Automatic Summarization of Endoscopic Surgical Videos

Daniel King

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Electrical Engineering

University of Washington

2022

Reading Committee:
Blake Hannaford, Chair
Linda Shapiro

Program Authorized to Offer Degree:
Department of Electrical Engineering

University of Washington

Abstract

Automatic Summarization of Endoscopic Surgical Videos

Daniel King

Chair of the Supervisory Committee:
Blake Hannaford
ECE

Endoscopic endonasal surgery is a medical procedure that utilizes an endoscopic video camera to view and manipulate a surgical site accessed through the nose. Despite thethese surgeries being video recorded, these videos are seldom reviewed or even saved in patient files due to the size and length of the video file. Editing to a manageable size may necessitate viewing 3 hours or more of surgical video and manually splicing together the desired segments. We suggest a novel multi-stage video summarization procedure utilizing deep semantic features, tool detections, and video frame temporal correspondences to create a representative summarization. Summarization by our method resulted in a 98.2% reduction in overall video length while preserving 84% of key medical scenes on our data set. Furthermore, resulting summaries contained only 1% of scenes with irrelevant detail such as endoscope lens cleaning, blurry frames, or frames external to the patient. A commercial summarization tool not designed for surgery only preserved 57% of key medical scenes in similar length summaries, and included 36% of scenes containing irrelevant detail.

The specific research contributions are then as follows:

1. A method of automatic summarization of endoscopic videos

2. The application of convolutional neural network frame classifiers based on convolutional neural networks towards more accurate summarization

3. A method of detecting shot boundaries based on convolutional features

4. The creation of a tool detection dataset in endoscopic videos

5. The analysis of several computer vision object detectors in the application of the created dataset

6. The utilization of tool presence to identify surgical stage

7. The use of surgical stage to improve item 1 summarization accuracy

8. The creation of an application and GUI to automatically create video summarizations

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

CNN: Convolutional Neural Network

HMM: Hidden Markov Model

ESS: Endoscopic Sinus Surgery

ESBS: Endoscopic Skull Base Surgery

SVM: State Vector Machine

LSTM: Long-Short-Term Memory

SHOT: In the context of video summary, this represents a series of roughly continuous, generally similar frames. A series of shots comprise a full video

SUBSHOT: In the context of video summary, this represents a series of continuous similar frames within a shot. A series of subshots comprise a shot.

BOUNDARY: In the context of video summary, this represents the frame that separates two distinct shots

IRRELEVANCE: Ratio of irrelevant shots to total shots of a summary. An irrelevant shot includes frames that contain no meaningful information or information irrelevant to intended understanding of the full parent video.

ACCURACY: Percentage of ground-truth, medically relevant scenes that are included in a summary

MAP: Mean Average Precision, the average of the AP of all classes.

AP: Average Precision, the area under the precision-recall curve

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to Prof. Blake Hannaford, who granted time, attention, and opportunity beyond my expectation. Additionally, the author would like to extend thanks to Dr.s Adidharma, Bly, Moe, and Abuzeid for their wonderful support and ideas for this project. Finally, the Author would like to thank Prof. Yangming Li for his help and advice.

DEDICATION

to the cutest thing in my life

Chapter 1

INTRODUCTION

1.1 Background

Chronic sinusitis is one of the top twenty ambulatory care diagnoses and generates approximately 2.7 million office visits per year in the US alone[1]. It is often treated by endoscopic sinus surgery (ESS) in which an endoscopic camera and medical instruments are used to gain access to the sinus through the nose. Endoscopic endonasal surgery is also an integral component of endoscopic skull base surgery (ESBS), a more complex surgery to treat pathology of the deeper skull base and pituitary gland. [2][3]. Over 250,000 ESS's are performed in the USA annually[4]. Figure 1.1 shows a graphic of ESBS suing endoscopic instruments to access a skull base brain tumor.

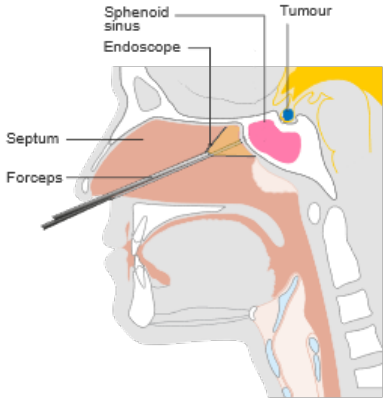

Figure 1.1: Side view of an endonasal surgical approach[5]

These surgeries vary widely in duration, ranging from one hour for simple ESS procedures to over 10 hours for the most complex ESBS. While it is common to use a live real-time video

feed from the endoscope to permit execution of these surgeries, it is uncommon for this feed to be archived and, on the rare occasions that the video is recorded, the result is a lengthy, large video file. Surgical videos can be used for documenting and communicating medical information between medical providers and with patients, training and supervision of novice or aspiring surgeons, assessment of surgical skill, or legal review. For easy use, an ideal record includes a concise, roughly 3 minute video summary that includes all of the most relevant stages of a surgery. It also preserves key examples throughout the temporal history of the surgery. For example, a surgery including tumor dissection and removal should include frames of the tumor being resected. Unfortunately, manual summarization is a tedious task requiring specialized medical knowledge. The intention of this work is to create a pipeline using Machine Learning and computer vision to create an automatic summarization of medical endoscopic surgical videos. More specifically, we initially focus on Endoscopic Skull Base Surgery video (ESBS).

Domain-Specific Knowledge To accomplish this goal, domain-specific surgical knowledge is applied to improve the selection methods over current techniques. This domain-specific knowledge helps to solve three primary problems: (i) classification of surgical frames to eliminate not informative or invalid frames,(ii) identification of surgical instruments to prioritize frames of high impact, and (iii) understanding of video content and surgical flow in order to improve the frame selection. An example of (i) can be seen in figure 1.2, which shows a series of frames that lend no information as to what is happening in the video. These can include frames from outside the patient, blurry frames, or frames in which the tool is obscured by flushing/cleaning of the endoscope lens. Such frames should be removed prior to creating representative summaries.

Based on input from expert surgeons, the vast majority of key medically relevant events within a video occur during instrument use. These events, for example, include dissecting a tumor, and cutting or relocating tissue. Furthermore, important video segments generally utilize a subset of instruments found in the video. For instance,during a key scene of 're-

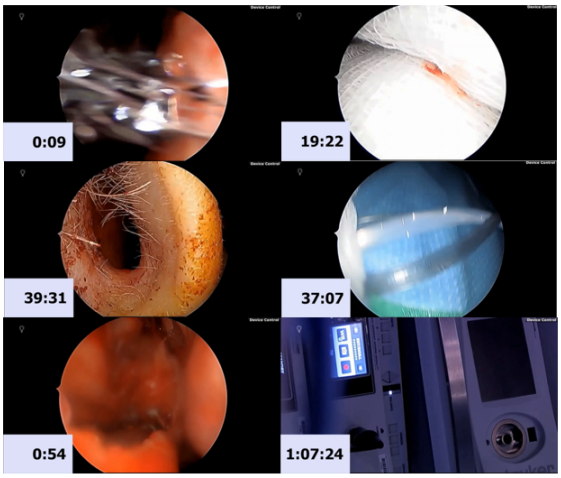

Figure 1.2: Frames from endonasal surgery that are not informative and should not be included in a sugical summary

moving sellar floor', frame sequences with a drill are more representative than with a frame sequence with only the suction instrument. It is therefore useful to identify what instruments are in each video frame (ii).

For domain-specific knowledge of video content(iii), there two primary mechanisms that are exploitable in the creation of summarization. First, a surgical procedure may generally be broken into three stages:

1. Approach

2. Operation

3. Reconstruction

The Approach stage is when the surgeon navigates to the key surgical site. This generally involves navigating through the nasal cavity and removing tissue and bone to provide optimal tool access to the surgical field. The Operation stage involves the primary medical operation such as tumor resection. Finally, during the Reconstruction stage, the conduit created

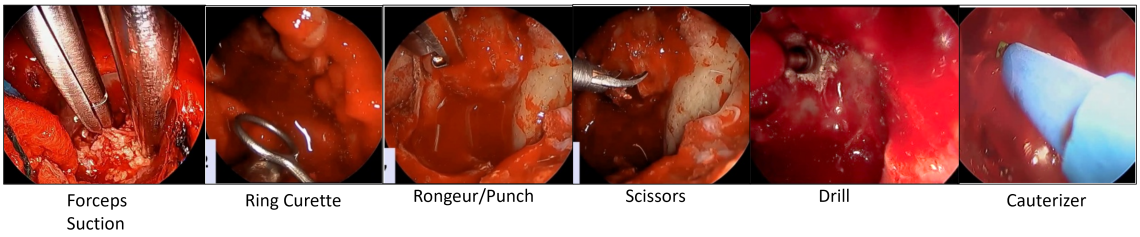

Figure 1.3: Tool Examples. (Left to Right): (Forceps, Suction), Ring Curette, Rongeur/Punch, Scissors, Drill, Cauterizer

between the nasal cavity and the intracranial space is closed using synthetic or patient-derived autologous tissue such as bone or mucosa. Often the reconstruction construct is fixed in position using fibrin glue. The primary tools used are shown in Table 1.1.

Surgical Stage	Tools
Approach	Suction, Forceps, Drill, Rongeur, Punch
Operation	Ring Curette, Forceps,Scissors, Suction
Reconstruction	Scalpel, Forceps, Suction, Cauterizer

Table 1.1: Surgical Tools in use at each stage

Secondly, a surgical video is naturally broken into segments: A surgeon often controls the endoscope to achieve stable, continuous video on a surgical site until the surgeon has completed a given action, at which time the scene changes. It is common for these scene changes to accompany removal and reinsertion of the endoscope entirely as the surgeon changes instruments or shifts view.

Therefore, we propose a pipeline that identifies potentially valid video frames, detects instruments in such frames, separates the video into its constituent stages, finds shot boundaries by identifying high frame movement, identifies best segment within each shot, and

creates a resultant video summary.

1.2 Research Contributions

The specific research contributions are then as follows:

1. A method of automatic summarization of endoscopic videos
2. The application of convolutional neural network frame classifiers based on convolutional neural networks towards more accurate summarization
3. A method of detecting shot boundaries based on convolutional features
4. The creation of a instrument detection dataset in endoscopic videos
5. The analysis of several computer vision object detectors in the application of the created dataset
6. The utilization of instrument presence to identify surgical stage through the application of a Hidden Markov Model
7. The use of surgical stage to improve item 1 summarization accuracy

Chapter 2

LITERATURE REVIEW

Video Summarization is a large field of research with contributions originating from many domains. These domains include works on: (i) video summarization, (ii) frame classification and object detection, and (iii) scene understanding

2.1 Video Summarization

Video Summarization is an algorithmic procedure in which a lengthy video is the input and a video of reduced length and content is the output. Ideal summaries may be used as a stand-in for the original video while minimizing information lost. Potapov et al. [6] sets a video summarization terminology that breaks a single video into shots: video segments that are distinct in nature (such as a shot of someone walking vs. a shot of someone blowing out a candle within the same video) and subshots: representative subsections of the whole shot. The general flow of video summarization is therefore a two-stage approach: 1) Detect and generate boundaries between unique shots and 2) select frames within those boundaries that are most representative (Figure 2.1).

Shot Boundaries are identified in a variety of ways. Mei et al. [7] and Kim et al. [8] propose detecting shots based on overall frame movement as determined by calculating an affine transformation between consecutive video frames. Jin, Tao, and Xu [9] found shot boundaries by creating a hidden Markov model of distinctive scene qualities in soccer videos: blur, zooming, playback, and object detection. Potapov et al. [6] uses a kernalization method to compare dot product distance of frame descriptors, then chooses shot boundaries that minimize the distance for a pre-chosen number of boundaries.

Boiman et. al, creators of the commercial Magisto[10] system, [11] patented a system

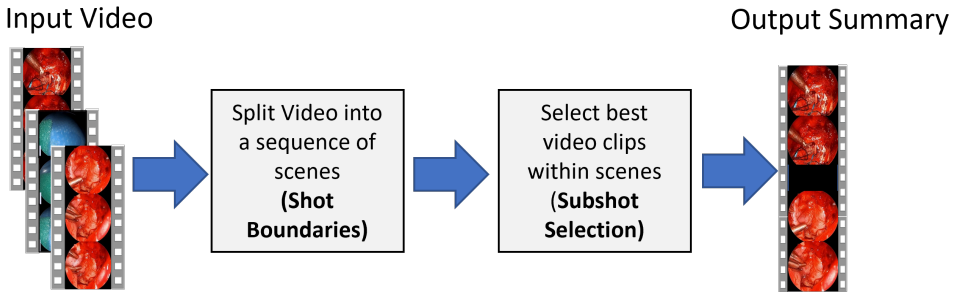

Figure 2.1: High Level concept of Sub-problems video summarization

of proprietary scene descriptors, which was later extended to Scene subshot selection by comparing frames to an external database of descriptors, allowing an importance ranking and frame grouping by reference distance. [12]

Subshot Selection also has a large variety of methods including a method of quantifying the relevance or importance of frames along with a method to select the best result. Potapov et al. [6] trains a Support Vector Machine (SVM) classifier to predict frame importance based on keypoint feature descriptors that have been dimensionally reduced through Singular Value Decomposition (SVD). More recent methods include Convolutional Neural Networks (CNNs) to generate feature descriptions rather than traditional keypoint methods. Otani et al. [13] utilizes a trained CNN to vectorize frames into an n-dimensional semantic feature space and a K-means classifier to select segments closest to the segment mean. Zhu et al. [14] similarly extracts semantic features from a trained CNN, but includes temporal information in a Deep Neural Network combined with skip connections to learn temporally consistent proposals in an end-to-end manner. Zhang, Grauman, and Sha [15] further combines CNN features with an encoder/decoder based Long Short-Term Memory Network (LSTM) structure that chooses a sequence of images with the lowest weighted distance when compared with the overall segment sequence.

2.2 Frame Classification and Object Detection

Although the previous section establishes some procedure options for general video summarization, none utilize domain-specific knowledge of endonasal surgeries. In practice, any general summarization method would rapidly encounter difficulties. (See Magisto comparison in Results section) This is because that there are large sections of video that are not necessary (such as footage from outside the patient or video of a surgeon navigating to the surgical site) (Figure 1.2). Furthermore, none utilize the instrument knowledge to more effectively find the best sequence of frames.

Frame Classification is a well-studied problem in computer vision literature. Popular models powered by Convolutional Neural Networks (CNNs) such as Resnet[16] and Efficientnet[17] have dominated recent work. These networks allow for effective and robust feature extraction that is easily adapted for classification when combined with a softmax output. Within the domain, Adidharma et al. [18] utilizes K-means clustering to separate endonasal video frames and identify noninformative frames based on color and blurriness of features.

General Object Detection is also a well explored space in computer vision, utilizing large image sets like the popular COCO[19] dataset. Yolo-V3[20] and Yolo-V4 [21] use a fully convolutional neural network bottle-necking into 1x1 convolutions which are then upscaled to a variety of scales, simultaneously feeding into convolutional heads for objectness, box center position, box size, and classification. Faster-RCNN [22] uses a convolution neural network to compare a series of box-shaped 'anchors' at different scales and perform classification of each anchor box. Centernet [23] uses a convolutional network to regress a pixel-wise keypoint prediction for each class, where the keypoint is the center of the bounding box. The box size is then regressed from the bounding box center. These just represent a small number of top-performing proposed Neural-Network based architectures, which have been applied to a large variety of tasks and datasets.

Medical Object Detection is a more specific application of object detection in which instruments or other items are identified. Lee et al. [24] compares several detector frameworks

to identify and detect medical tools sitting on a bench. Liu et al. [25] compares several architectures for the purposes of bounding box detection on the EndoVis dataset [26], and also modifies the Centernet architecture to achieve slightly improved performance. Shi et al. [27] similarly measures a variety of architecture performances on the Chloec80 dataset and proposes an attention-based architecture for improved performance. Although the surgical tool detection field has been well studied, previously used datasets typically contain visually distinct objects in a relatively clear environment. Practical endonasal surgical tool video scenes are much visually challenging, and are much less studied. To our knowledge, there is no publication regarding tool identification and detection in the endonasal surgical setting.

Endonasal Instrument Identification is an unstudied problem. However, tool segmentation in an endonasal environment is a well considered problem. Qin et al. [28] Considers instrument segmentation in an endonasal surgical environment, but only considers single-class binary 'tool' or 'no-tool' instrument localization. Lin et al. [29] extends this work by considering consecutive frames as input, but keeps binary representation. Qin et al. [30] also seeks to use spacio-temporal information in the analysis of binary tool segmentation in an endonasal environment. Unfortunately, all are unsuited to tool identification due to lack of multiclass labels.

Annotation for segmentation is known to be extremely expensive to the point where a variety of augmented learning frameworks have been proposed to reduce the annotation expense [31]. Bounding box annotations are easier to generate and are thus more economical.

2.3 Scene Understanding

Surgical Stage Estimation is the identification of the general step taking place in a section of surgical video. This may include precise surgical action such as 'Removing Tumor' or 'Controlling Bleed', or may include more general actions such as 'Reconstruction'. This topic has a large amount of academic literature that attempts to define the optimal way to identify the surgical stage. Garrow et al. [32] summarizes the current state of literature, noting that most methods that attempt to understand surgical stage utilize Hidden Markov Models

(HMMs), Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs) or Recurrant Neural Networks (RNNs) to achieve these goals. Nearly all attempts on surgical stage estimation are in regards to laparoscopic cholecystectomy (LC), and do not extend their methodologies to other types of surgery such as endonasal surgery. LC is beneficial because of relatively clean images and well-defined phases. One top performer in LC application utilized a LSTM tied to the deep convolutional output of a CNN to directly classify surgical stage[33]. This work was later expanded to directly add a multi-task function to simultaneously learn surgical tool detections along with the tool stage[34]. On the same dataset, Cadène et al. [35] uses a CNN to extract frame-by-frame features, then uses a temporal smoothing approach using Hidden Markov Models to improve detection performance.

Hidden Markov Models In addition to aiding cameras in surgical stage estimation, Hidden Markov Models have been found useful for action recognition in a variety of tasks. DiPietro et al. [36] uses a HMM trained with sensor input to estimate surgical phase. Hannaford and Lee [37] uses HMM models for task prediction and analysis for a telerobotic operator. Overall, Hidden Markov Models are extremely robust mathematical models that help encode human domain knowledge about state probabilities in action recognition.

Chapter 3

TECHNICAL BACKGROUND

3.1 Convolutional Neural Networks

3.1.1 Neural Networks

Neural Networks and their more advanced counterparts are the backbone of modern vision processing. A neural network consists of a vector input 'X', some expected vector output 'Y', and some number of 'hidden' states Z. each distinct value state is colloquially known as a 'neuron'. This can be seen visually seen in figure 3.1. The value assigned to a non-input

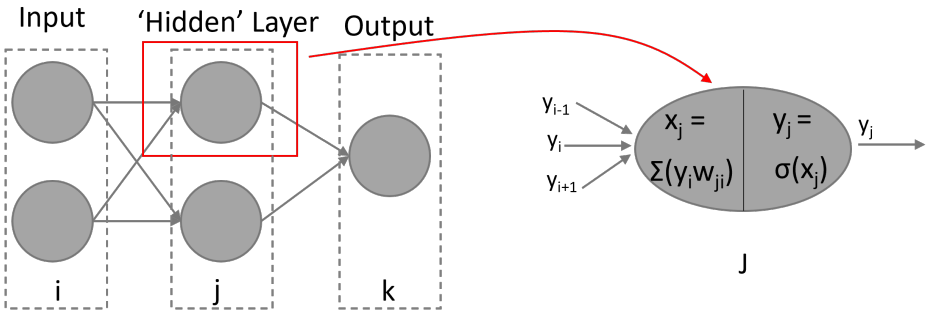

Figure 3.1: Simple model of a linear neural network (Left) a set of 'neurons' organized into neural network. (Right) Single Neuron

neuron is then given in equation 3.1, also known as the forward step.

$$y_j = \sigma(\sum_i W_{ji} y_i + b_j) = \sigma(k) \tag{3.1}$$

where y_j is the output of the j'th neuron, w_{ji} represents the linear weight from neuron i to neuron j, σ is an activation function such as ReLu or sigmoid.

For a given loss function $\ell(Y) : \Re^n \Rightarrow \Re$, where Y is the vector concatenation of all output neurons in a given layer, the neural network can be updated according to gradient descent. A single Neuron weight update is shown below

$$w_{ji} = w_{ji} - a(\frac{\partial(\ell)}{\partial w_{ji}}) \tag{3.2}$$

where a is the learning rate and $(\frac{\partial(\ell)}{\partial W})$ may be calculated by finding:

$$\frac{\partial(\ell)}{\partial(w_{ji})} = \frac{\partial(\ell)}{\partial(x_j)}\frac{\partial(x_j)}{\partial(w_{ji})} = \frac{\partial(\ell)}{\partial(x_j)}y_i \tag{3.3}$$

$\frac{\partial(\ell)}{\partial(x_j)}$ is known from the following calculation, shown in further detail in [38].

$$\frac{\partial(\ell)}{\partial x_j} = \frac{\partial(\ell)}{\partial(y_j)}\frac{\partial(y_j)}{\partial(x_j)} \tag{3.4}$$

This function has slightly different form of each activation function. For the simply differentiable and common sigmoid[38],

$$\frac{\partial(\ell)}{\partial x_j} = \frac{\partial(\ell)}{\partial(y_j)}y_j(1 - y_j) \tag{3.5}$$

Where $\frac{\partial(\ell)}{\partial(y_j)}$ is known by delta approximation of the output function compared to the actual result. This procedure can be generalized to any number of layers and any size of hidden layers. Analogous calculations can be used to determine the update parameter for b.

Most commonly, and in the context of this thesis, neural networks are trained in a supervised manner. That is, loss ℓ is directly calculated from predicting an output from a given input, and comparing to a known exact reference.

3.1.2 Convolutionional Neural Networks

While neural networks form the backbone for theory and implementation of many modern machine learning tasks, modern machine learning frameworks for vision processing rely on convolutional operations rather than linear connections between layers as shown in the previous section. Consider a 2-dimensional image I with dimensionality [nxm]. The convolutional

output matrix Y (forward equation) is given by equation

$$Y[m,n] = (I * h)[m,n] = \sum_j \sum_k h[j,k] I[m-j, n-k] \tag{3.6}$$

where j,k are the dimensions of a kernel h. This may be visually shown by the figure below. 3.2.

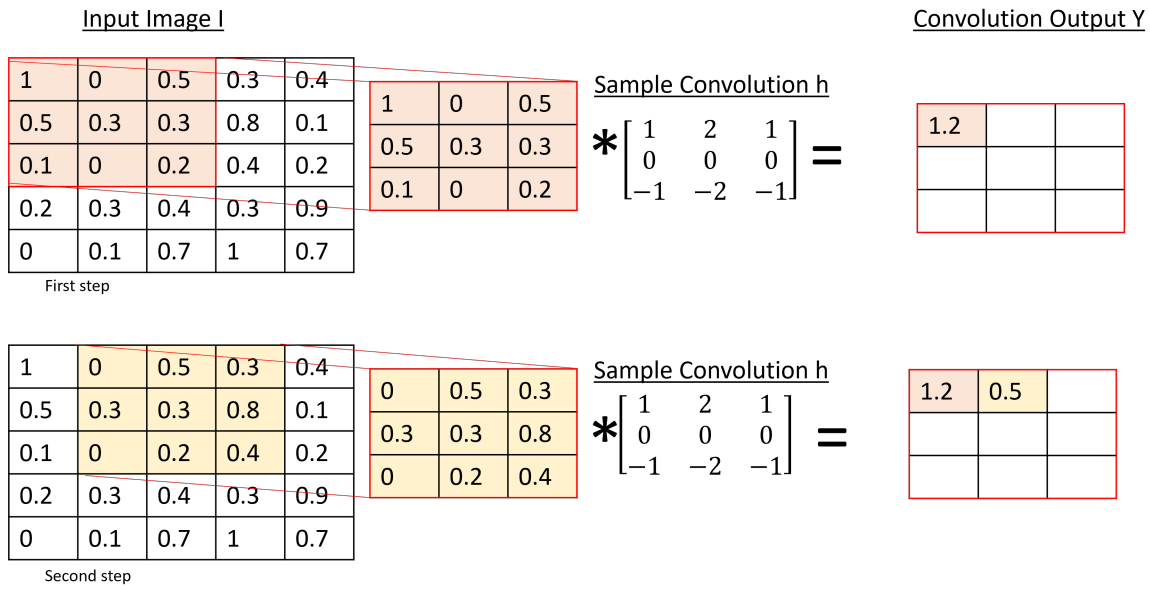

Figure 3.2: Simple convolution example. (Left) Input Image has a sample element-wise convolution applied to create an example convolutional output

In this instance, the weights may once again be calculated by back-propagation where the goal is once again to calculate $\frac{\partial(\ell)}{\partial I_{jk}}$ and $\frac{\partial(\ell)}{\partial h_{mn}}$:

$$\frac{\partial(\ell)}{\partial I_{jk}} = \frac{\partial(\ell)}{\partial(Y_{mn})} \frac{\partial(Y_{mn})}{\partial(I_{jk})} \tag{3.7}$$

where $\frac{\partial(Y_{mn})}{\partial(I_{jk})}$ is the summation of the filter weights of h_{jk} that impact the output Y_{mn}. Finally,

$$\frac{\partial(\ell)}{\partial h_{jk}} = \frac{\partial(\ell)}{\partial Y_{mn}} \frac{\partial(Y_{mn})}{\partial h_{jk}} \tag{3.8}$$

where $\frac{\partial(Y_{mn})}{\partial(h_{jk})}$ is the linear combination of every term of the input image I impacted by the filter element h_{jk}

3.1.3 Resnet

There are many neural networks architectures in use today, to the point where it is impossible to mention them all. For the purposes of this paper, two backbones of this paper will be mentioned: Resnet[16] and Efficientnet [39].

Resnet [16] uses a series of cascaded convolution blocks to learn image features at various levels. This utilizes a series of 3x3 convolutions at a given image size. Then, a 3x3 convolutional block with stride = 2 is used to reduce the image size in half for the next series of convolutions. This allows the convolutions to adapt to features of different scales. In addition, Resnet adds 'residual' connections, which is simply an identity tie the input of a set of convolutional blocks to the output as follows:

$$y = F(x) + x \tag{3.9}$$

where y is the output of a set of convolutional blocks, x is the input, and $F(x)$ is the feature

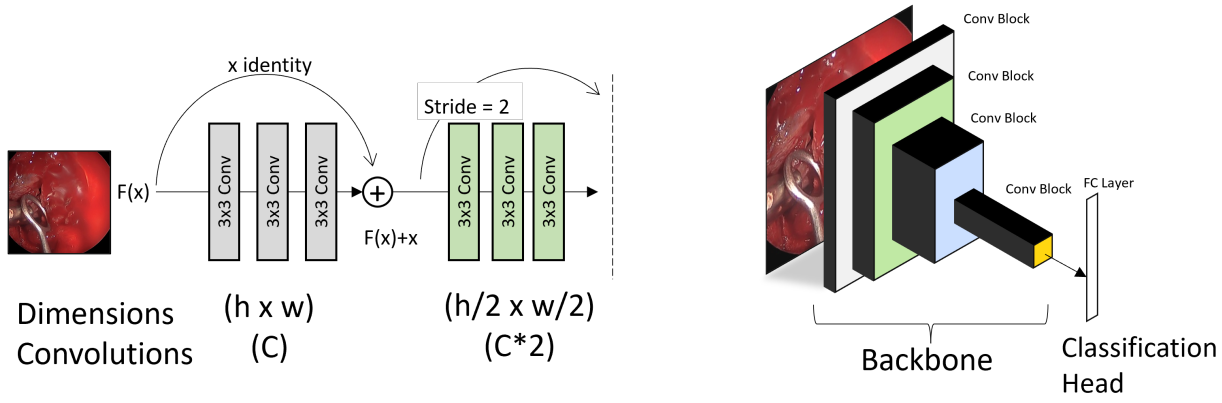

Figure 3.3: Resnet Architecture

updates made by the convolutional set. This helps to avoid a common issue in Convolutional

neural networks: vanishing gradient. By incorporating deeper and deeper Convolutional neural networks that include more and more layers, it is easy to imagine that the gradient or contribution of loss for a great many model parameters approach 0. Especially given the constraints of floating-point arithmetic, this causes numeric instability. By including a 'side path' in which the output of a set of convolutions is added to its identity input, numerical instabilities are reduced, as any locally tiny gradients would simply cause the layer to adopt its identity. See Figure 3.3 for a graphical representation of the residual connections and backbone. 'Backbone' is used to relate the unique convolutional feature architecture. Such a CNN can be generalized to many tasks: Resnet has been used for image classification by applying a simple fully connected layer to the output of the Resnet, has been used for object detection as we will see in a later section, or image segmentation.

3.1.4 Efficientnet

Efficientnet[39] is the Second CNN used in the final implementation of this paper. In essence, this is extremely similar to the Resnet architecture, with only a couple structural differences. First, the convolutional 'blocks' are structured in an inverted way according to [40]. This essentially increases the number of channels in a block by adding a large number of inexpensive 1x1 convolutions before the 3x3 convolutions. Moreover, Efficientnet meticulously analyzes the optimal depth (number of sequential layers in the CNN) vs the optimal width (size of each layer). Both depth and width increase the complexity and improve the theoretical performance, but both increase number of parameters as well. Efficientnet finds the empirically best balance between these factors.

3.2 Faster RCNN Object Detection

As will be discussed further in this paper, the faster-RCNN object detection framework[22] will be the primary methodology used to detect objects in an image. Faster-RCNN is a two-stage object identification and classification pipeline. (See Figure 3.4 In the first stage, an input image is run through a fully convolutional neural network backbone such as Resnet.

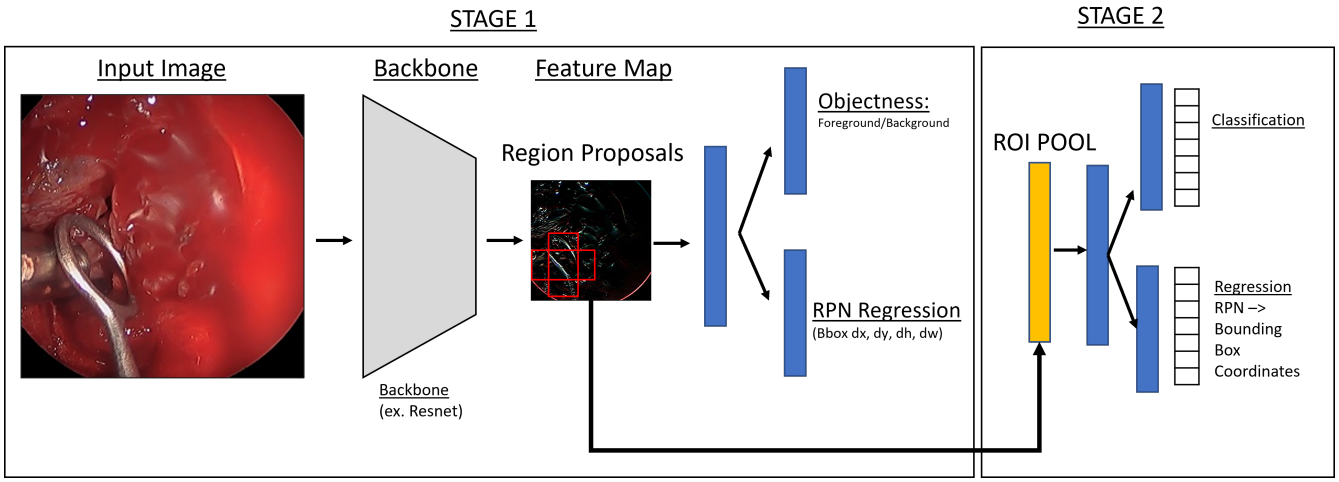

Figure 3.4: Faster R-CNN Architecture

The output is a reduced-dimensionality feature map similar in shape to the original image but smaller in height and width. 9 'anchors' are then used for each corresponding 'pixel' of the output feature map of various sizes. For example, a 3x1 rectangular box, 1x3 rectangular box, and 2x2 square box centered on a target pixel are used to evaluate different shape and size regions of interest. To account for scale, this set is repeated for 3 different size region boxes. See Figure 3.5 for representation.

Once regions are proposed, they are evaluated for Objectness and Regression through the following formula (Equation 3.10:)

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{3.10}$$

where L_{cls} is the classification loss,L_{reg} is the classification loss, i is the index of an anchor, p_i is the predicted probability of anchor i being an object, p_i^* is the ground truth label (1 if anchor is positive, 0 if anchor is negative), $t_i^*$is the ground-truth box associated with a positive anchor, and λ is a balancing parameter.

Loss associated with the second stage regressor and classifier has the same form of the equation (Eq. 3.10. Training may be done in an alternating fashion: training the RPN stage,

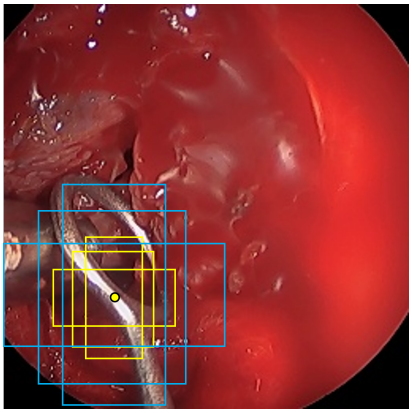

Figure 3.5: Faster R-CNN Anchors. During Training, 3 different anchor boxes are evaluated for object presence (yellow) This is repeated for different scale boxes (blue) Actual anchors are computed on feature representation, not on input image as is shown

followed by the discriminator stage, and repeat. In our implementation, these are trained concurrently.

3.3 Hidden Markov Models

L. Rabiner[41] does a very good job explaining the background and math of the Hidden Markov Model. A hidden markov model (HMM) is a statistical Markov Model characterized by a state S which is the 'true' representation of a statistical object, but this state is not directly observable. Instead, a set of observations tied to the states is able to be seen. Overall, a hidden markov model requires 5 things:

1. A set of 'true' states S = $\{S_1, S_2, ...S_n\}$

2. A set of distinct possible observations V = $\{v_1, v_2, ...v_n\}$

3. State Transition possibility A = $\{a_{ij}\}$ where $a_{ij} = P[q_{t+1} = S_j | q_t = S_i$, where q_t is the true state. i.e. the possibility that a state transitions from one state to another.

4. Observation possibility distribution j, B = $\{B_j(k)\}$ where $B_j(k) = P[v_k|q_t = S_j]$, or the probability that an observation occurs in a given state.

5. the initial state distribution $\pi = \{pi_i\}$

For a visual representation, please see Figure 3.6

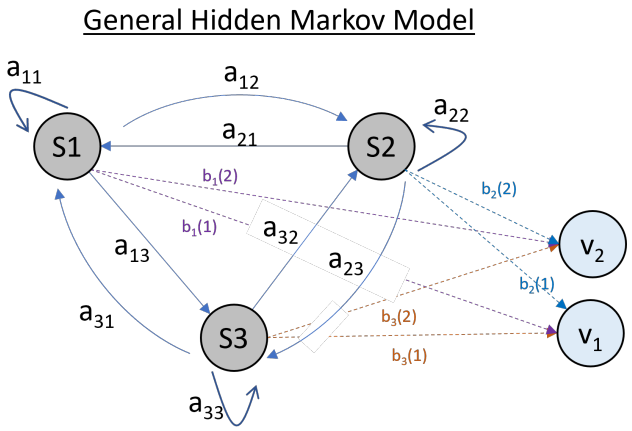

Figure 3.6: Simple example of a fully connected Hidden Markov Model (HMM) with three states and two possible observations

Based on the 5 criteria given above, we will also use $\lambda = (A, B, \pi)$ to represent the whole parameters of a HMM. Then, there are 3 basic problems for HMMS as given by L. Rabiner[41]:

1. Given an Observation sequence and model, how to determine the probability of that observation sequence given the model

2. Given an Observation sequence and model, how to uncover the 'optimal' hidden states. For our application, this roughly to finding the most likely set of hidden states that explains the sequence of observations.

3. How to adjust the parameters λ to best maximize $P(O|\lambda)$ where O is a sequence of observations

of these three, only 2 and 3 are important problems in the context of this application. However, the mathematics behind 1 are crucial to the derivations of 2 and 3.

For 1. Determining $P(O|\lambda)$, this may be simply conceived by saying:

$$P(O|\lambda) = \sum_{allQ} P(O|Q,\lambda)P(Q|\lambda) \tag{3.11}$$

However, making this calculation is computationally infeasible. We therefore use the forward algorithm: (letting $\alpha_t(i) = P(O_1O_2...O_t, qt = S_i|\lambda)$ or the probability of the partial observation sequence until time t and state S_i at time t):

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i)a_{ij}]b_j(O_{t+1}) \tag{3.12}$$

that is, the probability of being in the given state is the summation of the sum of the probability that the model is in each previous state at the previous time step multiplied by the transition possibility. Then, at time T, $P(O|\lambda) = \sum_i \alpha_T(i)$

Analogously, the backwards algorithm is calculated to find the probability that, from a given time t, the following states will adhere to a certain sequence of states:

$$\beta_t(i) = P(O_{t+1}O_{t+2}...O_{\tau)|q_t=S_i,\lambda} \tag{3.13}$$

$$\beta_t(i) = [\sum_{j=1}^{N} a_{ij}b_j(O_{t+1})\beta_{t+1}(j) \tag{3.14}$$

3.3.1 Viterbi Algorithm

With this background in place, we are ready to understand the solution to problem 2: Determining the most likely state sequence for a given observation sequence. This is done via the Viterbi Algorithm[41]: let

$$\delta_t(i) = max_{q_1,q_2,...}P[q_1q_2...q_t = i, O_1O_2...O_t|\lambda] \tag{3.15}$$

ie $\delta_t(i)$ is the highest probability along a single path that ends in state S_i. Then, to find iteratively:

$$\delta_{t+1}(j) = [max_i \delta_t(i) a_{ij}] b_j(O_{t+1}) \tag{3.16}$$

and

$$\phi_t(j) = argmax_i[\delta_t(i) a_{ij})] \tag{3.17}$$

where ϕ is keeping track of the argument that maximizes the iterative equation.

Finally, after the forward run is complete, the optimal path is given by:

$$q_t^* = \phi_{t+1}(q_{t+1}^*) \tag{3.18}$$

3.3.2 Baum-Welsh Algorithm

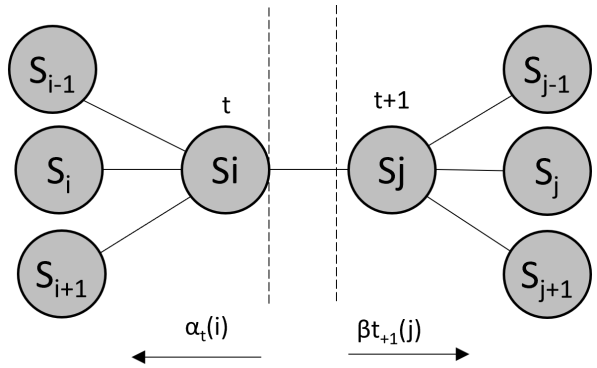

Figure 3.7: Visual representation of HMM forward and backward parameters. α represents the probability that a model will be in state $S_i$at time t. β represents probability that following states may be seen from state S_j

The third problem: (how to update the parameters of a HMM based on available data) is done as follows: define the term $\xi_t(i,j)$ as the probability of being in state S_i at time t and S_j at time t+1:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} \tag{3.19}$$

and define $\gamma_t(i)$ as the probability of being in state S_i at time t:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{3.20}$$

Baum-Welsch helps define the parameter update for a_{ij} and $b_j(k)$:

$$\bar{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.21}$$

$$\bar{b_j}(k) = \frac{\sum_{t=1 \; s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.22}$$

For our formulation in the following Methods section, we utilize a dedicated 'start' state, so calculations regarding initial probability are trivial.

Chapter 4

DATASET

A new surgical instrument endoscopic video dataset was created with bounding box descriptions and class labels to apply modern detection algorithms in the detection of surgical instruments and eventual summarization. Bounding box descriptions outline each surgical instrument in pascal-voc format [x_min, y_min, x_max, y_max], and class labels state the surgical instrument of each bounding box. This data expands the UW Sinus Surgery Dataset

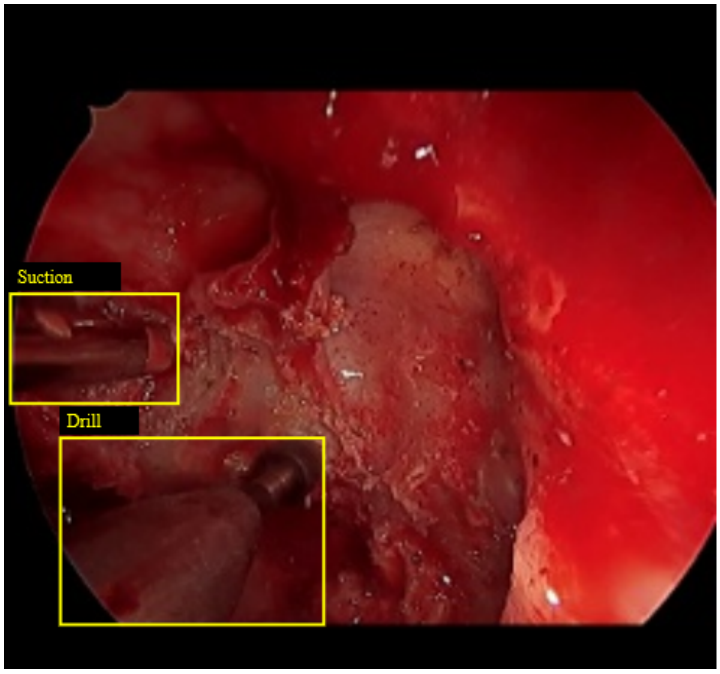

Figure 4.1: Example of a Bounding box annotation

[42], which provides segmentation labels for several cadaver and live videos, taken at 1hz increments. Then, representative samples were pulled from an additional 7 videos biasing the

selection towards lower frequency instrument objects, bringing the total number of images to 7749 images. Separately, additional representative frames were taken from 3 additional, independent videos to form a test set of 1743 images. Bounding box annotations were manually created for each image (Figure 4.1). Category labels were chosen from {No Instrument, Suction, Rongeur/Punch, Drill, Forceps, Misc, Ring Curette, Cauterizer, Scissors}. These represent the most common instruments in endonasal surgery.

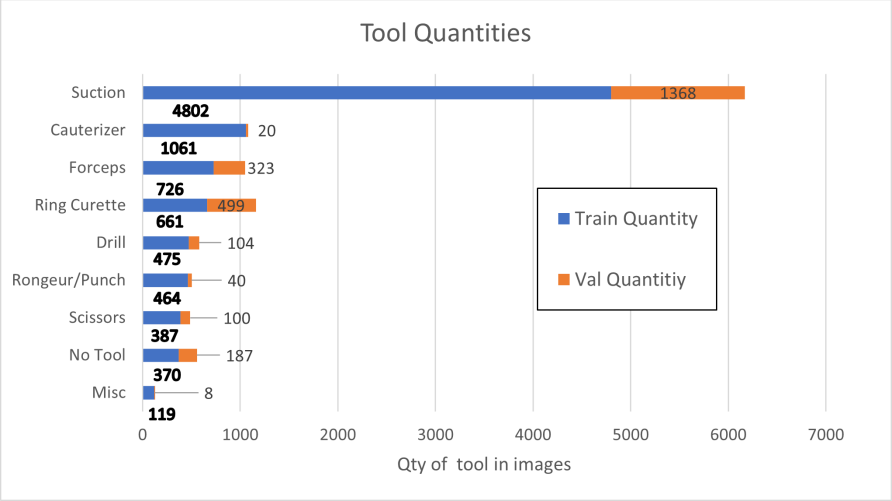

Figure 4.2: Number of images contained within a dataset

Video quality was challengine as the dataset contained many occlusions, specular reflections, and similar geometry between objects. For example, the frame shown in Figure 4.3 shows 2 instruments with their ends occluded by skull geometry, making their identities unclear. It is therefore expected that individual frame reference mean Average Precision (mAP) values will be low.

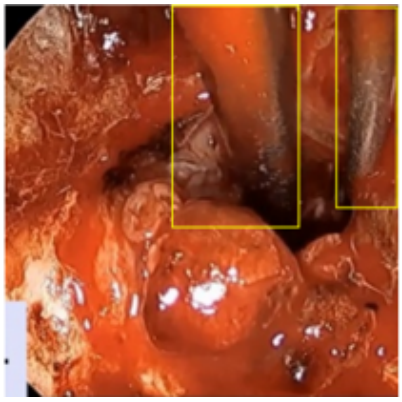

Figure 4.3: Example of a Difficult Bounding box annotation. An unfamiliar user is unable to tell what tool is accompanying the suction tool (Left box) in this frame

Chapter 5

METHODS

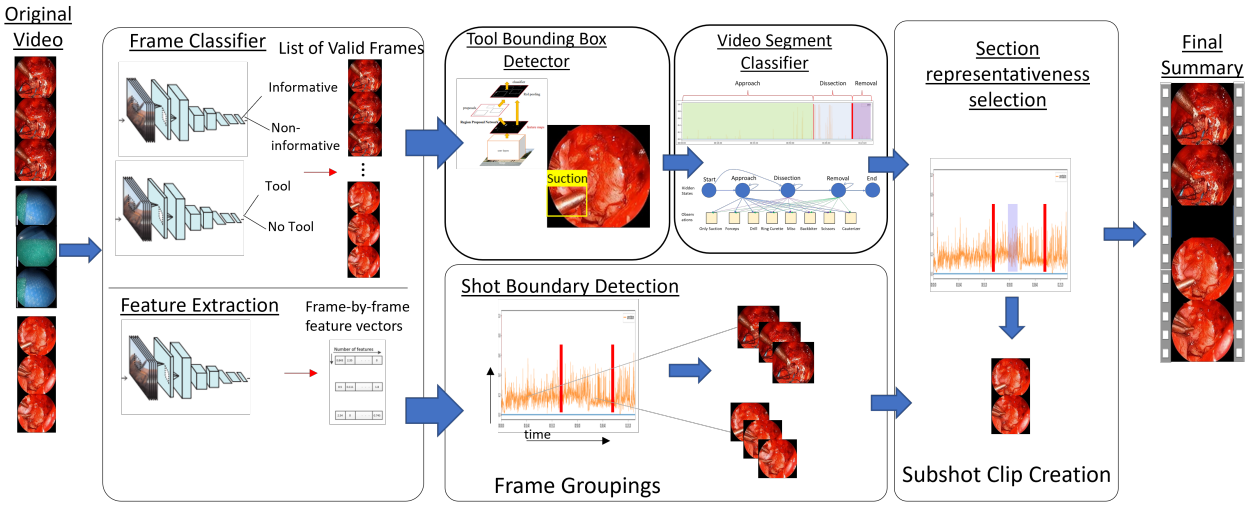

Figure 5.1: Block Diagram of overall process. (Left) An original video is fed into a computer application. A series of classifiers identify if a scene is surgically informative by identifying if a tool is in a scene and if the frame is inside a patient. (Middle) Frames are fed into a tool detector to determine what section of surgery the video time is at. Meanwhile, a feature extractor itentifies major scene jumps that correspond to scene changes. (Right)Finally, a subshot is created from each scene that is determined to be medically relevant.

Similar to the reference literature, the Video Summarization task is broken up into several sub-problems:

1. Irrelevant Frame Removal

2. Frame-by-Frame surgical instrument detection

3. Scene understanding and Surgical Stage Estimation

4. Shot Boundary Detection

5. Video Frame Subshot Selection

6. Combination into Final Summary

The outline of this process is shown in Figure 5.1.

5.1 Irrelevant Frame Removal

To reduce video size into relevant parts, first blurry and non-operative frames (frames outside the patient) were removed through a trained CNN. This is an extension of the work described in Adidharma2021, in which frames were separated by k-means clustering. These separations were then used to train an Efficientnet[39] CNN to discriminate against invalid frames. To preserve a reasonable frame fidelity while balancing calculation time, each video was sampled at 5Hz. A second CNN classifier was used to eliminate frames with no instruments present.

5.2 Tool Detection

After individual frames are approved by the irrelevant frame removal stage, the instruments are identified by the next stage of the detection model. In this stage, we evaluated Faster-RCNN[22], YOLO-V4[43], and Centernet [23]. All models utilized pretrained model weights from the COCO dataset except for the final classification output layers. These models were then fine-tuned on the dataset generated in section 4.

5.3 Surgical Stage Estimation

Surgical stage estimation is a crucial aspect towards generating meaningful and informed summaries. As previously discussed, surgical stages are most simply broken down into an Approach, Operation, and Reconstruction. Instruments are useful in identifying the surgical

stage, but this is a non-deterministic relationship. For example, drills are most commonly used in the approach stage while the surgeon is navigating to the surgical site to provide instrument access. However, in some sample videos, the surgeon will later enlarge the access feature by applying the drill in the middle of dissection. It is therefore helpful to model the stages as hidden probabilistic states with sequential ordering. The result is a framework that can easily be turned into a Hidden Markov Model (HMM)[41]. The available observations of our HMM are the instrument detections: Forceps, Drill, Ring Curette, Misc, Rongeur, Scissors, Cauterizer, and Only Suction. The resultant state model is shown in Figure 5.2. Any frame that did not have a instrument present was ignored.

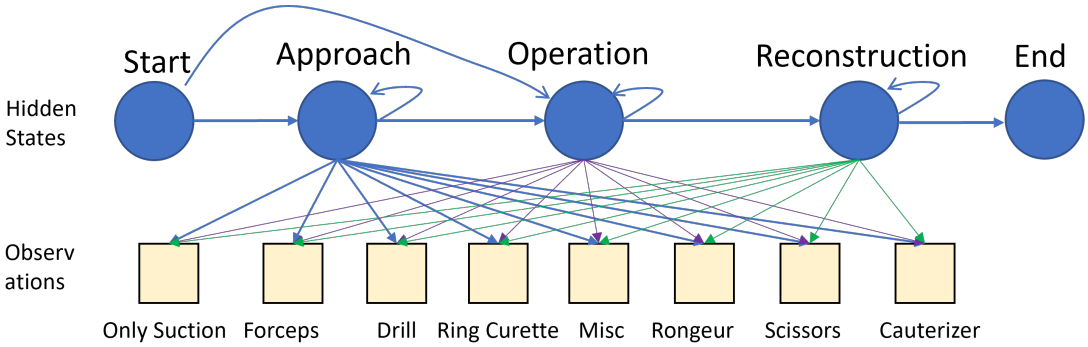

Figure 5.2: Hidden Markov Representation of surgical states. "Hidden" states are: Approach, Operation, and Reconstruction. Possible Observations are: Only Suction, Forceps, Drill, Ring Curette, Misc, Rongeur, Scissors, and Cauterizer

Tuning the HMM The Baum–Welch algorithm[41] is applied to tune the HMM. First, we apply a prior set of information (initial HMM parameters) that was manually derived from low-level knowledge of the state dynamics. A HMM is based on a set of 'Emission' probabilities describing the probability of seeing an observation in a given state: $P(Y_n|X_n)$, where Y_n is the observation and X_n is the hidden state. Additionally, an HMM is dependent on the 'Transmission' probabilities $P(X^+ = X_n|X_i)$ where X_i is the current state, X^+ is the

next state, and X_n is another state. The prior emission and transmission probabilities are initially estimated in tableA.1 and A.2 which may be found in the appendix. The result is then updated based on the first 10 videos which are fit using the Baum–Welch algorithm.

Evaluating states using the HMM is done by the Viterbi algorithm [41], which creates a state estimation by maximizing the probability over all possible sequences by using the pre-calculated transition and emission probabilities.

$$X^*_{0:T} = argmax_{X_{0:T}} P[X_{0:T}|Y_{0:T}] \tag{5.1}$$

where $X^*_{0:T}$ is the optimal estimation, $Y_{0:T}$ is the observation and $X_{0:T}$ is the hidden state.

5.4 Video Frame Shot Boundary Detection

Feature Extraction In endonasal surgical cases, there is a natural boundary between surgical scenes caused by the rapid movement of the endoscope. Scenes of extended, similar actions are generally followed by a removal of the endoscope in order for the Surgeon to perform intermediate tasks such as changing or cleaning instruments, resting, or discussion of next steps with other members of the surgical team. Furthermore, rapid endoscope movement also corresponds to change in focus or scene. For example, a surgeon may linger on a single location while removing material, then rapidly change the endoscope position to the next location once the way is clear. Therefore, it is useful to identify segments containing rapid endoscope movement and as natural video shot boundaries. As we did with the computation of invalid frame classification, we used an Efficientnet CNN to extract the feature vector of the CNN, creating 256-dimensional output. Efficientnet CNN was pretrained from the Imagenet dataset with no additional modifications necessary to the CNN weights or parameters.

Boundary Detection After the 256-dimensional feature vector was created for each frame as described as above, these features were then subject to a simple L2 distance measurement along with a smoothing factor:

$$d\left(I_k, I_{k-1}, m\right) = \frac{\sum_{n=k-m}^{n=k+m} \sqrt{\sum_{i=1}^{256} \left(I_{n,i} - I_{n-1,i}\right)^2}}{2m} \tag{5.2}$$

where d is the distance between consecutive frames, I is the feature vector at frame k and m is a smoothing factor in number of frames. Time stamps with high mean frame distances then correspond to high scene motion.

Finally, frames with highest weighted distance are chosen in decreasing order until a user pre-defined limit is reached. For example, to generate 3-minute videos, b = 45 frame boundaries were chosen.

The number of boundaries were then split according to the number of relevant frames, with the distribution favoring the Operation and Reconstruction stages. For example, if a video shows {40%,40%,20%} of valid frames in the approach,operation, and reconstruction stages, respectively, then the number of boundary weighting is chosen to be {30%,40%,30%}. The mathematical breakdown is shown in equations 5.3 through 5.8

$$n_{approach} = n(\{I_1 ... I_o | I_t \in Valid\}) \tag{5.3}$$

$$n_{operation} = n(\{I_o ... I_r | I_t \in Valid\}) \tag{5.4}$$

$$n_{reconstruction} = n(\{I_r ... I_{max} | I_t \in Valid\}) \tag{5.5}$$

$$\%_{operation} = \frac{n_{operation}}{N} \tag{5.6}$$

$$\%_{reconstruction} = 1.5\frac{n_{reconstruction}}{N} \tag{5.7}$$

$$\%_{approach} = \frac{n_{approach}}{N} - \%_{reconstruction} \tag{5.8}$$

where I_o is the threshold between approach and operation, I_r is the threshold between operation and reconstruction, and N is the total number of valid frames The approach can be lengthy and have fewer medically important actions, and is therefore under-sampled as a

result. The Reconstruction, meanwhile, has quick actions in comparison and is over-sampled. Therefore, the reconstruction is over-sampled by 50% compared to the detected ratio of valid frames detected. This number of boundaries was then subtracted from the Approach stage of the video. For videos with no approach such as those manually pre-cut to remove the lengthy approach, the boundaries are instead removed from the operation stage of the video. The instruments considered as valid were stage-specific and are shown in Table 5.1.

Surgical Stage	Tools
Approach	All except {Only Suction, Cauterizer}
Operation	All except {Only Suction, Cauterizer}
Reconstruction	All except {Cauterizer}

Table 5.1: Stage-specific Surgical Tools allowed when generating clips and deciding number of boundaries per surgical stage

To prevent scenes of high movement from being over-represented, a frame boundary was not chosen if it fell within a certain time of another boundary: a 'lockout time'. This prevents the same scene boundary from being sampled multiple times. For example, if the video is transitioning for 20 time stamps with very rapid movement, the lockout time prevents the same endoscope movement from being chosen as a boundary multiple times.

5.5 Video Frame Subshot Selection

For our implementation of video summarization, we chose to use subshot representativeness to dictate the best subshot.

$$R(k) = \sum_{n=k}^{n=k+clip\_length} \frac{\sum_{i=1}^{N} d(I_n, I_{n+i}, 1)}{N} \tag{5.9}$$

where R(k) is the "representativeness" of the clip starting at frame k, N is the total number of frames in the scene, and I is the feature vector at frame i. In essence, this is a measure of

the average L2 distance of a set of N frames to every other frame within a shot. Since low L2 distance indicates that two frames are similar, the the lowest "representativeness" score indicates a clip that is most similar to the entire shot. Therefore, the best clip was chosen as the clip that had the best "representativeness" for each shot.

$$\text{Best Clip} = argmin_k(R(k)) \tag{5.10}$$

Only frames with stage-specific instrument detections were used when calculating the representativeness according to Table 5.1.

Chapter 6

RESULTS

This section states the results and compares our final developed algorithm results with commercial video summarization product (Magisto) [10].

6.1 Obscure Frame Removal

Blurry and External to the Patient frames were removed using the method of Adidharma et al. [18] . See Adidharma et al. [18] for performance.

Frames with no tools were identified using the Convolutional Neural Network of Section 5.1 to identify frames containing tools. The performance is shown in figure 6.1. Overall accuracy was 88%.

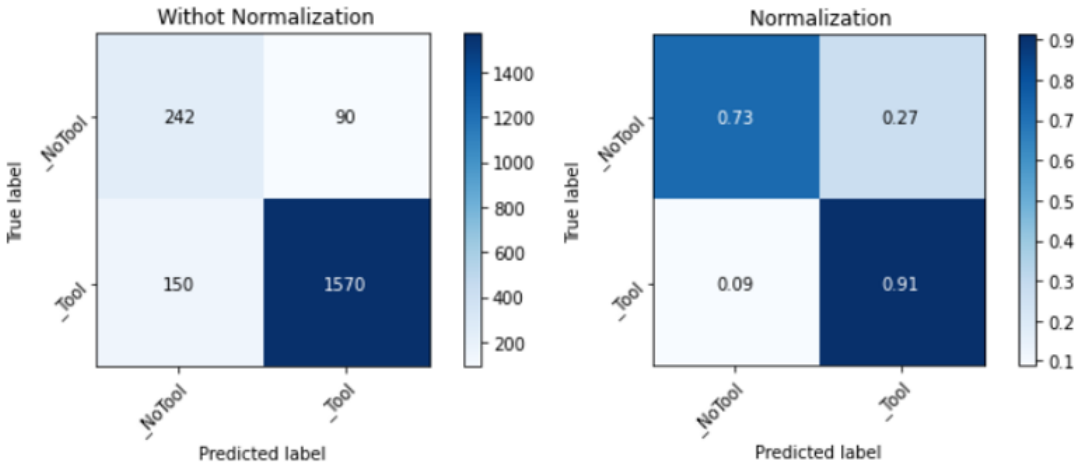

Figure 6.1: Confusion Matrix showing accuracy of different classes of video

Some misidentified frames are shown in Figure 6.2. With the characteristics of our video,

even a human may have trouble detecting a tool in individual frames, and misidentified images are often part of lower quality video segments in which the tool is occluded, difficult to identify, or subject to poor lighting conditions. Such scenes would be poor selections for automatic summaries.

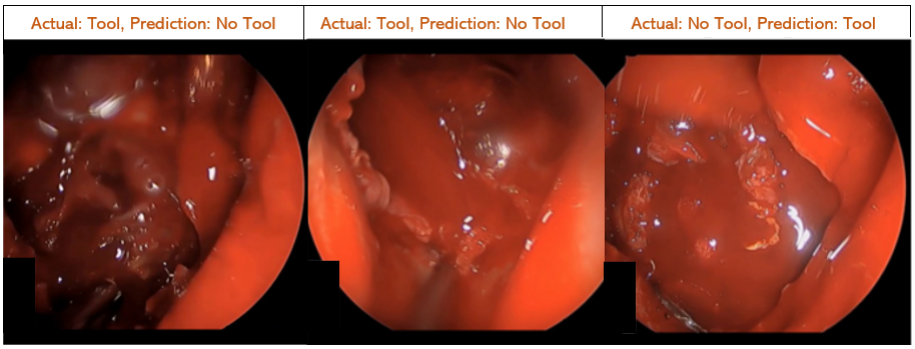

Figure 6.2: MisClassified Pictures

6.2 Tool Identification

Tool Identification performance for the primary surgical tools is shown in Table 6.1. The Faster RCNN structure shows the best results on the dataset. Therefore, FasterRCNN was used in the subsequent stages of the summarizing pipeline.

6.3 Surgical Stage Estimation

The HMM model of Section 5.3, shown in Figure 5.2 was trained using the Viterbi Algorithm on 15 surgical videos of a variety of lengths and stage durations. The shortest video was 54 minutes, and the longest video was 272 minutes. Results were obtained by combining all runs into the python pomegranate package [44] and an HMM was fit without ground-truth labeling. One video result is shown in figure 6.3, which shows the tools detected in all video frames and the corresponding surgical stage. The results for all 15 videos are shown in Figure A.3 in the Appendix, and is represented visually in Figure 6.4. The Hidden Markov Model

	Average Precision (AP)		
Tool	Faster RCNN	YoloV4	Centernet
Rongeur/Punch	0.48	0.44	0.229
Drill	0.63	0.267	0.641
Forceps	0.41	0.276	0.272
Ring Curette	0.48	0.35	0.345
Suction	0.7	0.545	0.52
cauterizer	0.83	0.721	0.741
scissors	0.35	0.169	0.277
Mean: (mAP)	.56	0.395	0.432

Table 6.1: Average Precision of different tools

trained by the Baum-Welsch algorithm accurately labeled the surgical stage in 95% of all frames.

6.4 Summarization Performance

There are several ways to quantify performance, and we primarily do so through two metrics. First, we quantify percentage of chosen video clips that include irrelevant frames. Such irrelevant frames may include flushing scenes, scenes outside the patient, scenes including rapid camera movement, or blurry scenes. To have an easily readable summary, such scenes should not be included. These were graded by expert surgeons to decide which scenes were irrelevant.

$$Irrelevance = \frac{\#IrrelevantShots}{TotalShots} \tag{6.1}$$

Second, we quantify what percentage of key medical scenes were included in the video summarization. Expert Surgeons constructed a checklist of high priority scenes in each video as shown in Figure A.4. A high quality summary includes as many of these diverse scenes as possible. We compare three methodologies: (i) Our Method, (ii) Fully Randomly Chosen

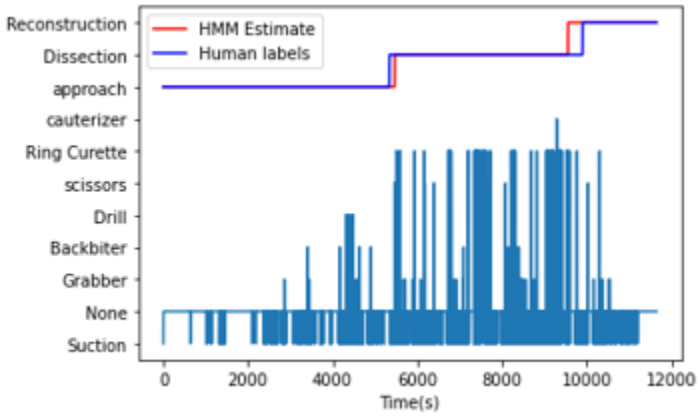

Figure 6.3: Stage Result Example. Video time in seconds is located in the X axis with split Y axis showing the surgical tool present in each frame and the surgical stage of the video

Clips, (iii) Magisto [10]. Magisto is a commercial summarization tool and represents state-of-the-art general summarization performance without the specialized domain knowledge. Our method utilized the stage estimation as previously reported and the methodology established in Section 5.5 using only frames that containing the tools according to Table 5.1

An example output surgery video is detailed in Figure 6.5. Calculated shot boundaries (purple vertical lines) are calculated that correspond naturally to moments in which the tool is removed from the patient or rapidly changes (see top row of frame). Representative clips are shown in blue according to Equation 5.9. In tests, summarization cut 10 videos from an average of 138 minutes down to 3 minutes.

6.4.1 Frame Irrelevance

Irrelevance performance, defined in equation 6.1, is shown in table 6.2. Our method includes Irrelevant frames in only 1% of the shots in an average video, compared to 36% of Commercially[10]-summarized videos.

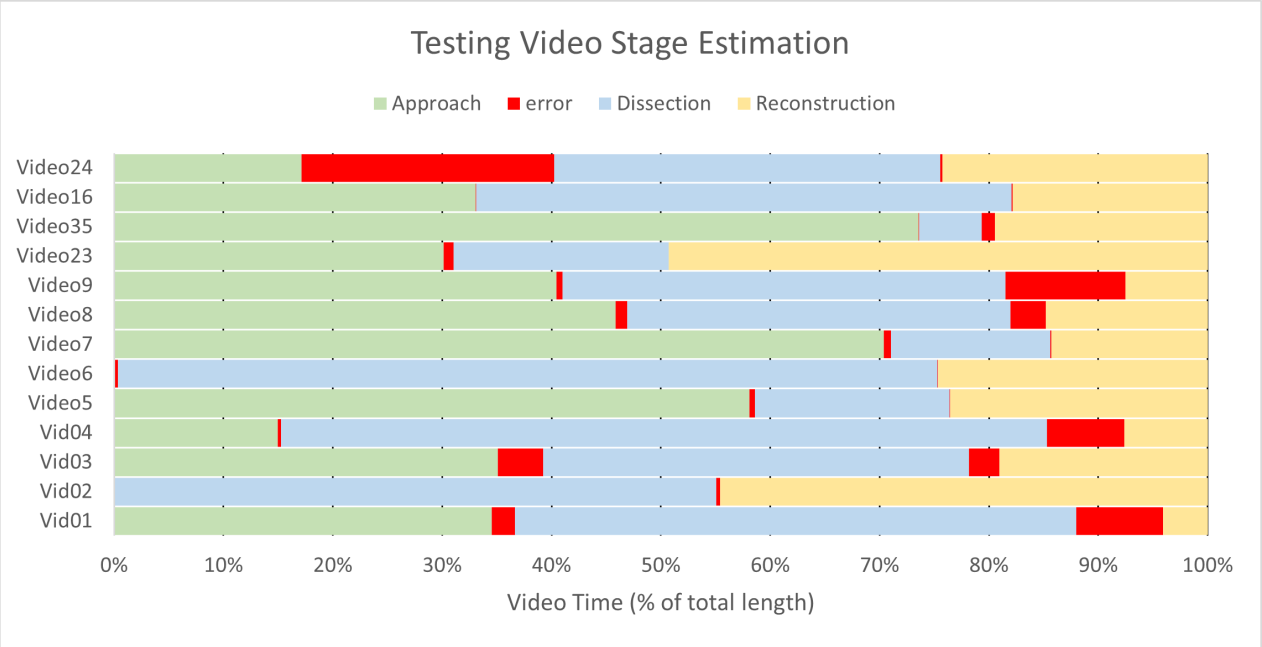

Figure 6.4: Stage Result for multiple videos. Video time is represented as a percentage of total video length. Error in stage estimation is represented by red bars

6.4.2 Medically Relevant Frame Sensitivity

As previously mentioned, we quantify the percentage of key medical scenes included in the video summarization. A checklist was created as 'gold standard' ground truth (viewable in appendix Table A.4) for each video by expert surgeons. 10 videos were annotated using this 'gold standard': video segments that clearly showed the gold standard elements were assigned a score of '1'. Video segments that partially show the gold standard element were assigned a score of '0.5'. For example, if fibrin glue is added as part of the 'gold standard' key medical frames, a score of '1' shows the direct application of fibrin glue. A score of '0.5' may represent a scene after the fibrin glue is applied, but clearly shows the application site and extent of application. The results across 10 videos are shown in Figure 6.6, in which the results are shown for our method (orange), the commercial product [10] (blue) and a random selection of clips (grey). Our Method had the highest performance across all videos,

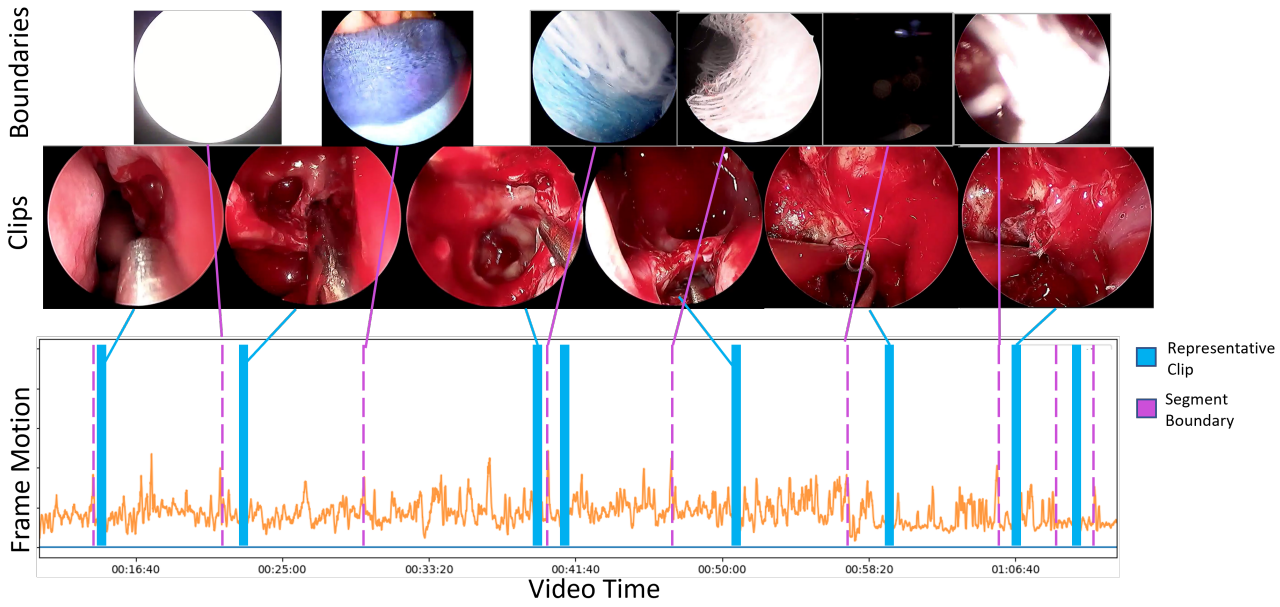

Figure 6.5: Overall Process example. Calculated frame motion is shown in orange. Peak motion events are used as natural shot boundaries (purple) in which the tool is typically removed from the patient. From the resultant shots, representative clips (blue, size is exaggerated), are chosen from the lowest mean distance to the whole segment

accurately capturing 84% of medically relevant scenes.

6.4.3 Time Reduction

Figure 6.7 shows the breakdown of each stage of the summarization process averaged on all 10 videos. Irrelevant frames (Frames outside patient, blurry/obscured, or containing no tools) account for 50% of overall video length, but 48% of frames must still be reduced in the summarization step. Overall, an average of 98% of video frames were removed to create a summary.

	Irrelevance		
	Our Method	Commercial	Improvement
Vid01	0.04	0.34	0.29
Vid02	0.00	0.14	0.14
Vid03	0.00	0.24	0.24
Vid04	0.02	0.15	0.13
Vid05	0.00	0.46	0.46
Vid06	0.00	0.45	0.45
Vid07	0.00	0.29	0.29
Vid08	0.00	0.71	0.71
Vid09	0.00	0.45	0.45
Average	0.01	0.36	0.35

Table 6.2: Ratio of Irrelevant Frames (Irrigation,flushing, Static Scenes, etc)

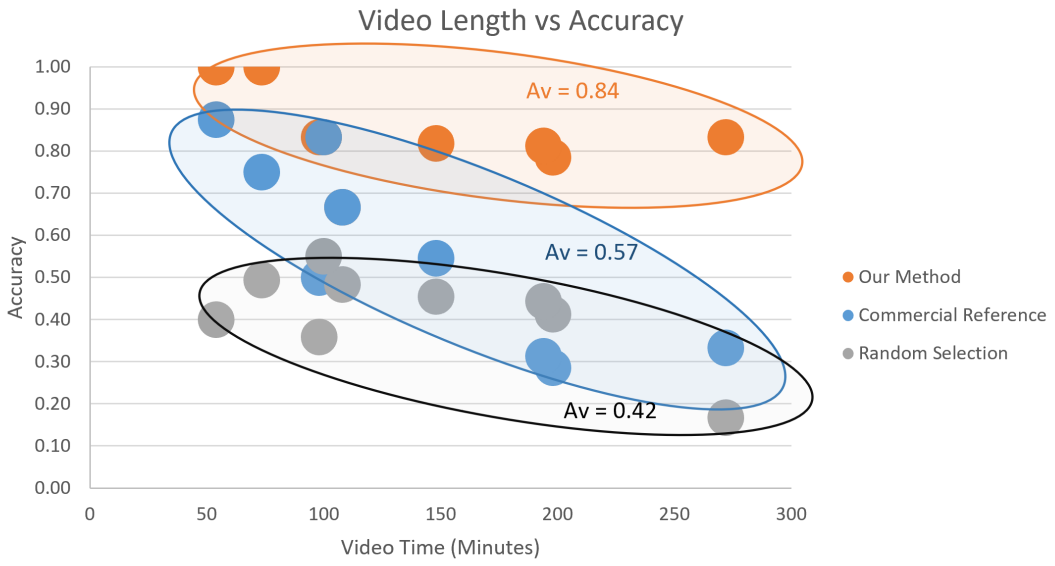

Figure 6.6: Accuracy of Videos percentage of medically valid clips used) X axis represents video length (Orange) = Our method, (Blue) = Commercial product[10]. (Grey) = Random Scene selection

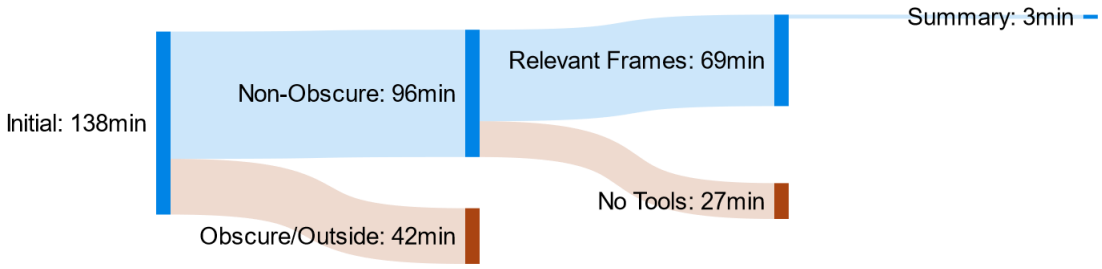

Figure 6.7: Overall Process achievement. Videos were condensed from an average of 138 minutes down to 3 minutes

Chapter 7

DISCUSSION

7.1 Tool Identification

On the surface, the tool identification performance seems relatively low, with a mAP of only 0.56. However, combined with a temporal mean smoothing and a probabilistic Hidden Markov Model, the end result is qualitatively acceptable. Furthermore, the performance is further improved by 'grouping' tools during the summarization stage. For example, a common misdetection for tools includes the scissors and the Ring Curette while the end of each tool is buried in tissue and unable to be detected. However, scissors and ring curette are kept as valid summarization targets for all stages, so the impacts of misdetections are strongly muted. The most important tools for detection are the suction tool and cauterizer, which are discarded in multiple surgical stages. These have a higher average precision (AP).

7.2 Summarization Performance

Overall, this pipeline has a greater accuracy compared to the commercial summarization tool. However, it still misses some scenes while discovering the best annotations. Notably, the major misses in this pipeline are regarding the visualization of the surrounding important anatomical structures/inspecting for additional tumor after removal, and capturing fibrin glue application. Regarding the visualization of anatomic structures, this is an expected result of the current pipeline, as this visualization occurs with no surgical tools in the scene, and is therefore removed by the software by design. Additional work must be done with a larger dataset of videos to detect the variety of nerves and post-tumor tissues similar to current tool detections. Additionally, the fibrin glue application is similarly troublesome, as the effect of the application of the glue results in a heavy 'blurring' effect on the frame,

making the frame unsuitable for general selection. Similarly, this fibrin glue application could be improved by dataset expansion and including the fibrin glue as detection targets.

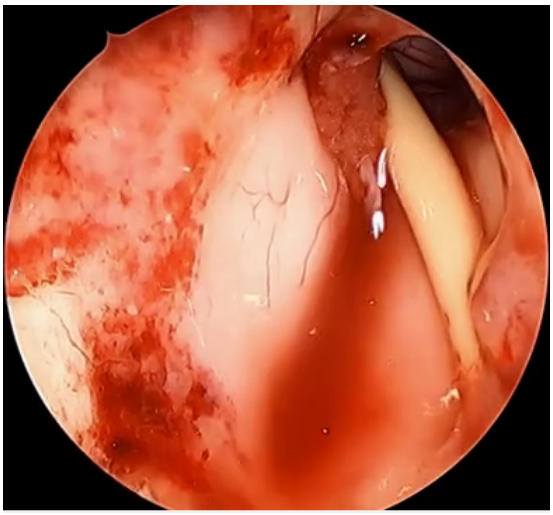

Figure 7.1: Visualization example. Surgeon visualizes nerve (top right) and checks for damage or ingress of tumor after removal. No tools are present in scene, making this difficult for the algorithm to identify. It is likely possible to expand the dataset and identify distinctive nerves to address

Chapter 8

APPLICATION

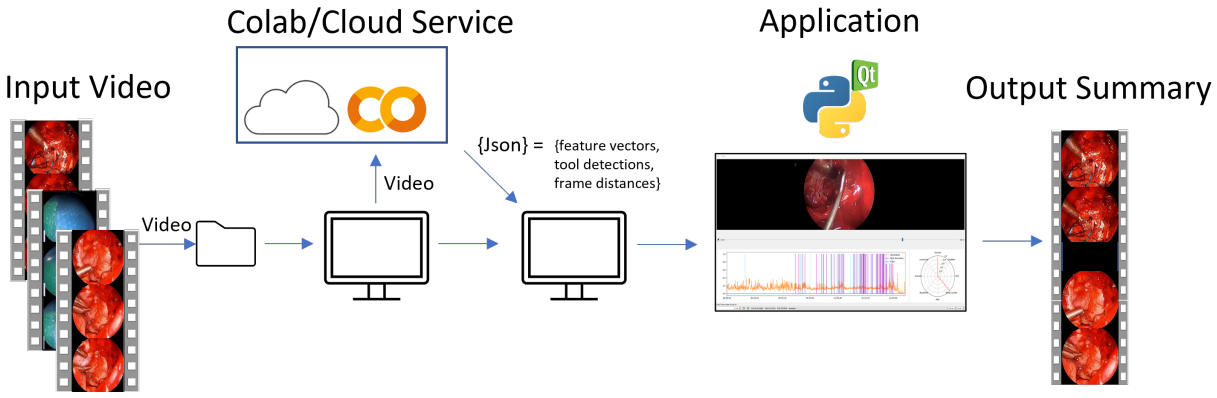

Figure 8.1: Application general flow. Videos are fed to cloud-based machine learning backend that caclulates the tool detection information and extracts feature vectors from the constituent frames. A frontend application written in PyQT is used to create, play, and visualize summaries

Towards the goal of creating a usable software for surgeon use, a two-stage pipeline was created to facilitate the use of the algorithms described as previously. Figure 8.1 shows the general flow of this application. A user submits video to a cloud service with GPU enabled. For all testing, Google Colab was used as this backend service. The backend service analyzes the video at a rate of 5Hz, and compiles the information into a json-structured file pickled in NumPy. This file contatins the following information: (i) The frame-by-frame feature vector as detailed in Section 5.4, (ii) the frame-by-frame tool detections in the form of a confidence. Faster-RCNN can return up to 2k bounding box proposals, so all proposals with a confidence lower than 0.5 are removed. Of the remaining boxes, overlapping bounding

boxes with iou 0.2 are removed to retain the highest confidence guess for a given tool. The final confidences for each tool class are recorded in the json files. Finally,(iii) the 'annotation' vector of consecutive frame differences (Equation 5.7) is recorded and sent to the front-end application.

8.0.1 Front-End Application

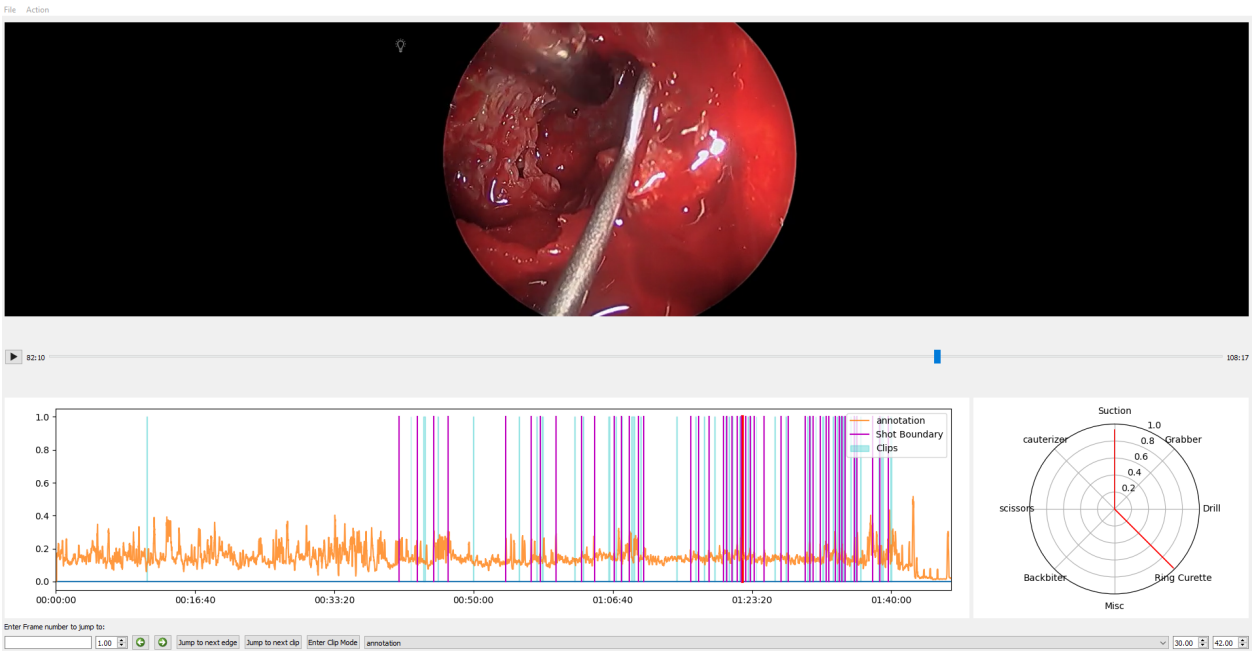

Figure 8.2: Front-end Video playing application. User controls the number of boundaries (Purple) and clips(blue). Orange line shows frame movement. Video is played in the viewer on the top and the constituent tool estimations are shown in the radial plot on the left.

The front-end application written in PyQT (image shown in Figure 8.2) is applied to provide simple usability and parameter tuning. A user may view the video within the application alongside the corresponding frame data such as tool presence. A user may then specify the smoothing factor for equation 5.7 and the number of boundaries b for summarization. The user may then view the resulting boundaries and clips according to Figure 8.2.

8.0.2 Future Work

While this application is sufficient for summary generation and demonstration, additional work should be done to make this application fully surgeon usable. For example, converting the back-end application to mesh with a proper front-end app to be runnable on any computer without GPU acceleration and without logging into a secondary backend application. Furthermore, additional user improvements can be made such as individually user-draggable clips.

Chapter 9

CONCLUSION AND FUTURE WORK

In this work, we apply a combination of visual data techniques to solve the video summarization problem for ESS surgeries. Our combination of frame classifiers, tool detection framework, and hidden markov model for estimating surgical stage, makes it possible to create effective video summaries while reducing the overall video size by 98%, while retaining 84% of key medical frames. This method outperforms general baselines such as the commercial Magisto system.

Future work would include extending this framework by adding additional HMM models for other surgery types or to expand the tool detection and valid frame detection stages to optimize for other endoscopic surgeries such as endoscopic sinus surgery. Additionally, some additional scenes could be added by simple expansion of dataset. As mentioned previously, the visualization of the optic nerve was a medically relevant video segment that did not have tools present. Expanding the dataset to explicitly recognize non-tool medical features such as the optic nerve would improve the ability to accurately capture additional surgical scenes. Finally, the detection results for some tools such as the scissors and forceps can be improved.

Ultimately, we successfully implement a pipeline to identify valid video frames, detect tools in such frames, separate the video into its constituent stages, find shot boundaries by identifying high frame movement, and create a resultant video summary.

Appendix A

A

State X	Suction	Forceps	Drill	R Curette	Rongeur	Scissors	Cauterizer
Approach	0.55	0.3	0.2	0.01	0.02	0.05	0.1
Operation	0.5	0.2	0.05	0.3	0.007	0.18	0.01
Reconstruct	0.52	0.3	0.01	0.1	0.002	0.14	0.1

Table A.1: Initial Emissions Estimate representing the probability an observation is seen in each state

	Start	Approach	Dissection	Reconstruction	End
Start		0.9	0.1		
Approach		0.9	0.1		
Operation			0.9	0.1	
Reconstruct				0.9	0.1
End					1

Table A.2: Initial Transmission Probability estimates

		GroundTruth		Baum-Welch Estimate		Result	
Video	Total Length (min)	Dissection start	Re-construction start	Dissection start	Re-construction start	Error (minutes)	Accuracy
Vid01	54	18.7	47.5	19.8	51.8	5.4	0.90
Vid02	148	0	82	0	81.5	0.5	1.00
Vid03	108.3	38	84.7	42.5	87.7	7.5	0.93
Vid04	73.5	11	62.7	11.2	67.9	5.4	0.93
Video5	198	116	151.3	115	151.4	1.2	0.99
Video6	272	0	205	0.9	204.8	1.1	1.00
Video7	98	69	84	69.6	83.9	0.7	0.99
Video8	88	89	165.3	91	159	8.3	0.91
Video9	194	41	92.5	40.5	81.5	11.6	0.94
Video23	68	20.5	34.5	21.1	34.5	0.60	0.99
Video35	144	106	116	105.9	114.3	1.80	0.99
Video16	195	64.5	160	64.6	160.2	0.3	1.00
Video24	103	17.7	78	41.5	77.8	24.0	0.77
						Average	0.95

Table A.3: Results from surgical stage estimation in minutes. Information from tool presence alone is able to predict the surgical stage with 95% accuracy.

		Surgical Stage										—	Accuracy		
Video	Length (min)	Drill/ Expose floor of Sella	Removing floor of sella	Cutting tumor capsule/ dura	Tumor resection	check / visualize tumor	Control bleed/ Hemostasis	Surgical Cavity Packed	Sizing fascia	bone graft	Fibrin glue	Flap	Our Algorithm	Random Choice	Commercial [10]
Vid01	54	x	x		xxxx			x		x	x		1.00	0.40	0.88
Vid02	148			x	xxxx	x			x	x	x	x	0.82	0.45	0.55
Vid03	108		x	x	x	x	x				x		0.67	0.48	0.67
Vid04	74		xx	x	xxxx			x		x	x		1.00	0.49	0.75
Video5	198		xx	x	x	x	x				x		0.79	0.41	0.29
Video6	272				x	x			x	x		x	0.83	0.17	0.33
Video7	98	x	x		x		x			x	x		0.83	0.36	0.50
Video8	194	x	x	x	x	x			x	x	x		0.81	0.44	0.31
Video9	100	x	x	x	x						x	x	0.83	0.55	0.83
												Average	0.84	0.42	0.57

Table A.4: Video Summarization Results. Tested Videos with 'x' indicating which surgical scene is present in the video. Multiple 'x' marks indicate that multiple independent scenes fall within this descriptor. Accuracy for each video (percentage of target scenes present in summarization) is shown in the columns on the right.

BIBLIOGRAPHY

[1] Okeyode T Santo L. *National Ambulatory Medical Care Survey, 2018 Summary Tables.* 2018. URL: \url{https://www.cdc.gov/nchs/data/ahcd/namcs_summary/2018-namcs-web-tables-508.pdf}.

[2] Steve C Lee and Brent A Senior. "Endoscopic Skull Base Surgery". In: *Clinical and experimental otorhinolaryngology vol 1,2* (2008). DOI: 10.3342\/ceo.2008.1.2.53.

[3] Paolo Cappabianca and Luigi Maria Cavallo. *Endoscopic pituitary and skull base surgery, Anatomy and surgery of the endoscopic endonasal approach.* Germany: Endo Press GmbH, 2016.

[4] Neil Bhattacharyya. "Ambulatory sinus and nasal surgery in the United States: demographics and perioperative outcomes". In: *The Laryngoscope* 120.3 (2010), pp. 635–638.

[5] Cancer Research UK. *Diagram showing surgery through the nose.* 2014. URL: https://commons.wikimedia.org/wiki/File:Diagram_showing_surgery_through_the_nose_CRUK_275.svg.

[6] Danila Potapov et al. "Category-Specific Video Summarization". In: *Computer Vision – ECCV 2014.* Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 540–555. ISBN: 978-3-319-10599-4.

[7] Tao Mei et al. "Near-Lossless Semantic Video Summarization and Its Applications to Video Analysis". In: *ACM Trans. Multimedia Comput. Commun. Appl.* 9.3 (2013). ISSN: 1551-6857. DOI: 10.1145/2487268.2487269. URL: https://doi.org/10.1145/2487268.2487269.

[8] Jae-Gon Kim et al. “Efficient camera motion characterization for MPEG video indexing”. In: *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532)*. Vol. 2. 2000, 1171–1174 vol.2. DOI: 10.1109/ICME.2000.871569.

[9] Guoying Jin, Linmi Tao, and Guangyou Xu. “Hidden Markov Model Based Events Detection in Soccer Video”. In: vol. 3211. Sept. 2004, pp. 605–612. ISBN: 978-3-540-23223-0. DOI: 10.1007/978-3-540-30125-775.

[10] Magisto. *Magisto Web Online Video Editor*. 2021. URL: http://www.magisto.com/.

[11] Oren Boiman and Michal Irani. *Data Similarity and Importance using Logical and Global Evidence Scores*. June US Patent 8,200,648, Jun. 2012.

[12] Oren Boiman and Rav-Acha. *System and method for semi-automatic video editing*. US Patent 9,502,073, Nov. 2016.

[13] Mayu Otani et al. “Video Summarization Using Deep Semantic Features”. In: *Computer Vision – ACCV 2016*. Ed. by Shang-Hong Lai et al. Cham: Springer International Publishing, 2017, pp. 361–377. ISBN: 978-3-319-54193-8.

[14] Wencheng Zhu et al. “DSNet: A Flexible Detect-to-Summarize Network for Video Summarization”. In: *IEEE Transactions on Image Processing* 30 (2021), pp. 948–962. DOI: 10.1109/TIP.2020.3039886.

[15] Ke Zhang, Kristen Grauman, and Fei Sha. “Retrospective Encoders for Video Summarization”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

[16] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[17] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *CoRR* abs/1905.11946 (2019). arXiv: 1905.11946. URL: http://arxiv.org/abs/1905.11946.

[18] Lingga Adidharma et al. “Semiautomated Method for Editing Surgical Videos”. In: *Special Virtual Symposium of the North American Skull Base Society*. Georg Thieme Verlag KG, Feb. 2021. DOI: 10.1055/s-0041-1725452. URL: https://doi.org/10.1055/s-0041-1725452.

[19] Tsung-Yi Lin et al. “Microsoft COCO: Common Objects in Context”. In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

[20] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767. URL: http://arxiv.org/abs/1804.02767.

[21] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. “YOLOv4: Optimal Speed and Accuracy of Object Detection”. In: *ArXiv* abs/2004.10934 (2020).

[22] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[23] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. “Objects as Points”. In: *CoRR* abs/1904.07850 (2019). arXiv: 1904.07850. URL: http://arxiv.org/abs/1904.07850.

[24] Jiann-Der Lee et al. “Automatic Surgical Instrument Recognitionâ€”A Case of Comparison Study between the Faster R-CNN, Mask R-CNN, and Single-Shot Multi-Box Detectors”. In: *Applied Sciences* 11.17 (2021). ISSN: 2076-3417. DOI: 10.3390/app11178097. URL: https://www.mdpi.com/2076-3417/11/17/8097.

[25] Yuying Liu et al. “An Anchor-Free Convolutional Neural Network for Real-Time Surgical Tool Detection in Robot-Assisted Surgery”. In: *IEEE Access* 8 (2020), pp. 78193–78201.

[26] Max Allan et al. “2017 Robotic Instrument Segmentation Challenge”. In: *CoRR* abs/1902.06426 (2019). arXiv: 1902.06426. URL: http://arxiv.org/abs/1902.06426.

[27] Pan Shi et al. "Real-Time Surgical Tool Detection in Minimally Invasive Surgery Based on Attention-Guided Convolutional Neural Network". In: *IEEE Access* 8 (2020), pp. 228853–228862. DOI: 10.1109/ACCESS.2020.3046258.

[28] Fangbo Qin et al. "Towards Better Surgical Instrument Segmentation in Endoscopic Vision: Multi-Angle Feature Aggregation and Contour Supervision". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 6639–6646. DOI: 10.1109/LRA.2020.3009073.

[29] Shan Lin et al. "Multi-Frame Feature Aggregation for Real-Time Instrument Segmentation in Endoscopic Video". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6773–6780. DOI: 10.1109/LRA.2021.3096156.

[30] Fangbo Qin et al. "Surgical instrument segmentation for endoscopic vision with data fusion of cnn prediction and kinematic pose". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 9821–9827. DOI: 10.1109/ICRA.2019.8794122.

[31] Haonan Peng et al. *Reducing Annotating Load: Active Learning with Synthetic Images in Surgical Instrument Segmentation*. 2021. DOI: 10.48550/ARXIV.2108.03534. URL: https://arxiv.org/abs/2108.03534.

[32] Carly Garrow et al. "Machine Learning for Surgical Phase Recognition A Systematic Review". In: *Annals of Surgery* Publish Ahead of Print (Nov. 2020). DOI: 10.1097/SLA.0000000000004425.

[33] Yueming Jin et al. "EndoRCN : Recurrent Convolutional Networks for Recognition of Surgical Workflow in Cholecystectomy Procedure Video". In: 2016.

[34] Yueming Jin et al. "Multi-Task Recurrent Convolutional Network with Correlation Loss for Surgical Video Analysis". In: *CoRR* abs/1907.06099 (2019). arXiv: 1907.06099. URL: http://arxiv.org/abs/1907.06099.

[35] Rémi Cadène et al. "M2CAI Workflow Challenge: Convolutional Neural Networks with Time Smoothing and Hidden Markov Model for Video Frames Classification". In: *CoRR* abs/1610.05541 (2016). arXiv: 1610.05541. URL: http://arxiv.org/abs/1610.05541.

[36] Robert DiPietro et al. "Automated surgical-phase recognition using rapidly-deployable sensors". In: *Proc MICCAI Workshop M2CAI*. 2015.

[37] Blake Hannaford and Paul Lee. "Hidden Markov Model Analysis of Force/Torque Information in Telemanipulation". In: *The International Journal of Robotics Research* 10.5 (1991), pp. 528–539. DOI: 10.1177/027836499101000508. eprint: https://doi.org/10.1177/027836499101000508. URL: https://doi.org/10.1177/027836499101000508.

[38] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Representations by Back-propagating Errors". In: *Nature* 323.6088 (1986), pp. 533–536. DOI: 10.1038/323533a0. URL: http://www.nature.com/articles/323533a0.

[39] Mingxing Tan and Quoc Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 6105–6114. URL: https://proceedings.mlr.press/v97/tan19a.html.

[40] Mark Sandler et al. "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation". In: *CoRR* abs/1801.04381 (2018). arXiv: 1801.04381. URL: http://arxiv.org/abs/1801.04381.

[41] L.R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286. DOI: 10.1109/5.18626.

[42] Shan Lin et al. *UW sinus surgery cadaver/live dataset (UW-sinus-surgery-C/L)*. 2020.

[43] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *CoRR* abs/2004.10934 (2020). arXiv: 2004.10934. URL: https://arxiv.org/abs/2004.10934.

[44] Jacob Schreiber. "Pomegranate: fast and flexible probabilistic modeling in python". In: *Journal of Machine Learning Research* 18.164 (2018), pp. 1–6.