

REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

**DEEP LEARNING BASED SURGICAL PHASE
RECOGNITION IN LAPAROSCOPIC VIDEOS**

Kadir KIRTAÇ

DOCTOR OF PHILOSOPHY THESIS
Department of Computer Engineering
Program of Computer Engineering

Supervisor
Prof. Dr. Nizamettin AYDIN

March, 2023

REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

**DEEP LEARNING BASED SURGICAL PHASE RECOGNITION IN
LAPAROSCOPIC VIDEOS**

A thesis submitted by Kadir KIRTAÇ in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY** is approved by the committee on 06.03.2023 in Department of Computer Engineering, Program of Computer Engineering.

Prof. Dr. Nizamettin AYDIN
Yildiz Technical University
Supervisor

Approved By the Examining Committee

Prof. Dr. Nizamettin AYDIN, Supervisor
Yildiz Technical University

Asst. Prof. Dr. Hamza Osman ILHAN, Member
Yildiz Technical University

Assoc. Prof. Dr. Atakan KURT, Member
Istanbul Cerrahpasa University

Assoc. Prof. Dr. Gökhan BİLGİN, Member
Yildiz Technical University

Prof. Dr. Mustafa Ersel KAMAŞAK, Member
Istanbul Technical University

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Deep Learning Based Surgical Phase Recognition In Laparoscopic Videos supervised by my supervisor, Prof. Dr. Nizamettin AYDIN. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Kadir KIRTAÇ

Signature



This study was performed when Kadir KIRTAÇ was working as a full-time employee at Caresyntax GmbH. Peer-reviewed articles resulted from this study have been published with financial support of Caresyntax GmbH.

Dedicated to my family



ACKNOWLEDGEMENTS

My dear wife Sinem, my son Ömer, and my daughter Zeynep: your support has always encouraged me during all the difficult times. Many thanks, and I love you so much!

To my esteemed family members who always supported me and taught me the importance of being a good and moral person; my dear mother Pakize and my father Zafer, my dear brother Fatih and my late grandfather Ali KIRTAÇ. I love you very much!

I would like to thank all technical contributors, especially to my supervisor Prof. Nizamettin AYDIN and my dear colleague at Caresyntax GmbH, Dr. Florian ASPART, for continuously guiding me through this process. I want to thank my thesis committee members Dr. Hamza Osman ILHAN, Assoc. Prof. Atakan KURT, Assoc. Prof. Gökhan BİLGİN, and Prof. Mustafa Ersel KAMAŞAK for their patience, support, and helpful feedback during the monitoring and defense of the thesis.

I would like to express my special thanks to my employer Caresyntax GmbH for allowing me to use the dataset, which is a very important part of this work. I enjoyed working with our clinical partners Dr. Joël L. LAVANCHY and Prof. Guido BELDI, who provided us with the data and helped with clarifying some unknowns in the clinical domain. Thank you very much!

I am grateful to my collaborators who helped me with data annotation and technical discussions. Jon Lindström BOLMGREN, Julien SCHWERIN, Dr. Isabell TWICK, Michael STENZEL, Dr. Amirhossein BAYAT, and Salih KARAGÖZ. And, to my latest supervisor at Caresyntax GmbH, Lamija PASALIC, for encouraging me to complete this thesis. Thank you so much!

I hereby commemorate my former supervisor Prof. Veli HAKKOYMAZ, who died due to Covid-19, with mercy and gratitude. May Allah bless his soul!

Kadir KIRTAÇ

TABLE OF CONTENTS

LIST OF ABBREVIATIONS	viii
LIST OF FIGURES	ix
LIST OF TABLES	xv
ABSTRACT	xvii
ÖZET	xix
1 INTRODUCTION	1
1.1 Literature Review	8
1.1.1 Before Deep Learning	8
1.1.2 Deep Learning Methods	8
1.1.3 Learning With Limited Data	11
1.1.4 Performance Evaluation	13
1.2 Objective of the Thesis	14
1.3 Original Contribution	15
1.4 Terminology in Surgical Phase Recognition	18
1.5 Challenges in Surgical Phase Recognition	19
1.6 Outline	20
2 DEEP NEURAL NETWORKS FOR SURGICAL PHASE RECOGNITION	22
2.1 Basics	22
2.2 Convolutional Neural Networks	24
2.3 Deep Residual Networks	26
2.4 Multi-stage Temporal Convolutional Networks (MSTCN)	27
2.5 Transformer Networks	29
3 DATASETS FOR SURGICAL PHASE RECOGNITION	31
3.1 Overview	31
3.2 Cholec80	32
3.3 Insel	35
3.4 HeiChole	39

3.5	Other Datasets	41
3.6	Discussion And Conclusion	43
4	SYSTEMATIC COMPARISON OF DATASETS AND MODEL PERFORMANCE FOR SURGICAL PHASE RECOGNITION	44
4.1	Overview	44
4.2	Methods	46
4.2.1	Datasets	46
4.2.2	Phase Recognition Models	46
4.2.3	Evaluation Methodology	48
4.3	Results	54
4.3.1	Comparative Analysis of Cholec80 and Insel Dataset	54
4.3.2	Surgical Phase Recognition on Insel Dataset	59
4.4	Discussion	66
5	CROSS-DATASET TRANSFER LEARNING FOR SURGICAL PHASE RECOGNITION	70
5.1	Overview	70
5.2	Methods	72
5.2.1	Using ImageNet Dataset as the Source	72
5.2.2	Using Insel dataset as the Source	73
5.3	Results	75
5.3.1	Performance in Terms of Aggregated Metrics	75
5.3.2	Performance in Terms of Individual Phases	76
5.3.3	Performance in Terms of Individual Videos	79
5.3.4	The Effect of the Learning Rate	81
5.4	Discussion	82
6	RESULTS AND DISCUSSION	84
	REFERENCES	87
A	APPENDIX	94
	PUBLICATIONS FROM THE THESIS	95

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
CAS	Computer Assisted Surgery
CNN	Convolutional Neural Network
CV	Computer Vision
DL	Deep Learning
DNN	Deep Neural Network
Lap Chole	Laparoscopic Cholecystectomy
LSTM	Long Short Term Memory
MIS	Minimally Invasive Surgery
ML	Machine Learning
MSTCN	Multi-Stage Temporal Convolutional Network
OR	Operating Room
ResNet	Residual Network
TCN	Temporal Convolutional Network

LIST OF FIGURES

Figure 1.1	Illustration of two different video signals that may occur in an operation room (OR), (a) a scene from a laparoscopic surgery video, and (b) a scene captured by the room camera mounted on the OR wall.	3
Figure 1.2	Illustration of the machine learning pipeline for surgical phase recognition. (Images are from the courtesy of Cholec80[3], CC BY 4.0). (a) A machine learning algorithm is trained with images from training data and the human-labeled annotation of these images. (b) The trained model is used to predict the phases in unseen data. Predicted outputs can be combined with different data for various levels of automation.	5
Figure 1.3	An illustration of two-stage modeling pipeline for surgical phase recognition. A CNN extracts features from the images. Features are stacked along the temporal axis, and a temporal model performs the final classification. Images are from the courtesy of Cholec80[3]. Best viewed in color.	10
Figure 1.4	Illustration of the qualitative results for surgical phase recognition. Each color bar represents a surgical phase, P1, P2, and P3, and the time axis is represented in the horizontal direction from left to right. (a) Ground truth of a video, (b) Predictions for the same video. A prediction is counted as a misclassification if it mismatches the true color, either on a block or frame basis.	19
Figure 2.1	Illustration of basic neural network structures. (a) How a single neuron works, (b) Fully-connected neural network organization. Each neuron is connected to each neuron in the previous and next layers.	23
Figure 2.2	An illustration of the CNN architecture. The network performs a series of spatial convolution and pooling operations following one another. Each operation results in a new set of feature maps with reduced size.	25

Figure 2.3	A more detailed look into ResNet architecture showing the parameters and interaction of layers. Every color represents a different residual block. The size of each residual block varies in the original architecture. Skip connections are repeated for every block but skipped in this picture for simplicity. (Figure adapted from He et al. [15], CC BY 4.0.)	27
Figure 2.4	An illustration of how "skip connection" works in ResNet. The input signal is transferred from an alternative pathway and added to the signal at the end of the ResNet block. (Figure adapted from He et al. [15], CC BY 4.0.)	27
Figure 2.5	An illustration of MSTCN architecture. Each layer features a set of 1D dilated convolutions. The dilation rate is doubled at each successive layer, which in turn increases the receptive field. (Figure adapted from [26, 29])	28
Figure 2.6	Illustration of the Trans-SVNet model. (Figure adapted from [31])	30
Figure 3.1	Illustration of the annotation software. Frames of a video are shown on the software screen according to their timestamps. The slider bar can be moved in both directions along the time, and phase labels are assigned to the segments of the video. Labels are logged into a database when the task is completed. Images are from the courtesy of Cholec80[3]. Best viewed in color.	32
Figure 3.2	Video frames sampled from a video from the Cholec80 dataset, representing each surgical phase described above [3]. Best viewed in color. (CC BY 4.0.)	34
Figure 3.3	Distribution of duration of phases in Cholec80 dataset.	34
Figure 3.4	Illustration of visual examples of several instruments from Cholec80 dataset. Best viewed in color.	35
Figure 3.5	Example video frames of major phases in Insel dataset. Best viewed in color.	38
Figure 3.6	Example video frames of phases that do not happen regularly in Insel dataset. Best viewed in color.	39
Figure 3.7	Distribution of phase durations in Insel dataset.	39
Figure 3.8	Other instruments from Insel dataset: (a) Argon beamer: used for coagulation, (b) Overholt forceps: tissue separation. (c) Cotton swab: for compress, (d) Ultrasound probe: to check for gallstones, (e) Ligasure: for tissue dissection.	40

Figure 4.1	Lengths and order of surgical phases are highly imbalanced in Insel dataset. Color-coded ribbons illustrate phase progression for ten different procedures. Relative operation duration is shown (normalized to 100%) (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	45
Figure 4.2	The calculation of metrics is illustrated as a concrete example. True labels and predictions are represented as horizontal bars. Each colored bar represents a different phase label, and the number shown under each box (phase) represents the duration of that phase. Best viewed in color.	51
Figure 4.3	Segment-based metrics compute two error types. (a) The first one, over-segmentation, computes the error if multiple predicted segments are contained in one true segment. (b) The second one measures the ordering of segments and allows for small temporal offsets. The offset might be due to inter-rater differences and they should not affect the performance. Best viewed in color. (Figure and caption adapted from Lea et al. [51], CC BY 4.0.)	53
Figure 4.4	Distribution of video durations (in minutes) in Insel dataset and Cholec80. Insel dataset shows more variability in durations and has longer videos than Cholec80. (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	55
Figure 4.5	Distribution of phase presence as a percentage of the total number of videos. Some phases are extremely underrepresented across videos. For example, Gallbladder drainage and Liver biopsy are present only in 0.6% and 0.4% of videos in our Insel dataset, respectively (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	55
Figure 4.6	Example of surgical scenes for surgical phases (from Insel dataset) which are absent in Cholec80: (a) liver biopsy; (b) gallbladder drainage; (c) intraoperative cholangiography; (d) application of a cotton tape for laparoscopic Pringle maneuver; (e) application of the endoscopic linear stapler; (f) intraoperative ultrasonography. These latter three examples include procedures that are not typical for cholecystectomy (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	56

Figure 4.7	Distribution of phase duration across videos (in seconds) in Insel dataset and Cholec80. Duration (y-axis) is in log scale. Some phases present in Insel videos are missing in Cholec80. Phases missing in a video are ignored (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	57
Figure 4.8	Distribution of phase duration as a percentage of total duration in Insel dataset and Cholec80. The imbalance between phases for both datasets is severe. Gallbladder drainage and Liver biopsy phases are extremely underrepresented in Insel dataset, with 0.15% and 0.06% duration, respectively (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	62
Figure 4.9	Comparative phase-specific performance analysis of the models in Insel dataset. For the minority phases, <i>gallbladder drainage</i> , and <i>liver biopsy</i> , the TeCNO model showed the best performance among others. The performance gap was highly significant for <i>liver biopsy</i> , which is an extremely under-represented phase in Insel dataset, between TeCNO and Trans-SVNet (28 times and 1.57 times greater for recall and precision, respectively) (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	63
Figure 4.10	Phase-specific performance analysis of the TeCNO model on Insel dataset. (a) Illustration of phase level precision and recall averaged across videos. One can notice the strong variation in performance between phases. (b) The confusion matrix illustrates the confusion between phases. Every sample (frame-wise prediction and ground truth) contributed equally during the computation. Confusion between temporally closer phases was more prevalent than in temporally distant phases (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	64
Figure 4.11	Phase-specific performance of the TeCNO model showed high variation across videos on Insel dataset (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	65
Figure 4.12	Performance based on individual videos of the TeCNO model showed high variation on Insel dataset (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)	65

Figure 4.13	Illustration of segmental performance metrics of the online TeCNO model on Insel dataset, on a video basis. (a) Frame-wise f1-score compared to segmental f1-score and edit score. The edit score is encoded by the color bar. F1@50 displays the individual performance in a wider range compared to the frame-wise f1-score. (b) Timeline plot of the ground truth (top) and predictions (bottom) for <i>video12</i> , illustrating the over-segmentation errors. Each phase is encoded by a different color explained in the legend (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.) .	66
Figure 5.1	Illustration of the transfer learning concept. A model trained on a source domain transfers its knowledge to a target domain. The target domain, where often data and labels are scarce, benefits from the transferred knowledge.	71
Figure 5.2	Illustration of transfer learning from ImageNet dataset. (a) First, Resnet50 is trained with ImageNet. (b) Weights except the final linear layer are transferred to the target domain, Cholec80. The transferred Resnet50 weights are fine-tuned with the Cholec80 training set. Finally, features are extracted from ResNet's final conv layer. (c) Online MSTCN (TeCNO 4.2.2) is trained from scratch, using features extracted in the previous step. Images in (a) are not part of the original ImageNet dataset, but they are representative of the classes in ImageNet. (https://commons.wikimedia.org/wiki , CC by 4.0)	74
Figure 5.3	Results of transfer learning from various sources. Training without transfer learning is included for completeness (the red curve). The results show that transferring from the Insel worked the best, both in terms of accuracy and f1-score.	75
Figure 5.4	Illustration of the performance in the level of individual phases. Both precision, recall, and f1-score are presented. Results of fine-tuning from both Insel and ImageNet are provided.	78
Figure 5.5	The illustration of video-based performance assessment using the quantiles. The overlaid text shows the video name associated with the video performing for the x and y at that point.	80
Figure 5.6	The effect of learning rate on the performance of the fine-tuned TeCNO model. The model transferred from ImageNet (left column) is very much affected by the learning rate. The best learning rate is 5×10^{-3}	81

Figure A.1 Results of transfer learning from various sources. Training without transfer learning is included for completeness (the red curve). The results show that transferring from the Insel worked the best, both in terms of precision and recall. 94



LIST OF TABLES

Table 3.1	Basic properties of the videos in Cholec80.	32
Table 3.2	Definitions of surgical phases in Cholec80 dataset.	33
Table 3.3	The exact duration of each phase (in seconds) present in the Cholec80. The std describes the deviation of the mean across the 80 videos.	35
Table 3.4	Basic properties of the videos in Insel dataset.	36
Table 3.5	Definitions of surgical phases in our Insel dataset.	37
Table 3.6	The exact duration of phases in our Insel dataset (in seconds). The std describes the deviation of the mean across the videos.	40
Table 3.7	Comparative illustration of surgical phase recognition datasets, including the Insel dataset we introduced in this work. Phase names are defined differently across datasets.	42
Table 4.1	Illustration of the mean precision calculation on a toy data: (1) the first mean is computed over the videos for each phase (from top to bottom in the vertical direction), (2) the final mean is computed over the phases (from left to right in the horizontal direction), resulting in 75% mean precision.	52
Table 4.2	Illustration of the mean recall calculation on a toy data: (1) the first mean is computed over the videos for each phase (from top to bottom in the vertical direction), (2) the final mean is computed over the phases (from left to right in the horizontal direction), resulting in 73% mean recall.	52
Table 4.3	Phase transitions in Insel dataset. Compared with Cholec80, transitions in Insel dataset possess more complexity. The numbers indicate the relative (in percent) transitions from one phase (row-axis) to another (column-axis) (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.)	58
Table 4.4	Phase transitions in Cholec80. The numbers indicate the relative transitions (in percent) from one phase (row-axis) to another (column-axis) (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.)	59

Table 4.5	Model performance (mean $\pm$ std) using aggregated metrics on Insel dataset. The performance of the Resnet50 backbone is shown in the first row. Stacking the MSTCN on top of Resnet50 boosted the phase segmentation performance (the last two columns in the second row). (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.) . . .	60
Table 5.1	Illustration of the video-based performance in terms of the f1-score and accuracy, for 5 fine-tuning videos. The highest-performing five, the median, and the lowest-performing five videos are shown, from top to bottom.	80



Deep Learning Based Surgical Phase Recognition In Laparoscopic Videos

Kadir KIRTAÇ

Department of Computer Engineering

Doctor of Philosophy Thesis

Supervisor: Prof. Dr. Nizamettin AYDIN

Deep learning and computer vision-based surgical decision support offer great potential for automating tasks in the operating room. In this thesis, we focused on deep learning-based methods for surgical phase recognition in laparoscopic surgery videos. In particular, we conducted a data-centric study that included datasets and model performance evaluation in depth.

First, we reviewed existing deep learning-based phase recognition methods and datasets in detail. To this end, we created a dataset of 384 laparoscopic cholecystectomy videos. This dataset has become one of the largest among the existing surgical phase recognition datasets. Second, we performed a systematic comparison between our dataset and Cholec80, which is the most frequently used publicly available dataset for surgical phase recognition, and observed significant differences between the two datasets. Our videos were much longer and included additional phases. The phase transitions in our dataset were more complex. Next, we trained two two-stage phase recognition models with our dataset, which had previously shown the highest performance on the Cholec80, and evaluated their performance in detail. The performance of the models varied greatly between surgical phases and videos. In particular, our results demonstrated the difficulty of recognizing the highly underrepresented surgical phases. Major phases were recognized with at least 76 percent recall. Third, we studied how a model trained on our large-scale dataset could be adapted to Cholec80, which is significantly smaller than our dataset. To this end, we presented fine-tuning strategies for model adaptation, experimenting with different numbers of labeled videos in the target dataset. Our experiments

showed that when fine-tuned with only five videos from Cholec80, the model achieved a higher performance than that obtained by training it with all the videos in the original set (40 videos). This result showed that transfer learning can make an important contribution to surgical phase recognition where data scarcity is significantly evident. Our results revealed that larger datasets are needed for the fully automatic use of surgical phase recognition models in the real world. We believe that our study will be a useful guide for experts who deploy and use these models in the real world, and for researchers who constantly work with public data having limited access to private datasets.

Keywords: Deep learning, computer vision, surgical phase recognition, laparoscopic surgery videos, transfer learning



Laparoskopik Videolarda Derin Öğrenme Tabanlı Cerrahi Faz Tanıma

Kadir KIRTAÇ

Bilgisayar Mühendisliği Anabilim Dalı
Doktora Tezi

Danışman: Prof. Dr. Nizamettin AYDIN

Derin öğrenme ve bilgisayarla görüye dayalı cerrahi karar desteği ameliyathanedeki görevleri otomatikleştirmek için önemli bir potansiyel sunmaktadır. Bu tezde, laparoskopik cerrahi videolarda faz tanıma için derin öğrenme tabanlı yöntemleri inceledik. Özellikle, veri kümelerini ve model performans değerlendirmesini içeren veri merkezli bir çalışma gerçekleştirdik.

İlk olarak, mevcut derin öğrenmeye dayalı faz tanıma yöntemlerini ve veri kümelerini ayrıntılı bir şekilde inceledik. Bu bağlamda 384 adet laparoskopik kolesistektomi videosundan oluşan veri kümesi oluşturduk. Bu veri kümesi mevcut cerrahi faz tanıma veri kümeleri arasında en büyüklerinden birisi haline geldi. İkinci olarak, veri kümemiz ile cerrahi faz tanıma için en sık kullanılan açık erişimli veri kümesi olan Cholec80 arasında sistematik bir karşılaştırma gerçekleştirdik. Sonuç olarak bu iki veri kümesi arasında önemli farklılıklar gözlemledik. Kendi veri kümemizin videoları çok daha uzun ve Cholec80’de bulunmayan ek fazlar içermektedir. Buna ek olarak, cerrahi faz geçişleri daha karmaşıktı. Sonra, Cholec80 veri kümesinde en yüksek performansı göstermiş olan iki adet iki aşamalı faz tanıma modelini veri kümemiz ile eğittik ve performanslarını ayrıntılı olarak değerlendirdik. Modellerin performansı cerrahi fazlar ve videolar arasında büyük farklılıklar gösterdi. Sonuçlarımız oldukça az temsil edilen cerrahi fazları tanımanın zorluğunu gösterdi. Ana fazlar en az yüzde 76 geri çağırma ile tanındı. Üçüncüsü, büyük ölçekli veri kümemizde eğitilen bir modelin, veri kümemizden önemli ölçüde daha küçük olan Cholec80’e transfer öğrenmesi ile nasıl uyarlanabileceğini çalıştık. Bu amaçla, hedef veri kümesinde farklı sayıda etiketli videolar kullanarak model uyarlanması için ince ayar stratejileri sunduk. Modelin

Cholec80 veri kümesinden yalnızca 5 video ile ince ayarı yapıldığında elde edilen performans orjinal kümedeki tüm videolarla (40 video) eğitilmesiyle elde edilenden daha fazla oldu. Bu sonuç transfer öğrenmesinin veri kıtlığının önemli ölçüde belirgin olduğu cerrahi faz tanıma için önemli bir katkı sunabileceğini gösterdi.

Sonuçlarımız, cerrahi faz tanıma modellerinin gerçek ortamda tam otomatik olarak kullanılması için daha büyük veri kümelerine ihtiyaç olduğunu ortaya koydu. Çalışmamızın bu modelleri gerçek ortamda kullanan uzmanlar ve özel veri kümelerine sınırlı erişimi olup sürekli olarak açık kaynak verilerle çalışan araştırmacılar için faydalı bir rehber olacağına inanıyoruz.

Anahtar Kelimeler: Derin öğrenme, bilgisayarla görü, cerrahi faz tanıma, laparoskopik cerrahi videolar, transfer öğrenmesi



1

INTRODUCTION

Surgery is a medical expertise that uses surgical guidelines and instrumental techniques on a person to treat a disease or injury in the body. Surgical techniques, which have been applied since the earliest times of human history, have developed in parallel with the innovations made in all areas of the industry. For example, today's operating rooms (OR) are powered by computerized systems. In particular, surgical video footage is captured and displayed on high-resolution screens to monitor the progress of the surgery, while also being saved on high-capacity servers and cloud storage for retrieval when needed. Moreover, the surgical operation can be managed or supported via a live remote connection.

Every progress and technological breakthrough is aimed at protecting patient life and minimizing the recovery process. One of the most important breakthroughs has been the development of minimally invasive techniques. For centuries, surgeries have been performed with open surgical techniques. In this method, deep and long incisions are made from the outside of the body. Although it is still used and maintains its importance, open techniques have significant disadvantages such as blood loss and delayed recovery time. However, the main advantages of laparoscopic, or minimally invasive surgery (MIS) over open surgery include but are not limited to reducing pain and hemorrhage, providing a shorter recovery time, and a lower complication rate for the patient [1]. During laparoscopy, a few small incisions (usually 0.5–1.5 cm) are performed in the abdomen. Video is captured by the laparoscope, a small camera inserted into one of these incisions, and simultaneously displayed on a monitor. This video and other patient health data are usually recorded in a suitable format and stored for later retrieval. To derive benefit from this stored data, artificial intelligence (AI) and machine learning (ML) algorithms are being developed.

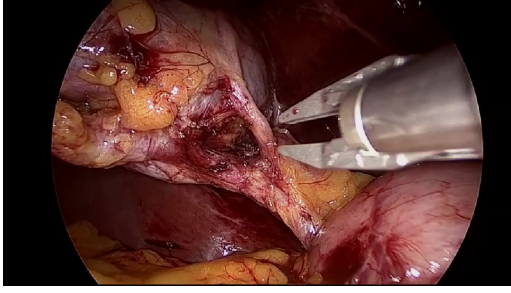
Artificial intelligence (AI) and machine learning (ML) use data to train algorithms that can act like humans, or support them in decision-making. With advances in high-performance computing and storage technologies, ML algorithms now provide better performance with large-scale data. For this reason, it has gained great

importance to store pre-operative, intra-operative, and postoperative health data related to the patient, and use them to train ML models for prediction. Such health data include, but are not limited to, patient electronic health records (EHR), age, gender, height, weight, blood values, previous medications, anesthesiology data, and surgical video recordings. These data provide a unique value about a patient's condition: they bring so much potential when collected in large quantities to develop computational methods for diagnostic and preventive healthcare. To set some standards on this topic, researchers and practitioners from the industry came together to create a new discipline called Surgical Data Science. The aim of this new discipline is to develop new techniques for computer-assisted surgery (CAS) and increase the knowledge sharing and collaboration between the research and industry [2].

We perceive the world and the objects around us with our eyes and the special receptors in our brain, which are adapted to process this visual signal and provide us with an excellent view of the visual scene. The function performed by our eyes in the real world is accomplished by digital cameras. With a similar analogy to the eye and brain connection, the camera is responsible for capturing the visual scene and converting it to electrical signals, and the computer is responsible for processing and interpreting these signals. This flow acts similarly in the operating room as well. Visual data is central to recognizing events in the operating room, and it can often occur in two different forms: (1) visual signals captured by the endoscopic camera (laparoscope), which are eventually served as a video stream of the patient's inner body 1.1(a), (2) video stream captured by the room camera mounted on the OR wall, which records the patient and the movements of the OR staff 1.1(b).

The field of AI which deals with processing visual input such as videos and images to develop decision-making algorithms is called Computer Vision (CV). Images often occur in the form of a two-dimensional matrix when digitized, and the smallest piece of information in this form is the "pixel". Each pixel is represented as a scalar number ranging between 0 and 255. The spatial grouping of the pixels represents the blobs, and the grouping of blobs forms the view of objects in a scene. A video, on the other hand, is an ordered collection of images (or "frames") with respect to the time they are recorded. Hence, a video is a 4-D signal, where the first three dimensions come from the image, i.e., the width, height, and color channels, and the last dimension is the time.

Computer vision presents computational methods or algorithms to solve various types of visual tasks. These tasks can be accomplished in the image domain, the so-called 2-D vision. Other tasks that are inferring 3-D information from 2-D images are the subject of 3-D vision. Example applications for the former case include but are not limited to identifying arbitrary objects and their precise boundaries in images, counting people



(a) An example view from a laparoscopic cholecystectomy video. (The courtesy of Cholec80 dataset [3], CC BY 4.0)



(b) An illustration of a surgery room (operating theater). This view is captured by a room camera mounted on the room wall. Original file (1600x884 pixels, MIME type: image/jpeg) (CC-BY-SA-4.0). [4]

Figure 1.1 Illustration of two different video signals that may occur in an operation room (OR), (a) a scene from a laparoscopic surgery video, and (b) a scene captured by the room camera mounted on the OR wall.

and identifying the speed of vehicles in a traffic video, and recognizing objects in video frames. Applications concerning the latter tasks include depth estimation from 2-D monocular videos, 3-D human pose estimation, or 3-D reconstruction from 2-D images. These latter tasks are often more tricky to handle, which might require precise data and sophisticated post-processing algorithms.

Computer vision has grown tremendously over the past decade. This progress is based on the development of new data-driven learning algorithms called *deep learning*. Now, almost all computer vision problems are solved with deep learning-based algorithms. The main benefit provided by deep learning is that it effectively allows learning with a huge amount of data. This is achieved by increasing the layer depth of conventional neural networks. Increased layer depth, in turn, requires learning more parameters. For example, the amount of data required for model training increases exponentially with the number of parameters. Fortunately, the developments in internet technologies and data storage in the last 15-20 years have made huge amounts of data accessible with low effort. In addition, many factors such as cheaper hardware costs and processing power have accelerated the development of deep learning techniques. In particular, high-performance deep learning models have been developed thanks to the available large amount of unstructured high-dimensional data such as images, text, and audio. Many innovations have been made to reduce the number of parameters, especially in deep networks designed for image classification. One of the most important among these was convolutional neural networks (CNNs) [5]. A CNN is a hierarchically layered learning architecture inspired by the human visual system. The neurons in the successive layer receive information from only their

spatial neighborhood in the previous layer. This kind of local connectivity reduces the number of trainable parameters dramatically. In addition to the image data type, deep network architectures specific to text[6, 7] and audio data[8] were also developed. In summary, developments in the last decade have been based on learning with big data, and the trend is expected to continue in this direction.

Surgical video streams and recordings are vital to surgeons. They help surgeons in many ways, but two of them stand out. First, the video streams are displayed on a monitor to guide surgeons during the surgery. Without this guidance, it would not be possible for the surgeon to perform the operation. Second, the video streams are often stored in local servers or cloud storage to serve for post-operative review. Post-operative video review has two main benefits; (1) to improve patient safety and quality of care, (2) to rate the surgeon's performance. For example, cases, where the staff or the surgeon fails during surgery, can be found in the video review process. This can help the hospital prevent such failures during subsequent surgeries. Second, the review could be performed to rate the surgeon's skill. This way of performance feedback is crucial for surgeons to improve their skills.

Beyond performing these tasks manually, AI can help automate these tasks in the operating room. A surgeon or an expert can sit in front of a monitor, and watch the full surgery video to assign one of the possible phase categories to every frame of it. This is the manual way of labeling a surgery video. It requires exhaustive human labor, which can be a very tiring and error-prone task for a human to do. Instead, AI can automate this task if properly trained.

The automation can be provided by training a machine learning model to predict the procedural phase for every timestep of a surgical video. The benefits of such automation can be in multiple folds. First, such a system would enable the performing of post-operative analyses effectively on a large scale. For example, the time a surgeon stays on every individual phase of a surgery can be automatically calculated and saved into a database. For a specific surgery type, this can help to establish a standard and error for phase durations between surgeons. For example, surgeons working above or below the standards can be extracted automatically. The second use case can be OR staff scheduling. Operational delays and lack of coordination among the surgical staff often cause delays in the surgeries. Delays cause mental stress for the patients, as well as loss of reputation and reduced revenue for the hospitals. AI-based online surgical phase recognition can help the staff prepare on time for the next surgery. Third, it can serve improved patient safety, which is the most important factor for a surgical operation. For better patient safety, the sensitivity of the model can be adjusted for critical stages where the surgeon's error can cause serious problems for the patient. In this way, AI-based phase recognition can help the surgeon identify this phase more

precisely, which will eventually increase the surgeon’s attention and reduce the risk of safety concerns. An example of such a critical phase is the hepatocystic triangle dissection phase of a Lap Chole procedure, where obtaining the critical view of safety is crucial.

In formal terms, AI-based surgical phase recognition is the task of recognizing the phase of the surgery for every frame of a video. This is accomplished by training an ML algorithm that predicts the most likely phase from a set of phase categories for a given video frame. This flow is illustrated in Figure 1.2. Some methods use only

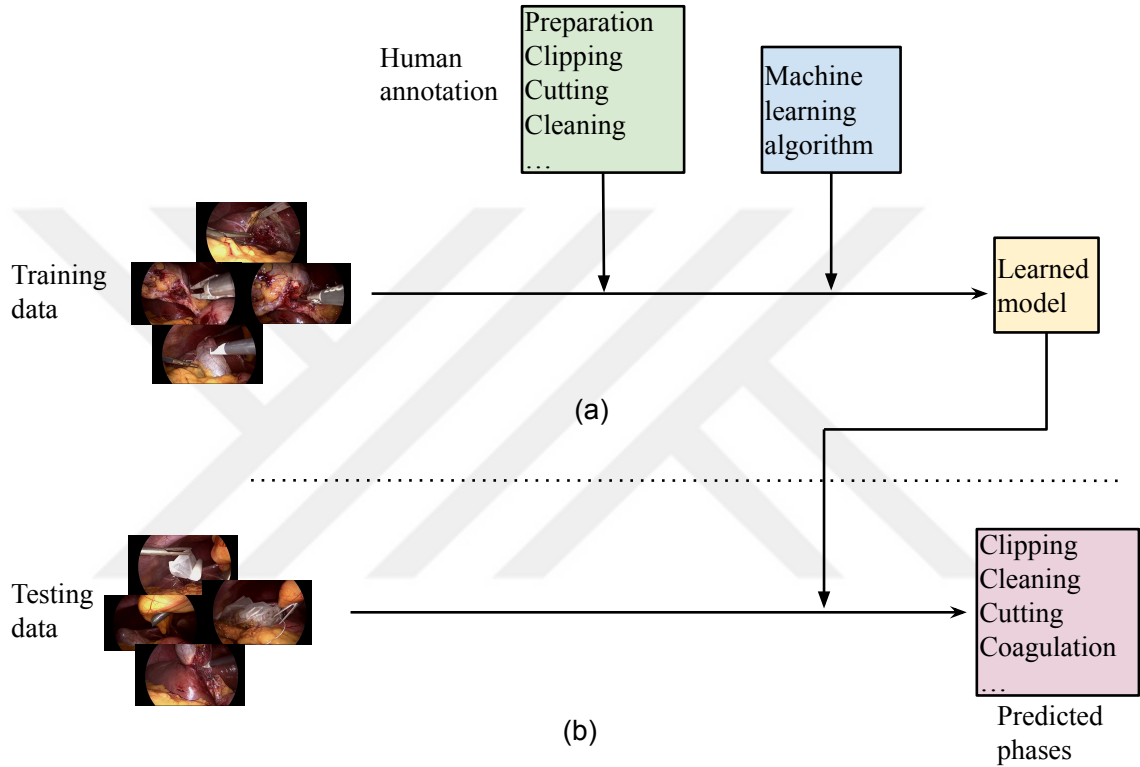


Figure 1.2 Illustration of the machine learning pipeline for surgical phase recognition. (Images are from the courtesy of Cholec80[3], CC BY 4.0). (a) A machine learning algorithm is trained with images from training data and the human-labeled annotation of these images. (b) The trained model is used to predict the phases in unseen data. Predicted outputs can be combined with different data for various levels of automation.

phase information as labels, while others also use surgical instrument information. In this study, we will focus on methods that use only phase information.

The vast majority of work in computer vision (CV) has been focusing on proposing new ML models and algorithms for visual recognition. In contrast, there are relatively few studies examining how the properties of training and test data affect these models. The surgical AI community is also following this trend. Many surgical phase recognition methods have been published recently. The ML models presented in these publications were mostly trained and tested on a single generic dataset named Cholec80 [9].

Cholec80 contains 80 videos of laparoscopic cholecystectomy (Lap Chole) videos from a single medical center. Although being the largest publicly available dataset for surgical phase recognition, it was curated on purpose. Videos containing alternate phases that do not occur regularly in Lap Chole surgeries were excluded. The aim was to develop a common dataset for benchmarking ML algorithms, rather than providing a complete one that contains all possible difficulties that could arise in the real world. For this reason, videos in Cholec80 present a linear workflow, meaning they start with the preparatory phase and end with the gallbladder retraction. However, emergency surgeries may involve additional steps such as *gallbladder drainage*, *liver biopsy*, and *intraoperative cholangiography*, which are missing in Cholec80. In addition, depending on the difficulty of the case, these surgeries can be converted to open surgery and progress with additional phases. Consequently, the surgical phases found in the Cholec80 are not fully representative of the surgical phase distribution observed in the real world. This motivated us to perform a data-centric analysis in this thesis.

In a data-centric analysis, performance assessment metrics are as important as the datasets used for the analysis. A performance metric involves a series of mathematical computations over the predictions of a model and expected outputs, and reports a scalar value indicating the performance of the model. Various kinds of performance metrics measuring the quality of machine learning models have been developed and used so far. Regression models predict a scalar value for a data sample, and they are often evaluated by computing the absolute difference between the predicted value and the expected output. This is repeated for every data sample in the evaluation set, and the average difference is reported. On the other hand, classification models are usually evaluated using standard metrics such as accuracy, precision, and recall. This has been applied similarly for surgical phase recognition. Existing research papers often focus on benchmarking their models against others, using standard performance metrics. However, these performance metrics do not always explain the usability of the models deployed in the real world. Consequently, the performance of the models needs to be assessed using larger-scale datasets and metrics that are reflecting real-world usability concerns. For example, to assess the performance of the surgical phase recognition, considering only the classification accuracy of individual video frames would not always be sufficient. Metrics considering the order of the predicted video segments should be considered in terms of usability concerns. As an example, for the online deployment of surgical phase recognition, frame-level errors in larger consecutive blocks of predictions may not be important for the user, i.e., for the surgeon. In contrast, if the model shows frequent switches in terms of larger blocks of predictions it can cause mental confusion for the surgeon. Such frequent false alarms generated by an online decision support system do more harm than good to the surgeon. It can

cause the surgeon to be distracted and make serious mistakes. Therefore, knowing the requirements of the system and the expectation of the user in advance, the models can be tuned for a specific performance metric during offline development, and the best model will be deployed for production. In the fourth chapter of this work, we will discuss several methods regarding performance assessment.

The performance of machine learning models and their generalizability depends on the amount and quality of the data used to train them. Nowadays, models trained by internet giants like Google, Facebook, or OpenAI use many millions of images or text documents to train high-performance models. However, that scale of data is not available for all domains (medical domain for example), and for all scales of companies. Smaller-scale companies have fewer financial resources to invest in purchasing data. The availability of surgical videos is also very limited as opposed to the vast amount of data available from daily content on the internet. Data privacy issues and stricter regulations make it even harder to collect and use surgical videos. Moreover, getting the annotations for surgical videos is quite difficult, because labeling requires high skill and domain knowledge. Consequently, generating large-scale labeled datasets from the surgical field is nearly impossible.

One way to tackle the absence of labeled data is to use transfer learning. Transfer learning is the act of transferring knowledge between different data domains and learning tasks, to improve the performance of the learning task in the target domain. In a typical transfer learning scenario, the data available in the source domain is greater than the data available in the target domain. Surgical phase recognition has the advantage that videos from the same surgery type share similar properties. The surgical workflow, the instruments involved, and the patient anatomy (the structure of the abdominal cavity) share a high visual similarity. This makes it appealing to use transfer learning techniques, from a source where data is more available to a source where data is scarce. In practice, data is often imbalanced across medical centers [10]. Moreover, it is difficult for the data to leave its place due to privacy and regulatory concerns. Therefore, it is often less likely to gather datasets from multiple medical centers together to train a model with a combined, larger dataset. Transfer learning could be a solution to this problem. In chapter five, we will discuss how a model that has been trained with a larger set of labeled data obtained from a particular medical center would help improve the performance on fewer labeled data from another medical center.

1.1 Literature Review

We organize the literature review under four subsections; surgical phase recognition (i) before deep learning, (ii) using deep learning, (iii) using limited data, and (iv) performance evaluation.

1.1.1 Before Deep Learning

The traditional workflow of earlier computer vision (CV) techniques was to apply a fixed set of feature extraction [11, 12] and dimensionality reduction algorithms [13, 14] to the visual input. The goal was to first compress the input data into a set of discriminative feature vectors, and then use them to train a classifier. This pipeline, which has lately been replaced by deep learning (DL) based feature extraction [5, 15], is often called hand-crafted feature engineering.

Automating surgical phase recognition with machine learning methods is a relatively new topic. Earlier works on phase recognition date back to 2006 and 2007 when Ahmadi et al. [16] and Padoy et al. [17] constructed multivariate instrument usage signals, and used Dynamic Time Warping (DTW) to temporally synchronize the signals of an unsegmented surgery to an average surgery. DTW algorithm was able to warp a time point in a time series to at least one point in the target time series, minimizing the sum of square distances between all warped points while respecting the temporal order. They evaluated their methods on a small set of Lap Chole videos, and the results were encouraging.

The next series of methods used Hidden Markov Models (HMM), which is a more sophisticated statistical modeling tool compared to DTW, to perform both online [18, 19] and offline [20, 21] recognition of surgical phases. Online recognition considers past signals in a time series to predict the signal in the current time step, whereas offline recognition can use both past and future signals to predict the signal in the current time step. With the HMM setup, these methods were able to predict the phases of an unlabeled surgery within the range of 3%–7% error. However, their testing setup only included a very small set of Lap Chole videos (10 to 16 videos).

1.1.2 Deep Learning Methods

Deep learning (DL) covers a set of new data processing techniques and deep neural network (DNN) architectures to produce models showing higher performance and scale better with big data. DNNs learn a discriminative feature representation of the input data, from the raw data to expected outputs, as well as perform implicit dimensionality reduction.

The rise of deep learning for surgical phase recognition began in 2016. Several deep learning-based methods emerged at the M2CAI-MICCAI Workshop. The first applications of CNNs and Residual Networks [22], Recurrent CNNs [23], spatiotemporal CNNs [24], contextual CNN features for phase and instrument recognition [25], were among the important ones. Although the introduction of CNNs for surgical image analysis was late compared to when they were first used in natural image classification, the impact was huge. Twinanda et al. proposed the first multi-task learning CNN architecture where both surgical phase and instrument present detection was performed in a single CNN [3]. They also used a temporal model (HMM) to improve their baseline classification results obtained by an AlexNet-like architecture. Their results were state-of-the-art, showing 92 percent accuracy for offline and 82 percent for online recognition performance. They also showed 81 percent mean average precision to recognize the instruments in the Lap Chole procedure. This was an important step in surgical phase recognition, especially because these results were reported on the largest publicly available dataset to date, Cholec80 [3]. Cholec80 introduced 80 videos of laparoscopic cholecystectomy procedures. Previous studies used considerably smaller datasets, the size ranging between 6 and 16 videos. In summary, the work of Twinanda et al. was a pioneering work, which clearly showed the promise of deep learning in the largest publicly available dataset.

The success of two-stage modeling in previous work accelerated the work in this direction. In this approach, first, a CNN is trained on a surgical phase dataset, and then it is used to extract features from video frames. The resultant representation is a one-dimensional vector (embedding) for every single frame of the video. The features extracted from a video are stacked horizontally to form the representation of that video. During training, this is repeated for every video in a training dataset. Then, a temporal model is trained using these video features and used as the final classifier. During the classification, the same process is applied to an input video, and the video frames are classified into surgical phases using the final temporal model. The two-stage modeling pipeline is illustrated in Fig.1.3. This pipeline can be implemented in various forms. For example, the first and second stages can be combined into a single model that is used for feature extraction from every video frame. This can be a pre-trained CNN model [26], or any model which is capable of encoding information from an image [11].

The temporal models, on the other hand, include but are not limited to Hidden Markov models [3], LSTM networks [10, 27, 28], temporal convolutional networks (TCNs) [29], and attention models [30, 31]. Since surgery is a well-defined sequential process, the phase in the current time step is closely related to previous phases and

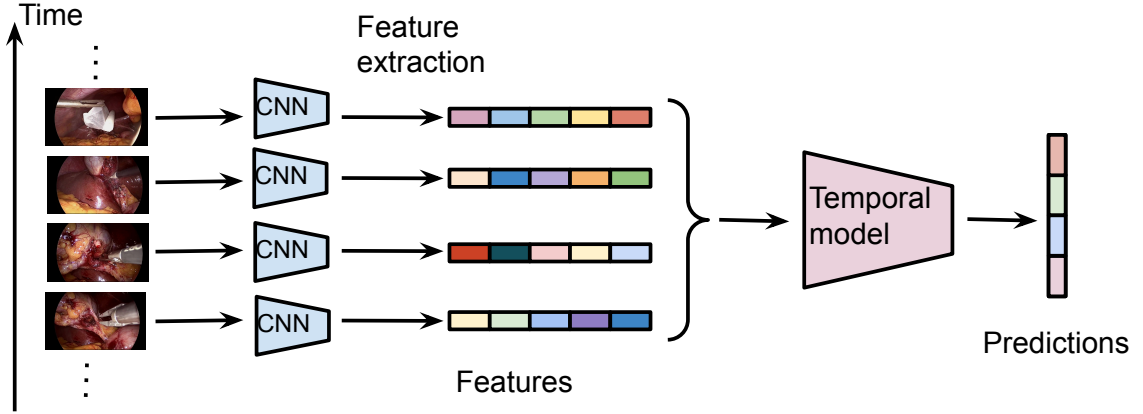


Figure 1.3 An illustration of two-stage modeling pipeline for surgical phase recognition. A CNN extracts features from the images. Features are stacked along the temporal axis, and a temporal model performs the final classification. Images are from the courtesy of Cholec80[3]. Best viewed in color.

events. The excessive length of surgical videos compared to videos of standard action recognition datasets [26] makes it particularly challenging to develop competitive ML models for recognition. To tackle this, methods using temporal models such as LSTMs [32, 33], transformers [30], and TCNs [29] all aim to capture the long-term relationship between current and past/future time steps to make more reliable predictions.

One specific example, SV-RCNet, combined spatial and temporal features from ResNet and LSTM modules to learn spatiotemporal features for surgical phase recognition [27]. Another one, MTRCNet-CL, proposed a new correlation loss to exploit the interactions between instrument presence and phase classification [28]. A memory bank that combines long-range and multi-scale temporal features for surgical phase recognition to capture long-range temporal dynamics was developed in [32]. A hybrid embedding aggregation transformer that learns the attention between low- and high-level features was proposed in [31]. Recently, a neural network that corrects noisy frame-level predictions by learning the segment-frame relationships using a segment-attentive attention module [34] was proposed. [35] introduced a Lap Chole dataset of 371 videos, which is comparable to the size of the dataset we introduce in this work. They performed phase recognition using a two-stage model (Resnet+MSTCN), and reached 89% accuracy on their test set. They used 10 phases, which are slightly different than what we propose in this work (12 phases), and Cholec80 (7 phases). They also annotated their videos based on case complexity and showed that the performance of the model drops from 92% (the simplest case) to 81% (the difficult case). Finally, they tested their model’s performance for robustness to different adverse events. The performance dropped the most when there was major bile leakage, and it was affected less by cholecystitis. These kinds of robustness

experiments are crucial to understanding the model performance under real-world difficulties. Recently, a two-stage model, which used graph networks for phase recognition in Lap Chole videos, was introduced. They reported 93.77% accuracy in the test split of the Cholec80 dataset, which was the top score reported to date. They argued positional encoding had key importance to improve their results.

Recently, transformer networks are gaining much popularity. The main reason for their popularity is that they are showing several advantages over CNNs. For example, they can relate temporally longer information in a sequence. Moreover, they compute attention between every element in a sequence. Higher attention weights indicate a closer relationship between two vectors, while the opposite indicates lower relation. This improves the explainability of the predictive model, i.e., showing the contribution of every individual or a group of tokens in the prediction. This is important, especially for medical applications where model explainability is important. Recently, Trans-SVNet, a transformers-based architecture that is built upon a TCN-based backbone achieved the top results in Cholec80 benchmark [31]. This method proposed the hybrid embedding aggregation transformer, which aggregates spatial and temporal information by actively querying spatial embeddings from temporal embedding sequences. The whole pipeline was able to show real-time performance. Another recent work demonstrated the benefit of using attention weights where the contribution of individual frames on the predicted result helped explainability of the model's predictions [30]. [36] modified the encoder part of the transformer by fusing features obtained from a single-stage TCN and features provided by the transformer encoder. They presented improved results on the Cholec80 dataset compared to previous methods. In summary, transformer architecture is highly flexible and it can provide real-time performance if designed properly, which could be an important factor for the online deployment of surgical phase recognition.

1.1.3 Learning With Limited Data

Deep learning algorithms are often trained with a large set of labeled data. The performance usually scales proportionally to the size of the labeled data. However, labeling surgical data often requires expertise and a lot of manual effort. To reduce the manual labeling effort, methods based on semi-supervised and self-supervised learning have been proposed in the last decade. A typical semi-supervised setting has few labeled data, and a vast amount of data is unlabeled. A model is trained with the labeled portion and then applied to the unlabeled portion to obtain synthetic labels. The generated labels are progressively merged with the labeled part and the model is retrained [37]. Self-supervised learning in contrast uses no human annotations. These

methods use self-supervision, for example, information that is already available in the data. Then, an auxiliary task is defined and trained by utilizing the self-supervision [38].

Semi-supervised learning methods concerning surgical videos gained high traction during the last few years. One of the first studies used conditional generative adversarial networks (cGANs) to re-colorize medical images as an auxiliary task for self-supervised learning. Their method reduced manual annotation effort for the image segmentation task by 75% and showed similar performance with the fully labeled case. Another work performed self-supervised pre-training of networks for remaining surgery duration (RSD) estimation. The network used inherent timestamps as self-supervision to predict the RSD. This pre-training initialized the network trained for the target task, phase recognition. The pre-training helped achieve a similar performance with fully manual annotation by 20% fewer manual annotations. Two other studies followed, performing similar self-supervised pre-training approaches [39, 40]. In [41] unpaired image-to-image translation is used to generate synthetic images of laparoscopic video frames to increase annotations for training. The generated synthetic images were gathered to form a dataset to pre-train a network, which is used for liver segmentation. A teacher-student approach was proposed in [42]. The teacher network that has a higher complexity was trained offline using a small set of labeled videos, and then it predicted labels for a larger set of unlabeled videos. A student network that has a lower complexity was trained by these synthetic annotations. Training on additional synthetic labels helped the model improve its performance from 0.7 to 0.73 in terms of f1-score, closing the performance gap compared to training with all manual annotations (an f1-score of 0.78). Recent work used only a single timestamp per phase as annotations [43]. The timestamp annotation is then propagated to the nearby frames with an iterative method until the beginning and the end of the phase were reached. Compared to full manual annotation, it showed comparable performance with a decrease of 74% manual annotation effort. In another recent work, four popular self-supervised learning methods used for other vision tasks were explored for surgical vision tasks [44]. Their experiments found that correct parameterization of these methods yielded up to a 14 percent performance increase compared to other semi-supervised learning approaches for surgical phase recognition.

DNNs are much deeper compared to traditional fully-connected networks, which require many more parameters to be estimated. Hence, they might lead to sub-optimal features when trained with small-scale data from scratch. Often, transfer learning is applied to tackle this problem [45]. Transfer learning is also important for surgical vision tasks because there is often a high data imbalance in surgical vision datasets [46–48]. With this regard, transfer learning across different surgeries has

been performed in [49]. Transfer learning from a model trained on a large-scale Lap Chole dataset (1243 videos) was fine-tuned with fewer data from Cholec80. Thanks to training the model on a large-scale source dataset, the performance of fine-tuning with much less data on a different Lap Chole dataset (Cholec80) was comparable to training the model using all labeled data in the target [10]. In their later work, the same group used transfer learning with a different modeling approach, to fine-tune the model trained with the Lap Chole dataset with different surgery types where data was fewer. Target datasets were surgeries of various types: Right Hemicolectomy, Sleeve Gastrectomy, and Appendectomy. They first trained their network with self-supervision, on the sequence sorting task. They fine-tuned the trained model for surgical phase recognition on other surgery types mentioned above. They showed that training the self-supervised learning task with the Cholecystectomy dataset helped slightly more than training it with other surgery datasets. This showed that a larger data scale is a key factor for better generalization.

1.1.4 Performance Evaluation

Relatively few works discussed performance metrics tailored for real-world deployment. In [50], application-dependent scores (AD-scores) are proposed. The standardized accuracy, precision, and recall metrics are recomputed by including the transitional time delay threshold. This was shown to be a more clinically meaningful metric because the use of the temporal delay threshold offered more flexibility which can be adjusted depending on the use case. In the Endonet [3] and ST-CNN works [24], temporal boundary distance is used as a performance metric. This metric sets a threshold for the temporal offset between the ground truth segment and the predicted segment. A predicted segment is accepted if it begins within the thresholded distance, otherwise rejected. Basically, temporal shift up to a certain level is tolerated with this metric, which can make it useful for practical applications. Another work used the same metric with a different naming [10]. In another work, the classification performance is assessed based on the duration of the phases [33]. The phases are grouped according to their durations, and performance according to each group was reported. This way, the performance in short phases is explicitly presented, which can be important for a specific use case. Segmental overlap and edit score are proposed in [51]. The first metric penalized over-segmentation errors, which means a predicted block overlapped with the wrong ground truth. The second metric penalized the wrong order but allowed smaller temporal delays. The performance is tested both on an action recognition dataset and a simulated robot-assisted operation, for surgical action recognition. Task-based efficiency metrics are proposed in [52]. In their work, they showed that besides using only standard accuracy for evaluation, surgeon

efficiency metrics were necessary to evaluate the performance in a robot-assisted procedure.

1.2 Objective of the Thesis

This thesis aims to conduct a data-centric study for surgical phase recognition. The study has two main objectives. The first is to show the extent to which differences in difficulty between datasets affect the performance of surgical phase recognition models. We will then discuss the feasibility of establishing and using the models in the real world. To this end, we will gather a large-scale laparoscopic surgery video dataset that contains all possible real-world difficulties. We will perform a systematic comparison of our own curated dataset and the most commonly used publicly available dataset to demonstrate the difference in the characteristics of videos and surgical phases. Next, a detailed comparison of two state-of-the-art deep CNN models will be demonstrated using a wide variety of metrics. Finally, we will present a detailed performance assessment in terms of surgical phases, videos, as well as aggregated metrics, using one of these methods. We will discuss our results, performance metrics, and the challenges of real-world deployment.

The second part of the thesis aims to study how the knowledge of a model trained with much larger data and human annotations available at a particular medical center helps to perform better on the data of a medical center providing much fewer data. We will report best practices to fine-tune a model with fewer data samples but having similar data characteristics with a larger dataset on which the source model is trained. To this end, we will test the transferability of the model trained on our curated dataset to the publicly available Cholec80 dataset. The effect of different parameters for fine-tuning will be investigated and best practices will be presented.

Our research objectives for surgical phase recognition will be implemented with the following steps:

1. A large-scale dataset of 384 laparoscopic cholecystectomy videos will be collected from Inselspital, University Hospital of Bern.
2. Annotation guidelines will be described based on the definition of surgical phases that appear in videos of our collection.
3. Each video in our collection will be annotated following the prepared annotation guidelines.
4. Cholec80 dataset will be obtained. In particular, 80 surgical videos and their annotations will be downloaded from the publicly available download link.

5. Total number of videos and video durations in each dataset will be extracted and compared.
6. Datasets will be compared based on the distribution of individual surgical phases. In particular, the existence of individual phases and their durations, as well as the transition between phases will be compared.
7. Two state-of-the-art deep neural network architectures will be implemented.
8. Our custom dataset will be partitioned into independent training and testing sets. The models implemented in the previous step will be trained individually on the training partition of our dataset.
9. The models trained on our training partition will be evaluated with the testing partition, using a wide variety of metrics such as precision, recall, and f1-score. Segment-based metrics will also be included, which will evaluate the models based on segmental errors rather than individual frame-based errors.
10. The practical use of our results, as well as the limitations of our study, will be discussed.
11. The original Cholec80 training dataset will be divided into 8 overlapping subsets. The first subset will contain the first 5 videos, and each new subset will add the next 5 videos from the original set.
12. The two-stage model trained on the dataset gathered at Inselspital, will be fine-tuned on each Cholec80 subset prepared in the previous step.
13. At each Cholec80 subset, the results of the model fine-tuning will be reported in terms of various performance metrics.
14. The performance of model fine-tuning will be reported for individual phases and videos of the Cholec80 test set.
15. The results of model fine-tuning will be discussed and the limitations of the study will be presented.
16. The thesis will be concluded with final remarks and directions for future research.

1.3 Original Contribution

Previous work in surgical phase recognition aimed at developing new model architectures to improve the classification performance on public datasets, the most

popular being the Cholec80 [3]. However, developing and comparing models only on a single public dataset may not provide us with sufficient clues to set up in the real world. We believe that increasing the amount, variety, and quality of the data appropriately is more important to deploy the models in the real world, rather than investing heavily in developing complex models which will never be fed with a sufficient amount of data. Therefore, in this thesis study, we aim at doing a data-centric study for surgical phase recognition. We focus more on the characteristics of datasets to compare and contrast the differences in their data distributions. Moreover, we aim at doing an in-depth performance analysis of a model with a wide array of performance metrics, to demonstrate the challenges in a large-scale real-world dataset containing 384 Lap Chole videos. Our objective is to shed a light on the differences between a complex real-world dataset and the most commonly used publicly available dataset, and consequently how these differences affect the model performance. We believe our results will present important guidance for real-world deployment. For researchers who solely develop models on publicly available datasets and constantly ignore real-world challenges, our results will shed light on the difficulties of developing models that will be deployed in real-world conditions.

In the second part of our study, we studied how a two-stage surgical phase recognition model transfers from one particular dataset to another. This problem is important because in most practical cases data is not balanced across datasets. This is problematic especially when there is a need to deploy a model to a medical center where data is quite limited. In this work, we investigated the limits of fine-tuning in a target dataset where data is scarce.

In this thesis, our contributions can be listed as follows,

- There are 80 Lap Chole videos in the most commonly used publicly available dataset, Cholec80. In this thesis, we developed a new dataset containing 384 Lap Chole videos, which is far larger than Cholec80.
- Cholec80 videos were curated on purpose. Videos having complex phases and alternating workflow were eliminated. On the contrary, we have not made any reductions in our videos. We have fully included all emergency surgeries and training videos that were performed over a continuous period of five years at Inselspital, Bern university hospital.
- We developed a data labeling guide to label the Lap Chole procedures in our dataset. In this guide, we have included procedural phases in the most comprehensive way, in that, we have included procedural phases that were never mentioned in previous studies.
- Contrary to past studies, we performed a comprehensive and systematic

comparison between our dataset and Cholec80. Consequently, we demonstrated the differences between our dataset and the Cholec80, showing that Cholec80 neglects some of the real-world data challenges.

- Previous studies have usually assessed their model’s performance only on the Cholec80 dataset, and in some cases on their private datasets. In this study, we created a large-scale dataset of 384 videos, which is greater than most of the existing private datasets. We trained two state-of-the-art deep learning-based methods on our dataset and extensively compared them against various performance metrics.
- In a long video, classification for a particular video frame may be different than the one in surrounding frames. This frame-based difference creates a noise effect when looking at long video segments. Therefore, these noise points negatively affect the average error calculated by looking at the frame-based error. Instead, the average error calculated by treating video segments as a single sample is less affected by individual noisy classifications. We evaluated the performance according to segment-based errors, in addition to individual frames-based errors. In this way, we have presented a more comprehensive analysis of the performance. In contrast, previous studies have mostly evaluated based on only frame-based errors.
- As a result of performance evaluations, we showed that the performance of procedural phase recognition remained very low for extremely underrepresented phases. It may be very important for surgeons that such rare phases, which are usually encountered in emergency surgeries, are recognized with high success by AI-based methods. We discussed the requirements and possible future work to improve the success of recognizing rare procedural phases.
- We perform a comprehensive study demonstrating the performance of transfer learning for surgical phase recognition across datasets. This problem is interesting when labeled data from a particular dataset or multiple datasets is greater than the data from a target dataset on which the model will be trained and deployed. We perform various experiments and report the performance for different configurations. Our basic methodology is to perform fine-tuning of the model that has been trained on the source dataset where almost 400 videos were available. Our target dataset featured Cholec80. Our experiments investigated how the performance changed when the model is fine-tuned with different amounts of labeled videos. We perform a detailed performance analysis, showing how the performance changes according to individual phases

and videos. The performance of fine-tuning varied greatly between phases and individual videos. We discuss several reasons for this difference, such as the difference in the same tool’s visual appearance across datasets. Our results present learned lessons both for the research community and practitioners who are trying to solve similar problems.

1.4 Terminology in Surgical Phase Recognition

In this section, we describe the concepts related to surgical phase recognition. We will first formulate phase recognition as a machine-learning problem and define each component that will be used throughout the rest of this thesis work.

Problem formulation: Given a surgical *video sequence*, $\mathbb{V}_{1:T} = [I_1, I_2, \dots, I_T]$ of length T , the aim of surgical phase recognition is to predict the *surgical phase* for every *frame* of the sequence, $I_t \in \mathbb{V}_{1:T}$, using a *machine learning model*, $\mathbf{M}(\cdot)$.

A *video sequence* refers to a time series of multi-dimensional signals or video frames. A *video frame* refers to the 3-D signal (height, width, and color channels) at a specific time step of a video sequence.

The typical workflow of the surgical phase recognition pipeline involves two separate stages, and one follows the other. The first stage in which the model $\mathbf{M}(\cdot)$ is built is called *training*. The second stage where the built model in the first stage is used for prediction is called the *inference*. To train and make an inference with the model, a *dataset* is used. The dataset is composed of videos and *labels* associated with those videos. The term *label* can be used interchangeably with *phase*, or *surgical phase* throughout this work. Basically, surgical phases are defined over a discrete, finite set of items, $\mathbb{C} = \{C_1, C_2, \dots, C_N\}$; $y_t \in \mathbb{C}, \forall t \in \{1, \dots, T\}$. Here, y_t denotes the ground truth phase label of the video frame at time step t , and C_i denotes the i^{th} category over a set of N categories.

During training, the model $\mathbf{M}(\cdot)$ receives an input *feature vector* $\mathbf{x}_t \in [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]$, and produces a prediction $\hat{y}_t = \mathbf{M}(\mathbf{x}_t)$. The quality of the prediction $\hat{y}_t \in \hat{\mathbb{Y}}_{1:T} = [\hat{y}_1, \hat{y}_2, \dots, \hat{y}_T]$ is determined by a *loss function* $L(\cdot)$ as shown in equation 1.1,

$$Loss = \frac{1}{T} \sum_{t=1}^T L(\hat{y}_t, y_t), \quad (1.1)$$

where, $y_t \in [y_1, y_2, \dots, y_T]$ indicates the expected output at time step t .

The loss function measures how well the output produced by the model for an input sample aligns with the actual output for that sample. The smaller loss value means a better fit between the predicted value and the ground truth.

The *procedure* term is used interchangeably with *surgery*. A surgical phase recognition

model is said to operate in *online* mode if the model uses only the signals from the past time steps of a sequence to predict a value for the current time step. On the other hand, a model is said to operate in *offline* mode if the model uses both past, current, and future signals in a video sequence to predict the output for the current time step. Online models are used for *online decision support*, whereas offline models are used for *post-operative analysis* purposes. The qualitative performance is shown with color-coded ribbon illustration [53]. An illustration is shown in Figure 1.4.

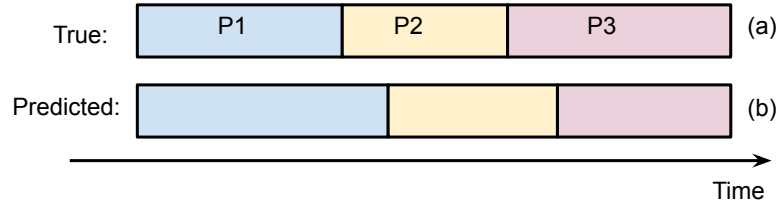


Figure 1.4 Illustration of the qualitative results for surgical phase recognition. Each color bar represents a surgical phase, P1, P2, and P3, and the time axis is represented in the horizontal direction from left to right. (a) Ground truth of a video, (b) Predictions for the same video. A prediction is counted as a misclassification if it mismatches the true color, either on a block or frame basis.

1.5 Challenges in Surgical Phase Recognition

Surgical phase recognition is a challenging problem. Several issues make it harder for real-world deployment.

First, videos are longer compared to the length of the videos of a similar task in the video action recognition and segmentation task. The length of the videos in the latter task often ranges between 2 to 5 minutes[26], whereas the length in the former is in the range of 30 to 60 minutes[54]. Therefore, models developed for the latter task do not easily transfer to the former task.

Second, collecting surgical videos to build a dataset is not as easy as collecting videos from the natural image domain. It is often easier to find videos from daily content, using popular search engines. However, surgical videos are rarely available for public collection. Publicly available datasets are often short of size, and they are available only for research purposes.

Third, data privacy issues make it challenging to collect and archive surgical videos. These videos might contain sensitive information, such as the parts of the patient body or organs, as well as the identity of the surgery staff can be visible in the video footage. Therefore, often additional protocols and agreements are needed to collect this data, which is a big barrier to model development.

Fourth, annotation of surgical videos is costly, and expert staff is hardly available.

Often, it is trivial to annotate things in a daily video. For example, tagging a video frame for its content, or labeling the identity and geometry of objects are trivial tasks. Surgical videos, in contrast, require domain knowledge and extensive experience for annotation. This is another big barrier to collecting large-scale data related to surgical videos.

Fifth, although phases of a particular surgery often follow a predefined order, this order may sometimes need to be changed. In case of an emergency, the surgeon may need to apply different steps, which creates an alternative workflow. Moreover, patient anatomy (an obese or skinny patient) can make it difficult for the surgeon to follow the usual phase order. These variances in the data make it harder for the model to learn a robust representation of the phases.

Sixth, different phases in a laparoscopic surgery often contain similar visual content. This is mostly because the operation takes place in a narrow region in the abdomen, and similar parts of the abdomen are captured several times across different phases. This causes confusion for the AI model during training. Using temporal models even come short because the temporal distance between similar images can be large.

Seventh, instruments used for the same task are often different across hospitals (or datasets). Due to the different branding of a clip applier used for the clipping task, or an electric hook used for dissection, the visual appearance of these tools can change dramatically. This creates an extra challenge for the AI model to recognize these instruments, hence, the phases across hospitals. We will show example cases of the difference between instruments in the third chapter of this thesis.

Eighth, surgical videos contain several types of degradations, such as smoke, motion blur, camera jitter, illumination, and zoom effects. For example, Lap Chole videos contain smoke caused by cauterization. The laparoscopic camera is held by an assistant surgeon during the surgery. Since the camera is not fixed, motion blur and camera jitter often happen due to the unintentional movement of the camera. The zoom and illumination changes usually take effect intentionally, for example, when the focus is shifted to a particular location because of an identified problem or an urgent matter. All of these variations are challenging factors for the modeling.

1.6 Outline

This thesis is organized as follows.

Chapter 1 presents the introduction and motivation of this work. Then, a detailed review of existing methods for surgical phase recognition is presented. The chapter is finalized by our research objectives and contributions.

Chapter 2 reviews the basics of neural networks, and existing deep neural network architectures used for surgical phase recognition.

Chapter 3 reviews the existing datasets used for surgical phase recognition, including publicly available and private datasets. Next, our curated dataset, Insel, is introduced and described in detail.

Chapter 4 describes the methodology and results of our main work: a systematic comparison of our curated dataset and the Cholec80 dataset. Next, the detailed performance analysis and comparison of the state-of-the-art models is presented. The study is concluded with a discussion of the results and the limitations of the study.

Chapter 5 describes our work on cross-dataset transfer learning for surgical phase recognition. Our fine-tuning methodology is described in the two-stage modeling pipeline, and results are presented. The chapter is concluded with a discussion of the results.

Chapter 6 concludes the thesis with final remarks and presents research directions for future work.



2

DEEP NEURAL NETWORKS FOR SURGICAL PHASE RECOGNITION

The methods we will discuss in this thesis are based on deep neural networks (DNNs). In this chapter, we will review popular deep neural network architectures used for surgical phase recognition. We will begin by reviewing the basics of neural networks and how learning happens, then we will review deep CNN-based architectures. Next, we will review a recently popular temporal convolutional network-based (TCN) architecture. We will close the chapter by reviewing a recently proposed transformers-based network.

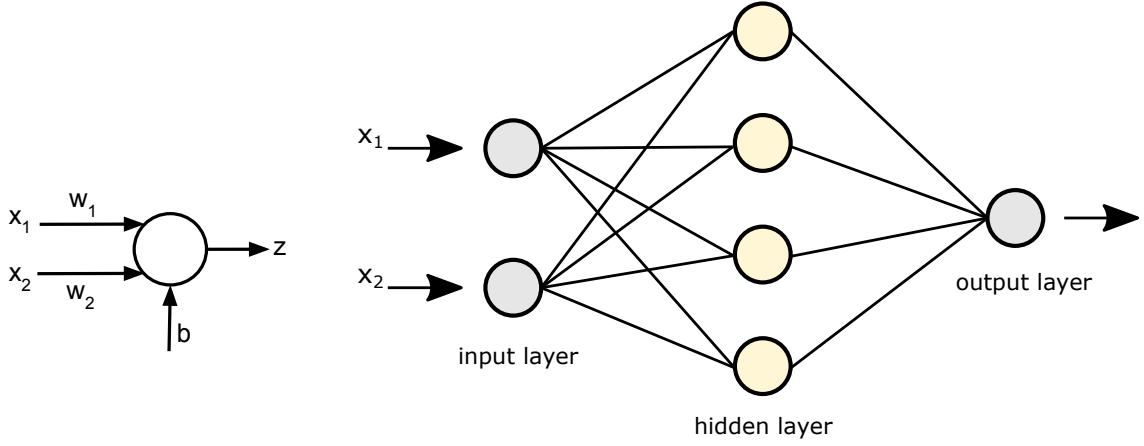
2.1 Basics

Neural networks (NNs) consist of interconnected computational blocks organized as layers. Each layer receives an input, transforms it, and passes it to the next layer. The basic building block of a layer is called a neuron. A neuron is capable of performing a linear transformation of its input and passes its output to the neurons it is connected to in the next layer. This way of computation and message passing is inspired by the functioning of biological neurons in the human brain.

A neural network contains multiple neurons in each layer. Each neuron has connections with all the neurons in the previous and the next layer. The schematic representation of this structure, called the fully connected structure, is shown in Figure 2.1. In the following sections, non-fully connected architectures will also be discussed. The computation that happens in a simple neuron (Figure 2.1(a)) can be formulated as follows,

$$z = w \cdot x + b, \quad (2.1)$$

where $x = [x_1, x_2]$ is the input vector, $w = [w_1, w_2]$ is the weight vector, and b is the bias value. The output is obtained by taking the dot product between the input and the weight vector, and then the scalar bias is added to the output. Our example here considers only a two-element input vector, but it can be generalized into an arbitrary



(a) An illustration of a single neuron.

(b) An illustration of a fully-connected neural network. Figure adapted from [55], CC-BY-1.0

Figure 2.1 Illustration of basic neural network structures. (a) How a single neuron works, (b) Fully-connected neural network organization. Each neuron is connected to each neuron in the previous and next layers.

length vector, $x = [x_1, \dots, x_N]$; $x_i \in \mathbb{R}, \forall i \in \{1, \dots, N\}$. This holds similarly for the weight vector, $w = [w_1, \dots, w_N]$.

The depth of a NN is usually related to its number of hidden layers. One can add as many hidden layers as possible, depending on the complexity of the problem. A NN is said to have L layers when it contains L hidden layers. The weights connecting outputs to the input neurons in the l^{th} layer of a NN are often expressed in the matrix form, as shown below,

$$\mathbf{W}_l = \begin{bmatrix} w_{11} & \dots & w_{1K} \\ \vdots & \ddots & \vdots \\ w_{M1} & \dots & w_{MK} \end{bmatrix} \quad (2.2)$$

where, w_{ij} denotes the weight value connecting the i^{th} neuron in the outgoing layer, $i \in \{1, \dots, M\}$, to the j^{th} neuron in the incoming layer, $j \in \{1, \dots, K\}$. Then, the forward pass equations of a layer l that computes the output vector z_l is formulated as a matrix-vector product,

$$z_l = \mathbf{W}_l^T \cdot a_{l-1} + b_l, \quad (2.3)$$

where a_{l-1} is the output vector of layer $l - 1$, or, the input vector of layer l . This equation computes the raw output at layer l , z_l , which is then transformed with a nonlinear function called the *activation function*, h ,

$$a_l = h(z_l). \quad (2.4)$$

The dimensions of vectors z_l , a_{l-1} and b_l in equation 2.3 are K , M , and K , respectively. This is because the l^{th} layer has M inputs (the dimensionality of vector a_{l-1}) and K outputs (the dimensionality of vectors z_l and b_l). The forward equations are repeated for every layer $l \in \{1, \dots, L\}$, and the final output is obtained at the output layer of the NN.

The output of a NN obtained for a particular input is compared to the desired output and the error is computed. This computation is performed by a special function called the *loss function* (equation 1.1). One particular loss function that works well for binary inputs is the *binary cross-entropy* loss. Let $X = \{x_1, \dots, x_N\}$ be a set of N data samples, and $Y = \{y_1, \dots, y_N\}$, $y_i \in \{0, 1\}$ are the actual binary outputs. The binary cross-entropy loss is defined as,

$$L(\mathbf{W}) = \frac{1}{N} \sum_{j=1}^N y_j \ln(h(z_j)) + (1 - y_j) \ln(1 - h(z_j)) \quad (2.5)$$

where $\ln(\cdot)$ is the natural logarithm, $h(\cdot)$ is the activation function, and z_j is the network output for j^{th} data sample. The activation function $h(\cdot)$ is usually the sigmoid function,

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \quad (2.6)$$

which is a non-negative function that squashes its input values to between zero and one.

Once a loss function has been defined the optimization of the NN becomes the minimization of the loss function. One way to do this is to use the gradient descent method [56]. During gradient descent, the input is forwarded through the network layers and the trainable parameters of the network, also called weights, are updated. The magnitude of weight updates depends on the loss value calculated by the loss function described above. The weights and biases of the network are first updated at the output layer. After that, the same update rule is applied to the layer before, and this process is repeated until the first layer. This iterative algorithm is called the *backpropagation* algorithm, which is widely used in practice.

2.2 Convolutional Neural Networks

DNNs trained for visual tasks are specifically called Convolutional Neural Networks (CNNs). CNNs process the data end-to-end in a hierarchical manner. Image features are extracted in every intermediate layer in the hierarchy by applying convolutional kernels [5, 15]. Each successive layer transforms the input, changing its shape and content, and passes it to the next layer. This process is repeated until reaching the final

classification layer. The general structure of CNNs is illustrated in Figure 2.2. The final

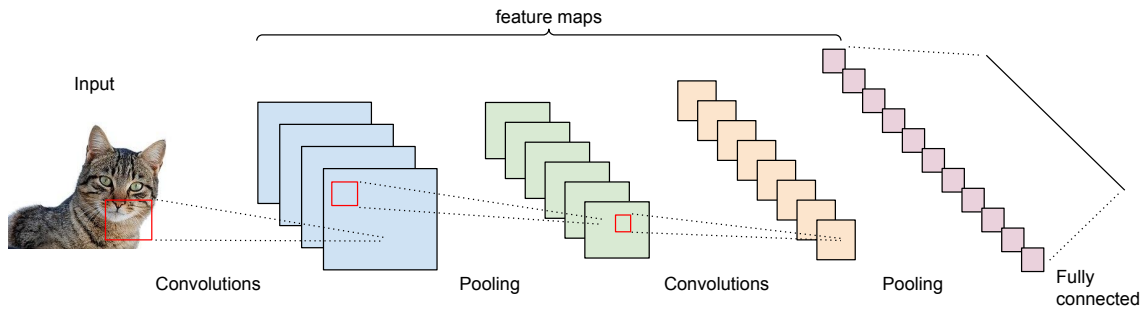


Figure 2.2 An illustration of the CNN architecture. The network performs a series of spatial convolution and pooling operations following one another. Each operation results in a new set of feature maps with reduced size.

layer before classification results in the feature representation named the *embedding*. When presented for classification, the embedding represents a highly compressed and discriminative representation of the original input data, which improves classification performance both in terms of computational efficiency and accuracy. Not only the final embedding is useful for recognition tasks, but features extracted from different intermediate layers of a CNN can also be useful. Eventually, features show different properties depending on which layer in the hierarchy they are extracted from. The earlier layers of a network learn low-level features such as edges and corners, whereas the middle and final layers of the network learn mid and high-level features, which are more specialized representations of the class of the input sample. This new approach for processing visual information achieved top scores in the popular ImageNet challenge [57]. The classification or regression is performed by training an additional classification layer on top of the embedding. For multi-class classification, the raw scores obtained in the output layer are converted to normalized probability scores by using the softmax function, or by using the sigmoid function in the case of binary classification.

CNNs have several properties that make them appealing to be used for high-dimensional visual data, such as weight sharing, local connectivity, and translation invariance. All of these properties reduce the number of trainable parameters which helps reduce the risk of overfitting.

The first successful implementation of deep CNNs was the AlexNet architecture, which was proposed in 2012, and won the first ImageNet image classification challenge, presenting a large performance gap between the other competitors [5]. This architecture was built out of 5 convolutional layers and 3 fully connected layers in addition to mid-level pooling layers. The pooling layers reduce the spatial resolution of the feature maps, which eventually drops the computational time. Moreover, dropout

regularization, which was shown to be highly effective to improve classification performance, was first introduced in that work. The significant success provided by AlexNet shifted the attention of researchers to develop new deep neural network architectures. The network architecture developed at Google introduced a 22-layered network [58]. Its special design, which was named Inception, allowed the network to grow deeper and wider without adding computational overhead. This architecture topped the leaderboard in ImageNet large-scale visual recognition challenge (ILSVRC) in 2014. Another study performed at Oxford university proposed the VGG-16 and VGG-19 CNN architectures that included 16 and 19 layers of networks, respectively [59]. The aim of this study was to test how increasing the depth of the network affects the performance of the model for large-scale visual recognition. To maintain computational efficiency they used smaller (3x3) convolution kernels in that work. Their network has shown top results in the localization and classification tracks of the same ILSVRC challenge in 2014. After 2014, the competition between strong research groups in the annually recurring ILSVRC challenge has significantly accelerated performance improvement, which led to many powerful architectures. One of these architectures is Residual Networks (ResNet). We will briefly review ResNet architecture in the next section.

2.3 Deep Residual Networks

Increasing the depth of networks caused two important problems: (i) overfitting issues, and (ii) the vanishing gradient problem. The first problem can be tackled simply by collecting more training data, or by applying different regularization techniques. However, the vanishing gradient problem, which is causing the gradient to decay to zero, and eventually prevents the earlier layers of the network from receiving weight updates, was more critical. To tackle the vanishing gradient problem while still increasing the depth of the network, Residual Networks (ResNet) were proposed [15]. This network architecture, which topped the leaderboard in ILSVRC 2015 competition, implemented the deepest series of networks up to date, namely 34, 50, 101 and 152 layers, respectively. Training such deep layers without being affected by the vanishing gradient problem was possible by using skip connections. A skip connection simply adds an alternative path to a residual block, enabling an easy pathway for the gradient signal to pass through during backpropagation. This is crucial to be able to keep earlier layers of the network learning. Beginning with Resnets, the idea of skip connections influenced many new generations of networks, including DenseNet [60] and U-Net [61], which achieved top performance in later classification and segmentation tasks. A schematic illustration of ResNet architecture is shown in Figure 2.3. The network receives an image of size 224×224 . The dimensions of the image are halved at the

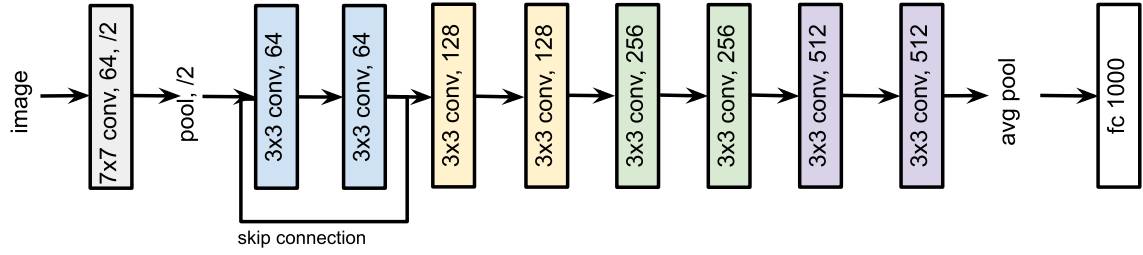


Figure 2.3 A more detailed look into ResNet architecture showing the parameters and interaction of layers. Every color represents a different residual block. The size of each residual block varies in the original architecture. Skip connections are repeated for every block but skipped in this picture for simplicity. (Figure adapted from He et al. [15], CC BY 4.0.)

first convolutional layer. The following pooling layer performs downsampling by two, which eventually leaves feature maps of 56×56 spatial size. Each color represents a residual block parameterized by the number of output channels (64, 128, 256, etc.). These blocks are repeated many times in the original architecture, however only a reduced version is shown in Figure 2.3. Spatial downscaling proceeds until the average pooling layer and the feature map dimensions become 7×7 . Finally, the network outputs a 1000-d vector, where the value in each index represents the probability of the input being a member of the class at that vector index. The network weights are frozen and the final classification layer is often replaced with a new one to train for the classes of the current target task. This is called transfer learning, which is widely applied in practice. Skip connection is the core building component of a ResNet. A zoomed view of a skip connection is illustrated in Figure 2.4.

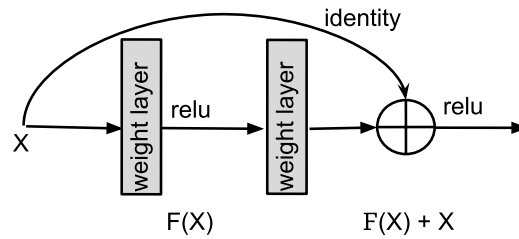


Figure 2.4 An illustration of how "skip connection" works in ResNet. The input signal is transferred from an alternative pathway and added to the signal at the end of the ResNet block. (Figure adapted from He et al. [15], CC BY 4.0.)

2.4 Multi-stage Temporal Convolutional Networks (MSTCN)

Recently, TCNs with large receptive fields have been very successful at segmenting actions in long untrimmed videos [26, 62]. Multi-stage Temporal Convolutional Networks (MSTCN) for example, employ a multi-stage architecture in which initial predictions generated at the first layer are refined at the layers in successive stages.

An illustration of MSTCN is shown in Figure 2.5. Each layer in a stage features

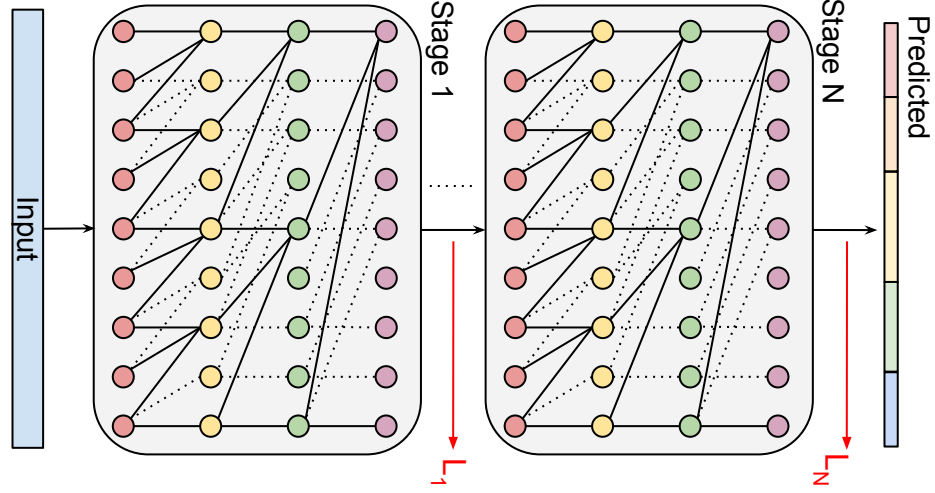


Figure 2.5 An illustration of MSTCN architecture. Each layer features a set of 1D dilated convolutions. The dilation rate is doubled at each successive layer, which in turn increases the receptive field. (Figure adapted from [26, 29])

a set of filters performing dilated temporal convolutions. Thanks to these dilated convolutional layers, MSTCN is capable of handling long-range dependencies with fewer parameters, which makes it an ideal choice for modeling phase information. Residual connections between layers allow for increasing the depth of the network. MSTCN optimizes a loss function that has two components: (i) cross-entropy loss which penalizes the frame-level classification error, (ii) truncated mean squared error (t-mse) loss that penalizes prediction difference in adjacent frames, leading to smoother predictions.

Cross entropy loss is formally defined as follows,

$$L_{ce} = -\frac{1}{T} \sum_{t=1}^T \log(y_{t,c}) \quad (2.7)$$

where T is the total number of video frames, and $y_{t,c}$ denotes the predicted probability for the correct class c at time t . T -mse loss is defined as,

$$L_{t-mse} = \begin{cases} \frac{1}{TC} \sum_{t,c} |\log(y_{t,c}) - \log(y_{t-1,c})|^2, & |\log(y_{t,c}) - \log(y_{t-1,c})| \leq \tau \\ \frac{1}{TC} \tau^2, & \text{otherwise} \end{cases} \quad (2.8)$$

where C is the total number of classes, and τ is the truncation value. This loss function enforces the log probability of the same class between adjacent frames as close as possible, thus helping to reduce over-segmentation errors. As a result, the stage-wise

loss is calculated by a weighted summation of both loss components as follows,

$$L_s = L_{ce} + \lambda * L_{t-mse} \quad (2.9)$$

where λ is a hyper-parameter that adjusts the contribution of the t -mse loss. Stage-wise losses are summed up to obtain the total loss,

$$L = \sum_s L_s \quad (2.10)$$

MSTCN has recently been applied for surgical phase recognition and the two-stage architecture, i.e., ResNet followed by MSTCN, showed very good performance in Cholec80 benchmark [29].

2.5 Transformer Networks

Transformers are sophisticated networks specialized for processing long, sequential data. The architecture was first proposed for the machine translation problem in 2017 [7]. It introduced the attention mechanism that associates different positions of a sequence to compute a representation of the sequence. The architecture depends on the encoding and decoding of vectors. Each word in a sentence is represented as a fixed-length vector and forwarded to the self-attention module in the encoder. During this forwarding, the position of each word in the sentence is preserved. The self-attention module computes the attention score in a pairwise manner, between every vector in the sentence, using the query, key, and value representations. The self-attention results in a transformed vector encoding of each word in the input sentence. In essence, a word's attention score determines how much emphasis is placed on other words in the sentence when encoding that word. The transformed vector encodings are passed through a feed-forward NN layer and they are normalized by layer normalization [63]. Several of these encoders are stacked on top of each other and the output of an encoder is fed into the next encoder on top of it. After the vectors pass through all encoders, they are fed into the decoder part. The structure of the decoder is similar to the encoder. The final output from the encoder (the transformed word vectors) is passed to the encoder-decoder attention layer of the decoder. This way, the decoder learns to relate the output vector to the vectors from the encoder part. Once a word is decoded by the decoder, it is placed in its original position in the input sequence at the bottom-most decoder, and then the decoder is run again to decode the next output word. This is repeated until the topmost decoder outputs the special "end of sequence" token.

This idea proposed by transformer architecture was also applied to the computer vision

field, for image classification [64]. An image is split into 16×16 patches and the self-attention between every image patch is computed. Various transformer-based methods for image recognition have been proposed. Most of these use feed-forward NNs instead of convolutional layers [65, 66]. Some others use convolutions and transformers together [67, 68]. Recently, the latter approach received much traction. The goal was to leverage the strengths of each method. For example, CNN is specialized to extract local features, while the transformer is specialized to extract global attention across the whole sequence. Therefore, applying convolutional computation in the first steps of the data pipeline to extract local features and then applying the transformer to the obtained features yielded better results. Lately, transformers were adapted for video classification tasks [69–71]. These models extend the Vision Transformer (ViT) to the spatiotemporal domain. The spatial and temporal dimensions are split, and image features are extracted with a backbone model. Self-attention is computed between these features along the temporal dimension. Recently, this idea is applied to surgical videos and achieved top performance in Cholec80 dataset [31]. The method named Trans-SVNet used ResNet to extract spatial embeddings and TCN to extract temporal embeddings. Then, spatial and temporal embeddings are projected and fused in a transformer encoder, by computing the self-attention between each spatial embedding and temporal embeddings of a past temporal neighborhood. This is illustrated in Figure 2.6.

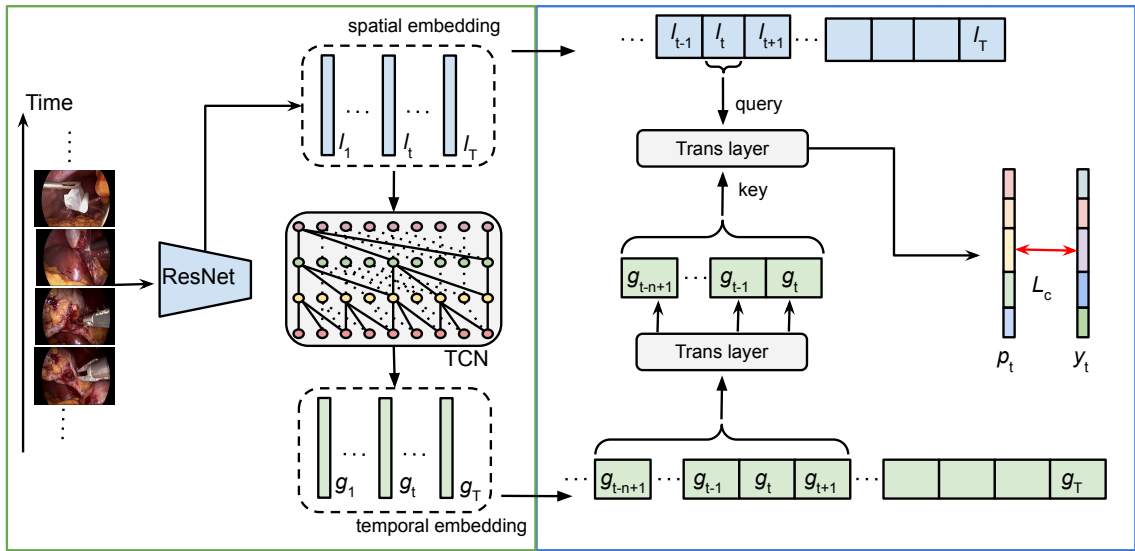


Figure 2.6 Illustration of the Trans-SVNet model. (Figure adapted from [31])

3.1 Overview

A machine learning system consists of three components: (i) the model definition, (ii) the code that implements this definition and runs it on a dataset, and (iii) the dataset itself. The dataset is the key component of a machine learning system because the model learns the relationships between the input signals and the outputs, and extracts patterns from the data. This methodology is called the "data-driven" solution to a problem. For a vision task such as surgical phase recognition, the dataset consists of a collection of surgical videos (or images) and associated annotations (phase labels). A surgical phase recognition model uses the videos, either individual frames or segments as inputs, to learn a mapping between the phase labels.

A particular surgery can be divided into unique procedural phases. A surgical phase defines the events that occur during a specific period of a surgical procedure. The events associated with a particular phase are well-defined so as not to cause misunderstanding or confusion with other events. A surgeon needs to follow these phase definitions, usually in a specific order, to complete a particular surgery. Every type of surgery has its own set of phase definitions, and the phases associated with each one rarely overlap between surgery types.

Besides surgical phases, surgical instruments are also key components of a particular surgery. The instruments act on the tissue or organs in different ways including but not limited to grasping, moving, pulling, cutting, and clipping. Often, multiple of these actions occur during a phase, with cooperation between multiple instruments.

The preparation of a dataset is often performed using an annotation software framework. The software includes a graphical user interface to enable video playback and viewing of the frames. The annotator plays the video and moves between the frames to assign phase labels to segments of the video. The video frames and annotations are associated in a database for scalability and ease of access. This is illustrated in Figure 3.1.

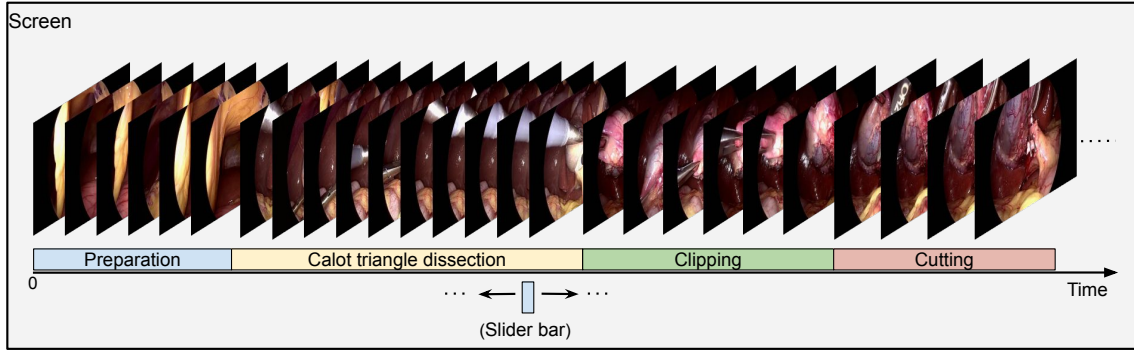


Figure 3.1 Illustration of the annotation software. Frames of a video are shown on the software screen according to their timestamps. The slider bar can be moved in both directions along the time, and phase labels are assigned to the segments of the video. Labels are logged into a database when the task is completed. Images are from the courtesy of Cholec80[3]. Best viewed in color.

In this section, we will review existing datasets of laparoscopic cholecystectomy for surgical phase recognition. In particular, we will review the definitions of surgical phases, as well as the associated surgical instruments. We will share some numbers regarding the duration and distribution of phases and instruments for a particular dataset. Besides existing publicly available datasets, we will also describe our curated private dataset we named the Insel.

3.2 Cholec80

Cholec80 is the largest publicly available dataset provided for surgical phase recognition and instrument presence detection in laparoscopic cholecystectomy videos [3]. The dataset contains 80 videos of varying durations. The videos are recorded at 25 frames per second, and each video frame has 854x480 spatial dimensions. The basic properties of videos in Cholec80 are shown in Table 3.1.

Table 3.1 Basic properties of the videos in Cholec80.

Dataset	Surgery type	Video count	Fps	Dimensions	Duration (minutes)			
					Mean (mean±std)	Median	Min	Max
Cholec80	Lap.Cholec.	80	25	854x480	38±17	35	12	99

Surgical phases in Cholec80 are split into seven categories, as described in Table 3.2. The purpose of laparoscopic cholecystectomy surgery is to remove the inflamed gallbladder from the patient’s abdomen. The Preparation phase starts as soon as the procedure starts. This phase often involves preparatory movements of the surgeon until the actual dissection of the gallbladder. The surgery follows the dissection of the

Table 3.2 Definitions of surgical phases in Cholec80 dataset.

Phase	Definition
Preparation	The period from the start of the video recording until the dissection of the Calot's triangle starts.
Calot's Triangle Dissection	The hepatocystic triangle is dissected to uncover the cystic duct and the cystic artery.
Clipping and Cutting	The cystic artery is clipped and divided by electric scissors.
Gallbladder Dissection	The gallbladder is dissected to separate it from the liver bed.
Gallbladder Packaging	After the gallbladder is separated from the liver bed, it is packed into a bag for removal.
Cleaning and Coagulation	The abdominal space is cleaned from remaining blood clots and scar tissues are coagulated.
Gallbladder Retraction	The gallbladder is retracted to move it to the port for the removal.

hepatocystic triangle to establish the critical view of safety, meaning that the cystic duct, the cystic artery, and the common bile duct are exposed. The cystic artery is clipped and cut to release the gallbladder. Next, the gallbladder is dissected to remove it from the liver bed. After this step, either the cleaning and coagulation of the liver bed or the packaging of the gallbladder occurs. The gallbladder is packed with a special instrument named the specimen bag. Finally, the gallbladder is retracted to remove it out of the abdomen through the removal port. In the following, example video frames from each of these phases are shown in Fig. 3.2. Here, we separated video frames from Clipping and Cutting phases for clarity, however, these phases are described and used as a single phase in Cholec80 practice.

The distribution of phase durations is shown in Fig. 3.3. Each phase duration is normalized with respect to the sum of phase durations in the dataset. As clearly shown in the figure, phase durations are highly imbalanced in the dataset. The dissection phases constitute the longest duration across all phases. Gallbladder retraction is the shortest phase, and it is not always visible in every video, which makes it difficult to decide on its existence during the annotation process. The exact duration of each phase present in the Cholec80 dataset is shown in Table 3.3.

Various surgical instruments are used to perform the surgical phases. These instruments perform actions like pulling/pushing, grasping, dissecting, burning, cleaning, etc. In addition to the phase definitions mentioned above, Cholec80 dataset contains annotations regarding the presence or absence of surgical instruments at a given time (in a video frame). The surgical instruments presented in Cholec80 dataset are listed as follows: (1) Grasper, (2) Bipolar, (3) Hook, (4) Scissors, (5) Clipper,

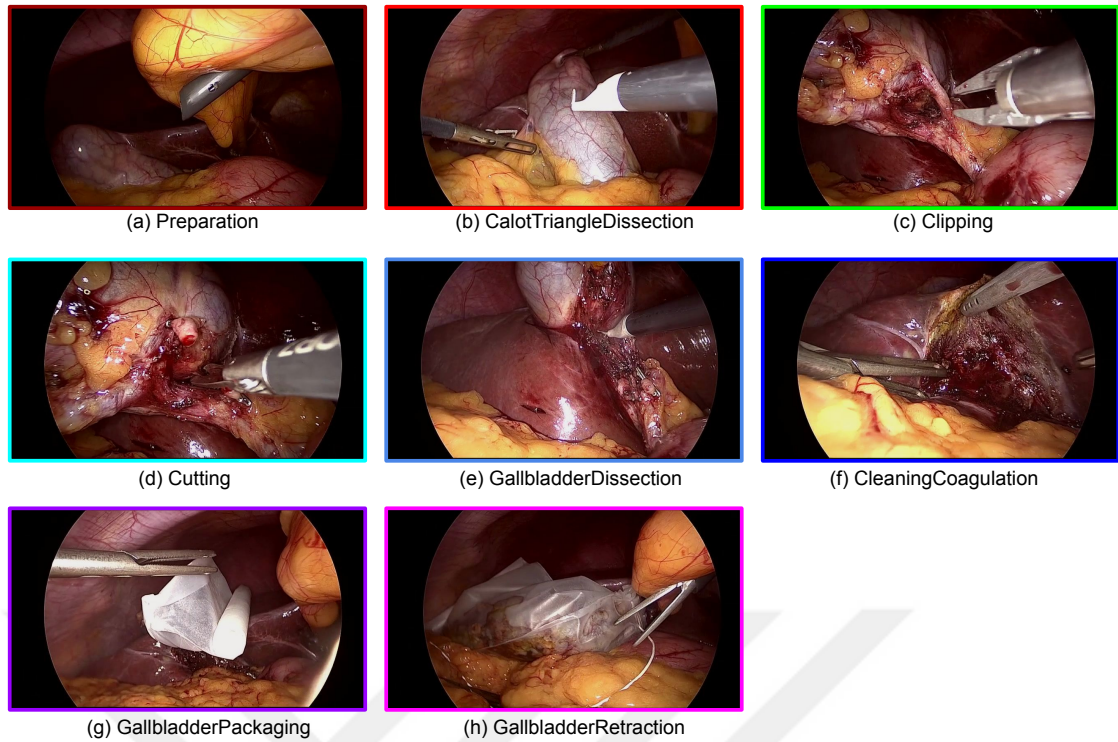


Figure 3.2 Video frames sampled from a video from the Cholec80 dataset, representing each surgical phase described above [3]. Best viewed in color. (CC BY 4.0.)

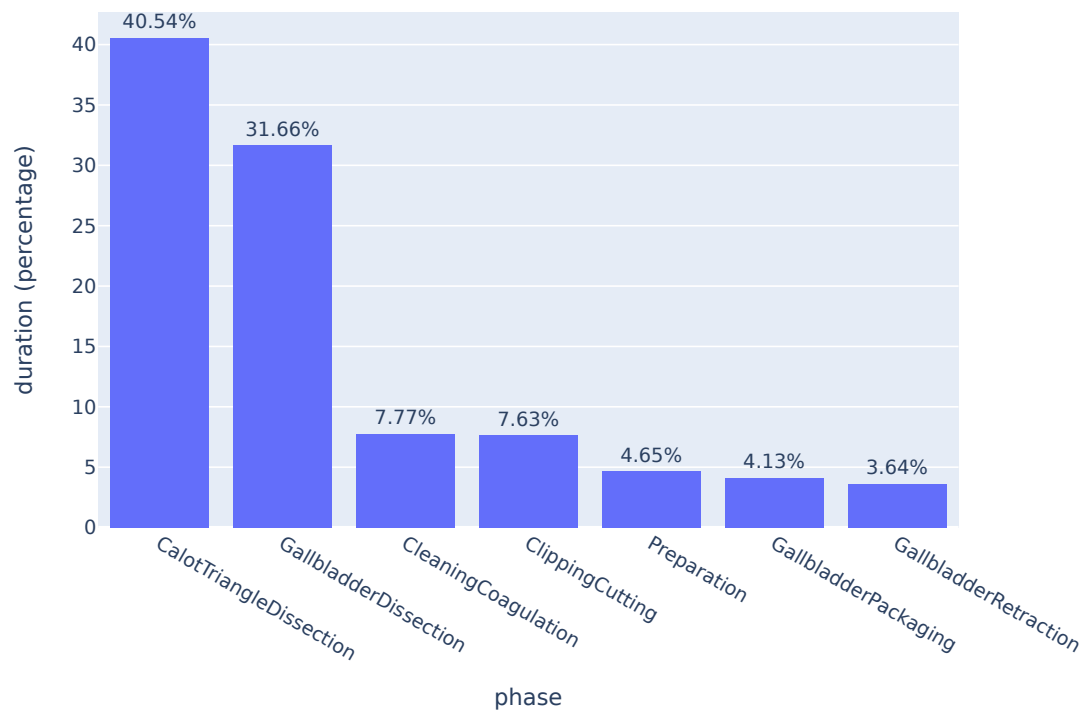


Figure 3.3 Distribution of duration of phases in Cholec80 dataset.

ID	Phase	Duration(secs.) (mean $\pm$ std)
1	Preparation	125 $\pm$ 95
2	Calot triangle dissection	954 $\pm$ 538
3	Clipping and Cutting	168 $\pm$ 152
4	Gallbladder dissection	857 $\pm$ 551
5	Gallbladder packaging	98 $\pm$ 53
6	Cleaning and coagulation	178 $\pm$ 166
7	Gallbladder retraction	83 $\pm$ 56

Table 3.3 The exact duration of each phase (in seconds) present in the Cholec80. The std describes the deviation of the mean across the 80 videos.

(6) Irrigator, (7) Specimen Bag. These instruments work solely in some phases and together in others. For example, during the dissection of the Calot’s triangle, often Grasper and Hook operate together. Grasper might need to stretch the tissue to coagulate it with Bipolar. Visual examples of these instruments as they appear in the Cholec80 dataset are illustrated in Fig. 3.4.

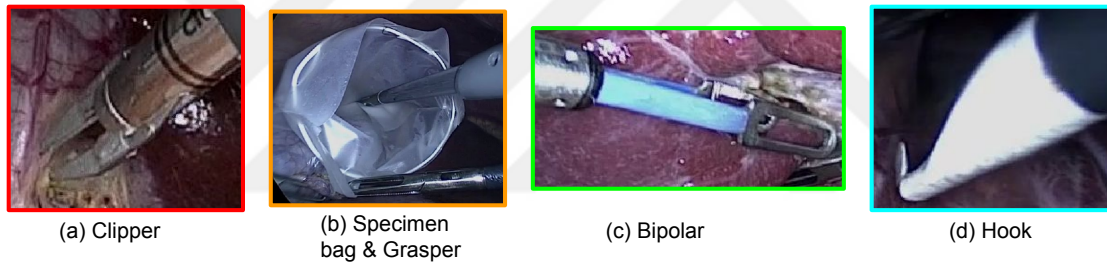


Figure 3.4 Illustration of visual examples of several instruments from Cholec80 dataset. Best viewed in color.

3.3 Insel

We have gathered a large dataset of 384 videos of laparoscopic cholecystectomy provided by our partners in the Department of Visceral Surgery and Medicine at Inselspital in Bern, Switzerland. We named the dataset Insel.

Ethical Approval: The ethics committee of the Canton of Bern approved the study and waived the need to obtain informed consent (KEK 2018-01964). All methods were performed according to the relevant guidelines and regulations.

The videos were selected from the surgeries performed between January 2014 and May 2019 at Inselspital. All video recordings including additional procedures, and emergency or teaching cases were included in the dataset.

The videos were recorded at 25 frames per second and had varying durations. Video frame dimensions had two different values: 1280x720 and 720x576. The basic properties of videos in the Insel dataset are shown in Table 3.4.

Table 3.4 Basic properties of the videos in Insel dataset.

Dataset	Surgery type	Video count	Fps	Dimensions	Duration (minutes)			
Insel	Lap.Cholec.	384	25	1280x720 720x576	Mean (mean $\pm$ std)	Median	Min	Max
					69 $\pm$ 49	54	13	374

Videos were annotated for a single surgical phase label per frame. To perform phase annotations, we adapted the guidelines provided by Endovis Surgical Workflow and Skill Analysis sub-challenge 2019 [72]. In collaboration with our partners, we defined additional phases, such as *gallbladder drainage*, *liver biopsy*, and *intraoperative cholangiography*, which were not present in Endovis guidelines. Annotations were completed in three rounds: (1) videos were shared among five annotators to complete the first round of annotations, (2) a cross-check was performed to raise indecisive cases, and (3) indecisive cases were resolved in a final review round. Surgical phases in our Insel dataset are split into twelve categories, as described in Table 3.5.

In the following, example video frames from each of the major phases are shown in Figure 3.5. As shown, the background view in closer phases is similar. For example, (b) Calot Triangle Dissection and (e) Gallbladder dissection, or (c) Clipping and (d) Cutting. These phases usually occur at the same place in the abdomen, i.e., where both the gallbladder and liver are visible in the view.

Example video frames from the second set of phases that happen less frequently than the phases in Figure 3.5 are illustrated in Figure 3.6.

The duration of phases is highly imbalanced in the dataset (Figure 3.7). *Calot triangle dissection* and *other procedure* phases cover 58% of the total duration in the dataset. *Liver biopsy* and *gallbladder drainage* are the two shortest phases, covering only 0.06% and 0.15% of the total duration, respectively. The exact duration of each phase present in the Insel dataset is shown in Table 3.6.

As demonstrated in Figure 3.5, surgical instruments have varying appearances. Almost all of the instruments except the *grasper* (bottom left in Figure 3.5(c)) and the *suction-irrigation tool* (Figure 3.5(f)) show strong visual differences from the instruments present in the Cholec80 dataset. The biopsy tool (Figure 3.6(b)), the *biliary drainage tube* used during intraoperative cholangiography (Figure 3.6(a)), and the drainage tool used for gallbladder puncture (Figure 3.6(c)) do not exist in Cholec80. The shape

Table 3.5 Definitions of surgical phases in our Insel dataset.

Phase	Definition
Preparation	The period from the start of the video recording until the dissection of the Calot's triangle starts.
Calot Triangle Dissection	Begins as soon as an instrument appears in the image. The phase includes the dissection of connective tissue and fat around the gallbladder and the nearby abdominal cavity to reach the Calot's Triangle.
Gallbladder Drainage	A metal needle is used to puncture the gallbladder before the dissection of Calot's Triangle, to reduce its volume if it is too full to be grasped. This phase ends as soon as another instrument for dissection/coagulation enters the picture.
Clipping	Begins with the appearance of a clipper, which clips the cystic artery or the cystic duct before cutting.
Cutting	Begins with the appearance of scissors, which cuts the cystic artery or the cystic duct after clipping.
Gallbladder Dissection	Begins after cutting the cystic duct and artery, as soon as the preparation scissors or the electric hook touches the gallbladder.
Gallbladder Packaging	Begins as soon as the specimen bag enters the picture. This phase ends with the complete closure of the bag or if the focus is diverted from the specimen bag to start the following phase.
Cleaning and Coagulation	Begins as soon as the focus of the camera is averted from the specimen bag to check for tissue damage, coagulation begins using electricity, or drainage enters the picture.
Gallbladder Retraction	Begins when the specimen bag is finally grasped to be removed from the abdomen.
Liver Biopsy	Tissue samples are collected from the liver, i.e., generally towards the end of the procedure.
Other Procedure	Begins once non-cholecystectomy-related surgical procedures such as intraoperative ultrasonography, and liver or pancreas resection are performed. This phase ends as soon as another instrument for dissection/coagulation/packaging enters the picture.
Intraoperative Cholangiography	The cystic duct is cut and a plastic tube is inserted to apply contrast medium for intraoperative cholangiography. This is an X-ray imaging of the biliary tract to check whether gallstones block the common bile duct and/or the biliary tract is dilated. This phase ends as soon as another instrument for dissection/coagulation enters the picture.

and color of the *electric hook dissector* (Figure 3.5(b), (e)), the *clip applier* (Figure 3.5(c)), the *scissors* (Figure 3.5(d)) and the *specimen bag* (Figure 3.5(g)) are different in Cholec80. Comparison in terms of other instruments and features between the Cholec80 and Insel will be performed in the next chapter.

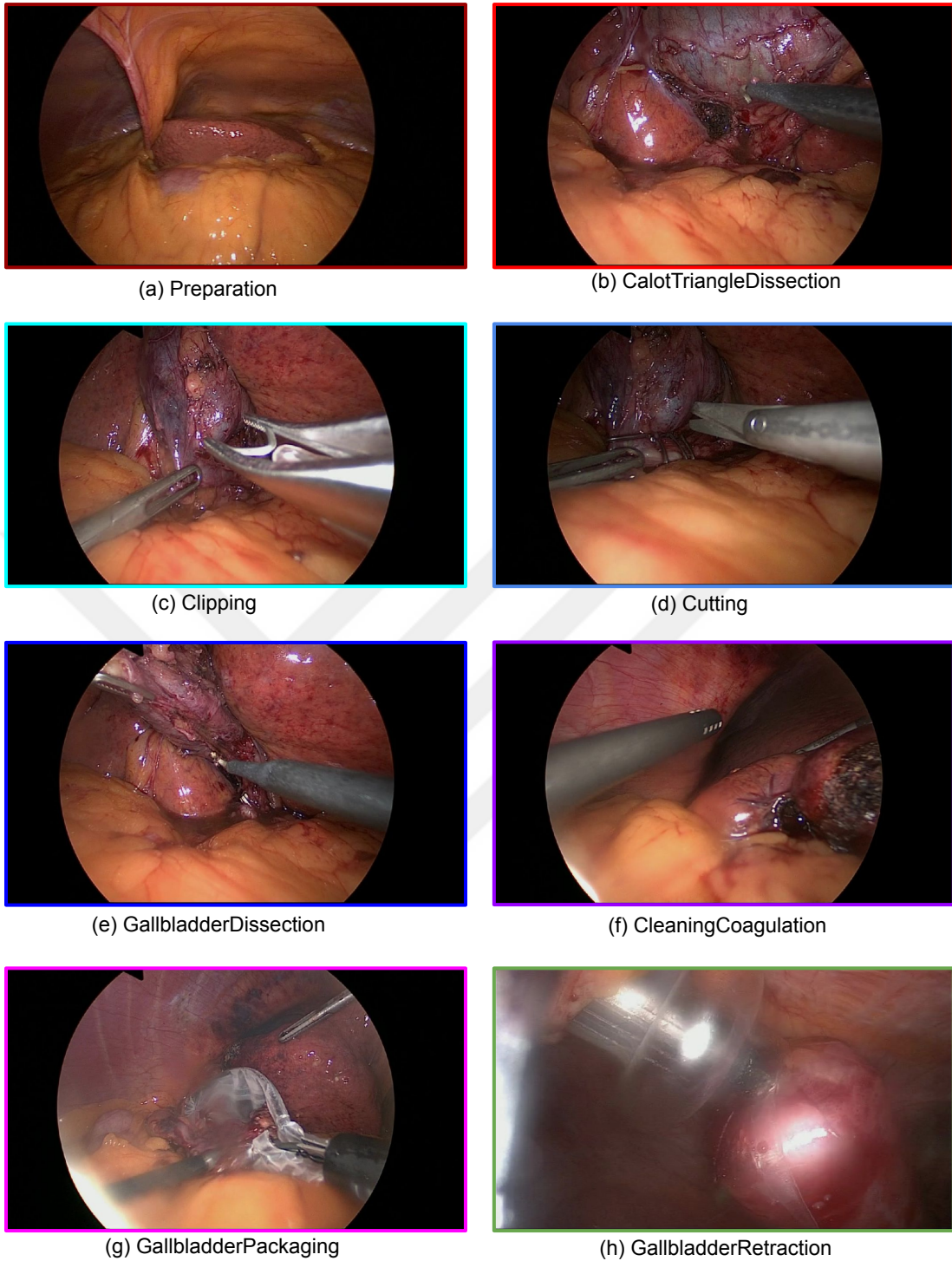


Figure 3.5 Example video frames of major phases in Insel dataset. Best viewed in color.

Examples of other instruments present in our Insel dataset but not shown in Figure ?? are shown in Figure 3.8. The difference in appearance, shape, color, or presence and absence of these instruments pose a challenge for modeling as we will discuss in the next chapters.

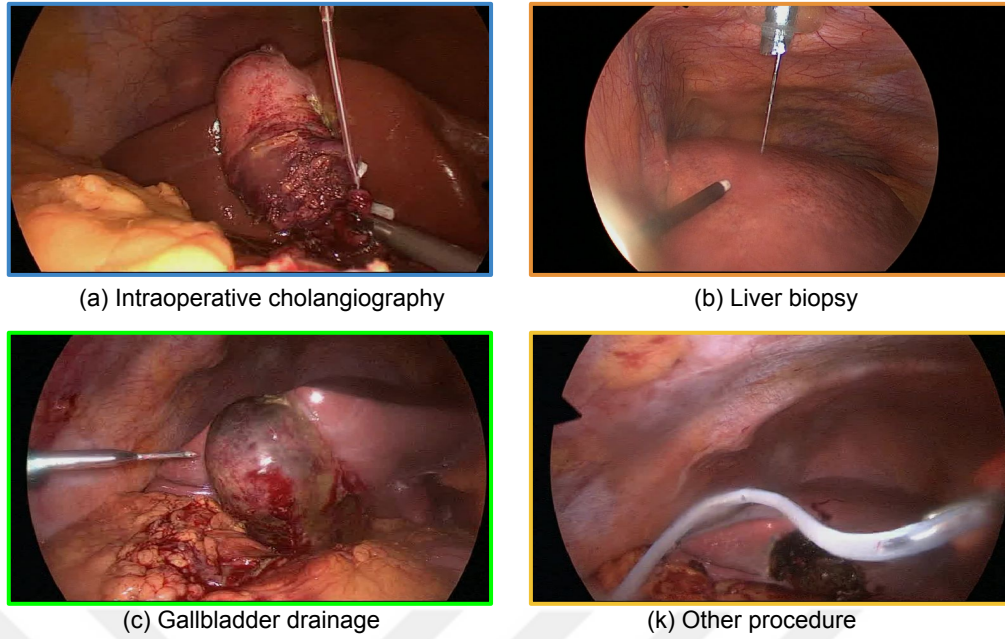


Figure 3.6 Example video frames of phases that do not happen regularly in Insel dataset. Best viewed in color.

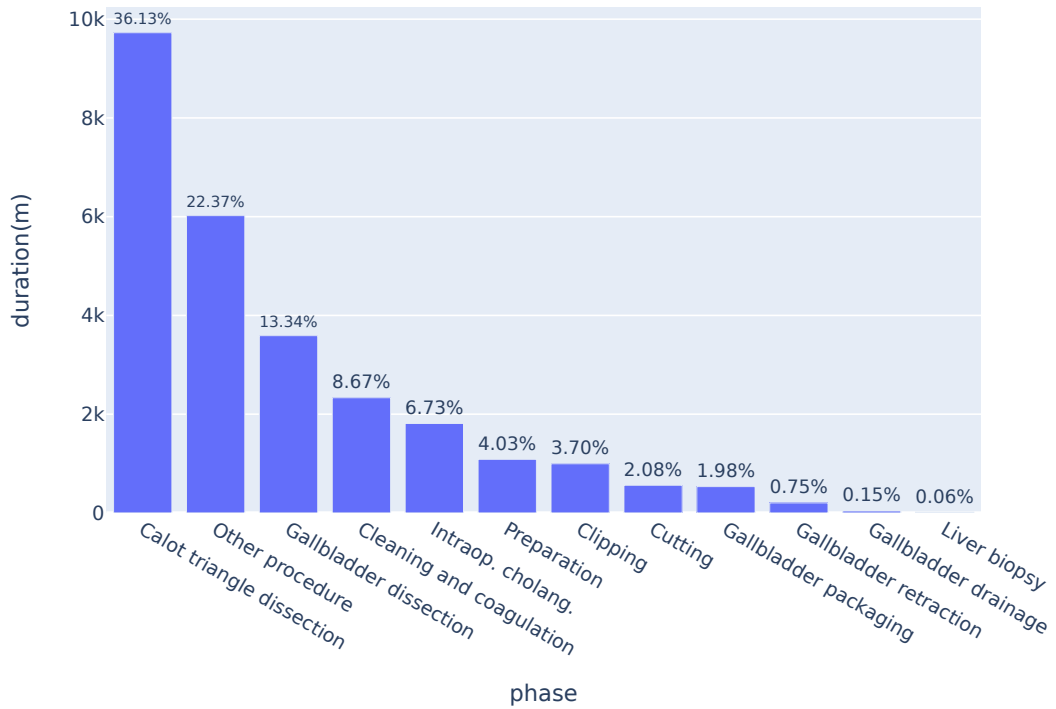


Figure 3.7 Distribution of phase durations in Insel dataset.

3.4 HeiChole

The HeiChole benchmark dataset was released for the international Surgical workflow and skill analysis challenge organized by EndoVis in 2019 [72]. The

ID	Phase	Duration(secs.) (mean $\pm$ std)
1	Preparation	201 $\pm$ 113
2	Calot triangle dissection	1531 $\pm$ 947
3	Gallbladder drainage	93 $\pm$ 38
4	Clipping	162 $\pm$ 141
5	Cutting	91 $\pm$ 102
6	Gallbladder dissection	585 $\pm$ 461
7	Gallbladder packaging	92 $\pm$ 95
8	Cleaning and coagulation	414 $\pm$ 375
9	Gallbladder retraction	112 $\pm$ 114
10	Liver biopsy	59 $\pm$ 35
11	Other procedure	5313 $\pm$ 3665
12	Intraoperative cholangiography	2313 $\pm$ 1980

Table 3.6 The exact duration of phases in our Insel dataset (in seconds). The std describes the deviation of the mean across the videos.

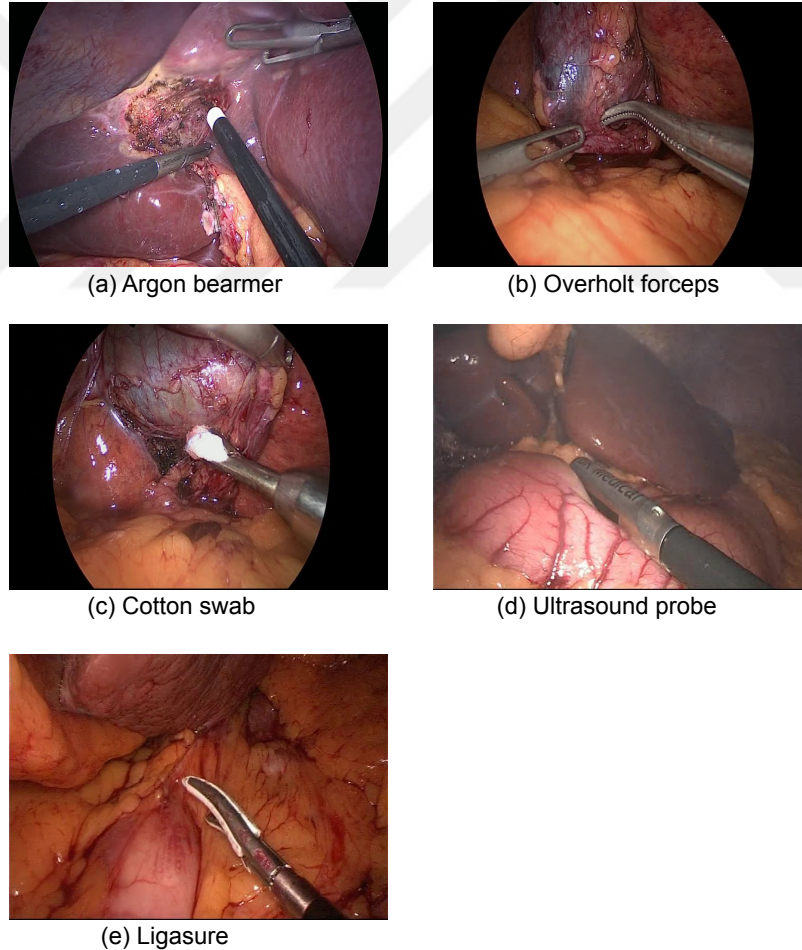


Figure 3.8 Other instruments from Insel dataset: (a) Argon beamer: used for coagulation, (b) Overholt forceps: tissue separation. (c) Cotton swab: for compress, (d) Ultrasound probe: to check for gallstones, (e) Ligasure: for tissue dissection.

challenge contained four sub-challenges: (i) surgical workflow recognition, (ii) surgical instrument detection, (iii) surgical action recognition, and (iv) surgical skill assessment.

The organizers presented 33 videos of Lap Chole surgery from 3 medical centers. The total duration of these videos was 22 hours. The average duration of the videos was 40 ± 18 minutes.

Seven phases were defined: (1) Preparation, (2) Calot triangle dissection, (3) Clipping and cutting, (4) Gallbladder dissection, (5) Gallbladder packaging, (6) Cleaning and coagulation, (7) Gallbladder retraction. Phase definitions were similar to the definitions of Cholec80. Phase durations and presence showed high variability across videos. As part of the challenge, 10 teams participated in the phase recognition task. The average f1-score of each participant ranged between 24% and 68%.

3.5 Other Datasets

Besides the datasets we mentioned in previous sections, there are several other datasets that were introduced in other publications. These Lap Chole datasets include but are not limited to TUM LapChole[73], Bar et al.[10], Ban et al. [33], and Golany et al. [35]. In this section, we will present a comparison of all datasets mentioned in this chapter based on their size and phase definitions. Table 3.7 illustrates this comparison. The comparison shows that phases are defined differently between the datasets. Some datasets include the trocar placement as a phase before the preparation. Preparation is a common phase across the datasets. Datasets are in agreement for Calot triangle dissection, Gallbladder dissection, and Gallbladder packaging phases. The datasets introduced by Bar et al.[10] and Ban et al.[33] show pretty different naming than other datasets. For example, Adhesiolysis happens before Dissection in Bar et al. This phase is probably accepted as part of the Calot triangle dissection in other datasets. Division and Separation correspond to Clipping, Cutting, and Gallbladder dissection in other datasets. Ban et al. introduced Checkpoint phases, i.e., Checkpoint 1 and 2, which define the inactive period before the clipping, and removal of the gallbladder, respectively. Moreover, they represented the preparation phase that is common in other datasets with two steps that have non-standard naming: "Fundus retraction", and "Release GB peritoneum". Gallbladder dissection which is common in other datasets is named differently, "Remove GB from liver bed" [33].

ID	TUM LapChole[73]	Cholec80	HeiChole	Bar et al.[10]	Ban et al. [33]	Golany et al. [35]	Insel
1	Trocar placement	Preparation	Preparation	Preparation	Port placement	Trocar insertion	Preparation
2	Preparation	Calot triangle dissection	Calot triangle dissection	Adhesiolysis	Fundus retraction	Preparation	Calot triangle dissection
3	Calot triangle dissection	Clipping and cutting	Clipping and cutting	Dissection	Release GB peritoneum	Calot triangle dissection	Gallbladder drainage
4	Clipping and cutting	Gallbladder dissection	Gallbladder dissection	Division	Calot triangle dissection	Clipping and cutting	Clipping
5	Gallbladder dissection	Gallbladder packaging	Gallbladder packaging	Separation	Checkpoint 1	Gallbladder dissection	Cutting
6	Gallbladder packaging	Cleaning and coagulation	Cleaning and coagulation	Packaging	Clip Cystic Artery	Gallbladder packaging	Gallbladder dissection
7	Cleaning and coagulation	Gallbladder retraction	Gallbladder retraction	Final inspection	Divide Cystic Artery	Cleaning and hemostasis	Gallbladder packaging
8	Gallbladder retraction				Clip Cystic Duct	Gallbladder extraction	Cleaning and coagulation
9					Divide Cystic Duct		Gallbladder retraction
10					Checkpoint 2		Liver biopsy
11					Remove GB from liver bed		Other procedure
12					Bagging		Intraoperative cholangiography
13					Other step		

Table 3.7 Comparative illustration of surgical phase recognition datasets, including the Insel dataset we introduced in this work. Phase names are defined differently across datasets.

3.6 Discussion And Conclusion

Datasets provided for surgical phase recognition are an integral part of machine learning model development. In this chapter, we presented a detailed description of our dataset, Insel, and the publicly available Cholec80 dataset. We also presented other private datasets and briefly compared their phase definitions. Our aim was to share visual samples and phase definitions from the datasets rather than compare all datasets in detail.

Datasets have less consensus on Lap Chole phase definitions. The reason for this should be clarified with clinicians. This difference can cause problems when transferring the model between datasets. It may be necessary to standardize the phase definitions to minimize the risk of problems.

Public datasets contain much less number of videos than private datasets. Bar et al. introduced a dataset having 1243 videos from five medical centers[10]. This has become the largest Lap Chole dataset to date. Our Insel dataset contains 384 videos, which makes it the second-largest Lap Chole dataset, to the best of our knowledge. Insel is followed by the Lap Chole dataset introduced by Golany et al.[35] (371 videos), and the one described by Ban et al.[33] (100 videos).

We will present a detailed comparison of the characteristics of our Insel and the Cholec80 datasets in the next chapter.

SYSTEMATIC COMPARISON OF DATASETS AND MODEL PERFORMANCE FOR SURGICAL PHASE RECOGNITION

4.1 Overview

Automated surgical phase recognition is a crucial task to bring context-aware decision support to the operating room (OR). The benefits of this automation include, but are not limited to, support for online decision-making, online OR staff scheduling, and automated report generation. The development of such automated solutions heavily relies on the datasets used for training. Therefore, dataset development is an important part of the model development lifecycle. In this chapter, we perform a systematic comparison of datasets and machine learning model performance for automated surgical phase recognition.

Datasets for surgical phase recognition are often developed for a specific type of surgery, laparoscopic cholecystectomy (abbreviated as Lap Chole) [9], due to the large availability and simplicity of this procedure. We observed two main problems for dataset development in existing literature, (i) datasets are usually not provided for public access, (ii) contributors do not have a strong agreement on phase definitions. (section 3.5)

The first issue brings a barrier to the reproducibility of the developed solutions, and the second issue introduces dataset bias for the developed models. Due to the limited availability of strong real-world datasets, practitioners often tend to train and benchmark their ML models on existing publicly available datasets, which are currently very limited. Therefore, the majority of the publications provide their results on the well-known public dataset, Cholec80[3]. There are a few works that presented results on their private large-scale datasets[10], however, their data is not shared. This impedes the reproducibility of the provided results. The second issue brings dataset bias, which means the model's architecture and hyperparameters are adjusted based on a particular dataset. However, these settings might not be optimal for a dataset that defines phases differently. Although Lap Chole has well-defined steps,

due to emergency cases or depending on the patient’s anatomy, surgeons may need to perform alternative steps. Some of the datasets keep their phase definitions more general, such as Cholec80, whereas some other private datasets keep it very specific [10, 33, 35]. To highlight the consequences of this difference on model development, we perform a data-centric study and compare two different datasets in a systematic way. To this end, we gathered a new private dataset of 384 Lap Chole videos for surgical phase recognition, namely the Insel. We included in our dataset all videos, including emergency surgeries and teaching cases, recorded in a continuous time frame of five years at Inselspital, Bern university hospital. We carefully analyzed the characteristics of our dataset and compared it with the most commonly used publicly available dataset, Cholec80. We observed strong differences between the two datasets. For instance, we observed the presence of surgical phases such as *gallbladder drainage*, *liver biopsy*, and *intraoperative cholangiography*, which did not appear in other datasets before (see Figure 4.1 for a qualitative example).

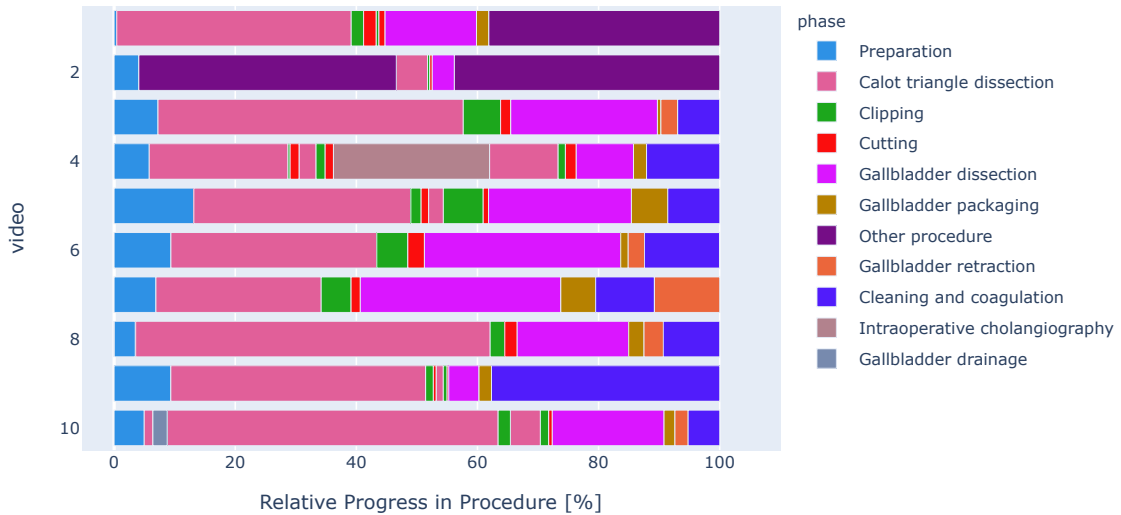


Figure 4.1 Lengths and order of surgical phases are highly imbalanced in Insel dataset. Color-coded ribbons illustrate phase progression for ten different procedures. Relative operation duration is shown (normalized to 100%) (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

ML models for surgical phase recognition are often developed using the publicly available dataset, Cholec80. However, models should be developed and compared on larger and more heterogeneous datasets, to obtain a more reliable performance that would better reflect the actual real-world performance. To achieve this, we assessed the performance of several state-of-the-art phase recognition models on our large-scale real-world dataset. We focused on two-stage neural network architectures since they reported higher performance in the latest benchmarks. We used transfer learning: we initialized the networks with weights from ImageNet, and fine-tuned models in both

stages with our own dataset. Finally, we performed an exhaustive model performance analysis on a selected model and pinpointed the challenge of severe class imbalance for a few phases which did not appear in other datasets before.

The results presented in this chapter have been previously published [54]^{1 2}.

4.2 Methods

4.2.1 Datasets

We used two datasets in this work. The first dataset is the Cholec80, as we described in section 3.2. The second dataset is our Insel dataset that was presented in our previous work [54] and described in section 3.3 in this work. We used our Insel dataset to train the surgical phase recognition models that we will describe in the next section 4.2.2.

For model training, we randomly partitioned our dataset with 70/10/20 percent training/validation/test ratios, respectively. Partitioning was performed on the video level, i.e., frames from a single video only appeared in one of the partitions. The training, validation, and test splits included 269, 38, and 77 videos, respectively.

4.2.2 Phase Recognition Models

We consider two different architectures in this work, MSTCN-based [26] and a recent transformer-based architecture, Trans-SVNet [31].

These architectures are composed of a backbone for extracting features from single frames, and a temporal model which combines features from consecutive frames to predict the surgical phase for each frame.

The feature extraction backbone is based on the Resnet-50 [15] architecture. The features extracted from the final Resnet block yield a vector of 2048 dimensions for each frame. Per-frame extracted features of a video are stacked to form a sequence, which is then fed into the temporal model.

As a temporal model, we consider (i) the online trained MSTCN model, which was

¹The author contributions to the original article are: Conceptualization, Kadir Kirtac and Florian Aspart; Data curation, Kadir Kirtac, Joël Lavanchy and Guido Beldi; Formal analysis, Kadir Kirtac; Investigation, Kadir Kirtac and Florian Aspart; Methodology, Kadir Kirtac and Florian Aspart; Project administration, Marco Smit and Michael Woods; Resources, Marco Smit and Michael Woods; Software, Kadir Kirtac; Supervision, Nizamettin Aydin, Marco Smit, Michael Woods and Florian Aspart; Validation, Kadir Kirtac; Visualization, Kadir Kirtac; Writing – original draft, Kadir Kirtac; Writing – review & editing, Kadir Kirtac, Joël Lavanchy, Guido Beldi and Florian Aspart

²The article is published under the Creative Commons Attribution License (CC BY 4.0).

named TeCNO [29], and (ii) the offline trained MSTCN model, which we named TeCNO-Off in this work.

MSTCN consists in cascading multiple stages of temporal convolutional networks (TCNs) [74]. This cascade prediction scheme allows every stage to refine the predictions of the previous stage. Every stage contains multiple layers of TCNs, which use dilated convolutions and skip connections. Dilated convolutions offer a larger temporal receptive field with fewer parameters. Hence, this results in faster training and inference, which helps process long surgical videos.

While the TeCNO online model performs causal convolutions in its TCN layers, whereas its offline version, TeCNO-Off, performs acausal convolutions. In other words, to predict the phase for each frame, the offline version considers all surrounding frames (preceding and succeeding), while the online model considers only preceding frames.

Additionally, we considered Trans-SVNet, which fuses the temporal features extracted from MSTCN and spatial features extracted from Resnet-50 in its transformer layers. This spatial and temporal information fusion yielded top results on Cholec80 dataset. We trained two versions of this model, one with cross-entropy loss and one with weighted cross-entropy loss (see details in Section 4.2.2.1).

4.2.2.1 Model Training

Videos were processed at 1 frame per second. The backbone Resnet-50 model was pre-trained on ImageNet [57] and fine-tuned on the Insel dataset. Video frames were downsampled to 224×224 pixels and data augmentation including shifting, scaling, and rotation was applied. For fine-tuning the backbone, the learning rate was 5×10^{-4} and the batch size was 64 frames. In order to tackle the class imbalance inherent to different phase lengths, we used weighted cross-entropy loss. The weights were set to the inverse sample count of each phase.

The number of stages was set to 3 both for the (i) TeCNO, and (ii) TeCNO-Off models, which have been trained on top of the extracted Resnet-50 features. Each MSTCN stage included 15 layers of TCNs and each layer outputs 64 feature maps. The models operate on the full temporal resolution of the videos. Both models were trained for a maximum of 50 epochs, with a learning rate of 5×10^{-4} . The training batch size was equal to the length of the video (one video per batch).

Next, we trained the Trans-SVNet model, (iii) with cross-entropy loss as described in [31], and (iv) with weighted cross-entropy loss. In the following, we reference the

latter as Trans-SVNet-WCE. Trans-SVNet-WCE used the same class weights computed for the backbone Resnet model.

Both Trans-SVNet and Trans-SVNet-WCE used features from the same Resnet and MSTCN backbones described above. Both models had two transformer layers, each having eight attention heads. Due to memory limitations, the causal temporal sequence length could not be set higher than 15.

Models were trained for a maximum of 30 epochs, with a learning rate of 1×10^{-3} . The batch size was equal to the length of the video (one video per training batch). Early stopping with a patience of 5 epochs on the validation loss was used for all the models described above.

4.2.3 Evaluation Methodology

Evaluation metrics are computational methods that explain how a particular method works. Evaluation methodology explains how a classification model is evaluated and what metrics are used for evaluation. This section describes our methodology to evaluate the surgical phase recognition models we described in the previous section. Surgical phase recognition is a multi-class time-series classification problem. The problem involves surgical phases as classes, i.e., the dependent variable to be predicted. In machine learning, multi-class classification problems are often evaluated with metrics such as accuracy, precision, recall, and f1-score.

4.2.3.1 Accuracy

In a multi-class time-series classification setting where we have an ordered sequence of predictions P with length T and an ordered sequence of ground truth labels G with the same length T , the accuracy is defined as the proportion of the number of correct predictions to the sequence length T . It can be formally defined as

$$Acc = 100 \times \frac{1}{T} \sum_{t=1}^T I(t), \quad I(t) = \begin{cases} 1, & P_t = G_t \\ 0, & \text{otherwise} \end{cases} \quad (4.1)$$

where P_t is the predicted label at time step t and G_t is the ground truth label at time step t . Note that the above definition is a definition of global accuracy, where individual classes are not taken into account. For the surgical phase recognition problem, the accuracy is first calculated over individual frames of each video (eq. 4.1). Then, per-video accuracies are averaged across videos, and the mean accuracy is reported together with the standard deviation of the mean (mean $\pm$ std). The mean

accuracy calculation can be formulated as shown in equation 4.2,

$$Mean Acc = \frac{1}{N} \sum_{v=1}^N Acc_v, \quad (4.2)$$

where $v \in \{1, 2, \dots, N\}$ represents the v^{th} video in the evaluation set, and Acc_v is the accuracy calculated for that video.

4.2.3.2 Precision

To understand the intuition behind the precision and the recall metrics, true positive (TP), true negative (TN), false positive (FP), and false negative (FN) terms are explained. For a particular class, TP represents the samples that are predicted correctly, and TN represents the correctly predicted samples for the remaining classes. In a similar intuition, FN represents the samples of the same class that are predicted incorrectly, and FP represents the samples from the remaining classes that are predicted incorrectly. These terms are easier to express in terms of a binary classification problem, where there are two classes. In multi-class classification problems, the one-against-all approach is often applied, where a class is considered the positive class at a time, and the remaining classes are considered the negative class. The positive class is shifted until every class is visited once.

Given the definitions of TP, TN, FN, and FP, precision can be defined as shown in equation 4.3,

$$Precision = \frac{\# TP}{\# TP + \# FP} \times 100. \quad (4.3)$$

Equation 4.3 considers predictions for the positive class only. Hence, it measures how well the model predicts the positive class. As an intuition, the cost of predicting a sick patient (positive class) as healthy (negative class) would have a higher risk than predicting a healthy person as sick. The person might die as a result of the first failure, however, an unnecessary medication would be taken in the second failure case, which is likely not to cause death. Therefore, we would like our classifier to not miss sick patients, which means we require a high-precision classifier. The opposite of this case would be to have a high-recall classifier, as explained below.

4.2.3.3 Recall

With a similar intuition, the recall metric measures the ability of a classifier to be able to retrieve all samples of the positive class. The recall is defined in formal terms in equation 4.4,

$$Recall = \frac{\# TP}{\# TP + \# FN} \times 100. \quad (4.4)$$

False negatives (FN) are the samples that are mistakenly predicted as negative. The count of FN samples is considered in the denominator (eq. 4.4). Therefore, this metric penalizes the miss of true positive samples.

4.2.3.4 F1-score

Given the definitions of the precision and the recall, the f1-score by definition is the harmonic mean of the precision and the recall, which combines both metrics to achieve a single score between 0 and 100. The f1-score is formulated by the equation 4.5,

$$F1 - score = 2 \times \left(\frac{precision \times recall}{precision + recall} \right) \times 100. \quad (4.5)$$

4.2.3.5 Mean Precision

In our case, the precision, recall, and f1-score are first computed for each phase $p \in \{P_1, P_2, \dots, P_{N_p}\}$ of a video $v \in \{V_1, V_2, \dots, V_{N_v}\}$, as formulated by equations 4.3, 4.4, and 4.5. Then, this is repeated for every video in the evaluation set. Scores computed for each phase are averaged across videos. Finally, these phase-wise scores are averaged again to obtain a single score. The below equation (eq.4.6), formalizes the aggregation of the precision to obtain the mean precision,

$$Mean\ Precision = \frac{1}{N_p} \sum_{j=1}^{N_p} \left[\frac{1}{N_v} \sum_{k=1}^{N_v} Precision_{jk} \right], \quad (4.6)$$

where $j \in \{1, 2, \dots, N_p\}$ represents the j^{th} phase in N_p phase classes, and $k \in \{1, 2, \dots, N_v\}$ represents the k^{th} video of the evaluation set which has the cardinality of N_v . Then, $Precision_{jk}$ shows the precision score calculated for the j^{th} phase belonging to the k^{th} video in the evaluation set. Similarly, the same equation applies to the recall and f1-score metrics, but we do not repeat them here. The mean scores for precision, recall, and f1-score are often reported together with the standard deviation (std) of the mean across the phases. Here, we also report the mean and std (mean $\pm$ std).

A concrete example showing how accuracy, precision, recall, and f1-score are calculated for our problem is illustrated in Figure 4.2. The example video contains three phases with different lengths, each represented with a different color: P1 (blue) with length 9, P2 (yellow) with length 6, and P3 (pink) with length 10. The example shows three different predictions for this video, i.e., (a), (b), and (c). We can assume each predicted sequence corresponds to a different sample. We can calculate the accuracy, precision, recall, and f1-score as we described earlier.

Calculation of accuracy is trivial: we divide the count of correct predictions by the

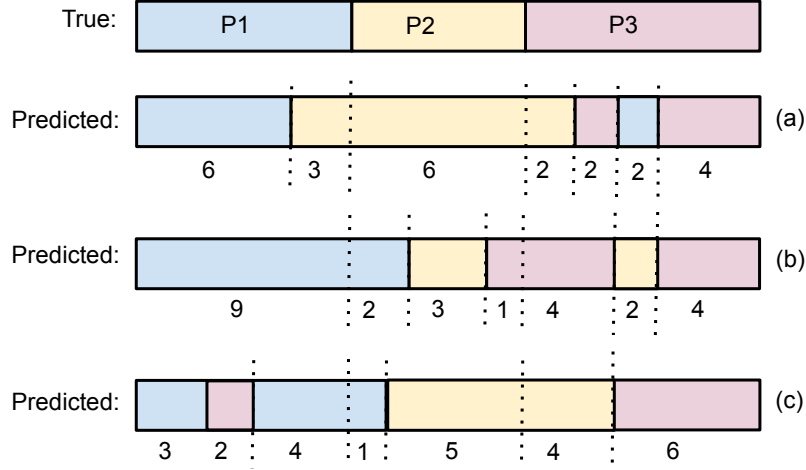


Figure 4.2 The calculation of metrics is illustrated as a concrete example. True labels and predictions are represented as horizontal bars. Each colored bar represents a different phase label, and the number shown under each box (phase) represents the duration of that phase. Best viewed in color.

count of all predictions for each video: (a) 18/25, (b) 20/25, (c) 18/25. Then, the mean accuracy is calculated as the average of these three video-level accuracies, resulting in 0.746. If we want to express the accuracy in $[0, 100]$ range, we can multiply the result by 100 and obtain 74.6%.

To calculate the mean precision, we need to calculate the precision for each phase in each video. For the predicted sequence in (a) class P1 has a TP of 6 and FP of 2 (the second blue bar in (a)), thus, $Precision_{a1} = 6/8$. Similarly, class P2 has a TP of 6 and an FP of 5, thus $Precision_{a2} = 6/11$. Class P3 has a TP of 6 and an FP of 0, resulting in $Precision_{a3} = 6/6$.

We can repeat the calculation similarly for the second (b) and third (c) predicted sequences. (b) $Precision_{b1} = 9/11$, $Precision_{b2} = 3/5$, $Precision_{b3} = 8/9$.

(c) $Precision_{c1} = 7/8$, $Precision_{c2} = 5/9$, $Precision_{c3} = 6/8$.

We can tabularize these values to make the interpretation easier (Table 4.1). First, for each phase, the mean across videos is computed, yielding 0.81 for phase P1, 0.56 for phase P2, and 0.88 for phase P3. Next, another mean is calculated over these three scores, yielding the final score of 0.75. Again, this score can be scaled to $[0, 100]$ range to obtain 75% percent. Similarly, we can compute the recall and show the values in tabular form for easier interpretation (Table 4.2).

According to the calculations described above, we report the performance in terms of individual frames and compare the above-mentioned metrics on the test split of our Insel dataset.

Table 4.1 Illustration of the mean precision calculation on a toy data: (1) the first mean is computed over the videos for each phase (from top to bottom in the vertical direction), (2) the final mean is computed over the phases (from left to right in the horizontal direction), resulting in 75% mean precision.

video	phase			mean(across phases)
	P1	P2	P3	
(a)	6/8	6/11	6/6	-
(b)	9/11	3/5	8/9	-
(c)	7/8	5/9	6/8	-
mean(across videos)	0.81	0.56	0.88	-
-	-	-	-	0.75

Table 4.2 Illustration of the mean recall calculation on a toy data: (1) the first mean is computed over the videos for each phase (from top to bottom in the vertical direction), (2) the final mean is computed over the phases (from left to right in the horizontal direction), resulting in 73% mean recall.

video	phase			mean(across phases)
	P1	P2	P3	
(a)	6/9	6/6	6/10	-
(b)	9/9	3/6	8/10	-
(c)	7/9	5/6	6/10	-
mean(across videos)	0.77	0.77	0.66	-
-	-	-	-	0.73

4.2.3.6 Segmental metrics

Although computing the performance metrics based on the frame-level classification is widely used in the research community, these metrics might sometimes come short of explaining the actual performance of the model. First of all, if the classes are imbalanced for example, which is often the case for surgical phases, the accuracy metric is less useful. In this case, the result mostly reflects the performance of the dominant classes, and the performance of under-represented classes is hidden. Second, predicting the correct ordering of phases is crucial for surgical phase recognition. However, the accuracy metric does not take into account the ordering of the phase predictions. It may still respond high even if the order of phases is wrongly predicted. Two types of errors are related to the ordering of predicted sequences, and two segment-based metrics were proposed in [51] to tackle these errors. These errors are illustrated in Figure 4.3. The metrics will be explained at the end of the next paragraph.

Over-segmentation errors can occur in surgical phase recognition. Classification models often make errors, and these errors often occur as blocks of segments of another phase overlapping the correct phase (Figure 4.3(a)). This type of error can be problematic in online surgical phase recognition. When surgical phases switch

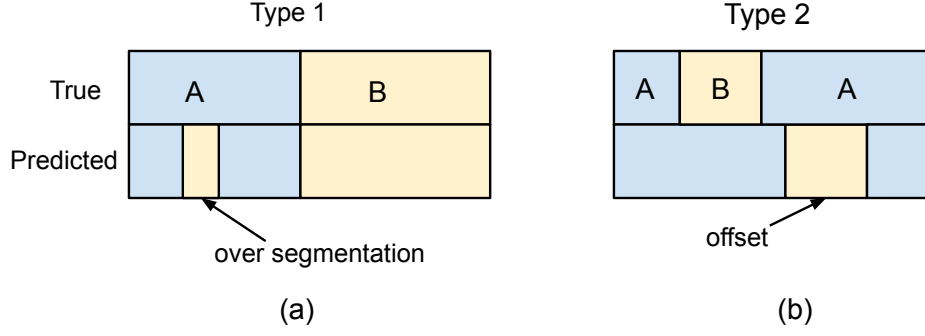


Figure 4.3 Segment-based metrics compute two error types. (a) The first one, over-segmentation, computes the error if multiple predicted segments are contained in one true segment. (b) The second one measures the ordering of segments and allows for small temporal offsets. The offset might be due to inter-rater differences and they should not affect the performance. Best viewed in color. (Figure and caption adapted from Lea et al. [51], CC BY 4.0.)

too often due to this over-segmentation error, it might cause a mental distraction to the surgeon. Therefore, it is important to evaluate the model performance in terms of such errors. The second issue, segment offsetting, can happen due to annotator variability (Figure 4.3(b)). This can happen when the start of a surgical phase is interpreted slightly differently between annotators. However, the slight shift between the predicted sequence and its ground truth at the starting points might not be important for the surgeon. Thus, a performance metric that can tolerate these small differences could be helpful for evaluation. To remedy these issues, segment-based metrics have been proposed [51]. Segment-based computation considers a predicted segment as a sample instead of a frame. In this work, we incorporate the use of segmental f1-score [74] and segmental edit score [51], which have been used to assess video action detection and segmentation algorithms [74].

The edit score penalizes incorrect order between predictions and ground truth while tolerating small temporal offsets between the predicted segment and the ground truth. For each ground-truth segment G , the segment-wise labels G_s are defined such that if $G = \{[KKK], [LL], [MMM]\}$ then $G_s = \{KLM\}$, where K, L, M are different labels. The predicted frame-wise (P) and segment-wise (P_s) sequences are defined analogously. The edit score is defined using the edit distance [75], $s_e(G_s, P_s)$, with insertions, deletions, and replacements. The final score is obtained by normalizing the edit distance according to the maximum between the length of G_s , L_G , and of P_s , L_P . It can be formally defined as shown in equation 4.7,

$$S_{edit} = \left(1 - \frac{s_e(G_s, P_s)}{\max(L_G, L_P)}\right) \times 100, \quad (4.7)$$

with 100 being the best and 0 being the worst score.

The segmental f1-score evaluates the segmental overlap between ground truth and predicted sequences. In this evaluation, over-segmentation errors are penalized. True and false positives are computed over each predicted segment, by comparing its intersection over union (IoU) with respect to the corresponding ground truth using a fixed threshold, k . Precision and recall are computed for each class and then summed over classes. The segmental f1-score computed for a particular class can be formally defined as shown in equation 4.8,

$$F1@k = 2 \times \left(\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \right) \times 100, \quad (4.8)$$

where $k \in \{25, 50, 75\}$ is the IoU threshold.

4.3 Results

In the following, we first present the results of our dataset labeling. We perform a comparative analysis of our Insel dataset with the most commonly used public dataset for surgical phase recognition: Cholec80 [3]. Finally, we thoroughly analyze the performance of our phase recognition pipeline with the Insel dataset.

4.3.1 Comparative Analysis of Cholec80 and Insel Dataset

Surgeries in the Insel dataset were much longer than surgeries in Cholec80. Insel's surgeries had a median duration of 54 minutes (min: 13, max: 374) against 34 minutes (min:12, max: 99) for the Cholec80 videos. Figure 4.4 highlights the distribution of the durations of video lengths.

Attentively inspecting each video in the Insel dataset, we found some surgical phases which were not present in our guidelines and in the public Cholec80 dataset. In collaboration with a board-certified surgeon, we identified these phases as *gallbladder drainage*, *liver biopsy*, and *intraoperative cholangiography*, as described in the previous chapter 3.6.

Qualitative examples of video frames associated with these surgical phases are displayed in Figure 4.6a–c. These three phases appeared in relatively fewer videos compared to other phases: *gallbladder drainage* in 26 videos, *liver biopsy* in 18 videos, and *intraoperative cholangiography* in 47 of all 384 videos (Fig. 4.5). Moreover, we observed less visual variability in the surgical instruments used across interventions associated with the *gallbladder drainage*, *liver biopsy*, and *intraoperative cholangiography* phases in Insel dataset.

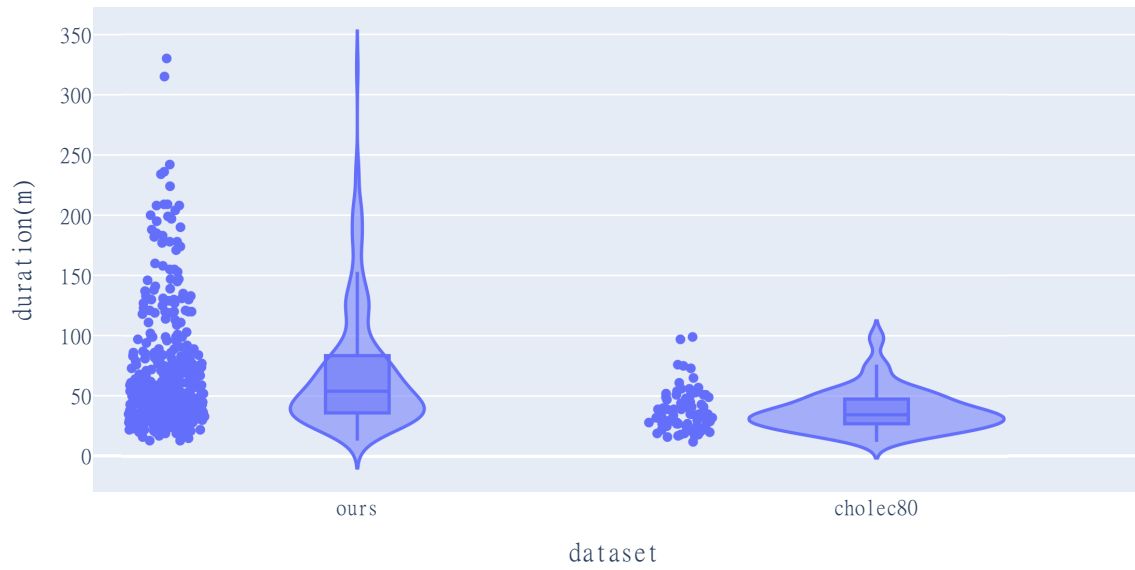


Figure 4.4 Distribution of video durations (in minutes) in Insel dataset and Cholec80. Insel dataset shows more variability in durations and has longer videos than Cholec80. (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

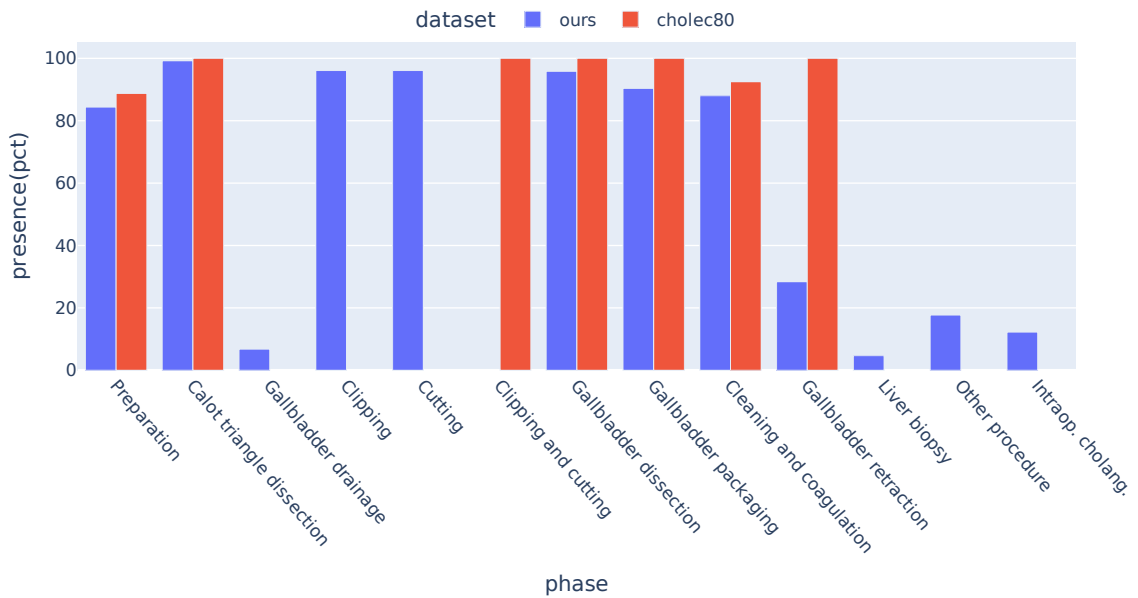


Figure 4.5 Distribution of phase presence as a percentage of the total number of videos. Some phases are extremely underrepresented across videos. For example, Gallbladder drainage and Liver biopsy are present only in 0.6% and 0.4% of videos in our Insel dataset, respectively (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

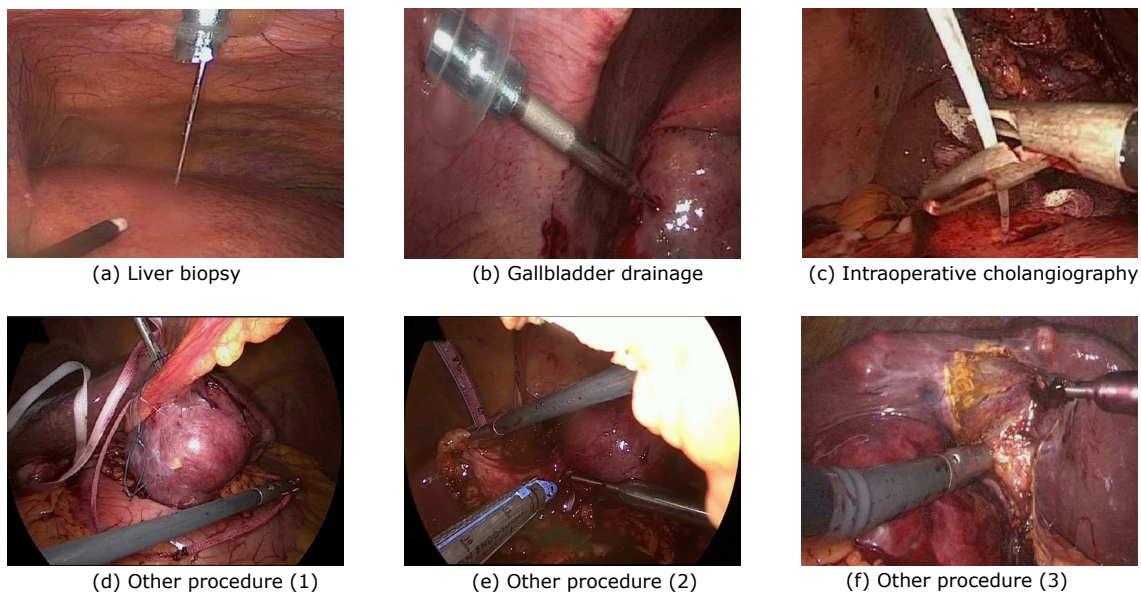


Figure 4.6 Example of surgical scenes for surgical phases (from Insel dataset) which are absent in Cholec80: (a) liver biopsy; (b) gallbladder drainage; (c) intraoperative cholangiography; (d) application of a cotton tape for laparoscopic Pringle maneuver; (e) application of the endoscopic linear stapler; (f) intraoperative ultrasonography.

These latter three examples include procedures that are not typical for cholecystectomy (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

In addition to these phases, we observed the presence of non-cholecystectomy-related surgical procedures. These included procedures such as intraoperative ultrasonography and liver or pancreas resection (see Figure 4.6d–f for qualitative examples). To avoid mixing these miscellaneous procedures with cholecystectomy-specific phases, we explicitly labeled these procedures as *other procedure*.

We observed these non-cholecystectomy-related procedures in 68 of the 384 videos we gathered. With a median duration of 82 minutes (min: 1.2, max: 263), this phase was one of the longest in Insel dataset (Figure 4.7). Due to the variability of the tasks involved, the surgical instruments used during this *other procedure* phase varied across the interventions.

Videos in the Cholec80 dataset did not include any of the *gallbladder drainage*, *liver biopsy*, *other procedure*, and *intraoperative cholangiography* phases. Consequently, Cholec80 did not include any definition for these phases.

Carefully looking at the clipping and cutting scenes, we observed that the surgical flow between these phases was not necessarily linear. We found many cases where *clipping* and *cutting* were performed multiple times where the surgeons came back and forth between the two. To enable our annotation to capture this non-linear workflow, we

chose to define *clipping* and *cutting* as two separate phases. Cholec80 had a unified *clipping and cutting* phase instead.

Surgical phase duration was strongly imbalanced in both datasets (Figure 4.7). Besides non-cholecystectomy procedures (*other procedure*), *calot triangle dissection* was the longest phase in both datasets. In accordance with the full-video length, Cholec80 had shorter phase durations across videos and shorter video durations compared to the Insel dataset (see Fig. 4.4).

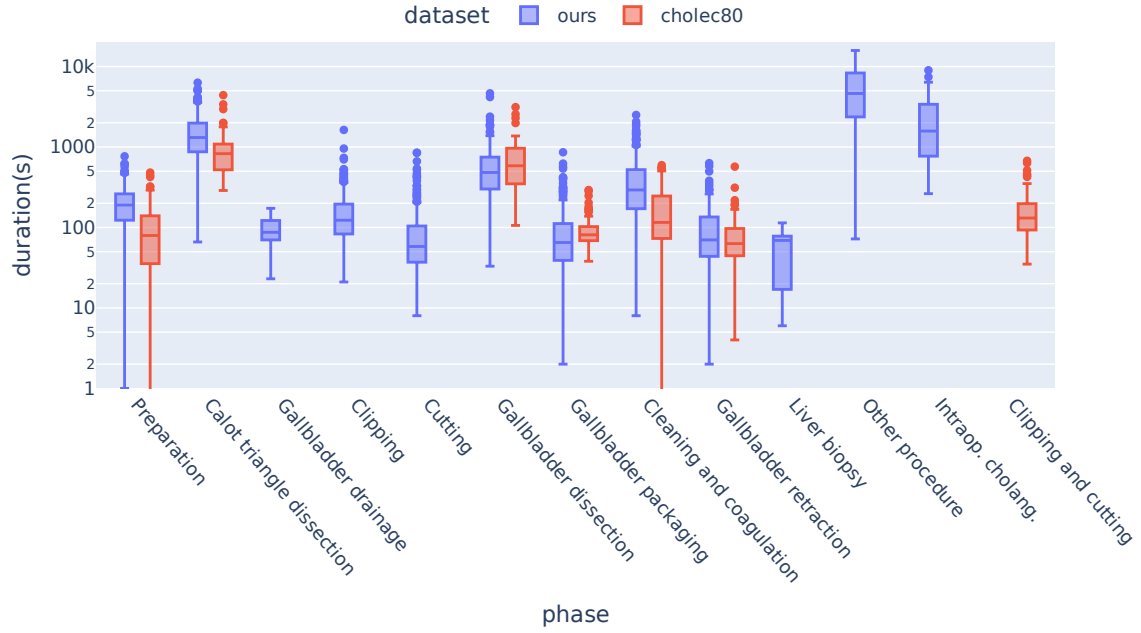


Figure 4.7 Distribution of phase duration across videos (in seconds) in Insel dataset and Cholec80. Duration (y-axis) is in log scale. Some phases present in Insel videos are missing in Cholec80. Phases missing in a video are ignored (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

Finally, we compared the phase transitions in both datasets (Tables 4.3 and 4.4). This table shows in what proportion (percent) a transition from a phase in the row axis to another phase in the column axis happens.

The surgical workflow in the Cholec80 dataset was mostly linear. Each phase had a clear following phase: four out of the six surgical phases had the same following phase in over 99% of the cases. On the contrary, the transitions of surgical phases in Insel dataset were less deterministic. This was particularly the case for *other procedure* and *intraoperative cholangiography* which did not have a clear succeeding phase. Transitions to the *other procedure* phase could take place from diverse phases. In other words, the *other procedure* had no clear position in the Lap Chole workflow.

Table 4.3 Phase transitions in Insel dataset. Compared with Cholec80, transitions in Insel dataset possess more complexity. The numbers indicate the relative (in percent) transitions from one phase (row-axis) to another (column-axis) (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.)

Phase	Next Phase (%)											
	<i>Preparation</i>	<i>Calot triangle dissection</i>	<i>Gallbladder drainage</i>	<i>Clipping</i>	<i>Cutting</i>	<i>Gallbladder dissection</i>	<i>Gallbladder packaging</i>	<i>Cleaning&coagulation</i>	<i>Gallbladder retraction</i>	<i>Liver biopsy</i>	<i>Other procedure</i>	<i>Intraop. cholangiography</i>
Preparation	-	94.44	0	0	0	0	0	0	0	0	0	5.56
Calot triangle diss.	0	-	4.42	85.84	1.77	1.77	0	0.17	0	0.53	1.78	3.71
Gallbladder drainage	0	96.15	-	0	0	0	3.85	0	0	0	0	0
Clipping	0	8.38	0	-	86.93	2.9	0	0.16	0	0	0	1.61
Cutting	0	15.85	0	14.59	-	62.52	1.26	1.08	0	0	0.9	3.78
Gallbladder diss.	0	0.52	0	4.17	0.26	-	67.36	22.19	0.52	0	4.17	0.78
Gallbladder packaging	0	0	0.31	0	0	0	-	69.4	27.44	1.26	1.57	0
Cleaning&coagulation	0.59	3.55	0	1.18	0	1.18	47.93	-	13.61	10.65	21.3	0
Gallbladder retraction	0	0	0	0	0	1.26	0	96.20	-	0	2.53	0
Liver biopsy	0	0	0	0	0	0	0	96.66	0	-	3.33	0
Other procedure	0	52.72	0	5.45	0	9.09	3.63	20.0	0	7.27	-	1.81
Intraop. cholangiography	0	21.42	0	58.93	8.93	5.35	0	0	0	0	5.35	-

Interestingly, procedures in Insel dataset mostly transitioned to *cleaning and coagulation* after *gallbladder retraction*, meaning that surgeons preferred to do the cleaning after the specimen is pulled out of the body (Table 4.3).

This was different in Cholec80, in which procedures regularly end with the *gallbladder retraction* phase (Table 4.4).

Table 4.4 Phase transitions in Cholec80. The numbers indicate the relative transitions (in percent) from one phase (row-axis) to another (column-axis) (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.)

Phase	Next Phase (%)						
	Preparation	Calot triangle dissection	Clipping and cutting	Gallbladder dissection	Gallbladder packaging	Cleaning and coagulation	Gallbladder retraction
Preparation	-	100	0	0	0	0	0
Calot triangle dissection	0	-	100	0	0	0	0
Clipping and cutting	0	0	-	100	0	0	0
Gallbladder dissection	0	0	0	-	85	15	0
Gallbladder packaging	0	0	0	0	-	73.75	26.25
Cleaning and coagulation	0	0	0	0	17	-	83
Gallbladder retraction	0	0	0	0	0	0	-

4.3.2 Surgical Phase Recognition on Insel Dataset

We trained and tested two recent deep network architectures which achieved top performance on Cholec80 dataset. Specifically, we considered two different architectures: MSTCN-based (TeCNO) and transformer-based (Trans-SVNet).

In accordance with existing studies on Cholec80, stacking temporal models (MSTCN or Trans-SVNet) on top of Resnet-50 backbone boosted the results on all aggregated metrics (Table 4.5). In particular, the segment-based metric, which takes into account the temporal fragmentation of the predictions, was almost 50 times greater (*F1@50*: from 0.83 to 40.68 in the second row).

Table 4.5 Model performance (mean $\pm$ std) using aggregated metrics on Insel dataset. The performance of the Resnet50 backbone is shown in the first row. Stacking the MSTCN on top of Resnet50 boosted the phase segmentation performance (the last two columns in the second row). (Table and caption adapted from Kirtac et al. [54], CC BY 4.0.)

Method	Accuracy	Precision	Recall	F1-Score	F1@50	Edit
Resnet-50	68.94 $\pm$ 13.89	61.54 $\pm$ 18.13	57.46 $\pm$ 18.18	52.89 $\pm$ 16.54	0.83 $\pm$ 1.17	1.79 $\pm$ 1.31
TeCNO-Off	84.75 $\pm$ 13.49	67.89 $\pm$ 23.23	69.27 $\pm$ 24.09	64.02 $\pm$ 22.53	40.68 $\pm$ 23.75	39.01 $\pm$ 25.3
TeCNO [29]	82.76 $\pm$ 14.20	69.54 $\pm$ 18.45	70.82 $\pm$ 20.0	64.65 $\pm$ 19.97	21.45 $\pm$ 18.28	18.31 $\pm$ 16.18
Trans-SVNet [31]	84.05 $\pm$ 14.41	69.02 $\pm$ 21.66	63.73 $\pm$ 30.05	61.70 $\pm$ 26.88	30.04 $\pm$ 20.13	26.18 $\pm$ 16.75
Trans-SVNet-WCE	82.97 $\pm$ 14.48	68.33 $\pm$ 22.08	70.67 $\pm$ 19.72	64.10 $\pm$ 21.95	26.46 $\pm$ 18.76	22.41 $\pm$ 15.66

The offline MSTCN-based model (TeCNO-Off) reached a top average accuracy of 84.75%, as well as top segment-based scores ($F1@50$ and edit score). Yet, the best performance in terms of frame-wise precision, recall, and f1-score was achieved by the online TeCNO model.

Looking at the phase-wise performance of the models (Figure 4.9), we observed the largest variability across models on the minority phases (e.g., *gallbladder drainage*, *liver biopsy*). The performance gap between the models was the greatest in the extremely under-represented phase *liver biopsy*.

Interestingly, the online TeCNO model performed better than its offline version (TeCNO-Off) in the minority phases. This was not reflected in the aggregated accuracy metric (Table 4.5). Similarly, the use of weighted cross-entropy loss to train the Trans-SVNet model was crucial to increase the model recall on the minority phases (e.g., *gallbladder drainage*, *liver biopsy*). The Trans-SVNet model without weighted cross-entropy loss had a higher aggregated metric.

In the following, we performed a detailed performance analysis of the online TeCNO model. We focused on this model given its higher performance on the minority phases, the characteristic of Insel dataset.

The high standard deviation for precision, recall, and f1-score (Table 4.5) hinted at a strong variation in the model performance across phases, as observed during the model comparison (Figure 4.9). This variation was better reflected in the phase-wise precision/recall plots (Figure 4.10a).

In particular, the model struggled to accurately predict the *gallbladder drainage*, *gallbladder retraction*, *liver biopsy*, and *intraoperative cholangiography*. These four phases were less prevalent in Insel dataset (see Fig. 4.5 and Fig. 4.8). Note that there were only five and four test videos for *gallbladder drainage* and *liver biopsy*, respectively. The remaining phases had much higher precision and recall. Notably, *preparation*, *Calot triangle dissection*, *gallbladder dissection*, *cleaning and coagulation* and *other procedure* phases had above 76 percent recall. The confusion matrix emphasized the phase-specific errors of the model (Figure 4.10b). Importantly, confusion with temporally closer phases was more prevalent than with temporally far phases. For example, *clipping* was mostly misclassified as *cutting* and *cutting* as *gallbladder dissection*; in both cases, these are their natural successive phase (Table 4.3). Similarly, *gallbladder packaging* was mostly misclassified as its most likely successor: *cleaning and coagulation*.

Phase-wise performance varied strongly across videos (Figure 4.11). While the

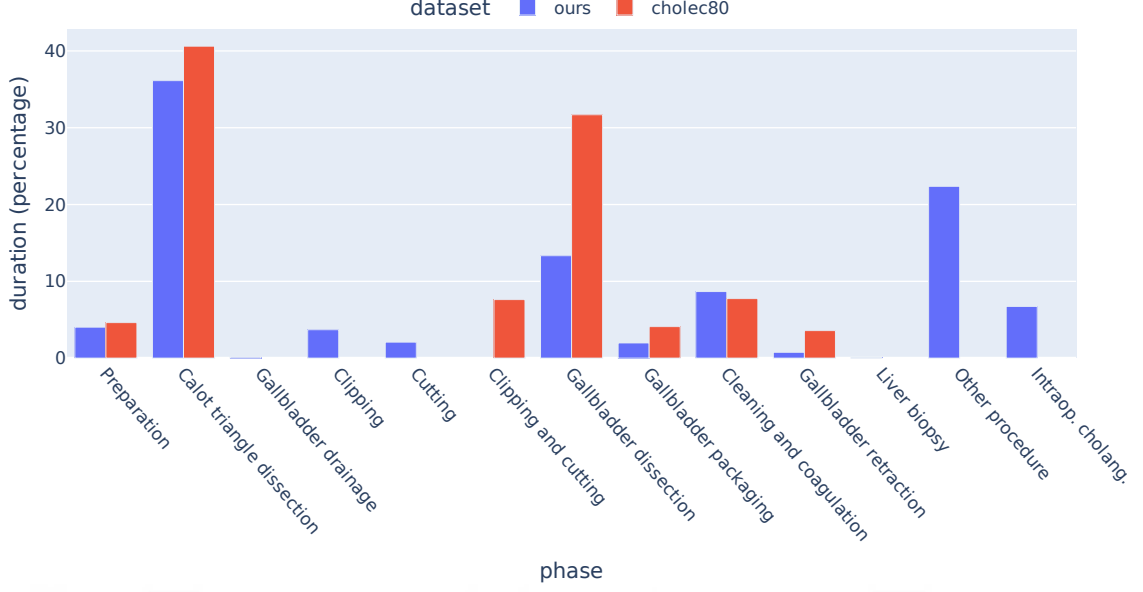


Figure 4.8 Distribution of phase duration as a percentage of total duration in Insel dataset and Cholec80. The imbalance between phases for both datasets is severe. Gallbladder drainage and Liver biopsy phases are extremely underrepresented in Insel dataset, with 0.15% and 0.06% duration, respectively (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

precision/recall averaged across all videos was rather high for most of the phases, some videos displayed poor performance for particular phases. The *gallbladder drainage*, *gallbladder retraction*, *liver biopsy*, and *intraoperative cholangiography* phases had the highest variability.

The video-level performance showed high variation (Figure 4.12).

Beyond frame-wise metrics, we assessed the model performance using segment-based metrics (Figure 4.13).

The segmental f1-score averaged over phases, $F1@k$, depended on the overlapping IoU threshold k . We measured $F1@25 = 25.48 \pm 18.82$, $F1@50 = 21.45 \pm 18.28$, $F1@75 = 15.42 \pm 13.96$ and the edit score $S_{\text{edit}} = 18.31 \pm 16.18$. Similar to frame-wise metrics, the high standard deviation underlined a strong disparity in the model performance across videos.

The lower segment-based scores compared to the frame-wise f1-score emphasized an over-segmentation of the predictions. Frame-wise f1-score, segment-based edit score, and F1-score were correlated (Figure 4.13a). Nevertheless, segment-based scores covered a larger range than the frame-wise f1-scores. As an example, *video12* in Figure 4.13a had high frame-wise f1-score (84.69%), but low $F1@50$ (7.36%) and edit score (6.74%). Looking closer at its phase labels, this video presented two long

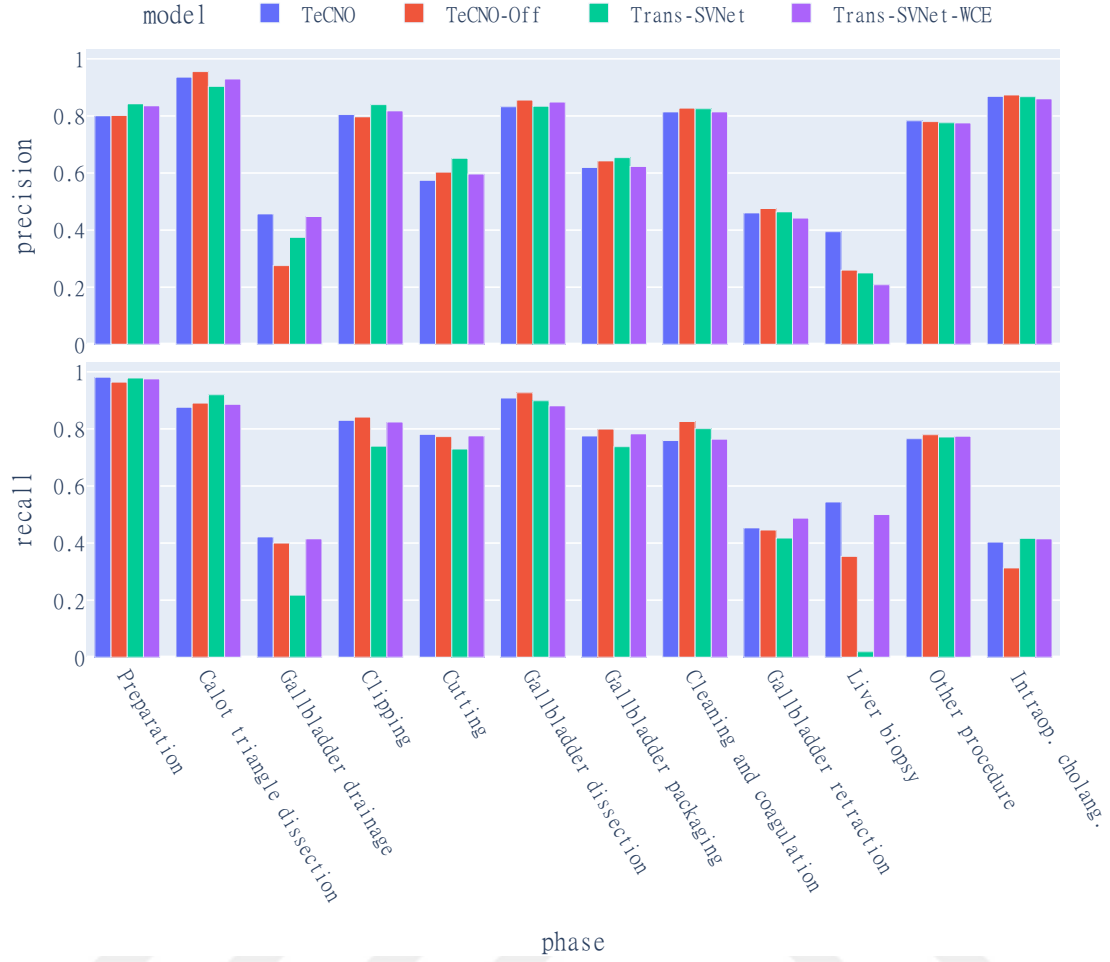


Figure 4.9 Comparative phase-specific performance analysis of the models in InseI dataset. For the minority phases, *gallbladder drainage*, and *liver biopsy*, the TeCNO model showed the best performance among others. The performance gap was highly significant for *liver biopsy*, which is an extremely under-represented phase in InseI dataset, between TeCNO and Trans-SVNet (28 times and 1.57 times greater for recall and precision, respectively) (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

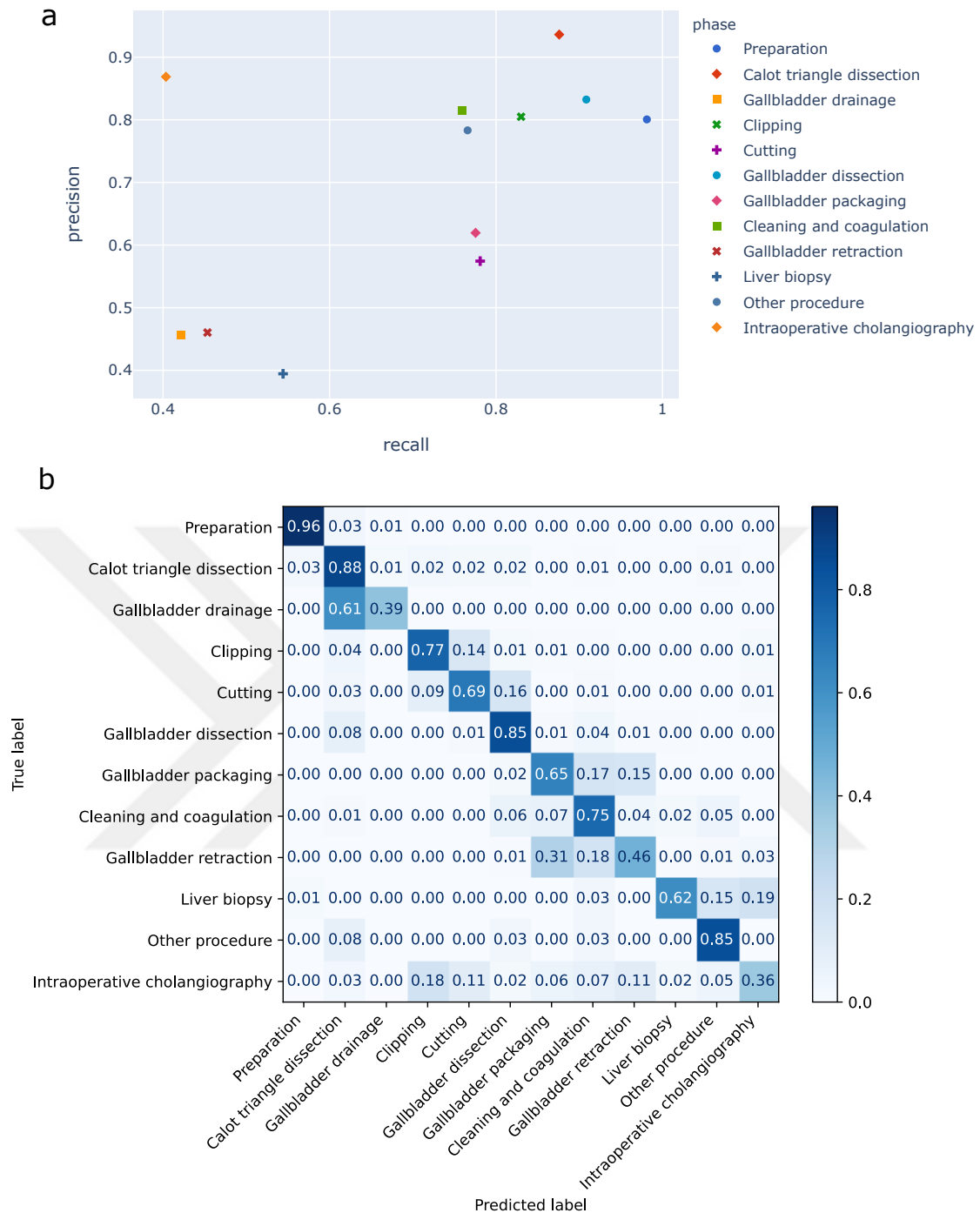


Figure 4.10 Phase-specific performance analysis of the TeCNO model on Insel dataset. **(a)** Illustration of phase level precision and recall averaged across videos.

One can notice the strong variation in performance between phases. **(b)** The confusion matrix illustrates the confusion between phases. Every sample (frame-wise prediction and ground truth) contributed equally during the computation. Confusion between temporally closer phases was more prevalent than in temporally distant phases (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

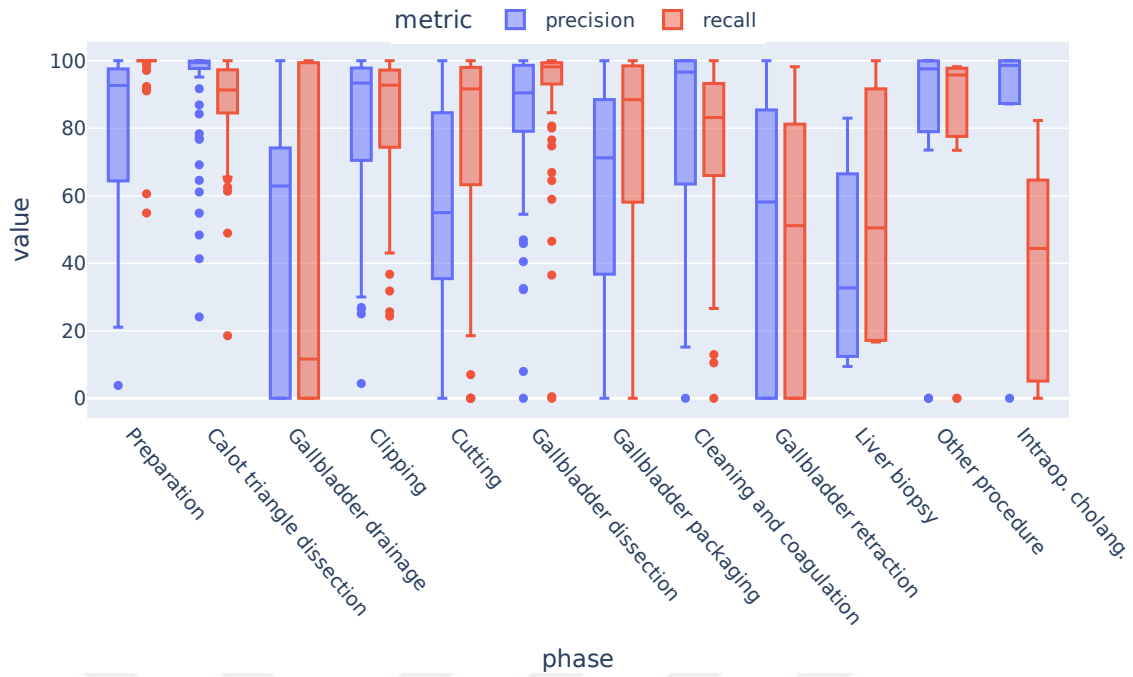


Figure 4.11 Phase-specific performance of the TeCNO model showed high variation across videos on Insel dataset (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

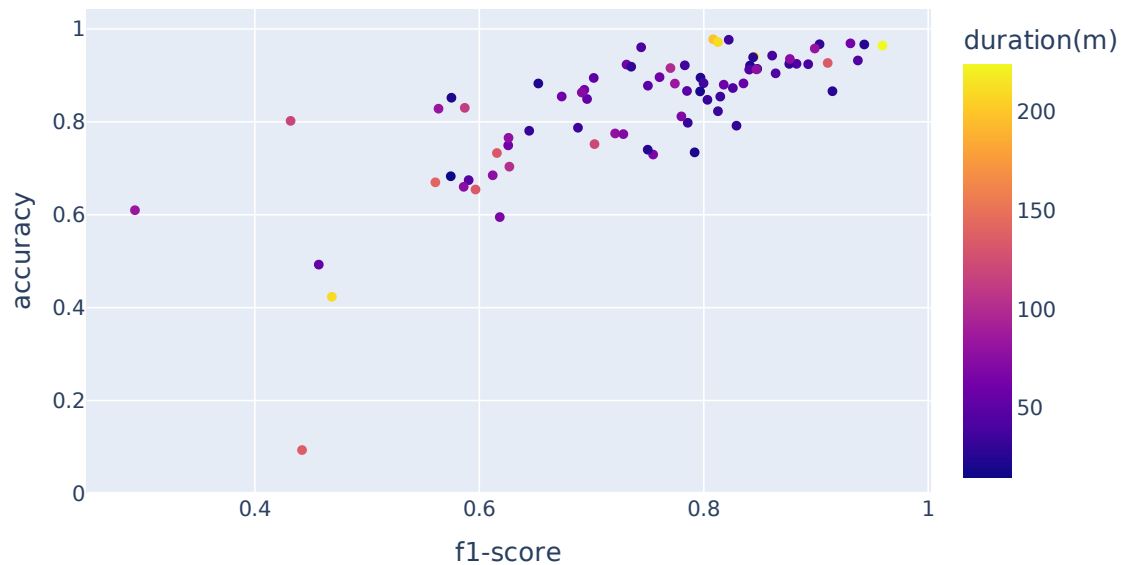


Figure 4.12 Performance based on individual videos of the TeCNO model showed high variation on Insel dataset (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

phases, e.g., *Calot triangle dissection* and *gallbladder dissection*. These two phases were recognized with low frame-wise error rates which increased the model accuracy. Yet, predictions had lots of over-segmentation errors (fluctuating thin bars); this decreased the performance in terms of segment-based metrics.

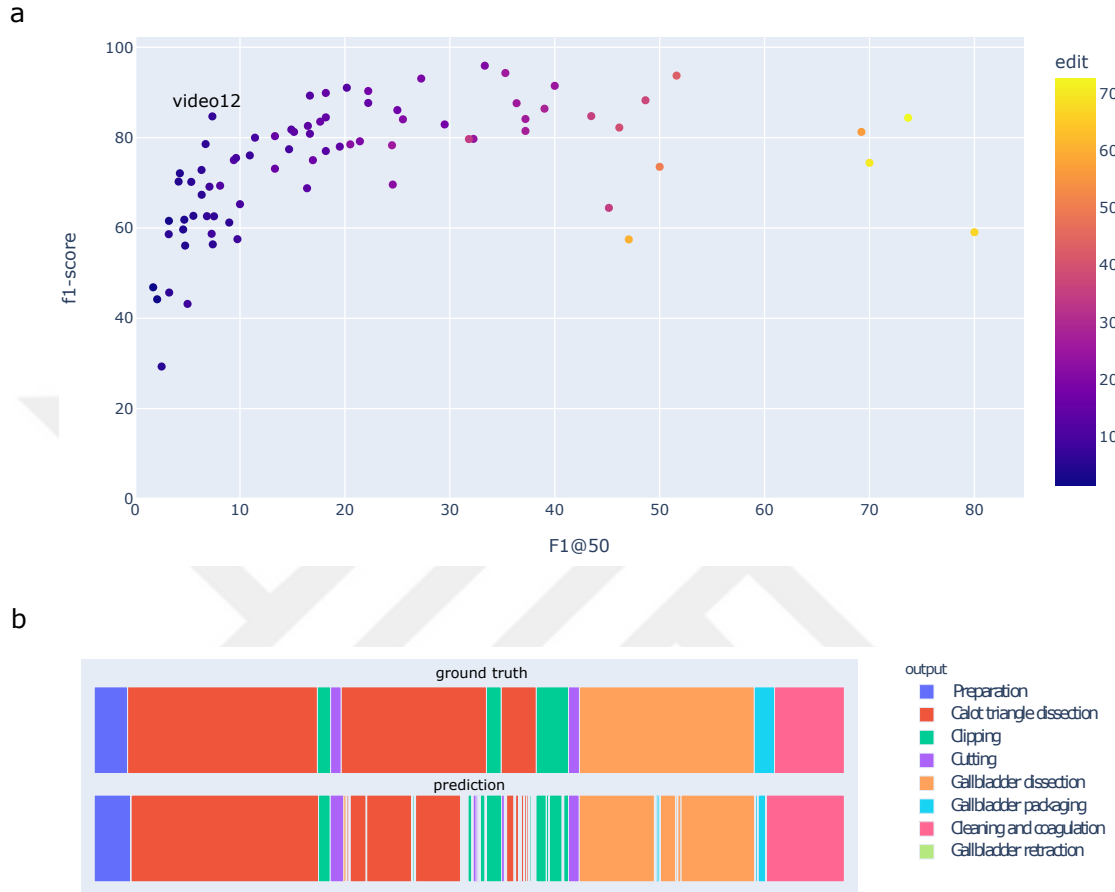


Figure 4.13 Illustration of segmental performance metrics of the online TeCNO model on Insel dataset, on a video basis. (a) Frame-wise f1-score compared to segmental f1-score and edit score. The edit score is encoded by the color bar. F1@50 displays the individual performance in a wider range compared to the frame-wise f1-score. (b) Timeline plot of the ground truth (top) and predictions (bottom) for *video12*, illustrating the over-segmentation errors. Each phase is encoded by a different color explained in the legend (Figure and caption adapted from Kirtac et al. [54], CC BY 4.0.)

4.4 Discussion

In this work we thoroughly (i) analyzed the distribution of laparoscopic cholecystectomy video data recorded in a given medical center over time, and (ii) retrained and tested existing deep neural network architectures which achieved top scores on the Cholec80 dataset on such video data.

First, we gathered a large-scale dataset of 384 laparoscopic cholecystectomy videos.

We compared and contrasted the characteristics of Insel dataset with the most commonly used publicly available dataset for surgical AI, Cholec80. Unlike curated public datasets, Insel dataset contained all videos, including emergency surgeries and teaching cases, recorded in a continuous time frame of five years. Our comparison highlighted key differences. Our Insel dataset was larger (384 vs 80 videos), the videos were longer and included additional phases. In particular, Insel dataset contained extremely underrepresented phases such as *gallbladder drainage* and *liver biopsy*, which were missing in public and other private datasets. Also, *intraoperative cholangiography*, a phase that rarely appears in Insel videos, but is relatively longer when it happens, posed an extra challenge for the model. These three phases are often applied in emergency cases or training scenarios, which were ignored in public datasets. Because public datasets often focus on method benchmarking, so they are purposefully kept as general as possible, by mostly ignoring alternative workflow. To work with these rare phases, one possibility for us was to ignore and leave them as part of the current phase. However, this would have two drawbacks. First, different instruments perform during these rare phases, so leaving them unlabeled can complicate the learning process and result in less optimal features for the current phase. Second, it may be inevitable that these phases are not defined for a particular use case, meaning that the cost of ignoring them would be very high. In such a situation, we believe that labeling them is the most appropriate step to take rather than ignoring them at all. To our knowledge, such analysis of closer to real-world data distribution was lacking in the current literature on surgical phase recognition. Previous studies were performed on different surgery types, e.g., cataracts [76], gastric bypass [77], sleeve gastrectomy [49, 78, 79], right hemicolectomy [49], appendectomy [49]), laparoscopic videos collected in a single medical center but from different surgery types [80], and different scale of laparoscopic cholecystectomy datasets [10, 33, 35]. The three latter studies used private datasets larger than Cholec80, which might have included the phases presented in this study. Nevertheless, the authors focused on the model performance without detailing the difference in data distribution observed between their dataset and public ones.

Next, we studied how existing deep neural network architectures which had shown top performance in Cholec80 performed when trained on Insel dataset. We focused on two-stage methods due to their higher performance in Cholec80 benchmark, namely TeCNO [29] and Trans-SVNet [31]. Our aim was not to make an exhaustive comparison between many models. Instead, we explored the potential impact of video data distribution on model selection. The models trained on Insel dataset recognized major phases with at least 76 percent recall. The performances of models trained on Insel dataset were lower than the performance reported on Cholec80, with a margin

of 7% for accuracy, and almost 20% for precision and recall. This was expected given the higher number of phases and higher complexity of phases in Insel dataset.

Comparing Trans-SVNet and MSTCN-based models on Insel dataset, the performance gap was not as significant as it was reported on the Cholec80 dataset [31]. For example, the gap between TeCNO and Trans-SVNet-WCE was less than 1% for accuracy, precision, and recall on Insel dataset (Table 4.5), compared to the 4% gap for precision and 2% gap for accuracy on Cholec80 [31]. This difference may be explained by the longer tail in Insel phase distribution compared to Cholec80 (Fig. 4.4 and Fig. 4.5).

The absence of some phases in individual videos in Insel dataset made it difficult for the temporal models to capture these phases. Consequently, Trans-SVNet could not bring much improvement compared to the TeCNO model on which it was based.

The models struggled to detect less prevalent surgical phases. We attempted to tackle this imbalance by weighting the samples in the Trans-SVNet loss function according to the phase prevalence. This improved the model’s capability at detecting extremely under-represented phases such as *gallbladder drainage* and *liver biopsy*. Yet, the model still displayed lower performance in minority phases. Interestingly, the online version of MSTCN (TeCNO) proved better at recognizing minority phases compared to its offline version (TeCNO-Off). This was counter-intuitive since the offline model is intrinsically able to process more contextual information: for each frame, it considers both preceding and succeeding frames. This lower performance on minority phases may be due to offline models producing smoother predictions that favor longer and more common phases. One solution might be to apply few-shot learning (FSL) methods or different data augmentation techniques to tackle this extreme class imbalance. However, video frames of neighboring or distant phases have high visual similarity, which would make it challenging for FSL methods to find good features to recognize classes having small sample sizes. To the best of our knowledge, such a detailed analysis of the data characteristics is missing in the surgical AI literature. Future studies should explore in detail how to make the best use of these rare phases and propose the most suitable solution for recognizing them.

Our study has several limitations: (i) Insel dataset consisted of data from a single medical center. Although we considered all videos being recorded over a given period, the exact distribution of the videos (e.g., the presence of specific surgical phases) may differ across medical centers, (ii) we used a single publicly available dataset for comparative analysis. Combining data from more medical centers in future studies may yield interesting results. Besides, once more public datasets become available

it would be interesting to repeat our experiments, (iii) we studied a single type of surgery: laparoscopic cholecystectomy. Although this is the most commonly studied surgery type for surgical data science [9], the results presented in this study might change across surgery types. Similar studies are needed on other surgery types for better comparison, (iv) we used two types of neural network architectures in our benchmarking experiments because it was not possible in terms of time and effort to exhaustively compare all neural network models in our study. However, future studies can apply new models, or models that we had overlooked in our current study, (v) out-of-body (oob) frames were ignored in our study. OOb frames are content that is not actually part of the surgical phases. During the surgery, the laparoscope is often pulled out for cleaning. The camera still keeps recording the outside content, i.e., the room and the surgery staff, therefore this content becomes part of the surgery video. For privacy reasons, this content was detected to anonymize or deidentify the patient and the surgery staff [81]. Another work identified oob frames in the first stage of their two-stage model and avoid feeding them to their second-stage model [82]. The reason was to protect their final second-stage model from the disruptive effect of oob frames because these frames have a very different pixel intensity distribution than inside-body frames. However, identifying oob frames was not included in the scope of this thesis. In a real-world application, an initial oob classification model can be trained and applied before the phase recognition pipeline. Only, frames that are predicted as inside-body is further sent as input to the phase recognition pipeline.

Overall, our results shed the light on a possible model selection bias when developing models on a single publicly available dataset. The performance of surgical phase recognition has already reached a promising 90 percent accuracy on Cholec80. Yet, the discrepancy between Cholec80 data distribution and data observed in more real-world scenarios might bias the model selection, e.g. when deciding the best architecture, weights, and loss function. For example, working solely on Cholec80, the developed models might become better at recognizing predominant phases but fail at capturing extremely under-represented phases.

Automation of phase recognition is a prerequisite to numerous applications ranging from post-operative analysis to intra-operative assistance. In a real-world AI application, surgeons will perform in the rare phases we observed in Insel dataset. Therefore, it is important to analyze and understand the data distribution in real-world scenarios to develop robust models that are tailored at recognizing these rare phases. Nevertheless, more data and dedicated modeling efforts are necessary to improve the performance of these complex phases.

CROSS-DATASET TRANSFER LEARNING FOR SURGICAL PHASE RECOGNITION

5.1 Overview

Transfer learning for visual recognition tasks has been a widely studied topic in the context of deep learning [45, 83, 84]. The aim is to transfer the knowledge from a model trained on a source dataset to a model that will be trained on a target dataset. Due to the depth of the deep learning models, the parameters to be learned reach many millions or more. Eventually, learning such a huge set of parameters requires samples in orders of magnitude sizes compared to the parameter count. In case of data scarcity, the trained model suffers from the generalization problem, which is one of the main problems of machine learning. The lack of model generalization means that the model performs well enough in the source data distribution that it has been trained on, however, performing poorly on data having a different distribution. This is known as the domain shift problem, which is tackled by the domain adaptation methods as described in [84]. The particular benefit of transfer learning is to train models where data is limited. This is particularly useful in settings where data and annotations from a source domain are more available than in a target domain.

Transfer learning terminology describes four main concepts: the source domain, the source task, the target domain, and the target task, where the domain and dataset concepts are sometimes used interchangeably. These concepts are illustrated in Figure 5.1. A model trained on the source domain can transfer its knowledge, i.e., the weights and model parameters, to a model trained on the target domain. The widely adapted way of transfer learning is that the architecture of the target model is a copy of the source one, and it borrows the weights and other parameters from the source. Usually, the weights from the earlier layers are kept frozen in the target model, meaning they are not retrained. If they will be trained, the learning rate is adjusted to be smaller to allow the target model gradually adapt the weights to the target domain. If the weights are changed too fast, the model might risk overshooting the global minima.

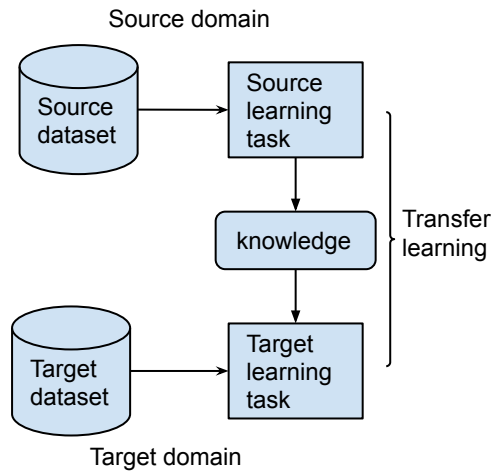


Figure 5.1 Illustration of the transfer learning concept. A model trained on a source domain transfers its knowledge to a target domain. The target domain, where often data and labels are scarce, benefits from the transferred knowledge.

The similarity between the domains and learning tasks affects the convergence of the target model and the parameters that will be transferred. If both domains have a similar distribution, meaning that images are recorded in similar conditions and have similar content, then the target model can benefit from using the source model with minimal change. The weights and model hyperparameters can be copied, and they can mostly remain unchanged in the target. This can result in faster training of the target model. On the other hand, if domains are completely different, transfer learning can still be useful but mostly for initializing the target model. Not only between the domains, but the dissimilarity between learning tasks is important and it can mostly occur in three different ways. First, both the source and the target tasks can be the same, as well as the labels. An example can be the image classification task involving the same sets of classes in the source and the target. This can be considered the easiest case, and transfer learning is trivial. Second, the source and the target tasks can be the same, but labels can be different. For example, the same image classification can involve different sets of classes (objects) in the source and the target. Third, the tasks and the associated labels can be different between the source and the target. An example is the image classification task as the source and the object localization task as the target. Here, both the tasks and the labels are different. However, transfer learning can still help in that case. For example, weights of the earlier convolutional layers can be transferred and fine-tuned in the target domain.

Transfer learning methods have been applied many times to date in the field of computer vision. Because image classification, object localization, and segmentation algorithms often require a lot of labeled data, transfer learning has been one of the most common methods for these tasks. This fact did not change for surgical phase

recognition. Since the surgical videos available in almost all datasets are not sufficient to train from scratch, the preferred method was mostly transfer learning. A large dataset of laparoscopic cholecystectomy videos was used in a recent study to test the performance of transfer learning for surgical phase recognition[10]. Their dataset included 1243 videos from 5 different data centers, including Cholec80. This amount of data was the largest known to date. In this study, using transfer learning from a large amount of source data and fine-tuning their network with almost 200 videos in the target dataset, it was possible to achieve comparable performance to the one with using all videos in the target dataset.

In this work, we are interested in transferring knowledge for the surgical phase recognition task learned in different domains (datasets). The source domain is described by the data (videos and phase annotations) that has been provided by Insepsital 3.3. The target domain is described by the Cholec80 dataset explained in an earlier section 3.2. The Insel dataset is larger than the Cholec80, containing 384 videos of Lap Chole videos while Cholec80 contains 80 videos. In the rest of this chapter, the Methods section describes our methodology for transfer learning, and the Results section presents our results. We discuss the results and future work in the Discussion section.

5.2 Methods

Our methodology is based on transfer learning with model fine-tuning. We relabeled the Cholec80 dataset based on our Insel annotation guidelines (3.5) because phases in Insel were a superset of Cholec80. Hence, upgrading Cholec80 labels would allow us to apply the model that has been trained with Insel directly on Cholec80 videos, even without the need for fine-tuning. We perform two different fine-tuning strategies and compare their results in the Cholec80 dataset. We experimented with two slightly different learning rates, 5×10^{-4} and 5×10^{-5} . The learning rate affects the model's convergence speed. Using a larger learning rate has the risk of rapidly changing the weights of the target model, which would cause overshooting the global minima. On the other hand, using extremely small learning rates might require too many iterations to converge. The best strategy would be to try with reasonable small values and tune it on the validation set.

5.2.1 Using ImageNet Dataset as the Source

Our first strategy assumes that no labeled data from the surgical video domain is available as the source dataset. Therefore, we use another domain as our source

domain. One of the widely accepted strategies for fine-tuning vision tasks is using pre-trained models from ImageNet [57]. ImageNet contains 1.2 Million images of daily objects (1000 categories) in its training set, which makes it one of the largest publicly available datasets. Using ImageNet as the source dataset, although its data distribution is quite different from surgical images its weights can be used to initialize the weights of the target model [45, 54, 83].

Here, we use the same two-stage pipeline, namely the ResNet followed by the online MSTCN (the TeCNO model we described earlier in 4.2.2). We also report the results of the ResNet model alone, to demonstrate a comparison of both models.

Our source ResNet model has been trained with ImageNet, and weights of all layers except the final linear layer are transferred to initialize the target model. The final linear layer is replaced with an empty one having 8 outputs, i.e., the number of classes in the Cholec80. Then, all layers are retrained with varying numbers of labeled Cholec80 videos at each experiment. We experiment with different fine-tuning subset sizes, starting from 5 and ending with 40. For every new experiment, we add 5 more videos and repeat the fine-tuning. At each experiment, the fine-tuned model is evaluated by the standard validation and test splits of Cholec80. The cholec80 validation set contains 8 videos (videos between 41 and 48), and the test set contains the remaining 32 videos (videos between 49 and 80).

The second stage temporal model (TeCNO), on the other hand, is trained from scratch, using the features extracted from the layers of the model fine-tuned with Cholec80. The number of outputs in the final layer of the TeCNO is 8, which reflects the count of classes in Cholec80. This fine-tuning strategy is illustrated in Figure 5.2. As shown, images in ImageNet and Cholec80 have different distributions. First, the model is trained in the source dataset (ImageNet). Weights of the trained Resnet50 source model except the final fc layer are copied and transferred to the model in the target dataset (Cholec80). Finally, the target model is fine-tuned with the Cholec80 dataset. During fine-tuning with Cholec80, we used early stopping to reduce the risk of overfitting. The f1-score computed on the validation set is used to decide on early stopping. We allowed 5 epochs without improvement before early stopping. The maximum number of epochs was set to 40. The learning rate was 5×10^{-3} and the batch size was 64. We temporally downsampled the videos from 25 to 1fps.

5.2.2 Using Insel dataset as the Source

We repeat the same two-stage modeling pipeline used in the previous experiment, i.e., ResNet and the TeCNO models. This experiment simulates a scenario where we have much more samples in a source domain, which has a similar data distribution

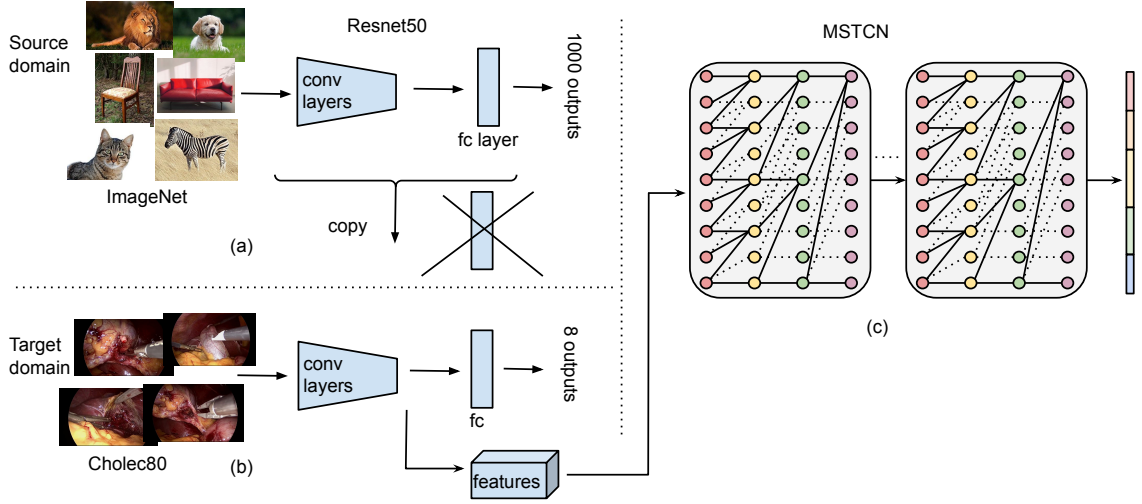


Figure 5.2 Illustration of transfer learning from ImageNet dataset. (a) First, Resnet50 is trained with ImageNet. (b) Weights except the final linear layer are transferred to the target domain, Cholec80. The transferred Resnet50 weights are fine-tuned with the Cholec80 training set. Finally, features are extracted from ResNet’s final conv layer. (c) Online MSTCN (TeCNO 4.2.2) is trained from scratch, using features extracted in the previous step. Images in (a) are not part of the original ImageNet dataset, but they are representative of the classes in ImageNet. (<https://commons.wikimedia.org/wiki,> CC by 4.0)

to the target domain data. Hence, the constraints are more relaxed compared to the previous experiment. Our source dataset in this case is the Insel dataset. Our target dataset is the same as the previous one, Cholec80. A considerable difference between this experiment and the previous one is, both the backbone Resnet50 model and the second-stage temporal model, MSTCN, are pre-trained with the Insel dataset. Hence, it is reasonable to expect that this can boost the performance of fine-tuning. Moreover, all layers including the final fully-connected layer in ResNet50 are used without replacement. This was because the phases in Insel dataset is a superset of the phases in Cholec80.

The parameters of the fine-tuning are as follows. The early stopping patience was 5 epochs. The batch size was equal to the length of the video (1 video per batch). The videos were temporally downsampled from 25fps to 1fps. The number of stages was 3, the number of layers in each stage was 10, and the number of feature maps at each layer was 32. The model was fine-tuned with a maximum number of 100 epochs, using a learning rate of 5×10^{-4} .

5.3 Results

5.3.1 Performance in Terms of Aggregated Metrics

We compare the performance of the models, i.e., ResNet alone, and the TeCNO (Resnet+MSTCN), with respect to varying fine-tuning sizes (x-axis), source datasets (ImageNet or InseI), and performance metrics (accuracy, and f1-score). We add the results obtained without fine-tuning for the completeness of the comparison. Figure 5.3 shows our results. For both performance metrics and both models, transferring

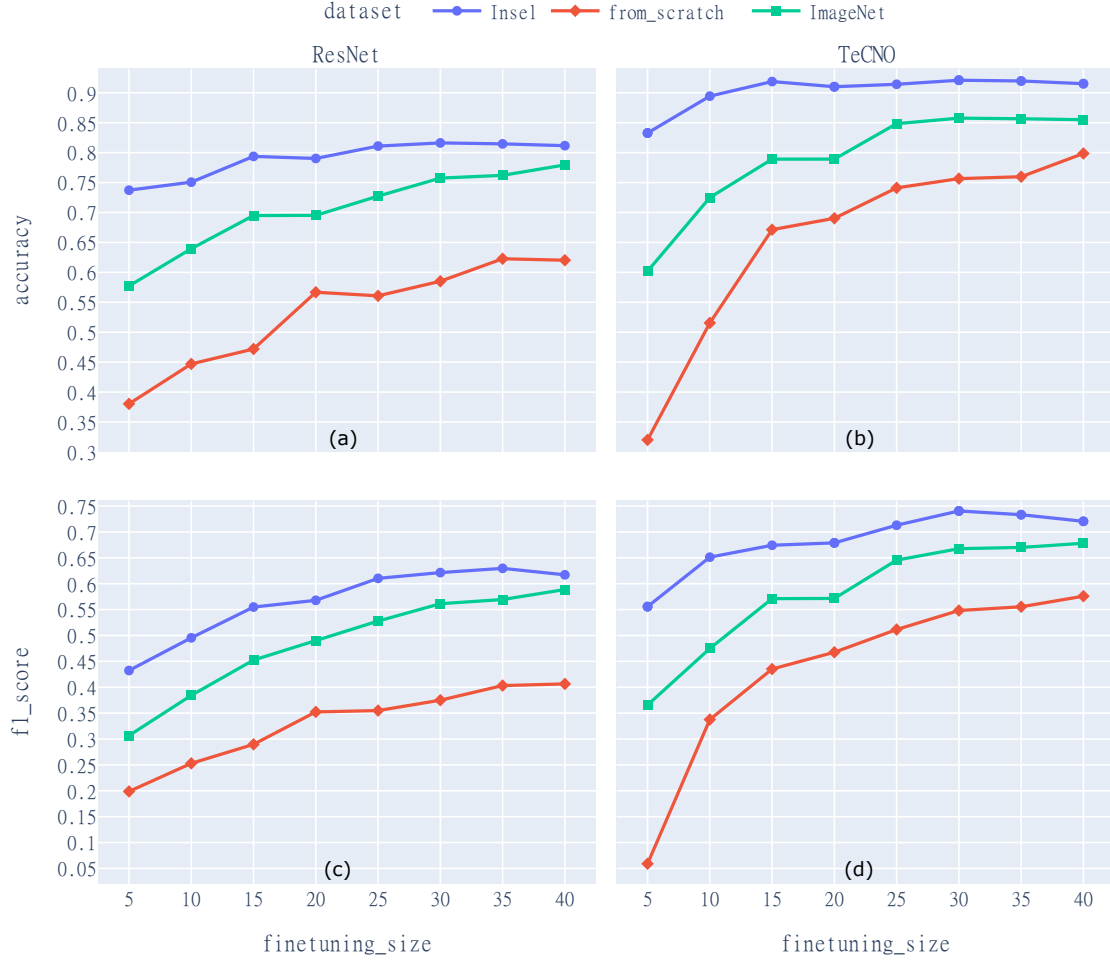


Figure 5.3 Results of transfer learning from various sources. Training without transfer learning is included for completeness (the red curve). The results show that transferring from the InseI worked the best, both in terms of accuracy and f1-score.

the model parameters from a pre-trained model worked much better than training it from scratch with the target dataset. The gap was even larger for the ResNet model than it was for the TeCNO model (the left column vs the right column in Figure 5.3). The TeCNO model performed better than the ResNet in both performance metrics. This is expected, as the TecNO performs temporal modeling on the outputs of the ResNet. The performance of both models improves as the finetuning video count increases, which is also expected. For transferring from the InseI dataset, it is interesting to see

that the TecNO model's accuracy is already saturated at 15 finetuning videos, and reaches its top performance among other finetuning video counts (Figure 5.3(b)). We can see a similar behavior for the ResNet model too, however, it reaches its top performance at a later point, i.e., at 25 finetuning videos. This result is important for a use case where labeled data is limited. It shows that the accuracy we can get with 40 labeled training videos in the target dataset is already achievable with fewer videos, i.e., less than 15 videos for the ResNet model (Figure 5.3(a), blue and green curves), and less than 10 videos for the TecNO model (Figure 5.3(b), blue and green curves), when both are transferred from the ImageNet. The accuracy of both models is lower when the source dataset is ImageNet but not Insel. This is also expected because the data distribution shift between the ImageNet and the Cholec80 is much higher than the one between the Insel and the Cholec80. Hence, the former is a more difficult use case, where the model needs more fine-tuning videos to close the performance gap between the latter case.

The f1-score for both models and both datasets show a similar trend with accuracy. An interesting result regarding the f1-score is that it reaches its top performance at 30 finetuning video count for the Insel experiment (the blue curve in Figure 5.3(d)). Then, the performance starts to drop with more fine-tuning videos, probably due to model overfitting. The reason for the difference between the accuracy result (top performance in 15 vs 30 videos) could be that performance of some of the individual phases, i.e., maybe the shorter phases, continued improving with more videos than 15, and finally topped at 30 videos. However, this is not the case for the accuracy metric. When we compare these metrics, accuracy calculation simply performs micro-averaging. It does not consider the classes individually during the calculation. So, the longer or more prevalent classes can be more dominant in its result. If a few longer phases and many short phases are present in the dataset, which is the case for the Cholec80 3.2, the accuracy could still give a high score. However, f1-score is badly affected by the shorter phases, because they are likely to perform poorly and in the end results from the classes are averaged.

The results for the precision and the recall show a similar trend, therefore, we do not find it necessary to discuss them here. These results can be found in the Appendix (Figure A.1).

5.3.2 Performance in Terms of Individual Phases

Aggregated performance metrics provide an overall performance of the models. For example, the accuracy metric presents a good overall performance of a classification model. However, more prevalent classes (longer phases in case of surgery) have a

larger effect on the final outcome. Metrics such as precision, recall, and f1-score cancel out the support of the classes, i.e., normalize the metric according to the count of the predictions for each class. Hence, it is a good practice to do performance analysis based on individual classes, in our case, phases. Therefore, in order to present a more detailed analysis of the model performance, we conduct individual phase-based performance evaluation on the target dataset.

Cholec80 originally includes seven phases of Lap Chole surgery (section 3.2). The Clipping and the Cutting phases which are defined separately in the Insel dataset are presented as a combined single phase in Cholec80. We relabeled the Cholec80 videos to have a phase distribution closer to the one of the Insel dataset. We performed our phase-based performance analysis by using the fine-tuned TeCNO model only.

Our results are illustrated in Figure 5.4. For a given metric, each phase shows a different performance. Phases Calot triangle dissection and Gallbladder dissection are the highest performers. The lowest performers are the Gallbladder retraction and Preparation phases.

The performance for each phase often improves with respect to increasing fine-tuning size. An exceptional case is the Calot triangle dissection phase in Figure 5.4(b). Transferring the weights of the model pre-trained on the Insel dataset already provides very good knowledge to recognize this phase, even fine-tuned with 5 videos from the Cholec80 dataset. Adding more Cholec80 videos probably distorted the pre-trained weights so the model is confused to recognize this phase in Cholec80. However, the precision started to increase after fine-tuning size reached and passed 30 videos. The trend looks like it would have surpassed the initial performance, which is 0.96, unfortunately, we can not see the performance beyond 40 videos. Similarly, the recall for Gallbladder dissection starts with 0.98 at 5 fine-tuning videos and drops after (Figure 5.4(d)). It starts to raise again after the fine-tuning size of 30 and continues to improve. These two examples show that when the target dataset size is significantly smaller than the source one, i.e., 5 from Cholec80 and 269 from Insel, we might see a performance drop from the beginning until some cut-off point (the recall for GallbDissection and Clipping). After that point, we should expect the performance to rise again, as long as we get more samples from the target dataset.

The performance change (especially for precision and recall) with respect to the fine-tuning size is irregular for the model transferred from ImageNet (Figure 5.4 (a) and (c)), compared to the one transferred from Insel (Figure 5.4 (b) and (d)). This is reasonable because the domain shift between ImageNet and Cholec80 is greater than the one between Insel and Cholec80. Therefore, we see more performance fluctuations in Figure 5.4 (a) and (c).

The biggest improving phases are the Clipping and Cutting phases in all figures. This

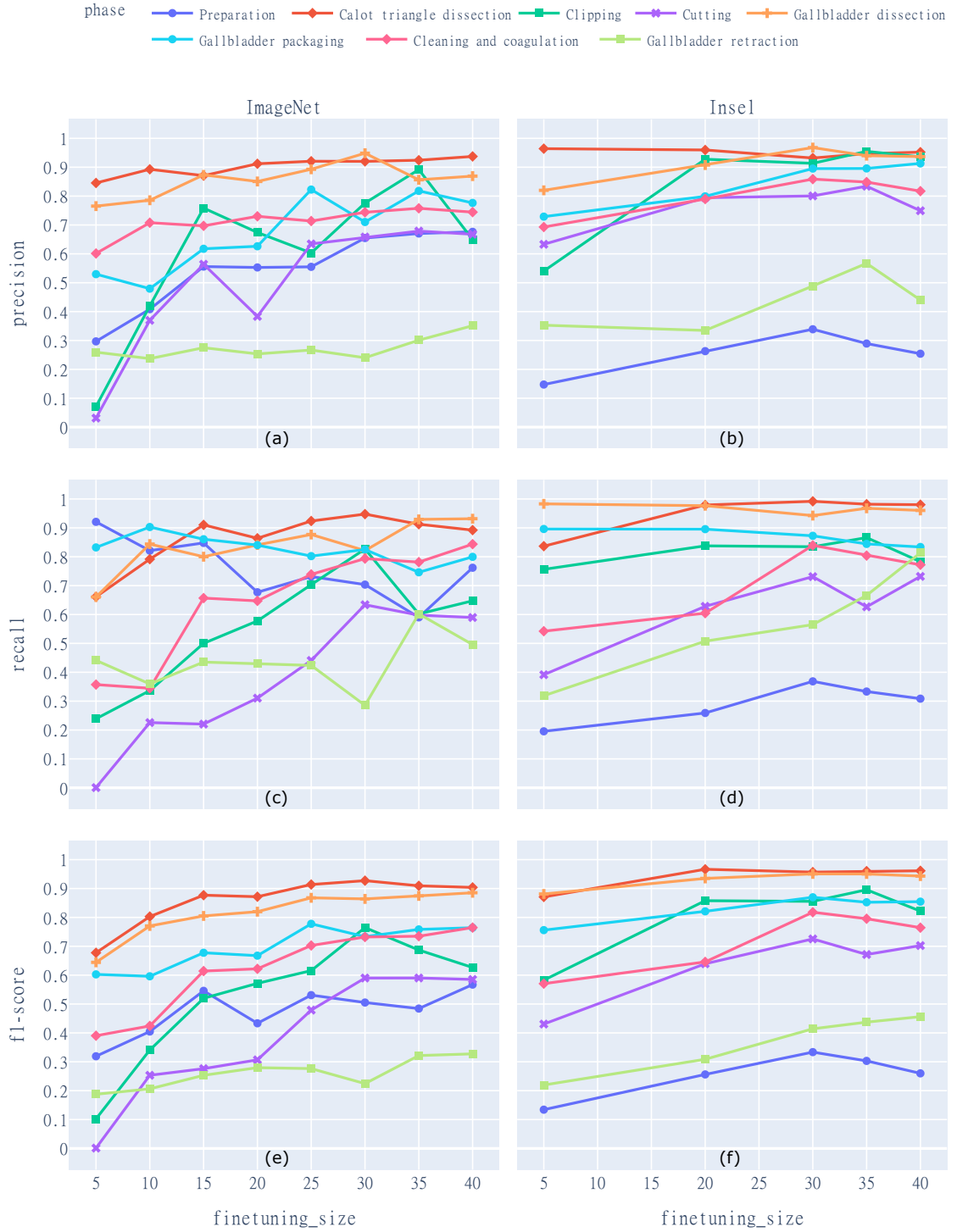


Figure 5.4 Illustration of the performance in the level of individual phases. Both precision, recall, and f1-score are presented. Results of fine-tuning from both Inset and ImageNet are provided.

is likely because there is a unique surgical instrument dedicated to each phase (clip applier for Clipping and scissors for Cutting) and they are present in the scene during these phases. So, it is easier to recognize these phases, i.e., better discriminate from others, thanks to these unique instruments. Even fine-tuning with a small number of

videos from the target domain helps with bigger improvement for such phases. On the contrary, phases such as Calot triangle dissection, Gallbladder dissection, Preparation, and Gallbladder retraction benefited less from fine-tuning. Multiple instruments often act together in these phases, which could have different brands and shapes (see the difference between instruments in 3.2 and 3.3). This requires more samples for the fine-tuned model from the target domain to adapt to such differences and perform more robustly in the target domain.

5.3.3 Performance in Terms of Individual Videos

Previous sections showed the performance of fine-tuning in terms of aggregated metrics and phase-level scores. However, evaluating the performance of a classifier in terms of single videos would demonstrate important benefits. For example, identifying the video-based error would allow for performing video-based error analysis. Finding poorly performing videos would allow improving the classifier performance by fixing potential labeling errors in these videos. Or, other issues such as different data distribution, imbalanced classes, or difficult labels can be identified. Hence, video-based error analysis can be used to find such low-performing videos, and allow identify the root causes of these errors.

In this section, we demonstrate the results in two different schemes for the 32 videos in the test set of Cholec80, i.e., for every video $v \in \{video49, video50, \dots, video80\}$. First, the video level f1-score for the highest and lowest 5 performing videos, and the median video, are shown for a fine-tuning size of 5. The video-level f1-score is computed by averaging the phase-level f1-score computed for that video. Similarly, the video-level accuracy is computed over the predictions and ground truth for that video. The results are shown in Table 5.1. Table 5.1 shows that there is quite a high variability between f1-score and accuracy across the videos. The lowest 5 f1-scores range between 0.3 and 0.43, while the highest 5 f1-scores range between 0.75 and 0.93. The median f1-score is 0.59, scored by *video71*. On the other hand, the lowest 5 accuracies range between 0.5 and 0.7, while the highest 5 accuracies range between 0.95 and 0.98. The median accuracy is 0.859, scored by *video56*.

Second, the performance is shown in terms of quantiles: $q \in \{0, 25, 50, 75, 100\}$ percent quantile of the accuracy, and f1-score values are shown for every fine-tuning size $s \in \{5, 10, 15, 20, \dots, 40\}$. Each quantile is a cut-off point indicating the percentage of the data up to that point. The first and the last quantiles show the lowest and the highest-performing video, respectively. The results are illustrated in Figure 5.5. Both sets of results are shown for the model transferred from the Insel dataset. The greatest change in f1-score across quantiles occurs at the lowest quantile (for the

Table 5.1 Illustration of the video-based performance in terms of the f1-score and accuracy, for 5 fine-tuning videos. The highest-performing five, the median, and the lowest-performing five videos are shown, from top to bottom.

rank	metric			
	f1-score		accuracy	
	video	score	video	score
1	video55	0.937	video70	0.981
2	video76	0.929	video76	0.965
3	video70	0.868	video74	0.961
4	video78	0.775	video50	0.957
5	video67	0.749	video58	0.950
median	video71	0.597	video56	0.859
28	video80	0.431	video61	0.702
29	video73	0.417	video62	0.664
30	video52	0.332	video65	0.578
31	video65	0.318	video77	0.546
32	video77	0.303	video54	0.498

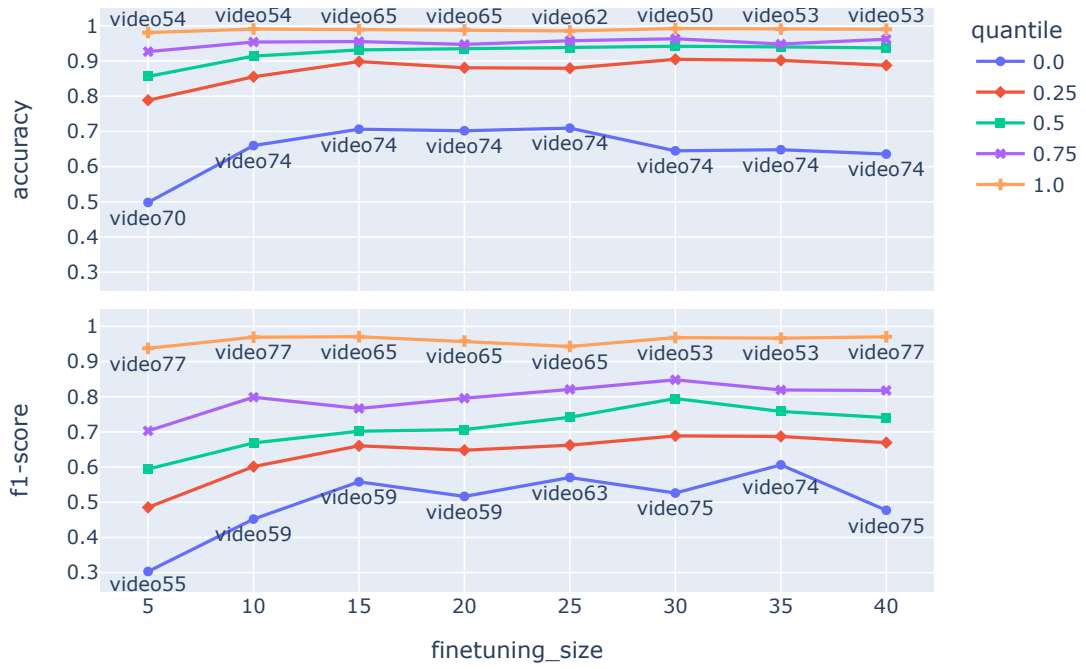


Figure 5.5 The illustration of video-based performance assessment using the quantiles. The overlaid text shows the video name associated with the video performing for the x and y at that point.

lowest performing video), between 5 and 35 fine-tuning sizes (f1-score improved from 0.3 to 0.6). Similarly, the greatest accuracy change occurred at the lowest quantile.

The highest quantiles for the accuracy and the f1-score have the least performance difference across varying fine-tuning sizes. *video74* is the lowest-performing video in terms of accuracy across varying fine-tuning sizes. *video59* and *video75* showed the lowest performance in terms of f1-score, which means these videos had high phase-based performance variance. *video74*, *video75*, *video59* are the most common low performers while *video53*, *video54*, *video77*, *video65* are the high performers.

5.3.4 The Effect of the Learning Rate

We compare the performance of the TeCNO model with respect to different learning rates used for fine-tuning. Figure 5.6 shows the change in accuracy with respect to both different learning rates and different fine-tuning subset sizes (labeled videos used for fine-tuning). The model transferred from ImageNet is extremely affected by being

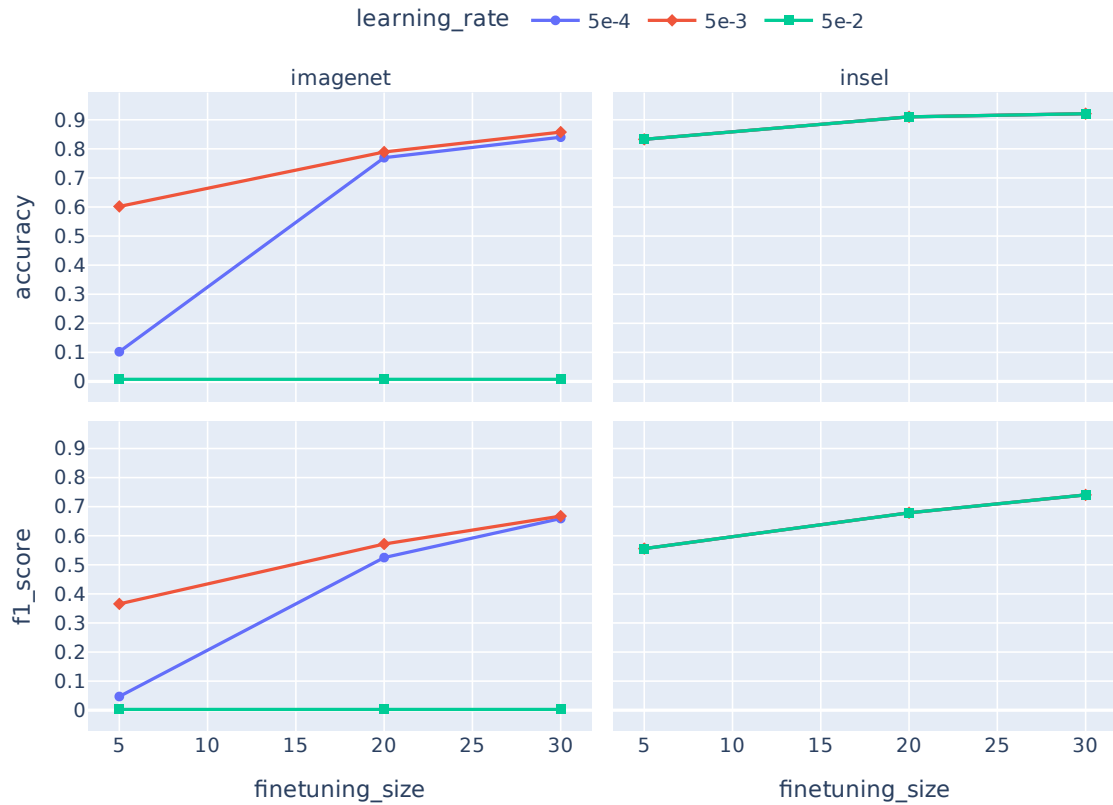


Figure 5.6 The effect of learning rate on the performance of the fine-tuned TeCNO model. The model transferred from ImageNet (left column) is very much affected by the learning rate. The best learning rate is 5×10^{-3} .

fine-tuned with different learning rates, while the model transferred from Inset is not affected. When we inspect the figures based on the source dataset we can list what we observe for the ImageNet case (the left column in Figure 5.6) as follows,

- 5×10^{-3} learning rate works better than 5×10^{-2} and 5×10^{-4} . 5×10^{-2} is the worst of all,
- the performance gap between 5×10^{-3} and 5×10^{-4} learning rates decreases as the fine-tuning size grows, and almost disappears after reaching a fine-tuning size of 20 videos,
- for the same fine-tuning size, the f1-score is lower than the accuracy for all learning rates,
- the accuracy and the f1-score for 5 fine-tuning videos was very low for 5×10^{-2} and 5×10^{-3} learning rates (top-left). Both metrics are highly correlated.

For the Insel case (the right column in Figure 5.6), the performance scores are not affected by the learning rate. Therefore, they seem to overlap in the figure. The accuracy was already above 80 percent for 5 fine-tuning videos, and it increases to above 90 percent for 20 fine-tuning videos. With a similar trend, the f1-score is 55 percent for 5 fine-tuning videos, and it increases to 67 percent for 20 fine-tuning videos.

5.4 Discussion

We explored different fine-tuning strategies for transfer learning across datasets. We simulated scenarios where limited labeled data is available for fine-tuning in the target dataset. Our target dataset featured the Cholec80, which is the most commonly used dataset for surgical vision tasks. One of the benefits of using Cholec80 is that our results would allow easier comparison with the state-of-the-art methods since Cholec80 is still widely used for research.

We included training the model from scratch, i.e., without fine-tuning, in Cholec80 for the completeness of the study. Model performance was the lowest when trained from scratch (at least 15% lower for the ResNet and 10% for the TeCNO). The performance was lower when the source dataset was ImageNet but not Insel. This is expected because the ImageNet data characteristics are quite different than Cholec80's and Insel's. This brings a challenge for the source model to adapt to these variations. However, for the TeCNO model, the performance gap became closer as the fine-tuning size increased. This was not true for the ResNet model (Figure 5.3). This was likely due to TeCNO being a temporal model, but ResNet was trained on individual frames. Hence, the temporal model is better adapted once more data from the target domain is available. Compared to training from scratch, the performance gain was huge when

the Insel was the source dataset and the TeCNO was the source model. The accuracy provided by training from scratch was surpassed even when 5 videos were used (Figure 5.3(b), blue and red curves). The performance difference between training from scratch and transferring from Insel dataset was even larger for the ResNet model. The Insel-ResNet model showed 74% accuracy with 5 fine-tuning videos whereas only 62% accuracy was obtained by fine-tuning the model from scratch with 40 videos. This experiment showed the impact of transfer learning from a domain with similar data distribution but having many more labeled videos (269 vs. 40).

Individual phases often showed an increased performance for increasing fine-tuning size, which was expected (Figure 5.4). A few phases (Clipping and Cutting) showed huge improvement quickly, by fine-tuning even with a small number of videos as 5 or 10. This was likely due to the appearance of unique surgical instruments in each of these two phases. The clip applier, for example, is mostly from a single brand in Cholec80, however, the Insel dataset contains two different brands (one with metal clips and the other with a plastic one) (see the clip appliers in 3.2 and 3.3). Since a single and more specific clip applier type occurs in Cholec80, the model probably adapted to it quicker with a small number of videos. However, for phases in which multiple instruments with more general shapes appeared together, the adaptation of the model was slower due to more uncertainty (Preparation, Gallbladder retraction, Gallbladder dissection, and Calot triangle dissection phases). Therefore, the improvement for such ambiguous phases was smaller.

In this thesis, we focused on an important task of surgical AI: deep learning-based surgical phase recognition. Our study was organized into three main parts.

In the first part of our study, we conducted a detailed literature review. We examined deep learning and computer vision-based methods for surgical phase recognition in recent years. Then, we presented a detailed description of our Insel dataset and Cholec80. We described the acquisition and annotation process of the Insel dataset that contains 384 Lap Chole videos. We also briefly mentioned other public and private datasets and compared their phase definitions.

In the second part, we described our methodology and presented the results of our Insel dataset and model performance comparison. We compared the characteristics of our dataset with the most commonly used publicly available dataset for surgical AI, Cholec80. Our comparison showed a discrepancy in the data distributions of videos between these datasets. Next, we studied how existing deep neural network architectures which had shown top performance in Cholec80 performed when trained on our dataset. Models struggled to recognize extremely underrepresented surgical phases which are characteristics of our dataset. Our results highlighted the need to better understand the distribution of the video data phase recognition models trained on. In particular, we shed the light on the risk of model selection bias when developing models on video data that under-represents real-world scenarios. Our analysis and results can be particularly interesting to researchers who do not have access to complex real-world datasets. Future work should focus on improving the model performance in recognizing extremely underrepresented surgical phases.

Surgical phase recognition is still not a solved problem. The results obtained in Cholec80 dataset have already reached 90 percent accuracy and above[31]. However, the performance of the same models trained and tested on our more complex dataset was much less. Our dataset contained challenges such as extremely short and long surgical phases both within and across videos. In addition, some phases were only present in less number of videos. For real-world deployment concerns, these problems

pose a significant challenge. It is extremely challenging to solve such problems with only extensive modeling, but we believe that scaling the data is the key factor for improved performance. Models need to be trained and compared on larger-scale datasets having closer-to-real-world distribution. Moreover, in addition to standard performance metrics, models need to be evaluated with more clinically relevant performance metrics. Collaboration between clinicians and computer scientists is necessary to derive more useful metrics and gather more data.

Several limitations apply to our study. (1) We used a single surgery type, laparoscopic cholecystectomy (Lap Chole). According to the literature [9] and discussions with our clinical partners, Lap Chole was the easiest to collect data and relatively easier in terms of procedural phases. (2) Our dataset consisted of videos obtained from a single medical center. It may be interesting to use data collected from different centers for more comprehensive performance analysis and comparison. (3) Our study examined the performance of two state-of-the-art two-stage models[29, 31]. Future studies may expand this study to compare more methods. It is not yet available, but it would be interesting to expand the study when other publicly available large-scale datasets become available. (4) Procedures are usually performed in two different ways: laparoscopy, and robot-assisted. Our study examined only laparoscopic videos. However, robot-assisted surgeries present different surgical tools and artifacts such as text overlay in the video frames. Therefore, our results may not directly transfer to robot-assisted surgery videos.

In the third part of the thesis, we studied the performance of transfer learning from a two-stage model that has been trained on a data source where the amount of labeled data was much higher than in a target dataset. Our target dataset was Cholec80. We experimented with varying numbers of labeled videos for fine-tuning in Cholec80. We incremented the fine-tuning set, starting with five labeled videos and adding five more videos in every new experiment until reaching the full set (40 videos). We tested the performance using various performance metrics and from different perspectives: the performance aggregated over all phases and videos in the dataset and the performance across individual phases and videos.

Model adaptation showed quicker convergence when the source dataset was the Insel, both for the ResNet backbone and the temporal model. When the source dataset was Insel, the performance obtained when the TeCNO was fine-tuned with only 5 Cholec80 videos was greater than the performance obtained by training the model from scratch with 40 Cholec80 videos (the blue vs. red curve in Figure 5.3(b)). Similarly, the Insel-TeCNO model performed better with 5 fine-tuning videos, than the ImageNet-TeCNO model with 20 fine-tuning videos (the blue vs. green curve in Figure 5.3(b)). The performance gap was getting closer as the fine-tuning video count

increased. In summary, when the source model is a large-scale dataset having similar characteristics to the target (Insel vs. Cholec80), fine-tuning the model with only fewer videos (5 videos) was even better than training the model from scratch with 40 target dataset videos (83% vs 80% accuracy).

The phase-based and video-based performance showed high variability. A subset of the phases benefited much more from the fine-tuning than another subset of the phases. Particularly, phases in which the variation of the active surgical instruments is lower, both in terms of count and type, benefited more from fine-tuning. These phases were Clipping, Cutting, and Cleaning&Coagulation. Other phases in which uncertainty is higher due to lack of instrument presence (this brings high ambiguity, especially in the absence of a temporal model) or multiple instruments having different brands and shapes between the source and the target benefited less. These phases were Preparation, in which instruments are less active, and Gallbladder retraction, which is a short and ambiguous phase due to less instrument presence. Future work should focus on more sophisticated adaptation methods. Methods for reducing the bias caused by the instruments would help with easier adaptation and better generalization to a new domain.

Deep learning is a key factor for the success of surgical phase recognition. The two-stage modeling architecture, i.e., a backbone model used for feature extraction and a temporal model trained on the time series of extracted features, is a powerful architecture. The use of dilated convolutions combined with residual learning[26] was also effective for long surgical videos. Vision transformer showed comparable or higher performance when a TCN backbone was used [30, 31]. The combination of TCNs and transformers provides a powerful combination where aggregation of spatial and temporally long information could be possible. We believe that future work should more focus on improving the data rather than model architecture. This is because the currently available models are data-hungry, and the size of the currently available datasets is not sufficient to feed these deep architectures. To increase the scale of the data and reduce the need for labeling experts AI researchers can develop methods for semi-automated labeling. It can be very useful to implement a model that only involves experts' feedback when the model has less confidence in difficult samples. Moreover, Self-supervision can also help with the initialization of phase recognition models [44, 49]. It is difficult for data scientists to grasp clinical domain knowledge on their own, to improve their models. Clinicians and data scientists should collaborate more often for knowledge sharing and discussing how ML models can be used in clinical practice.

REFERENCES

- [1] Y. Yamashita, T. Takada, Y. Kawarada, Y. Nimura, M. Hirota, F. Miura, T. Mayumi, M. Yoshida, S. Strasberg, H. A. Pitt, *et al.*, “Surgical treatment of patients with acute cholecystitis: Tokyo guidelines,” *Journal of hepato-biliary-pancreatic surgery*, vol. 14, no. 1, pp. 91–97, 2007.
- [2] L. Maier-Hein, S. S. Vedula, S. Speidel, N. Navab, R. Kikinis, A. Park, M. Eisenmann, H. Feussner, G. Forestier, S. Giannarou, *et al.*, “Surgical data science for next-generation interventions,” *Nature Biomedical Engineering*, vol. 1, no. 9, pp. 691–696, 2017.
- [3] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, N. Padoy, “Endonet: A deep architecture for recognition tasks on laparoscopic videos,” *IEEE transactions on medical imaging*, vol. 36, no. 1, pp. 86–97, 2016.
- [4] *Surgery room*. [Online]. Available: https://commons.wikimedia.org/wiki/File:Surgery_room.jpg (visited on 12/23/2022).
- [5] A. Krizhevsky, I. Sutskever, G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [6] S. Hochreiter, J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [8] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” *arXiv preprint arXiv:1609.03499*, 2016.
- [9] C. R. Garrow, K.-F. Kowalewski, L. Li, M. Wagner, M. W. Schmidt, S. Engelhardt, D. A. Hashimoto, H. G. Kenngott, S. Bodenstedt, S. Speidel, *et al.*, “Machine learning for surgical phase recognition: A systematic review,” *Annals of Surgery*, vol. 273, no. 4, pp. 684–693, 2021.
- [10] O. Bar, D. Neimark, M. Zohar, G. D. Hager, R. Girshick, G. M. Fried, T. Wolf, D. Asselmann, “Impact of data on generalization of ai for surgical intelligence applications,” *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.
- [11] N. Dalal, B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*, Ieee, vol. 1, 2005, pp. 886–893.

- [12] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.
- [13] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 711–720, 1997.
- [14] K. Kirtac, O. Dolu, M. Gokmen, "Face recognition by combining gabor wavelets and nearest neighbor discriminant analysis," in *2008 23rd International Symposium on Computer and Information Sciences*, IEEE, 2008, pp. 1–5.
- [15] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [16] S.-A. Ahmadi, T. Sielhorst, R. Stauder, M. Horn, H. Feussner, N. Navab, "Recovery of surgical workflow without explicit models," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2006, pp. 420–428.
- [17] N. Padoy, T. Blum, I. Essa, H. Feussner, M.-O. Berger, N. Navab, "A boosted segmentation method for surgical workflow analysis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2007, pp. 102–109.
- [18] T. Blum, N. Padoy, H. Feußner, N. Navab, "Modeling and online recognition of surgical phases using hidden markov models," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2008, pp. 627–635.
- [19] N. Padoy, T. Blum, H. Feussner, M.-O. Berger, N. Navab, "On-line recognition of surgical activity for monitoring in the operating room.," in *AAAI*, 2008, pp. 1718–1724.
- [20] T. Blum, H. Feußner, N. Navab, "Modeling and segmentation of surgical workflow from laparoscopic video," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2010, pp. 400–407.
- [21] N. Padoy, T. Blum, S.-A. Ahmadi, H. Feussner, M.-O. Berger, N. Navab, "Statistical modeling and recognition of surgical workflow," *Medical image analysis*, vol. 16, no. 3, pp. 632–641, 2012.
- [22] R. Cadene, T. Robert, N. Thome, M. Cord, "M2cai workflow challenge: Convolutional neural networks with time smoothing and hidden markov model for video frames classification," *arXiv preprint arXiv:1610.05541*, 2016.
- [23] Y. Jin, Q. Dou, H. Chen, L. Yu, P.-A. Heng, "Endorn: Recurrent convolutional networks for recognition of surgical workflow in cholecystectomy procedure video," *IEEE Trans on Medical Imaging*, 2016.
- [24] C. Lea, J. H. Choi, A. Reiter, G. D. Hager, "Surgical phase recognition: From instrumented ors to hospitals around the world," in *Medical image computing and computer-assisted intervention M2CAI—MICCAI workshop*, 2016, pp. 45–54.
- [25] M. Sahu, A. Mukhopadhyay, A. Szengel, S. Zachow, "Tool and phase recognition using contextual cnn features," *arXiv preprint arXiv:1610.08854*, 2016.

- [26] Y. A. Farha, J. Gall, “Ms-tcn: Multi-stage temporal convolutional network for action segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3575–3584.
- [27] Y. Jin, Q. Dou, H. Chen, L. Yu, J. Qin, C.-W. Fu, P.-A. Heng, “Sv-rcnet: Workflow recognition from surgical videos using recurrent convolutional network,” *IEEE transactions on medical imaging*, vol. 37, no. 5, pp. 1114–1126, 2017.
- [28] Y. Jin, H. Li, Q. Dou, H. Chen, J. Qin, C.-W. Fu, P.-A. Heng, “Multi-task recurrent convolutional network with correlation loss for surgical video analysis,” *Medical image analysis*, vol. 59, p. 101 572, 2020.
- [29] T. Czempiel, M. Paschali, M. Keicher, W. Simson, H. Feussner, S. T. Kim, N. Navab, “Tecno: Surgical phase recognition with multi-stage temporal convolutional networks,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2020, pp. 343–352.
- [30] T. Czempiel, M. Paschali, D. Ostler, S. T. Kim, B. Busam, N. Navab, “Opera: Attention-regularized transformers for surgical phase recognition,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2021, pp. 604–614.
- [31] X. Gao, Y. Jin, Y. Long, Q. Dou, P.-A. Heng, “Trans-svnet: Accurate phase recognition from surgical videos via hybrid embedding aggregation transformer,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2021, pp. 593–603.
- [32] Y. Jin, Y. Long, C. Chen, Z. Zhao, Q. Dou, P.-A. Heng, “Temporal memory relation network for workflow recognition from surgical video,” *IEEE Transactions on Medical Imaging*, vol. 40, no. 7, pp. 1911–1923, 2021.
- [33] Y. Ban, G. Rosman, T. Ward, D. Hashimoto, T. Kondo, H. Iwaki, O. Meireles, D. Rus, “Aggregating long-term context for learning laparoscopic and robot-assisted surgical workflows,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 14 531–14 538.
- [34] X. Ding, X. Li, “Exploring segment-level semantics for online phase recognition from surgical videos,” *IEEE Transactions on Medical Imaging*, vol. 41, no. 11, pp. 3309–3319, 2022.
- [35] T. Golany, A. Aides, D. Freedman, N. Rabani, Y. Liu, E. Rivlin, G. S. Corrado, Y. Matias, W. Khoury, H. Kashtan, *et al.*, “Artificial intelligence for phase recognition in complex laparoscopic cholecystectomy,” *Surgical Endoscopy*, pp. 1–9, 2022.
- [36] B. Zhang, B. Goel, M. H. Sarhan, V. K. Goel, R. Abukhalil, B. Kalesan, N. Stottler, S. Petculescu, “Surgical workflow recognition with temporal convolution and transformer for action segmentation,” *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–10, 2022.
- [37] J. E. Van Engelen, H. H. Hoos, “A survey on semi-supervised learning,” *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.
- [38] L. Jing, Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 4037–4058, 2020.

- [39] I. Funke, A. Jenke, S. T. Mees, J. Weitz, S. Speidel, S. Bodenstedt, "Temporal coherence-based self-supervised learning for laparoscopic workflow analysis," in *OR 2.0 context-aware operating theaters, computer assisted robotic endoscopy, clinical image-based procedures, and skin image analysis*, Springer, 2018, pp. 85–93.
- [40] A. P. Twinanda, G. Yengera, D. Mutter, J. Marescaux, N. Padoy, "Rsdnet: Learning to predict remaining surgery duration from laparoscopic videos without manual annotations," *IEEE transactions on medical imaging*, vol. 38, no. 4, pp. 1069–1078, 2018.
- [41] M. Pfeiffer, I. Funke, M. R. Robu, S. Bodenstedt, L. Strenger, S. Engelhardt, T. Roß, M. J. Clarkson, K. Gurusamy, B. R. Davidson, *et al.*, "Generating large labeled data sets for laparoscopic image processing tasks using unpaired image-to-image translation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2019, pp. 119–127.
- [42] T. Yu, D. Mutter, J. Marescaux, N. Padoy, "Learning from a tiny dataset of manual annotations: A teacher/student approach for surgical phase recognition," *arXiv preprint arXiv:1812.00033*, 2018.
- [43] Z. Wang, X. Ding, W. Zhao, X. Li, "Less is more: Surgical phase recognition from timestamp supervision," *arXiv preprint arXiv:2202.08199*, 2022.
- [44] S. Ramesh, V. Srivastav, D. Alapatt, T. Yu, A. Murali, L. Sestini, C. I. Nwoye, I. Hamoud, A. Fleurentin, G. Exarchakis, *et al.*, "Dissecting self-supervised learning methods for surgical computer vision," *arXiv preprint arXiv:2207.00449*, 2022.
- [45] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, "How transferable are features in deep neural networks?" *Advances in neural information processing systems*, vol. 27, 2014.
- [46] M. Byra, G. Styczynski, C. Szmigielski, P. Kalinowski, Ł. Michałowski, R. Paluszkiewicz, B. Ziarkiewicz-Wróblewska, K. Zieniewicz, P. Sobieraj, A. Nowicki, "Transfer learning with deep convolutional neural network for liver steatosis assessment in ultrasound images," *International journal of computer assisted radiology and surgery*, vol. 13, no. 12, pp. 1895–1903, 2018.
- [47] Y.-Y. Tsai, B. Huang, Y. Guo, G.-Z. Yang, "Transfer learning for surgical task segmentation," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 9166–9172.
- [48] J. Jaafari, S. Douzi, K. Douzi, B. Hssina, "Towards more efficient cnn-based surgical tools classification using transfer learning," *Journal of Big Data*, vol. 8, no. 1, pp. 1–15, 2021.
- [49] D. Neimark, O. Bar, M. Zohar, G. D. Hager, D. Asselmann, "'train one, classify one, teach one'—cross-surgery transfer learning for surgical step recognition," *arXiv preprint arXiv:2102.12308*, 2021.
- [50] O. Dergachyova, D. Bouget, A. Huaultmé, X. Morandi, P. Jannin, "Automatic data-driven real-time segmentation and recognition of surgical workflow," *International journal of computer assisted radiology and surgery*, vol. 11, no. 6, pp. 1081–1089, 2016.

- [51] C. Lea, R. Vidal, G. D. Hager, "Learning convolutional action primitives for fine-grained action recognition," in *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2016, pp. 1642–1649.
- [52] A. Zia, L. Guo, L. Zhou, I. Essa, A. Jarc, "Novel evaluation of surgical activity recognition models using task-based efficiency metrics," *International journal of computer assisted radiology and surgery*, vol. 14, no. 12, pp. 2155–2163, 2019.
- [53] B. Zhang, A. Ghanem, A. Simes, H. Choi, A. Yoo, "Surgical workflow recognition with 3dcnn for sleeve gastrectomy," *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, no. 11, pp. 2029–2036, 2021.
- [54] K. Kirtac, N. Aydin, J. L. Lavanchy, G. Beldi, M. Smit, M. S. Woods, F. Aspart, "Surgical phase recognition: From public datasets to real-world data," *Applied Sciences*, vol. 12, no. 17, p. 8746, 2022.
- [55] *Dense neural network*. [Online]. Available: https://commons.wikimedia.org/wiki/File:Neural_network.svg (visited on 01/15/2023).
- [56] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.
- [57] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [58] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
- [59] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [60] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [61] O. Ronneberger, P. Fischer, T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.
- [62] Y. Souri, Y. A. Farha, E. Bahrami, G. Francesca, J. Gall, "Robust action segmentation from timestamp supervision," *arXiv preprint arXiv:2210.06501*, 2022.
- [63] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, T. Liu, "On layer normalization in the transformer architecture," in *International Conference on Machine Learning*, PMLR, 2020, pp. 10 524–10 533.
- [64] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

- [65] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.
- [66] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 568–578.
- [67] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 22–31.
- [68] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollár, R. Girshick, "Early convolutions help transformers see better," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 392–30 400, 2021.
- [69] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, C. Schmid, "Vivit: A video vision transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6836–6846.
- [70] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, H. Hu, "Video swin transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3202–3211.
- [71] D. Neimark, O. Bar, M. Zohar, D. Asselmann, "Video transformer network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3163–3172.
- [72] M. Wagner, B.-P. Müller-Stich, A. Kisilenko, D. Tran, P. Heger, L. Mündermann, D. M. Lubotsky, B. Müller, T. Davitashvili, M. Capek, *et al.*, "Comparative validation of machine learning algorithms for surgical workflow and skill analysis with the heichole benchmark," *arXiv preprint arXiv:2109.14956*, 2021.
- [73] R. Stauder, D. Ostler, M. Kranzfelder, S. Koller, H. Feußner, N. Navab, "The tum lapchole dataset for the m2cai 2016 workflow challenge," *arXiv preprint arXiv:1610.09278*, 2016.
- [74] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.
- [75] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, Soviet Union, vol. 10, 1966, pp. 707–710.
- [76] O. Zisimopoulos, E. Flouty, I. Luengo, P. Giataganas, J. Nehme, A. Chow, D. Stoyanov, "Deepphase: Surgical phase recognition in cataracts videos," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2018, pp. 265–272.
- [77] S. Ramesh, D. Dall’Alba, C. Gonzalez, T. Yu, P. Mascagni, D. Mutter, J. Marescaux, P. Fiorini, N. Padoy, "Multi-task temporal convolutional networks for joint recognition of surgical phases and steps in gastric bypass procedures," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–9, 2021.

- [78] D. A. Hashimoto, G. Rosman, E. R. Witkowski, C. Stafford, A. J. Navarrete-Welton, D. W. Rattner, K. D. Lillemoe, D. L. Rus, O. R. Meireles, “Computer vision analysis of intraoperative video: Automated recognition of operative steps in laparoscopic sleeve gastrectomy,” *Annals of surgery*, vol. 270, no. 3, p. 414, 2019.
- [79] B. Zhang, A. Ghanem, A. Simes, H. Choi, A. Yoo, A. Min, “Swnet: Surgical workflow recognition with deep convolutional network,” in *Medical Imaging with Deep Learning*, PMLR, 2021, pp. 855–869.
- [80] H. Wei, F. Rudzicz, D. Fleet, T. Grantcharov, B. Taati, “Intraoperative adverse event detection in laparoscopic surgery: Stabilized multi-stage temporal convolutional network with focal-uncertainty loss,” in *Machine Learning for Healthcare Conference*, PMLR, 2021, pp. 283–307.
- [81] M. Zohar, O. Bar, D. Neimark, G. D. Hager, D. Asselmann, “Accurate detection of out of body segments in surgical video using semi-supervised learning,” in *Medical Imaging with Deep Learning*, PMLR, 2020, pp. 923–936.
- [82] A. Bayat, K. Kirtac, J. Schwerin, M. Stenzel, M. Smit, F. Aspart, *et al.*, “Robustness against out of distribution video frames in online surgical workflow recognition with temporal convolutional networks,” in *Medical Imaging with Deep Learning*, 2022.
- [83] M. Oquab, L. Bottou, I. Laptev, J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724.
- [84] Z. Luo, Y. Zou, J. Hoffman, L. F. Fei-Fei, “Label efficient learning of transferable representations across domains and tasks,” *Advances in neural information processing systems*, vol. 30, 2017.

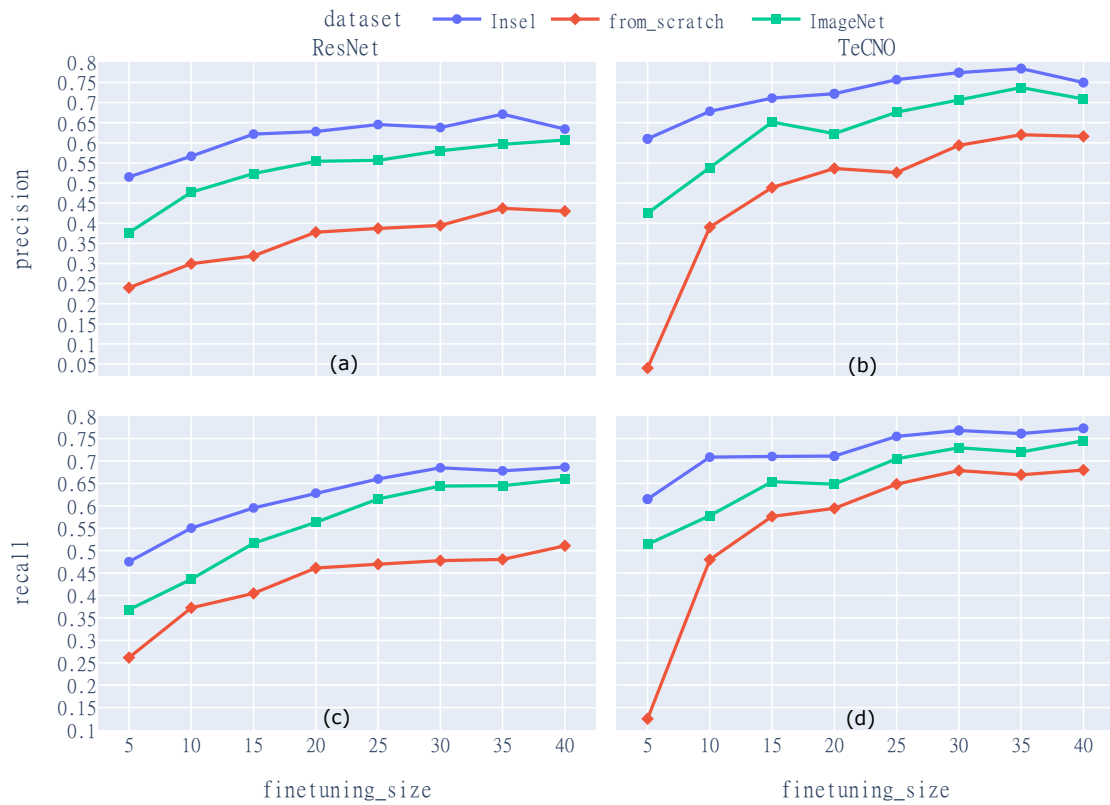


Figure A.1 Results of transfer learning from various sources. Training without transfer learning is included for completeness (the red curve). The results show that transferring from the Inset worked the best, both in terms of precision and recall.

PUBLICATIONS FROM THE THESIS

Papers

1. K. Kirtac, N. Aydin, J. L. Lavanchy, G. Beldi, M. Smit, M. S. Woods, F. Aspart, "Surgical Phase Recognition: From public datasets to real-world data," *Applied Sciences*, vol. 12, no. 17, p. 8746, 2022.

