Recognition and retrieval tasks in large quasi-unannotated surgical video databases

Tong Yu

UNIVERSITÉ DE STRASBOURG

DOCTORAL SCHOOL MSII

ICube Laboratory (UMR 7357)

THESIS presented by :

Tong YU

Defended on: **December 9th, 2021**

For obtaining the degree of **Doctor of Philosophy**
from the University of Strasbourg

Field: Computer Science

Recognition and Retrieval Tasks in Large Quasi-Unannotated Surgical Video Databases

Thesis directors:

Prof. Nicolas Padoy	Professeur, Université de Strasbourg
Prof.Didier Mutter	Professeur, Hôpitaux Universitaires de Strasbourg

Chair of the Committee:

Dr. Marie-Odile Berger	Directrice de recherche, INRIA Nancy

Examiners:

Prof. Danail Stoyanov	Professeur, University College of London
Dr. Pierre Jannin	Directeur de recherche, INSERM Rennes

Contents

Abstract

The work presented in this thesis tackles the problem of video analysis for laparoscopic interventions, in the case of very scarcely annotated datasets. During laparoscopic surgery, the live video feed from inside the abdominal cavity of the patient is the keystone of the entire procedure, providing all the visual feedback required by the surgeon and the staff to carry out the intervention. Our objective is to develop methods capable of leveraging this source of information and understand it; thus providing the foundation for context-aware, vision-based systems to be developed in the future for decision support in the operating room. Recent breakthroughs in computer vision, spearheaded by deep learning methods brought major advances to surgical video analysis. However the current default approach of full supervision is an obstacle for future developments. Laparoscopic procedures generate vast amounts of video data which, even if stored, will for the most part almost certainly remain unannotated.

This thesis investigates the use of unannotated data across multiple visual tasks. In our first approach, we propose an automatic annotation method relying on a very small ratio of manually annotated data, and demonstrate the usability of the automatically annotated data for training real-time CNN-LSTM predictors. Our second approach then shows how this unannotated data can be leveraged and explored in a scalable, OR-compatible manner and without any annotations using video hashing. By learning, in a self-supervised manner, searchable binary representations of surgical videos, we are able to retrieve video content matching a given scene, represented by a video clip, in terms of surgical phase or surgical critical event.

Acknowledgements

This manuscript is the end result of several years of doctoral studies at the University Hospital of Strasbourg, as a member of team CAMMA. Before presenting the research carried out over this period of time, I will use the next few lines to honor the help and support that was given to me throughout my thesis.

First of all, I would like to thank the members of the jury: Dr. Marie-Odile Berger, Dr. Pierre Jannin and Pr. Danail Stoyanov. It was an honor to have my work evaluated by such influential members of the community.

I would like to thank my main supervisor Pr. Nicolas Padoy for the consistently valuable feedback given to me during my thesis. The growth of our team under his leadership over the past few years has been inspiring to say the least. Thanks also to Pr. Didier Mutter, who co-supervised this thesis and authorized access to the data necessary for all my experiments.

Much of this work would not have been possible without a number of key collaborators. In that regard I would like to thank to the clinicians who annotated the data used in this work: Dr. Pietro Mascagni, Dr. Juan Verde, Dr. Cristians Gonzalez. Thanks to the MOSaiC team as well, for providing excellent annotation software.

I am thankful of course for every member of team CAMMA - past and present. All of them contributed to creating a work environment I will remember very fondly long after my time here; especially Chinedu and Vinkle, who, as fellow PhD students, were in the same boat I was the entire time.

Outside of the lab, I am definitely thankful for the company of my friends from high school as well as various people who supported me along the way. Cara, thank you. Special thanks to Diane and JB for the postcards, which I very much enjoyed.

Of course, all of this would not be complete without properly thanking my parents. Thank you from the bottom of my heart, Mom and Dad.

List of Figures

List of Tables

List of Abbreviations

ANN Artificial Neural Network

CNN Convolutional Neural Network

DNN Deep Neural Network

CRF Conditional Random Field

LA-CODE Look-Ahead Code

LA-RECO Look-Ahead Reconstruction

LSTM Long Short-Term Memory

OR Operating Room

RNN Recurrent Neural Network

SEQ2SEQ Sequence-to-Sequence

SSTH Self-Supervised Temporal Hashing

SDS Surgical Data Science

ULA-CODE Uncertain LA-CODE

CHAPTER 1

Introduction

"“Even this must have a preface - that is, a literary preface,” laughed Ivan, “and I am a poor hand at making one.””

- Fyodor Dostoevsky, *The Brothers Karamazov*

Contents

1.1 Clinical & technological context

1.1.1 The OR in the minimally invasive surgery era

Among all the options available to healthcare providers for treatment, surgery often stands out as the most heavy-handed, and is only prescribed to a patient after careful diagnosis and deliberation. The surgeon is given the "authority to cure by means of bodily invasion" [Gaw12] and is therefore able to intervene directly on the area of interest; however it is well understood that this can be a double-edged sword, and carries risks of major collateral damage. For this reason the history of technological advances in surgery is defined by an increasing amount of concern and respect for the patient's integrity: from the discovery of anesthesia in the XIXth century, passing by the discovery of antiseptics, to medical imaging and finally the advent of minimally invasive surgery in the XXth century.

Minimally invasive surgery (Figure 1.1) decreases adverse outcomes and patient discomfort by restricting the size of the incisions employed for accessing a body cavity. The resulting loss in direct visibility in those procedures is generally compensated by the use of intraoperative imaging devices: in the case of laparoscopy, which is the modality featured in the work presented here, the instruments are inserted through very small incisions, along with an optical camera called a laparoscope (Figure 1.1). The surgeon is fully reliant on the picture from its video feed, displayed on a separate monitor in the operating room.

Figure 1.1: Left: trocar inserted into the patient's abdomen, as preparation for the laparoscopic procedure. Right: Surgeon performing a suture, with instruments inserted through trocars. The image on the right is the feed from the laparoscope used for visual feedback.

1.1.2 The OR as a high-stakes, timing-sensitive environment

While minimally invasive surgery has overall been a major step forward in the development of surgery, the introduction of any new technology to the OR comes at an expense, both in terms of time and material resources. According to a 2012 review by Ramsay et al. [RPR+12], total costs for a laparoscopic setup exceed 161K $: this includes the laparoscope as well as the display unit, surgical instruments, and the generator powering energy-based instruments. Yet, this number is dwarfed by the 2M $ cost of Intuitive Surgical's Da Vinci robot, used in many gastrointestinal surgeries. In addition to these already high upfront costs, complex medical devices often require

long, specialized training: the IRCAD's course for laparoscopic general surgery takes place over a week full-time, for a total of 2780 euros per participant. Finally each new item entering the OR generates supplementary costs and delays due to sanitization requirements and maintenance.

Accumulating those technological expenses contributes to surgery's colossal footprint on total healthcare costs, which was evaluated at 1/3 in the United States in a 2014 study [CMG18]. Within expenses related to surgical care, the operating room ranks second: in 2014, it was established that a single minute of OR usage in California costs between 36 and 37 $ [CMG18]. Considering the average duration of 110 minutes for a Roux-en-Y gastric bypass in our dataset, a crude estimation for only one of these procedures puts its OR costs in the 4000 $ range. More importantly, the standard deviation in duration is considerable: 30 minutes, which translates to a 1100 $ swing. By cumulating over a year for the 43000 [EDH+20] Roux-en-Y gastric bypass surgeries performed in the US, OR expenses would range between 129 and 215 M$, and are therefore largely impacted by in-OR workflow and efficiency. Beyond financial considerations, the impact on patient outcomes and quality of treatment - which are obvious top priorities - is critical as well. Tighter and more accurate timings enable treating more patients, keeping them under anesthesia for shorter periods of time and decreasing odds of adverse events. For instance, a study conducted by Peng et al. [fPZY+09] shows how prolonged insufflation of the abdominal cavity in laparoscopy can lead to hypothermia. CO_2 is also known to cause acidosis in the exposed tissues of the abdominal cavity [Bon17].

Yet, OR time management is often up to the surgeon. It is common practice for the surgeon to have the next patient brought in 15 minutes before the end of their current intervention. This requires taking a guess on a complex procedure, accounting for intraoperative conditions, surgical skill and critical events. "Active bleeding" critical events during laparoscopic interventions, for example, can severely disrupt the intervention; they require the surgeon to stop their course of action and switch to cleanup, using a dedicated irrigation & suction device. According to Travis et al. [TTW+14], surgeons underestimate procedure duration by 31 minutes on average.

1.1.3 The proliferation of signals in the OR & the challenge of data

Under such high-pressure conditions, one would think the technological resources given to surgeons would help counterbalance the difficulty and make the workflow of surgery easier to navigate. In many aspects the opposite is true: while the proliferation of devices and signals in the operating room provides many benefits both in terms of comfort for the surgeon and patient outcomes, its current burden on the surgical staff is not to be neglected. Each device in the OR increases pre and post operative workload, due to sanitization and setup. Intraoperatively, many systems failure events need to be accounted for: loss of video feed, loss of power, loss of visibility are issues sometimes found in recordings from our Endocorpus dataset. Each device may also be a source of disruptions during the procedure: through visuals such as blinking lights and monitor displays, but also sounds. A study by Wung et al. [WMS18] lists alarms as the number one technological hazard in the emergency room, while the complexity of surgeries themselves can be a heavy burden [BML+20]. This cognitive

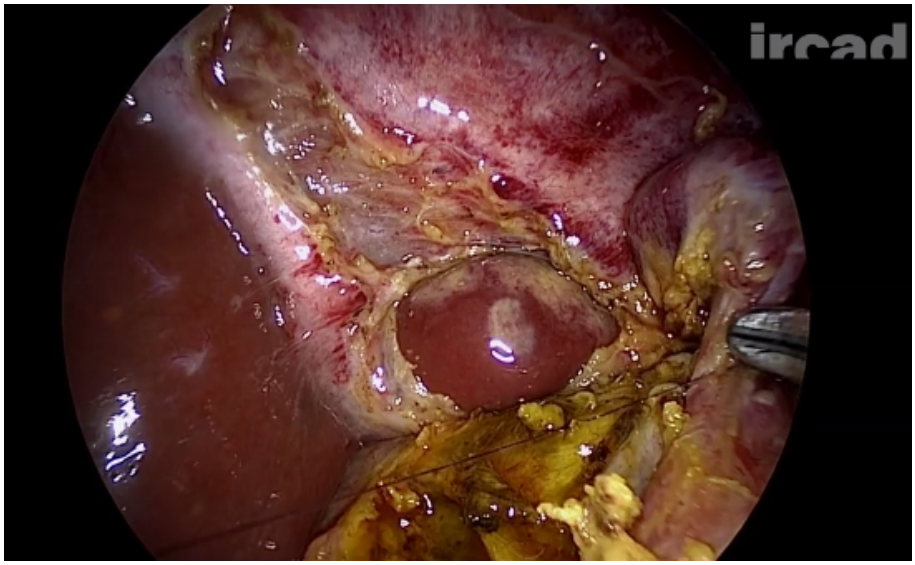


Figure 1.2: Aftermath of iatrogenic bile duct injury. The common bile duct is sometimes mistaken for the cystic duct and cut as a result. Courtesy of Websurg.

workload may manifest in increased stress, burnout and ultimately surgical errors. Cholecystectomy provides an eloquent example supporting this: for the most part the surgical community has switched from open surgery to the laparoscopic approach, which has improved outcomes in all areas except one, bile duct injury (BDI) rate. Having increased from 0.5 to 1.5% [MVA+20], this incident exposes the discomfort and cognitive burden of indirect vision on a separate monitor, which occasionally results in involuntarily damage to certain anatomical structures (figure 1.2).

In recent years, the surgeon has been given highly advanced technology to extend their perception and manipulation capabilities; on the other hand, little effort has been made to provide them with adequate tools to process information that is ever-increasingly rich and difficult to process. This very information, however, may be the key to a solution: surgical data, if systematically collected on a large enough scale, can be used to develop powerful support tools for surgeons. This is the core premise behind *Surgical Data Science* [MHVS+17, HMHV20]; according to its originators [MHVS+17], this emerging discipline "focuses on the acquisition, modeling and analysis of data to improve the quality of interventional healthcare". Three key clinical applications are targeted:

- Decision support: by using comprehensive data to expand the information available to the surgeon, otherwise limited to their own experience
- Surgical training and skill evaluation: by using objective analytics to provide automated feedback to trainees
- **Context-aware assistance: by building AI models capable of understanding clinical situations relevant previous cases - in real time and responding accordingly with properly timed support**

This last item is the focal point of the work presented here: knowing how and when to interact with the surgeon based on the real-time context ultimately determines

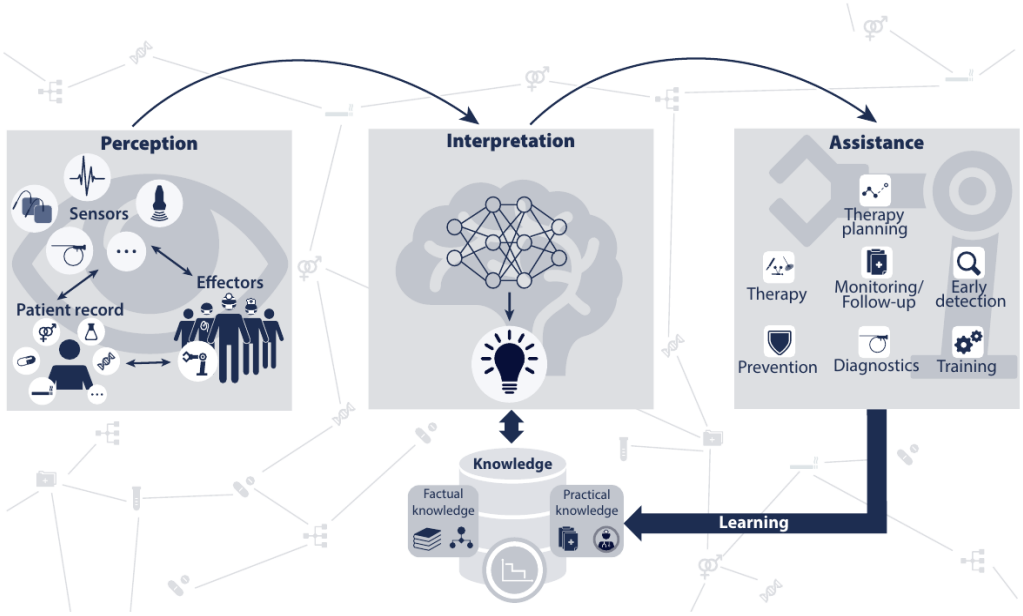


Figure 1.3: Overview of Surgical Data Science concepts. Image credits: Maier-Hein et al. [MHVS+17]

whether a system is an asset or a liability in the OR. For instance, to guarantee a safe procedure, sending safety check alarms is a straightforward task. However doing so at the correct time is not: too often and the alarm system turns into a distraction at best, a hazard at worst [WMS18]; too sporadically and the enforcement of safety checks loosens, allowing for more surgical errors. Both are characterized by a lack of awareness with regards to context: for example not knowing the overall progress of the surgery [TYM+19], or the current surgical phase [Twi17], or the surgical actions [NGY+20] taking place. In all of these areas, the improvement AI support is capable of bringing is considerable.

1.2 Models of AI-enhanced support

1.2.1 The AI-enhanced control unit

1.2.1.1 Context-awareness in non-clinical domains

Surgical Data Science is a critical first step before the actual deployment of new context-aware applications in the OR. Surgery, however, is far from the only domain regimented by tight timings and low margins for error - and yet several of these domains are far closer to the deployment stage, or even reached it years ago with AI or vision-enhanced support infrastructures already heavily in place. Examples from those other fields may serve as models for what future developments for an "OR control tower", as envisioned by Mascagni et al. [MP21], could ultimately look like.

In the automotive industry, one of the major motivations for self-driving vehicles besides limiting accidents is the ability to drastically cut down traffic congestion, and thereby CO_2 emissions and fuel costs. Replacing human drivers with a context-aware

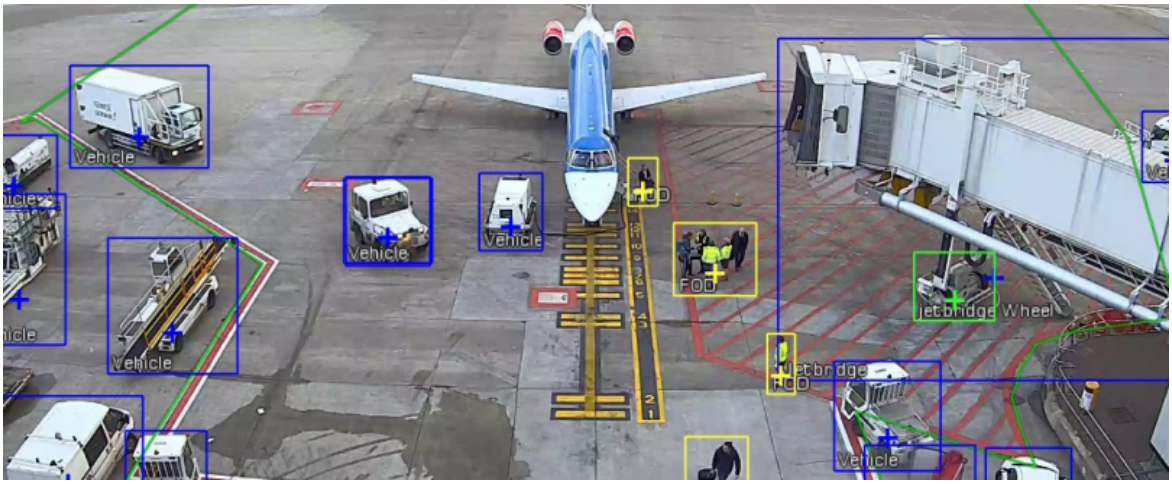


Figure 1.4: AI-powered control tower concept by Searidge. Image credits: NVIDIA.

system that integrates vision from onboard cameras but also nearby traffic data enables quick and optimal decision making, thus removing traffic jam causes such as human driver reaction time accumulation [Fri16].

In 2015, the Canadian company Searidge released AIMEE (Figure 1.4), an AI framework that can process up to 200 simultaneous video feeds in order to track aircrafts in airports. This information can be used to provide AR overlays to the airport security staff and report anomalies. The end goal is to enable fully remote, vision-based control towers, to replace current ones that are pinned down by radar infrastructure requirements.

Live sporting events are particularly ahead of the curve in terms of vision-enhanced support. Tennis tournaments have employed ball tracking technologies since the 2000s; this information is transmitted to TV production trucks, which use it to generate overlays for the audience. The "1st and 10 line" (Figure 1.5) in American football has been implemented by SportsMEDIA [Spo21] for the 2013 Super Bowl, as a visual cue for the audience. Hawk-Eye's vision-based goal line technology has become critical in soccer for supporting referree decisions.

1.2.1.2 Potential clinical applications

The key takeaway from the other domains mentioned before is the **control tower** concept: a unit equipped with ample computing resources, continuously collecting signals from the center of activity, determining its context and returning feedback to it. Similar to production trucks for sporting events or next-generation flight control towers, an infrastructure should collect video feeds from the various operating rooms in one or even several clinical facilities, interpret them using artificial intelligence and provide appropriately timed support to surgeons. The ABITO project [MVA+20] is an example of context awareness driving in-OR support: during laparoscopic cholecystectomy, a control unit reads the endoscopic video feed from an operating room 1.6, and orders an intra-operative timeout when it determines the surgeon is about to cut the cystic duct.

However while future approaches to context awareness in the OR should draw inspiration from other domains, it is important to recognize challenges that are specific

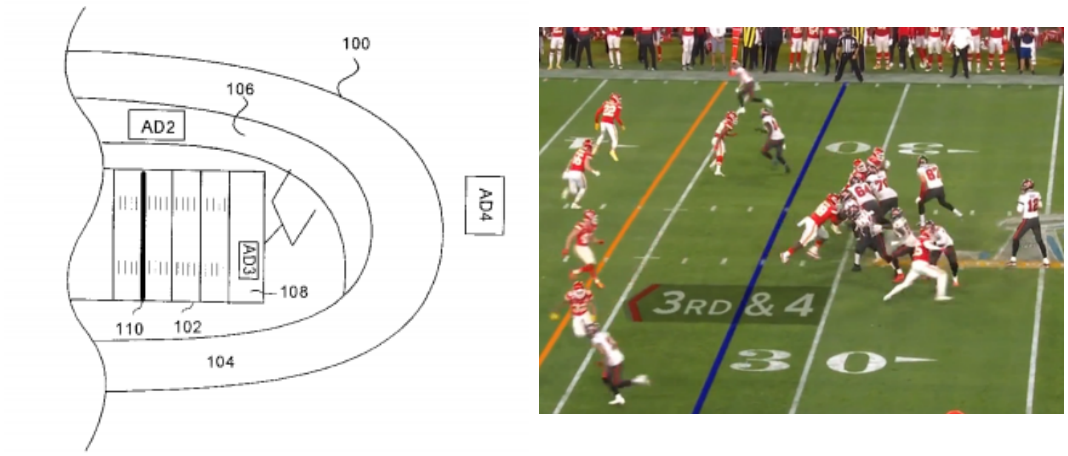


Figure 1.5: Left: diagram from the original patent by Honey et al. for the 1st down line overlay, using multiple cameras in the stadium for registration. Right: Image from Super Bowl LV, with the 1st down line as a dynamic, real-time overlay. It leaves players unoccluded, behaving like a physical line. Image credits: NFL.

to surgery. Aircrafts, road infrastructure, football fields and player uniforms can all be standardized to a fairly advanced degree. Patient anatomy, for obvious reasons, currently cannot; which logically translates to extremely high variability in surgical workflow - for example the presence of adhesions in the abdominal cavity varies a large amount from one patient to another. This in turn induces variability in the workflow, forcing the surgeon to dedicate more or less time to adhesiolysis (Figure 1.7). Even in a highly standardized procedure such as cholecystectomy, actions may look drastically different between surgeries due to changing teams and habits of clinicians: this manifests, for instance, in non-standard usage of surgical tools; e.g, pulling apart tissue with graspers instead of using the laparoscopic hook, or electrifying a grasper with an energy-based tool to perform cauterization.

Ultimately, with this variability in mind, the current understanding of context awareness needs to be challenged. From a computer vision algorithm's perspective "understanding the context" in an endoscopic video can mean "recognizing one of seven standardized surgical phases", "recognizing one of seven standardized surgical tools" or even "recognizing one of a hundred standardized surgical actions". However when standardized categories fail to capture the full understanding of an endoscopic video, the best way a system can describe it may very well be another endoscopic video from a large enough database.

1.2.2 Harnessing large databases with search

A common approach for navigating unknown pieces of data is to search for similar items in a large and diverse database. Despite its simplicity this is a well-accepted principle; most notably, even state-of-the-art AI assistants such as Google Assistant (Google) or Siri (Apple) may defer to a popular search engine upon failure to process an instruction or question.

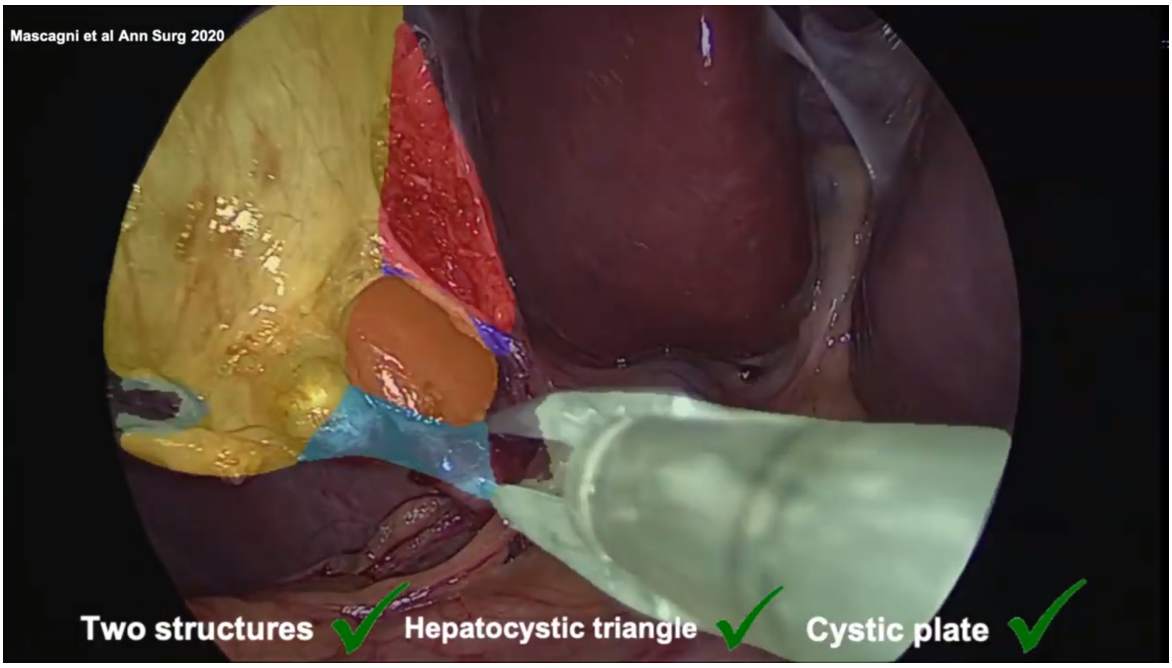


Figure 1.6: AI-controlled safety check: critical view of safety in cholecystectomy. Based on its reading of the anatomy and the instruments, the system determines all three criteria for safely cutting the cystic duct are met. Image credits: Mascagni et al. [MVA+20]

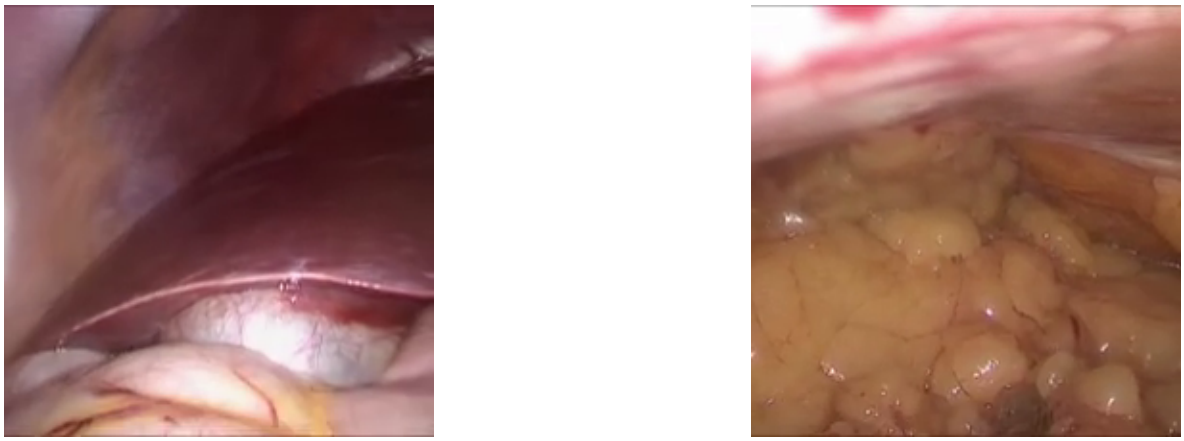


Figure 1.7: (a): Abdominal cavity presenting low amounts of adhesions. (b): Abdominal cavity with higher amounts of adhesions. Not only does this anatomical variation increase the visual difficulty of recognition problems, it also alters the workflow, requiring the surgeon to perform adhesiolysis prior to the actual intervention.

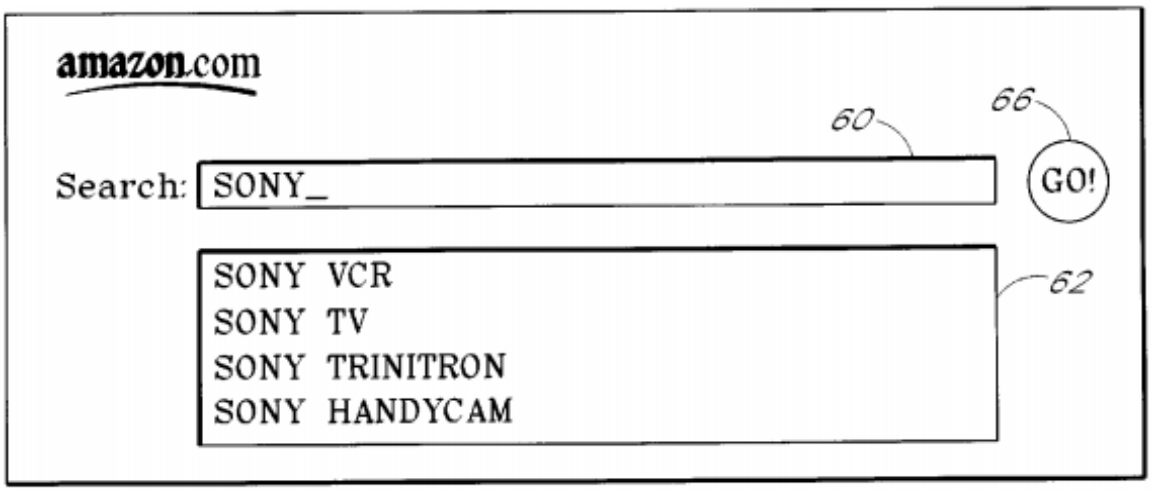


Figure 1.8: Excerpt from the original autocomplete patent; relevant items appear beneath the search bar before the query is even spelled out.

1.2.2.1 From static to dynamic queries

Queries used to be considered static objects by search engines - everything happening before submission by the user would be invisible to the algorithm. With technologies such as AJAX in the early 2000s enabling dynamic behavior for web pages, one of the most important advances for text-based search engines in recent years is the emergence of dynamic and automatic completion - commonly referred to as "autocomplete" [REO03]. Based on several factors such as popularity or location, a scrolling menu below the search bar displays several options ranked by relevance (Figure 1.8), and incrementally updates the ranking at every character entry or deletion. Even if the user does not have a clear sense of what to look for or how to spell it, autocomplete will most likely provide them with the necessary help. Recent developments for autocompletion include the display of rich information along the queries, as well as direct access to web pages.

According to Google, this feature reduced typing by 25 %, which amounts to 200 hours of typing saved each day [Sul18]; to mobile device users this is particularly beneficial. Search started as a task the user would perform all the way to completion, thinking ahead before before submitting; it has now become a rapid-fire tool that can be pulled out at any given moment, to assist the user on the fly in tight situations.

1.2.2.2 From tags to content queries

The other exciting development for search in recent years is the emergence of content-based querying. Text is still by far the most dominant search modality; for text retrieval, but also across all types of media, notably images or videos. This relies for the most part on keywords, or text data pertaining to the piece of media - but not the content itself. This severely limits retrieval capabilities in those situations; even if a database is well tagged, which can be prohibitively time-consuming, visual or audio data can never truly be captured by a few keywords or a text description.

This limitation is the motivation behind content-based retrieval methods. Shazam, launched in 2002, is a popular example of service built around such a method: the

Figure 1.9: Examples from Bing Visual Search. A picture of a kitchen island and a wallaby (left) serve as queries; for the wallaby the species is correct on every search result. Image credits: Hu et al.

users directly use their mobile devices' audio input to search, in real time, for a song heard in the background (e.g. in public spaces or during commercials). Google Image Search, TinEye and Bing [HWY+18] provide powerful reverse image search features that allow the user to submit their own pictures and find ones that are visually similar. The versatility of this tool is quite overlooked: by retrieving, this can identify objects, places, people, fraudulent postings - often down to very fine-grained details (Figure 1.9).

Outside of publicly available tools, content-based retrieval is already deployed in a multitude of scenarios. Youtube, the largest video content platform, systematically reviews the audio in user-submitted content (Figure 1.10), searching for unauthorized use of copyrighted songs - this is referred to as ContentID. Twitch currently uses a similar procedure for recordings of livestreams, automatically cutting out the audio in parts featuring DMCA-protected songs in the background. Uber uses a technique called Locality-Sensitive Hashing [IM98] in order to retrieve fraudulent trips. Digimarc, Ivitec are other lesser known actors in the content control space. Although the inner workings of these techniques were never made public, monitoring and recognizing content on this scale inevitably requires quickly sifting through very large video databases in an automatic manner.

1.2.2.3 Potential clinical applications

Dynamic behavior and rich content understanding are two pillars of modern search, making it a highly versatile and expressive tool for investigating content. The surgeon, the surgical staff or the personnel in charge of the OR control tower mentioned in 1.2.1.2 could greatly benefit from the ability to rapidly access and explore a large database of surgical recordings. For reasons mentioned in 1.1.3, this process should

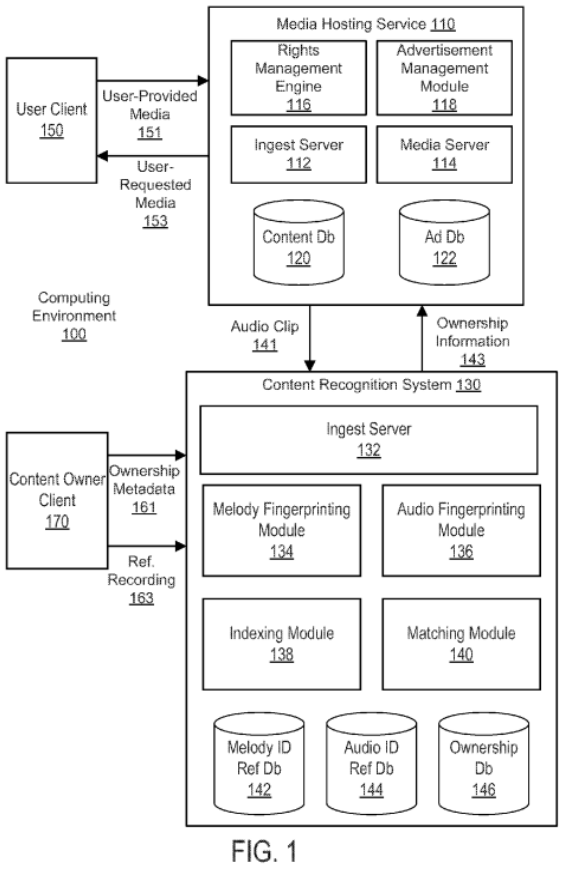


Figure 1.10: Patent from Google for automatic melody identification from user-submitted audio for copyright management purposes.

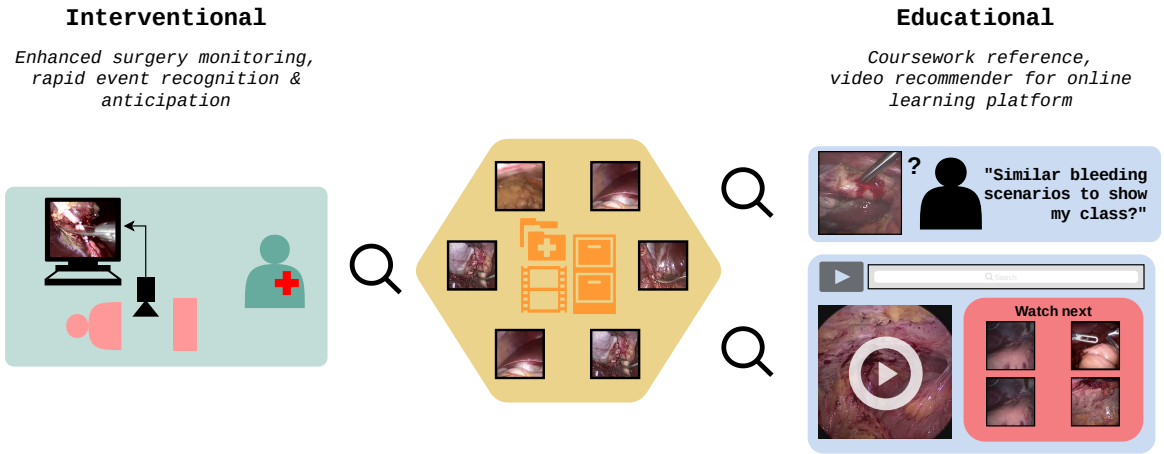


Figure 1.11: The place of video search in the clinical space. Besides logistical aspects such as indexing, two main branches of applications can be envisioned: educational and interventional, with the latter requiring real-time operability.

interfere as little as possible with the actual surgical task at hand. One way to achieve this is to directly plug live video feeds from the OR into a content-based retrieval algorithm, thus removing the need for the surgeon to interrupt their course of action to formulate queries. This "surgical video recommender system" would act as a dynamic and responsive reference tool, fit not only for surgical education or indexing, but even for intraoperative decision support (Figure 1.11). As it is the case for current reverse image search services, this tool may be used as an always accessible, all-purpose recognition utility, presenting all the information attached to the retrieved videos. Depending on the system's performance, this information could range from broad - e.g. surgical phase - to very fine-grained, such as critical events or surgical actions.

1.3 Data

Clinical application ideas suggested in the previous section require powerful surgical activity understanding algorithms; the data at our disposal for developing those is presented in the following section.

1.3.1 Laparoscopic video data collection

In conventional or open surgery, the tissue covering the surgical site is cut open with an incision large enough to provide the surgeon full unobstructed view, as well as direct access for the hands and surgical instruments. For instance, the incision for open cholecystectomy is usually at least 20cm long [BMS+11]. In contrast, laparoscopic surgery relies on one or several very small (less than 3 cm) incisions of the abdominal wall called *ports*, fitted with trocars. After general anesthesia, a preliminary phase establishes the *pneumoperitoneum*: the abdominal cavity is insufflated with carbon dioxyde in order to create space for the instruments, which are then inserted through the trocars. Two main types of instruments are employed: mechanical (graspers, staplers, scissors ...) and energy-based (hook, bipolar grasper, monopolar cautery devices), with the latter requiring a dedicated power generator in order to deliver the heat necessary for dissection and cauterization.

A laparoscope - an optical fiber imaging device - is inserted as well, showing the anatomy and the instruments to the surgeon via a separate monitor. While a wide variety of signals can be collected from those procedures - such as RFID tracking data or energy-based tool usage [PSMB12, MLCH16], the laparoscopic video feed stands out as the richest and most comprehensive by far. For this reason we chose it as our source of data for studying surgical workflow.

Data collection has been made possible for this research by a joint effort from the Image-Guided Surgery Institute of Strasbourg (IHU Strasbourg), the Research Institute against Cancers of the Digestive Tract (IRCAD) and the Nouvel Hôpital Civil de Strasbourg (NHC). Surgeries were conducted at the NHC; the IRCAD, an international research and training institute for minimally invasive surgery, collected video recordings of those between 2015 and 2020 for research purposes. Researchers from team CAMMA and hepato-biliary surgeons affiliated with IHU Strasbourg selected videos and applied various types of annotations to generate several anonymized datasets for surgical workflow understanding.

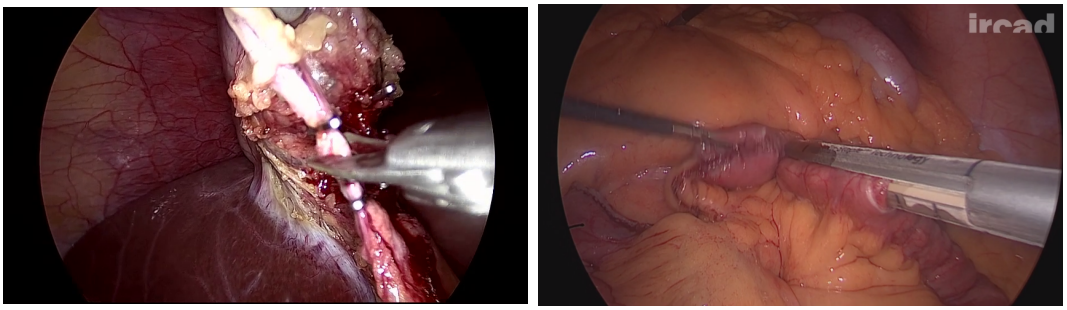


Figure 1.12: Left: Cutting of the cystic duct in cholecystectomy, to release the gallbladder. Right: Jejunojejunal anastomosis in bypass. The intestine is stapled to itself, closing the digestive path coming from the stomach.

1.3.2 Cholec120

Phase	Duration (min)
Preparation	1.8 ± 1.7
Calot triangle dissection	15.6 ± 11.1
Clipping and cutting	2.9 ± 2.1
Gallbladder dissection	12.2 ± 8.9
Gallbladder packaging	1.6 ± 0.8
Cleaning and coagulation	3.0 ± 2.6
Gallbladder retraction	1.4 ± 1.2

Table 1.1: Phases in cholecystectomy

Cholecystectomy (Figure 1.12) is a procedure prescribed in cases of chronic or acute cholecystitis. Out of all surgical procedures performed laparoscopically, this one is the most common with over 700000 interventions annually in the United States [cho93], and has now established itself as the gold standard over open cholecystectomy. It consists in removing the patient's gallbladder, the organ reponsible for storing bile produced by the liver. To study its workflow, a dataset of 120 cholecystectomy videos annotated with phases named *cholec120* was previously built by team CAMMA; a subset of it named *cholec80* is publicly available [Twi17], carrying surgical instrument presence annotations as well. A smaller subset of 40 videos, *cholec-T40* also comes with fine-grained action triplet annotations [NGY+20]. Table 1.1 gives a breakdown of the seven phases of the surgery in *cholec120*, which obey a simple and mostly linear workflow.

1.3.3 Bypass40

Among bariatric surgical procedures, i.e. weight loss surgeries, Roux-en-Y gastric bypass (Figure 1.12) or simply gastric bypass is the most common. It is sometimes prescribed in cases of morbid obesity, in order to facilitate weight loss by reducing the patient's stomach. It does so by creating a gastric pouch and connecting it to the small intestine in a way that, as the name suggests, bypasses the rest of the stomach, which

Phase	Duration (min)
Preparation	7.1 ± 2.9
Gastric pouch creation	25.6± 8.1
Omentum division	23.1± 5.5
Gastrojejunal anastomosis	3.3 ± 0.6
Anastomosis test	2.8 ± 0.8
Jejunal separation	3.1 ± 0.6
Closure of Petersen space	7.4 ± 2.2
Jejunojejunal anastomosis	23.9± 7.6
Closure of mesenteric defect	9.2 ± 2.1
Cleaning and coagulation	4.2 ± 1.0
Disassembling	5.9 ± 1.4

Table 1.2: Phases in gastric bypass

is then discarded. This connection process is called anastomosis, and is reported as one of the critical events mentioned in Section 1.3.4. Compared to cholecystectomy, the workflow of this procedure is more complex, and less linear. In order to study it, the *Bypass40* dataset [RDG+21] was created, with in it 40 complete recordings of gastric bypass. We provide a breakdown of *Bypass40*'s surgical phase statistics in Table 1.2.

1.3.4 CEV64

Event type	Count	Duration (s)
Abdominal access	185	71 ± 22
Mesh placement	98	228± 45
Approximating	102	371± 59
Sealing	34	63 ± 10
Out of body	516	177± 41
Anastomosing	73	602± 142
Dividing	96	72 ± 23
Incising	69	149± 28
Bleeding	116	68 ± 19
Idle	87	164± 45

Table 1.3: Overview of surgical critical events

Laparoscopic surgeries are highly diverse; just the two examples listed above are characterized by vastly different workflows, surgical instruments and anatomical sites. However, certain crucial traits are shared across workflows of many surgeries. Two GI surgeons from IHU Strasbourg collaborated to define a common set of surgical situations called **critical events** (Figure 1.13), found in 6 different types of laparoscopic procedures: cholecystectomy and gastric bypass, as well as eventration, hernia, nissen and sigmoidectomy. This led to the *CEV64* dataset, generated during this thesis. Beginning and end for each event were marked in 64 videos, which helped generate a

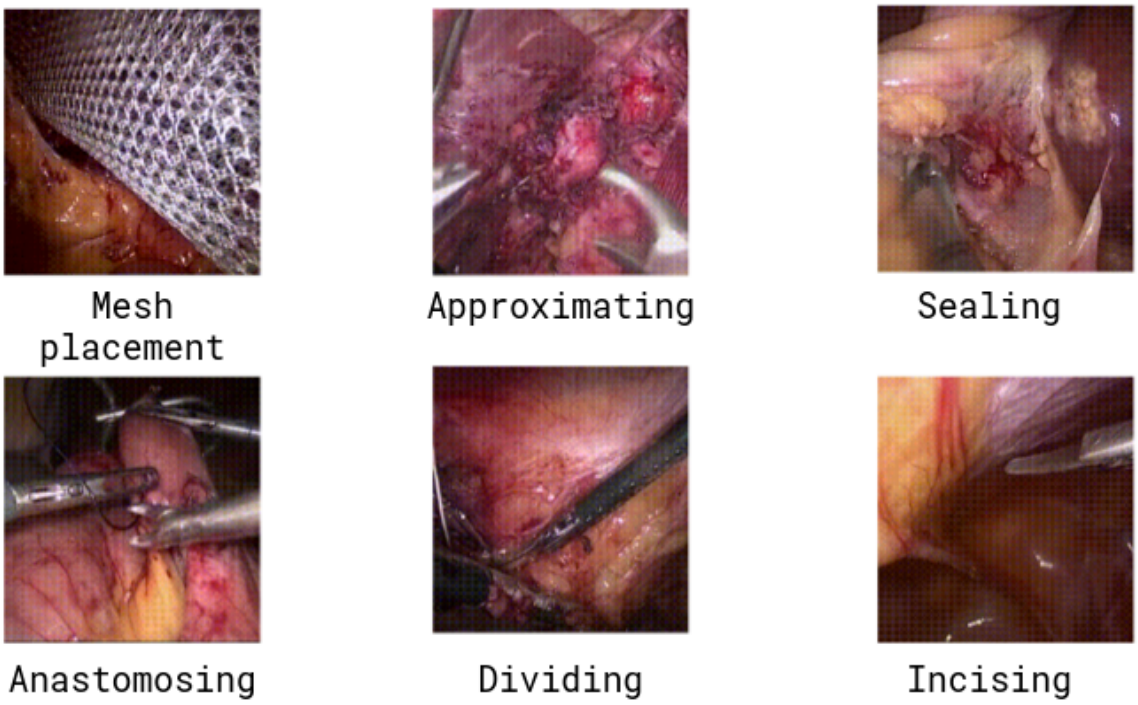


Figure 1.13: Examples of critical events in surgery, from the CEV64 dataset.

database of 20000 surgical video clips of approximately 30 seconds each. Annotations were applied using MOSaiC, an internally developed tool with dedicated features for temporal annotation. The 11 reported events are detailed in table 1.3.

1.3.5 The gap with large-scale video datasets

Video datasets employed within the rest of the computer vision community present major differences with the ones just mentioned. This is often a concern when adapting methods trained on those sources to surgical content. A comparison with a few prominent datasets is shown in table 1.4.

Event	Average video duration	Number of videos
cholec120	2280 ± 841	120
bypass40	6600 ± 1802	40
cev64	6911 ± 2008	64
ActivityNet	117 ± 67	18000
FCVID	134 ± 92	91000
Kinetics400	10 ± 0	240618

Table 1.4: Comparison of surgical video datasets against generic activity datasets

The first major difference is duration; generic activity videos tend to be a lot shorter, sometimes lasting only a few seconds. This adds convenience for training; the content is a lot simpler and visually distinct as well. Surgeries often exceed feature-length films in terms of length, with many phases, events and actions taking place over

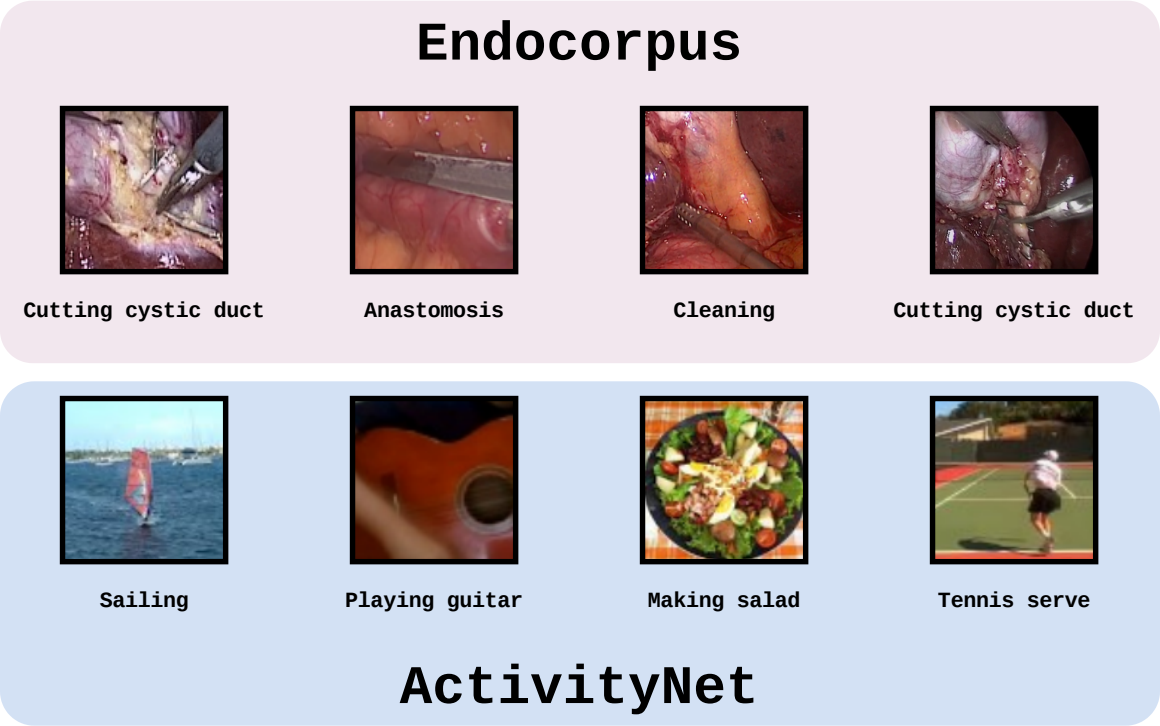


Figure 1.14: Frames from Endocorpus and ActivityNet. Note the difference in visual diversity and complexity.

their 140-minute average span. Processing them in a way that accounts for the entire workflow is a difficult challenge.

Accurately annotating this type of data requires highly specific clinical knowledge; which is why in our case this was done by experienced gastrointestinal or hepato-biliary surgeons and surgical fellows. As a result the number of available annotated videos is quite low, only reaching 204 for *cholec80*, *bypass40* and *CEV64* combined.

Finally, the appearance of laparoscopic video feeds is challenging in itself, as evidenced by Figure 1.14. Scenes from inside the abdominal cavity tend to have a high degree of similarity with each other. The field of view of 62° [WKK+20] is quite narrow; oftentimes relevant anatomy or instruments exit the frame. Visual landmarks for specific phases or events can be very sparse: for example cutting the cystic duct in cholecystectomy is often brief, with scissors appearing in the frame only for a few seconds. Recording quality is routinely affected by smoke, blood splatters and jittery motion as well as inconsistent lighting.

1.3.6 The big picture: Endocorpus

Endocorpus is the CAMMA team's first very-large-scale video dataset, containing the previously mentioned surgical datasets as well. In total, 1558 full-length recordings of surgeries are present, for a combined runtime of 3700 hours. 12 types of surgical procedure are featured; two of those, robotic sleeve gastrectomy and robotic bypass, involve Intuitive Surgical's Da Vinci. As such, this dataset therefore covers most of abdominal endoscopy, containing an amount of experience equivalent to several surgeon lifespans. Videos last on average 140 min, with major discrepancies between

Surgery	Description	Count	Duration (h)
Cholecystectomy	Removal of gallbladder	518	1.4 ± 0.4
Bypass	Stomach reduction	388	1.8 ± 0.5
Hernia	Inguinal hernia repair	341	2.1 ± 0.3
Eventration	Diaphragmatic eventration repair	133	1.9 ± 0.5
Bypass robot	Robot-assisted stomach reduction	125	2.0 ± 0.5
Sigmoidectomy	Colon removal	118	1.2 ± 0.4
Sleeve gastrectomy	Stomach reduction	90	1.7 ± 0.4
Nissen	Esophageal sphincter reinforcement	87	1.9 ± 0.3
Adrenalectomy	Adrenal glad removal	25	1.9 ± 0.4
Hepatic surgery	Liver-related interventions	17	2.4 ± 0.6
Pancreatic surgery	Pancreas-related interventions	12	5.3 ± 1.1
Sleeve robot	Robot assisted stomach reduction	11	2.4 ± 0.8

Table 1.5: Overview of Endocorpus

the various types of surgery. We provide a detailed breakdown in Table 1.5.

Unless the videos belong to one of the previously mentioned subsets, no annotations other than the surgery types are available. The work presented in this thesis is the first attempt to exploit it and explore its contents; even without annotations, the scale of Endocorpus makes it an appropriate target for unsupervised video retrieval.

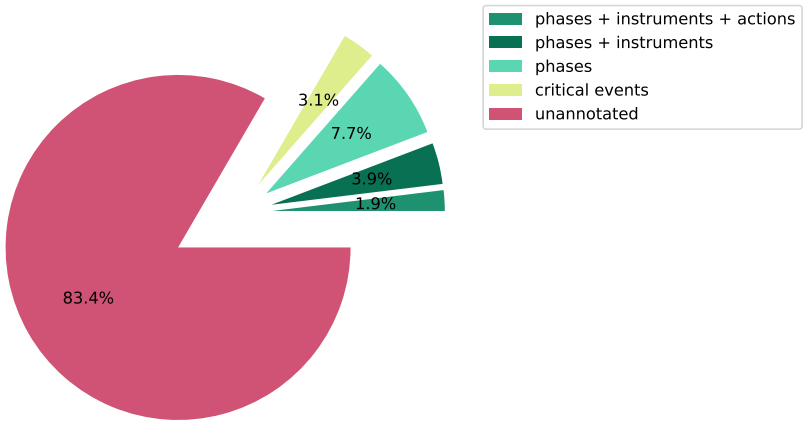


Figure 1.15: Current state of the Endocorpus dataset. Annotations are only available for a small fraction of it - barely over 15%.

1.4 Contributions

Figure 1.15 sets the stage for the work of this thesis, overwhelmingly dominated by unannotated data. Collecting, selecting and storing endoscopic video data into datasets is relatively easy compared to having human experts annotate it, which explains the disproportionate 83.4 % of unlabelled videos. This situation suggests that in future developments of Surgical Data Science, overreliance on full supervision with

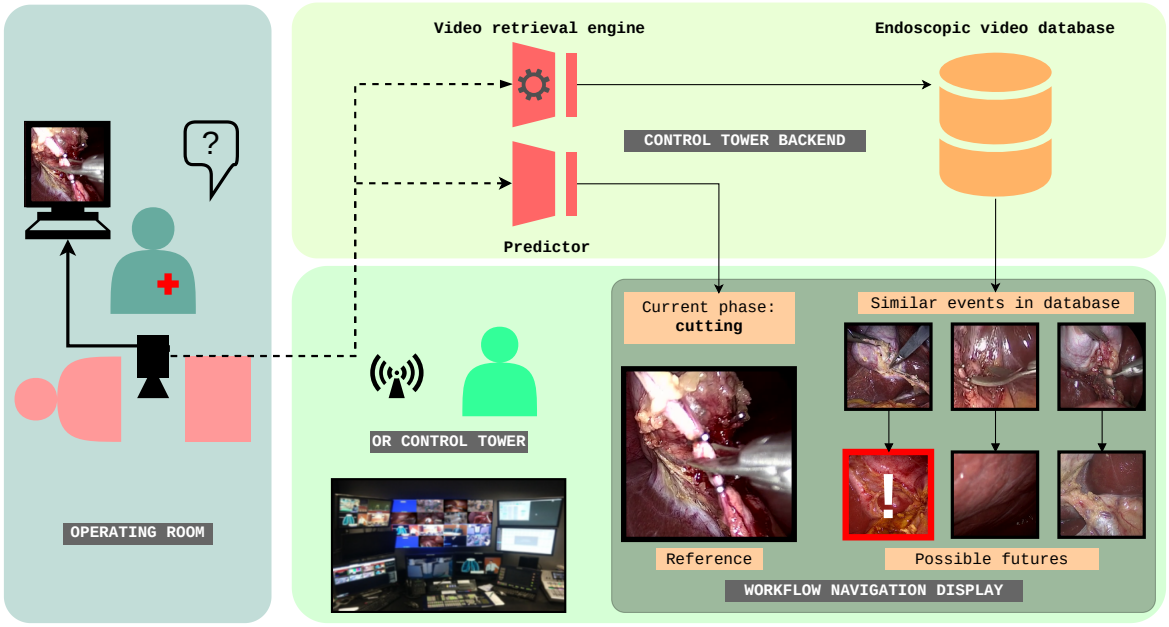


Figure 1.16: OR control tower concept, exploiting surgical video retrieval. An unfamiliar situation prompts the surgeon to request assistance. Control room staff is shown, in real time, the surgical phase and a collection of visually similar clips as reference, with corresponding future outcomes - one of which being a case of bile duct injury. In addition to the control room, the workflow navigation display may be redirected for other purposes: for example, an online classroom for surgical training.

expert-annotated data might be ill-advised: assuming surgical video data collection is to expand and become more systematic, ramping up manual annotations accordingly would come at an excessive cost, while leaving large swaths of data underused and underexplored would be a waste.

Our goal is precisely to overturn this situation: we aim for semi- or self-supervised methods in order to avoid the dependency on manual labels, while providing the means to take advantage of the unannotated data and navigate it in a way that serves real-time, context-aware support in the OR. We propose two main paths for achieving this: the first is automatic annotation, an obvious alternative to expert-generated labels that is far more scalable; this is done for surgical phases, to provide a broad and standardized understanding of the context. The second is search, in the form of unsupervised video retrieval, capable of instantly delivering a richer and more expressive description of a captured surgical scene. The final vision for the innovations we propose in terms of context awareness is summed up in Figure 1.16, in the form of a control tower for the OR powered by large amounts of unlabelled data.

Listing them in detail, our first contribution is a method that leverages the unannotated data through automatic annotation of surgical phases, with a combination of strong offline predictor and weak real time predictor. This new semi-supervised approach is a breakaway not only from traditional phase recognition [TSM+16, JDC+18, JLD+19] with full supervision, but also from the other semi-supervised methods in this space [YMMP18, FJM+18] since it enriches the unannotated data instead of only using it for pretraining.

Our second contribution is a new method for a problem never tackled before: con-

tinuous video retrieval in real time. As opposed to previous video retrieval approaches [ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19], ours is the very first tailored for live video sources; as such, we evaluate it with a new protocol reflecting livestreaming conditions. In the absence, at the time, of an adequate very-large-scale surgical dataset, and also in the absence of a method built for this specific task in the general vision community, we first demonstrated our approach on two generic activity datasets.

Our third contribution is the application of continuous video retrieval to a large, more complex video dataset of surgical recordings. This is the first study of its kind in terms of scale, surpassing the few works on surgical video retrieval [DQL+14, PS18] by a large margin in that regard. We show that content relevant to an ongoing surgical procedure from a massive database can be retrieved in real time. We also demonstrate the all-purpose ability of video retrieval, by using the same unsupervised model to return search results relevant for critical events in six types of procedures, as well as for phases in both cholecystectomy and bypass.

Our fourth contribution extends the previous retrieval method by introducing uncertainty awareness for video hashes: by adding compressed uncertain bit patterns to the database, the relevance of retrieval results is improved, while keeping additional space and time expenses to a minimum.

Finally, our last contribution is the generation of large-scale and very-large-scale datasets for activity recognition in surgery. In particular, our critical event dataset *CEV64* is the very first of its kind.

1.5 Outline

The work presented in this thesis is organized according to the following outline:

- Chapter 2 provides a structured overview of previous publications relevant to our work.
- Chapter 3 demonstrates a semi-supervised method for surgical phase recognition relying on automatic annotation, using unlabeled surgery videos.
- Chapter 4 lays down the theoretical foundations for learning hash functions for content-based retrieval.
- Chapter 5 presents a methods for content-based video retrieval in real time, on generic activity large datasets from the general computer vision community.
- Chapter 6 brings the content-based video retrieval task to a very-large-scale surgical dataset and extends the previous methods by managing the uncertainty of database entries in a lightweight fashion.

CHAPTER 2

Related Work

"Beware of finding what you're looking for."
- Richard Hamming

Contents

2.1 Video Activity Understanding

2.1.1 Model architectures & general activity recognition

Video as a data format is particularly challenging to process in recognition tasks. High dimensionality already is a concern when it comes to static images; for a typical $[224, 224, 3]$ RGB image used in CNN models the number of dimensions is 150528. For a single second of video at 25 frames per second with the same frame size, this number goes up to 3763200. This is without even considering the presence of an audio track; multimodal data is a rich and active research area of its own, which we will not go into here.

In addition to the high dimensionality of videos, the temporal information is a major source of difficulty as well. Just like the words in a sentence, the succession between frames matters, and determines motion and event logic in the video. An adequate video model should ideally learn those temporal dependencies in order to correctly make predictions.

2.1.1.1 Extraction - aggregation approaches

Early approaches have attempted to aggregate 2D static convolutional neural networks for video data, applying them to each frame separately. Karpathy et al. [KTS+14] presented a few elementary aggregation possibilities through pooling. A two-stream approach was proposed by [SZ14], with, in parallel, one CNN processing RGB pixel data and another procesing the optical flow. [WXW+16] reuses the two-stream architecture with snippets sparsely sampled across the video. In the examples above, there is no sequence modeling; the aggregation is not sensitive to the order between the frames.

Recurrent Neural Networks (RNN) are a type of DNN dedicated to sequential inputs. A function commonly referred to as the recurrent cell or recurrent unit is called at each timestep of the input, forwarding a state value taken into account during the next call. This enables sharing parameters across timesteps, as well as receiving inputs of arbitrary length. The recurrent unit from the original RNN, proposed in [RHW86], was prone to issues, namely vanishing and exploding gradients. To counteract those, [HS97] introduced the Long Short-Term Memory unit, which made RNNs viable for training on longer sequences. [GS05] later proposed the bidirectional LSTM recurrent neural network. With two LSTM-RNNs processing the input sequence in opposite orders, both types of temporal dependencies can be captured: present to past and present to future.

The idea of combining this architecture with CNNs for video tasks was introduced by [DHG+15]. The CNN maps video frames to visual feature vectors, which the LSTM then processes as a sequence, learning the temporal dependencies within it. End-to-end [YMMP18] training of the two components is possible, however with only a very limited small number of frames at each training iteration due to hardware constraints. Very long videos can be downsampled and have their frames extracted separately to facilitate training [TSM+16].

The TCN [LVRH16], or Temporal Convolutional Network, is a faster alternative to the LSTM. Its architecture featuring stacked convolutions is inspired by 2D CNNs,

except with 1D temporal convolution kernels.

2.1.1.2 3D CNN

[TBF+15] first introduced the concept of 3D Convnets. The architecture, stacking convolutions, ReLu activations and max pooling operations, is similar to ordinary 2D CNNs. The convolution operations, however, are performed using 3D 3x3x3 kernels over height, width and time. As such, it is capable of learning 3D features from video clips. [QYM17] brought the idea of residual connections first found in ResNet [HZRS16] to 3D CNNs. [CZ17] used a pretrained Inception model, and expanded its convolution kernels in the time dimension to make them video-capable. The authors also offered a two-stream version of their model, with optical flow and RGB data. In [WGGH18], a non-local 3D Resnet is presented; it is capable of learning long-range dependencies that would be overlooked by ordinary 3D CNNs due to their limited recptive field. [Fei20] refined the process for expanding a Convnet from 2D to 3D. The authors of [CZ17] explored options for mixing 2D and 3D convolutions within the same model, resulting in a more efficient and accurate model called S3D. Using a feature aggregation approach, similarly to the ones previously mentioned, [GCDZ19] used I3D as a feature extractor; the aggregation layer employed is a Transformer model [VSP+17]. This recently developed sequence model relies on a scaled dot product attention mechanism to inspect relationships between any two items in the input sequence. The main benchmarks used for publications with 3D CNNs were Kinetics [CZ17] and UCF-101 [SZS12], with action recognition as the task.

2.1.2 Early activity recognition

In ordinary action recognition, only the output of the model from a complete video observed all the way to the end is considered. This ensures the model takes all the visual information available into account; however this may not be a realistic use case, especially if it is meant to be used in real time. The task of early action recognition (EAR) works with this constraint in mind; the model is asked for a prediction at various stages of progress throughout the video. This far increases the task's difficulty, especially at the very beginning of the video since the available visual information is very minimal.

Several methods [ASS+17, HZM+19, MSS16] proceed by incorporating elapsed time or progression to inform the training process. [ASS+17] trains an LSTM model with a temporally weighted loss enhancing the importance of predictions made early on in the video. [HZM+19] proposes a method for early recognition based on soft labels at different levels of video progress.

A different type of approach for early recognition tasks is to synthesize the content in the video's future: [RFL19] attempts to generate future video frames using a DCGAN [RMC16] model. [GDSF19] employs video representations fed to generator models trained to synthesize future frame embeddings.

Teacher-student distillation methods train a student model that has not seen future frames to replicate representations learnt by a teacher model that has. [KTF17] applies a similar principle to representations from a 3D CNN. A different method introduced in [WHL+19] uses another distillation approach: it involves training a forward-directional LSTM to match at each timestep the hidden states of a trained bidirectional LSTM.

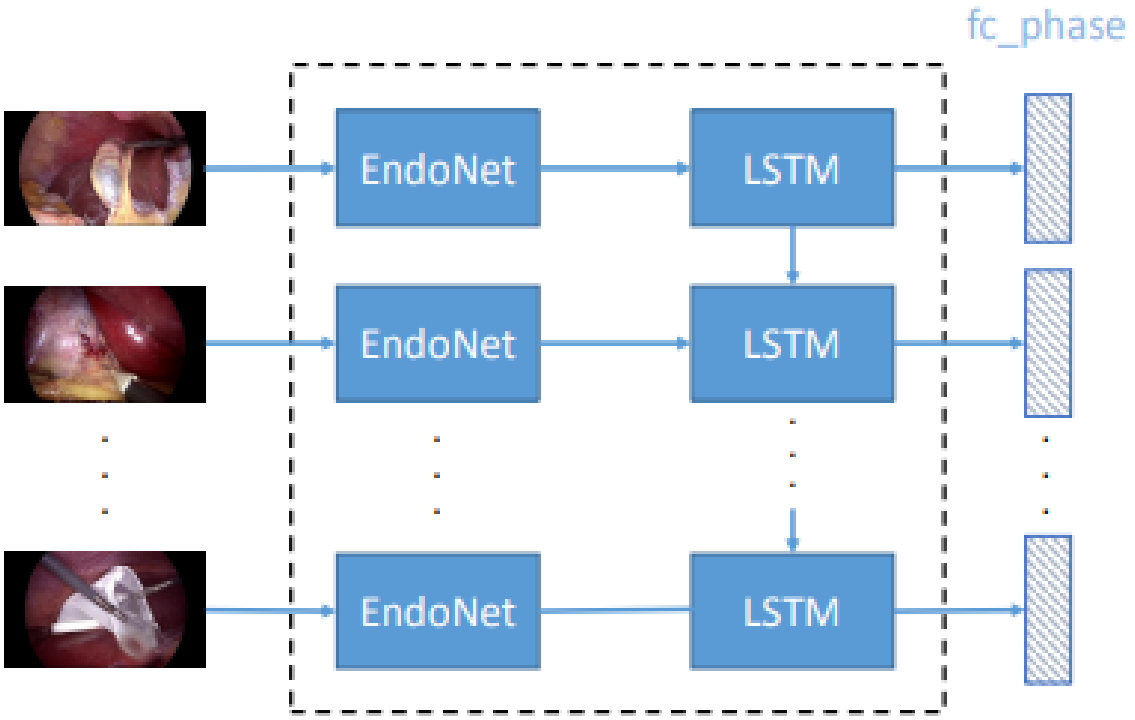


Figure 2.1: Endonet CNN-LSTM model for surgical phase recognition on videos. Courtesy of Twinanda et al.

An approach incorporating reinforcement learning is presented in [WJZY20]. Here an agent applies a mask to the output probabilities, allowing or forbidding classes depending on the time.

2.1.3 Methods in surgical video analysis

The problems studied in the field of Surgical Activity Understanding include recognizing surgical phases, tool usage, events, actions and timings in surgical procedures.

Before the introduction of deep learning to the field, methods mostly relied on traditional ML models processing handcrafted features, sometimes combined with sensor data indicating, for instance, surgical tool usage [BFN10]. However, DNN-based models have taken center stage for the most part, replacing former video analysis methods as it did in many other domains.

Surgical phases form the foundation for analyzing workflows in surgery. Lalys et al. [LJ13] define them as the first level of granularity for decomposing surgical workflows. Deep vision models trained to recognize them were first introduced by Dergachyova et al. [DBH+16] as well as Twinanda et al. in 2016 [TSM+16]. Twinanda et al.'s EndoNet incorporated a Resnet18 CNN used as a feature extractor followed by a high-level classifier, either a Support Vector Machine or a Hidden Markov Model. Training was performed in a multi-task manner, with both surgical instrument and phase recognition as learning objectives. Results were shown on the cholec80 dataset, which was introduced in this work as well. Its successor EndoLSTM [TMM+16, YMMP18] replaced the previous high-level classifier layer with an LSTM. SVRCNet [JDC+18] uses a similar architecture combining Resnet and LSTM, with an added Prior Knowledge

Inference to post-process predictions. Al Hajj et al. [HLC+18] improved on the CNN-RNN combination using a boosting strategy. MTRCNet-CL [JLD+19] exploits the correlation between tool usage and surgical phase; for instance the specimen bag only appears in later phases, as the gallbladder is placed there only after it is dissected. The correlation loss term attempts to teach the model this type of relationship. Czempiel et al.'s TeCNO [CPK+20] were the first to introduce temporal convolutional networks to surgical phase recognition. All the works mentioned so far using deep neural networks for phase recognition did so on cholecystectomy videos, which has a fairly simple workflow. Ramesh et al.'s MTMS-TCN [RDG+21] was the first to bring phase recognition to gastric bypass; a much more complex and non-linear type of surgery with many phases themselves composed of multiple smaller steps.

2.2 Alternatives to full label supervision

The canonical setup for training deep neural networks involves a large, fully annotated dataset, with the full ground truth available for the task to train the model for. This section presents alternatives to this fully supervised setup.

2.2.1 Weak and semi supervision

Weakly supervised learning relies on supervision signals containing less information than actual labels corresponding to the actual task to perform; when those signals are much easier to procure than fully fleshed out annotations, this type of method is particularly adequate.

In videos, a common weakly supervised learning task is action localization. In contrast to video-level action recognition done on benchmarks such as Kinetics, where the model assigns a prediction to an entire video clip, action localization involves finding, within a video, frames where the action actually takes place. Achieving this without any frame-level annotations, with only video-level tags as the weak supervision source, is a challenging task attempted in several works [RKG17, WXLG17, PRRC18, NRF19, ASNS20, MGVY21, BLG+15].

Outside of videos, another form of weak supervision involves noisy labels. Mislabeled data, while non-existent in major computer vision benchmarks such as ImageNet, can be a threat in real-life situations, with annotations from unreliable public sources. Patrini et al. [PNNC16] performed a theoretical study of label noise effect on common loss functions. [XXY+15, NLX15] both leverage large amounts of web-sourced images for image classification. Cheng et al. [CZZ+20] used an auxiliary model called Side Information Network to estimate the correctness of images with label noise.

In semi-supervised learning methods, a small amount of annotated training data is available, as well as a certain amount of completely unannotated data. Those methods are appropriate in contexts where data is abundant, but annotations are scarce due to time constraints or costs.

Rosenberg et al. [RHS05] presented a self-training method for object detection. Dai et al.'s [DG13] method builds prototype sets from the annotated data points, trains classifiers on each set then uses them as an ensemble. Sajjadi et al. [SJT16] introduced a loss function enforcing robustness to random transformations on the

unlabeled data. Radosavovic et al. [RDG+18] labeled the unannotated images with ensembled predictions on transformed duplicates. Iscen et al. [ITAC19] used graph methods to propagate labels to the unannotated part of the dataset.

2.2.2 Unsupervised representation learning

Unsupervised learning or self-supervised learning methods work from a completely unannotated training set. The condition for such a method's success is to have the model understand the underlying organization of the data purely based on its content, learning a representation that accounts for this organization. The training task chosen to teach this representation is referred to in the literature as the *proxy* or *pretext* task.

We present a few methods for static image data first. Noroozi et al. [NF16] introduced the jigsaw pretext task, consisting in identifying a permutation applied to the tiles of an image. Another work [NPF17] from the same authors trained a model to count visual primitives in images without labels, using a visual consistency rule enforced across tiles in the image. Zhang et al. [ZIE16] trained an autoencoder on images converted to grayscale, to reconstruct the original color images. BYOL or Bootstrap Your Own Latent was developed by Grill et al. [GSA+20]; in this method the model has to predict its own representation of a transformed image using the original. SimCLR introduced by Chen et al. [CKNH20] combined multiple data augmentation methods with a contrastive cross-entropy.

For video data, Srivastava et al. [SMS15] trained an LSTM autoencoder to learn video representations. Wang et al. [uns] used a handcrafted method to define image patches for the network to track across video frames. In Misra et al.'s work [MZH16], the model is trained to validate the temporal order of a sequence of frames. Vondrick et al. [VSF+18] introduced a video colorization task; unlike the original image colorization task [ZIE16], the image to colorize is not the input but a further frame in the video.

2.2.3 Distillation by teacher supervision

Distillation approaches, for the most part, are not conceived with data scarcity issues in mind like the approaches mentioned previously. The objective for most of those methods is model compression, i.e. transferring the behavior of a large-capacity model to another one, with fewer parameters and lower hardware requirements. The large model in this framework is referred to as the teacher, while the smaller one is called the student. Wang [WY21] as well as Gou et al [GYMT21].

Outside of model compression, a few publications have used similar distillation concepts for other purposes; for example as a form of weakly supervised learning, or to condition specific behaviors from the student model. In Radosavovic et al's work [RDG+18], distillation is used in a semi-supervised setup, performed from the ensemble to the student on the unlabeled data. The role of distillation in Wang's work [WHL+19] on early action recognition is to enforce anticipation for the student.

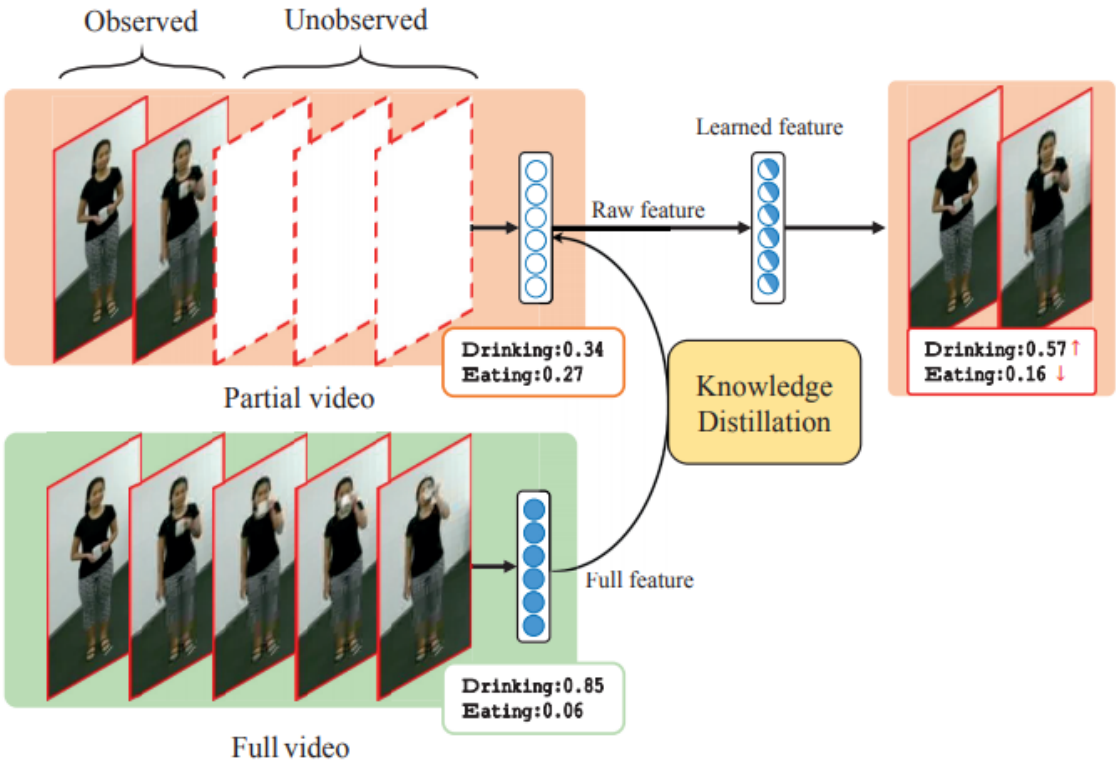


Figure 2.2: Distillation for early action recognition. The student model, handling partial videos, is trained to copy the representation of the teacher that has knowledge of the full videos. Image credits: Wang et al.

2.2.4 Uses in medical context

A common obstacle in the fully supervised training of DNNs is the limited amount of annotated data available. This is especially true for surgical videos, which are difficult to acquire, and require expert medical knowledge for annotations. For surgical phase recognition under semi supervision, Bodenstedt et al. [BWK+17] used a frame sorting task on unannotated videos. Funke et al. [FJM+18] employed a method called second order temporal coherence to pretrain the visual feature extractor in their model, enforcing distance constraints in embedding space that reflect the temporal distance between frames of the same video. Their experiments used the cholec80 dataset for cholecystectomy phase recognition; with 60 total videos available for training, annotations were limited to 60, 40 and 20 videos. In Yengera et al.'s [YMMP18] work, remaining time and progress regression were used as unsupervised pretraining tasks. From the cholec120 dataset, 8 to 80 annotated videos were used for phase recognition training.

Weak supervision has been employed for surgical instrument detection on cholec80, using binary presence annotations only [NMMP19, VMMP18]. A form of distillation was shown in Kannan et al.'s work [KYM+20], to encourage anticipation in real-time surgery type recognition tasks.

2.3 Retrieval

The task of retrieval is, at its core, the task of learning representations that are adequate for searching. We break down the corresponding methods into two main types based on the representations used; then we examine a few uses of hashing in the medical community.

2.3.1 Embeddings

Embeddings are a byproduct of any DNN training process (see 4.2). Those compact vector representations are trained to incorporate rich content and to have good geometrical properties; it is therefore natural to use them for retrieval.

2.3.1.1 Image retrieval

For image retrieval, Hoang et al. [HDTC17] proposed a scheme for generating embeddings for retrieval, by selecting and aggregating convolutional features from a VGG model. Jimenez et al. [JAiN17] used class activation maps to add weight to more discriminative features in their embedding construction process. Liu et al. [LYV+19] used Graph Convolutional Networks to compute a similarity score between images, serving as their basis for learning seachable embeddings. In Revaud et al.'s work [RARS19], mAP, the metric generally used for retrieval evaluation, is approximated by a differentiable function and learnt by their model.

2.3.1.2 Video retrieval

Content-based video retrieval was performed by [XYH15] with features extracted from a VGG CNN pretrained on ILSVRC 2014. The features were pooled using either

Fisher vectors or VLAD. Experiments using CNN embeddings were also performed by Kletz et al. [KLS18], and showed that those yielded higher retrieval performance than handcrafted features.

2.3.2 Hashing

The previous principle of looking for compact and discriminative representations of large data is taken by hashing to an extreme level: instead of floating-point vectors with a few thousand components, hashing looks for representations in the form of binary arrays with only a few hundred entries. We present a few methods for static images first, followed by methods for video content.

2.3.2.1 Image hashing

2.3.2.1.1 Approaches outside of deep learning

A number of methods were originally developed without deep neural networks, although the hash functions described can be applied to features extracted by DNNs.

Locality-Sensitive Hashing [AI08] is a highly popular algorithm used in a wide variety of application scenarios, including genome sequencing [BKC+14] and fraud detection [NCB17]. This method however is completely data- independent, using random projections to build the hash function. Since then, a number of methods have attempted to tailor this function to the data to hash. Weiss et al. introduced Spectral Hashing [WTF08], a method based on a PCA of the data. Iterative Quantization[GLGP13], discovered by Gong et al., relies on a rotation of the data that minimizes the quantization error. Spherical Hashing[HLH+12] uses binarization functions with spherical separation surfaces instead of hyperplanes, and optimizes the position of the sphere centers, referred to as pivots.

2.3.2.1.2 Hashing with deep neural networks

Liong et al. [LLW+15] first introduced two hash functions built around a multilayer perceptron on top of GIST [OT01] features, one supervised and another unsupervised. HashNet [CLWY17] used AlexNet and ResNet CNNs with a continuous relaxation method, increasing the sharpness of a *sign* function approximator over the course of training. Deep Cauchy Hashing is a contrastive method [CLLW18b] relying on a modified AlexNet and a pairwise loss based on a Cauchy distribution. DistillHash uses noisy similarity labels [YLD+19] based on embedding distance between data pairs to learn bitcode assignments. HashGAN [CLLW18a] incorporates synthetic images generated by a GAN to augment the training data. K-Nearest-Neighbors Hashing [HWC19] takes advantage of the neighborhood structure in embedding space to binarize based on an optimal space partition. Wu et al. [WDL+19] introduced a method for incremental hashing, in the sense that expanding the database does not require complete retraining of the hash function on the full dataset.

2.3.2.2 Video hashing

A number of methods [LLTZ17, WLG+17, WHG+19] for video hashing are pooling-based: a fixed number of frame representations is generated, then combined with

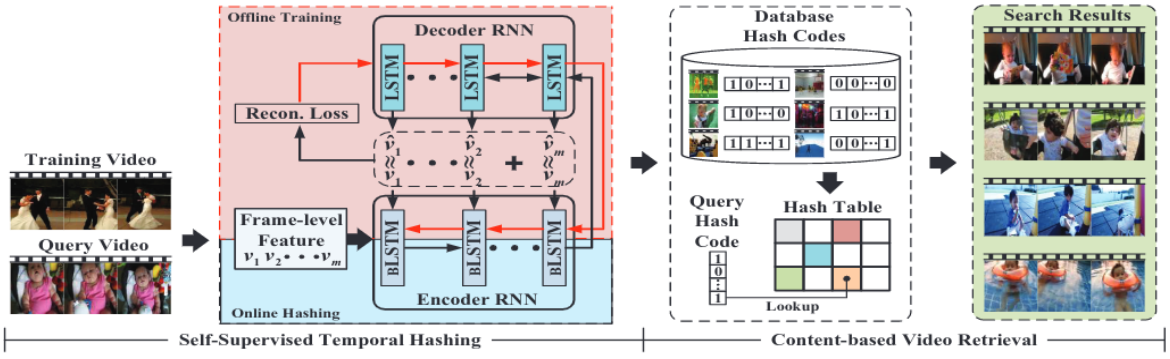


Figure 2.3: Self-supervised temporal hashing method. The binary keys or bitcodes used for retrieval are generated by an encoder trained on a self-supervised reconstruction task. Courtesy of Zhang et al.

temporal averaging to generate one feature to binarize. Even when temporal models are involved in generating the frame representations [WLG+17, WHG+19], averaging features over the course of an entire video inevitably restrains the bitcode's expressiveness with respect to temporal dynamics[ZWHC16]. Other works have proposed methods with a higher degree of temporal awareness. Zhang et al.'s SSTH [ZWHC16] trained an encoder based on a differentiable binary LSTM recurrent unit, using the bitcode itself as a hidden state. Subsequent methods are similar: Song et al.'s SSVH [SZL+18] incorporated some temporal hierarchy modeling; Li et al. [LCL+19] added neighborhood structure preservation as a learning constraint.

2.3.3 Retrieval in medical computer vision

In medical computer vision, most works on content-based retrieval focus on diagnostic medical imaging [CCL+18, PBW+19, SPCR20, GVYY17]. One from Chen et al. [GVYY17] involves hashing, on a database of chest X-ray images.

A few works have used intraoperative video data: Droueche et al.[DQL+14]'s video retrieval method is based on MPEG-compressed video representations and handcrafted features. Amanat et al's [AIK+18] performed retrieval based on PCA applied to SIFT features. Funke et al. retrieved still frames from *cholec80*, however this only was a qualitative evaluation of their features trained with temporal coherence[FJM+18]. Petscharnig et al. [PS18] tackled the problem of image-to-video linking, i.e. retrieving the video source of a still frame. Their study was done on a dataset of gynecologic endoscopies.

2.4 Thesis Positioning

Our objective is to introduce new uses for the unannotated data which will directly or indirectly assist the surgeon in navigating the surgery in the operating room, and do so in real time.

An overview of the previous work using unannotated videos for surgical activity understanding tasks shows that the role of these videos was fairly limited. For the most part, they were used as pretraining fodder for visual feature extractors: frame sort-

ing [BWK+17], temporal distance comparisons [FJM+18], remaining time [YMMP18]. While those methods proved to be efficient for enhancing the downstream task of surgical phase recognition, the vast pool of unannotated videos they used served no further purpose beyond a self-supervised pretext task. No attempt was made to understand this data, navigate it and turn it into more valuable information for future use. Self-training [RHS05, RDG+18] is a semi-supervision modality addressing those shortcomings, since it generates synthetic labels for the unannotated data - however to the best of our knowledge, no application to surgical activity recognition or even general video activity recognition existed at the time. The real-time usability / recognition performance has also never been adequately addressed either. Pointed out by [Twi17] when observing a performance increase upon replacing the standard LSTM by one that is bidirectional, no attempt has been made in order to reduce this performance gap for real-time use. The method we propose [YMMP19] in the next section addresses all those issues , by introducing a semi-supervised self-training scheme using both LSTM and bidirectional LSTM in a complimentary manner.

Enriching the unannotated data is one way of using it; providing the tools to explore it and efficiently search through it is another, which can be done with video hashing. Literature on this topic is quite limited [LLTZ17, ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19]. The main shortcoming of all presented approaches is to consider video retrieval from a purely static standpoint. Meanwhile text-based search has evolved into a dynamic tool, capable of adapting to the anticipated course of events in order to be used out of the pocket in the middle of any task. One key concern is the sampling scheme employed: out of all the frames in a video, a small predetermined number of evenly spaced frames is considered, independently of the video's duration. The extracted information is therefore very sparse, and in the case of a real-time video source, needs to be constantly regenerated from the beginning of the video. Lack of anticipation is another concern: all those methods assume the entire video from beginning to end is available as the search query, and therefore no content is missing. In the case of livestreams, this working hypothesis is false. Our method [YP20] addresses both these concerns, making video hashing both incremental and predictive.

The existing work on video retrieval for surgical videos is minimal; Droueche et al.'s [DQL+14] work on retrieval for cataract surgery videos relied on handcrafted features from compressed videos instead of DNNs operating on the actual pixel data. The scale of the experiments conducted was fairly small, using datasets of 23, 250 and 69 videos. The training set was used as the database, which deviates from the stricter protocol enforced in video hashing publications [LLTZ17, ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19]. No hashing was involved either; retrieval was based on floating-point values. Retrieval times of 9 minutes were mentioned, which limits real-time usage. Our work, on the other hand, brings DNN-based hashing to a very-large-scale surgical video database, ensuring real-time video retrieval efficiency with delays of a few seconds at most.

CHAPTER 3

Semi-supervision for real-time surgical phase recognition

"It is much more rewarding to do more with less."
- Donald Knuth

Contents

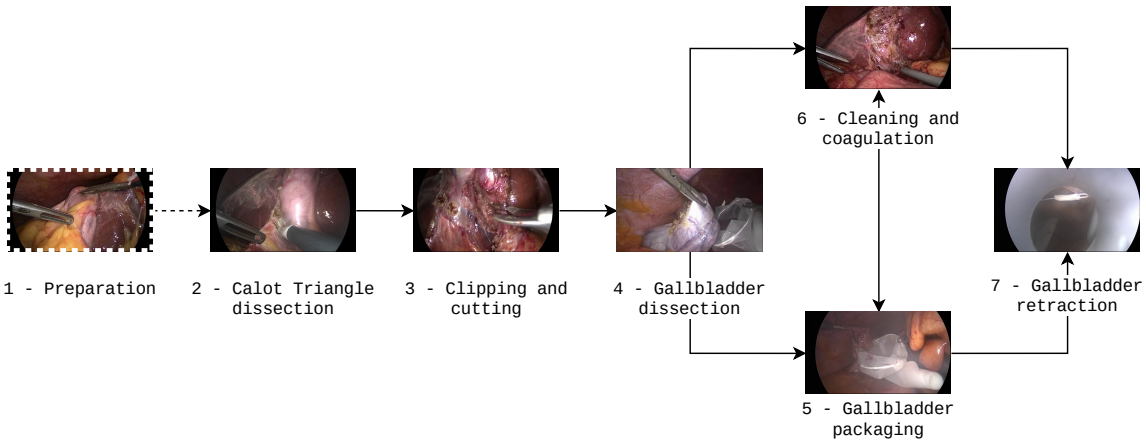


Figure 3.1: Surgical phases in the workflow of laparoscopic cholecystectomy

Between the standard approach of full supervision, and self-supervision where no annotations are available for training, semi-supervision occupies the middle ground, with methods relying on *partially annotated* datasets. With phase annotations being currently scarce, it is natural to consider methods that facilitate their production. **Assisted annotation**, explored by research efforts simultaneous to ours in a study on two cholecystectomy videos, is an option [LRM+19]. **Automatic annotation** is another which we study here, allowing both increased phase recognition performance and exploration of the unannotated data.

3.1 Objectives

The work presented in this chapter is a first attempt at answering the question "what can be achieved in the OR with the unannotated endoscopic video data?". Here we tackle the problem of surgical phase recognition in cholecystectomy, a staple in surgical activity understanding. As shown in Chapter 1, a fraction of the cholecystectomies registered in our database come with phase annotations: 120 videos are currently annotated with phases, while over 500 are registered in Endocorpus. This situation is therefore an excellent fit for a semi-supervised method. In the following experiments we recreate those conditions of annotation scarcity on a smaller scale: assuming 80 videos are at our disposal, how do our methods perform when only 20 out of those are annotated? How do they perform when this number drops down to 10, 5, 3, all the way down to a single video?

3.2 Methods

3.2.1 Data preparation

The chosen dataset for this task is *cholec120*, containing 120 recordings of laparoscopic cholecystectomy performed at Strasbourg's *Nouvel Hôpital Civil*. We reserve 30 for testing and 10 for validation, leaving $N = 80$ videos to choose from for training.

As shown in Figure 3.1, the workflow is mostly linear. A few videos do not feature a

preparation phase, starting directly with the Calot triangle dissection phase. The order between phases 5 (gallbladder packaging) and 6 (cleaning coagulation) is occasionally reversed. However, despite the dataset's uniformity with respect to phase ordering, workflow greatly varies within the dataset: video duration considerably changes from one case to another, with 956s of standard deviation around the 2287s average. The shortest video only lasts 741 s -barely over 10 min- while the longest lasts 5987s or over an hour and a half. When selecting small numbers of annotated videos for training, it is therefore important that the selection has similar duration statistics. Even then, the selection process should be repeated in order to mitigate the effect of potential outliers.

In the following lines, we will refer to:

$$E = \{(V_0, \mathcal{A}(V_0)), (V_1, \mathcal{A}(V_1)), ..., (V_{N-1}, \mathcal{A}(V_{N-1}))\}, \tag{3.1}$$

as the initial set of 80 manually annotated videos to choose from, V_k and $\mathcal{A}(V_k)$ being a video and its ground truth annotations respectively. Considering the voluntarily low number of annotated videos selected for training (from 20 down to 1), we sample 3 non-overlapping mini-training sets of every size in order to prevent biased results coming from the selection process, and ran our series of experiments independently for each mini-training set. In an effort to match the original training set in terms of video duration statistics, we divide the 80 videos into duration quartiles $Q1$ to $Q4$, then randomly sample videos from $Q1$, $Q2 \bigcup Q3$, and $Q4$ with a 20/60/20 ratio respectively. For mini-training sets of only one video the single choice is limited to $Q2 \bigcup Q3$ in order to avoid outliers. The mini-training sets are referred to as:

$$E_{i,j} = \{(V_k, \mathcal{A}(V_k)), ...\}, \tag{3.2}$$

where the first index $i \in \{1, 3, 5, 10, 20\}$ indicates the size of the set of ground truth labeled videos employed, and the second index $j \in \{0, 1, 2\}$ denotes the first, second or third repeat of the experiment for that particular size.

3.2.2 Teacher & student models

We present two model architectures for surgical phase recognition: the CNN-LSTM and the one newly introduced in our work, the CNN-biLSTM-CRF. The choice between the two is subject to a tradeoff between real-time usability for the first, and enhanced recognition performance for the second; those characteristics are directly tied to their distinct architectures, which we discuss in the following subsections. Using them in our setting as teacher (CNN-biLSTM-CRF) and student (CNN-LSTM) ultimately solves this tradeoff.

3.2.2.1 DNN components

Both architectures rely on a combination of spatial and temporal artificial neural networks. The spatial component in both is a ResNet-50 V2 CNN [HZRS16], serving as the visual feature extractor on RGB frames: for one $256 \times 256 \times 3$ input, a 2048-dimensional vector representation is generated.

The sequence of feature vectors extracted from a video is then aggregated by a recurrent neural network (RNN). In the case of the student, the chosen RNN is built

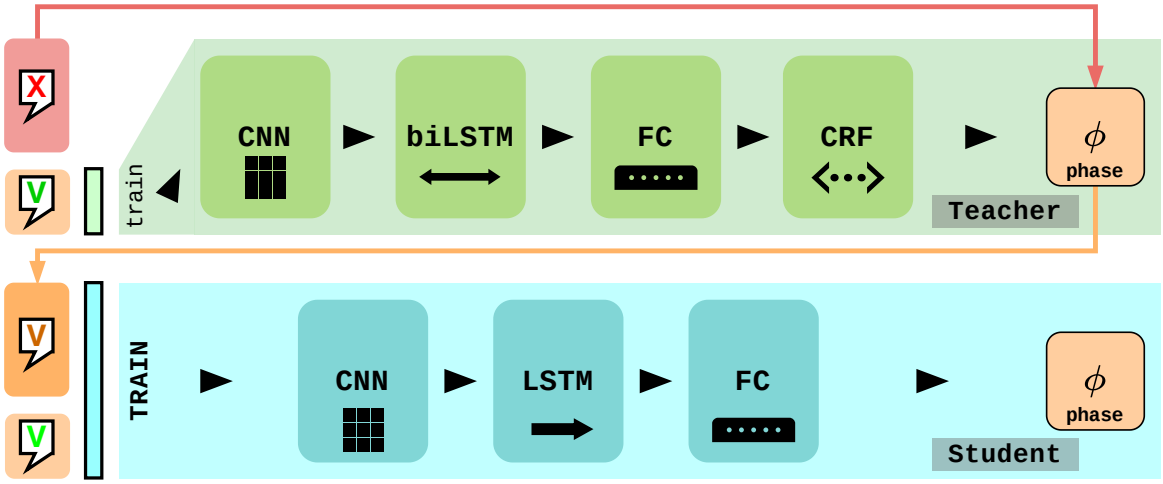


Figure 3.2: Overview of the teacher-student method.

around a forward-directional LSTM. At any given timestep, all the student needs for inference is the feature vector from the current frame as well as the previously computed cell state and hidden state. Functioning in real time is therefore entirely within its capabilities.

The teacher, however, feeds the feature vector sequence to a RNN based on a **bidirectional LSTM** [GS05], i.e. a pair of LSTMs, one chronological and one reverse. Its output at any given timestep depends on the current feature vector for the current frame and the chronological LSTM's previous states, computed using all past instants. It also depends on the reverse chronological LSTM's previous state, which requires input features for all future instants. This property of the biLSTM is a double-edged sword; because of it the teacher cannot be used intraoperatively, i.e. before the actual end of the surgery. On the flipside its predictions factor in much more useful information than the student, and are more likely to be correct.

3.2.2.2 Transition logic

A single fully connected layer projects the sequence of outputs from the temporal model to a sequence of per-class log-probabilities or logits. This is the case for both the teacher and the student. However, the student has its predictions decoded with a simple $argmax$, independently from timestep to timestep. Since the teacher is forced to see the entire sequence, we take advantage of this by incorporating an additional layer, enabling it to learn the transition logic between phases. This is the role played by the linear chain **conditional random field** or **CRF** in the model, the inner workings of which we describe here.

Given a sequence of logits $S = (s_0, ..., s_T)$, for any given timestep t and class index k we note the k^{th} entry of s_t as $s_t[k]$. Let Θ be an $N_c \times N_c$ real-valued matrix, with entries noted as $\Theta[i, j]$ for class indices i, j.

The score of any sequence of tags (i.e. predicted classes) $Y = (y_0, ..., y_T)$ can then be defined as:

$$C(S, Y, \Theta) = \sum_{t=0}^{T} s_t[y_t] + \sum_{t=0}^{T-1} \Theta[y_t, y_{t+1}]. \tag{3.3}$$

The first sum collects unary potentials, i.e. terms associated with single sequence elements, while the second one collects binary potentials resulting from transitions.

The trainable parameter of the model is Θ, called the transition matrix. Using the definition of the score C, we are able to define the likelihood of a tag sequence as:

$$p(Y|S) = \frac{e^{C(S,Y,\Theta)}}{\sum\limits_{U \in [\![1, N_c]\!]^{T+1}} e^{C(S,U,\Theta)}}, \tag{3.4}$$

which can be seen as a softmax value for Y over all possible tag sequences U. The scaling factor $Z = \sum\limits_{U \in [\![1, N_c]\!]^{T+1}} e^{C(S,U,\Theta)}$ is also called the **partition function**. Brute-force computation of Z is $O(N^{T+1})$, and generally intractable. A more adequate way to carry out this computation uses dynamic programming, via the *forward algorithm*. We can observe:

$$Z = \sum_{U \in [\![1, N_c]\!]^{T+1}} e^{\sum_{t=0}^{T} s_t[y_t] + \sum_{t=0}^{T-1} \Theta[y_t, y_{t+1}]} \tag{3.5}$$

$$= \sum_{k,n \in [\![1, N_c]\!]^2} e^{\sum_{t=0}^{T} s_t[k] + \sum_{t=0}^{T-1} \Theta[k,n]} \sum_{U \in [\![1, N_c]\!]^{T}} e^{\sum_{t=1}^{T} s_t[y_t] + \sum_{t=0}^{T-1} \Theta[y_t, y_{t+1}]} \tag{3.6}$$

This simply factors Z by the contribution of the first item in the sequence. The left-hand term of the product costs $O(N_c^2)$ operations, and the same factorization can be applied to the right-hand term. Recursive computation of Z in this manner is therefore $O((T+1)N_c^2)$.

We then employ $L = -log(p(Y_{true}|S))$ as our training loss, Y_{true} being the tag sequence corresponding to the ground truth.

At inference time, the CRF returns the best scoring sequence Y_{opt}. This is also computed using dynamic programming, with the *Viterbi algorithm*. We define:

$$\delta_\tau(k) = \max_{U \in [\![1, N_c]\!]^{\tau+1}, u_\tau = k} \left(\sum_{t=0}^{T} s_t[u_t] + \sum_{t=0}^{T-1} \Theta[u_t, u_{t+1}] \right) \tag{3.7}$$

This is the maximum scoring sequence up to timestep τ ending with tag k. Then:

$$\delta_\tau(k) = s_\tau(k) + \sum_{n=1}^{N_c} \delta_{tau-1}(n) + \Theta[n, k] \tag{3.8}$$

Recursive computation of all the δ values costs, again, $O((T+1)N_c^2)$. The best path can be found by recording the tags corresponding to each δ in a separate array to backtrack from.

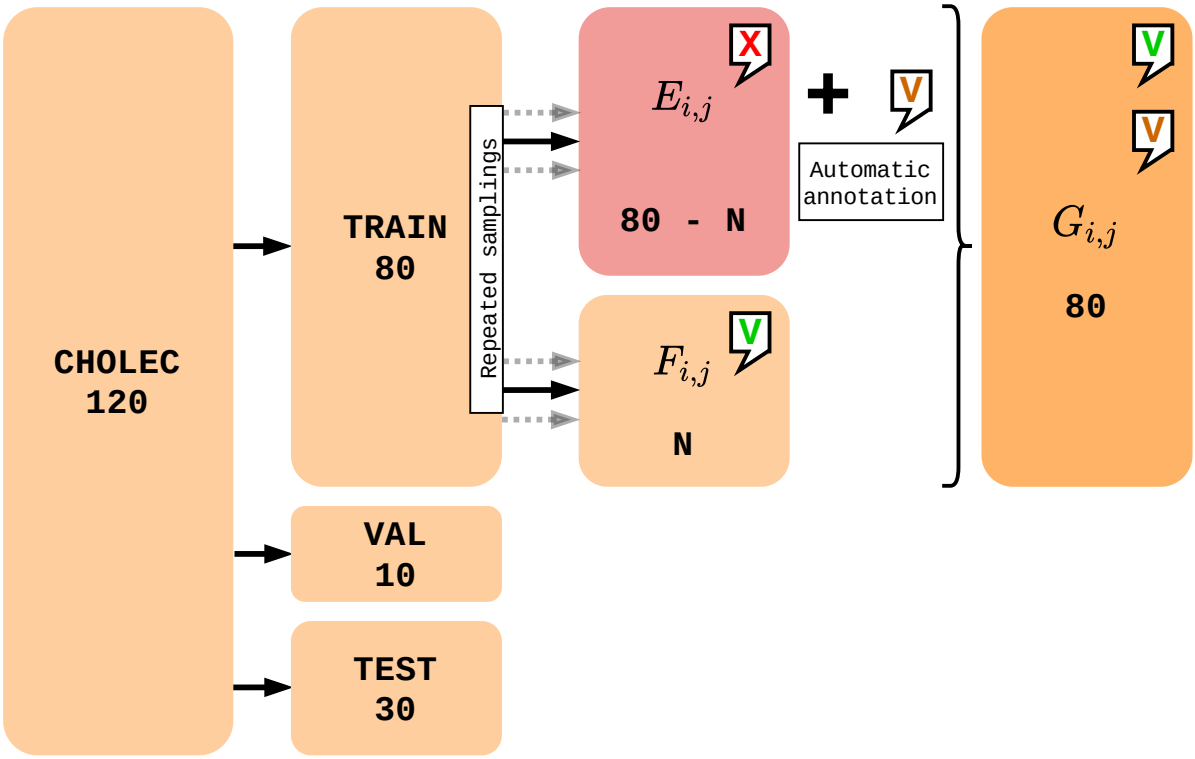


Figure 3.3: Data splitting and sampling process.

3.2.2.3 Training

In all of the following experiments, the Resnet-50 CNN is initialized with ImageNet pretrained weights. Test accuracy on ImageNet for those weights reaches 75.6%. The teacher model's CNN is first pretrained with only one fully connected layer on top, on $E_{i,j}$, directly for phase recognition. Weights from the first and second blocks of Resnet are frozen. Data augmentation is applied using random isometries. Visual feature vectors are then extracted from $E_{i,j}$ using the pretrained CNN.

The biLSTM - CRF is then trained end-to-end on the extracted features with untruncated backpropagation through time across the entire video, with binary cross-entropy as the loss function.

Using the trained biLSTM - CRF, we generate new annotations for videos in $E \backslash E_{i,j}$. This leads to a set of videos with synthetic annotations:

$$F_{i,j} = \left\{ (V_k, \widehat{\mathcal{A}_{i,j}}(V_k)), ... \right\}. \tag{3.9}$$

We then define $G_{i,j} = E_{i,j} \bigcup F_{i,j}$, which contains all videos from the original training set, and combines a small set of manual annotations with a majority of synthetic labels. The student CNN-LSTM model is trained in the same two-step manner, this time on $G_{i,j}$. Hyperparameters used for training are detailed in table 3.1. All experiments are performed using Tensorflow with the Adam optimizer [KB15], on servers fitted with Nvidia 1080Ti GPUs.

In order to justify every component in the teacher model, we conduct a series of ablation studies 3.4 by training and evaluating the following models:

- (M1) CNN (obtained from the pretraining step)

Table 3.1: Hyperparameter table

CNN pretraining	
Learning rate	$5 \cdot 10^{-5}$
# of epochs	27
Minibatch size	32
Weight decay	$5 \cdot 10^{-4}$
LSTM	
Learning rate	$5 \cdot 10^{-5}$
# of epochs	350
State size	128
Dropout	0.3
Weight decay	$5 \cdot 10^{-4}$
CRF	
Learning rate	$5 \cdot 10^{-5}$
# of epochs	350
Weight decay	$5 \cdot 10^{-3}$
BiLSTM/BiLSTM-CRF	
Learning rate (biLSTM)	$1 \cdot 10^{-3}$
Learning rate (biLSTM-CRF)	$1 \cdot 10^{-4}$
# of epochs	350
State size	$2 \cdot 64$
Dropout	0.4
Weight decay	$5 \cdot 10^{-4}$
Weight decay	$5 \cdot 10^{-4}$

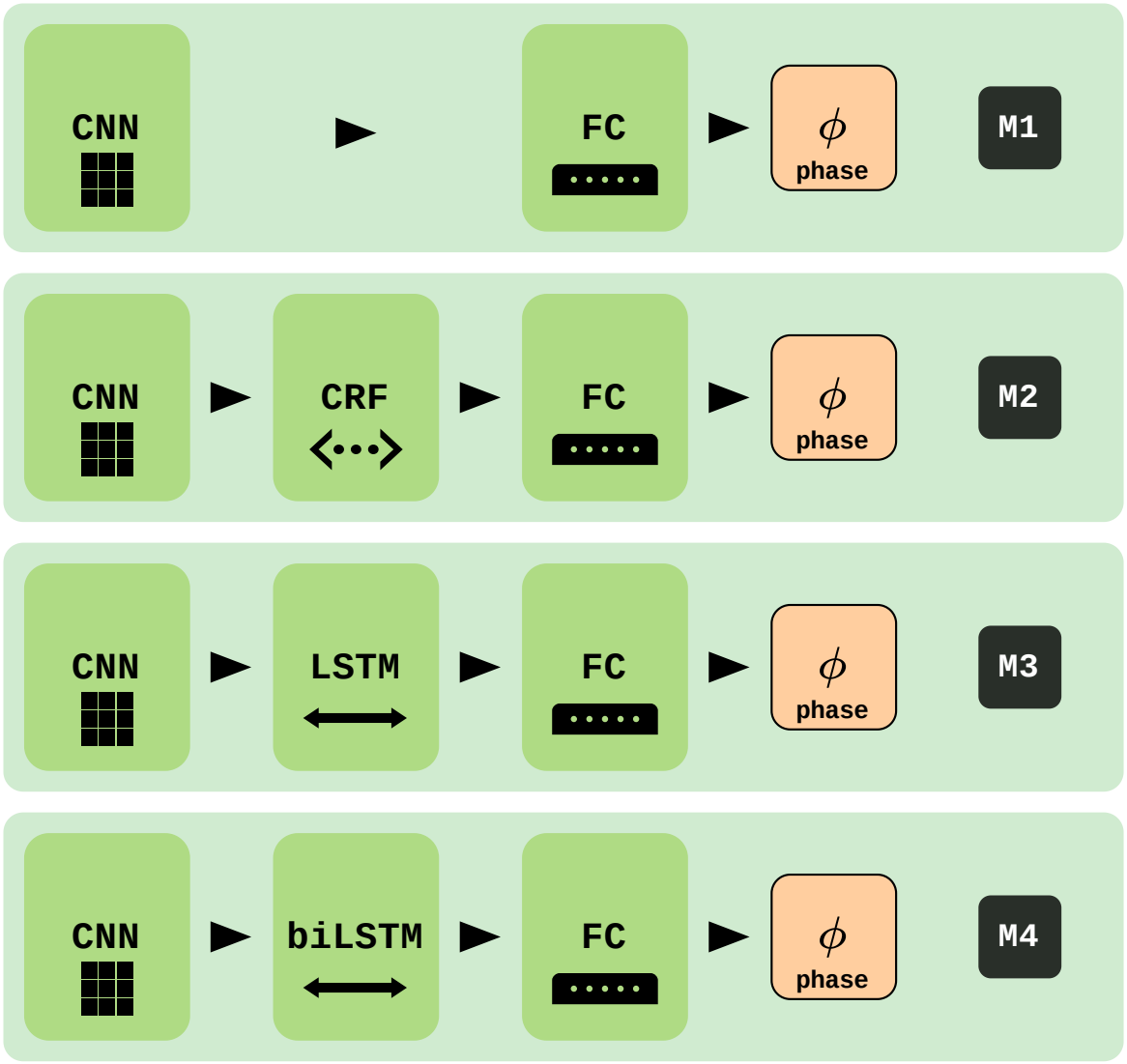


Figure 3.4: Ablation studies conducted for the teacher model. Left to right, top to bottom: M1, M2, M3, M4.

- (M2) CNN-CRF
- (M3) CNN-unidirectional LSTM
- (M4) CNN-biLSTM

Temporal models M2 to M4 are trained in the same two-step manner as the proposed CNN-biLSTM-CRF model, referred to as M5.

In order to provide comparison points with the fully supervised approach, the teacher model along with every model featured in the ablation studies are also trained on the original set of 80 manually annotated videos. The only student model mentioned so far is the CNN-unidirectional LSTM, due to its real-time inference capabilities. Another interesting possibility is to also use a CNN-biLSTM-CRF as the student, in order to obtain a stronger offline predictor.

3.3 Results

3.3.1 Teacher model ablation studies

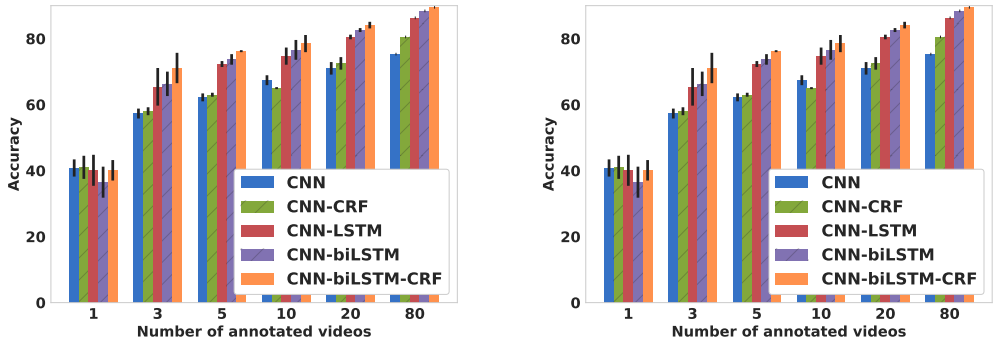


Figure 3.5: Accuracy (left) and F1 (right) for the ablation models, for all mini-training set sizes

To demonstrate the benefits of our approach, we have conducted a total number of 125 experiments, counting all mini-training set sizes and all the models featured in the complementary studies. To test our models, we reported their accuracy (Acc). This metric is valuable in the sense that it measures the amount of time spent making correct predictions. However, it lacks responsiveness to potential deficiencies in less populated classes. For this reason we also reported their precision (Pre), recall (Rec) and F1 score during inference on the test set. Unless otherwise specified, precision, recall and F1 are averaged over the 7 classes. For every metric at a given mini-training set size, we provide the mean and standard deviation over the 3 repeats of the corresponding experiment.

Results for every mini-training set size, the proposed teacher model (M5) and models from the ablation study (M1 to M4) are shown in Table 3.2. Directly applying the CRF after the CNN (M2) yields poor results, likely due to temporal noise affecting the logits emitted by the CNN. Temporal models trained on a single video are severely affected by overfitting, and therefore also exhibit subpar performance. With 3 or more manually annotated videos, however, the biLSTM and the CRF deliver significant performance improvements.

The CNN-biLSTM (M4) achieves stronger performance than the CNN-LSTM (M2), although not as much as the full CNN-biLSTM-CRF model (M5), which is consistently the best performer (Figure 3.5). This is observed on all mini-training set sizes except for single videos. As expected, increasing the number of videos improves all global metrics - accuracy, average F1, average precision, average recall - although per-phase precision and recall may fluctuate (Table 3.3). This establishes the CNN-biLSTM-CRF model as the strongest predictor, and therefore the best suited for the role of teacher.

To qualitatively appreciate improvements from the new model, the predictions on six videos from the test set are presented in Figure 3.6: the top three and bottom three of the CNN-biLSTM-CRF ranked by accuracy. The CNN-biLSTM-CRF makes the most sensible predictions with respect to the chronological order between the

Table 3.2: Ablation study for the teacher model, accuracy and average F1

		1	3	5	10	20	80
M1	Acc	40.8 ± 2.6	57.3 ± 1.5	62.2 ± 1.2	67.4 ± 1.5	71 ± 1.9	75.3
	F1	23 ± 4.3	39.9 ± 4.1	48.8 ± 2.6	56 ± 1.9	59.1 ± 1.6	63.8
M2	Acc	41 ± 3.5	58 ± 1.2	63 ± 0.6	65 ± 0.3	72.5 ± 1.9	80.5
	F1	22.1 ± 5.2	40.7 ± 4.6	48.5 ± 3.3	54.3 ± 1.3	59.9 ± 1.8	69.7
M3	Acc	40.1 ± 4.7	65.4 ± 5.7	72.3 ± 0.9	74.7 ± 2.6	80.5 ± 0.7	86.3
	F1	5 ± 2.9	47.4 ± 9.9	57.9 ± 1.8	62.7 ± 3	70.2 ± 1.3	78.2
M4	Acc	36.5 ± 4.7	66.3 ± 3.7	73.7 ± 1.6	76.6 ± 3	82.6 ± 0.6	88.4
	F1	2.9 ± 3	47.9 ± 8.5	60.1 ± 5	67 ± 3.8	74.5 ± 1.3	81.7
M5	Acc	40.1 ± 3.1	71.1 ± 4.6	76.2 ± 0.3	78.5 ± 2.6	84.1 ± 1	89.5
	F1	21.1 ± 11	55.3 ± 9	65.8 ± 1.2	69.7 ± 1.3	75.8 ± 1.5	82.5

Table 3.3: Per-phase precision and recall, CNN-biLSTM-CRF model

		1	3	5	10	20	80
P1	Pre	5.5 ± 8.4	59.6 ± 0.8	52 ± 14.3	56.9 ± 14.7	68.4 ± 8.3	86.6
	Rec	56.6 ± 26.7	77.4 ± 14.8	51.7 ± 4.8	54.4 ± 14.1	92.6 ± 4.5	96.4
P2	Pre	48.3 ± 42.3	48.1 ± 10.5	80.1 ± 3.7	77.1 ± 0.9	58.8 ± 10	68.9
	Rec	24.8 ± 23.7	83.5 ± 10.7	84.7 ± 3.1	90.6 ± 5	87.2 ± 2.2	83.4
P3	Pre	36.7 ± 25	76.9 ± 12.6	62.6 ± 17	80.4 ± 11.5	93.4 ± 3.9	96.5
	Rec	50.5 ± 44.9	64.3 ± 15.8	39.4 ± 26.1	31.6 ± 10.9	75.9 ± 2.9	79.9
P4	Pre	61.3 ± 10.4	75.4 ± 3.2	81.1 ± 5.2	86.3 ± 6.7	83.1 ± 3.3	89.4
	Rec	45.9 ± 45.2	50.1 ± 29	84.4 ± 4.8	80.4 ± 1.8	70.3 ± 8.8	72.7
P5	Pre	0 ± 0	10.9 ± 15.5	74.4 ± 3.1	77.7 ± 0.8	42.6 ± 5.1	53.5
	Rec	12.9 ± 11.6	71.8 ± 16.2	83.3 ± 1.7	84 ± 3.1	83.2 ± 1.4	88.2
P6	Pre	3.7 ± 6.4	38.7 ± 29.5	80.4 ± 2.1	76.2 ± 4.2	85.1 ± 2.8	91.5
	Rec	11.3 ± 16.6	71 ± 11.9	46.7 ± 7.6	61.5 ± 5.4	81.2 ± 3.6	91.5
P7	Pre	60.3 ± 44.7	82.1 ± 8.5	64.1 ± 7.9	70.8 ± 1.6	87.1 ± 3.6	94.1
	Rec	50.3 ± 38.9	64.6 ± 8.3	77.7 ± 10.1	77.8 ± 9.5	80 ± 4.1	77.3
Avg	Pre	26.4 ± 4.5	62 ± 6.1	70.4 ± 4.2	75.1 ± 1.6	75.8 ± 1.4	82.6
	Rec	40.5 ± 18.5	62.9 ± 7.5	66.8 ± 4.7	68.6 ± 0.5	79.8 ± 1.3	84.5

phases; it more specifically avoids inferring incorrect phases in short isolated bursts as the biLSTM sometimes does.

In order to confirm the quality of the material the student model learns from, we ran our metrics on the annotations from $G_{i,j}$, which mix ground truth annotations and teacher-generated annotations. Results are shown in Table 3.4.

As expected, teachers trained on more manually annotated videos produce better annotations. The presence of more manually annotated videos in the $G_{i,j}$ sets also contributes to greater overall label quality, e.g. 83.5 % F1 score for $G_{20,j}$ on average. Per-phase results indicate stronger performance on phases P2 (90.3% precision, 91.6 % recall for 20 manually annotated videos) and P4 (87.4% precision, 92.5 % recall), which are usually the most prevalent. Despite uneven results across phases, $G_{i,j}$ sets obtained from 3 or more manually annotated videos appear to be overall serviceable for training a student model.

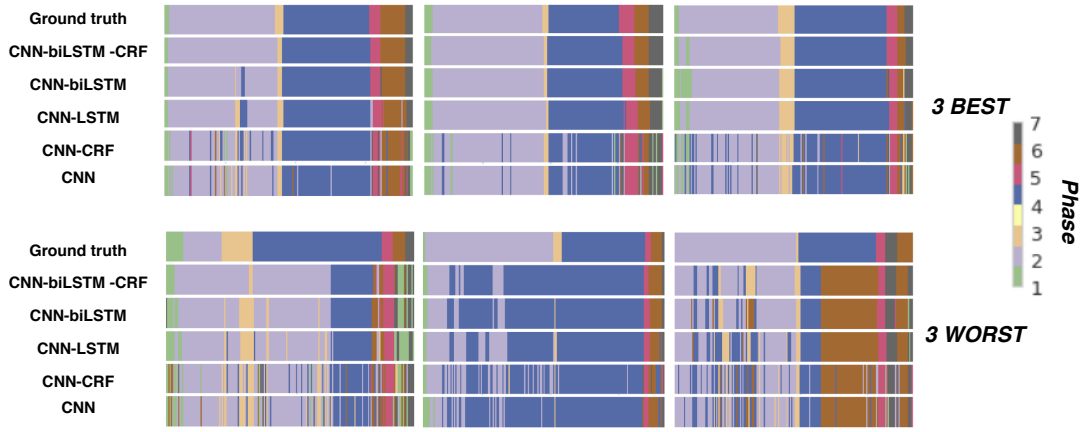


Figure 3.6: Qualitative results for the teacher, compared to ablation models - top 3 best and worst. Note the suppression of error bursts.

3.3.2 Student performance

To appreciate the benefits of using our synthetic labels, we compare in the first two groups of rows of Table 3.5 the CNN-LSTM only trained with few manually annotated videos against the CNN-LSTM trained with these same videos and annotations plus the videos annotated by the teacher. Results for a single ground truth-annotated video, as one can expect from the results of the corresponding teacher models, are quite poor. Even though the $G_{1,j}$ sets contain 80 times as many videos as the $E_{1,j}$ used for the teachers, the quality of the annotations, as shown in Table 3.4, is extremely low. CNN-LSTM models trained on those all exhibit sub-50% performance on every global metric, despite showing some improvement compared to the CNN-LSTM trained on $E_{1,j}$. Decent results are observed starting from 3 videos, with a 2.9 to 8.8 point increase in accuracy when adding synthetic annotations, and similar increase in F1 score.

With 80 ground truth-annotated videos, accuracy and F1 reach 86.3% and 78.2% respectively (Table 3.5). Therefore, the use of synthetic annotations roughly cuts down the performance gap between using 20 and 80 ground truth-annotated videos by half.

3.3.3 Self-learning of the teacher model

3.3.4 Teacher self-learning

By using the $G_{i,j}$ sets to train a new CNN-biLSTM-CRF model, the offline performance increases even further (Table 3.5, last 2 groups of rows) - except for the 1-video case, where performance actually degrades. Results with 20 ground truth-annotated videos are particularly notable, as they match those obtained with 80 ground truth-annotated videos from the CNN-LSTM model (86.3% accuracy, 78.2% F1).

3.4 Conclusion

The work presented in this section shows the superior performance of the new CNN-biLSTM-CRF teacher architecture compared to the previous CNN-LSTM used for surgical phase recognition. Although this model is restricted to offline inference, we

Table 3.4: Teacher-completed annotation set metrics

		1	3	5	10	20
P1	Pre	44.4 ± 33.4	63.5 ± 10.5	68.8 ± 10.3	78.2 ± 5.3	85.9 ± 5.5
	Rec	4.6 ± 5	55.2 ± 3.4	47.9 ± 2.8	52.4 ± 12.7	73.5 ± 4.3
P2	Pre	60.9 ± 4.7	81.1 ± 1.9	86 ± 2.5	83.1 ± 2.3	90.3 ± 1.6
	Rec	35.4 ± 17.2	77.4 ± 5.7	79.1 ± 6.6	88.9 ± 0.3	91.6 ± 3
P3	Pre	79.7 ± 28.8	67.9 ± 9.4	66.2 ± 18.7	76.3 ± 5.4	84.3 ± 2.5
	Rec	0.6 ± 0.2	19.5 ± 12.3	47 ± 23.2	40.9 ± 7	60 ± 5.8
P4	Pre	53.6 ± 18.6	72.7 ± 10.5	74.5 ± 4	83.2 ± 0.6	87.4 ± 3.6
	Rec	60.8 ± 36.9	84.6 ± 5.2	89.2 ± 2.9	86.4 ± 4.4	92.5 ± 1.5
P5	Pre	87.2 ± 11.4	66.6 ± 14.9	77.6 ± 1.9	82.8 ± 3.1	83.9 ± 2.9
	Rec	23.2 ± 16.5	78.6 ± 9.1	81.7 ± 1	81.1 ± 4.2	88.3 ± 2.1
P6	Pre	43.4 ± 39.5	70.1 ± 9.1	71.8 ± 7.5	76.9 ± 4.2	87.3 ± 1.2
	Rec	43.9 ± 38.9	52.2 ± 24.1	51.2 ± 10.1	71.2 ± 10.1	82 ± 4
P7	Pre	58.1 ± 31.3	62.4 ± 5	63.3 ± 9.2	71.8 ± 5.9	84.4 ± 4.3
	Rec	13.4 ± 14.8	72.3 ± 8.1	77.6 ± 7.9	81.2 ± 1.8	84.2 ± 2.6
Avg	Pre	61.3 ± 3.8	69.2 ± 2	72.6 ± 5.3	78.9 ± 0.8	86.2 ± 1.3
	Rec	26 ± 15.7	62.8 ± 5.3	67.7 ± 4.7	71.7 ± 2.3	81.7 ± 0.6
Acc		**39.2 ± 4.7**	**73.1 ± 3.1**	**76.6 ± 1.2**	**81.6 ± 1**	**88 ± 1.2**
F1		**23.5 ± 3.2**	**62 ± 4.7**	**67.6 ± 0.8**	**73.7 ± 2.1**	**83.5 ± 1.1**

also propose a teacher/student strategy that leverages the new model for real-time prediction, by exploiting it as a source of synthetic annotations for a CNN-LSTM student model.

Experimental results, obtained using manual annotations for 25% or less of all available training data, show serious potential for scaling surgical phase recognition to a large number of videos while alleviating the burden of collecting manual annotations. The performance deficit between scenarios with 25% and 100% ground truth annotation availability is halved for a CNN-LSTM online prediction model when adding synthetic labels from the teacher. When swapping the student for a CNN-biLSTM-CRF offline prediction model, the gap is fully closed.

Many other types of surgical procedures than cholecystectomy for which large amounts of annotated data are not yet available, such as gastric bypass might be able to benefit from this method as well.

Table 3.5: Performance with and without teacher-generated annotations.

		1	3	5	10	20
CNN-LSTM, no teacher	Acc	40.1 ± 4.7	65.4 ± 5.7	72.3 ± 0.9	74.7 ± 2.6	80.5 ± 0.7
	Pre	20.7 ± 1.9	51.7 ± 7.1	60.4 ± 2.9	64.1 ± 1.5	71.1 ± 0.7
	Rec	17.5 ± 7.5	58.6 ± 8.2	64 ± 5.5	68 ± 3.1	64.1 ± 2.7
	F1	5 ± 2.9	47.4 ± 9.9	57.9 ± 1.8	62.7 ± 3	70.2 ± 1.3
CNN-LSTM, with teacher	Acc	42.1 ± 6.3	74.6 ± 3.6	77.7 ± 0.8	79.1 ± 0.9	83.4 ± 0.3
	Pre	24.6 ± 5.6	63 ± 5	65.8 ± 5.1	66.6 ± 1.7	73.5 ± 1.2
	Rec	38.9 ± 19.4	65.1 ± 7.3	72.3 ± 5.2	74.7 ± 1.7	76.8 ± 0.7
	F1	14.5 ± 13.5	56.1 ± 8.6	64.5 ± 2.9	66.9 ± 3.1	73.2 ± 0.8
CNN-biLSTM-CRF, no teacher	Acc	40.1 ± 3.1	71.1 ± 4.6	76.2 ± 0.3	78.5 ± 2.6	84.1 ± 1
	Pre	26.4 ± 4.5	62 ± 6.1	70.4 ± 4.2	75.1 ± 1.6	75.8 ± 1.4
	Rec	40.5 ± 18.5	62.9 ± 7.5	66.8 ± 4.7	68.6 ± 0.5	79.8 ± 1.3
	F1	21.1 ± 11	55.3 ± 9	65.8 ± 1.2	69.7 ± 1.3	75.8 ± 1.5
CNN-biLSTM-CRF, with teacher	Acc	43.1 ± 7.5	74.9 ± 4.6	78.7 ± 1	80.8 ± 0.8	86.3 ± 1
	Pre	26.3 ± 4.1	64.2 ± 5.8	67.4 ± 5.9	70.2 ± 2.7	78 ± 1.4
	Rec	38.8 ± 16.5	65 ± 7.3	73.8 ± 5	76.9 ± 1.9	81.7 ± 0.5
	F1	18.2 ± 10.9	55.7 ± 9.7	66.8 ± 2.7	69.9 ± 4	78.1 ± 1

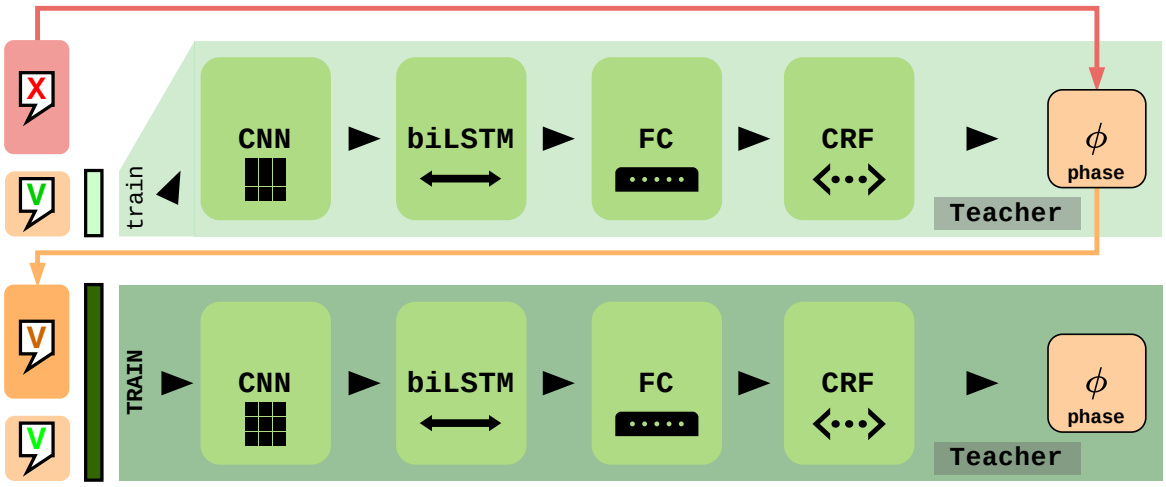


Figure 3.7: Single loop of self-learning for the teacher.

CHAPTER 4

Learning efficient video representations for retrieval

"In the afterlife you relive all your experiences, but this time with the events reshuffled into a new order: all the moments that share a quality are grouped together."

- David Eagleman, *Sum*

Contents

This chapter lays down theoretical foundations for video retrieval with hashing, which is at the core of the following chapters. After formalizing the problem of retrieval and search with the concept of nearest neighbors, our objective is to find a representation for videos that fit this concept: first, with combinations of deep neural networks that extract and aggregate information over the various time scales involved in video data; then, by compressing this information to an extreme degree via hashing to fit real-time requirements.

4.1 Nearest Neighbor Search

4.1.1 Definition

The problem of retrieving relevant entries from a database is a problem of *abstract semantics* (in our case, visual similarity with respect to surgical workflow). Nearest neighbors search attempts to solve it by turning it into a *geometry problem*; more specifically, a distance optimization problem. We assume we have at our disposal a set of data entries represented by data points $E = x_1, ... x_N$ in a metric space Ω with a distance function D. The goal is to find a function ϕ verifying, for any query point $u \in \Omega$,

$$\phi : \begin{cases} \Omega \to E \\ u \mapsto \underset{x \in E}{\operatorname{argmin}}\ D(u, x) \end{cases} \tag{4.1}$$

ϕ retrieves the closest data point, with the underlying assumption that *similar entries semantically are close geometrically.* Geometrically speaking, if $\Omega = \mathbb{R}^d$ and D is the Euclidean distance, then ϕ partitions Ω into N convex polyhedrons $(Y_1, ... Y_N)$. The face separating two adjacent polyhedrons Y_i, Y_j belongs to the plane passing through $\frac{1}{2}(x_i + x_j)$ orthogonal to $x_j - x_i$. This set of polyhedrons, or cells, is known as the **Voronoï tesselation** (Figure 4.1) for E.

The Nearest Neighbor problem can be generalized as such: given the same query x, we look for Ψ verifying

$$\Psi : \begin{cases} \Omega \to \sigma(E) \\ u \mapsto \pi \mid \pi(x_0) \leq \pi(x_1) \leq ... \leq \pi(x_N) \end{cases} \tag{4.2}$$

where σ is the set of permutations over E. In other words, Ψ returns a ranking of E's items from closest to furthest to x. One can also only consider the top k items; formally:

$$\xi : \begin{cases} \Omega \to E^k \\ u \mapsto (\Psi(u)(x_0), ... \Psi(u)(x_k)) \end{cases} \tag{4.3}$$

This is the **k-NN search** problem, which serves as the foundation for the **k-NN classifier** in machine learning; this method simply outputs the majority class among the top k closest retrieved entries.

4.1.2 Challenges & complexity

A straightforward approach for computing k nearest neighbors consists in computing the distance of every data point from the query before sorting. Known as linear search,

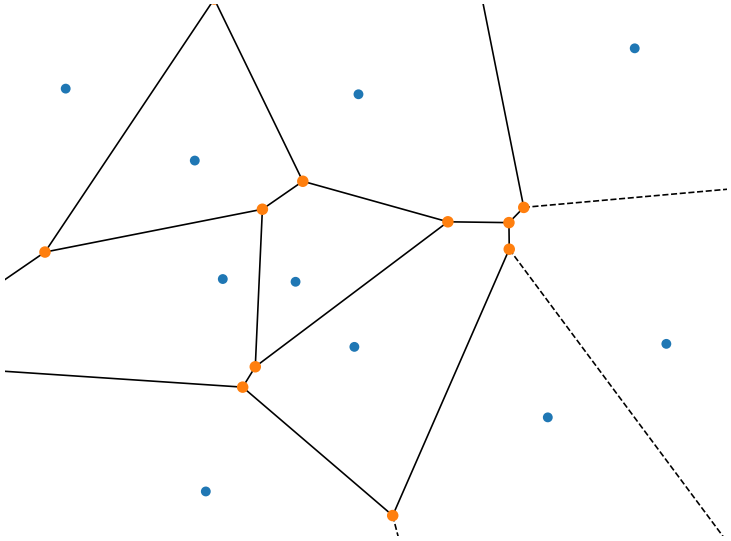


Figure 4.1: Voronoï tessellation of a set of data points (blue). Any point in a given convex polygon has the corresponding blue data point as its nearest neighbor.

this method has time complexity $\mathcal{O}(dN)$ in floating point operations, where d is the dimension of the data points and N the size of the database.

With this complexity in mind, we can immediately point out two major challenges for retrieval in our case:

- Typical values for N, in our datasets, fall into the $10^3 \sim 10^4$ range - for instance FCVID contains 91000 videos. Real-life video databases, however, can exceed this quantity by several orders of magnitude. With approximately 700000 cholecystectomies and 200000 bariatric surgeries per year in the United States alone, a database obtained by recording all procedures over a 10-year period would likely fall into the $10^6 \sim 10^7$ range. For public video platforms such as Youtube, with 500 hours of content uploaded every minute [Woj20], upwards of 10^9 would be a safe estimate.

- The working assumption for nearest neighbor search is that *similar entries semantically are close geometrically*. Finding an appropriate space for data points, with reasonably low d, where the distance function D presents this property is a difficult task in the case of videos. The original dimension of video data in pixel space, assuming $224 \cdot 224$ resolution, 25 fps -our preprocessing for the FCVID dataset- and 134s duration, almost reaches $1.7 \cdot 10^8$

We can therefore conclude that **the larger the database, the stronger the incentive to find low-dimensional, semantically rich representations of the data considered**.

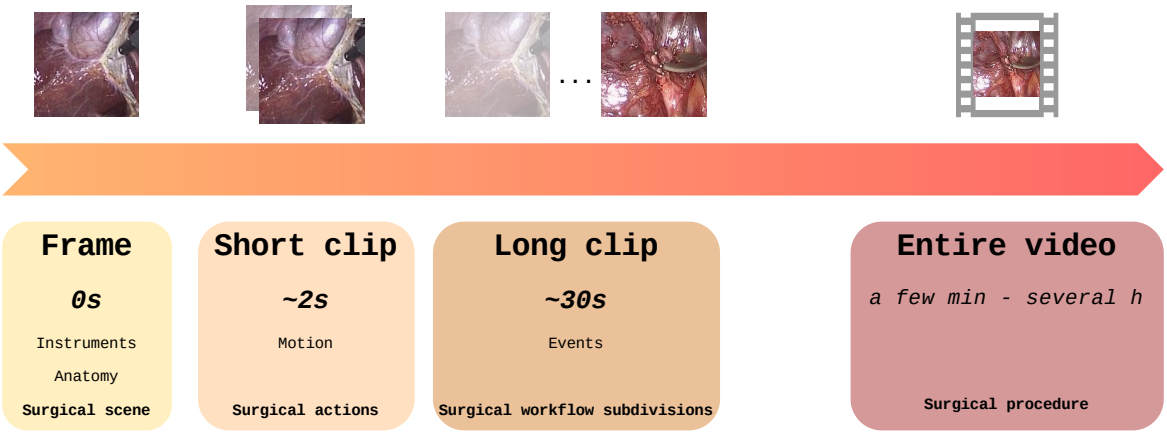


Figure 4.2: Temporal scales involved in surgical video processing.

4.2 Video representation learning with CNN-RNNs

4.2.1 General concepts

Learning rich and compact representations of its training data is the core task of a deep neural network (DNN). Assuming a single-class classification task with M classes and a perfect classifier f of the data:

$$f : \begin{cases} \Omega \to [0,1]^M \\ x \mapsto \mathbb{1}_{i \in [\![1,M]\!]}(x \in c_i) \end{cases} \tag{4.4}$$

We look for an approximation $\hat{f}$ in the form of a DNN that verifies, for some loss function $\mathcal{L}$:

$$\hat{f} = \underset{g}{argmin} \int_{x \in \Omega} \mathcal{L}(f(x), g(x))dx \tag{4.5}$$

Generally speaking, $\hat{f}$ can be separated into a **deep feature extractor** $\hat{f}_F$ and a **linear classifier** $\hat{f}_L$, such that $\hat{f} = \hat{f}_L \circ \hat{f}_F$. While learning $\hat{f}_L$ is trivial, the bulk of the difficulty (non-linearity of f, high input dimensionality) is carried by $\hat{f}_F$, which has to learn a **semantically rich, compact representation** of the input data for the classifier.

In the case of surgical videos this is a particularly challenging task, due to multiple levels of modeling (Figure 4.2) to account for. To our knowledge, no single model type is capable of adequately covering them all, which is why we resort to DNN combinations described below.

4.2.2 CNN

In its simplest form (ignoring maximum pooling layers or residual connections), a Convolutional Neural Network or CNN applies a succession of *convolutional layers*, followed by the final linear classifier: we have $\hat{f}(x) = \hat{f}_L \circ \hat{f}_{C,J} \circ \hat{f}_{C,J-1} ... \circ \hat{f}_{C,1}(x)$ where J is the number of layers and x is an $H \times W \times C$ input image.

Each convolutional layer $\hat{f}_{C,j} : \mathbb{R}^{H_j} \times \mathbb{R}^{W_j} \times \mathbb{R}^{D_j} \to \mathbb{R}^{H_{j+1}} \times \mathbb{R}^{W_{j+1}} \times \mathbb{R}^{D_{j+1}}$ is a mapping between two stacks of 2D feature slices. **Each slice in the output is**

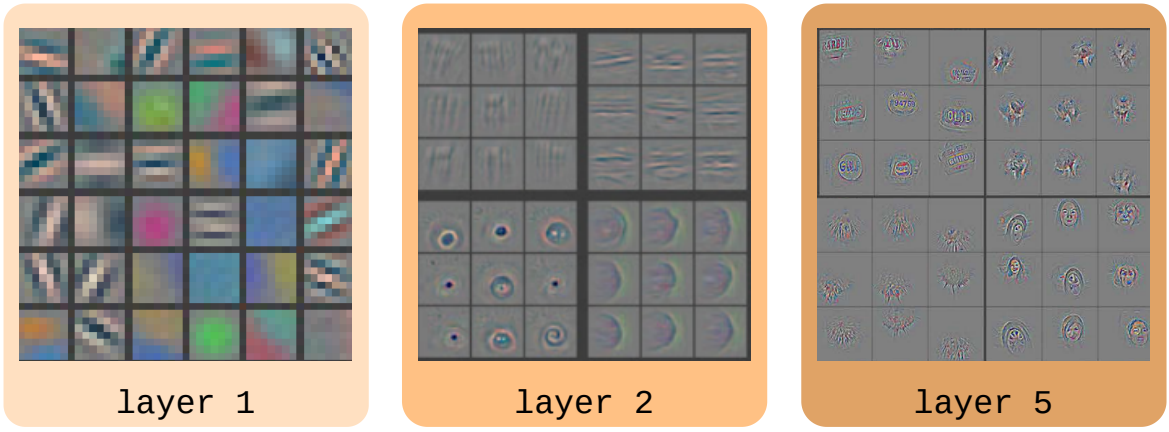


Figure 4.3: Features from a trained AlexNet-like CNN. Lower layers capture lower-level information such as color, texture and edges, while higher layers refer to recognizable objects. Image credits: Zeiler et al.

obtained by passing the input slices through linear 2D filters then summing before applying a bias:

$$\hat{f}_{C,j}(x)_{\blacksquare,\blacksquare,r} = ReLu(\sum_{r'=1}^{D_j} x_{\blacksquare,\blacksquare,r} \circledast \mathcal{K}_{\blacksquare,\blacksquare,r'} + \beta_{\blacksquare,\blacksquare,r}) \tag{4.6}$$

with $\mathcal{K} \in \mathbb{R}^{R_H} \times \mathbb{R}^{R_W} \times \mathbb{R}^{D_{j+1}}$ the *convolutional kernel*, β the bias and $\circledast$ the convolution operation (adequate padding of the input is implied). This use of convolutions ensures **the representations learnt in the CNN preserve the spatial structure found in the input**: each feature vector in the output $\hat{f}_{C,j}(x)_{p,q,\blacksquare}$ is a function of a 2D patch from the input around the same location, the size of the patch being the size of $\mathcal{K}$'s **receptive field** $\mathbb{R}^{R_H} \times \mathbb{R}^{R_W}$. Additionally, features learnt by convolutional layers have appealing visual properties: invariance to small translations is built-in, while other forms of invariance can be enforced using data augmentation [Bai93].

The nature of the learnt representations gradually evolves as they move up the CNN's layers, from low-level local properties related to color and texture and edges, to the high-level semantic properties useful for classification according to Zeiler et al. [ZF14]. A example of visualization of those features is given in Figure 4.3

4.2.3 3D CNN

The principles for learning visual representations from images with CNNs can be extended to *short* video clips as well. 3D CNN architectures, introduced in [TBF+15], rely on stacked convolutional layers as well:

$$\hat{f}_{C,j}(x)_{\blacksquare,\blacksquare,\blacksquare,r} = ReLu(\sum_{r'=1}^{D_j} x_{\blacksquare,\blacksquare,\blacksquare,r} \boxdot \mathcal{K}_{\blacksquare,\blacksquare,\blacksquare,r'} + \beta_{\blacksquare,\blacksquare,\blacksquare,r}) \tag{4.7}$$

The added dimension is of course **time**. The 2D convolution operator $\circledast$ is therefore replaced with the 3D convolution $\boxdot$. $\mathcal{K}$, as a set of spatio-temporal filters, can capture temporal variations between frames such as object motion. In practice however, the

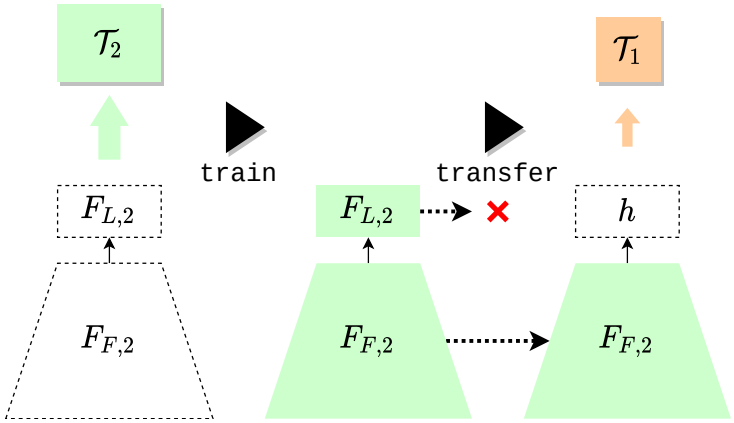


Figure 4.4: Principle of transfer learning.

number of timesteps is subject to memory limitations, since all frames in the video clip are simultaneously processed. Typical inputs [CZ17] contain up to 64 frames sampled at 25 fps, making up barely over 2 seconds of footage.

4.2.4 Transfer learning

Parameter counts for CNNs typically fall into the 10^7 range [HZRS16], often making them difficult to train from scratch (i.e. starting from randomly initialized weights) to perform a task $\mathcal{T}_1$ for which low amounts of annotated data are available. **Transfer learning** attempts to circumvent this issue by repurposing a model already trained on a different task $\mathcal{T}_2$. Taking for example a CNN $\hat{f}_2$ trained on $\mathcal{T}_2$, in the general case the decomposition $\hat{f}_2 = \hat{f}_{L,2} \circ \hat{f}_{F,2}$ applies, with $\hat{f}_{L,2}$ as the linear classifier and $\hat{f}_{F,2}$ as the **feature extractor** or *backbone*. The linear classifier $\hat{f}_{L,2}$ is mostly task-specific; however one can take advantage of the representation learnt by the linear classifier $\hat{f}_{F,2}$ in order to obtain a well-initialized model for $\mathcal{T}_1$, by only grafting a head $\mathbb{h}$ to it as shown in Figure 4.4:

$$\hat{f}_1 = \mathbb{h} \circ \hat{f}_{F,2} \tag{4.8}$$

Training the entirety of this model on $\mathcal{T}_1$ is a lightweight process, since $\hat{f}_{F,2}$'s weights are already considered close to optimal. Generally performed with low learning rate values for a relatively small number of epochs, this is commonly referred to as *fine-tuning*. If the dataset involved in $\mathcal{T}_1$ is particularly small, the common practice is to freeze $\hat{f}_{F,2}$ in part or in its entirety, in order to prevent overfitting.

The key assumption in this approach is that $\mathcal{T}_2$ is *visually generic*, so that features learnt by $\hat{f}_{F,2}$ generalize well to other tasks. In large-scale image or video benchmarks it is generally assumed to be the case; therefore in our work, we used a **Resnet-50 CNN** [HZRS16] **pretrained on the ImageNet** [DDS+09] dataset, as well as an **I3D backbone pretrained on Kinetics** [CZ17] in later chapters.

An important observation is that the head $\mathbb{h}$ is not necessarily another linear classifier, as evidenced by the next section.

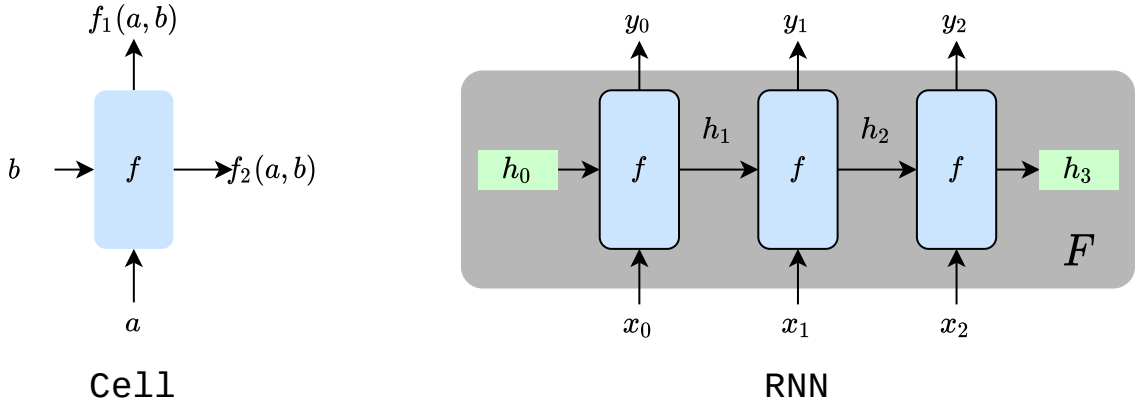


Figure 4.5: Recurrent cell function, and its deployment into an RNN.

4.2.5 RNN

In our approaches to video processing, CNN backbones handle the scale of frames or short clips; when moving to larger time scales, we add **Recurrent Neural Network** models, which we explain here.

Considering a 2-to-2 vector function:

$$f : \begin{cases} (\mathbb{I}, \mathbb{H}) \to (\mathbb{O}, \mathbb{H}) \\ (a, b) \mapsto (f_1(a, b), f_2(a, b)) \end{cases} \tag{4.9}$$

and an initial state h_0, the Recurrent Neural Network (Figure 4.5) corresponding to f is defined as:

$$F : \begin{cases} \mathbb{I}^* \to \mathbb{O}^* \\ (x_0, ... \, x_T) \mapsto (y_0, ... \, y_T) \end{cases} \tag{4.10}$$

with, for any $t \in [\![0, T]\!]$:

$$(y_t, h_{t+1}) = f(x_t, h_t) \tag{4.11}$$

This recurrent equation updates the internal temporal representation or state h_t to h_{t+1}, from one timestep to the next. The length of the sequence T is arbitrary; therefore inputs of any length in time can be processed. Since h_t depends on all previous inputs $x_0 ... \, x_t$, **the state functions as the RNN's memory or internal summary of past events**; this is a temporal representation learnt by the RNN.

f's parameters - noted W - are learnt via *backpropagation through time*. We consider an error $E(y_t)$ to minimize, computed from f's last output. $\frac{\partial E}{\partial W}$ depends on $\frac{\partial h_t}{\partial W}$, which itself depends on recurrent gradients $\frac{\partial h_{j+1}}{\partial h_j}$ for $j \in [\![0, T-1]\!]$; each of these terms can be computed using the next one via the **chain rule**, which facilitates training.

Similar to stacked layers in CNNs or multilayer perceptrons, multiple recurrent neural networks $F_1 ... \, F_L$ can form a single **stacked RNN** F with function composition (Figure 4.6), as $F = F_L \circ F_{L-1} ... \, \circ F_1$. This means, for a given timestep, the external output from a layer is forwarded to the external input of the next one. According to Graves et al. [GMH13] this provides "depth in space" for the RNN's

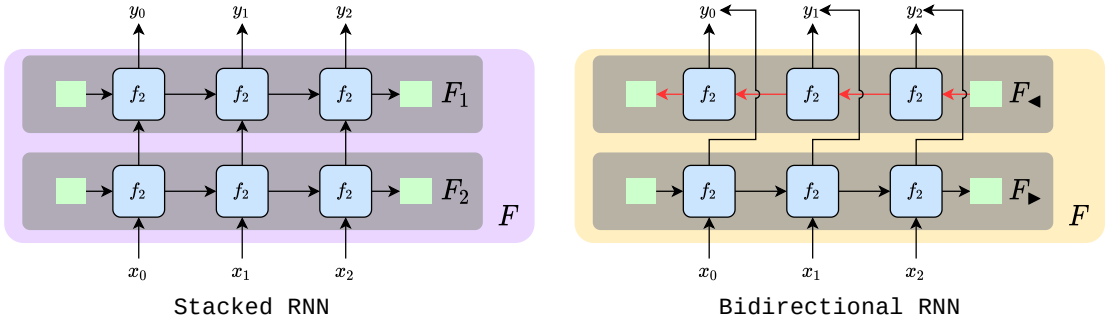


Figure 4.6: RNN variants. Left: stacked RNN. Right: bidirectional RNN.

learnt representations, in addition to "depth in time", which is naturally obtained by forwarding the state.

Another classic enhancement comes in the form of the **bidirectional RNN** (Figure 4.6). Two RNNs $F_{\blacktriangleright}, F_{\blacktriangleleft}$ process the input sequence in opposite orders; at any given timestep j, their outputs $y_{\blacktriangleright,j}$ and $y_{\blacktriangleleft,j}$ are concatenated together into a single output y_j, which is a function of both states $h_{\blacktriangleright,j}$ and $h_{\blacktriangleright,j}$, accounting for past and future events respectively. The entire temporal context is therefore available throughout the sequence - the downside, as previously stated in Chapter 3, is the inability to use this model in real time.

4.2.6 LSTM

A commonly used model for f is the LSTM (Figure 4.7) [HS97], defined by the following set of equations:

$$\begin{cases} f_t & = \sigma(W_{x,f} \cdot x_t + W_{h,f} \cdot h_t + b_f) \\ i_t & = \sigma(W_{x,i} \cdot x_t + W_{h,i} \cdot h_t + b_i) \\ g_t & = tanh(W_{x,g} \cdot x_t + W_{h,g} \cdot h_t + b_g) \\ o_t & = \sigma(W_{x,o} \cdot x_t + W_{h,o} \cdot h_t + b_o) \\ c_{t+1} & = c_t * f_t + i_t * g_t \\ h_{t+1} & = o_t * tanh(c_{t+1}) \\ y_t & = h_{t+1} \end{cases}$$

"$\cdot$" is the dot product, while "$*$" is the element-wise multiplication. The internal state is composed of two vectors (c_t, h_t), named respectively cell state and hidden state; the hidden state is also used as the output y_t. Optionally, the state h_t can be treated with Ioffe et al's **batch normalization** [IS15] in order to improve the model's convergence and generalizability. While most common in CNNs, its use has previously been seen in LSTM models as well [CBL+17, ZWHC16].

The total number of parameters is:

$$N_{prm} = 4 \cdot dim(x) \cdot dim(h) + 4 \cdot dim(h)^2 + 4 \cdot dim(h) \tag{4.12}$$

while the number of floating-point operations for one iteration is:

$$N_{op} = 4 \cdot dim(x) \cdot dim(h) + 4 \cdot dim(h)^2 + 17 \cdot dim(h) \tag{4.13}$$

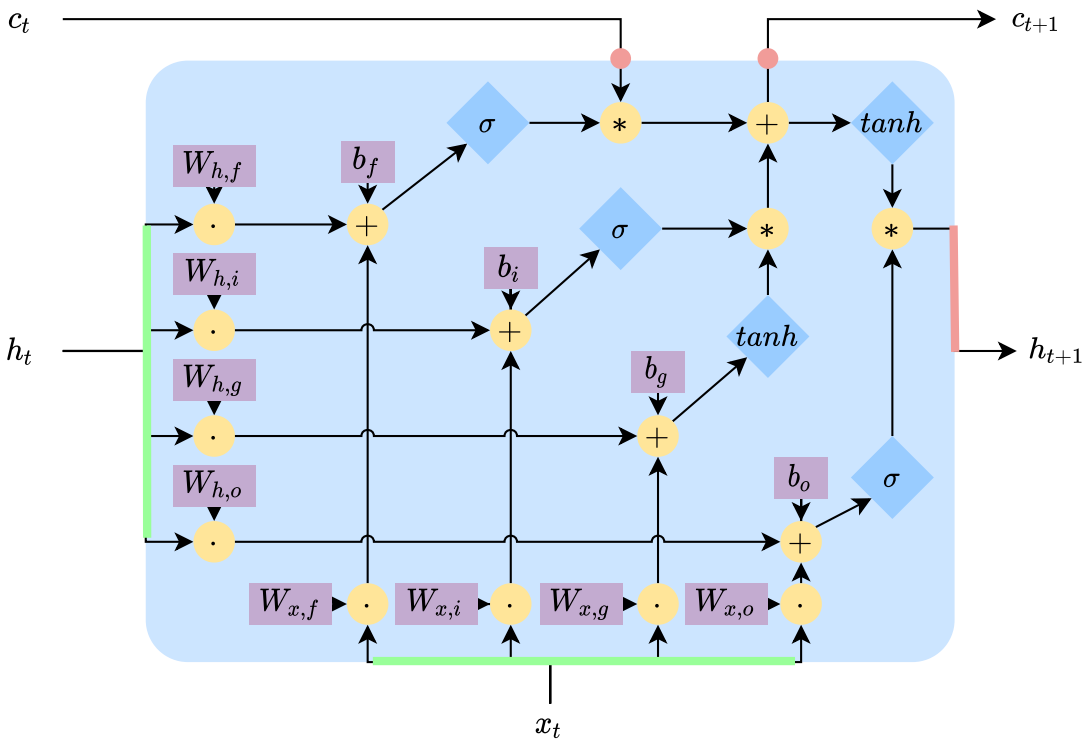


Figure 4.7: LSTM computation.

While, to our knowledge, no formal constraints exist for dimensions $dim(x)$ and $dim(h)$, values for those overwhelmingly tend to fall into the $10^1 \sim 10^3$ order of magnitude. Direct usage on video frames ($10^5 \sim 10^6$) is generally ruled out.

It has been shown [HS97] that, during the training process, the LSTM learns to write into the state the information needed for predictions at much later timesteps; hence "long term". According to Tallec et al. [TO18], the gate values f_t and i_t act as "time contraction or time dilation coefficients", rendering the LSTM's learnt representations "quasi-invariant" to time warping.

4.2.7 Seq2Seq architectures

A sequence-to-sequence mapping is a function $f : A^* \to B^*$, with A and B vector spaces of finite dimension and A^*, B^* the corresponding spaces of sequences. The key consideration here is that **the input and output sequence lengths are arbitrary**; approximating f therefore requires a high degree of flexibility in the model's architecture.

Some of that flexibility can be provided by a Recurrent Neural Network, which, on its own, naturally maps a sequence to another sequence. However in this approach outputs and inputs are matched timestep for timestep, which constrains input and output sequence lengths to be identical. Limited input visibility is also an issue: for machine translation tasks, the information required by the target language in the middle of a sentence found at a further location in the source language sentence.

Sequence-to-sequence learning [SVL14] or Seq2seq refers to a family of models designed to solve those issues (Figure 4.8). They consist in a pair of RNNs, one

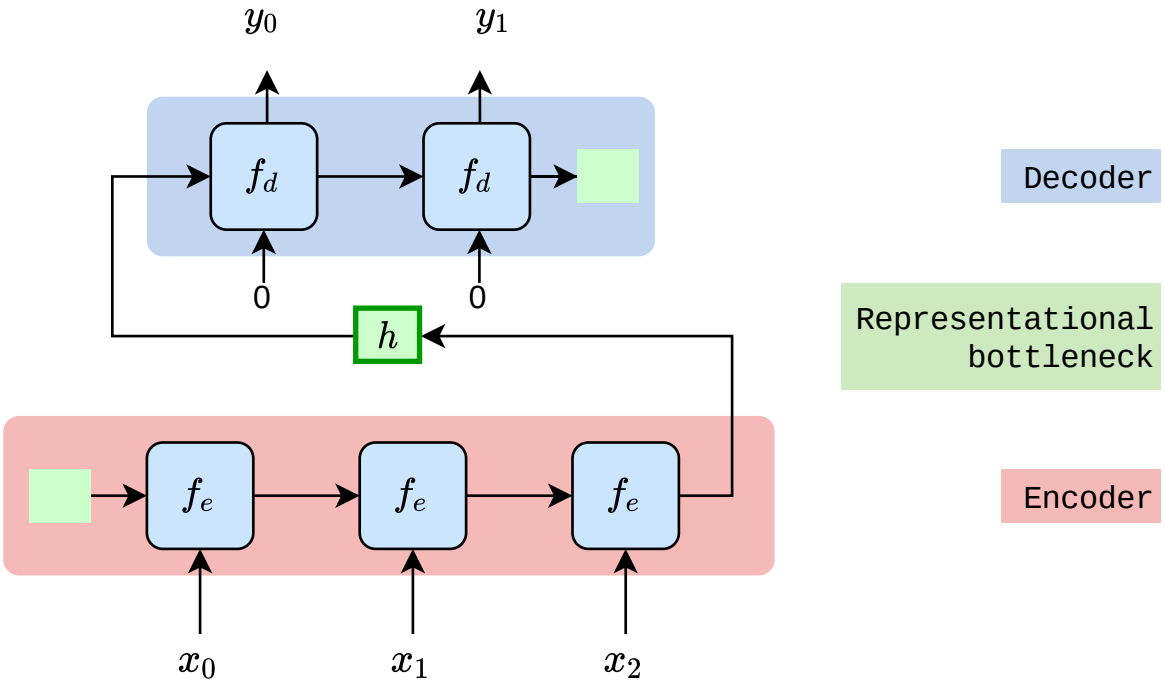


Figure 4.8: Seq2Seq architecture.

encoder and one decoder: the encoder reads the input sequence from beginning to end. The final state h_T is then forwarded to the decoder, which sets it as its own initial state. From there, the decoder generates the target sequence, without any external inputs - for example the original Seq2seq publication connects the external input of the previous timestep to the output of the previous one.

Here h_T acts as a *representational bottleneck*. All the information required by the decoder to generate the target sequence needs to be written into h_T by the encoder, regardless of input or output sequence length.

Combined with a pretrained CNN used as a feature extractor, this results in a model capable of generating compact video representations, accounting for all temporal scales involved in surgical videos.

4.3 Hashing

The video representation obtained with ordinary CNN-RNNs is the RNN's memory, i.e. a floating-point vector of a few hundred coordinates in general. Hashing enables compressing this representation to an extreme degree, resulting in binary arrays that are convenient for indexing.

4.3.1 Hash functions: definitions, basic properties

The term "hashing" refers to a family of methods employed for computing a fixed-size value called a *hash* from a given data input, which the hash will then identify - similar to a signature or a fingerprint. Formally speaking, a hash function is a mapping

$$\mathfrak{h} : E^* \to \Sigma^n \tag{4.14}$$

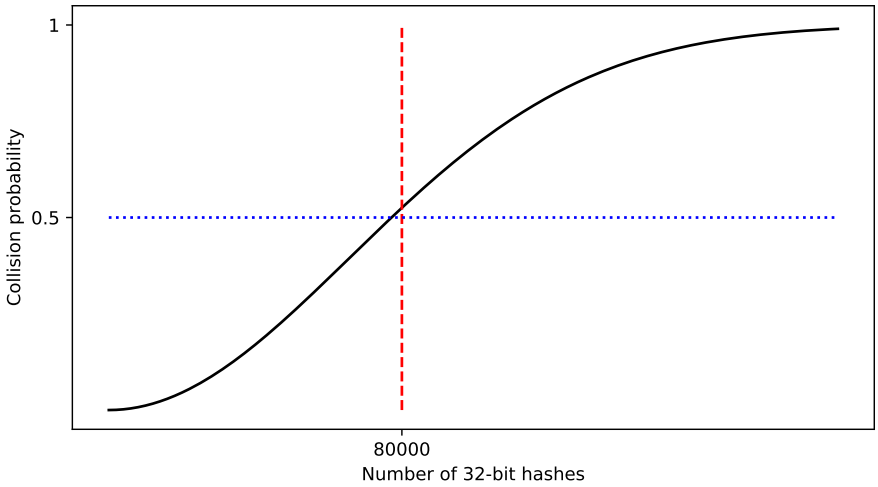


Figure 4.9: Collision probability for hashes of size 32. Collision odds quickly exceed 50%.

with n being the hash length. From this point onwards we will consider $\Sigma = \{-1, 1\}$ i.e. binary hashes, or **bitcodes**. For example:

$$\mathfrak{h} : 0, 1^* \to 0, 1 \tag{4.15}$$

$$: (b_1 ... \; bT) \mapsto b_1 \otimes b_2 ... \; \otimes b_T \tag{4.16}$$

$\otimes$ denotes the logical XOR operator.$\mathfrak{h}$ is a basic single-bit hash function. We can already point out $\mathfrak{h}(1,0,1) = \mathfrak{h}(1,1,0)$; in general, any hash function is redundant. As a mapping from a set to a smaller one it indeed falls under the pigeonhole principle. Redundencies, i.e. pairs (a, b) such that $a \neq b, \mathfrak{h}(a) = \mathfrak{h}(b)$ are called *collisions*. Under uniformity assumptions, the probability of finding at least one collision in a set of k random data inputs is given by

$$P_C = 1 - \prod_{j=1}^{k-1} \frac{2^n - j}{2^j} \tag{4.17}$$

$$= 1 - \frac{2^n!}{2^{kn}(2^n - k)!} \tag{4.18}$$

Stirling's factorial approximation gives an asymptotic equivalent for P_C:

$$P_C = 1 - e^{-\frac{k^2}{2^{n+1}}} \tag{4.19}$$

Notably, when $k > \sqrt{2^{n+1} ln(2)} \sim 1.18 \cdot 2^{n/2}$, then the collision probability exceeds $\frac{1}{2}$; this is commonly known as the **Birthday Paradox**. This is an important principle in hashing: although many distinct bitcodes (2^n) are available, collisions are more likely than one might first expect. For instance 32-bit hashes offer $4 \cdot 10^9$ possible values, but collisions are more likely to occur than not in a set of only 80000 32-bit hashes (Figure 4.9). Along with computation speed, collision behavior is one of the main factors determining the choice of $\mathfrak{h}$. This heavily depends on $\mathfrak{h}$'s purpose: for encryption, accidental collisions should be avoided at all costs, which is why the chosen hash

functions for this use case exhibit extremely high sensitivity to input perturbations. This property is known the *Avalanche Effect*; formally, the Strict Avalanche Criterion dictates that one flipped bit in the input causes every bit in the output to flip with a 50% probability for each one, independently.

An example of a situation where this property is useful is file verification upon transfer - this is routinely performed using a hash function called `md5sum`. Instead of examining a file f for errors bit by bit, the file's receiver computes $\mathfrak{h}(f)$ and compares it against the original signature $\mathfrak{h}(f_{original})$. While a collision between a corrupted file and the original is possible ($f \neq f_{original}, \mathfrak{h}(f) = \mathfrak{h}(f_{original})$), it is so exceedingly improbable for reasonably large n that the received file is considered clean if the hashes match.

4.3.2 Retrieving data with hashing

The term *hashing* covers several families of tasks, with vastly different methods and applications. Retrieval with hashing is most commonly understood in the context of hash tables, which we will explain here in order to clear potential confusions with similarity-preserving hashing. Hash tables are data structures storing values to be retrieved with a corresponding key. Unlike ordinary arrays, which are indexed by integer indices, the data type for the key is arbitrary. Well-known implementations of this data structure include C++'s `std::unordered_map` or Python's `dict`.

During storage, a hash function hashes the key - the resulting code is the address in an array where the corresponding value is stored. Retrieving can then be done using the same key, which, evaluated by the hash function, immediately gives the exact location to retrieve from in the array. This makes retrieval $O(1)$. This is much faster than retrieving from an ordinary array with n (key, value) pairs, which takes at best $O(ln(n))$ comparisons if sorted by keys, $O(n)$ otherwise.

Once again collisions are to be avoided; if two keys share the same hash, the value to retrieve is ambiguous. In those cases an ad hoc collision resolution routine is performed; querying with unseen keys is also forbidden.

In **content-based data retrieval**, the requirement for hash functions is radically different (Figure 4.10). *Similarity preservation* is crucial here; data entries that are very distinct from one another should produce many conflicting bits. Conversely, data entries with a high degree of similarity should produce similar bitcodes or even collide, a property considered unacceptable in cryptographic hashing or exact retrieval. This is formalized by the *locality-sensitive* property [AI08], as stated in [IM98]:

$$d(a,b) \leq r_1 \Rightarrow P(\mathfrak{h}(a) = \mathfrak{h}(b)) \geq p_1 \tag{4.20}$$

$$d(a,b) > r_2 \Rightarrow P(\mathfrak{h}(a) = \mathfrak{h}(b)) \leq p_2 \tag{4.21}$$

The choice of distance d in input space is of course crucial and to be discussed later on. In bitcode space we use the *hamming distance*, defined as

$$d(a,b) = \sum_{j=0}^{n} \mathbb{1}(a_j \neq b_j) \tag{4.22}$$

This is simply the number of conflicting bits between the two codes.

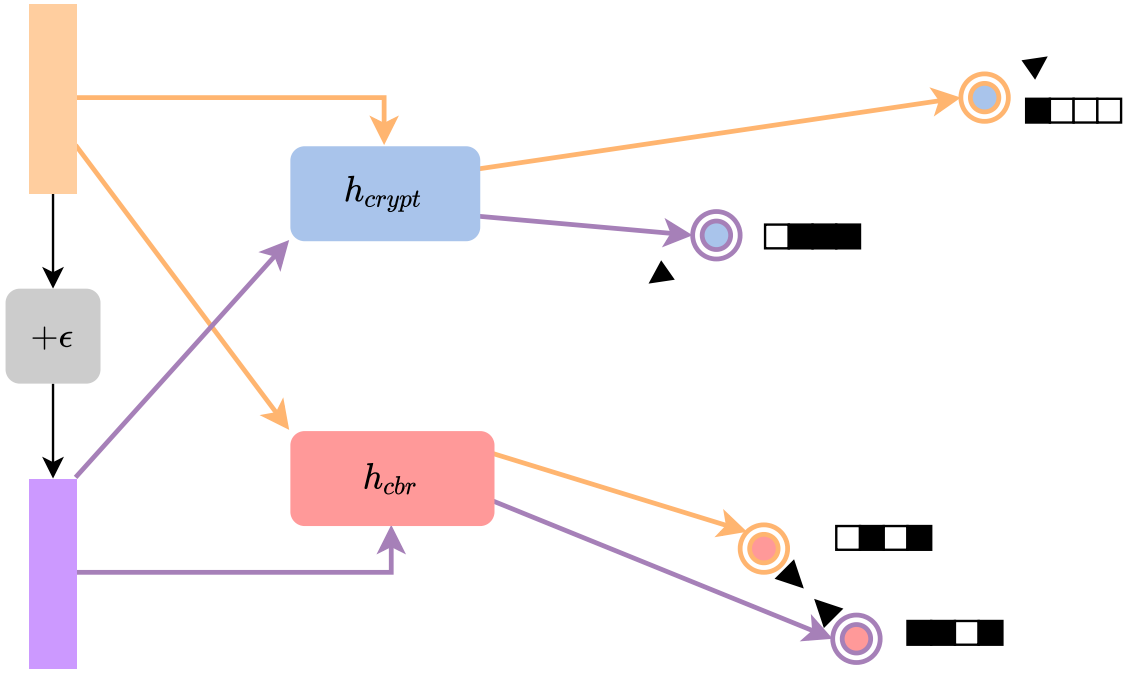


Figure 4.10: Cryptographic and locality-sensitive hash function purposes. The two types of hash functions have opposite priorities.

Assuming $\mathfrak{h}$ exhibits decent locality-sensitive properties, one can use it to perform Approximate Nearest Neighbors Search [IM98], as a substitute for the costly exact nearest neighbors previously mentioned. While exact retrieval is $O(1)$ (no comparisons), approximate nearest neighbors retrieval requires $O(n)$ distance computations just like the exact version. However depending on the implementation the corresponding computation can be extremely quick; using true bits, a single bitwise XOR operation leads to the result, instead of 1 floating-point operation for every vector component. This solution is much more storage-efficient as well.

4.3.3 Challenges of learning hash functions

Finding a good candidate for h is a difficult task. Assuming a set of data points to hash $x_1 ... x_N$

Finding h with the best similarity-preserving properties can be formulated as the following discrete optimization problem:

$$\underset{b_1 ... b_N}{argmin} \sum_{i,j \in [\![1,N]\!]} d(x_i, x_j)|b_i - b_j|^2 \tag{4.23}$$

under:

$$b_i \in \{-1, 1\}^n \text{ (binary constraint)} \tag{4.24}$$

$$\sum_{i=1}^{n} b_i = (0, ... 0) \text{ (even bit distribution)} \tag{4.25}$$

$$\frac{1}{n} \sum_{i=0}^{n} b_i b_i^T = 0 \text{ (decorrelated bits)} \tag{4.26}$$

This is known to be NP-hard [IM98]; with a large database of inputs, solving it quickly becomes impractical.

As mentioned earlier, $\mathfrak{b}(x)$ should, for any x, contain rich discriminative information extracted from x. It is therefore natural to turn to learning-based approaches, considering their success extracting information from complex and high-dimensional input data. We now assume $\mathfrak{b}$ is a function with parameters W - noted $\mathfrak{b}_W$ from now on.

The immediate issue we encounter is that, strictly speaking, $\mathfrak{b}_W$ is not differentiable. Indeed, it is not even continuous.

While h can be differentiated outside of its discontinuity points, it is locally constant there, and the result will therefore always be a zero derivative. When backpropagating to the parameters W this is an issue; as an effect of the chain rule, $\vec{\nabla}_W L(\mathfrak{b}_W(x)) = 0$. Workarounds, however, do exist. First, we can look for a relaxation in the form of

$$\mathfrak{b}_W(x) = sgn(A \cdot g_V(x)) \tag{4.27}$$

with sgn mapping < 0 inputs to -1, and the rest to +1. g_V is a differentiable function with learnable parameters V, A is a projection matrix. We will refer to the argument of sgn as the *prebitcode* $\hat{b}$.

It is now clear how learning hash functions can be posed as a representation learning problem, i.e. mapping the data to a feature vector space that can be linearly partitioned by N affine hyperplanes P_i. Each column A_i of A is a normal vector of one of the P_i hyperplanes, and the position of the input in feature space relative to the hyperplane determines the corresponding bit value.

Training g is a real-valued optimization problem instead of a discrete one, and can be solved using gradient descent. Using conventional approaches for training deep neural networks, we can learn a function that will incorporate rich, discriminative information into $\hat{b}$. This does, however, result in a certain amount of quantization error which can be measured by [GLGP13]:

$$\mathbb{Q}(\mathfrak{b}, x_1 ... \, x_N) = \sum_{i=1}^{N} ||b_i - A \cdot g(x_i)||^2 \tag{4.28}$$

Prebitcode positioning with respect to the zero line in each component is crucial in managing this error. Methods such as ITQ [GLGP13] or UDVH [WLG+17, WHG+19] try to enforce this with geometric transformations. However this can be handled by the training process itself in an elegant and end-to-end manner. First, batch normalization [IS15], by centering activations around zero, achieves this effect. Second, although sgn is not differentiable, a workaround is given in [HCS+16].

We can introduce:

$$f : \begin{cases} x \in]-1, -\infty] & \mapsto -1 \\ x \in [-1, 1] & \mapsto x \\ x \in [1, +\infty[& \mapsto 1 \end{cases} \tag{4.29}$$

This is sometimes referred to as *hardtanh*, due to similar behavior near $-\infty$, 0 and ∞. Then:

$$\vec{\nabla}^*_W(\mathfrak{b}(x)) = \mathbb{1}_{[-1,1]}(A \cdot g_v(x)) \cdot A \cdot \vec{\nabla}_V(g_V(x)) \tag{4.30}$$

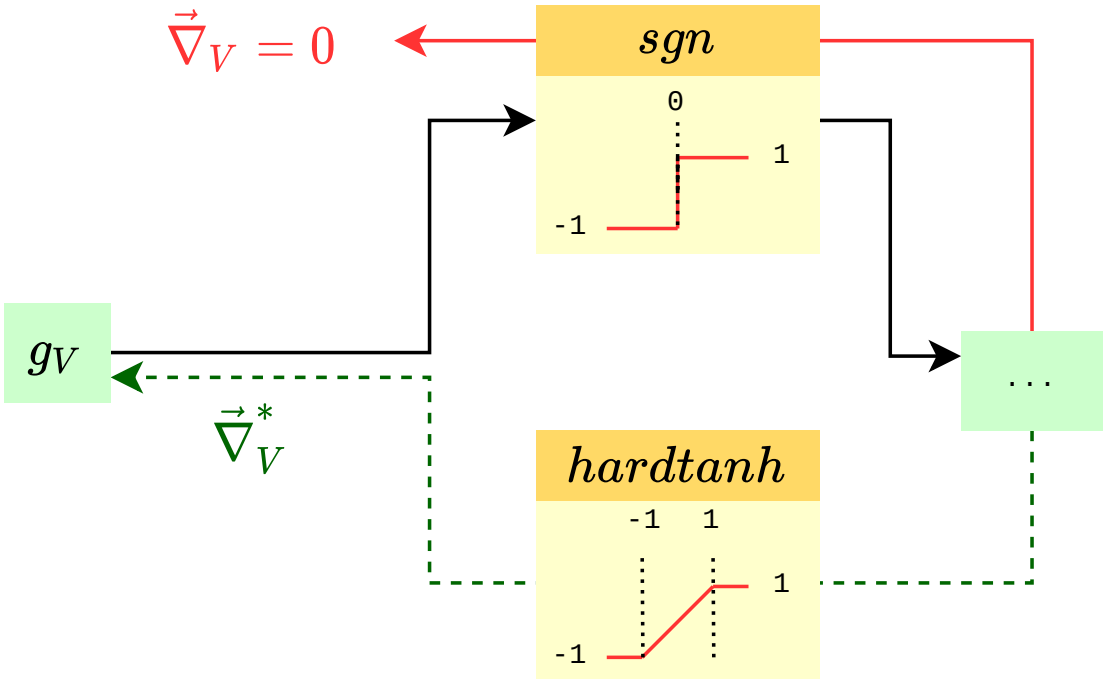


Figure 4.11: Workaround for training a network with binary outputs.

Forward pass is done with a true *sgn*, while backpropagation replaces the gradient with $\vec{\nabla}^*_W(\mathfrak{b}(x))$ (Figure 4.11).Experimentally, Zhang et al. [ZWHC16] observed that this training method, along with **batch normalization**, improved prebitcode spread around zero in their modified LSTM.

CHAPTER 5

Real-time retrieval by hashing live video sources

"Real generosity towards the future lies in giving all to the present."

- Albert Camus, *Notebooks, 1935 - 1942*

Contents

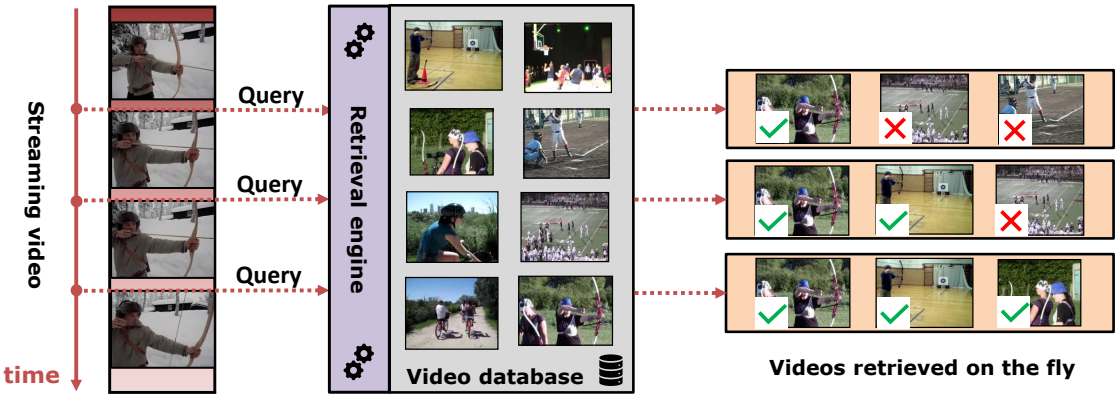


Figure 5.1: Real-time video retrieval concept. An archery event is being streamed. Simultaneously, a search engine scans a large video database for similar content.

5.1 Objectives

As established in the previous chapter, learning video representations for large-scale retrieval is a challenging problem: information from a wide range of temporal scales needs to be accounted for, and undergo extreme levels of compression. Here we introduce an additional technical difficulty, by examining the case of **live video sources**. Obvious examples include TV event broadcasts, livestreams from platforms such as Youtube Live or Twitch. However, those also encapsulate a wide variety of settings and devices: from surveillance cameras and dashcams to personal smartphones, and medical video capture devices such as endoscopes.

The difficulty of this task comes from two major challenges:

1. Search needs to be **fast** enough for real-time use. If too slow to compute, search results risk becoming irrelevant to the content currently displayed. This constraint is extremely harsh: large-scale video databases can consist of tens of thousands if not millions of items, each one the size of many static images.

2. When streaming in real time, the video content is *de facto* **incomplete**. At the beginning of a video especially, only a fraction of the total information is available.

As stated in Chapter 2 the previous literature on video retrieval does not fit the requirements of real-time usage. The few recent works involving deep computer vision models [ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19] only retain a fixed, very small ($N_{frames} = 20 \text{or} 25$) number of evenly spaced frames out of the entire video. In addition to destroying most of the visual content, this process is also impossible to exactly replicate live, since the sampling rate of $\frac{\text{Total duration}}{N_{frames}}$ would be unknown.

5.2 Methods

5.2.1 Setup and data preparation

5.2.2 Overview

The problem can be formally stated as follows: given a **level of observation** $\alpha \in (0, 1]$, a video $\mathcal{Q}$ with duration $T(\mathcal{Q})$, a query $\mathcal{Q}_\alpha$ consisting of $\mathcal{Q}$ truncated mid-stream at time $\alpha T(\mathcal{Q})$, and a database of videos $\mathcal{B} = \{V_0...V_N\}$, use only $\mathcal{Q}_\alpha$ to find the K videos from $\mathcal{B}$ most similar to $\mathcal{Q}$.

Since hashing is the method employed here, $\mathcal{B}$ and $\mathcal{Q}$ are represented by binary codes during search operations. The overall protocol is therefore composed of the following steps:

- video-level feature extraction
- self-supervised hash function training
- codebook, query encoding
- search operations

5.2.2.1 Datasets

Experiments shown in this chapter involve two very large scale, public datasets of generic human activities. The first is FCVID [JWW+18], a public video dataset used for research on video retrieval [ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19] and activity understanding depicting various activities, objects, sports and events. 234 video classes are present. The average video duration is 134 ± 92 seconds. Data was split as follows: 45K for training, 45K for testing. Within the test videos, 42.5K constitute the database while the remaining 2.5k (about 10 videos per class on average) are used for querying. The second dataset we employed is ActivityNet [HEGN15]. We used the 18K videos available for download. 200 activity classes are featured. A number of videos had their labels concealed by the organizers of the ActivityNet challenge; those videos were put in the training set. The average duration is 117 ± 67 seconds. Data was split as follows: 9K for training, 9K for testing, and within the testing videos 8K for the codebook, 1K (about 5 per class on average) for the query. In both datasets, the videos are untrimmed, which implies that the content reflecting a given video's class may not be shown for its entire duration. All videos have been resampled at 30 fps and a few excessively long videos have been cropped to 4 min 30s; we then sort them into 12 duration buckets of 22s in diameter each.

5.2.3 Sampling & feature extraction

As mentioned in the related work section, previous approaches have chosen sparse, fixed-length video-level representations which can result in the loss of large amounts of visual information in long videos. More importantly, such sampling and feature extraction schemes are inadequate for real-time usage: the sampling rate in that case is dependent on the length of the full video, which would be unknown mid-stream. An

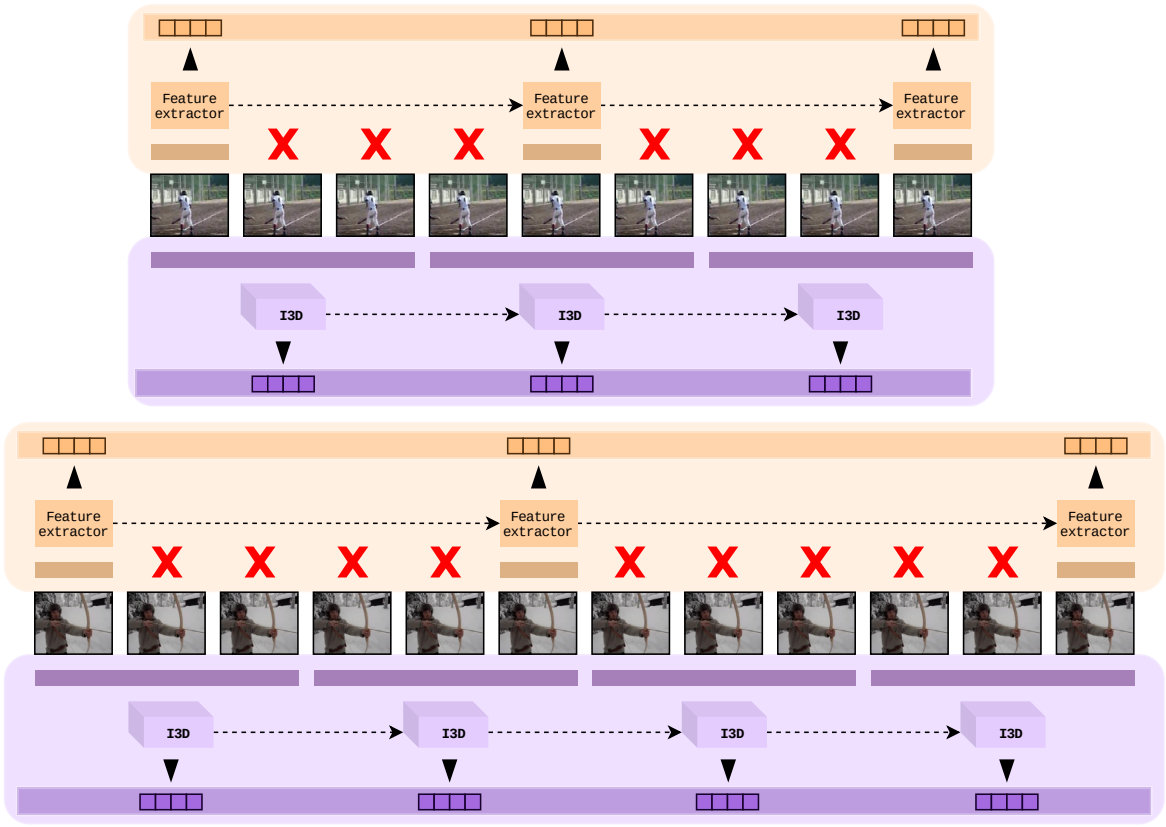


Figure 5.2: Comparison of video sampling approaches for two videos of distinct durations. The previous video sampling paradigm (orange) outputs a constant number of features per video, but the rate is variable and determined with knowledge of the entire video length. Our real time-compatible sampling approach (purple) outputs a variable number of features at a constant rate.

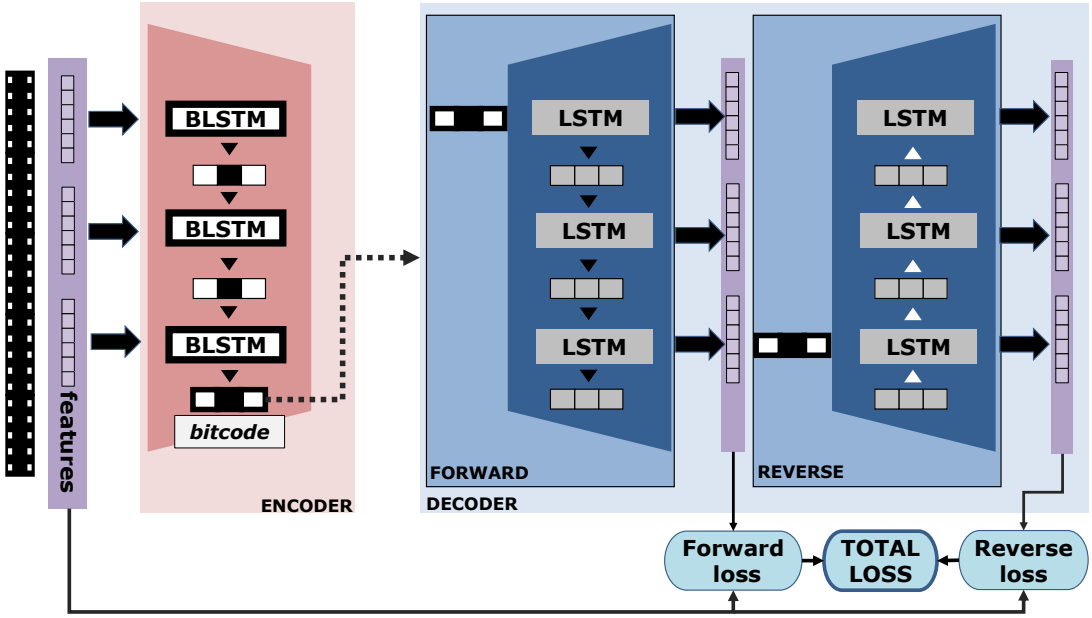


Figure 5.3: Binary autoencoder training process.

early retrieval algorithm should be capable of delivering search results at a fast and steady pace.

To solve this issue, we have chosen a dense, fixed-rate sampling and feature extraction scheme that leverages a 3D CNN to aggregate visual information over short time windows. The chosen architecture is I3D [CZ17], pretrained on the Kinetics dataset for action recognition. Input frames from the videos are center-cropped to 224×224. Clip input length is 64 frames (roughly 2 seconds at 30 fps). This yields a feature tensor with dimensions $8 \times 7 \times 7 \times 1024$, which we average pool into a feature vector of size $N_f = 1 \times 2 \times 2 \times 1024 = 4096$. The entire video is processed in consecutive clips, resulting in a $30 \ / \ 64 = 0.47$ Hz constant feature output rate.

5.2.4 Binary RNN encoder

An RNN (Figure 5.3) takes the feature vectors $f_0, f_1, ..., f_t, ..., f_{T(V)}$ from I3D in chronological order and maps each one to a bitcode of size N_{bits}. This RNN is built on two stacked LSTMs, with a binary LSTM as introduced in [ZWHC16]. The hidden state circulating from one RNN timestep t to the next is the above mentioned bitcode noted $b(V, t)$ for a video V. Batch normalization is applied to the cell state as done in [ZWHC16]. This RNN is trained as the encoder part of an autoencoder model. The final hidden state of the decoder - the video's bitcode - is passed to two decoder RNNs tasked with reconstructing the sequence of input feature vectors in opposite orders. This training process encourages the encoder to incorporate rich, discriminative information into the bitcodes.

Formally, the autoencoder is trained to minimize the following **unsupervised** loss function:

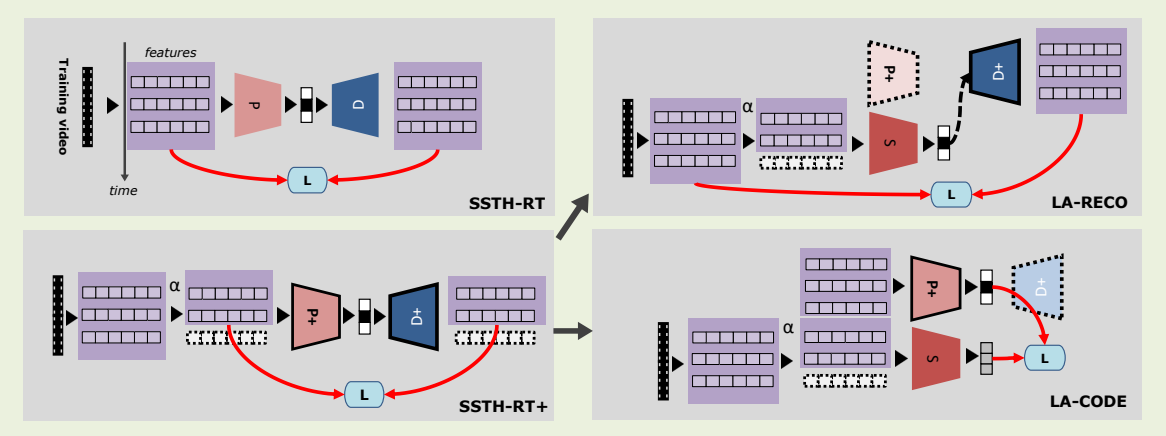


Figure 5.4: Training approaches compared for real-time video retrieval.

$$\mathcal{L}_{decoder}(V) = \sum_{j=0}^{T(V)} \|f_j - \overleftarrow{f}_j\|_2 + \|f_j - \overrightarrow{f}_j\|_2. \tag{5.1}$$

$\overleftarrow{f}$, $\overrightarrow{f}$ are the reverse and forward reconstructed sequences, respectively.

Backpropagation relies on the gradient substitution method shown in [ZWHC16]. Trained as such with full videos, we call this configuration `SSTH-RT` (Figure 5.4, top left) for real-time, as each new increment of the video playing in real-time is incorporated into the bitcode. This is, in the form presented here, our first baseline approach.

5.2.5 Data-augmentated encoder via truncated training duplicates

Since the training process for the baseline setting always involves complete videos, a straightforward data augmentation method for training the autoencoder is to simply give the autoencoder videos truncated at various levels of observation to reconstruct (Figure 5.4, left). This approach provides some training exposure to videos that are incomplete. This is referred to as `SSTH-RT`$^+$ in the later sections.

5.2.6 Augmented codebook with truncated database duplicates

Another idea to improve early retrieval results is to incorporate trimmed videos, both during training - as in `SSTH-RT`$^+$ - and codebook construction (Figure 5.4, center). We refer to this later on as `SSTH-RT`$^{++}$. While this idea might seem appealing at a first glance, it causes the search protocol to slow down due to the insertion of bitcodes from trimmed duplicates. This requires inserting as many duplicates per video as levels of observation employed - in our case: 10 from 0.1 to 1.0 - which is also a source of scalability issues. Assuming N_α levels of observation are employed, search time and storage space requirements both get multiplied by N_α. Duplicates in the search results, namely, videos truncated at different levels of observations from the same original video, would also need to be purged, the cost of which has $N_{cb} \times N_\alpha$

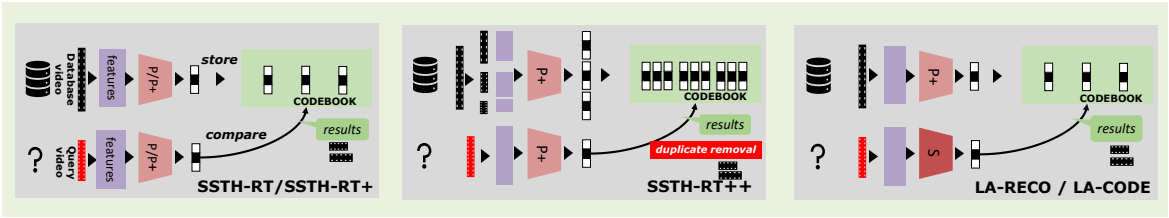


Figure 5.5: Evaluation protocols for real-time video retrieval.

complexity, N_{cb} being the size of the codebook without duplicates. Correctly retrieved results that are duplicates may improve retrieval metrics, but do not suit general application cases: a search operation returning truncated versions of the same video would be unhelpful to the user. The approaches proposed in the next section are faster as they employ codebooks with the exact same population size as the original video database.

5.2.7 Look-ahead distillation for encoders

We propose two forms of distillation in order to train **predictive encoders** that can anticipate the future content when presented with incomplete videos, and therefore generate richer, more discriminative bitcodes that are more likely to yield better search results.

5.2.7.1 Indirect distillation: look-ahead reconstruction (LA-RECO) loss

An encoder, which will from now on be referred to as the primary encoder (P in Figure 5.4), is trained following the same process as SSTH-RT$^+$ jointly with a decoder. Once this training is done, the main encoder is set aside and a secondary encoder (S in Figure 5.4, top right) is introduced. Only incomplete videos are fed into it - i.e. videos truncated to $\alpha T(V)$ with randomized *alpha* for each training step. The resulting bitcode is passed to the trained decoder with frozen parameters. The output of the decoder is compared to the full sequence in the loss, not just the truncated part fed to the secondary. This forces the secondary encoder to guess a representation accounting for future video frames.

$$\mathcal{L}_{decoder}(V) = \sum_{j=0}^{T(V)} \|f_j - \overleftarrow{f}_j\|_2 + \|f_j - \overrightarrow{f}_j\|_2 \tag{5.2}$$

5.2.7.2 Direct distillation: look-ahead bitcode (LA-CODE) loss

A secondary encoder is introduced as in the previous section, however this time the primary is put to use and the decoder is set aside (Figure 5.4, bottom right). During training, the primary encoder receives the entire video while the secondary encoder is only fed the αT first frames, again with randomized *alpha*. From the secondary, we extract the real-valued input β to the *sgn* function that leads to the last bitcode - we refer to this as the *prebitcode* - and compare that to the bitcode b given by the primary using an $\mathcal{L}_2$ loss function:

$$\mathcal{L}_{LA-CODE} = \|\beta(V, \alpha T) - b(V)\|_2. \tag{5.3}$$

This conditions two behaviors:

- the secondary, despite being only fed part of a video, mimics the full video's bitcode
- the primary adapts the bitcode of the full video, making it more robust to temporal crops

5.2.8 Training

Video-level features are extracted separately by I3D and loaded from binary files. Training is done in batches, which involves certain technical difficulties linked to sequence duration. Unlike in other video hashing approaches presented earlier, for which videos were downsampled to a fixed sequence length and evenly-sized batches could always be built, in our case the feature sequence length varies from one video to another due to the fixed sampling rate. Batching would require either zero padding, which is not an option here due to the use of batch normalization, or cropping to the shortest sequence; which, if performed carelessly, might discard significant portions of training videos due to length discrepancies within a batch. As a countermeasure, batching is based on the buckets defined in 5.2.2.1. We dynamically pick randomized batches from a single bucket at a time, guaranteeing a maximum length difference of about 22s inside a batch. Those batches are then trimmed to the shortest duration present in each of them; at most 22s of video are lost through this process.

Gradient descent is performed using the RMSProp algorithm. We introduce a new method named **recurrent gradient clipping**, as detailed in Appendix B; applied to hidden states and cell states, this entirely prevents gradient explosion, a long-standing issue of RNNs [PMB13]. Encoder/decoder pairs are trained for 60 epochs with a learning rate of 5.10^{-3} and a batch size of 40. The encoder is a 2-layer RNN with $2 \cdot N_{bits}$ in the first layer and N_{bits} units in the second. The decoder also has two layers, with N_{bits} in the first layer and $2 \cdot N_{bits}$ units in the second. Secondary encoders in `LA-RECO` or `LA-CODE` are trained for 15 epochs with a learning rate of 5.10^{-4}. Their weights are initialized from the values of the trained primary encoder's weights.

5.2.8.1 Evaluation

Once the primary and secondary encoders are trained, we use the primary to generate the codebook from the 9.7K videos picked from the test set. The remaining 1K form the query set. Evaluation is based on the video classes: for a given query video, a retrieved video is considered correct if its class matches the query's. The query encoder (i.e. the secondary encoder in `LA-RECO` or `LA-CODE`) generates a bitcode for each query video, truncated at αT frames. The code is compared to codebook entries, which are then ranked by Hamming distance to the query bitcode starting from the closest. Query videos are truncated using 10 different values for α ranging from 0.1 to 1.0.

As in [ZWHC16] we use the following definition for average precision at K - or AP@K - for a single video:

$$AP@K = \frac{1}{K}\sum_{j=0}^{K}\frac{N_{correct}(j)}{j} \tag{5.4}$$

where $N_{correct}(j)$ is the number of matching results among the top j.

The mean over all query videos is the Mean Average Precision at K - or mAP@K. We report this metric by level of observation for all methods and all bitcode sizes for K = 20. Lower values of K are more relevant in a real-time use case, where one can only handle a small number of search results; nonetheless we provide mAP@K curves for K up to 100 for our best model. We also provide qualitative retrieval results in two cases.

All experiments are repeated for four different bitcode sizes: 64, 96, 128, 192 bits. The compared approaches are: `SSTH-RT` (binary autoencoder trained to reconstruct full sequences from full sequences) `SSTH-RT`$^+$ (binary autoencoder trained to reconstruct full sequences from full sequences, as well as partial sequences from partial sequences) `SSTH-RT`$^{++}$ (`SSTH-RT`$^+$ with truncated duplicates hashed into the codebook) `LA-RECO` (`SSTH-RT`$^+$ as primary encoder, secondary trained to reconstruct full sequences from partial sequences) `LA-CODE` (`SSTH-RT`$^+$ as primary encoder, secondary trained to predict the primary's full-sequence-bitcodes from partial sequences)

5.2.9 Implementation & optimization details

Deep models are implemented in Tensorflow 1.13, and training is performed on servers fitted with either NVIDIA Tesla P100, K40m, K40c, K80 or GTX1080Ti GPUs. On a Tesla P100, training is done in under two hours for an encoder-decoder pair, and under 30 minutes for a secondary encoder. As stated in Chapter 4, distance computation consumes $O(N_{bits}\cdot N_{cb})$ operations; ranking costs $O(N_{cb}\cdot log(N_{cb}))$ using Python's built-in *sort* function. It is worth noting that for convenience reasons the search protocol is implemented in Numpy, with bitcodes represented as 8-bit Numpy booleans. An implementation with actual bits via a dedicated bit manipulation library would likely yield higher speeds than what we were able to achieve.

5.3 Results

5.3.1 Method comparison

mAP@20 is compared by level of observation for each method in Figures 5.6 5.7 with separate graphs for each size of bitcode. `SSTH-RT` serves as reference baseline. `SSTH-RT`$^+$ is a much more competitive model to compare to, `SSTH-RT`$^{++}$ even more so but at the cost of 10 times the space requirements and search time.

Both `LA-RECO` and `LA-CODE` outperform the `SSTH-RT` baseline by a substantial margin: 4 to 8%. `LA-RECO` outperforms `SSTH-RT`$^+$ by a small amount for 128 bits (less than 1%), but usually reaches similar or slightly worse mAP at other bitcode sizes. `LA-CODE` on the other hand outperforms both `SSTH-RT`$^+$ and `SSTH-RT`$^{++}$ significantly at most observation levels. For 128 bits, `LA-CODE` is superior to `SSTH-RT`$^+$ by 4 to 5% on

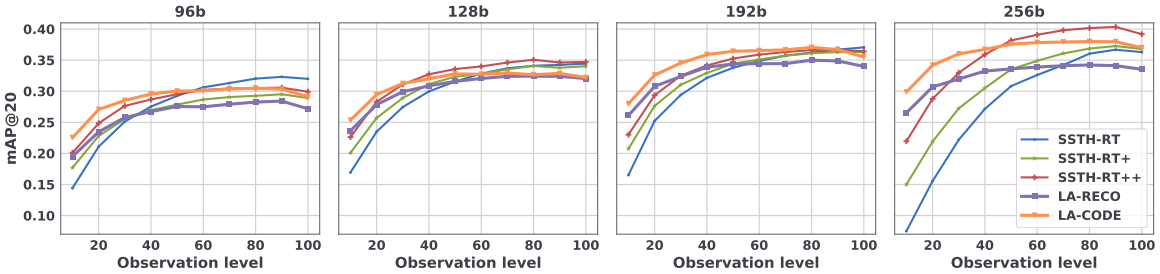


Figure 5.6: mAP@20 for all methods, all levels of observation, on FCVID.

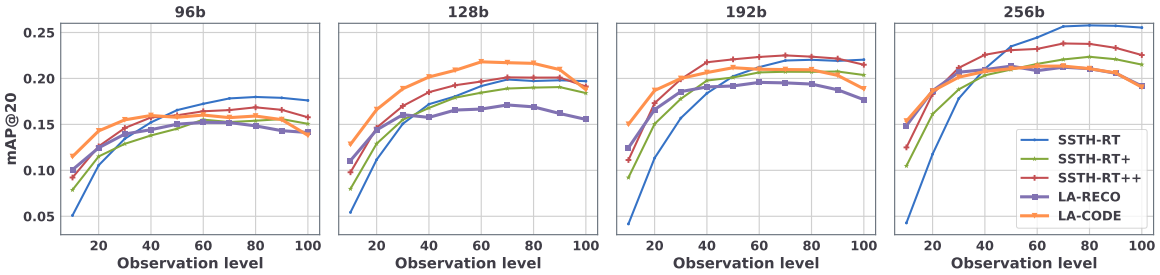


Figure 5.7: mAP@20 for all methods, all levels of observation, on ActivityNet.

	96 bits			128 bits			192 bits			256 bits		
FCVID	VE	E	O	VE	E	O	VE	E	O	VE	E	O
SSTH-RT	17.8	23.5	27.6	20.2	25.9	29.9	20.9	27.4	31.8	11.5	20.6	27.9
SSTH-RT^{+}	20.3	24.2	26.6	22.9	27.6	30.6	24.2	29.4	32.6	18.5	25.6	31.0
SSTH-RT^{++}	22.5	26.2	28.2	25.5	29.7	32.1	26.2	30.9	33.6	25.4	31.6	35.6
LA-RECO	21.5	24.6	26.2	25.7	28.8	30.5	28.4	31.5	33.0	28.6	31.2	32.6
LA-CODE	**24.8**	**27.5**	**28.8**	**27.4**	**30.2**	**31.4**	**30.3**	**33.5**	**35.0**	**32.0**	**34.9**	**36.3**
ActivityNet	VE	E	O	VE	E	O	VE	E	O	VE	E	O
SSTH-RT	7.8	12.2	14.9	8.3	13.4	16.5	7.8	14.0	17.9	8.0	15.7	20.6
SSTH-RT^{+}	9.7	12.1	13.7	10.4	14.2	16.5	12.1	16.4	18.5	13.3	17.3	19.6
SSTH-RT^{++}	10.9	13.6	**15.0**	12.2	15.8	17.8	14.2	18.4	**20.3**	15.4	**19.5**	**21.4**
LA-RECO	11.3	13.2	14.0	12.7	14.8	15.6	14.5	17.2	18.1	16.8	19.3	19.9
LA-CODE	**12.9**	**14.6**	**15.0**	**14.7**	**17.9**	**19.4**	**16.8**	**19.1**	19.8	**17.0**	19.2	19.9

Table 5.1: Results breakdown by α range. For each bitcode size: left column shows very early results (VE, average mAP@20 for α from 0.1 to 0.2); Middle column shows early results (E, average for α from 0.1 to 0.5); Right column shows overall results (O, average over all α).

most observation levels. Compared to SSTH-RT^{++} the improvement is smaller: around 2% on most observation levels. Outdoing this approach, even by a small margin, is however significant since this came at no extra storage and search time cost compared to the baseline, while SSTH-RT^{++} requires a codebook 10 times larger, takes 10 times as long to search (plus extra time to remove duplicate results).

Compared to LA-CODE, LA-RECO enforces a softer constraint: the bitcode-to-sequence mapping performed by the decoder may not be one-to-one. A learnt representation may therefore reconstruct well without approaching the bitcodes of relevant complete videos.

In terms of bitcode size, the general tendency observed in Figure 5.6 5.7 is that performance increases with size, as more bits provide more capacity for encoding information from the video. Looking at one method, peak mAP@20 for LA-CODE starts at 48% for 64 bits, then moves up to 51% for 96, 54% for 128, 55% for 192.

To highlight the compared methods' behavior in very early and early parts in a video, we consider the average of mAP@20 over three ranges of observation levels in Table 5.1.

Even though the focus of this work is early retrieval (50% of the video or less), in real-time the current level of observation would be unknown. Sustaining acceptable performance at higher levels of observation would therefore still be valuable.

The performance increase in the very low α range is generally accentuated compared to the whole range of α. This makes sense since the losses employed emphasize anticipation. For 128 bits, LA-RECO surpasses SSTH-RT^{+} by 1% on average. However for α specifically in $[0.1, 0.2]$ the performance increase is 1.9%. This also holds true for LA-CODE, which beats SSTH-RT^{+} by 2.8% overall and 4% in the very early α range, and SSTH-RT++ as well by a 1.9% margin overall and 2.7% for very low α.

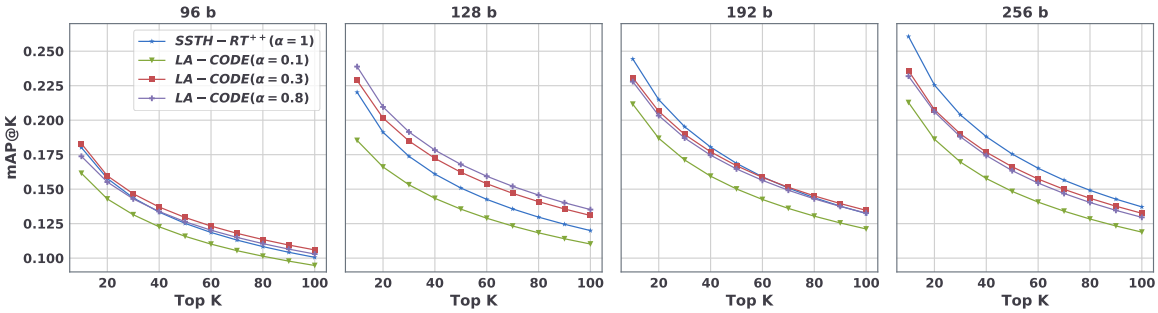


Figure 5.8: mAP@K for LA-CODE on FCVID

We display mAP@K for LA-CODE for all bitcode sizes for increasing observation levels, compared with SSTH-RT^{++}'s mAP@K on full queries (SSTH-RT$^{++}_{1.0}$) in Figures 5.8, 5.9. With 20% of video observed at 128 bits, LA-RECO loses to SSTH-RT$^{++}_{1.0}$ by 5%, which is expected due to the amount of missing information compared to the full video. However the gap is almost nullified with 40% of video observed. At 60% then 80% of video observed LA-RECO surpasses SSTH-RT$^{++}_{1.0}$ by up to 1 then 2%.

MAP@K plots for other bitcode sizes and methods are provided separately in Appendix C.

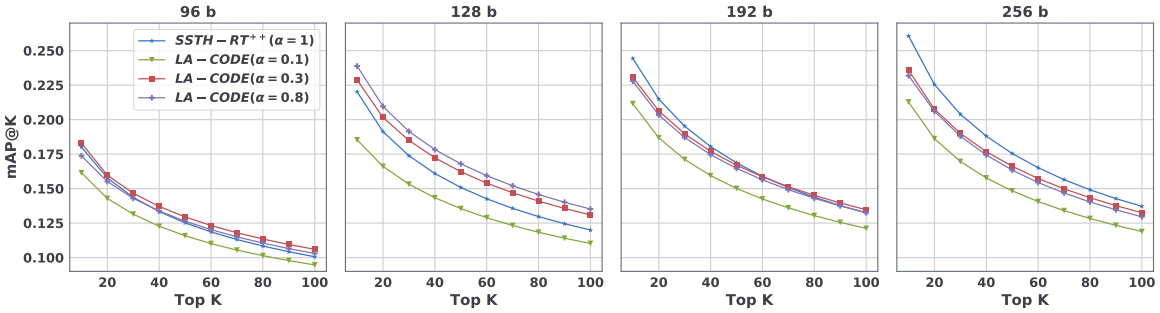


Figure 5.9: mAP@20 for `LA-CODE` on ActivityNet

5.3.2 Qualitative results

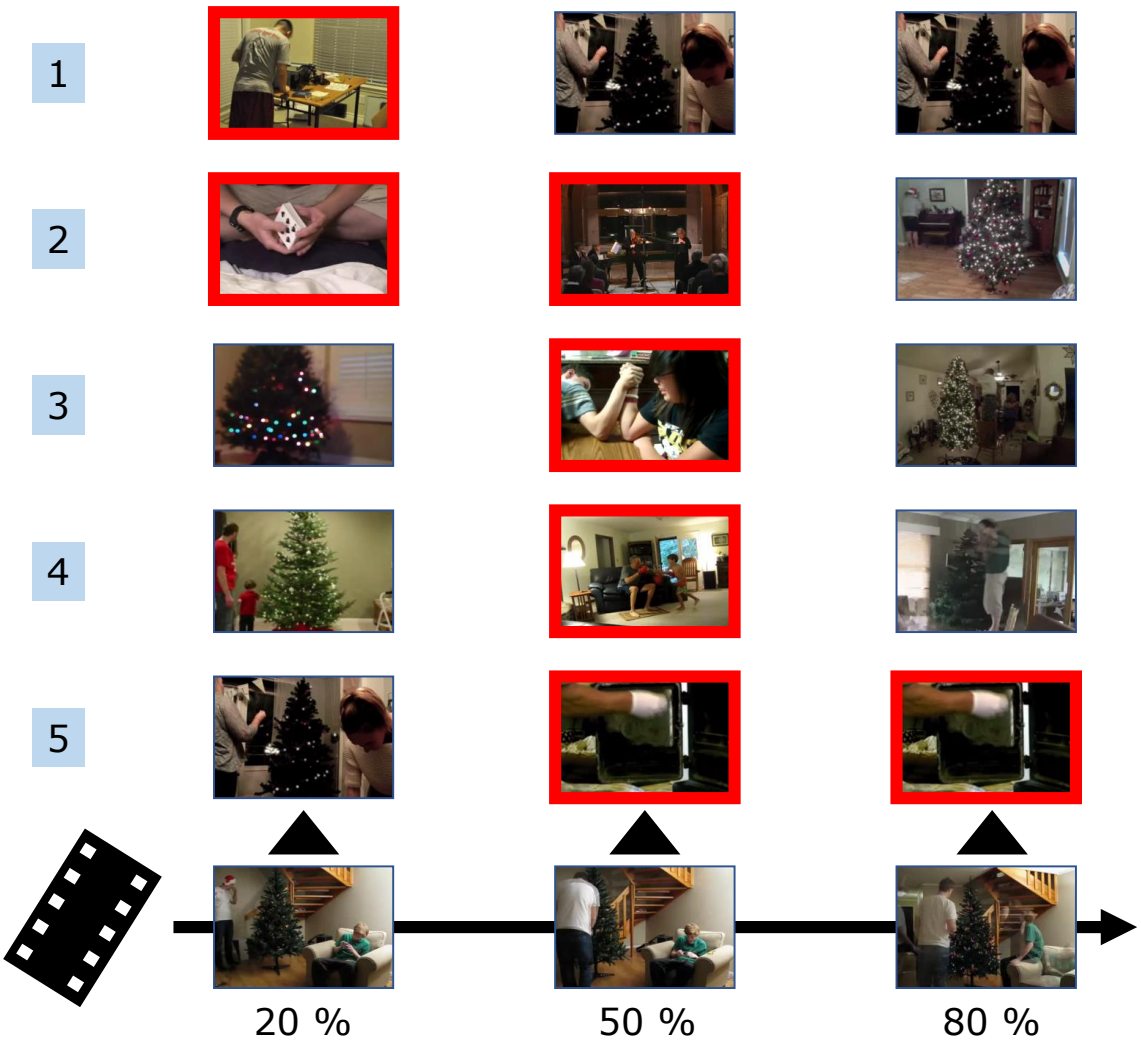


Figure 5.10: Example of retrieval over time from FCVID, showing the top 5 results for the beginning, middle and end. Red borders indicate class mismatches.

We selected two queries from the query set on which `LA-RECO` returned successful results (all results correct in top 10) at the end of the video, and followed the evolution

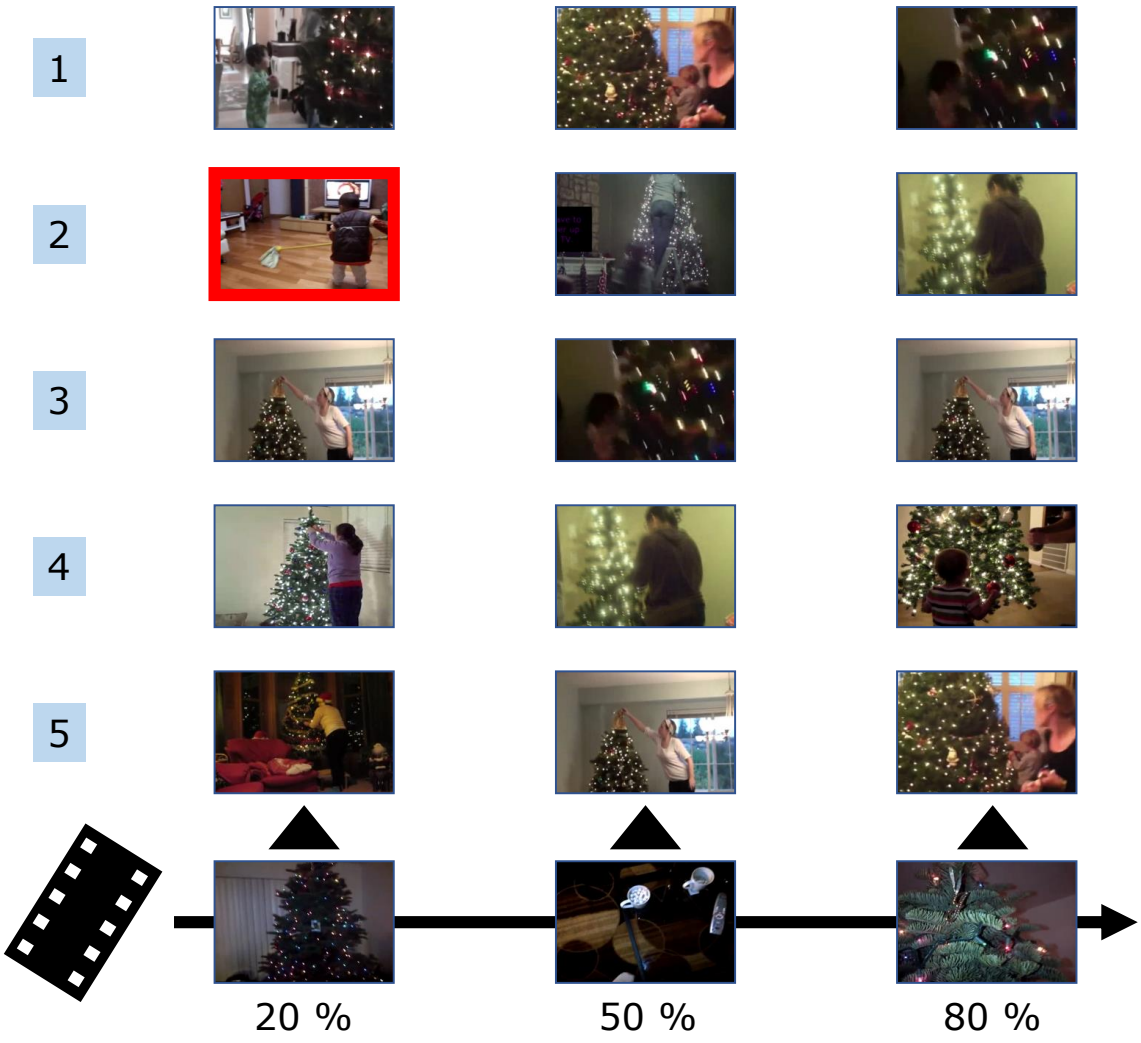


Figure 5.11: Successful example of retrieval over time from FCVID.

of the top 5 search results at different points in time. Both of them depict the action "decorating a christmas tree". Retrieval might be subpar early on in the video but the model should ideally adapt and return increasingly more relevant database entries as the video plays.

Figure 5.11 is an example of a successful case. A mistake is made early on at 20%, but top 5 search results improve to a perfect 5/5 after 50% of the video is seen. On the other hand, in Figure 5.10 the model starts with 3/5 correct results, then drops to 1/5 in the middle. Retrieval eventually improves towards the end with 4/5 correct results at 81%. In-depth qualitative results are available in the video at `https://www.youtube.com/watch?v=Eq-lIUipd4E`.

5.4 Conclusion

In this chapter, we presented an unsupervised video hashing approach designed for retrieval from incomplete video queries. We proposed to optimize the bitcode rep-

resentation of incomplete queries by distilling knowledge from an encoder trained on complete videos into a secondary encoder used only for the queries. For this purpose, we introduced two losses that are used to train the secondary encoder to match the encoding performance of the first encoder while using only truncated data as input. The secondary encoder thereby learns to produce informative bitcodes that can anticipate the future video content. The approach yields a large performance improvement over the baseline. It also outperforms a naive approach that inflates the codebook by adding codes generated from truncated copies of the videos in the database, while being at the same time much faster and more scalable for live hashing and retrieval on video streams.

CHAPTER 6

Real-time surgical video retrieval with uncertainty

"Man can only be certain about the present moment.
But is that quite true either?"

- Milan Kundera, *Ignorance*

Contents

In our binary representations of videos, every bit contributes equally when performing retrieval. Not all bits, however are equally reliable. When moving from generic human activity to endoscopic video data, we address this concern by proposing a new, lightweight method relying on compressed uncertain bit patterns corresponding to each database entry. As an extension of the previous work published in [YP20], the work presented here is the subject of a separate manuscript titled *Real-time Laparoscopic Video Retrieval with Compressed Uncertainty*, in preparation for Medical Image Analysis.

6.1 Objectives

The endoscopic video data in our dataset is vastly unused and unexplored, due to a lack of annotations for nearly 90 % of it. Available exploitation possibilities from the preexisting literature are scarce:

- Supervised learning using surgery type as the label [KYM+20]. This is the lowest possible resolution for describing surgical activity.
- Self-supervised or semi-supervised learning, which for the most part [FJM+18, YMMP18] uses the unannotated data without providing information about it. Even our previous method [YMMP19] has its limitations, since it is specific to the task of phase recognition in cholecystectomy.

An adequate content-based video retrieval method would be a considerable breakthrough for the exploitation of this dataset, essentially turning a passive video archive into a surgical video index that can be browsed visually, without requiring any tags. This opens up opportunities for various applications: post-operatively for indexing or reporting, or during surgical training as an educational tool. Intra-operative use would be valuable as well, but requires the ability to function in real time. Real-time retrieval, as shown in the previous chapter, brings its own set of challenges. With Endocorpus at our disposal, the first very-large-scale surgical video dataset, we study the problem of surgical video hashing in real time. What we verify here is retrieval's versatility, i.e. its ability to capture a wide variety of semantics in surgery. Therefore, the same model trained on one large collection of unlabelled videos is evaluated under three protocols: one based on cholecystectomy phases, one based on bypass phases, and one based on surgical critical events.

6.2 Methods

6.2.1 Data preparation

Out of the 1558 videos, we pick 81000 clips of 32s each for unsupervised training, and 16000 for validation. The training set incorporates all 12 available types of surgery, including the 6 not appearing in any of the test protocols.

Three types of labels are available for retrieval test:

- 7 cholecystectomy phase labels on *Cholec80*

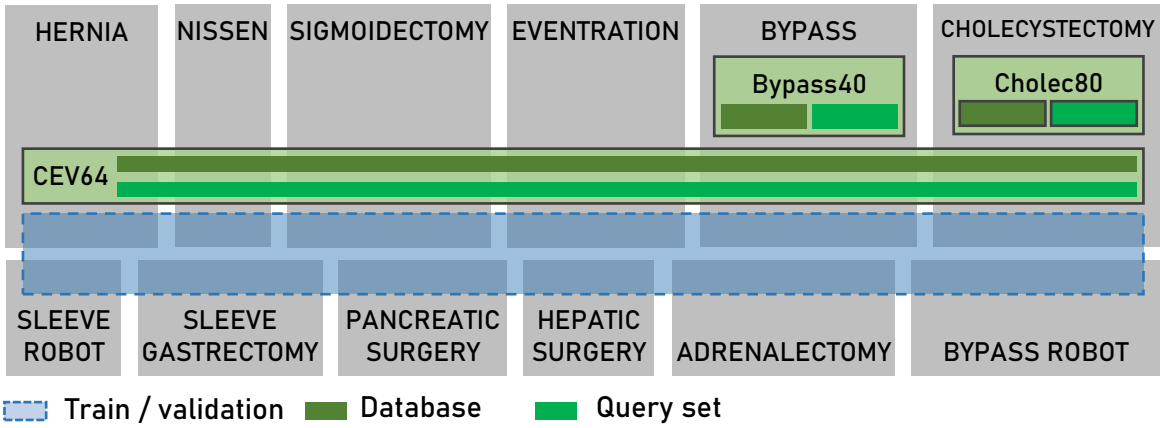


Figure 6.1: Endocorpus splitting for retrieval experiments (the size of graphical elements in the diagram does not reflect data quantities).

	Bypass40	Cholec80	CEV64
Query	284	307	580
Codebook	1409	928	1659
Training		**81000**	
Validation		**16000**	

Table 6.1: Data splitting for all three test protocols. Note that the training set and validation set are common to all three.

- 11 gastric bypass phase labels on *Bypass40* (Chapter 1, Section 1.3.3)
- 11 critical event labels on *CEV64* 1.3.4 (Chapter 1, Section 1.3.4)

Data splitting is done according to Table 6.1. Due to high imbalances in CEV64, both the query set and codebook set for this dataset were built by evenly collecting clips from the 11 event classes.

6.2.2 Uncertainty

When hashing live video sources dynamically using the approaches proposed in Chapter 5, the expected behavior is that the hash function updates the bit representation of the video content it has seen so far at regular time intervals (approximately 2 seconds in our case). Over the course of a given video, any given bit in the representation may flip several times. While this behavior is what makes our approach dynamic and fit for real-time use, excessive fluctuations for a particular bit make its value uncertain. So far, this uncertainty has been left unaccounted for in the codebook: the final bitcode, obtained after the primary encoder reads a video from the database from beginning to end, is taken at face value and stored as is, regardless of previous bit fluctuations.

In the case of `LA-CODE`, introduced in Chapter 5, another source of uncertainty is at play: the secondary encoder is trained to copy the primary encoder's representations of full videos, and is then used for querying. For a given clip, any bit where the primary and the secondary often enter in conflict should therefore be considered untrustworthy.

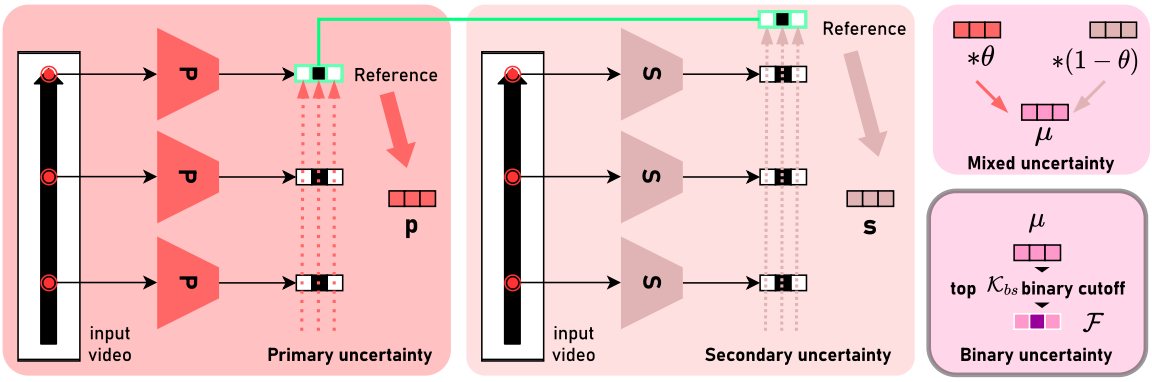


Figure 6.2: Primary and secondary uncertainty on the bitcode stored in the codebook.

From now on, we will refer to those as **primary uncertainty** and **secondary uncertainty** respectively.

Formally, let us consider a sequence of features representing a video clip $V = V_1, ...V_T$. We write the $t-$th subclip as $U_t = (V_1, ...V_t)$. Assuming we use d bits, let π and σ be `LA-CODE´`s primary and secondary encoders. Then throughout the course of the video V, π outputs the bitcode sequence:

$$\pi(U_1), \pi(U_2), ...\pi(U_t), ...\pi(U_T). \tag{6.1}$$

$\pi(U_T)$ in particular is the one stored in the codebook. For any timestep t, the $i-$th bit in the primary´s bitcode is written as $\pi_i(U_t)$. σ for the secondary follows the same notation.

The **primary uncertainty** for video V at bit i is defined as follows:

$$p_i(V) = \frac{1}{N-1} \sum_{t=1}^{N-1} \pi_i(V_T) \otimes \pi_i(V_t). \tag{6.2}$$

$\otimes$ is the bitwise `XOR` operation. Concretely speaking, this is the fraction of the time the primary spends in disagreement with the bitcode stored in the codebook. The **secondary uncertainty**, on the other hand, is defined as:

$$s_i(V) = \frac{1}{T} \sum_{t=1}^{N} \pi_i(V_T) \otimes \sigma_i(V_t). \tag{6.3}$$

Or, more simply, the fraction of the time the secondary spends disagreeing with the bitcode stored in the codebook. p and s can then be blended into a single uncertainty score, using a balance factor θ:

$$\mu(V, \theta) = \theta \cdot p(V) + (1 - \theta) \cdot s(V). \tag{6.4}$$

Storing μ itself along with $\pi(U_T)$, however, would be an extremely disproportionate way of communicating uncertainty (d floating-point values for d bits; with 32-bit floats that would be a factor 32). A much more space-efficient way to proceed is, for a given *bit skepticism level* K_{bs} (i.e. presumed number of untrustworthy bits in a bitcode), to flag the position of the K_{bs} most uncertain bits:

$$\mathcal{F}(V, \theta, K_{bs}) = \Phi_{K_{bs}}(\mu(V, \theta)). \tag{6.5}$$

The i-th coordinate of $\Phi_{K_{bs}}(X)$ is 1 if X_i is in X's top K_{bs} values, 0 otherwise. This is simply a binary mask suppressing non-top K entries.

With this information at our disposal, the querying mechanism can be readjusted to account for uncertainty: consider a query video Q at time t, and a database entry R to compare it to. In `LA-CODE`, the hamming distance would be computed as the number of conflicting bits between the query and the database entry's representations:

$$d(Q_t, R) = \sum_{i=1}^{d} \pi_i(R) \otimes \sigma_i(Q_t), \tag{6.6}$$

Using the binary uncertainty $\mathcal{F}$ and a discounting factor γ, we can modulate the contribution of each bit in the sum:

$$\Delta(Q_t, R, \gamma) = \sum_{i=1}^{d} [\pi_i(R) \otimes \sigma_i(Q_t)] \cdot (1 - \gamma \cdot \mathcal{F}_i(V, \theta, K_{bs})). \tag{6.7}$$

In summary, our new method, which we will refer to as `ULA-CODE`, performs the following steps:

- compute primary and secondary uncertainty values π, σ
- blend the two using a balance factor θ
- flag the position of the top K_{bs} uncertain bits
- when querying, discount uncertain bits in the hamming distance computation by a factor γ

The 3 free hyperparameters of the method are the discounting factor γ, the balance factor θ, and the bit skepticism level K_{bs}.

6.2.3 Computational footprint of uncertainty awareness

Accounting for bit uncertainty during retrieval comes at a certain cost, both in terms of time and space. With execution speed and compactness both being key selling points of hashing, it is crucial that the impact of our additions on the overall computational footprint is kept at a minimum.

Formally, we would like the following two constraints to be respected:

1. **Redundancy limit**: the additional space consumed remains strictly under dN bits(i.e. the size of the original codebook)
2. **Speed conservation**: the number of additional bit operations required per query is small compared to dN (the number of `xor` operations for hamming distance computations required in the original algorithm)

	d / 8		d / 4		3d / 8		d / 2	
	k	$log_2\binom{d}{k}$	k	$log_2\binom{d}{k}$	k	$log_2\binom{d}{k}$	k	$log_2\binom{d}{k}$
96	12	50	24	75	36	89	48	93
128	16	67	32	101	48	119	64	125
192	24	101	48	152	72	180	96	188
256	32	136	64	204	96	241	128	252

Table 6.2: Selected examples for $log_2\binom{d}{k}$; d indexes rows, k as a ratio of d indexes columns. In each cell, left is k and right is $log_2\binom{d}{k}$.

`SSTH-RT`$^{++}$, in the previous Chapter 5, egregiously contradicted both; with n_{dupl} truncated duplicates, both the space consumption and the number of operations were multiplied by n_{dupl}. This shows how careless tampering with the codebook purely for the sake of retrieval performance can severely undercut computational performance, which should not happen with `ULA-CODE`.

We first examine the redundancy limit constraint. Communicating the position of K_{bs} uncertain bits in an array of d bits can trivially be done with another d-bit array acting as a binary mask, i.e. set at 1 at uncertain bit positions. Doing so for each of the N codebook therefore requires $d \cdot N$ additional bits - exactly the limit. However keeping the space consumption **strictly** underneath is possible by **compressing the uncertainty pattern**.

The number of possible binary masks of d bits is of course 2^d. Yet, among those, we only need to account for the ones with a predetermined number k of bits set to 1. This drops the number of possibilities to the number of k-combinations of d elements, also known as the **binomial coefficient** $\binom{d}{k} = \frac{d!}{k!(d-k)!}$. Encoding an uncertainty pattern for k bits therefore only requires $d_u = \lceil \log_2 \binom{d}{k} \rceil$ bits instead of d. Evaluating the difference is not straightforward - we provide examples in Table 6.2. However we are able to provide a lower bound for the number of bits we are able to save:

$$d - log_2\binom{d}{k} > \frac{1}{2}\, log_2(\frac{\pi \cdot d}{2}). \tag{6.8}$$

A proof for this result is given in the corresponding Appendix F.

Practically speaking, any of the 2^{d_u} binary masks of k uncertain bits can be indexed by a binary array of size d_u; for example, by using the lexicographic order position written in base 2 (Figure 6.3).

Storing this **compressed mask** instead of the mask itself preserves space - however restoring the mask using its index is not trivial: this is referred to as the **combination unranking** problem.

For very low values of k, one can maintain a look-up table during retrieval. Looking up a mask costs $\mathcal{O}(1)$; space complexity, however, is $\mathcal{C}_s = d \cdot \binom{k}{d}$ bits. In the worst case of $k = \frac{d}{2}$, we can use Stirling's approximation to gauge this quantity:

$$\mathcal{C}_s \sim 4^d \sqrt{\frac{d}{\pi}}. \tag{6.9}$$

This is roughly exponential; to provide one example, a look-up table for uncertainty patterns of 48 bits in arrays of 96 bits would approximately consume $3.5 \cdot 10^{58}$ bits

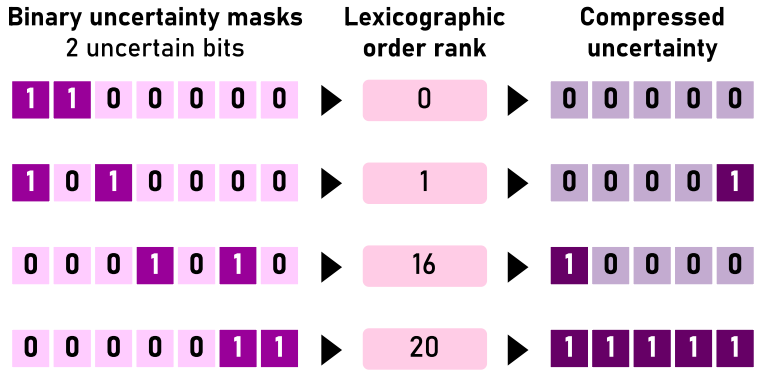


Figure 6.3: Uncertainty pattern compression. Each 7-bit pattern of two uncertain bits can be assigned a rank from 0 to $\binom{7}{2} = 5$; for example by using the lexicographic order. This rank is then written in base 2. In this case this saves two bits over the original pattern.

of memory ($4.3 \cdot 10^{45}$ TB). For this reason, algorithms for unranking combinations on the fly have been developed. Notably, Donnot et al. [GP21] proposed a fast algorithm named `unranking_factoradic` with $\mathcal{O}(d^2 \cdot log_2(d))$ complexity in bit operations. Even then, unranking for all N codebook entries would raise the overall time complexity from $\mathcal{O}(d \cdot N)$ to $\mathcal{O}(d^2 \cdot log_2(d) \cdot N)$, clearly violating the second constraint.

Fortunately, we can use **partial sorting** to drastically cut down the overall number of operations. In practice, N is extremely large - applications to extremely large databases is indeed a key motivation of hashing. On the other hand the number K of top items to retrieve should be quite small. Even though we show results for mAP@K for K going up to 100, in practice going through a list of 100 search results every two seconds does not make much sense for the end user or user application.

It is therefore safe to assume we can find K' such that $N \gg K' \gg K$. Using partial sorting, $N - K'$ irrelevant items can then be filtered out based on the raw Hamming distance, without accounting for uncertainty. This costs the same $d \cdot N$ `XOR` operations as previously. Within the remaining K' items, using uncertainty only requires an additional $\mathcal{O}(d^2 \cdot K')$, which can be considered small next to $d \cdot N$.

The overall encoding and retrieval pipeline for `ULA-CODE`, including the use of compressed uncertainty, is shown in Figure 6.4

6.2.4 Encoder training & codebook preparation & evaluation

Bitcodes of size 96, 128, 192, 256 are employed. Independently of the test protocol, the same model trained on the 81000 training set clips is used, in order to assess the versatility of the retrieval system. As previously done, we first train `SSTH-RT`$^+$ as the teacher for `LA-CODE`. This time, `LA-CODE` is trained for a maximum of 30 epochs, with early stopping based on bitwise accuracy measured on the validation set.

`ULA-CODE` reuses `LA-CODE`'s pair of encoders; the difference is in the way the codebook is built, since we incorporate the uncertainty defined above. `ULA-CODE`'s hyperparameter space, which is fairly small, is explored with all 80 combinations shown in Figure 6.5.

The metric employed is still top K Mean Average Precision (mAP@K), with mAP@10

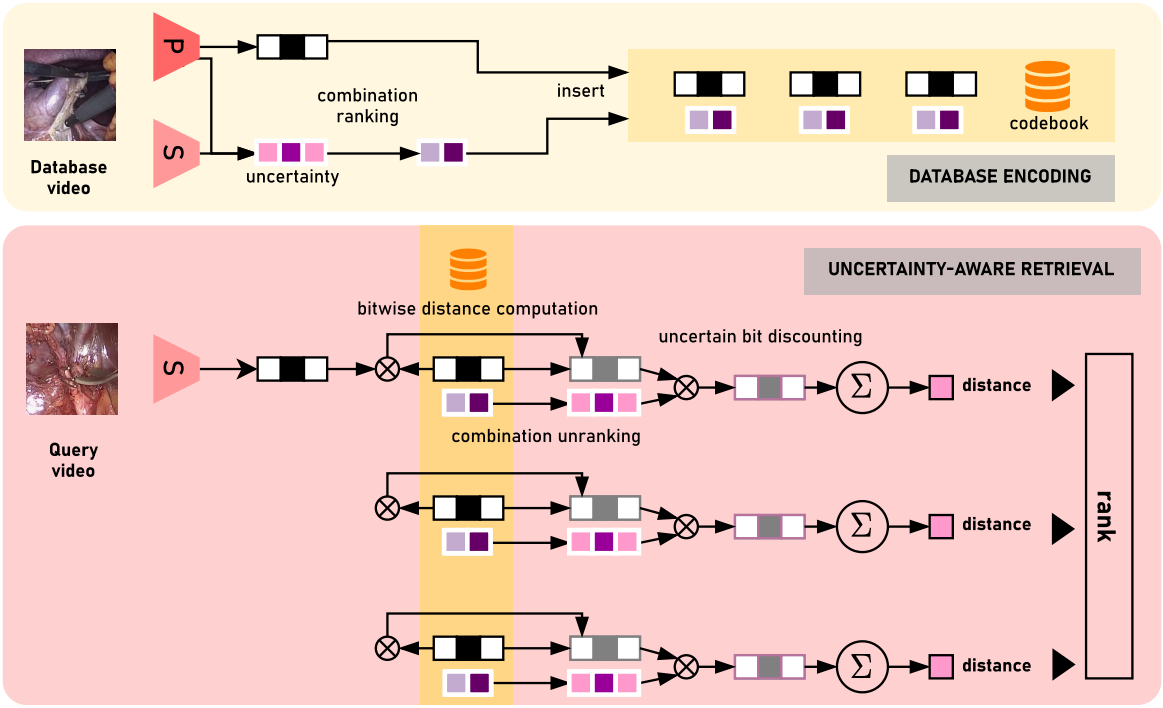


Figure 6.4: ULA-CODE pipeline overview. When building the codebook, the uncertainty pattern is compressed then stored alongside the bitcode. During retrieval, the uncertainty pattern is restored using combination unranking, and applied as a mask in the Hamming distance computation.

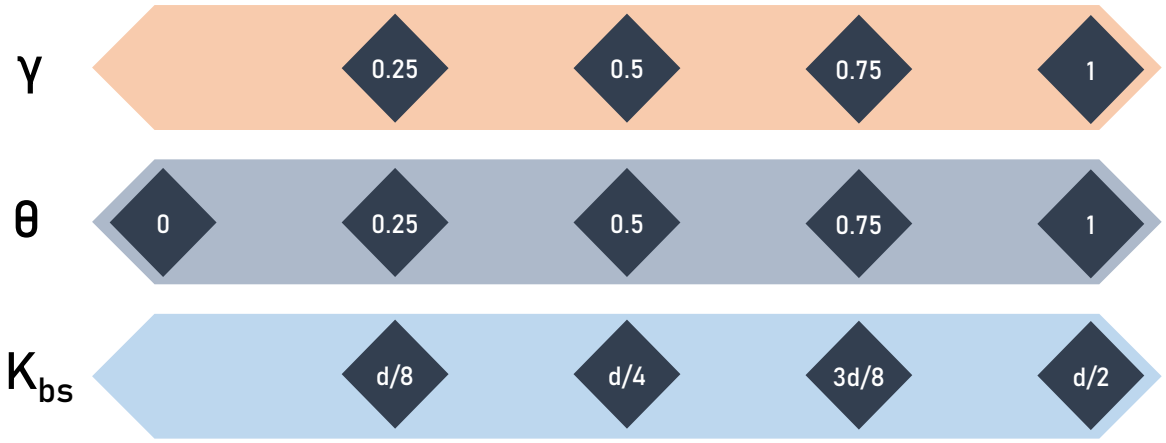


Figure 6.5: ULA-CODE hyperparameter ranges: bit skepticism K_{bs}, primary/secondary balance factor θ, uncertain bit discounting factor γ. Highlighted values are used in our experiments. Note that $\gamma = 0$ or $\mathcal{K}_{bs} = 0$ are equivalent to the regular LA-CODE, without using uncertainty

.

as the main reference. Even though we constrained the problem by dividing videos into units of uniform duration, the incremental nature of our methods enable efficient retrieval, in real time, at any point inside the clip; which was impractical and not attempted with video hashing methods from the rest of the literature [ZWHC16, SZL+18, WLG+17, WHG+19, LCL+19]. By doing so during testing, we essentially try to examine the retrieval system's responsiveness to the context, or very short-term anticipation; after only watching a portion of a clip, can the system quickly return relevant videos? We therefore report mAP@10 for one third and two thirds of a clip, in addition to the full clip (roughly 10, 20 and 30s respectively).

6.3 Results

6.3.1 Influence of hyperparameters

We start by examining the influence of each of the three hyperparameters in `ULA-CODE` separately. We report two quantities: first, the maximum mAP@10 obtained by fixing the value of one hyperparameter. We refer to this as `max ULA-CODE`. Second, the average mAP@10 obtained by fixing the value of a parameter and averaging over all combinations containing that value. This is referred to as `avg ULA-CODE`. `LA-CODE` sets our baseline. Results are displayed in Figures 6.6, 6.7, 6.8. In those figures a performance data point is obtained at every combination of:

- protocol (*Bypass40*, *Cholec80*, *CEV64*) - plot grid column
- bitcode size - plot grid row
- hyperparameter value - x-axis
- observation level α - line color
- mode or method (`LA-CODE`, `avg ULA-CODE` or `max ULA-CODE`) - line texture

Across all hyperparameters, `max ULA-CODE` is obviously superior to `avg ULA-CODE`, which is itself in the vast majority of cases above `LA-CODE`. For 256 bits on Bypass40 at full observation, `LA-CODE` achieves 38%, which is beaten by `avg ULA-CODE` with a 1.2% minimum margin, and by `max ULA-CODE` by 2 to 3%. For the same code size and observation level on Cholec80, `LA-CODE` sets the baseline at 39%, again surpassed by `avg ULA-CODE` (41.5%) and `max ULA-CODE` (41.3 to 42%). Similar observations can be made for critical events, with `LA-CODE` at 38.8%, `avg ULA-CODE` ranging from 39.5 to 41% and `max ULA-CODE` ranging from 40.2% to 42.2%.

This means even with a random choice of hyperparameters, the odds of achieving higher performance than `LA-CODE` are favorable.

For the γ parameter (Figure 6.6), we can see a slight trend favoring higher values: for instance, for 256 bits on critical events at 2/3 observation, `max ULA-CODE` goes from 41.2% at $\gamma = 0$ to 41.5% at $\gamma = 1$. In general, this suggests stricter suppression of uncertain bits improves retrieval performance.

For θ (Figure 6.7), it appears that lower values generally lead to higher mAP@10; such as for critical events with 256 bits at 2/3 observation, the dropoff from $\theta = 0$ to

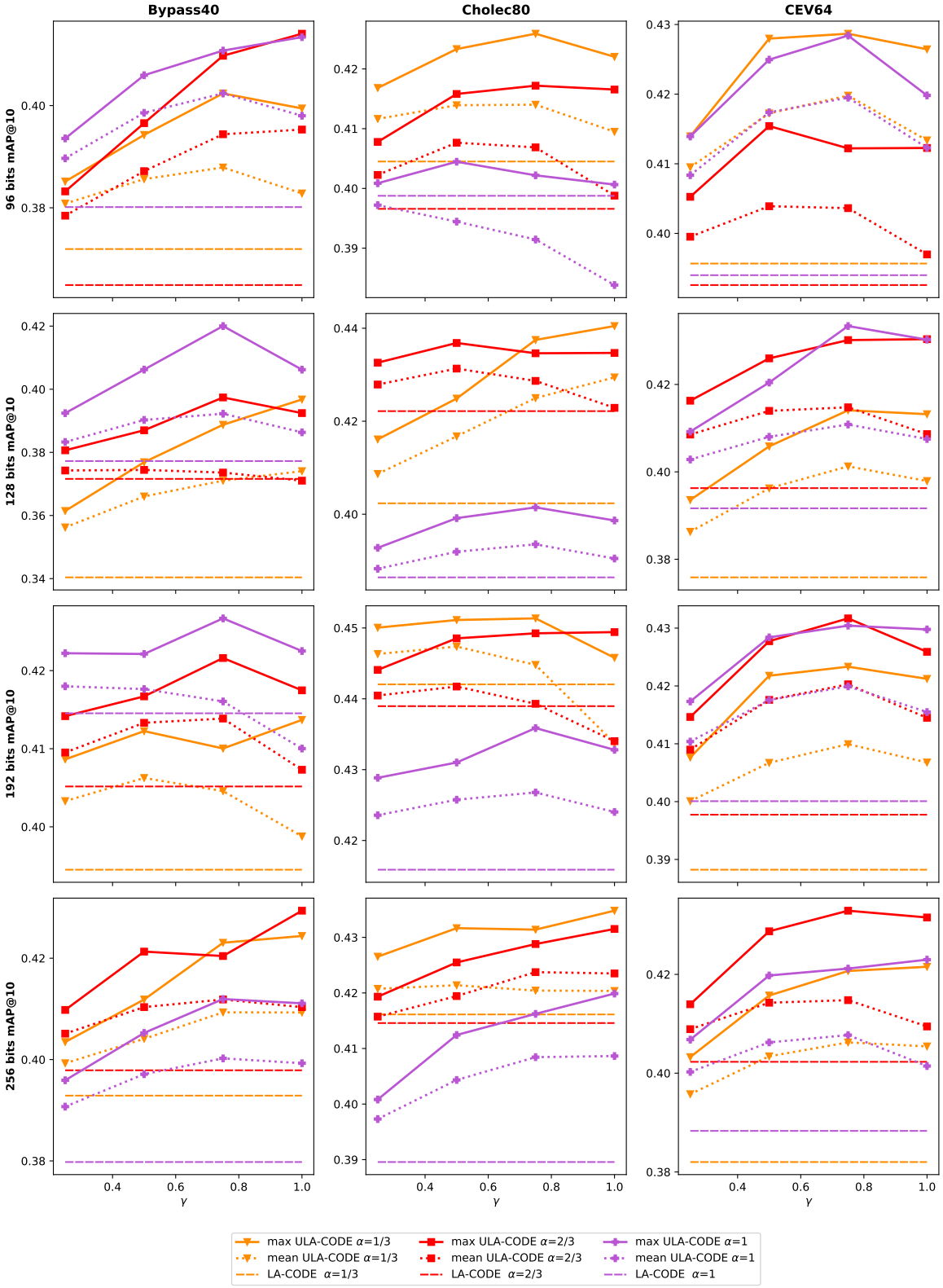


Figure 6.6: Influence of γ on mean average precision.

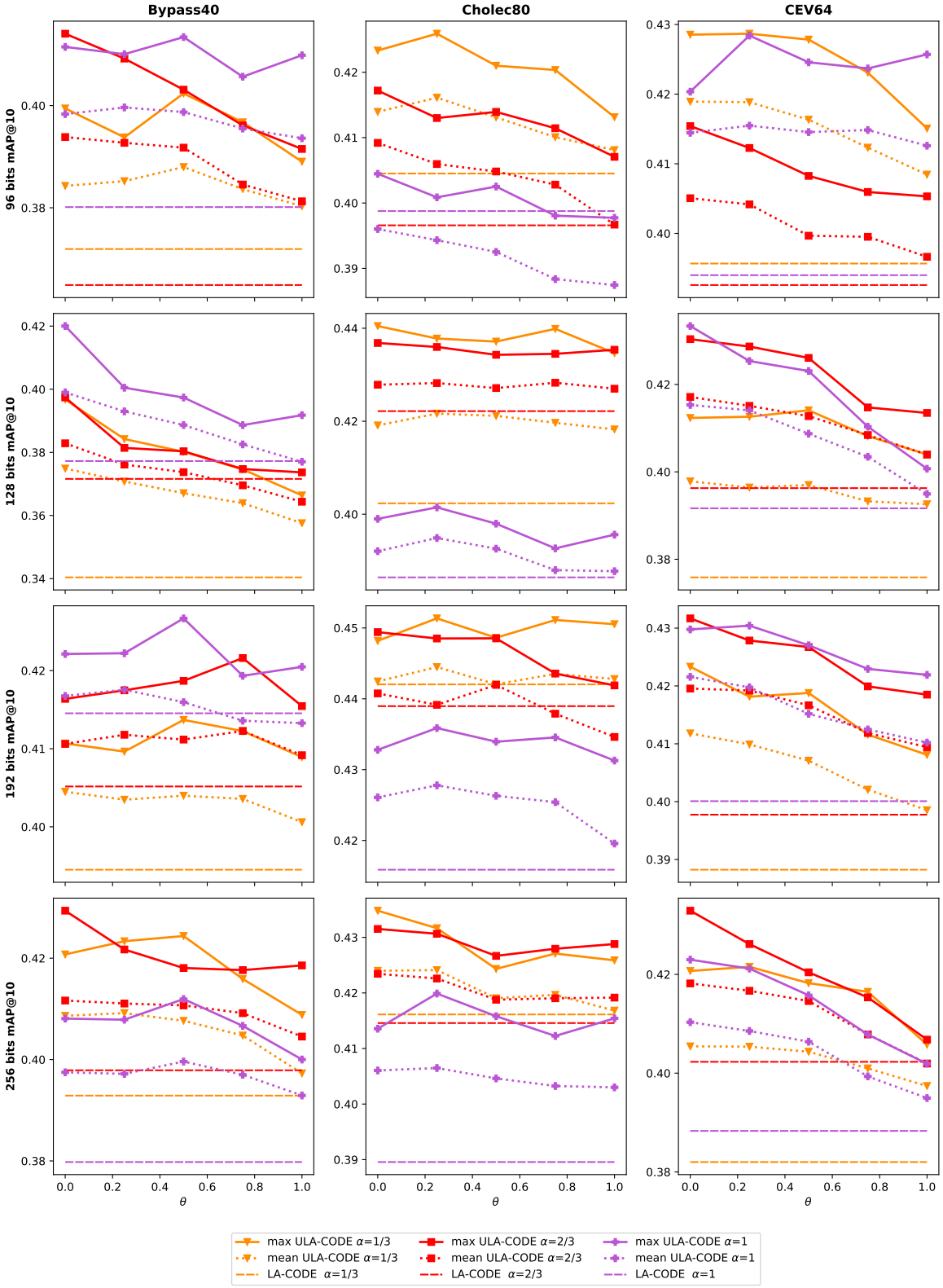


Figure 6.7: Influence of θ on mean average precision.

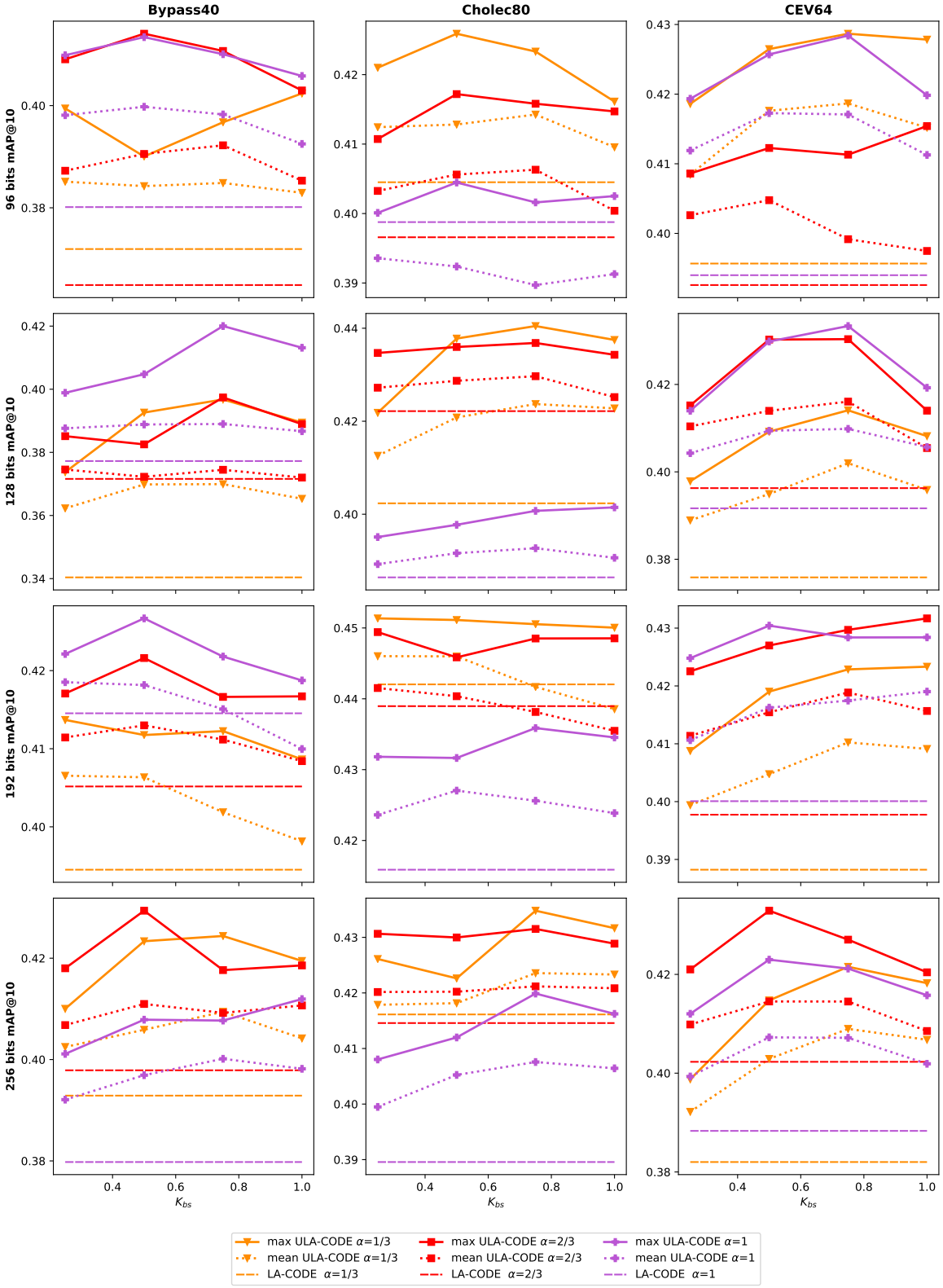


Figure 6.8: Influence of k on mean average precision.

$\theta = 1$ is over 2%. We can therefore assume that, generally speaking, the secondary uncertainty is more informative, and has higher odds of pointing towards faulty bits in the bitcode.

The trend for $\mathcal{K}_{bs}$ is more difficult to identify (Figure 6.8), although the results seem to slightly lean towards higher values. This would mean the number of untrustworthy bits is generally close to half the size of the bitcode. However this comes with a caveat: the compression rate of the uncertainty pattern decreases as the number of uncertain bits to report gets close to $d/2$.

For every code size and protocol, we are able to obtain an optimal value based on the highest mAP@10 achieved by `ULA-CODE`, averaged over all levels of observation. These optimal hyperparameters are reported in 6.3. The combinations found seem to confirm the trend observed: higher values of γ and $\mathcal{K}_{bs}$, lower values for θ.

	Bypass40			Cholec80			CEV64		
	γ	θ	$\mathcal{K}_{bs}$	γ	θ	$\mathcal{K}_{bs}$	γ	θ	$\mathcal{K}_{bs}$
96	1	0	2	0.5	0	3	0.75	0.25	3
128	0.75	0	3	0.75	0.25	4	0.75	0	3
192	0.75	0.5	2	0.5	0.25	3	0.75	0	4
256	1	0	2	1	0	3	1	0	2

Table 6.3: `ULA-CODE` optimal parameter combinations for each bitcode size and test protocol.

6.3.2 Comparison against baselines

The `max ULA-CODE` reported here uses the hyperparameter combinations found in Table 6.3. This time, `avg ULA-CODE` uses the average performance over the entire hyperparameter space. Additionally, we report the performance of `SSTH-RT`$^+$. mAP@10 is plotted for all three levels of observation: 10s, 20s and the complete clip of 30s.

On average, `ULA-CODE` outperforms our previous approach `LA-CODE` by 1 to 2 %. An optimized choice of parameters doubles this gain, surpassing `LA-CODE` by a 3 to 4 % margin. Exceptions to the order between the approaches - `SSTH-RT`$^+$ followed by `LA-CODE`, then average `ULA-CODE` and best `ULA-CODE` - in terms of performance are rare. For critical events, across all bitcode sizes and levels of observation, `avg ULA-CODE` exceeds the baseline `LA-CODE` by a mostly consistent 2% margin, Generally speaking, overall performance is higher for *Cholec80* (especially for 128 and 192 bits), which makes sense as it has a simpler workflow than *Bypass40* and is visually simpler than *CEV64*.

6.3.3 Qualitative results

Here we provide qualitative results in Figure 6.10 for surgical video retrieval, comparing search results returned by `LA-CODE` and `ULA-CODE` for a surgical clip used as the query. We display the bitcode corresponding to a video beneath its thumbnail, as well as the corresponding uncertainty pattern. The code size used is 128; for `ULA-CODE`, the parameters used were $\gamma = 0.75$, $\theta = 0$, $\mathcal{K}_{bs} = 64$.

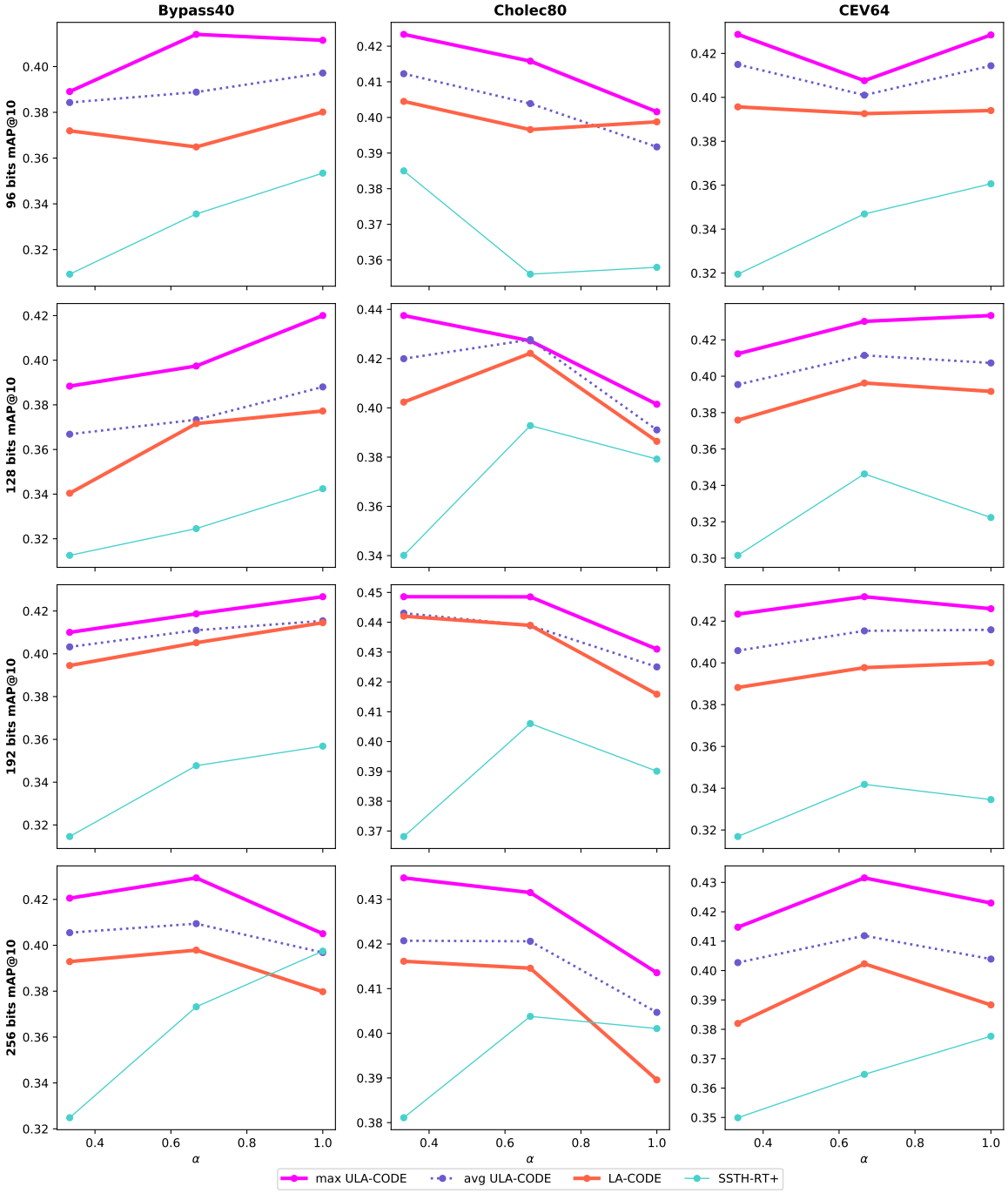


Figure 6.9: Mean average precision over time.

The query is from *CEV64*, a clip showing *incising*. By discounting uncertain bits, irrelevant results originally returned by `LA-CODE` (*abdominal access* and *idle*) are pushed down `ULA-CODE`'s ranking, while more accurate results appear higher.

6.4 Conclusion

In this chapter, we achieved the task of real-time video retrieval on a very-large-scale surgical dataset for the first time. Our proposed method `ULA-CODE` builds upon its predecessor `LA-CODE`, and addressed the problem of uncertain bits used in the codebook by measuring their degree of uncertainty, then reporting it in the codebook in a highly compressed manner: by doing so, `ULA-CODE` provides up to 4 % improvement in terms of retrieval mAP@10, measured using three semantic contexts in surgery: phases for cholecystectomy, phases for bypass and surgical critical events, introduced for the first time in our work. Usability on this wide range of semantics is encouraging for the generalizability of our method.

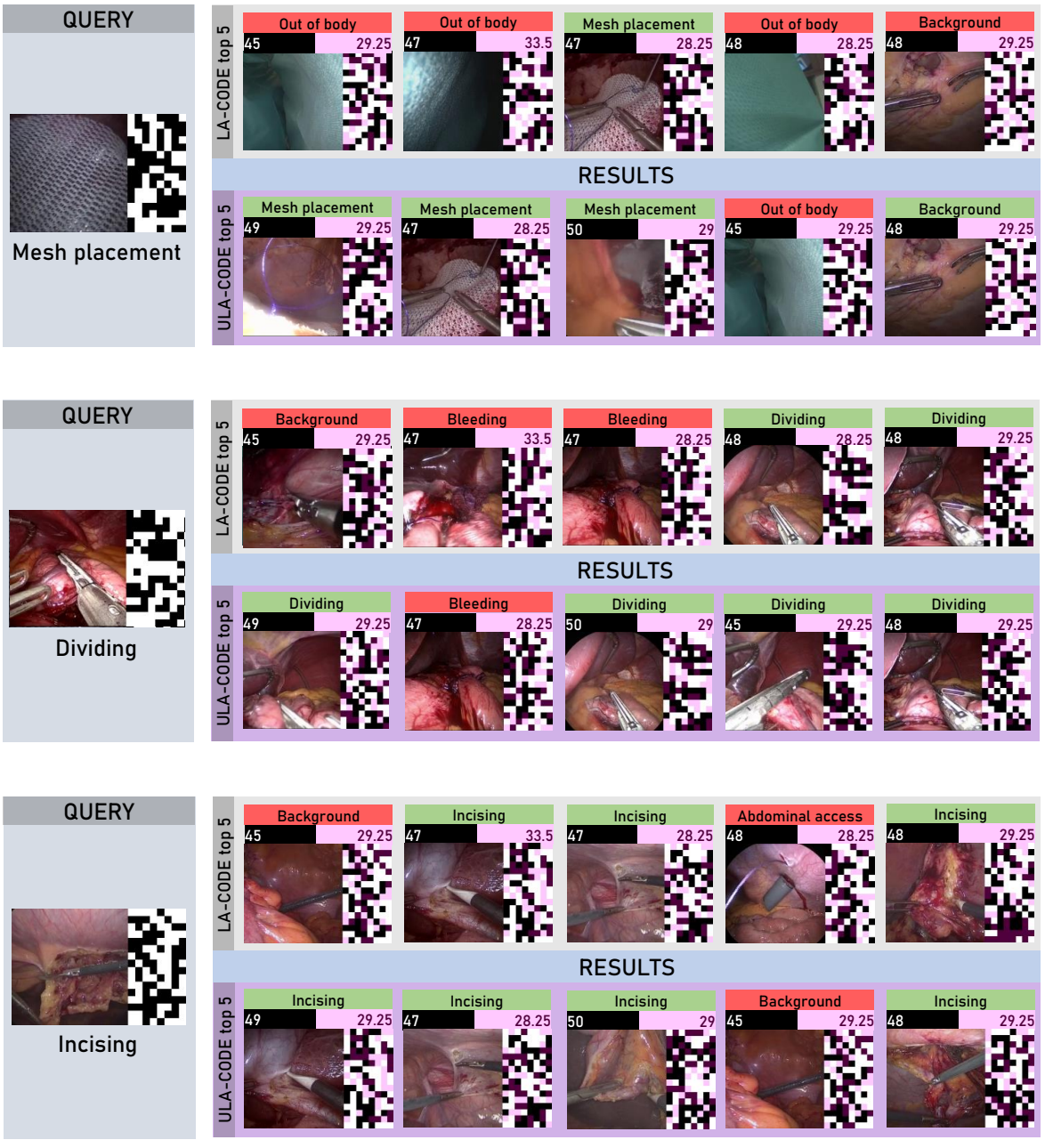


Figure 6.10: Qualitative results, with and without using uncertainty. Left: query clip. Center: `LA-CODE` search results, green highlight if correct and red otherwise. Right: `ULA-CODE`. Under thumbnails: black is the bitcode, red is the uncertainty pattern. Several correct videos overlooked by the first approach are ranked higher by the second.

CHAPTER 7

Conclusion

Contents

This final chapter recapitulates the contributions presented throughout this work, and how they respond to our original problem of providing real-time support in the operating room with vast amounts of unannotated data. Application possibilities, as well as current limitations and paths for further developments are discussed next.

7.1 Summary

The primary goal in the work presented here is the development of methods for surgical activity understanding with endoscopic videos, under **two key constraints**:

1. **Extensive use of unlabelled video data**
2. **Real-time capability**

These constraints fit our overarching vision of an OR control tower (Figure 1.16) with minimal reliance on human-annotated video data, providing well-timed support to surgeons during interventions. The methods we proposed successfully addressed those.

7.1.1 Addressing unlabelled data domination

As mentioned in Figure 1.15, the state of the Endocorpus dataset reflects the general state of endoscopic video data: generated in massive amounts by operating rooms, yet unannotated and unexplored for its vast majority. Since systematic human-generated annotations are not a scalable option, we turned to semi- and self-supervision in order to make efficient use of unlabelled videos.

The semi-supervised approach presented in Chapter 3 used automatic annotation to turn large quantities of unannotated videos into usable training material for other models: labels generated by a CNN-biLSTM-CRF, while not perfect with respect to ground truth, are sufficient to enhance a CNN-LSTM's performance.

In Chapters 5 and 6, the hash function generating compact binary representations of videos was trained in a self-supervised manner. Since no labels were used, training data was abundantly available - especially in the case of Endocorpus. Additionally, the resulting encoder was label-agnostic: distance between codes was based on visual similarity. Not only retrieval gives the unlabelled data a purpose, it also enables exploring it, functioning as a visual index.

7.1.2 Addressing real-time conditions

While surgical activity understanding from endoscopic videos certainly has potential outside of the operating room - for example in surgical education, skill evaluation or video indexing - our primary target for applications has been **in-OR context-aware support** from the start. Since this requires the ability to function in real time, the level of difficulty is raised by a considerable amount. In particular, a certain degree of **anticipation** is needed, which is challenging to achieve due to the variability and unpredictability of surgical workflows.

In Chapter 3, our method took advantage of a model that is not real-time compatible, the CNN-biLSTM-CRF, to annotate data with anticipation of the future, by

reading the video frame sequence in both directions. This artificially annotated data was then fed to the real-time compatible CNN-LSTM.

In Chapter 5, we transformed the previously established approach to video retrieval, by sampling video data in a way that makes retrieval usable in real time with live sources. We also incorporated anticipation in our `LA-CODE` model, in order to account for missing future frames in real-time conditions.

Chapter 6 introduced a method to cut down the contribution of noisy bits during retrieval. Computational burden is kept at a minimum, in order to preserve the efficiency of hashing for real-time use.

7.2 Discussion and future work

Discussion points opened by the work from this thesis can be grouped in three key areas. The first one is **methodology**, concerning technical details of our approaches. The second is **scale**, with questions on expanding the data used for our methods. Finally, we discuss the more practical aspects of **applications and deployment**.

7.2.1 Methods

Evaluating video retrieval in a satisfying manner is a delicate question - to determine whether a returned entry is relevant or not, all video retrieval protocols in the literature rely on class labels provided by the dataset employed. To a certain degree, we mitigated this issue in Chapter 6 by evaluating the same model on multiple sub-datasets, each with its own label type, showing the method can capture visual similarity in a way that is generic across multiple surgical contexts. Still, a protocol that is less reliant on labels would be a major step forward. Indeed, the power of content-based retrieval lies not in its ability to recognize queries based on predetermined classes, but rather in its expressiveness and ability to find information users did not know they were looking for, as it is often the case for popular services such as Bing [HWY+18] or Google Image Search.

Another pending question is **retrieval granularity**; even though we are able to retrieve relevant videos from a query, our methods - as well as all video retrieval methods from the literature - do not expose which parts inside of the retrieved videos are relevant. The leeway for achieving this is razor thin: incorporating uncertainty without severely cutting down performance was already a challenging problem. Adding intra-video retrieval on top of ordinary retrieval is likely even more difficult.

7.2.2 Scale

Within the literature on endoscopic video processing, the body of work presented here sits at the top in terms of scale, with 12 types of surgery and over 1.5K recordings of surgical procedures in Endocorpus. Still, this amount is negligible compared to what an actual full-scale database could be: as stated in Chapter 1, Subsection 4.1.2, an ideal case scenario where all procedures are systematically collected would likely lead to databases in the $10^6 - 10^7$ range. In addition to the issue of raw size, Endocorpus only features a single location and a relatively small number of surgeons. Other clinical institutions and other teams might largely deviate from our dataset in terms of

workflow and appearance, due to, among other factors, different surgical tool brands and different habits when performing surgery. A larger study spanning multiple clinical facilities would be necessary in the future, potentially with the help of **federated learning**.

In addition to data sourcing and quantity, using our methods with a wider variety of annotation types would be an appropriate next step. For retrieval, we introduced the first dataset of surgical critical events, which could be expanded with more event types in the future. The automatic annotation method from Chapter 3 has so far only been used on cholecystectomy; other more complex surgery types may be interesting targets as well, such as gastric bypass with *Bypass40*.

7.2.3 Applications & deployment

The efficiency of visual search in other domains leaves no doubt on the potential of video retrieval in surgery. However, a number of questions surrounding in-OR implementation would need to be addressed. Assuming we are able to achieve context awareness as presented in our work and implement it within a system, how would it interact with the surgeon and the clinical staff inside the OR? Video retrieval, in particular, is unprecedented in this kind of context. For publicly available tools such as Google Image Search, the user prompts the search by manually uploading the query, or taking a picture. In the tightly controlled environment of the OR, however, this interaction needs to be carefully designed in order to be as unobtrusive as possible. A vocal command, or a separate display automatically refreshing search results at regular time intervals are possible options. Another one is to delegate the review of retrieved videos to the OR control tower staff, as suggested in Figure 1.16.

Querying, however, is only half of the user experience; careful attention should be given to the presentation of search results. Showing the retrieved raw video data itself may not be the best option - reviewing it takes time, and paying attention to multiple videos simultaneously is nearly guaranteed to result in cognitive overload. An adequate preview system, as found on popular video platforms, would need to be designed; for instance with animated thumbnails giving a brief and compact overview of a video's content. Showing any relevant video metadata would also be a valuable feature; although in our experiments this metadata was limited to class labels and links to future clips (Chapter 7), content-based retrieval can collect far more descriptive information, as shown by current reverse search engines.

Appendices

APPENDIX A

List of publications

First author

1. **T. Yu**, D. Mutter, J. Marescaux, N. Padoy

   Learning from a Tiny Dataset of Manual Annotations: a Teacher/Student Approach for Surgical Phase Recognition

   IPCAI, 2019

2. **T. Yu**, N. Padoy

   Encode the Unseen: Predictive Video Hashing for Scalable Mid-Stream Retrieval

   ACCV, 2020

3. **T. Yu**, P. Mascagni, J. Verde, N. Padoy, D. Mutter

   Real-time Laparoscopic Video Retrieval with Compressed Uncertainty

   In preparation for submission to Medical Image Analysis, 2021

Co-author

1. P. Mascagni, C. Fiorillo, T. Urade, T. Emre, **T. Yu**, T. Wakabayashi, E. Felli, S. Perretta, L. Swanstrom, D. Mutter, J. Marescaux, P. Pessaux, G. Costamagna, N. Padoy, B. Dallemagne

   Formalizing video documentation of the Critical View of Safety in laparoscopic cholecystectomy: a step towards artificial intelligence assistance to improve surgical safety

   Surgical Endoscopy, 2019

2. C.I. Nwoye, C. Gonzalez, **T. Yu**, P. Mascagni, D. Mutter, J. Marescaux, N. Padoy

   Recognition of Instrument-Tissue Interactions in Endoscopic Videos via Action Triplets

   MICCAI, 2020

3. 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, steps in gastric bypass procedures

   IPCAI, 2021

4. C. I. Nwoye, **T. Yu**, C. Gonzalez, B. Seeliger, P. Mascagni, D. Mutter, Jacques Marescaux, Nicolas Padoy

   Rendezvous: Attention Mechanisms for the Recognition of Surgical Action Triplets in Endoscopic Videos

   Medical Image Analysis, 2021

5. M. Wagner, B. Müller-Stich, A. Kisilenko, D. Tran, P. Heger, L. Mündermann, D. Lubotsky, B. Müller, T. Davitashvili, M. Capek, A. Reinke, **T. Yu**, A. Vardazaryan, C. I. Nwoye, N. Padoy, X. Liu, E. Lee, C. Disch, H. Meine, T. Xia, F. Jia, S. Kondo, W. Reiter, Y. Jin, Y. Long, M. Jiang, Q. Dou, P. Heng, I. Twick, K. Kirtac, E. Hosgor, J. L. Bolmgren, M. Stenzel, B. Siemens, H. G. Kenngott, F. Nickel, M. Frankenberg, F. Mathis-Ullrich, L. Maier-Hein, S. Speidel, S. Bodenstedt

   Comparative Validation of Machine Learning Algorithms for Surgical Workflow and Skill Analysis with the HeiChole Benchmark

   Under review for Medical Image Analysis, 2021

APPENDIX B

Recurrent gradient clipping

B.1 Motivation: gradient explosion

Gradient explosion is a major obstacle in the training process for Recurrent Neural Networks, which were used throughout the work presented in this thesis. If any value involved suring training, such as a gradient or a model parameter, exceeds the maximum value tolerated by the system (e.g. $3.4e^{38}$ in Tensorflow), a not-a-number or `NaN` is raised and propagates to all model parameters within one cycle of forward and backward computation; thereby undermining the entire training process.

B.2 Solution

B.2.1 An incomplete answer: standard gradient clipping

To prevent model parameters from being updated to exceedingly large values over training iterations, Pascanu et al. [PMB13] proposed *gradient clipping*. After computing the gradient $\vec{\nabla}_W L$ of the loss L with respect to model parameters $W \in \mathbb{R}^d$, the following function is applied:

$$\phi_k : \begin{cases} u \mapsto u \text{ if } ||u|| \leq k \\ u \mapsto k\frac{u}{||u||} \text{ if } ||u|| > k \end{cases} \tag{B.1}$$

This method of scaling back gradients above a certain magnitude threshold is called *norm clipping*; using this, Pascanu et al. were able to successfully train RNNs to generate longer sequences than with unclipped gradients.

In practice however, extreme cases can cause this workaround to fail. Those are cases where explosion occurs in a single training iteration, i.e. if $\vec{\nabla}_W L$ yields `NaN` in the first place. In that case attempting to scale the result back still yields `NaN`, returning to the original issue.

B.2.2 Clipping between RNN iterations

Drawing inspiration from artificial gradient methods featured in [ZWHC16, HCS+16], we define the following computation node:

$$\begin{cases} \mathcal{Z}(v) = v \\ \vec{\nabla}_v \cdot U(Z(v)) = \phi(\vec{\nabla}_v \cdot U(v)) \text{ for any differentiable function U} \end{cases} \tag{B.2}$$

Those gradient clipping nodes are placed after the computation of the cell state and the hidden state (Figure B.1).

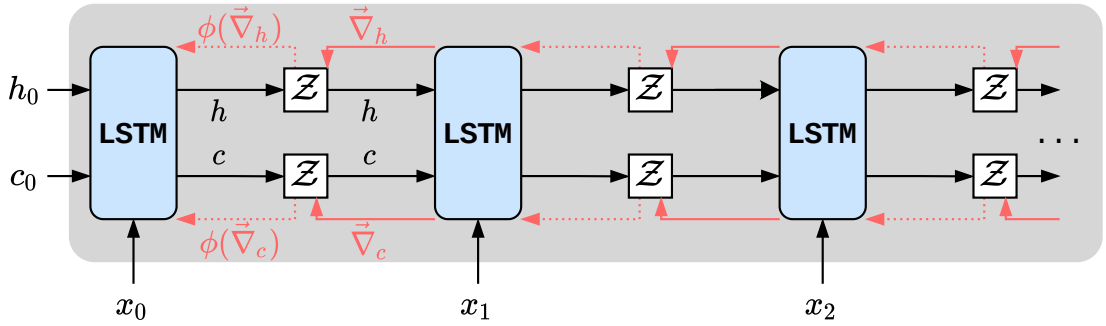


Figure B.1: Recurrent gradient clipping.

Experiments first performed in Chapter 5 were prone to `NaN` errors, which persisted after implementing standard gradient clipping. After implementing recurrent gradient clipping, no `NaN` errors were reported again.

APPENDIX C

Detailed mAP@K results for retrieval on generic activity datasets

This appendix expands on mAP@K retrieval results shown in Chapter 5. Each group of four sets of curves corresponds to a method, with each set of curves accounting for a particular code size and each curve showing mAP@K for a certain level of observation.

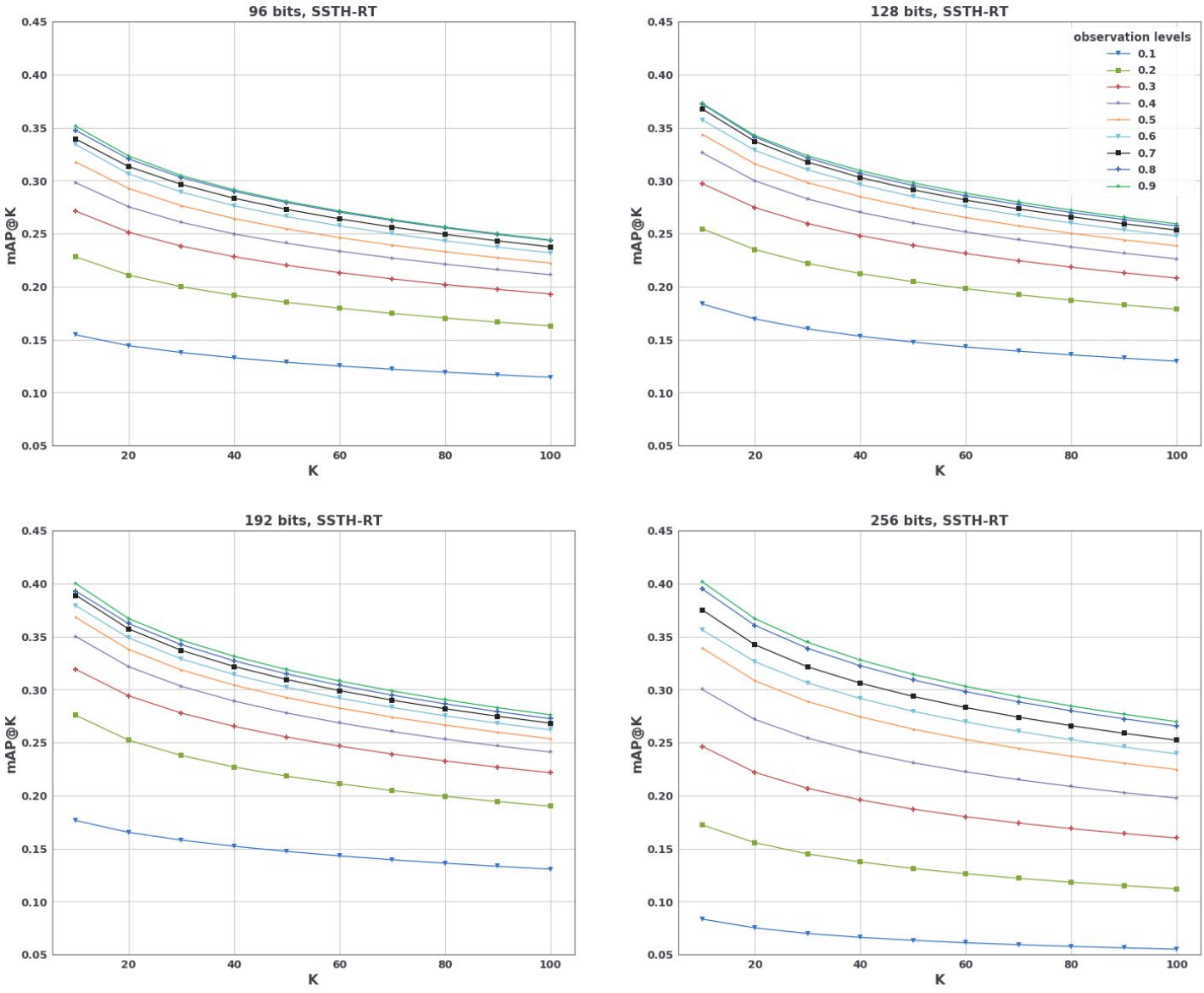


Figure C.1: FCVID: results for `SSTH-RT`

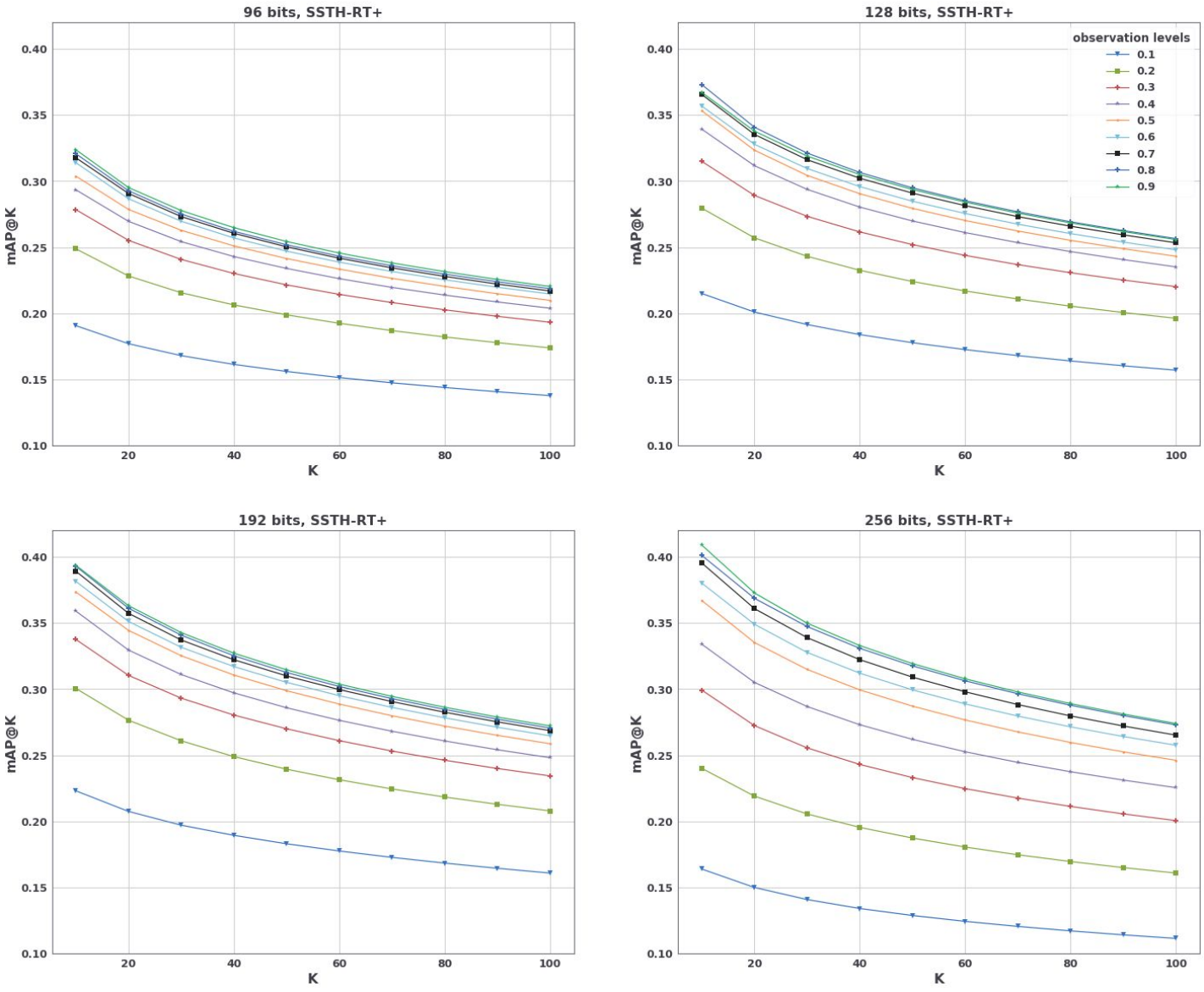


Figure C.2: FCVID: results for SSTH-RT$^+$

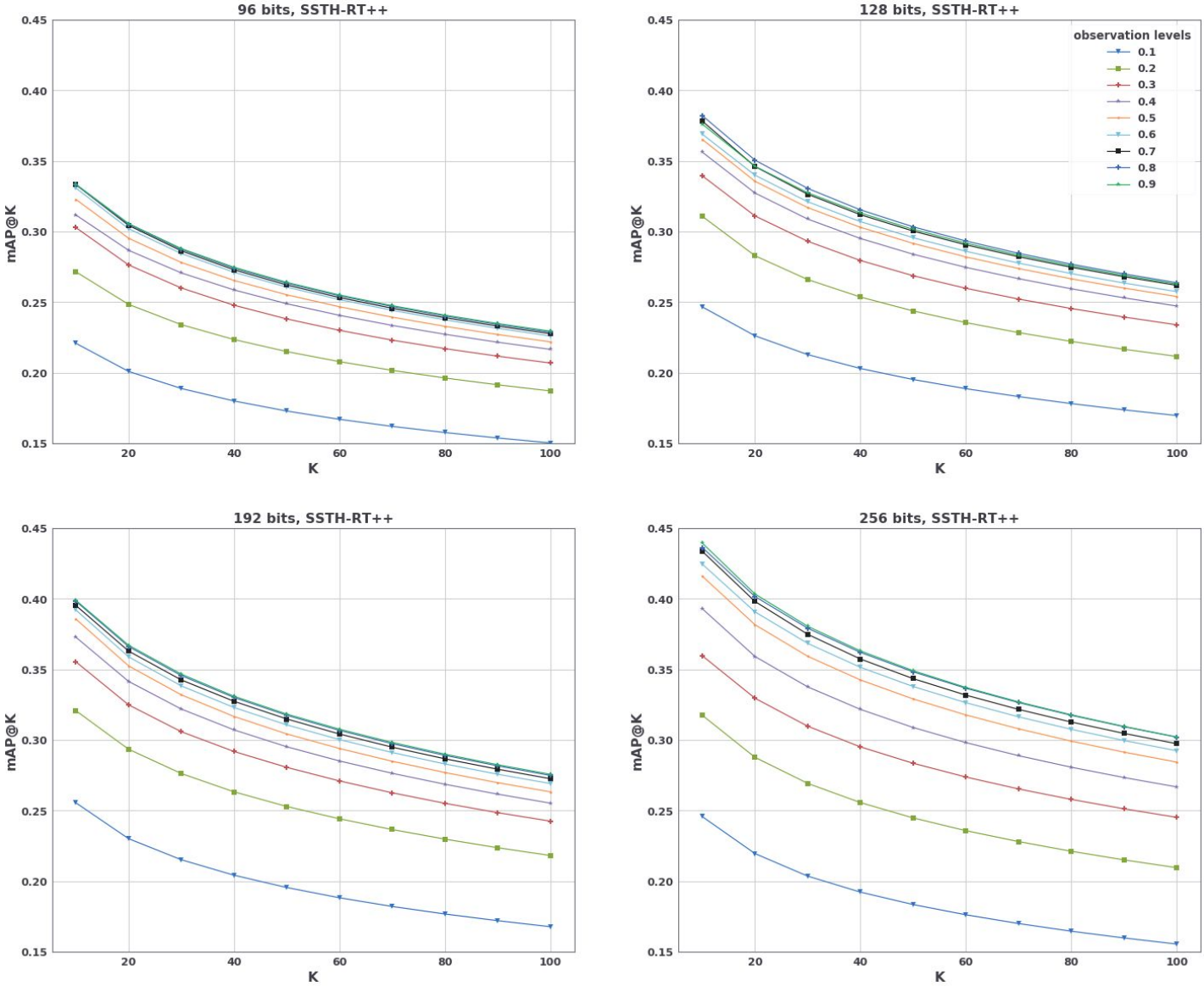


Figure C.3: FCVID: results for SSTH-RT^{++}

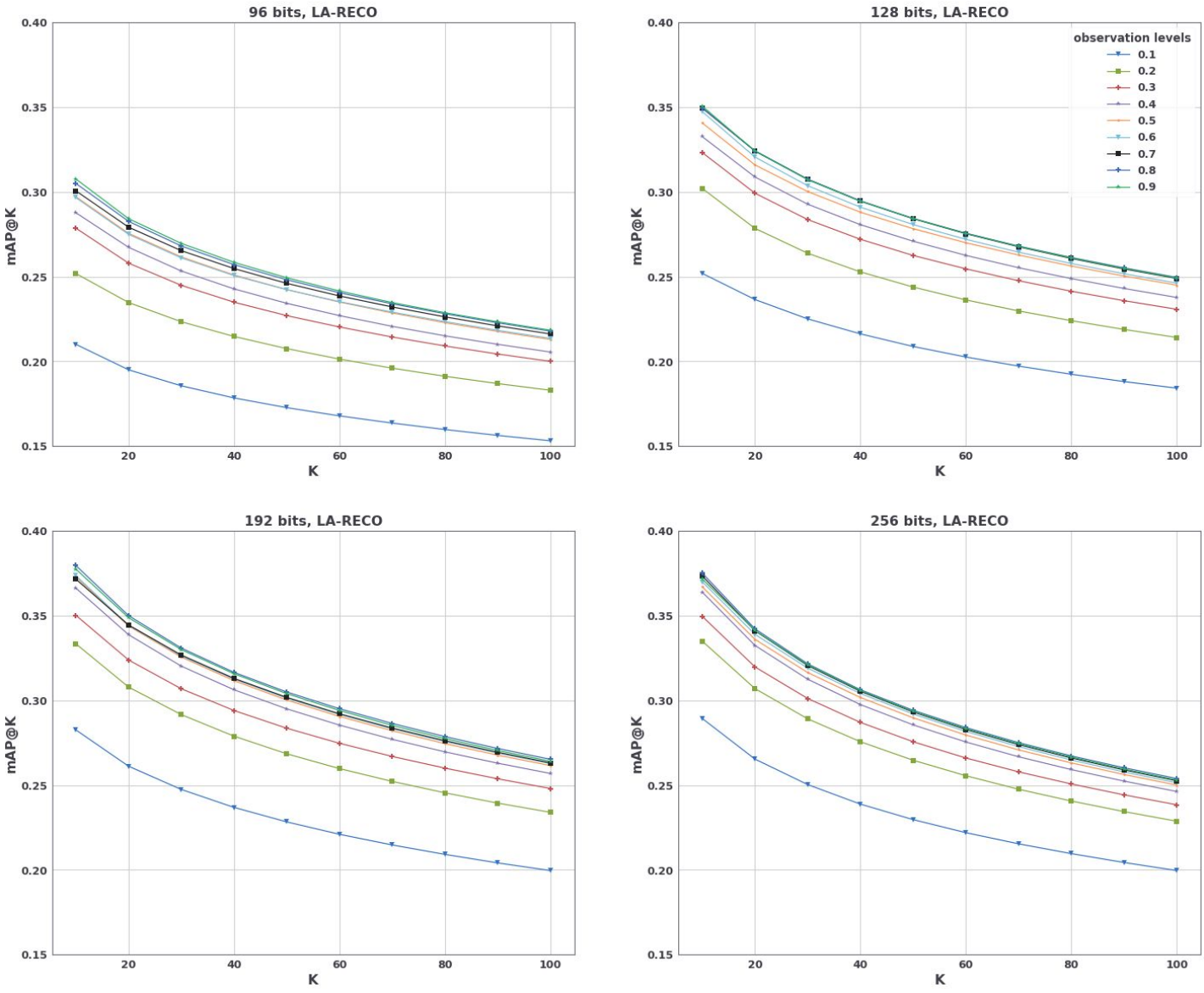


Figure C.4: FCVID: results for LA-RECO

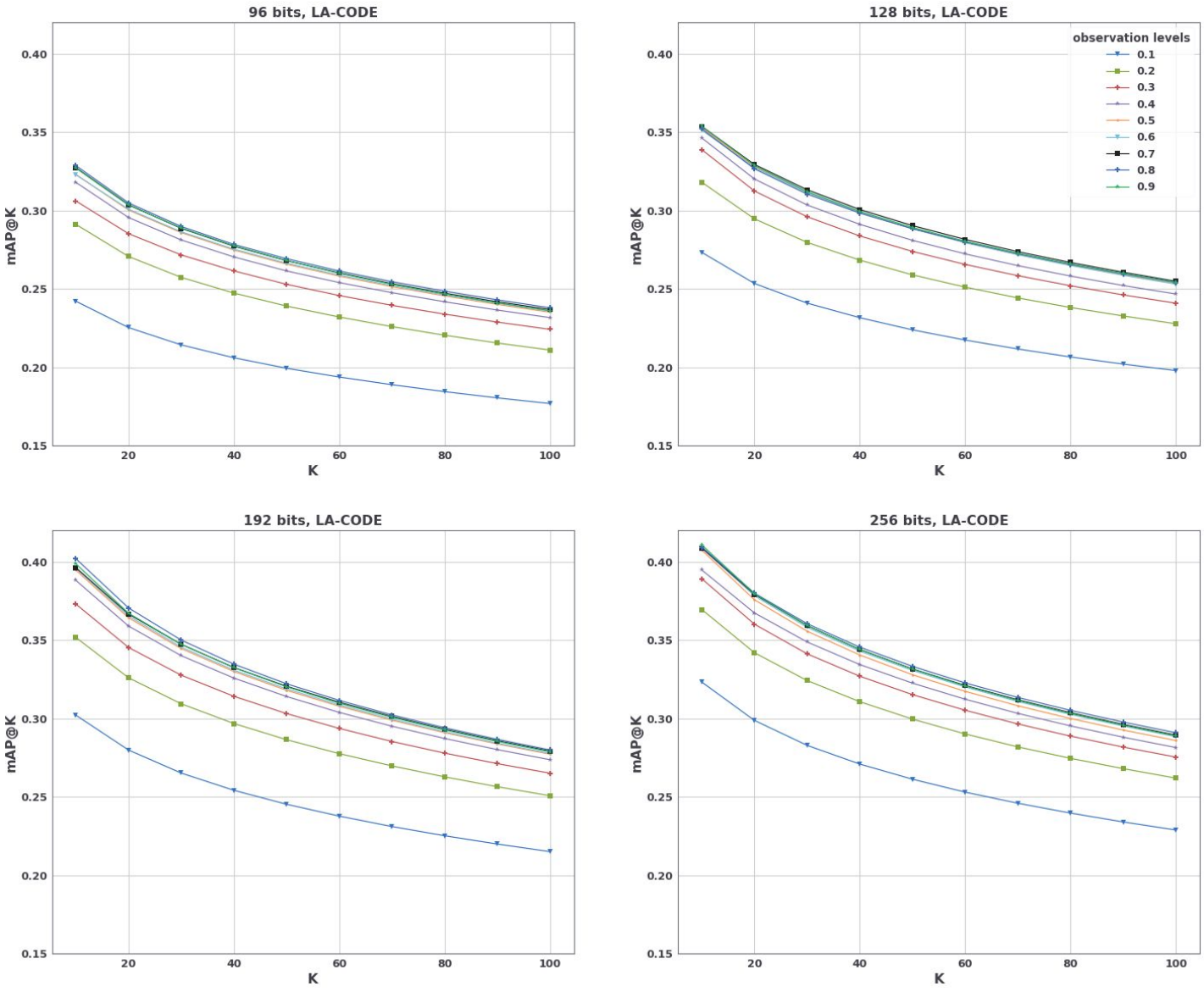


Figure C.5: FCVID: results for LA-CODE

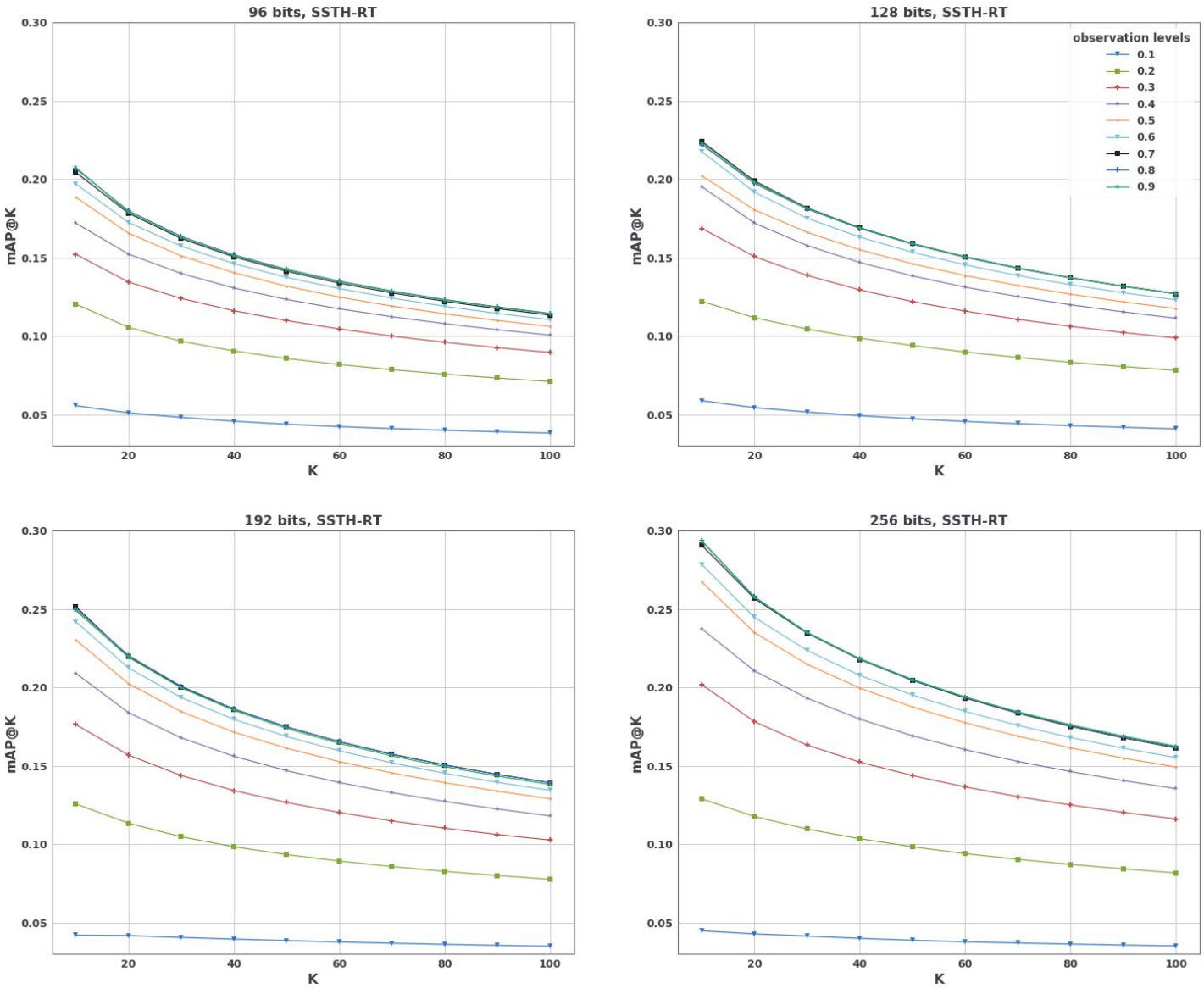


Figure C.6: ActivityNet: results for `SSTH-RT`

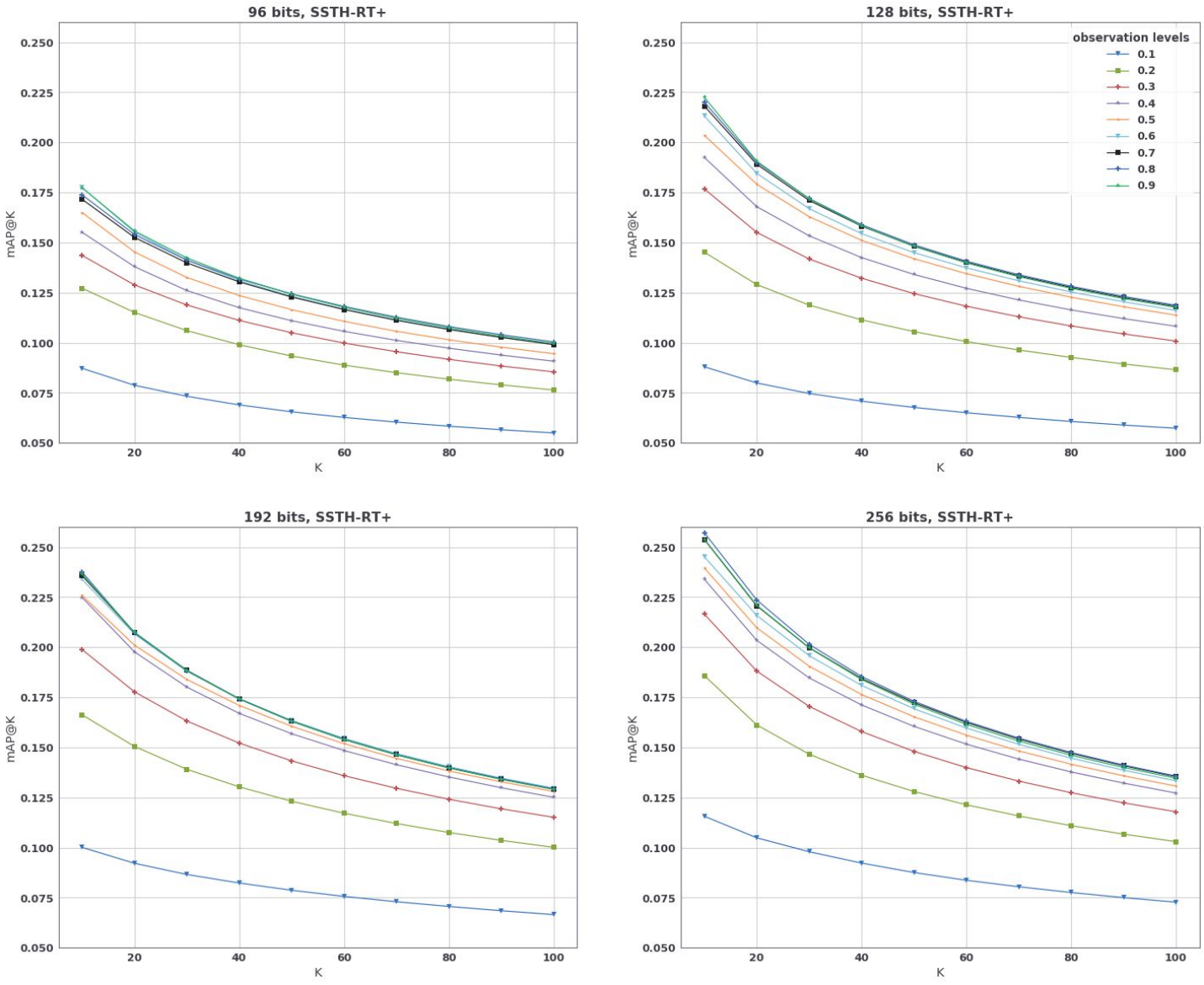


Figure C.7: ActivityNet: results for SSTH-RT$^+$

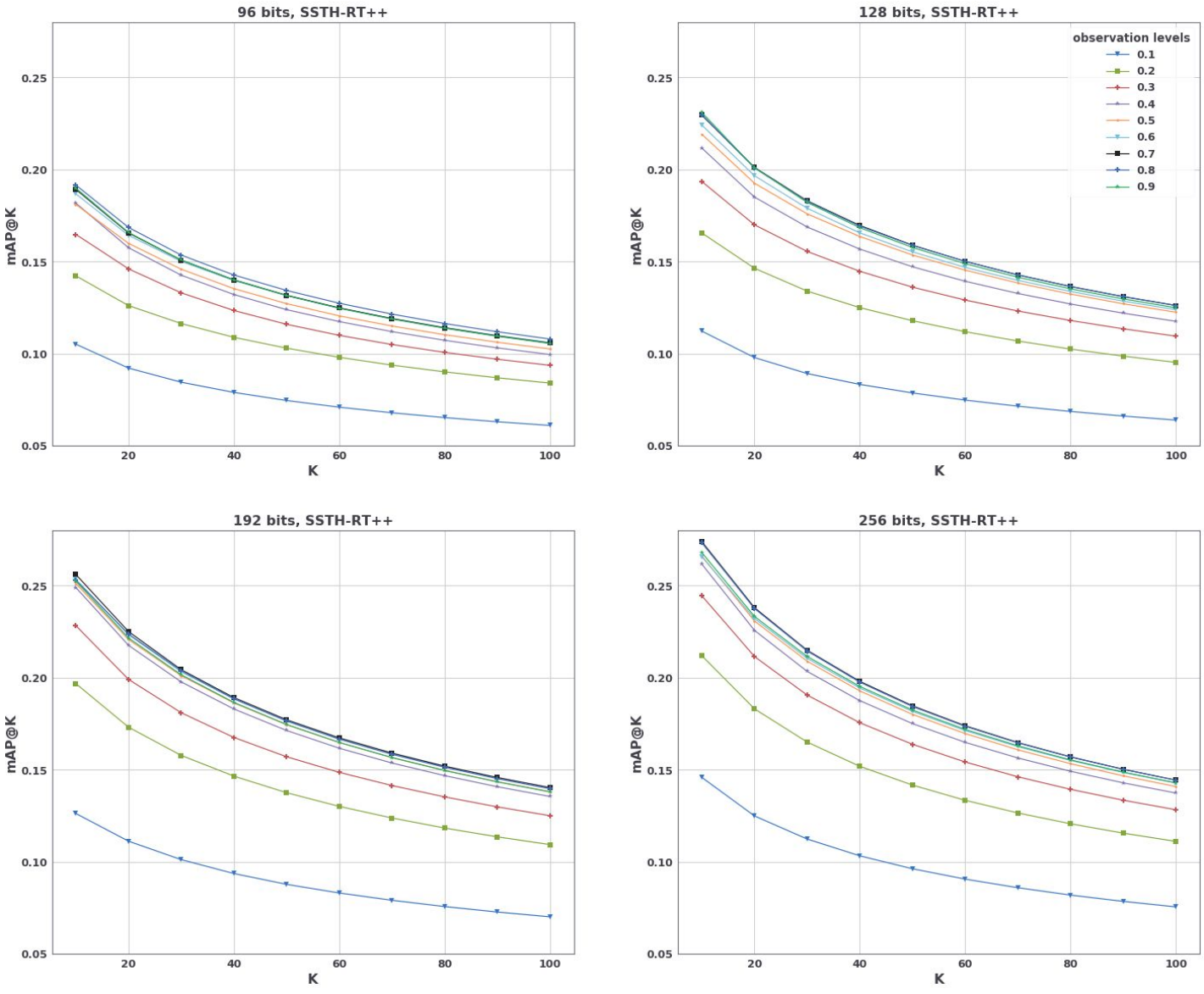


Figure C.8: ActivityNet: results for SSTH-RT^{++}

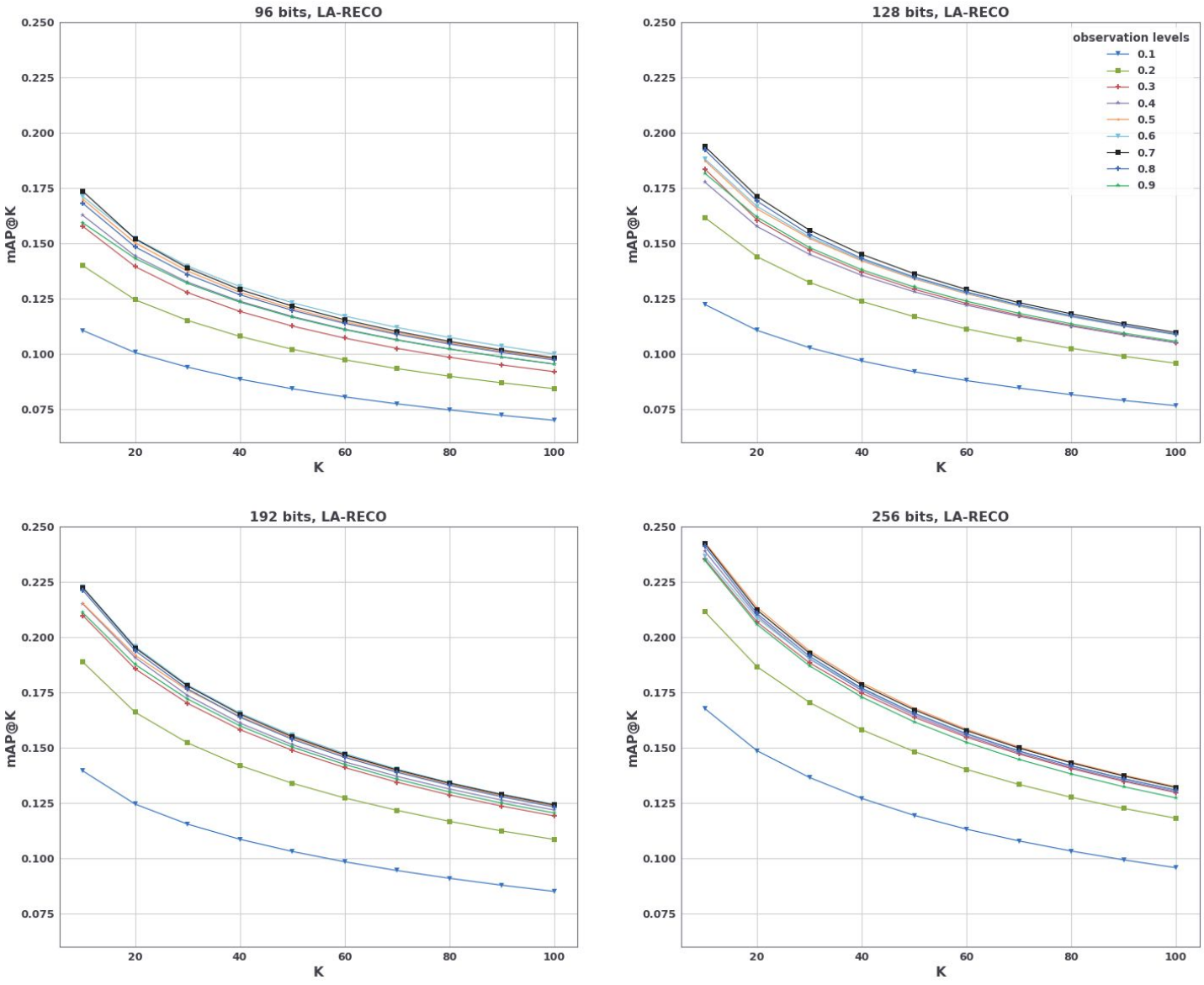


Figure C.9: ActivityNet: results for `LA-RECO`

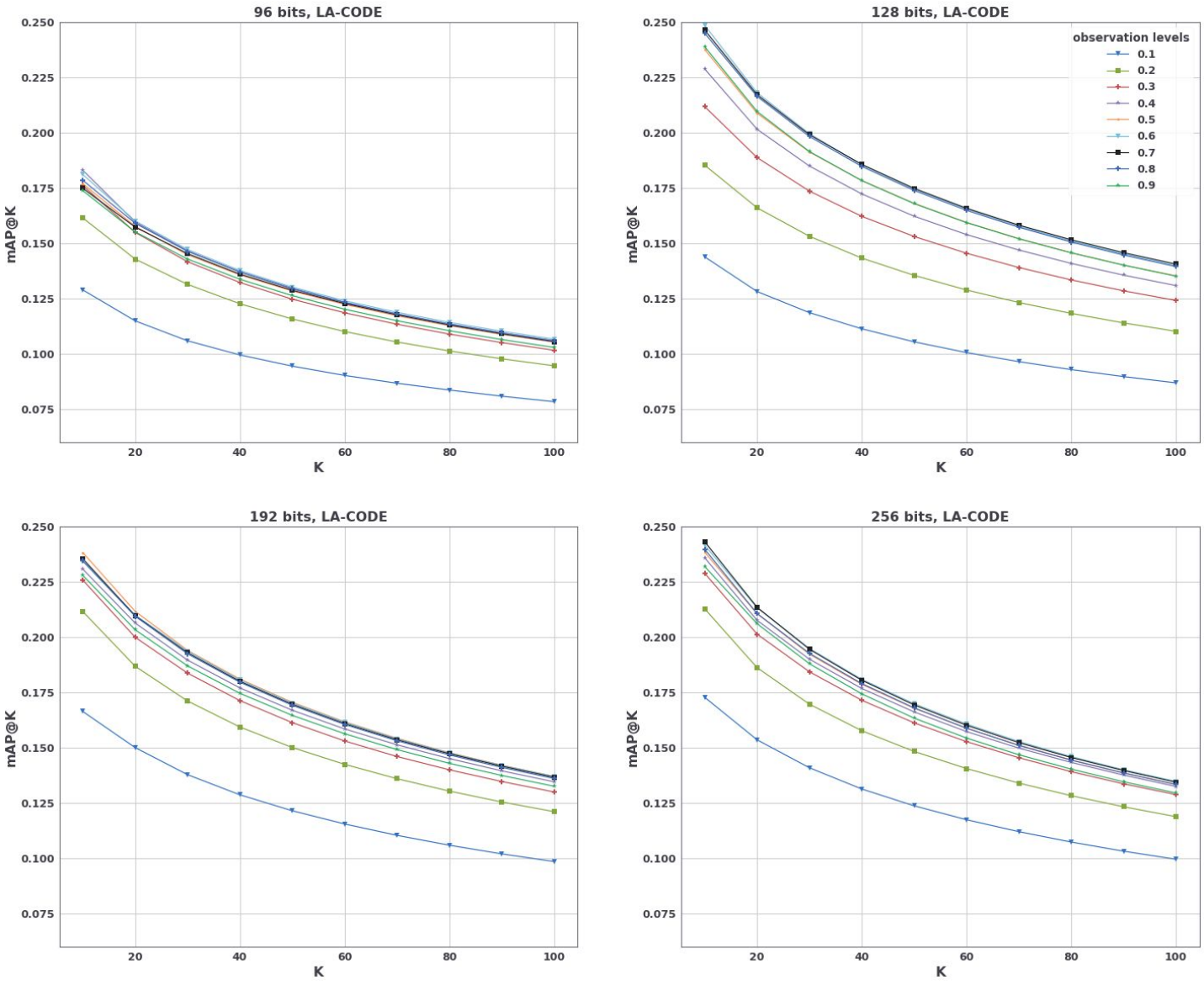


Figure C.10: ActivityNet: results for `LA-CODE`

APPENDIX D

EndoVis 2019 Surgical Workflow challenge

The EndoVis 2019 Surgical Workflow Challenge was organized as a side event for the 2019 edition of MICCAI. Participants were given a dataset of 24 annotated recordings of laparoscopic cholecystectomy from Heidelberg University Hospital, named *HeiChole*, to be used as training material for three recognition tasks: surgical phase recognition, instrument recognition and action recognition.

After the competition, submissions remained open, inviting non-competing teams such as ours to contribute solutions for the final publication. Our submission as Team CAMMA 1 took on the surgical phase recognition task.

D.1 Methods

D.1.1 Overview

Two visual feature extractors are employed in our approach: Inception-Resnet (IRN) [SIVA17] and Inflated-3D (I3D) [CZ17]. Features extracted by each of them serve as input for separate LSTM models. A third LSTM trained on the combined features is also added. Predictions from the three temporal models are ensembled with majority voting to obtain the final predicted surgical phase class (Figure D.1).

D.1.2 Data preparation

We used 4-fold cross-validation for our experiments, splitting the dataset in a training set of 18 and a validation set of 6 in each fold. Frames were resized to (256, 455) pixels.

D.1.3 IRN-LSTM

IRN was initialized using Imagenet-pretrained parameters. We finetuned it on simultaneous phase & tool recognition on the training set downsampled at 5 fps. Layers up

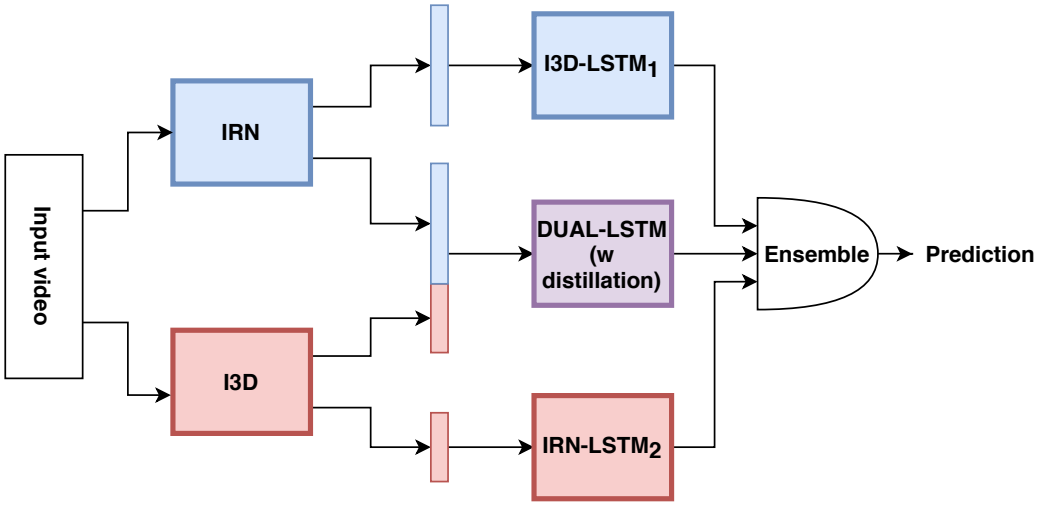


Figure D.1: Overview of the proposed model for the Endovis phase recognition challenge.

to "Mixed_7a" were frozen during this process. The 1536-d output from "Conv2d_7b" was sent to two separate fully connected layers - one for tools and one for phases.

We then used the trained IRN model to extract 1536-d feature vectors from videos at 1 fps. A 1-layer LSTM with 192 units was trained on the feature vector sequences, again on simultaneous phase & tool recognition.

D.1.4 I3D-LSTM

I3D was initialized using Kinetics-pretrained parameters. To finetune it, we supplied it with 25-frame clips from training videos at 25 fps. Layers up to "Mixed_5b" were kept frozen. The (4, 8, 15, 1024) output feature was average pooled into a (1, 1, 2, 1024) = 2048-d vector, which served as input for two separate fully connected layers for instrument and phase prediction.

2048-d feature vectors were extracted from videos at 1 fps, then fed to a 1-layer LSTM with 512 units.

D.1.5 DUAL-LSTM

In this setup both IRN and I3D are employed. We pretrained them as described in the previous sections, then concatenated features corresponding to the same timestep into a single 3584-d vector, which goes into a 1-layer LSTM with 512 units. For this LSTM we found that employing the distillation method presented in [**?**] slightly improves phase recognition performance: a bidirectional LSTM is first trained on phase & tool recognition, then the (monodirectional) LSTM is trained on the same task plus an L2 loss term forcing its hidden states to be similar to those of the bidirectional model. This is meant to incorporate anticipation into the LSTM, which does not have access to future frames in the video but must learn to mimic the behavior of a model that does.

D.1.6 Ensembling

The final phase prediction model is an ensemble of the three architectures presented above. We used majority voting as a way to aggregate the predictions from each of the networks. The contribution of each network is weighted equally.

In the dataset some phases are more common and last across more frames than others. Because of this imbalance, the networks may perform worse for the less prevalent classes. To mitigate this tendency, we double the votes for the gallbladder packaging and retraction phases during the ensembling process.

D.1.7 Model selection

Hyperparameters were selected based on average validation performance over all 4 folds. We submit two different models; the first one trained on one fold's training set, with the corresponding validation set used for early stopping. The other one is trained on all 24 videos, with training stopped on the average stopping epoch over 4 folds.

D.1.8 Experimental setup

Tensorflow 1.14 was used for all experiments. Models were trained on servers fitted with NVIDIA GTX1080Ti or P100 GPUs.

D.2 Conclusion

Several methods were incorporated into our approach in order to capture as much information as possible from a limited dataset of videos: multitask phase and instrument training, two sets of visual features from a 2D and a 3D CNN, three separate LSTMs - one of them having been trained with distillation from a bidirectional model. **With an average F1 score of 68.78%, our team placed first out of 9 contributing teams**.

APPENDIX E

Critical events in CEV64

The *CEV64* dataset was built from 64 complete recordings of surgical interventions taken from *Endocorpus*: its objective was to study moments in surgery with a high degree of clinical importance, for reasons detailed in Table E.1. The value in terms of clinical application is substantial: documentation, surgical reports, education, training and skill assessment are a few examples that can be envisioned outside of the OR. Intra-operatively, critical event lookup may serve as a powerful monitoring reference tool as mentioned many times throughout this work.

Unlike the most prominent datasets for surgical activity understanding - such as *Cholec80* - which focus on a single type of procedure, here a wide range of abdominal procedures is covered with 6 distinct types featured:

- **Bypass**
- **Cholecystectomy**
- **Eventration**
- **Hernia**
- **Nissen**
- **Sigmoidectomy**

Events tend to be well-spread across videos in the dataset as shown in Figure E.1, suggesting their high potential for generalizability. Most events can be found in nearly half the videos.

Annotations were defined then performed by two hepato-biliary surgeons from IHU Strasbourg, Dr. P. Mascagni & Dr. J. Verde, using the *MOSaiC* video annotation platform. For each instance of an event, the start and the end were set by the annotator. Annotated videos were then broken into clips of 30 seconds each for retrieval; each clip was marked with a given event if said event overlapped with it for over 50% of the clip duration. Statistics for collected clips are presented in Figure E.2.

Event classes are noticeably imbalanced; "Out of body" is particularly predominant with over 3200 clips while "sealing" tends to be quite rare, with under 100 clips. The

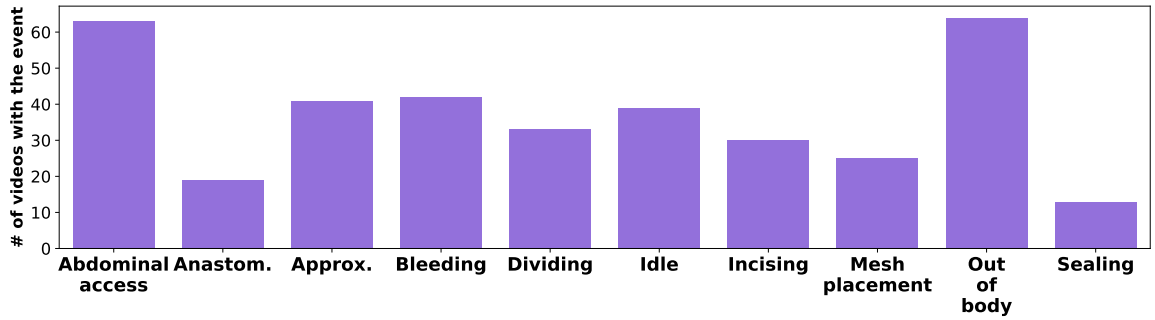


Figure E.1: Critical event spread in *CEV64* videos.

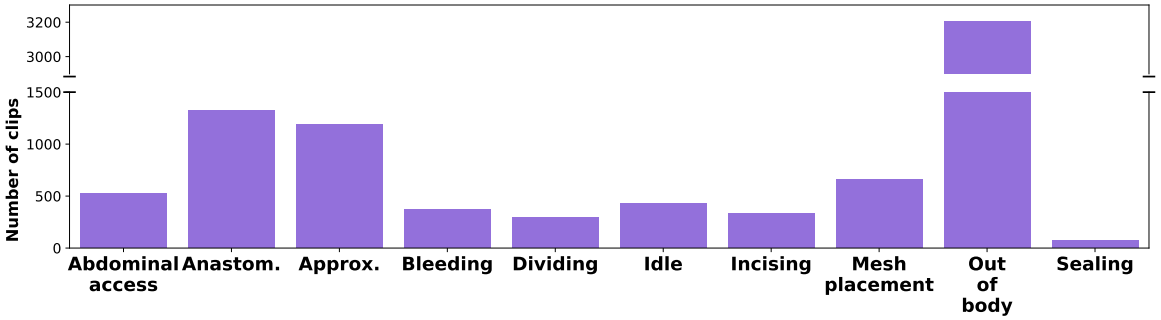


Figure E.2: Number of clips collected per event type.

rest tends to be mostly balanced with a few hundred clips each. In Chapter 6's retrieval experiments, query and codebook sets were therefore sampled evenly across all classes. To the 10 reported event classes, a final *background* class was added with clips sampled outside of any marked critical event, thus completing the 11 classes found in *CEV64*.

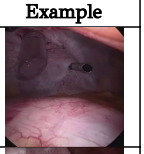
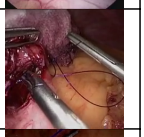
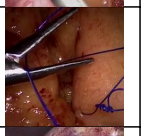
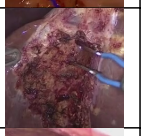
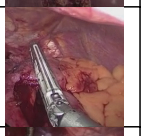
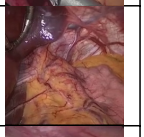
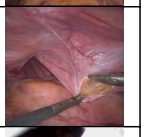
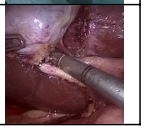
	Description	Motivation	Example
Abdominal access	Trocar insertion or removal	Perforation injury risk (insertion), abdominal wall hemostasis (removal)	
Anastomosing	Hollow organ approximation	Injury risk, leaks	
Approximating	Sutures for leaks, hernia defects	Injury risk, leaks	
Bleeding	Active bleeding that requires cauterization and / or cleanup	Factor in recovery and overall clinical outcome	
Dividing	Transection of tissue	Injury risk	
Idle	Prolonged inactivity or waiting period	Workflow interruption	
Incising	Cuts made without full transection	Injury risk	
Mesh placement	Placement of mesh for damaged tissue support	Primary goal in abdominal wall procedures	
Out of body	Laparoscope exiting the abdominal cavity	Privacy threat	
Sealing	Application of clips or sutures for vessel ligation	Injury risk, leaks	

Table E.1: Description of critical events reported in *CEV64*

APPENDIX F

Uncertain bit flagging

As stated in Chapter 6, the position of exactly k uncertain bits in a $2n$-bit array can be flagged using **strictly** fewer than $2n$ bits, with the following upper bound:

$$d - log_2\binom{d}{k} > \frac{1}{2}\, log_2(\frac{\pi \cdot d}{2}) \tag{F.1}$$

The number of possible configurations for the position of k uncertain bits in an array of size $2n$ is given by the binomial coefficient:

$$\binom{2n}{k} = \frac{(2n)!}{(2n-k)!(k!)} \tag{F.2}$$

The maximum is reached for the **central binomial coefficient** $\binom{2n}{n} = \frac{(2n)!}{(n!)^2}$. Indeed, $\binom{2n}{k+1} = \frac{2n-k}{k+1}\binom{2n}{k} > \binom{2n}{k}$ for $k < n$. Therefore $\binom{2n}{k}$ monotonously increases with k over the left side of n, and monotonously decreases over the right side due to $\binom{2n}{k} = \binom{2n}{2n-k}$, leaving $\binom{2n}{n}$ as the maximum.

This means we have at most $\binom{2n}{n}$ configurations to map to an m-bit array, requiring $log_2\binom{2n}{n} \leq m$. On the other hand, we set for ourselves $m < 2n$ as our limit.

A useful result published by Dutton [DB86] states:

$$\frac{1}{n+1}\binom{2n}{n} < \frac{4^n}{(n+1)\sqrt{\pi n}} \tag{F.3}$$

Or, equivalently:

$$\binom{2n}{n} < \frac{4^n}{\sqrt{\pi n}} \tag{F.4}$$

Note that it also is an asymptotic equivalent for $\binom{2n}{n}$ (easily derived from Stirling's approximation $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$), making it a tight upper bound.

A draft for an unpublished but more accessible proof for this inequality than Dutton's [DB86] can be found at `https://mathoverflow.net/a/133752` (credits: Noam

D. Elkies). Its author observes:

$$\binom{2n}{n} = \frac{4^n}{\pi}\int_{-\pi/2}^{\pi/2}\cos^{2n}x\,dx < \frac{4^n}{\pi}\int_{-\pi/2}^{\pi/2}e^{-nx^2}dx < \frac{4^n}{\pi}\int_{-\infty}^{\infty}e^{-nx^2}dx = \frac{4^n}{\sqrt{\pi n}}. \quad \text{(F.5)}$$

Taking log_2 of the last term yields the following upper bound:

$$log_2\binom{2n}{n} < 2n - \frac{1}{2}\,log_2(\pi \cdot n) \quad \text{(F.6)}$$

Meaning in the worst case scenario of n bits to flag out of $2n$, we can still use $m = \lceil 2n - \frac{1}{2}\,log_2(\pi \cdot n)\rceil$ and strictly remain under the $2n$ bit limit, for $n > 1$ ($n = 1$ trivially requires 1 bit).

The compression rate $log_2(\pi n)/4n$ tends to 0 as n tends towards $+\infty$; however it should be noted that this is in the one worst case of $k = n$. Values closer to 0 or $2n$ lead to much more favorable compression rates.

APPENDIX G

Résumé en français

G.1 Introduction

En l'espace de quelques années, les techniques de vision par ordinateur ont progressé de manière considérable grâce aux récents développements réalisés dans le domaine de l'apprentissage profond. Ces techniques, désormais capables de rivaliser avec la vision humaine sur un grand nombre de tâches visuelles, permettent d'offrir une assistance en temps réel dans de nombreux domaines d'application où les enjeux sont importants, la marge d'erreur est faible et la cadence de travail est soutenue.

Un tel système d'assistance, construit à partir d'algorithmes de vision performants, serait donc particulièrement adéquat au sein d'un bloc opératoire pour venir en aide au chirurgien et au personnel clinique. En particulier, le flux vidéo intra-abdominal utilisé en chirurgie minimalement invasive est une source d'information extrêmement riche, dont le contenu peut être analysé pour fournir au praticien en temps réel des informations pertinentes sur l'action ou l'événement en cours de réalisation. Cependant, un obstacle majeur au développement de ces algorithmes est la rareté des données annotées; entraîner un réseau de neurones de manière complètement supervisée en requiert une quantité considérable.

La base de données dont nous disposons est vaste: plus de 4800 heures de procédures chirurgicales complètes, réparties sur 12 types de procédures mini-invasives en chirurgie abdominale. Les annotations sont cependant fortement limitées: sur les 518 cholecystectomies et 318 pontages gastriques disponibles, respectivement 120 et 40 sont annotées avec leur décomposition en phases. En l'absence d'annotations, cette base de donnée est pour l'essentiel inexploitée et inexplorée. Collecter plus d'annotations semble être une solution évidente; à grande échelle cependant celle-ci entraînerait des coûts et des délais prohibitifs, d'où la nécessité de solutions alternatives.

Ce constat est le point de départ des travaux présentés dans cette thèse, où nous cherchons à utiliser et explorer cette immense réserve de données brutes pour obtenir des modèles temps réel performants sur des tâches d'analyse vidéo en chirurgie minimalement invasive.

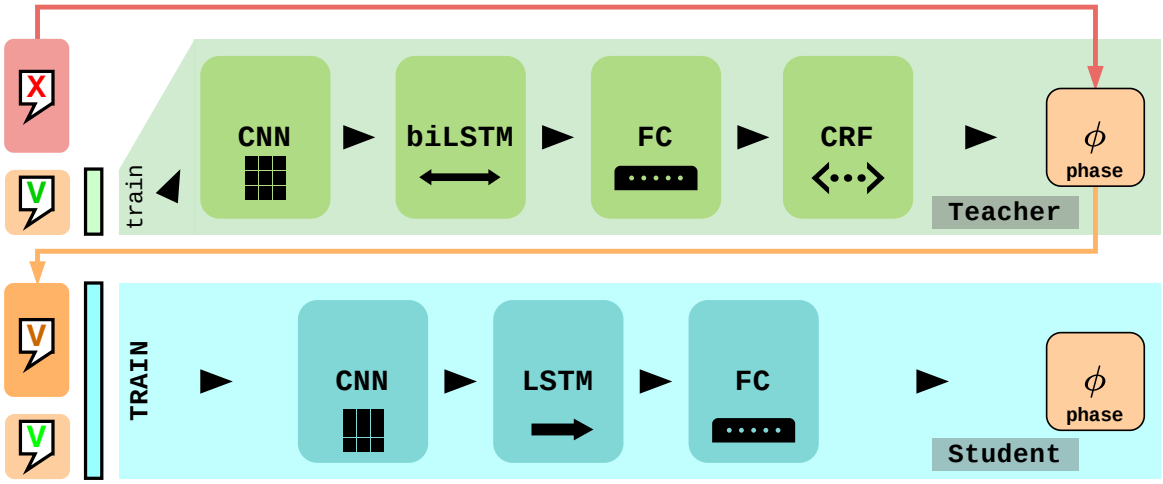


Figure G.1: Le modèle CNN-biLSTM-CRF (*maître*) annote des données pour le CNN-LSTM (*élève*), améliorant ainsi ses performances en reconnaissance de phase chirurgicale.

G.2 Méthodes

G.2.1 Annotations automatiques pour l'identification temps réel des phases

Une première approche pour naviguer au sein d'une base de données brutes, en l'absence d'annotateurs experts humains, consiste à remplacer ces derniers par un modèle entraîné sur un jeu de données réduit. Nous nous intéressons ici à l'un des problèmes fondamentaux de l'analyse de vidéo chirurgicales, à savoir la reconnaissance des phases. Conformément à notre hypothèse de données annotées rares, nous procèdons de manière semi-supervisée: sur 80 vidéos, nous nous autorisons à utiliser les annotations manuelles sur seulement 1 à 20 vidéos. Les modèles utilisés dans ce problème sont sujets à un compromis: temps réel avec performances limitées (CNN-LSTM), ou a posteriori avec performances maximales (CNN-biLSTM-CRF introduit dans nos travaux).

Nous parvenons à dépasser ce compromis en utilisant les deux types de modèle de manière complémentaire G.1: le modèle a posteriori, dans le rôle du *maître*, est entraîné sur les 1 à 20 vidéos manuellement annotées, puis annote automatiquement le reste des 80 enregistrements. L'ensemble sert à entraîner le modèle temps réel, qui tient le rôle d'*élève*. Le F1-score ainsi obtenu est de 78.2 %, contre 70.2 % avec les 20 vidéos manuellement annotées sans annotations automatiques supplémentaires.

G.2.2 Fouille temps réel de vastes bases de données vidéo génériques

Obtenir des informations sur du contenu vidéo à partir d'un classifieur entraîné à reconnaître un nombre prédéterminé de classes est l'approche canoniquement préconisée en vision par ordinateur, qui pourrait être qualifiée de *directe* et *explicite*. À l'inverse, se renseigner sur un contenu en l'utilisant pour rechercher les éléments pertinents d'une vaste base de donnée est une approche indirecte et implicite, qui propose davantage

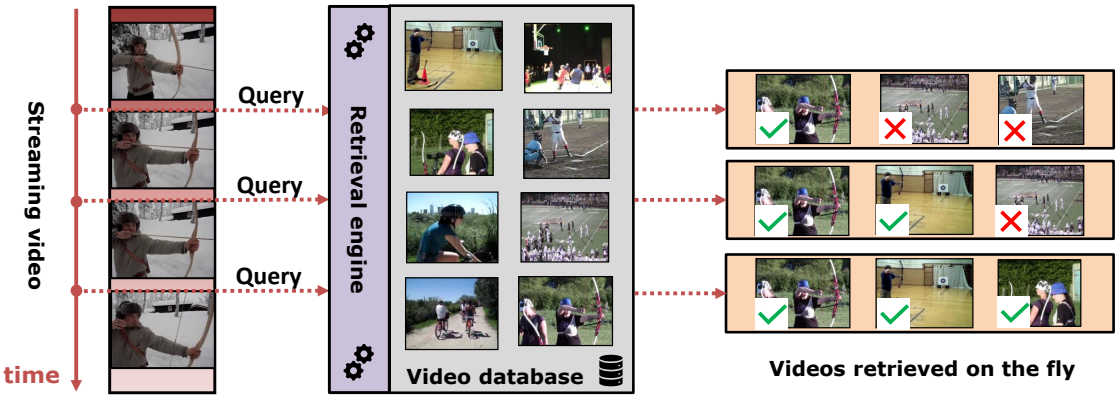


Figure G.2: Notre technique de hachage permet de fouiller à l'intérieur de bases de données vidéo de manière incrémentale, au fur et à mesure du déroulement en direct de la vidéo utilisée comme requête. Un archer est filmé; le flux vidéo est soumis au moteur de recherche, qui retrouve deux résultats corrects sur trois

de flexibilité et de polyvalence.

Les fonctions de hachage profondes constituent une méthode efficace pour la fouille de bases de données; celles-ci calculent des clés binaires de taille réduite à partir de contenus de très grandes dimensions. Ces clés étant optimisées pour préserver les similarités et différences entre contenus, il est alors possible de rapidement effectuer des requêtes par simples comparaisons binaires. Les fonctions de hachage sont souvent entraînées de manière auto-supervisée; leur fonctionnement est dans ce cas complètement indépendant des annotations, qui servent uniquement à leur évaluation quantitative. Le hachage présente donc un double intérêt dans notre situation, en permettant d'utiliser les données non annotées et de les explorer via des requêtes.

La vidéo est cependant fortement en retard par rapport aux autres modalités de recherche telles que les images statiques concernant le hachage. Les quelques travaux traitant de ce problème pour les vidéos ne considèrent de plus que la recherche a posteriori, proposant des méthodes incompatibles avec des recherches temps réel à partir de flux vidéo instantanés.

Le cas temps réel G.2 est justement celui que nous choisissons de traiter: considérant une vidéo de durée totale T, nous souhaitons rechercher au sein d'une base de données les vidéos les plus similaires, en observant uniquement les αT premières secondes pour $\alpha \in]0, 1]$ - cette situation correspond au cas d'une diffusion en direct, ou du *streaming*. Pour cela nous proposons un encodeur vidéo binaire incrémental et prédictif: un modèle LSTM met à jour la clé binaire toutes les deux secondes environ afin que celle-ci tienne compte du contenu visuel en temps réel. Ce modèle est de plus entraîné à anticiper le futur contenu vidéo, afin de compenser le manque d'information visuelle dû à la diffusion en direct.

Nous présentons notre méthode sur deux jeux de données vidéo génériques de très grande taille: FCVID et ActivityNet. L'évaluation s'effectue à partir des étiquettes d'activité fournis par ces jeux de données: dans le classement des meilleurs résultats de recherche, un résultat est considéré correct si sa classe correspond à celle de la requête. Notre méthode, `LA-CODE`, surpasse les méthodes non prédictives avec notamment 27.4 % de mAP@20 sur FCVID, 14.7 % sur ActivityNet avec moins de 30% de la vidéo observés, pour des clés binaires de 128 bits.

G.2.3 Fouille temps réel de vastes bases de données vidéo chirurgicales

Pour mettre à l'épreuve nos méthodes de fouilles de données vidéo sur du contenu chirurgical, nous introduisons le jeu de données *Endocorpus*, contenant 1558 enregistrements complets de 12 types de procédures minimalement invasives. Ces enregistrements sont divisés en clips de 30 secondes environ, formant une base de données de plus de 500000 clips.

Nous montrons qu'un même modèle, entraîné sur un vaste ensemble de clips non annotés, permet de rechercher du contenu au sein de trois bases de données: cholecystectomie, pontage gastrique et événements chirurgicaux critiques. L'évaluation s'effectue à partir des phases pour les deux premières; pour la dernière, à partir du type d'événement critique.

À cela, nous ajoutons une nouvelle méthode de fouille tenant compte de l'incertitude sur les bits des clés binaires représentant les vidéos. Nommée `ULA-CODE`, cette méthode consiste à comprimer un masque signalant les bits incertains puis à stocker ce masque avec les clés binaires. En décompressant ce masque et en l'utilisant au moment d'une requête pour réduire la contribution des bits incertains, la pertinence des résultats retrouvés augmente de 2 à 4%..

G.3 Conclusion

Les méthodes proposées permettent de tirer parti avec succès des données vidéo endoscopiques non annotées, permettant ainsi de les annoter automatiquement pour l'entraînement de classifieurs temps réel, de les utiliser pour l'entraînement autosupervisé de modèles pour la fouille de bases de données vidéo, et d'explorer ces dernières par le biais de requêtes vidéo instantanées. Ce succès est encourageant pour un passage à l'échelle à partir d'annotations raréfiées, ainsi que le déploiement dans le bloc opératoire de ces méthodes temps réel pour assister le chirurgien.

Bibliography

[AI08] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

[AIK+18] Sana Amanat, Muhammad Idrees, M. U. Khan, Z. Rehman, Hangbae Chang, I. Mehmood, and S. Baik. Video retrieval system for meniscal surgery to improve health care services. *J. Sensors*, 2018:4390703:1–4390703:10, 2018.

[ASNS20] Anurag Arnab, Chen Sun, Arsha Nagrani, and Cordelia Schmid. Uncertainty-aware weakly supervised action detection from untrimmed videos. In *ECCV 2020*, 2020.

[ASS+17] Mohammad Sadegh Ali Akbarian, Fatemehsadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging lstms to anticipate actions very early. In *IEEE International Conference on Computer Vision (ICCV)*, pages 280–289, 2017.

[Bai93] Henry S. Baird. Document image defect models and their uses. In *2nd International Conference Document Analysis and Recognition, ICDAR '93, October 20-22, 1993, Tsukuba City, Japan*, pages 62–67. IEEE Computer Society, 1993.

[BFN10] Tobias Blum, Hubertus Feussner, and Nassir Navab. Modeling and segmentation of surgical workflow from laparoscopic video. volume 13, pages 400–7, 09 2010.

[BKC+14] Konstantin Berlin, Sergey Koren, Chen-Shan Chin, James Drake, Jane M. Landolin, and Adam M. Phillippy. Assembling large genomes with single-molecule sequencing and locality sensitive hashing. *bioRxiv*, 2014.

[BLG+15] P. Bojanowski, R. Lajugie, E. Grave, F. Bach, I. Laptev, J. Ponce, and C. Schmid. Weakly-supervised alignment of video with text. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4462–4470, 2015.

[BML+20] M. Bretonnier, E. Michinov, E. Le Pabic, P.-L. Hénaux, P. Jannin, X. Morandi, and L. Riffaud. Impact of the complexity of surgical procedures and intraoperative interruptions on neurosurgical team workload. *Neurochirurgie*, 66(4):203–211, 2020.

[BMS+11] Deepraj Bhandarkar, Gaurav Mittal, Rasik Shah, Avinash Katara, and Tehemton E. Udwadia. Single-incision laparoscopic cholecystectomy: How i do it? *Journal of minimal access surgery*, 7(1):17–23, Jan 2011.

[Bon17] H. Bonjer. *Surgical Principles of Minimally Invasive Procedures: Manual of the European Association of Endoscopic Surgery (EAES)*. 01 2017.

[BWK+17] Sebastian Bodenstedt, Martin Wagner, Darko Katic, Patrick Mietkowski, Benjamin F. B. Mayer, Hannes Kenngott, Beat P. Müller-Stich, Rüdiger Dillmann, and Stefanie Speidel. Unsupervised temporal context learning using convolutional neural networks for laparoscopic workflow analysis. *CoRR*, abs/1702.03684, 2017.

[CBL+17] Tim Cooijmans, Nicolas Ballas, César Laurent, Çaglar Gülçehre, and Aaron C. Courville. Recurrent batch normalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[CCL+18] Z. Chen, Ruojin Cai, Jiwen Lu, J. Feng, and J. Zhou. Order-sensitive deep hashing for multimorbidity medical image retrieval. In *MICCAI*, 2018.

[cho93] National institutes of health consensus development conference statement on gallstones and laparoscopic cholecystectomy. *American journal of surgery vol. 165,4*, 1993.

[CKNH20] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 2020.

[CLLW18a] Yue Cao, Bin Liu, Mingsheng Long, and Jianmin Wang. Hashgan: Deep learning to hash with pair conditional wasserstein GAN. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1287–1296. IEEE Computer Society, 2018.

[CLLW18b] Yue Cao, Mingsheng Long, Bin Liu, and Jianmin Wang. Deep cauchy hashing for hamming space retrieval. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1229–1237. IEEE Computer Society, 2018.

[CLWY17] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Philip S. Yu. Hashnet: Deep learning to hash by continuation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5609–5618. IEEE Computer Society, 2017.

[CMG18] Christopher P. Childers and Melinda Maggard-Gibbons. Understanding Costs of Care in the Operating Room. *JAMA Surgery*, 153(4):e176233–e176233, 04 2018.

[CPK+20] Tobias Czempiel, Magdalini Paschali, Matthias Keicher, Walter Simson, Hubertus Feussner, Seong Tae Kim, and Nassir Navab. Tecno: Surgical phase recognition with multi-stage temporal convolutional networks. In *Medical Image Computing and Computer Assisted Intervention - MICCAI 2020 - 23rd International Conference, Lima, Peru, October 4-8, 2020, Proceedings, Part III*, volume 12263 of *Lecture Notes in Computer Science*, pages 343–352. Springer, 2020.

[CZ17] J. Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, 2017.

[CZZ+20] Lele Cheng, Xiangzeng Zhou, Liming Zhao, Dangwei Li, Hong Shang, Yun Zheng, Pan Pan, and Yinghui Xu. Weakly supervised learning with side information for noisy labeled images. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 306–321. Springer, 2020.

[DB86] Ronald D. Dutton and Robert C. Brigham. Computationally efficient bounds for the catalan numbers. *European Journal of Combinatorics*, 7(3):211–213, 1986.

[DBH+16] Olga Dergachyova, David Bouget, Arnaud Huaulmé, Xavier Morandi, and Pierre Jannin. Automatic data-driven real-time segmentation and recognition of surgical workflow. *Int. J. Comput. Assist. Radiol. Surg.*, 11(6):1081–1089, 2016.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[DG13] Dengxin Dai and Luc Van Gool. Ensemble projection for semi-supervised image classification. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 2072–2079. IEEE Computer Society, 2013.

[DHG+15] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Trevor Darrell, and Kate Saenko. Long-term recurrent convolutional networks for visual recognition and

description. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2625–2634, 2015.

[DQL+14] Zakarya Droueche, Gwenole Quellec, Mathieu Lamard, Guy Cazuguel, Béatrice Cochener, and Christian Roux. Computer-aided retinal surgery using data from the video compressed stream. *International Journal Of Image And Video Processing: Theory And Application*, 1:1–10, 04 2014.

[EDH+20] Wayne English, Eric Demaria, Matthew Hutter, Shanu Kothari, Samer Mattar, Stacy Brethauer, and John Morton. "american society for metabolic and bariatric surgery 2018 estimate of metabolic and bariatric procedures performed in the united states". *Surgery for Obesity and Related Diseases*, 16, 01 2020.

[Fei20] Christoph Feichtenhofer. X3D: expanding architectures for efficient video recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 200–210. IEEE, 2020.

[FJM+18] Isabel Funke, Alexander Jenke, Sören Torge Mees, Jürgen Weitz, Stefanie Speidel, and Sebastian 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*, pages 85–93. Springer, 2018.

[fPZY+09] Yuan fei Peng, M. Zheng, Qing Ye, X. Chen, Beiqin Yu, and B. Liu. Heated and humidified co2 prevents hypothermia, peritoneal injury, and intra-abdominal adhesions during prolonged laparoscopic insufflations. *The Journal of surgical research*, 151 1:40–7, 2009.

[Fri16] Bernhard Friedrich. *The Effect of Autonomous Vehicles on Traffic*, pages 317–334. 05 2016.

[Gaw12] Atul Gawande. Two hundred years of surgery. *New England Journal of Medicine*, 366(18):1716–1723, 2012.

[GCDZ19] Rohit Girdhar, João Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 244–253. Computer Vision Foundation / IEEE, 2019.

[GDSF19] Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Predicting the future: A jointly learnt model for action anticipation. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[GLGP13] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes

for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2916–2929, 2013.

[GMH13] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. IEEE, 2013.

[GP21] Antoine Genitrini and Martin Pépin. Lexicographic unranking of combinations revisited. *Algorithms*, 14:97, 2021.

[GS05] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. 18:602–10, 07 2005.

[GSA+20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020.

[GVYY17] Yun Gu, Khushi Vyas, Jie Yang, and G. Yang. Unsupervised feature learning for endomicroscopy image retrieval. In *MICCAI*, 2017.

[GYMT21] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *Int. J. Comput. Vis.*, 129(6):1789–1819, 2021.

[HCS+16] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4107–4115, 2016.

[HDTC17] Tuan Hoang, Thanh-Toan Do, Dang-Khoa Le Tan, and Ngai-Man Cheung. Selective deep convolutional features for image retrieval. In Qiong Liu, Rainer Lienhart, Haohong Wang, Sheng-Wei "Kuan-Ta" Chen, Susanne Boll, Yi-Ping Phoebe Chen, Gerald Friedland, Jia Li, and Shuicheng Yan, editors, *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017*, pages 1600–1608. ACM, 2017.

[HEGN15] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 961–970, 2015.

[HLC+18] Hassan Al Hajj, Mathieu Lamard, Pierre-Henri Conze, Béatrice Cochener, and Gwenolé Quellec. Monitoring tool usage in surgery videos using boosted convolutional and recurrent neural networks. *Medical Image Analysis*, 47:203–218, 2018.

[HLH+12] J. Heo, Y. Lee, J. He, S. Chang, and S. Yoon. Spherical hashing. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2957–2964, 2012.

[HMHV20] Gregory D. Hager, Lena Maier-Hein, and S. Swaroop Vedula. Chapter 38 - surgical data science. In S. Kevin Zhou, Daniel Rueckert, and Gabor Fichtinger, editors, *Handbook of Medical Image Computing and Computer Assisted Intervention*, The Elsevier and MICCAI Society Book Series, pages 931–952. Academic Press, 2020.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[HWC19] Xiangyu He, Peisong Wang, and Jian Cheng. K-nearest neighbors hashing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2839–2848. Computer Vision Foundation / IEEE, 2019.

[HWY+18] Houdong Hu, Yan Wang, Linjun Yang, Pavel Komlev, Li Huang, Xi (Stephen) Chen, Jiapei Huang, Ye Wu, Meenaz Merchant, and Arun Sacheti. Web-scale responsive visual search at bing. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 359–367, New York, NY, USA, 2018. Association for Computing Machinery.

[HZM+19] J. Hu, W. Zheng, L. Ma, G. Wang, J. Lai, and J. Zhang. Early action prediction by soft regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2568–2583, 2019.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 604–613. ACM, 1998.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[ITAC19] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5065–5074, 2019.

[JAiN17] Albert Jimenez, Jose M Alvarez, and Xavier Giro i Nieto. Class-weighted convolutional features for visual instance search. In *28th British Machine Vision Conference (BMVC)*, September 2017.

[JDC+18] Yueming Jin, Qi Dou, Hao Chen, Lequan Yu, Jing Qin, Chi-Wing Fu, and Pheng-Ann Heng. Sv-rcnet: Workflow recognition from surgical videos using recurrent convolutional network. *IEEE Transactions on Medical Imaging*, 37(5):1114–1126, 2018.

[JLD+19] Yueming Jin, Huaxia Li, Qi Dou, Hao Chen, Jing Qin, Chi-Wing Fu, and Pheng-Ann Heng. Multi-task recurrent convolutional network with correlation loss for surgical video analysis. *Medical Image Analysis*, 59:101572, 10 2019.

[JWW+18] Yu-Gang Jiang, Zuxuan Wu, Jun Wang, Xiangyang Xue, and Shih-Fu Chang. Exploiting feature and class relationships in video categorization with regularized deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 352–364, 2018.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[KLS18] Sabrina Kletz, Andreas Leibetseder, and Klaus Schoeffmann. Evaluation of visual content descriptors for supporting ad-hoc video search tasks at the video browser showdown. In Klaus Schoeffmann, Thanarat H. Chalidabhongse, Chong-Wah Ngo, Supavadee Aramvith, Noel E. O'Connor, Yo-Sung Ho, Moncef Gabbouj, and Ahmed Elgammal, editors, *MultiMedia Modeling - 24th International Conference, MMM 2018, Bangkok, Thailand, February 5-7, 2018, Proceedings, Part I*, volume 10704 of *Lecture Notes in Computer Science*, pages 203–215. Springer, 2018.

[KTF17] Yu Kong, Zhiqiang Tao, and Yun Fu. Deep sequential context networks for action prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3662–3670. IEEE Computer Society, 2017.

[KTS+14] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Fei-Fei Li. Large-scale video classification with convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 1725–1732. IEEE Computer Society, 2014.

[KYM+20] Siddharth Kannan, Gaurav Yengera, D. Mutter, J. Marescaux, and N. Padoy. Future-state predicting lstm for early surgery type recognition. *IEEE Transactions on Medical Imaging*, 39:556–566, 2020.

[LCL+19] Shuyan Li, Zhixiang Chen, Jiwen Lu, Xiu Li, and Jie Zhou. Neighborhood preserving hashing for scalable video retrieval. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 8211–8220, 2019.

[LJ13] Florent Lalys and Pierre Jannin. Surgical process modelling: A review. *International journal of computer assisted radiology and surgery*, 9, 09 2013.

[LLTZ17] Venice Erin Liong, Jiwen Lu, Yap-Peng Tan, and Jie Zhou. Deep video hashing. *IEEE Transactions on Multimedia*, 19(6):1209–1219, 2017.

[LLW+15] Venice Erin Liong, Jiwen Lu, Gang Wang, Pierre Moulin, and Jie Zhou. Deep hashing for compact binary codes learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2475–2483. IEEE Computer Society, 2015.

[LRM+19] Gurvan Lecuyer, Martin Ragot, Nicolas Martin, Laurent Launay, and Pierre Jannin. Assisted Annotation of Surgical Videos Using Deep Learning. In *Computer Assisted Radiology and Surgery*, June 2019.

[LVRH16] Colin Lea, René Vidal, Austin Reiter, and Gregory D. Hager. Temporal convolutional networks: A unified approach to action segmentation. In Gang Hua and Hervé Jégou, editors, *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, volume 9915 of *Lecture Notes in Computer Science*, pages 47–54, 2016.

[LYV+19] Chundi Liu, Guangwei Yu, Maksims Volkovs, Cheng Chang, Himanshu Rai, Junwei Ma, and Satya Krishna Gorti. Guided similarity separation for image retrieval. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[MGVY21] Junwei Ma, Satya Krishna Gorti, Maksims Volkovs, and Guangwei Yu. Weakly supervised action selection learning in video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[MHVS+17] Lena Maier-Hein, Swaroop Vedula, Stefanie Speidel, Nassir Navab, Ron Kikinis, Adrian Park, Matthias Eisenmann, Hubertus Feussner, Germain Forestier, Stamatia Giannarou, Makoto Hashizume, Darko Katić, Hannes Kenngott, Michael Kranzfelder, Anand Malpani, Keno März, Thomas Neumuth, Nicolas Padoy, Carla Pugh, and Pierre Jannin. Surgical data science for next-generation interventions. *Nature Biomedical Engineering*, 1, 09 2017.

[MLCH16] Anand Malpani, Colin Lea, Chi Chiung Grace Chen, and Gregory D. Hager. System events: readily accessible features for surgical phase detection. *Int. J. Comput. Assist. Radiol. Surg.*, 11(6):1201–1209, 2016.

[MP21] Pietro Mascagni and Nicolas Padoy. Or black box and surgical control tower: recording and streaming data and analytics to improve surgical care. *Journal of Visceral Surgery*, 158, 03 2021.

[MSS16] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1942–1950. IEEE Computer Society, 2016.

[MVA+20] Pietro Mascagni, Armine Vardazaryan, Deepak Alapatt, Takeshi Urade, Taha Emre, Claudio Fiorillo, Patrick Pessaux, Didier Mutter, Jacques Marescaux, Guido Costamagna, Bernard Dallemagne, and Nicolas Padoy. Artificial intelligence for surgical safety: Automatic assessment of the critical view of safety in laparoscopic cholecystectomy using deep learning. *Annals of Surgery*, Publish Ahead of Print, 11 2020.

[MZH16] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, volume 9905 of *Lecture Notes in Computer Science*, pages 527–544. Springer, 2016.

[NCB17] Yun Ni, Kelvin Chu, , and Joseph Bradley. Detecting abuse at scale: Locality sensitive hashing at uber engineering. eng.uber.com/lsh, 2017.

[NF16] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*, volume 9910 of *Lecture Notes in Computer Science*, pages 69–84. Springer, 2016.

[NGY+20] Chinedu Innocent Nwoye, Cristians Gonzalez, Tong Yu, Pietro Mascagni, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Recognition of instrument-tissue interactions in endoscopic videos via action triplets. In Anne L. Martel, Purang Abolmaesumi, Danail Stoyanov, Diana Mateus, Maria A. Zuluaga, S. Kevin Zhou, Daniel Racoceanu, and Leo Joskowicz, editors, *Medical Image Computing and Computer Assisted Intervention - MICCAI 2020 - 23rd International Conference, Lima, Peru, October 4-8, 2020, Proceedings, Part III*, volume 12263 of *Lecture Notes in Computer Science*, pages 364–374. Springer, 2020.

[NLX15] Li Niu, Wen Li, and Dong Xu. Visual recognition by learning from web data: A weakly supervised domain generalization approach. In *IEEE*

Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, pages 2774–2783. IEEE Computer Society, 2015.

[NMMP19] Chinedu Innocent Nwoye, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Weakly supervised convolutional LSTM approach for tool tracking in laparoscopic videos. *Int. J. Comput. Assist. Radiol. Surg.*, 14(6):1059–1067, 2019.

[NPF17] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5899–5907. IEEE Computer Society, 2017.

[NRF19] P. Nguyen, D. Ramanan, and Charless C. Fowlkes. Weakly-supervised action localization with background modeling. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5501–5510, 2019.

[OT01] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001.

[PBW+19] Tingying Peng, Melanie Boxberg, Wilko Weichert, Nassir Navab, and Carsten Marr. Multi-task learning of a deep k-nearest neighbour network for histopathological image classification and retrieval. *bioRxiv*, 2019.

[PMB13] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org, 2013.

[PNNC16] Giorgio Patrini, Frank Nielsen, Richard Nock, and Marcello Carioni. Loss factorization, weakly supervised learning and label noise robustness. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 708–717. JMLR.org, 2016.

[PRRC18] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 563–579, 2018.

[PS18] Stefan Petscharnig and Klaus Schöffmann. Binary convolutional neural network features off-the-shelf for image to video linking in endoscopic multimedia databases. *Multimedia Tools and Applications*, 77:28817–28842, 2018.

[PSMB12] Siddika Parlak, Aleksandra Sarcevic, Ivan Marsic, and Randall S. Burd. Introducing RFID technology in dynamic and time-critical medical settings: Requirements and challenges. *J. Biomed. Informatics*, 45(5):958–974, 2012.

[QYM17] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5534–5542. IEEE Computer Society, 2017.

[RARS19] Jerome Revaud, Jon Almazan, Rafael Rezende, and Cesar De Souza. Learning with average precision: Training image retrieval with a listwise loss. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5106–5115, 2019.

[RDG+18] Ilija Radosavovic, Piotr Dollár, Ross B. Girshick, Georgia Gkioxari, and Kaiming He. Data distillation: Towards omni-supervised learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4119–4128. IEEE Computer Society, 2018.

[RDG+21] Sanat Ramesh, D. Dall'Alba, C. Gonzalez, Tong Yu, P. Mascagni, D. Mutter, J. Marescaux, P. Fiorini, and 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*, pages 1 – 9, 2021.

[REO03] Robert Frederick Ruben E. Ortega, John W. Avery. Search query autocompletion. `https://patents.google.com/patent/US6564213B1/en`, 2003.

[RFL19] Cristian Rodríguez, Basura Fernando, and Hongdong Li. *Action Anticipation by Predicting Future Dynamic Images: Munich, Germany, September 8-14, 2018, Proceedings, Part III*, pages 89–105. 01 2019.

[RHS05] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. In *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, volume 1, pages 29–36, 2005.

[RHW86] D. Rumelhart, Geoffrey E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[RKG17] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1273–1282, 2017.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks.

In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[RPR+12] Craig Ramsay, Robert Pickard, C Robertson, Andrew Close, Luke Vale, Nigel Armstrong, Daniel Barocas, Cg Eden, Cyntthia Fraser, Tara Gurung, D Jenkinson, Shirley Xueli Jia, Thomas Lam, Graham Mowatt, David Neal, Mc Robinson, J Royle, Sp Rushton, Pawana Sharma, and Naeem Soomro. Systematic review and economic modelling of the relative clinical benefit and cost-effectiveness of laparoscopic surgery and robotic surgery for removal of the prostate in men with localised prostate cancer. *Health technology assessment (Winchester, England)*, 16:1–313, 11 2012.

[SIVA17] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence,*, 2017.

[SJT16] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1163–1171, 2016.

[SMS15] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 843–852. JMLR.org, 2015.

[SPCR20] Wilson Silva, Alexander Pollinger, Jaime S. Cardoso, and M. Reyes. Interpretability-guided content-based medical image retrieval. In *MICCAI*, 2020.

[Spo21] SportsMEDIA. Sportsmedia technology. `https://www.smt.com/technology`, 2021.

[Sul18] Danny Sullivan. How google autocomplete works in search. https://blog.google/products/search/how-google-autocomplete-works-search/, 2018.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014.

[SZ14] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 568–576, 2014.

[SZL+18] Jingkuan Song, Hanwang Zhang, Xiangpeng Li, Lianli Gao, Meng Wang, and Richang Hong. Self-supervised video hashing with hierarchical binary auto-encoder. *IEEE Trans. Image Processing*, pages 3210–3221, 2018.

[SZS12] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.

[TBF+15] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4489–4497. IEEE Computer Society, 2015.

[TMM+16] A. P. Twinanda, D. Mutter, J. Marescaux, M. Mathelin, and N. Padoy. Single- and multi-task architecture for surgical workflow at m2cai 2016. *arXiv: Computer Vision and Pattern Recognition*, 2016.

[TO18] Corentin Tallec and Yann Ollivier. Can recurrent neural networks warp time? In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[TSM+16] Andru Twinanda, Sherif Shehata, Didier Mutter, Jacques Marescaux, Michel De Mathelin, and Nicolas Padoy. Endonet: A deep architecture for recognition tasks on laparoscopic videos. *IEEE Transactions on Medical Imaging*, 36, 02 2016.

[TTW+14] Elizabeth Travis, Ruth Tan, Sarah Woodhouse, Sandeep Patel, Jason Donovan, and Kit Brogan. Operating theatre time, where does it all go? a prospective observational study. *BMJ: British medical journal*, 2014:g7182, 12 2014.

[Twi17] Andru Putra Twinanda. *Vision-based approaches for surgical activity recognition using laparoscopic and RBGD videos*. PhD thesis, 2017. Thèse de doctorat dirigée par De Mathelin, Michel Image et vision Strasbourg 2017.

[TYM+19] Andru Putra Twinanda, Gaurav Yengera, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Rsdnet: Learning to predict remaining surgery duration from laparoscopic videos without manual annotations. *IEEE Trans. Medical Imaging*, 38(4):1069–1078, 2019.

[uns]

[VMMP18] Armine Vardazaryan, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Weakly-supervised learning for tool localization in laparoscopic videos. In *Intravascular Imaging and Computer Assisted Stenting - and - Large-Scale Annotation of Biomedical Data and Expert Label Synthesis - 7th Joint International Workshop, CVII-STENT 2018 and Third International Workshop, LABELS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Proceedings*, volume 11043 of *Lecture Notes in Computer Science*, pages 169–179. Springer, 2018.

[VSF+18] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. Tracking emerges by colorizing videos. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII*, volume 11217 of *Lecture Notes in Computer Science*, pages 402–419. Springer, 2018.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[WDL+19] Dayan Wu, Qi Dai, Jing Liu, Bo Li, and Weiping Wang. Deep incremental hashing network for efficient image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9069–9077. Computer Vision Foundation / IEEE, 2019.

[WGGH18] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7794–7803. IEEE Computer Society, 2018.

[WHG+19] G. Wu, J. Han, Y. Guo, L. Liu, G. Ding, Q. Ni, and L. Shao. Unsupervised deep video hashing via balanced code for large-scale video retrieval. *IEEE Transactions on Image Processing*, pages 1993–2007, 2019.

[WHL+19] Xionghui Wang, Jian-Fang Hu, Jian-Huang Lai, Jianguo Zhang, and Wei-Shi Zheng. Progressive teacher-student learning for early action prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3556–3565, 2019.

[WJZY20] Junwu Weng, Xudong Jiang, Wei-Long Zheng, and Junsong Yuan. Early action recognition with category exclusion using policy-based reinforcement learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(12):4626–4638, 2020.

[WKK+20] A. Watras, Jae-Jun Kim, Jianwei Ke, Hewei Liu, J. Greenberg, C. P. Heise, Yu Hen Hu, and H. Jiang. Large-field-of-view visualization with small blind spots utilizing tilted micro-camera array for laparoscopic surgery. *Micromachines*, 11, 2020.

[WLG+17] Gengshen Wu, Li Liu, Yuchen Guo, Guiguang Ding, Jungong Han, Jialie Shen, and Ling Shao. Unsupervised deep video hashing with balanced rotation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, page 3076–3082, 2017.

[WMS18] Shu-Fen Wung, Daniel C. Malone, and Laura Szalacha. Sensory overload and technology in critical care. *Critical Care Nursing Clinics of North America*, 30(2):179–190, 2018. Human Factors and Technology in the ICU.

[Woj20] Susan Wojcicki. Youtube at 15: My personal journey and the road ahead. https://blog.youtube/news-and-events/youtube-at-15-my-personal-journey/, 2020.

[WTF08] Yair Weiss, Antonio Torralba, and Robert Fergus. Spectral hashing. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 1753–1760. Curran Associates, Inc., 2008.

[WXLG17] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6402–6411. IEEE Computer Society, 2017.

[WXW+16] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII*, volume 9912 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2016.

[WY21] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2021.

[XXY+15] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2691–2699. IEEE Computer Society, 2015.

[XYH15] Zhongwen Xu, Yi Yang, and Alexander G. Hauptmann. A discriminative CNN video representation for event detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1798–1807. IEEE Computer Society, 2015.

[YLD+19] Erkun Yang, Tongliang Liu, Cheng Deng, Wei Liu, and Dacheng Tao. Distillhash: Unsupervised deep hashing by distilling data pairs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2946–2955, 2019.

[YMMP18] Gaurav Yengera, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Less is more: Surgical phase recognition with less annotation through self-supervised pre-training of cnn-lstm networks. *CoRR*, abs/1805.08569, 2018.

[YMMP19] Tong Yu, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Learning from a tiny dataset of manual annotations: a teacher/student approach for surgical phase recognition. *IPCAI*, 2019.

[YP20] Tong Yu and Nicolas Padoy. Encode the unseen: Predictive video hashing for scalable mid-stream retrieval. *ACCV*, 2020.

[ZF14] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer, 2014.

[ZIE16] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, volume 9907 of *Lecture Notes in Computer Science*, pages 649–666. Springer, 2016.

[ZWHC16] Hanwang Zhang, Meng Wang, Richang Hong, and Tat-Seng Chua. Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing. In *Proceedings of the 24th ACM International Conference on Multimedia (MM)*, pages 781–790, 2016.

Tong YU

Recognition and Retrieval Tasks in Large Quasi-Unannotated Surgical Video Databases

Résumé

Les flux vidéos endoscopiques, riches en informations sur le site opératoire, ont un fort potentiel pour alimenter des algorithmes de vision fondés sur l'apprentissage profond. Ces algorithmes peuvent en effet opérer au sein de systèmes de chirurgie assistée par ordinateur, capables d'améliorer la qualité de vie des patients. Cependant, dans les conditions classiques de supervision complète, cette approche nécessiterait de vastes quantités de vidéos annotées. Or les annotations, contrairement aux vidéos elles-mêmes, sont rares, incitant ainsi à des méthodes utilisant des vidéos non-annotées. Nous proposons d'abord une méthode semi-supervisée de reconnaissance de phase, générant des annotations automatiques pour un modèle opérant en temps réel. Nous passons ensuite de la reconnaissance à la fouille de vidéos, avec des méthodes auto-supervisées recherchant en direct du contenu similaire à un flux vidéo au sein d'une grande base de données.

Mots-clés : Apprentissage profond – Vision par ordinateur – Fouille – Hachage – Apprentissage auto-supervisé – Apprentissage semi-supervisé – Endoscopie

Résumé en anglais

Endoscopic video streams, as rich sources of information on the operating field, show great potential for exploitation by deep learning-based computer vision algorithms. Such algorithms can indeed serve as the foundation for context-aware surgery systems, capable of improving clinical outcomes by assisting surgeons during interventions. However this approach would require, under ordinary circumstances of full supervision, vast quantities of annotated recordings. While video data is abundant in endoscopy, annotations are highly scarce, which calls for alternative solutions using unannotated videos. We first propose a semi-supervised surgical phase recognition method, where an offline teacher model automatically labels data for a real-time model. We then move from recognition to video retrieval tasks, with self-supervised methods capable of continuously scanning large video databases for content visually matching a video live stream.

Keywords: Deep learning – Computer vision – Retrieval – Hashing – Self-supervised learning – Semi-supervised learning - Endoscopy