x_t, t)$ is the predicted noise in x_t. The network does not predict the mean and the covariance directly, instead, it predicts the

noise from the image and through the noise determines the mean, while the covariance is a fixed value [93].

6.2.3 Semantic Diffusion Model (SDM)

SDM is a DDPM network and is based on U-Net. Like the other diffusion models, the network estimates the noise from the input image. It starts by processing the noisy image and the semantic map independently, where the noise is fed into the encoder, while the semantic map is into the decoder. The encoder is composed by semantic diffusion *resblocks*, which are blocks with convolution layers, followed by an activation function, SiLU ($f(x) = x.sigmoid(x)$), and group normalization (an alternative of batch normalization, where the channels are divided into groups and the mean and the variance for the normalization is computed for each group [94]). The decoder is where the semantic map is injected through SPADE blocks, followed by convolution layers and activation functions, SiLU [95].

Being a diffusion model, for each defined step, there is a network responsible for estimate the mean and the covariance of the Gaussian noises the network could learn to map the noise distribution. The authors also proposed a classifier-free guidance for generating photorealistic images. They defend that by perturbing the mean, the results could be improved, so during this second train phase, the network has a copy of itself, but this copy will not be fed with the semantic map, but by an empty label. The conditional noise and the unconditional noise are added, and multiplied by a scalar value, which is responsible for creating a tradeoff between diversity and fidelity. After that, unconditional noise is added with the aforementioned operation, resulting in the estimated noise. Figure 22 represents a schematic of the Markov chain in the diffusion model:

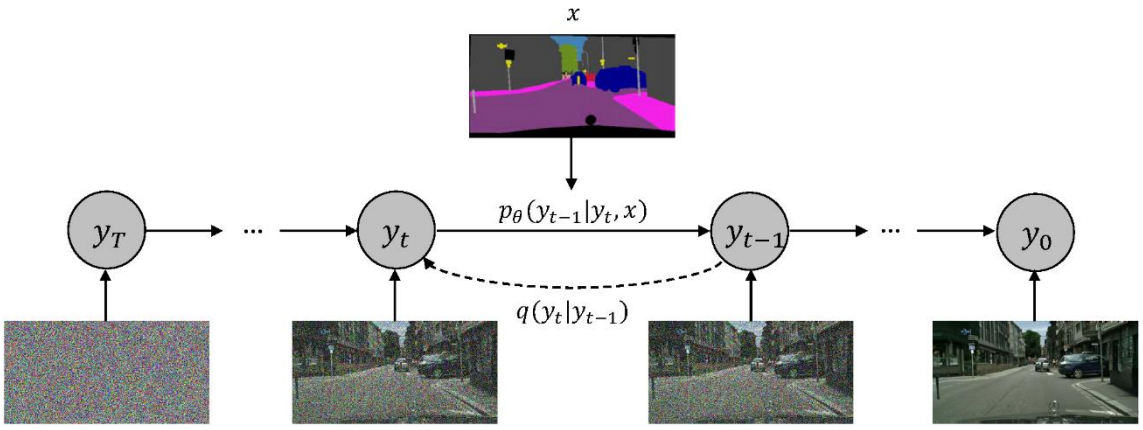

Figure 22. Markovian chain of SDM [95].

The diffusion model, like the mentioned GANs, was also analyzed, calculating its FID for the *'ADE20k'* dataset. The scoring results are in the Table 9:

Table 9. FID achieved scores by the diffusion model in 'ADE20k' dataset

Diffusion model	**FID**
Semantic diffusion model [95]	27.5

This model attained a low FID value, like the GANs. Generation models based on diffusion, have great potential in terms of synthetic generation data.

6.3 Variational Autoencoders

Variational Autoencoders (VAE) are *unsupervised* generative models with the main goal to learn complicate distributions. VAE consists of an encoder, which is responsible for compressing the input data into a latent space, and a decoder that has the capability to reconstruct the data from the latent space. In this class of generative models, the focus shifts from directly learning the encoding to modeling the latent space as a Gaussian distribution. During training, the neural network aims to minimize two key objective functions: the reconstruction loss and the regularization term loss.

The reconstruction loss plays a crucial role in ensuring that the decoder can faithfully recreate the original input data. For the specific task of image generation, the mean-squared error is a common choice for this objective function. It quantifies how well the generated data matches the input data, emphasizing the accuracy of the reconstruction. Concurrently, the regularization term loss influences the encoder's behavior. Its goal is to encourage the distribution of the latent space to closely resemble a Gaussian distribution. To achieve this, the Kullback-Leibler divergence is most frequently employed. The KL divergence measures the dissimilarity between the true distribution of the latent space and the desired Gaussian distribution, thereby promoting a latent space that exhibits Gaussian-like properties. By minimizing both the reconstruction loss and the regularization term loss, VAEs strike a balance between generating accurate reconstructions of the input data and shaping the latent space to have specific probabilistic characteristics. This duality enables VAEs to generate new data samples that not only resemble the training data but also exhibit useful properties in the latent space [96].

Once the network is trained, it can generate data from the learned latent space. These samples from the encoded space are decoded by the decoder, generating new data. Although VAEs leverage probabilistic encoding in a lower-dimensional latent space, they present innumerous disadvantages. This type of generative models is characterized by producing images with low quality compared to the GANs. VAEs are commonly created with the goal of acquiring disentangled representations in the latent space, where each dimension signifies a significant feature. Nonetheless, attaining complete disentanglement poses a challenge, and VAEs may not consistently succeed in effectively separating all the fundamental factors of variation present in the data. Furthermore, VAEs struggle to capture multimodal data distributions, such as when the data exhibit non-Gaussian characteristics [96]. Given the primary objective of generating realistic images in the context of generative models, VAEs may not be the ideal choice, as they might not reliably ensure the quality of generated images. Therefore, they were excluded from consideration for this task.

6.4 Conclusion

Generative networks constitute a subset of deep learning systems, distinguished by their capacity to generate new data while drawing inspiration from real data. In this chapter, three distinct types of generative models were discussed: GANs, diffusion models, VAEs. GANs are founded on a competitive learning paradigm where two networks enhance their knowledge through mutual competition. Diffusion models, on the other hand, employ a Markov chain to iteratively learn the data distribution by introducing Gaussian noise at each step. In contrast, VAEs are unsupervised networks that mold a latent space to possess specific probabilistic characteristics, enabling them to generate novel data samples.

The objective of employing generative models in this study was to generate synthetic data, thereby expanding the training dataset. This data augmentation technique yielded promising results.

Chapter 7 Experimental Results

This chapter presents the used dataset and describes the selection of the most adequate models for the purpose of the detection and segmentation of parts of robotic harms used in assisted surgery. Discussion of results is also provided based on the model's properties.

7.1 Dataset description

The dataset is from the "*Surgical Instrument Multi-Domain Segmentation Challenge"* [97] and is public. It is divided into four different surgical domains: laparoscopic in-vivo, laparoscopic ex-vivo, robotic in-vivo and robotic ex-vivo. For each domain, there are collections of images captured by the endoscope, organized based on the procedure/patient. For each reference image there is multiple segmentation masks, such as instrument type and parts. The instrument type provides information about the specific type of tool being represented, while the parts distinguish the different components of a tool. There are 11 different surgical instruments that are listed in Table 10:

Table 10. Tool type used during the surgical procedure

Tool Type
Clipper (C)
Coagulation Instrument (CI)
Scissors (Sc)
Stapler (St)
Needle Driver (ND)
Grasper (G)
Holding Instrument (HI)
Suction/Irrigation (SI)
Specimen Bag (SB)
Trocar (T)
Undefined instrument (UI)

As an example, in Figure 23 there are masks of instrument type, being represented a ND with CI and SI.

(a) (b)

Figure 23. Semantic map of (a) patient 01, frame 21, on the exvivo_robotic domain ;(b) patient 01, frame 6, on the invivo_laparoscopic domain.

On the other hand, each instrument could have at most 3 different parts, depending on its type. These parts are registered in Table 11:

Table 11. Parts that compose the instruments

Tool part
Jaws
Wrist
Shaft

For a better understanding of the instruments parts, in Figure 24 is illustrated a representation of an instrument with the three classes:

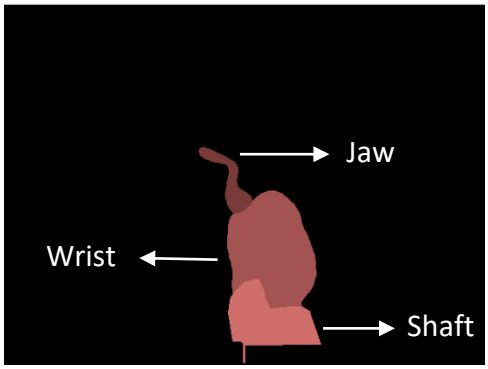

Figure 24. Semantic map of patient 01, frame 60, on the invivo_robotic domain.

It is necessary to mention that for instrument type in the dataset, there were two missing classes that were discarded for training purposes, namely, 'Staple' and 'Needle Driver'. As expected, the different classes mentioned before do not have the same representation, which could hinder the class segmentation by the lack of appearances.

As such, the first part was to understand how many times each class appears on the dataset. The dataset is composed by 45 patients, making a dataset size of 2349. For each patient is given the semantic map of instrument parts and type. Table 12 and Table 13 illustrate the number of occurrences of each class in tool's type and tool's part, respectively.

Table 12. Number of occurrences of each class of tool type

Tool type	Number of occurrences
C	213
CI	1310
Sc	536
St	0
ND	0
G	1282
HI	397
SI	95
SB	38
T	90
UI	43

The data for segmentation of tool parts is more evenly distributed than for segmentation of tool type. Observing the class distribution of tool parts, the most difficult to achieve a high metric will be the wrist class. For the tool type, is expected that less representative classes, such as SI, SB, T and UI, have a metric very close to 0.

Although some classes have a good representation on the dataset it is not guaranteed that these classes will achieve a high scoring value due to other factors, like image quality or if the instrument/part is cut in the visual field.

Table 13. Number of occurrences of each class of tool part

Tool part	Number of occurrences
Wrist	959
Shaft	2044
Jaw	2115

7.2 Cross-validation

Cross-validation is the most used method for error estimation. On a flawless world, data is split into two groups: training and validation. However, this is not a recommended practice when data is scarce. To contour the issue, it resorts to *K-folds* cross-validation, which applies part of the data to fit the model and a different part to test it. The dataset is divided into *K* sets with equal sizes, where the model is fitted exploiting $K - 1$ folds of the data. Then, the model is validated on the remaining portion of data. In each fold, a prediction error is obtained, and the model's performance is determined by taking the average of these score values. Through this method, it's estimated the average generalization of the model [98].

As such, the selected models to do the semantic segmentation of the data were trained by cross-validation. As mentioned before, the folds should have the same number of patients and all classes must be represented in each fold. In total, there are 44 available patients. Table 14 shows the patients' IDs for each fold:

Table 14. Description of patients by fold, being each fold divided by the 4 domains

Fold	Patients
Fold 1	Invivo robotic ➔ 01, 04
	Exvivo robotic ➔ 03
	Invivo laparoscopic ➔ 04, 14, 17,20
	Exvivo laparocopic ➔ 01, 07
Fold 2	Invivo robotic ➔ 03, manual_sampling, manual_sampling 2
	Exvivo robotic ➔ None
	Invivo laparoscopic ➔ 01, 05, 09, 18, 23
	Exvivo laparocopic ➔ 02, 05
Fold 3	Invivo robotic ➔ 02, 05
	Exvivo robotic ➔ 04, 05
	Invivo laparoscopic ➔ 03, 06, 13, 21
	Exvivo laparocopic ➔ 03
Fold 4	Invivo robotic ➔ None
	Exvivo robotic ➔ 01, 05
	Invivo laparoscopic ➔ 02, 07, 08, 11, 16, ClippingSnippets
	Exvivo laparocopic ➔ 03

These folds represent the validation folds so, for instance, during the first train, the model will assess its performance using patients from fold 1 for validation and train with the remaining folds. The main objective of this work was the application of a network with domain unaware, so it is not mandatory that all domains were represented in each fold. It is worth to mention that some patients will be only used for training purposes, namely: invivo laparoscopic 12, 15, 19, 22, 24; exvivo laparoscopic 04, 06, 08.

7.3 Results

This sub-chapter exposes the metrics achieved by the selected models, and the comparison between them. Experiment results must clarify three points: which pretrained backbone has the best performance in the studied dataset; if the implementation of attention in U-Net can provide better insights than other models; what generative models for synthetic data augmentation improves data diversity towards a more robust decision.

In order to answer the previous questions, the analysis was first directed to the instrument parts. Following step was migrate to the instrument type, where the networks performance was also performed.

7.3.1 Instrument parts

The complexity of the problem increases with the number of classes. All models were trained using one RTX3090 GPU with 24GB and evaluated in terms of dice and normalized Hausdorff distance (NHD). NHD is mostly often used to calculate the distance between two-point sets. Equations (7.1) and (7.2) mathematically the dice and NHD, respectively:

$$dice = \frac{TP}{2 \times TP + FP + FN} \tag{7.1}$$

$$NHD = min\{NOMD(\vec{X}_i, \vec{Y}_j), NOMD(\vec{Y}_i, \vec{X}_j)\} \tag{7.2}$$

Where TP is true positive, FP false positive and FN is false negative. NOMD, being the combination of two distances (cosine and Euclidian), that guarantees that NHD will be 1 (finite number) if the distance is infinite for one class that appears in the ground truth but not in the prediction, or vice-versa [99]. All images, before being injected into the model were subject to a histogram equalization. This method is a smoothing technique that allows the highlight of some details that were previously [100]. The

image intensities are evenly distributed on the full range, improving the image quality which benefits object perception.

For each model, the fold with the best validation score is presented. Table 17 shows the number of parameters and the computational time required to train each fold:

Table 15. Parameters and time consumption by each selected model

Models	Number of parameters	Training time (h)
U-Net + VGG16	4118932	3.717
U-Net + ResNet101	61943564	3.759
U-Net + VGG16 + Attention	42248464	3.47
U-Net +ResNet101 + Attention	62295096	3.515
Nested U-Net	58630852	4.774
Transformer U-Net	165166436	6.561

Transformer meets U-Net model has a high training time, which could be explained by the number of parameters of the model. The transformer, as mentioned in section 5.3, has a large number of multi-head self-attention modules, which originates a large number of parameters. Table 16 shows dice and NHD for the different models.

Table 16. Dice and NHD of each model by class for the validation dataset

Models	Pretrained Backbone	Dice				NHD			
		Wrist	Shaft	Jaw	Avg.	Wrist	Shaft	Jaw	Avg.
U-Net	VGG16	**0.5058**	0.738	0.5635	0.6024	0.28	0.12	0.22	0.21
U-Net	ResNet101	0.4762	0.7388	0.5395	0.5848	0.25	0.10	0.20	0.18
U-Net + Attention	VGG16	0.4182	**0.8161**	**0.6775**	**0.6372**	**0.22**	**0.09**	**0.18**	**0.16**
U-Net + Attention	ResNet101	0.3907	0.7981	0.6099	0.5997	0.24	0.10	0.19	0.18
Nested U-Net	VGG16	0.4082	0.8087	0.6364	0.6178	0.32	0.12	0.23	0.22
Transformer U-Net	VGG16	0.4011	0.7898	0.6206	0.6038	0.32	0.14	0.28	0.25

In order to have a valid conclusion, these models were trained in the same conditions. Each model was trained for 50 epochs with an initial learning rate of 1E-4. The adopted optimizer was the Adam with a learning decay with a factor of 10 on the epochs 25, 35, and 45. While training, these models were validated by cross-validation, so in the end, there were 4 different trained sub-models for the same network structure. Table 16 presents the values of the best-performing fold on average NHD and average dice. Overall, the best fold was fold 1, despite the fold 3 could be a possibility due to its high value in the wrist class, for all models.

The analysis was divided into two parts: understand which backbone has the best performance; and understand which is the best alternative. The U-Net model was trained with a VGG16 and a ResNet101 as the backbone. These encoders were pretrained on the ImageNet dataset [36] and no layers were frozen. VGG16 backbone outperformedResNet101 by 0.0172 in dice average. This improvement may be related to its simpler structure and the relatively small number of classes. Therefore, VGG16 was the selected backbone for the subsequent models. By looking into the correspondent examples of these models in Figure 25 and Figure 26 it can be seen that outer borders appear to be less affected by adjacent pixels that do not belong to the object itself. By using the gate attention in the U-Net an average dice score of 0.6372 was obtained, surpassing the original U-Net by 0.0348. The score of 0.6372 also beat the Nested U-Net, by 0.0194, and the Transformer U-Net, by 0.0334. Now taking a closer look at NHD, which could give a better perception since it is related to the shift of the predicted segmentation from the ground-truth, the NHD of baseline U-Net had a better performance than Nested U-Net and Transformer U-Net, with a difference of 0.01 and 0.03, respectively. This occurrence reinforces the conclusions taken from the dice metrics. However, the best backbone for the U-Net was ResNet101, which did not happen in the dice. The ResNet101 achieved a value of 0.18, while VGG16 reached 0.21. Following the reasoning, it is expected that ResNet101 overtakes the VGG16 in U-Net with gate attention. The previous statement did not occur, the ResNet101 stagnated, while VGG16 decreased by 0.16, catching the ResNet101. Observing the segmentations, represented in Figure 25 and Figure 26, the model of U-Net with the attention gate

achieved a better result. This model, in these two segmentations, obtained a segmentation map closer to the ground truth in all three classes. The implementation of the attention gate allowed the generation of segmentation with smoothed edges. The transformer meets U-Net model did not achieve the best results, contrary to what was observed using other types of data. In part, this could be due to the way patches are defined in this structure, i.e., although the same objects with the same parts are used while performing the same procedure in another patient, it cannot be guaranteed that those instruments will be in the same position and orientation. These variations are unlikely to be encompassed using datasets with a few dozen patients.

Overall, based on the dice and NHD values, and segmentation maps the U-Net with gate attention with the VGG16 as a backbone is the best combination to solve the segmentation of these three parts in surgical objects.

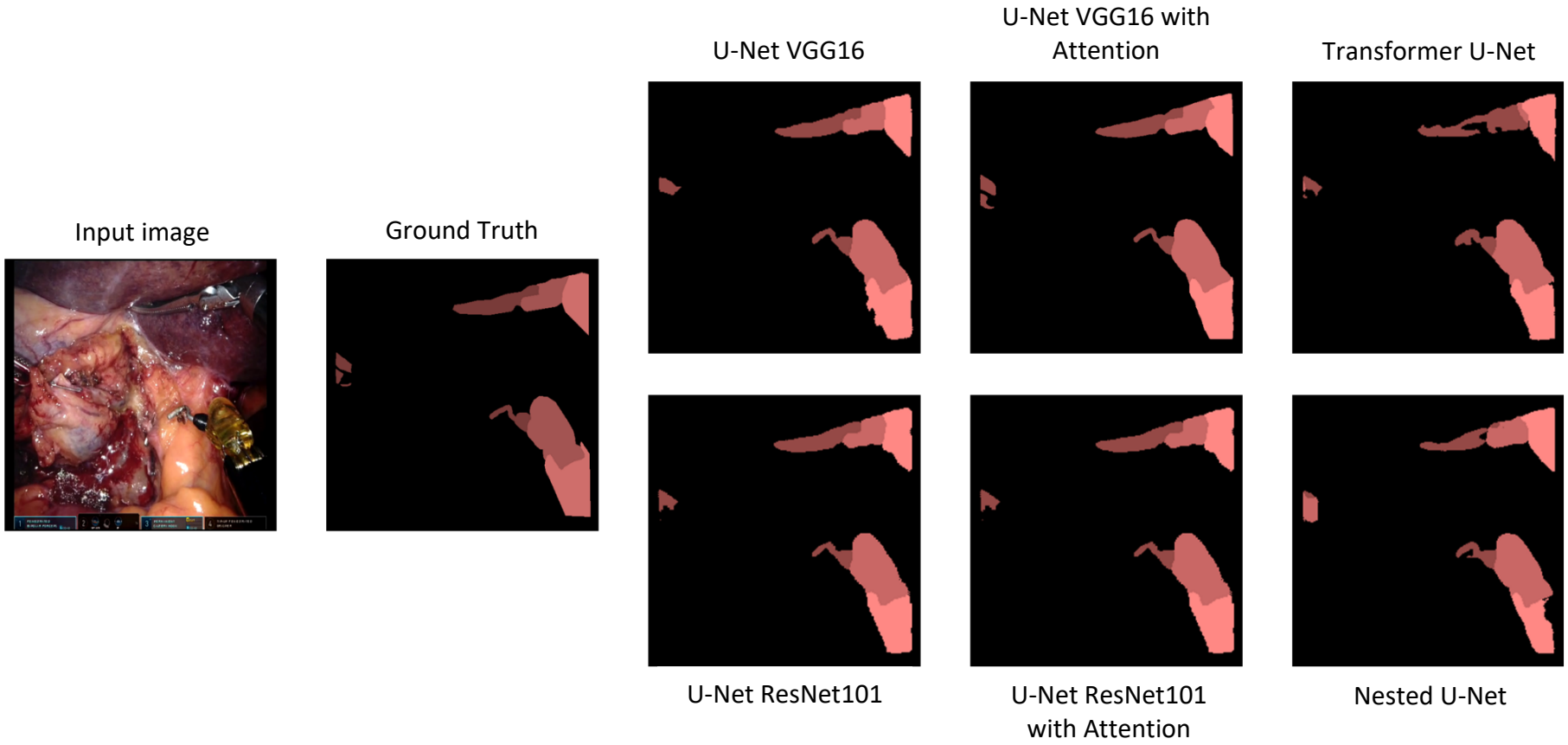

Figure 25. Semantic segmentation of instrument parts of patient 01, frame 80, on the invivo_robotic domain.

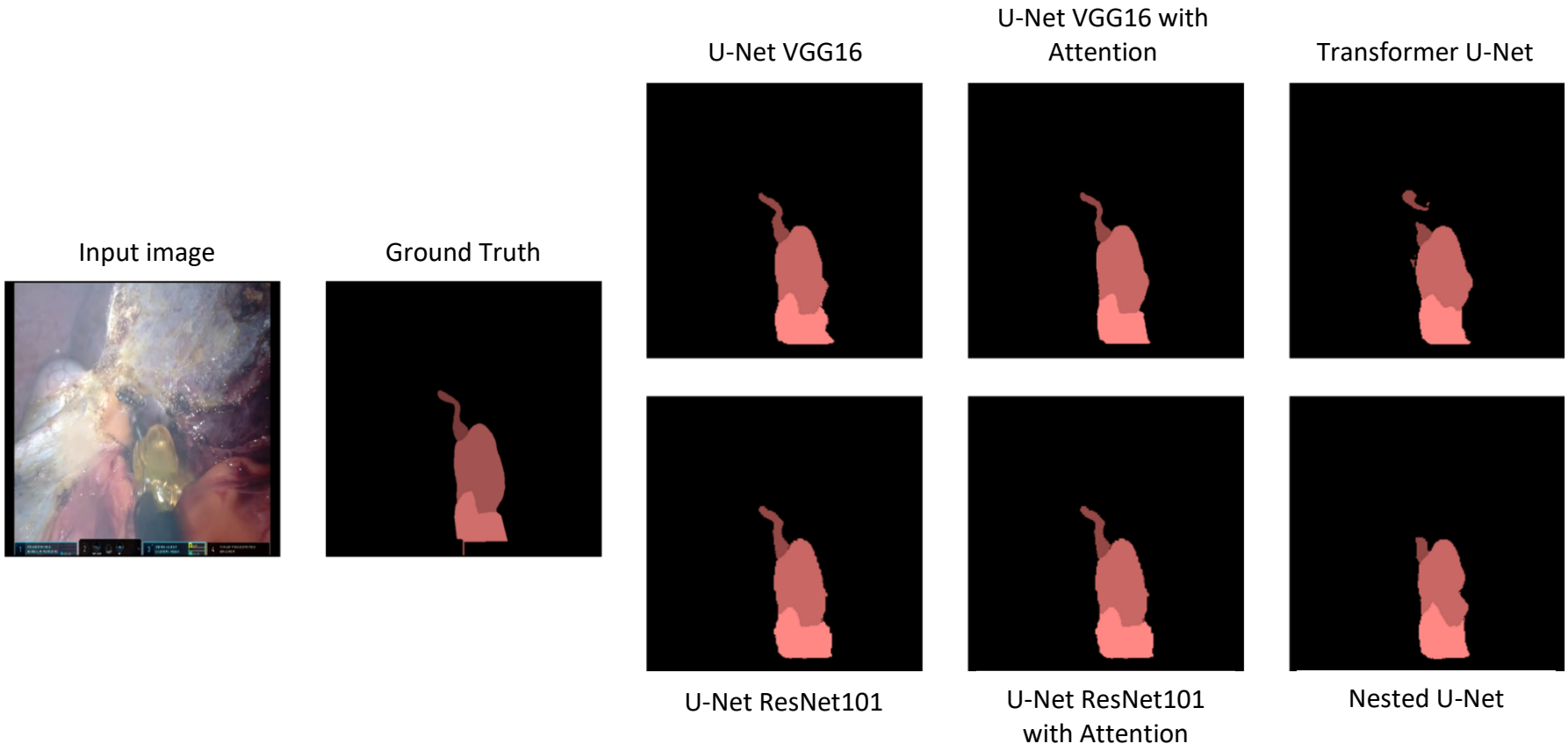

Figure 26. Semantic segmentation of instrument parts of patient 01, frame 60, on the invivo_robotic domain

Ideally, overfitting should be evaluated by looking for the validation metrics, but also observing the model behavior in a new unseen test dataset. The test dataset should have different patients of the train and validation data. Often, due to the lack of data, which is the current case, it is not possible to divide the dataset into three distinct sets. Since this dataset was from the challenge "*Surgical Instrument Multi-Domain Segmentation Challenge (MICCAI 2023)*" [97] and although the test set was not publicly available, it was possible to test one of the models. The results are shown in Table 17:

Table 17. Scoring results for domain-unware part segmentation [97] in the test set

Model	**Dice Avg.**	**NHD Avg.**
U-Net+attention+VGG16	0.64	0.17

As can be seen, both dice and HD are similar to the validation values shown in Table 16. As such, this model did not overtrain, therefore, overfitting was avoided, which is common when U-Net is used in medical images. It is worth mentioning that the U-Net model with attention gate and VGG16 achieved 3rd place in this challenge.

However, results can be improved by performing data augmentation. Synthetic data can replicate features presented in the original images or increase data diversity, providing more valuable information to the model. In the following discussion, two different approaches to generating synthetic images are discussed, GANs and diffusion models. Table 18 shows the scoring values achieved by aforementioned data augmentation techniques.

Table 18. Metric values of data augmentation on U-Net with attention gate

Synthetic data	Dice				NHD			
	Wrist	**Shaft**	**Jaw**	**Avg.**	**Wrist**	**Shaft**	**Jaw**	**Avg.**
CoCosNet-v2	**0.5831**	**0.7838**	**0.7094**	**0.6921**	**0.21**	**0.009**	**0.17**	**0.13**
SPADE	0.5094	0.7076	0.4782	0.5651	0.27	0.15	0.24	0.22
Diffusion Model	0.5562	0.7666	0.6628	0.6619	0.23	0.11	0.18	0.17

The generative models were evaluated by how they influence predictions using the model selected in the previous step, the U-Net model with attention gate blocks trained following the same strategy. Each generative model generated synthetic images that should be similar but diverse from the originals, increasing the dataset twice. Even though the stopping criteria for both GAN and Diffusion model training is not a well-defined method, they were trained to reach the minimum loss, where for the GANs is the Nash equilibrium and for the Diffusion Model is the maximum likelihood. Achieving the minimum loss function presupposes that the model can produce data similar to the observed sample. This is not ensured because there is no evidence that the network is not trapped in a local minimum. As can be seen in Table 18, data augmentation increases the performance of the model. The best generative model was CoCosNet-v2, raising the average dice by 0.0549 (0.6921), followed by the Diffusion Model with a value of 0.6619. SPADE network, unlike the previous ones, could not increase the dice scoring, on the contrary, it hinders the model performance. It is visible in Figure 27 that when the frame was smoggy, the U-Net without data augmentation had difficulty on the instrument's identification. For data augmentation, the network could be able to identify very clearly the instrument parts. Figure 28 represents another problem, when facing with several instruments in a frame identify all of them fails, contrarily what happens for data augmentation. SPADE based data augmentation underperforms dice and NHD metrics.

The CoCosNet-v2 generates new samples taking into account a reference image. This approach carries several advantages in small datasets. As all the existing images are

used patterns poorly represented do not vanishes, they maintain its relative value of representability. Even if data distribution is poor modelled a good seed guaranties some increasing in diversity which represents an improving for the dataset. On the other hand, the Diffusion Model due to its ability to create images with high diversity, made abnormal tools, i.e., built instruments composed of the corresponding parts but with a combination that does not belong to any reference object which suggests poor learned distributions and no way to deal with it. The SPADE network, due to lack of data and due to the diversity of backgrounds, where there were at least 4 distinct backgrounds, could not generate realistic images, explaining its performance. Some examples can be seen in Data augmentation sub-section.

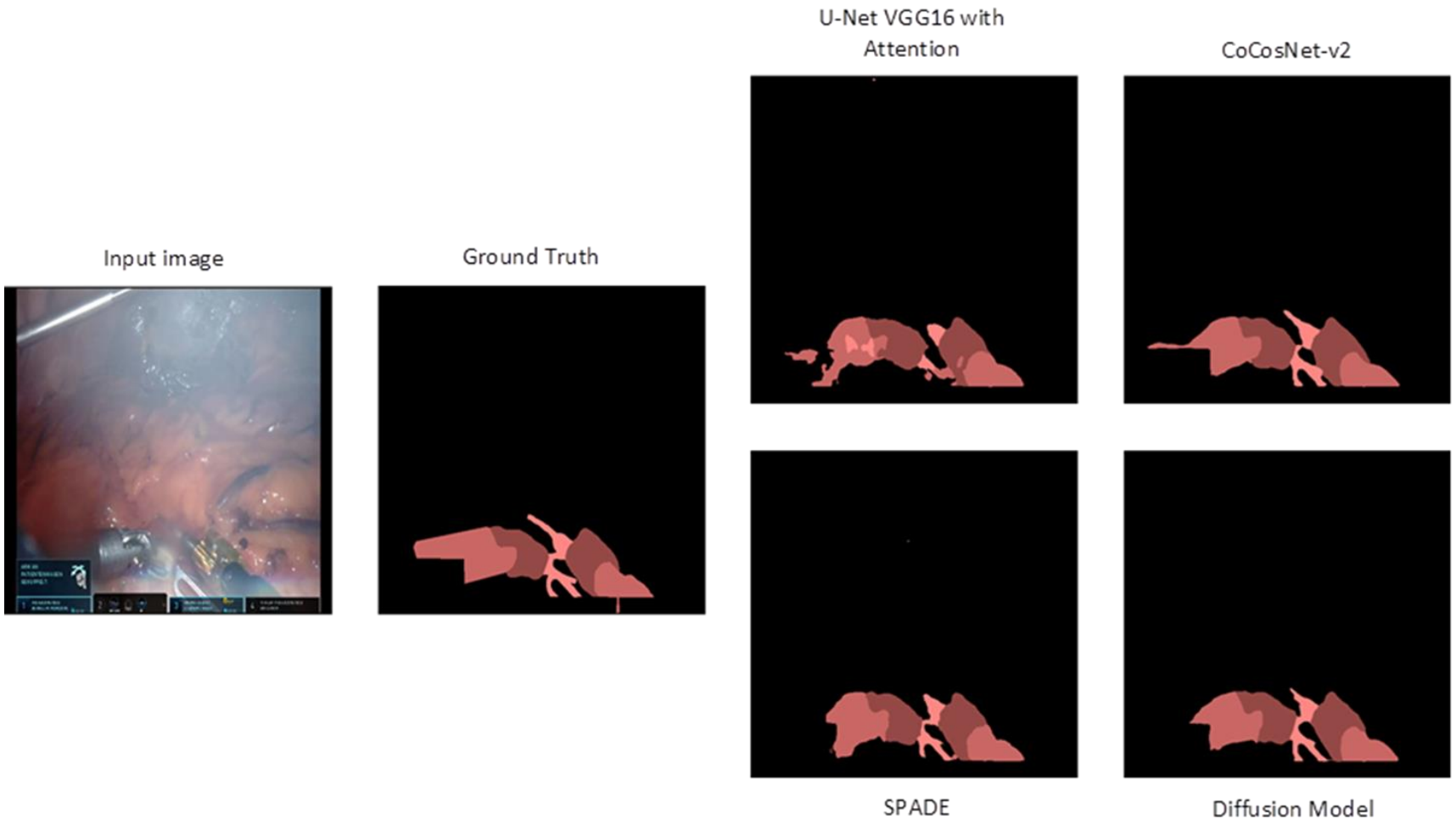

Figure 27. Semantic segmentation of instrument parts with data augmentation of the patient 01, frame 98, on the invivo_robotic domain.

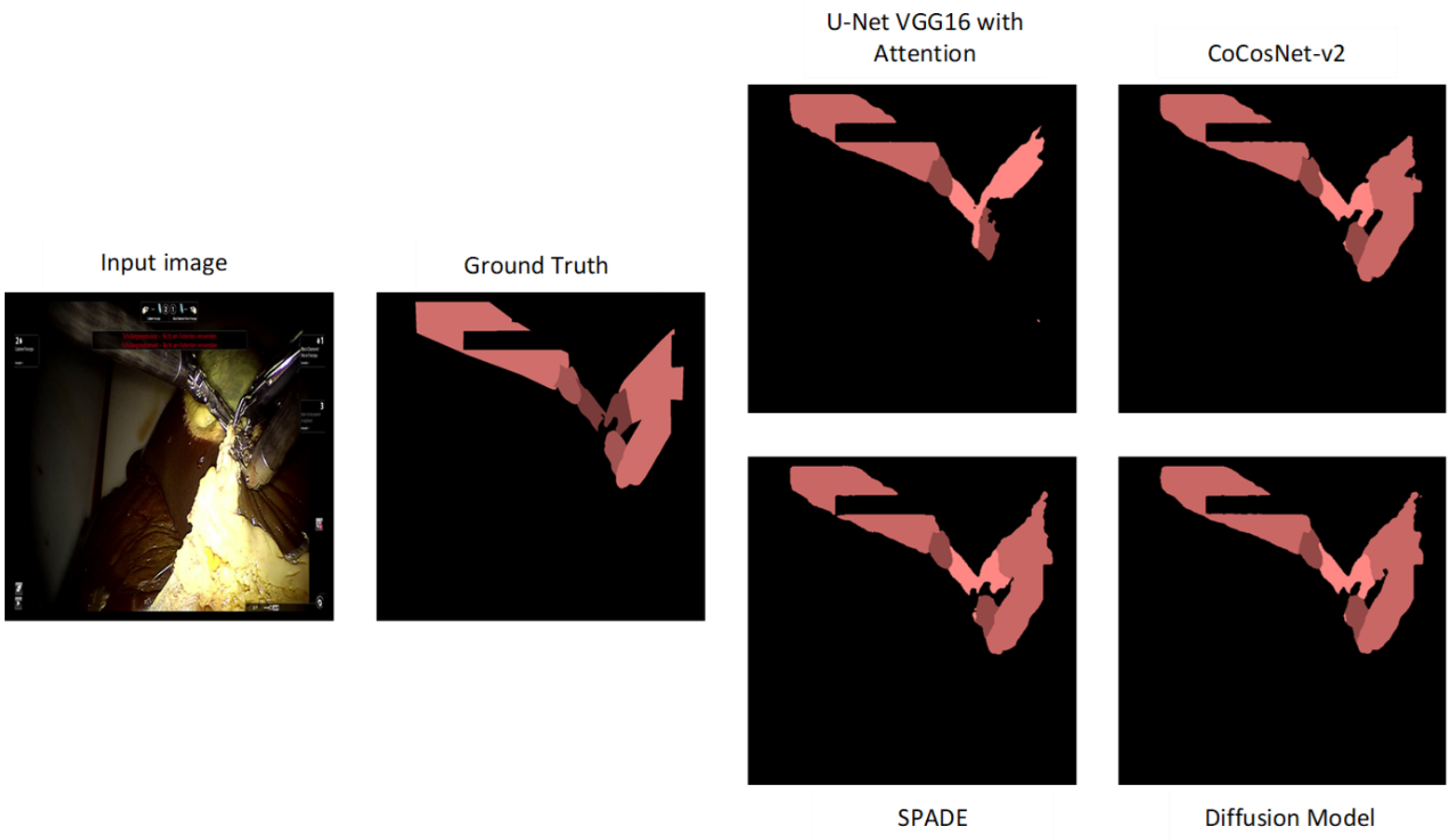

Figure 28. Semantic segmentation of instrument parts with data augmentation of the patient 03, frame 67, on the exvivo_robotic domain.

7.3.2 Instrument type

Increasing number of classes makes the problem more complex because the model has more difficulties looking for the global minimum. Often, due to poor or even lack of representation of some classes, the model is stuck on local minima and can not jump to the best minimum. Such as in the instrument part, in this section all the models were trained using a RTX3090 with 24GB of memory.

The main objective of this work was finding a model with the ability to answer the problem of segmentation of instrument type. As the metric values in Table 16 led to the selection of the U-Net with the gate attention model, the remaining models were discarded, except the U-Net, which is used as a baseline model. For this specific problem, the mAP was the chosen metric for the challenge "*Surgical Instrument Multi-Domain Segmentation Challenge*" [97]. It was not proposed a model to participate in this challenge, so unlike the instrument parts, there was not test scoring values. However, for comparation purposes, Table 19 illustrates the scores achieved by the teams:

Table 19. Scoring values achieved by each team

Team	*mAP (%)*
Team 1	3
Team 2	25
Team 3	30
Team 4	5
Team 5	1

As shown Table 20, the mAP values are not promising, due to the complexity of the problem and the lack of data. Although the Undefined instrument is a class while training, it was not considered by the organization to the calculation of the mAP.

Table 20. Dice and mAP values achieved for each class in instrument type

Model	Synthetic data	Dice										mAP(%)
		C	CI	G	HI	Sc	SB	SI	T	UI	Avg.	
U-Net	None	0.1484	0.1882	0.4467	0.1231	0.1200	0.0038	0.0040	0.0038	0	0.1153	1.25
U-Net + Attention	None	**0.2237**	**0.1976**	**0.5918**	**0.2572**	0	**0.0053**	0.0084	**0.0062**	0	0.1434	2.125
U-Net + Attention	CoCosNet-v2	0.1854	0.1878	0.4618	0.2037	0.3513	0.0051	0.0082	0.0050	0	0.1565	8.41
U-Net + Attention	Diffusion Model	0.2192	0.1883	0.5566	0.2015	**0.3678**	**0.0053**	**0.0107**	0.0052	0	**0.1727**	**9**
U-Net + Attention	SPADE	0.0423	0.1534	0.4327	0.1444	0.3374	0.0034	0.0005	0.0045	0	0.1242	1.95

All models were trained in the same initial hyperparameters for 60 epochs with an initial learning rate of 1E-2. Unlike the previous one, the adopted optimizer was SGD, due to its better results comparing with Adam. The learning rate decay was scheduled to 30, 40 and 50 epochs. Such as in the previous problem, the histogram equalization was applied in each image.

Observing the values from Table 20 the best approach of data augmentation was Diffusion Model, increasing the baseline value, in average by 0.03. The CoCosNet-v2 did not give to the model the best metrics, only growing 0.0131. The SPADE kept its behavior, biasing the metrics. This last mentioned point, supports that the quality of generated images has impact on the network, when it is learning patterns from an image. Taking a closer look to the mAP, the U-Net had an extremely small value, only 1.25%, which is the same value reached by the last team. While the U-Net plus the gate attention increases this value to 2.125. The result was promising, however it was far from the first place. The application of data augmentation, except with SPADE, increases the mAP metrics. The best was achieved by the Diffusion model, going in coherence with dice metric, getting 9%. The CoCosNet-v2 reached a mAP of 8.41, allowing it competing with the Diffusion Model. The worst data segmentation was U-Net VGG16 with attention block with SPADE data augmentation. In Figure 29 is visible semantic maps with classes that are not presented in the ground truth. The baseline model with attention gate without data augmentation shaped an instrument more similar to the ground truth than the original U-Net, only sinning in putting a mismatch class. The CoCosNet-v2 could not improve this problem. The Diffusion Model increase the performance not only discarding the error class, but also shaped an even better instrument type than the U-Net without data augmentation. On the other hand, Figure 30 illustrates a very evidently case because the data augmentation not only allowed the identification of the correct class, but also mold a better surgical tool. The U-Net with attention could segment better the top instrument, unlike the baseline network.

According to these results, it is always better applying data augmentation based in generation of synthetic data.

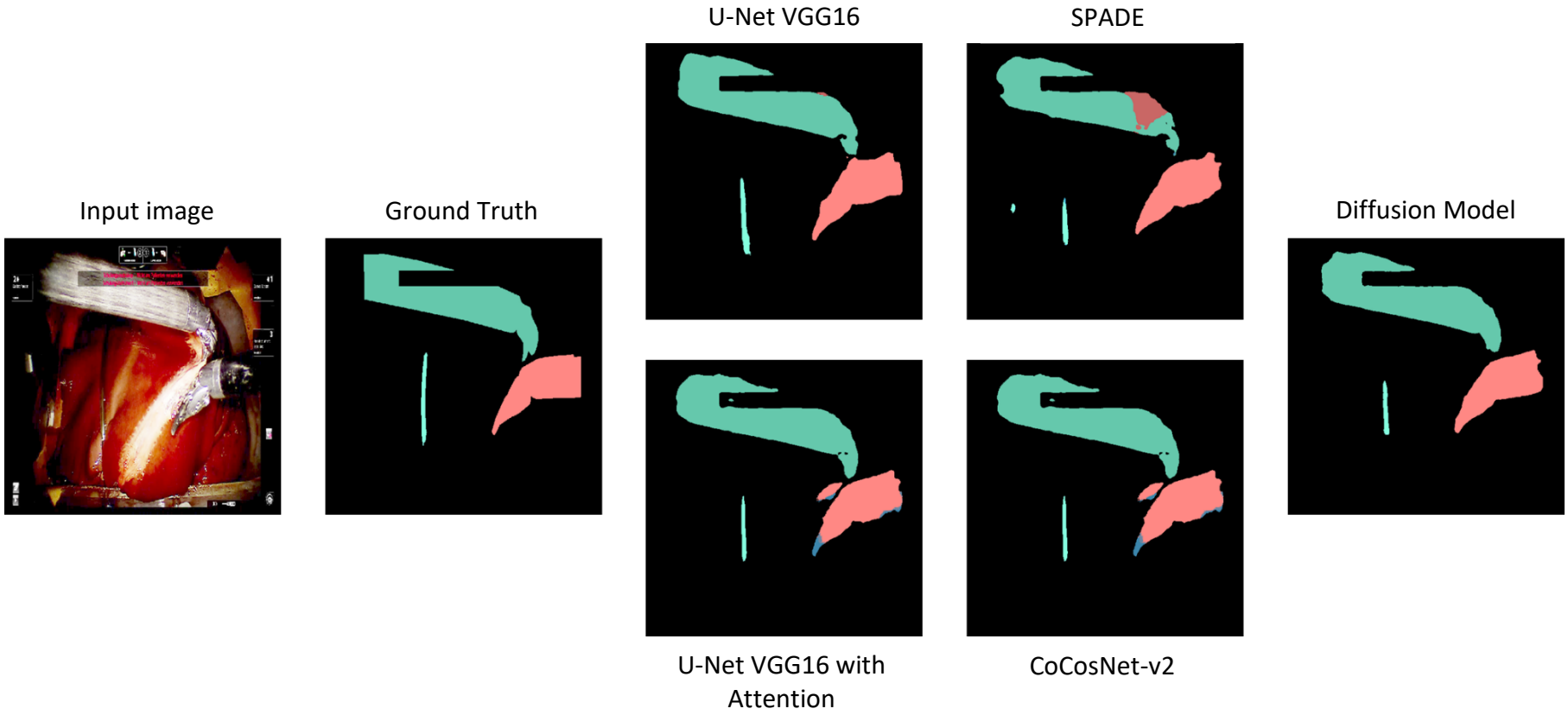

Figure 29. Semantic segmentation of instrument type with and without data augmentation of the patient 01, frame 21, on the exvivo_robotic domain.

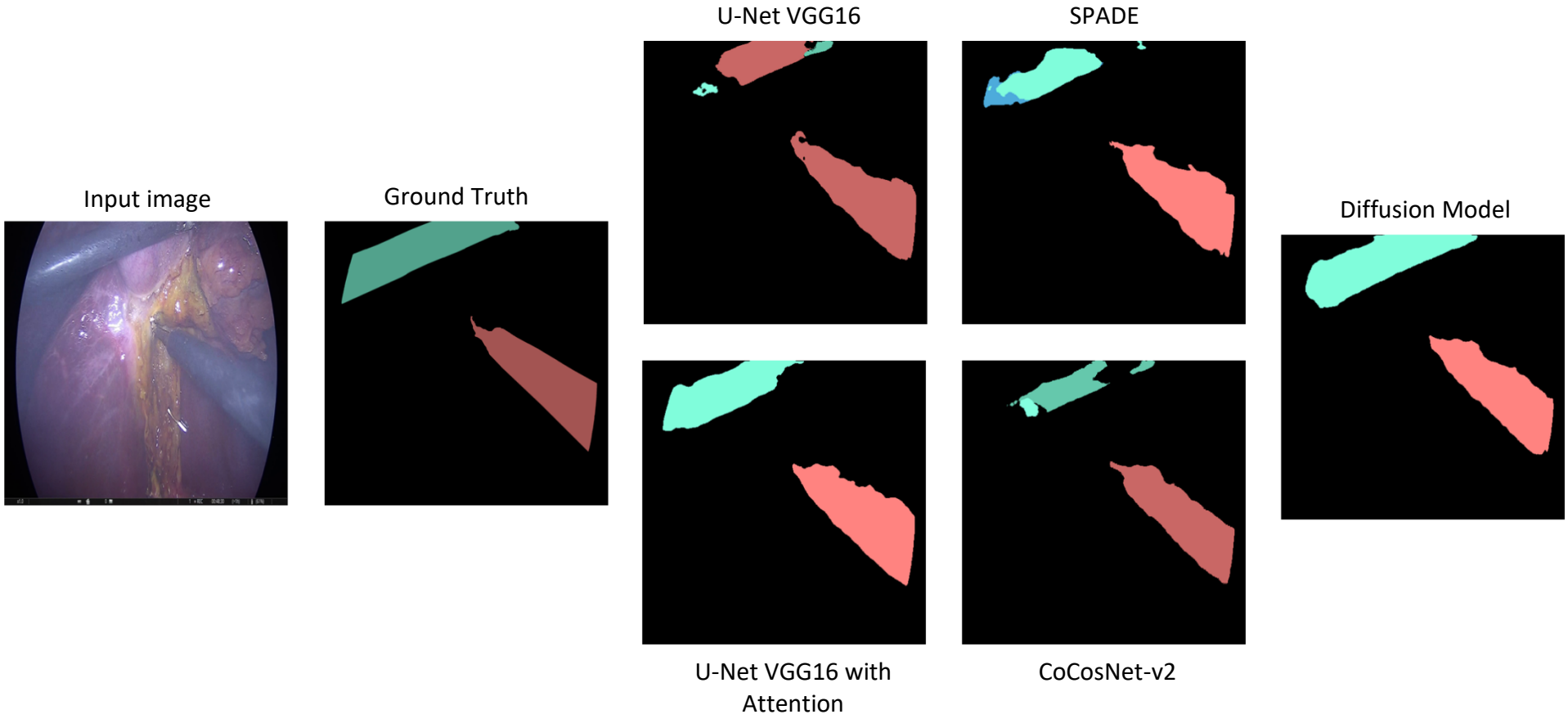

Figure 30. Semantic segmentation of instrument type with and without data augmentation of the patient 02, frame 24, on the invivo_laparoscopic domain.

7.3.3 Data augmentation

In this sub-section, in Figure 32 and

Figure 33, it is shown a sample of each domain from the three studied generative networks. We can see that due to the reference image, the CoCosNet-v2 could generate a very realistic and very similar image to the original one. Although the reference image helps mitigating the lack of data, it can not inject data diversity. This network usually collapses producing images too similar to the real ones. The other GAN, the SPADE, could not converge, generating images with noisy backgrounds. This is an example of destabilization of GAN parameters, where the generative network never converges. The Diffusion Model does not have the problem of imbalance because is not an adversarial network and due to its strategy of learning can inject diversity in the generated image. The model, based on the Markov chain, mapping the mean and the covariance of the noise can recover the original image with a new data distribution in the classes present in the semantic map. However, this network can not guarantee that the generated image has clinical context, i.e., the generated instruments could not make any sense and does not exist in the practice. An example of this statement is Figure 31:

Figure 31. Synthetic image from Diffusion Model without clinical context.

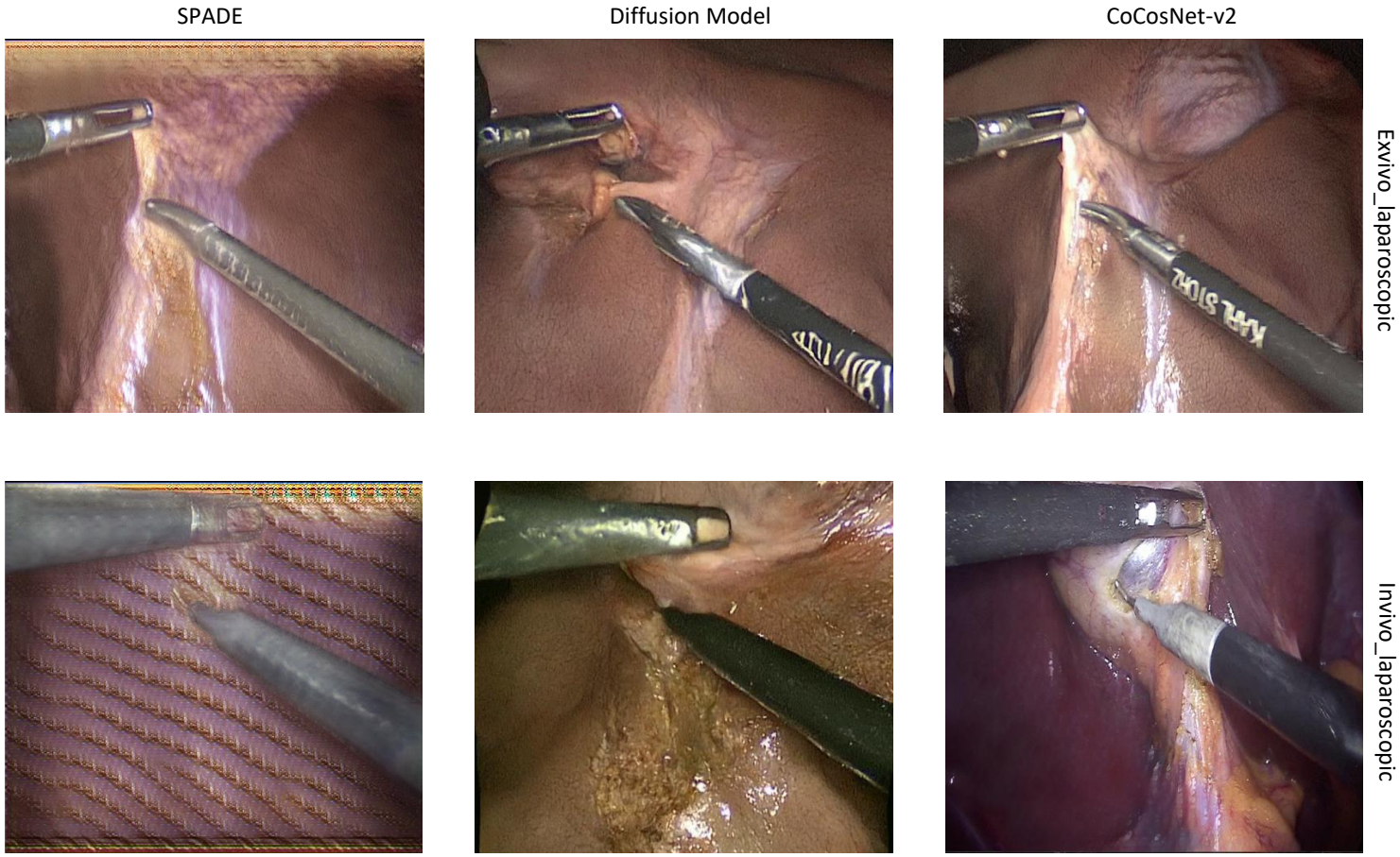

Figure 32. Sythetic data from the three generative models on exvivo_laparoscopic and invivo_laparoscopic domain.

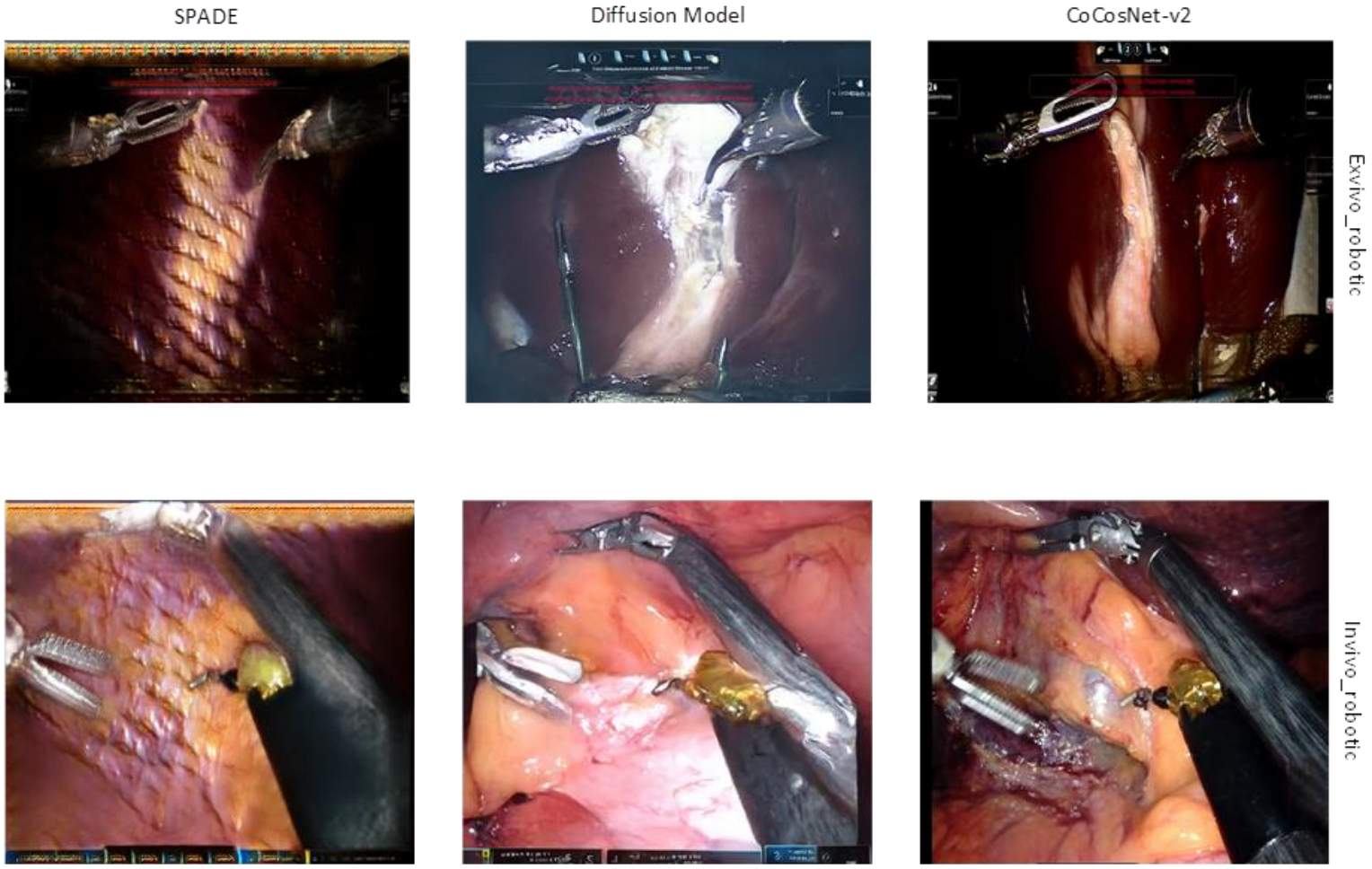

Figure 33. Sythetic data from the three generative models on exvivo_robotic and invivo_robotic domain.

7.4 Conclusion and future work

The robotic-assisted surgery is growing due to its advantages compared to the open surgery. However, there are some limitations, such as the extensive physician formation program and the lack of tactile stimulations.

The main goal of this dissertation was to propose a method to automatically segment the instrument parts and type following three research lines. First the search for models that can segment instrument parts and type. The other line was how to improve the baseline model and the final was what type and how data augmentation can be better, taking into account the problem.

The selected model, the U-Net, could generate a semantic map of instrument parts and instrument type. With the employment of the attention gate, the network, in the instrument parts, could identify better the different classes and even define more smoothly the instrument borders. One of issues with the baseline model was the appearance of holes in the semantic map, but with the attention block that problem was mitigated. Although the remarkable performance achieved by the proposed network, there is still some work to do. The next path should be to search for multi-task models, with more robustness and multisensorial. A possibility is the Mask2Former [101] model that can create three different segmentation maps (semantic, instance and panoptic). The instrument type is a complex problem due to the diversity of instruments that is necessary to identify and due to the lack of data. However, the baseline model with attention blocks achieved good metrics, allowing the standard network to identify more clearly the different surgical tools.

Data augmentation, especially synthetic data is a solution that could help some lack of data problems. GANs are perhaps the most used type of networks, however, such models do not have well defined stopping criteria and consequently the model never converges. This problem propagates to the diffusion model, but unlike the GAN, this model does not look for a Nash equilibrium, instead looks after to map the mean and the covariance of Gaussian noise. If the diffusion model does not train for a certain period of time, it can not capture the nonlinear nature of the real world, whereas if

training for too long the denoising algorithm fails on injecting diversity on the generated images. The best data augmentation model for surgical tool parts was CoCosNet-v2, while for instrument type was Diffusion Model. The main issue of Diffusion Model in the instrument parts was the lack of clinical context, explaining worst values than the CoCosNet-v2. Diffusion Model has a big potential because is a generative network that learn the data by its distribution, but instead to start by the latent space, injects step by step noise. This method of learning allows it to generate images with high diversity. However, there is no measurement in the loss function that control the diversity and the reality of generated images at the same time. In this case, the next steps should be creating a trade-off between diversity and realistic generated images.

Despite the semantic map quality from the proposed model some improvements should be done in order to become a robust complete system. For the next few works, the best approaches, in this area, should be the exploration of multi-modals networks. The particularity of multi-modal models is their combination of multiple sources of data from different types, therefore allowing the model to learn more cross-data dependencies and solve multiple tasks at once. It is expected that this can give the right tools for segmentation of instrument parts and type. The transformers have a big potential for semantic segmentation, however for some application with lack of data there is still a lot of work to do.

Bibliography

[1] N. Enayati, E. De Momi, and G. Ferrigno, “Haptics in robot-assisted surgery: Challenges and benefits,” *IEEE Rev. Biomed. Eng.*, vol. 9, pp. 49–65, 2016, doi: 10.1109/RBME.2016.2538080.

[2] G. I. Barbash and S. A. Glied, “New Technology and Health Care Costs — The Case of Robot-Assisted Surgery,” 2010.

[3] Tiago Barreto, “Cirurgia Robótica: uma comparação entre Prostatectomia Laparoscópica Assistida por Robô e Prostatectomia Radical Laparoscópica XV Curso de Mestrado em Gestão da Saúde,” p. 56, 2021.

[4] T. W. Hokken *et al.*, “Validation of a Three-Dimensional Computed Tomography Reconstruction Tool for Aortic Valve Calcium Quantification,” *Struct. Hear.*, vol. 7, no. 2, p. 100122, 2023, doi: 10.1016/j.shj.2022.100122.

[5] S. Borkman *et al.*, “Unity Perception: Generate Synthetic Data for Computer Vision,” 2021, [Online]. Available: http://arxiv.org/abs/2107.04259

[6] H. Abu Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, “Augmented Reality Meets Computer Vision: Efficient Data Generation for Urban Driving Scenes,” *Int. J. Comput. Vis.*, vol. 126, no. 9, pp. 961–972, 2018, doi: 10.1007/s11263-018-1070-x.

[7] L. K. Jacobs, V. Shayani, and J. M. Sackier, “Determination of the learning curve of the AESOP robot,” *Surg. Endosc.*, vol. 11, no. 1, pp. 54–55, 1997, doi: 10.1007/s004649900294.

[8] J. P. Ruurda, T. J. M. V. Van Vroonhoven, and I. A. M. J. Broeders, “Robot-assisted surgical systems: A new era in laparoscopic surgery,” *Ann. R. Coll. Surg. Engl.*, vol. 84, no. 4, pp. 223–226, 2002, doi: 10.1308/003588402320439621.

[9] S. Kalan *et al.*, “History of robotic surgery,” *J. Robot. Surg.*, vol. 4, no. 3, pp. 141–147, 2010, doi: 10.1007/s11701-010-0202-2.

[10] F. Pugin, P. Bucher, and P. Morel, “History of robotic surgery: From AESOP® and ZEUS® to da Vinci®,” *J. Visc. Surg.*, vol. 148, no. 5, pp. e3–e8, 2011, doi:

10.1016/j.jviscsurg.2011.04.007.

[11] D. Gerhardus, “Robot-Assisted Surgery: The Future Is Here,” *J. Healthc. Manag.*, vol. 48:4, 2003.

[12] K. Chang, F. Gokcal, and O. Y. Kudsi, “Robotic Biliary Surgery,” *Surg. Clin. North Am.*, vol. 100, no. 2, pp. 283–302, 2020, doi: 10.1016/j.suc.2019.12.002.

[13] W. S. Helton and S. Ayloo, “Technical Aspects of Bile Duct Evaluation and Exploration: An Update,” *Surg. Clin. North Am.*, vol. 99, no. 2, pp. 259–282, 2019, doi: 10.1016/j.suc.2018.12.007.

[14] R. Bustos *et al.*, “Robotic hepaticojejunostomy: surgical technique and risk factor analysis for anastomotic leak and stenosis,” *Hpb*, vol. 22, no. 10, pp. 1442–1449, 2020, doi: 10.1016/j.hpb.2020.02.007.

[15] P. Wang, Y. J. Su, and C. Y. Jia, “Current surgical practices of robotic-assisted tissue repair and reconstruction,” *Chinese J. Traumatol. - English Ed.*, vol. 22, no. 2, pp. 88–92, 2019, doi: 10.1016/j.cjtee.2019.01.003.

[16] J. Hoeppner, “Robotic cancer surgery,” *Cancers (Basel).*, vol. 13, no. 19, 2021, doi: 10.3390/cancers13194931.

[17] M. Gómez Ruiz, M. Lainez Escribano, C. Cagigas Fernández, L. Cristobal Poch, and S. Santarrufina Martínez, “Robotic surgery for colorectal cancer,” *Ann. Gastroenterol. Surg.*, vol. 4, no. 6, pp. 646–651, 2020, doi: 10.1002/ags3.12401.

[18] E. J. Park and S. H. Baik, “Robotic Surgery for Colon and Rectal Cancer,” *Curr. Oncol. Rep.*, vol. 18, no. 1, pp. 1–8, 2016, doi: 10.1007/s11912-015-0491-8.

[19] D. Bausch and T. Keck, “Minimally invasive surgery of pancreatic cancer: Feasibility and rationale,” *Visc. Med.*, vol. 34, no. 6, pp. 440–443, 2018, doi: 10.1159/000495324.

[20] Dr. Bryan Burt, “Q&A: What you should know about VATS lobectomy,” *Baylor College of Medicine*, 2023. https://blogs.bcm.edu/2015/06/23/qa-what-you-should-know-about-vats-lobectomy/

[21] P. Linsky and B. Wei, “Robotic lobectomy,” *J. Vis. Surg.*, vol. 3, pp. 132–132, 2017, doi: 10.21037/jovs.2017.08.12.

[22] M. Adams and G. Doherty, “Conventional thyroidectomy,” *Oper. Tech.*

Otolaryngol. - Head Neck Surg., vol. 20, no. 1, pp. 2–6, 2009, doi: 10.1016/j.otot.2009.03.002.

[23] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. 2012. doi: 10.1007/978-1-4614-6940-7_17.

[24] Ian Goodfellow, Yoshua Bengio, and A. Courville, *Deep learning*. 2016. doi: 10.1007/978-981-13-9113-2_16.

[25] Mahesh Batta, "Machine Learning Algorithms - A Review," *Int. J. Sci. Res.*, no. October, 2020, doi: 10.21275/ART20203995.

[26] J. A. A. da S. R. Pinto, "Automatic Prediction of Ischemic Stroke from MRI images using Deep Learning," Universidade do Minho, 2020.

[27] Nuno Renato Azevedo de Freitas, "Automatic Diagnosis of Urinary Tract Abnormalities (ADUTA)," Universidade do Minho, 2022.

[28] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A Comprehensive Survey of Loss Functions in Machine Learning," *Ann. Data Sci.*, vol. 9, no. 2, pp. 187–212, 2022, doi: 10.1007/s40745-020-00253-5.

[29] R. Y. Sun, "Optimization for Deep Learning: An Overview," *J. Oper. Res. Soc. China*, vol. 8, no. 2, pp. 249–294, 2020, doi: 10.1007/s40305-020-00309-6.

[30] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[31] S. Ruder, "An overview of gradient descent optimization algorithms," pp. 1–14, 2017, [Online]. Available: http://arxiv.org/abs/1609.04747

[32] A. Boudjelal, Z. Messali, and A. Elmoataz, "A Novel Kernel-Based Regularization Technique for PET Image Reconstruction," *Technologies*, vol. 5, no. 2, p. 37, 2017, doi: 10.3390/technologies5020037.

[33] X. H. Liu, C. H. Hsieh, J. Der Lee, S. T. Lee, and C. T. Wu, "A vision-based surgical instruments classification system," *2014 Int. Conf. Adv. Robot. Intell. Syst. ARIS 2014*, pp. 18–22, 2014, doi: 10.1109/ARIS.2014.6871520.

[34] M. J. Primus, K. Schoeffmann, and L. Böszörmenyi, "Instrument classification in laparoscopic videos," *Proc. - Int. Work. Content-Based Multimed. Index.*, vol. 2015-July, pp. 1–6, 2015, doi: 10.1109/CBMI.2015.7153616.

[35] S. Petscharnig and K. Schöffmann, “Learning laparoscopic video shot classification for gynecological surgery,” *Multimed. Tools Appl.*, vol. 77, no. 7, pp. 8061–8079, 2018, doi: 10.1007/s11042-017-4699-5.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 248–255, 2010, doi: 10.1109/cvpr.2009.5206848.

[37] J. Jaafari, S. Douzi, K. Douzi, and B. Hssina, “The impact of ensemble learning on surgical tools classification during laparoscopic cholecystectomy,” *J. Big Data*, vol. 9, no. 1, 2022, doi: 10.1186/s40537-022-00602-6.

[38] Cheolwhan Lee, Yuan-Fang Wang, D. R. Uecker, and Yulun Wang, “Image analysis for automated tracking in robot-assisted endoscopic surgery,” pp. 88–92, 1994, doi: 10.1109/icpr.1994.576232.

[39] J. Climent and P. Mars, “Automatic instrument localization in laparoscopic surgery,” *Prog. Comput. Vis. Image Anal.*, vol. 4, no. April, pp. 123–135, 2004, doi: 10.1142/9789812834461_0007.

[40] S. Deshpande, “Laparoscopic Instrument Localization using a 3-D Time-of-Light/RBG Endoscope,” *J. Am. Chem. Soc.*, vol. 123, no. 10, pp. 2176–2181, 2013, [Online]. Available: https://shodhganga.inflibnet.ac.in/jspui/handle/10603/7385

[41] B. Choi, K. Jo, S. Choi, and J. Choi, “Surgical-tools detection based on Convolutional Neural Network in laparoscopic robot-assisted surgery,” *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 1756–1759, 2017, doi: 10.1109/EMBC.2017.8037183.

[42] A. Jin *et al.*, “Tool detection and operative skill assessment in surgical videos using region-based convolutional neural networks,” *Proc. - 2018 IEEE Winter Conf. Appl. Comput. Vision, WACV 2018*, vol. 2018-Janua, pp. 691–699, 2018, doi: 10.1109/WACV.2018.00081.

[43] A. Vardazaryan, D. Mutter, J. Marescaux, and N. Padoy, “Weakly-supervised learning for tool localization in laparoscopic videos,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11043 LNCS, pp. 169–179, 2018, doi: 10.1007/978-3-030-01364-6_19.

[44] A. Verikas, A. Gelzinis, and M. Bacauskiene, “Mining data with random forests: A survey and results of new tests,” *Pattern Recognit.*, vol. 44, no. 2, pp. 330–349, 2011, doi: 10.1016/j.patcog.2010.08.011.

[45] D. Bouget, M. Allan, D. Stoyanov, and P. Jannin, “Vision-based and marker-less surgical tool detection and tracking: a review of the literature,” *Med. Image Anal.*, vol. 35, pp. 633–654, 2017, doi: 10.1016/j.media.2016.09.003.

[46] A. E. Abdel-Hakim and A. A. Farag, “CSIFT: A SIFT descriptor with color invariant characteristics,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, pp. 1978–1983, 2006, doi: 10.1109/CVPR.2006.95.

[47] Z. Pezzementi, S. Voros, and G. D. Hager, “Articulated object tracking by rendering consistent appearance parts,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 3940–3947, 2009, doi: 10.1109/ROBOT.2009.5152374.

[48] C. Doignon, P. Graebling, and M. De Mathelin, “Real-time segmentation of surgical instruments inside the abdominal cavity using a joint hue saturation color feature,” *Real-Time Imaging*, vol. 11, no. 5-6 SPEC. ISS., pp. 429–442, 2005, doi: 10.1016/j.rti.2005.06.008.

[49] D. Bouget, R. Benenson, M. Omran, L. Riffaud, B. Schiele, and P. Jannin, “Detecting Surgical Tools by Modelling Local Appearance and Global Shape,” *IEEE Trans. Med. Imaging*, vol. 34, no. 12, pp. 2603–2617, 2015, doi: 10.1109/TMI.2015.2450831.

[50] L. C. García-Peraza-Herrera *et al.*, “Real-time segmentation of non-rigid surgical tools based on deep learning and tracking,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10170 LNCS, pp. 84–95, 2017, doi: 10.1007/978-3-319-54057-3_8.

[51] M. Attia, M. Hossny, S. Nahavandi, and H. Asadi, “Surgical tool segmentation using a hybrid deep CNN-RNN auto encoder-decoder,” *2017 IEEE Int. Conf. Syst. Man, Cybern. SMC 2017*, vol. 2017-Janua, pp. 3373–3378, 2017, doi: 10.1109/SMC.2017.8123151.

[52] Z. L. Ni, G. Bin Bian, X. L. Xie, Z. G. Hou, X. H. Zhou, and Y. J. Zhou, “RASNet: Segmentation for Tracking Surgical Instruments in Surgical Videos Using Refined

Attention Segmentation Network," *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 5735–5738, 2019, doi: 10.1109/EMBC.2019.8856495.

[53] X. Wang *et al.*, "PaI-Net: A modified U-Net of reducing semantic gap for surgical instrument segmentation," *IET Image Process.*, vol. 15, no. 12, pp. 2959–2969, 2021, doi: 10.1049/ipr2.12283.

[54] M. Allan *et al.*, "2017 Robotic Instrument Segmentation Challenge," pp. 1–14, 2019, [Online]. Available: http://arxiv.org/abs/1902.06426

[55] V. Iglovikov and A. Shvets, "TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation," 2018, [Online]. Available: http://arxiv.org/abs/1801.05746

[56] L. C. Garcia-Peraza-Herrera *et al.*, "ToolNet: Holistically-nested real-time segmentation of robotic surgical tools," *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2017-Septe, pp. 5717–5722, 2017, doi: 10.1109/IROS.2017.8206462.

[57] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, 2018, doi: 10.1109/TPAMI.2018.2844175.

[58] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *IEEE Access*, vol. 9, pp. 16591–16603, 2015, doi: 10.1109/ACCESS.2021.3053408.

[59] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[61] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11045 LNCS, pp. 3–11, 2018, doi: 10.1007/978-3-030-00889-5_1.

[62] R. Azad, M. Heidari, Y. Wu, and D. Merhof, "Contextual Attention Network:

Transformer Meets U-Net," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 13583 LNCS, pp. 377–386, 2022, doi: 10.1007/978-3-031-21014-3_39.

[63] A. Vaswani *et al.*, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.

[64] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11211 LNCS, pp. 3–19, 2018, doi: 10.1007/978-3-030-01234-2_1.

[65] O. Oktay *et al.*, "Attention U-Net: Learning Where to Look for the Pancreas," no. Midl, 2018, [Online]. Available: http://arxiv.org/abs/1804.03999

[66] J. Schlemper *et al.*, "Attention gated networks: Learning to leverage salient regions in medical images," *Med. Image Anal.*, vol. 53, pp. 197–207, 2019, doi: 10.1016/j.media.2019.01.012.

[67] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *J. Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0197-0.

[68] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," pp. 1–7, 2014, [Online]. Available: http://arxiv.org/abs/1411.1784

[69] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F. Y. Wang, "Generative adversarial networks: Introduction and outlook," *IEEE/CAA J. Autom. Sin.*, vol. 4, no. 4, pp. 588–598, 2017, doi: 10.1109/JAS.2017.7510583.

[70] J. Luo and J. Huang, "Generative adversarial network: An overview," *Yi Qi Yi Biao Xue Bao/Chinese J. Sci. Instrum.*, vol. 40, no. 3, pp. 74–84, 2019, doi: 10.19650/j.cnki.cjsi.J1804413.

[71] Y. Pang, J. Lin, T. Qin, and Z. Chen, "Image-to-Image Translation: Methods and Applications," *IEEE Trans. Multimed.*, vol. 24, pp. 3859–3881, 2022, doi: 10.1109/TMM.2021.3109419.

[72] P. Salehi, A. Chalechale, and M. Taghizadeh, "Generative Adversarial Networks (GANs): An Overview of Theoretical Model, Evaluation Metrics, and Recent Developments," 2020, [Online]. Available: http://arxiv.org/abs/2005.13178

[73] D. Saxena and J. Cao, “Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions,” *ACM Comput. Surv.*, vol. 54, no. 3, 2021, doi: 10.1145/3446374.

[74] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5967–5976, 2017, doi: 10.1109/CVPR.2017.632.

[75] C. Wang *et al.*, “Discriminative Region Proposal Adversarial Network for High-Quality Image-to-Image Translation,” *Int. J. Comput. Vis.*, vol. 128, no. 10–11, pp. 2366–2385, 2018, doi: 10.1007/s11263-019-01273-2.

[76] T. C. Wang, M. Y. Liu, J. Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 8798–8807, 2018, doi: 10.1109/CVPR.2018.00917.

[77] T. Park, M. Y. Liu, T. C. Wang, and J. Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 2332–2341, 2019, doi: 10.1109/CVPR.2019.00244.

[78] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, “SEAN: Image synthesis with semantic region-adaptive normalization,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 5103–5112, 2020, doi: 10.1109/CVPR42600.2020.00515.

[79] X. Zhou *et al.*, “CoCosNet v2: Full-Resolution Correspondence Learning for Image Translation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 11460–11470, 2021, doi: 10.1109/CVPR46437.2021.01130.

[80] A. Marzullo, S. Moccia, M. Catellani, F. Calimeri, and E. De Momi, “Towards realistic laparoscopic image generation using image-domain translation,” *Comput. Methods Programs Biomed.*, vol. 200, p. 105834, 2021, doi: 10.1016/j.cmpb.2020.105834.

[81] B. Zhou *et al.*, “Semantic Understanding of Scenes Through the ADE20K Dataset,” *Int. J. Comput. Vis.*, vol. 127, no. 3, pp. 302–321, 2019, doi: 10.1007/s11263-018-

1140-0.

[82] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ADE20K dataset,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5122–5130, 2017, doi: 10.1109/CVPR.2017.544.

[83] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, 2015, doi: 10.1007/978-3-319-10602-1_48.

[84] D. S. Pavlichin and T. Weissman, “Chained Kullback-Leibler divergences,” *IEEE Int. Symp. Inf. Theory - Proc.*, vol. 2016-Augus, pp. 580–584, 2016, doi: 10.1109/ISIT.2016.7541365.

[85] M. K. Ghalati, A. Nunes, H. Ferreira, P. Serranho, and R. Bernardes, “Texture Analysis and Its Applications in Biomedical Imaging: A Survey,” *IEEE Rev. Biomed. Eng.*, vol. 15, pp. 222–246, 2022, doi: 10.1109/RBME.2021.3115703.

[86] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” pp. 1–9, 2014, [Online]. Available: http://arxiv.org/abs/1412.3555

[87] R. Dey and F. M. Salemt, “Gate-variants of Gated Recurrent Unit (GRU) neural networks,” *Midwest Symp. Circuits Syst.*, vol. 2017-Augus, no. 2, pp. 1597–1600, 2017, doi: 10.1109/MWSCAS.2017.8053243.

[88] D. Saxena and J. Cao, “Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions,” 2020, [Online]. Available: http://arxiv.org/abs/2005.00065

[89] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, “Diffusion Models in Vision: A Survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 8, pp. 1–25, 2022, doi: 10.1109/TPAMI.2023.3261988.

[90] K. L. CHUNG, *MARKOV CHAINS WITH STATIONARY TRANSITION PROBABILITIES*. 1960.

[91] Y. Song and S. Ermon, “Generative modeling by estimating gradients of the data

distribution," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, 2019.

[92] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling Through Stochastic Differential Equations," *ICLR 2021 - 9th Int. Conf. Learn. Represent.*, pp. 1–36, 2021.

[93] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Adv. Neural Inf. Process. Syst.*, vol. 2020-Decem, no. NeurIPS 2020, pp. 1–25, 2020.

[94] Y. Wu and K. He, "Group Normalization," *Int. J. Comput. Vis.*, vol. 128, no. 3, pp. 742–755, 2020, doi: 10.1007/s11263-019-01198-w.

[95] W. Wang *et al.*, "Semantic Image Synthesis via Diffusion Models," 2022, [Online]. Available: http://arxiv.org/abs/2207.00050

[96] C. Doersch, "Tutorial on Variational Autoencoders," pp. 1–23, 2016, [Online]. Available: http://arxiv.org/abs/1606.05908

[97] S. Bodenstedt, A. Jenke, S. Speidel, M. Wagner, M. Daum, and A. Tabibian, "Surgical Instrument Multi-Domain Segmentation Challenge," *synapse*, 2023. https://www.synapse.org/#!Synapse:syn47193563/wiki/

[98] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, vol. 2. 2008.

[99] M. P. Umugwaneza and B. Zou, "A Novel Similarity Measure using a Normalized Hausdorff Distance for Trademarks Retrieval Based on Genetic Algorithm," vol. 1, pp. 312–320, 2009.

[100] S. M. Pizer *et al.*, "Adaptive Histogram Equalization and Its Variations.," *Comput. vision, Graph. image Process.*, vol. 39, no. 3, pp. 355–368, 1987, doi: 10.1016/S0734-189X(87)80186-X.

[101] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention Mask Transformer for Universal Image Segmentation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2022-June, pp. 1280–1289, 2022, doi: 10.1109/CVPR52688.2022.00135.