University of Verona

Department of Computer Science
Ph.D. in Computer Science

Automatic extraction of robotic surgery actions from text and kinematic data

Marco Bombieri

Advisor: Prof. Paolo Fiorini
Co-advisor: Prof. Marco Rospocher

INF/01, XXXV cycle, 2023

Ph.D. Candidate:
Marco Bombieri, Università di Verona

Advisor:
Prof. **Paolo Fiorini**, Università di Verona
Co-advisor:
Prof. **Marco Rospocher**, Università di Verona

Thesis reviewers:
Dr. **Chiara Ghidini**, Fondazione Bruno Kessler
Prof. **Myra Spiliopoulou**, Otto-von-Guericke-Universität Magdeburg

Thesis committee:
Prof. **Paolo Fiorini**, Università di Verona
Dr. **Chiara Ghidini**, Fondazione Bruno Kessler
Prof. **Simone Paolo Ponzetto**, Universität Mannheim
Prof. **Marco Rospocher**, Università di Verona

University of Verona
Department of Computer Science
Strada le Grazie 15, Verona, Italy

Ph.D. in Computer Science
Cycle XXXV

To those who supported me

Abstract

The latest generation of robotic systems is becoming increasingly autonomous due to technological advancements and artificial intelligence. The medical field, particularly surgery, is also interested in these technologies because automation would benefit surgeons and patients. While the research community is active in this direction, commercial surgical robots do not currently operate autonomously due to the risks involved in dealing with human patients: it is still considered safer to rely on human surgeons' intelligence for decision-making issues. This means that robots must possess human-like intelligence, including various reasoning capabilities and extensive knowledge, to become more autonomous and credible. As demonstrated by current research in the field, indeed, one of the most critical aspects in developing autonomous systems is the acquisition and management of knowledge. In particular, a surgical robot must base its actions on solid procedural surgical knowledge to operate autonomously, safely, and expertly. This thesis investigates different possibilities for automatically extracting and managing knowledge from text and kinematic data. In the first part, we investigated the possibility of extracting procedural surgical knowledge from real intervention descriptions available in textbooks and academic papers on the robotic-surgical domains, by exploiting Transformer-based pre-trained language models. In particular, we released SurgicBERTa, *a RoBERTa-based pre-trained language model for surgical literature understanding. It has been used to detect procedural sentences in books and extract procedural elements from them. Then, with some use cases, we explored the possibilities of translating written instructions into logical rules usable for robotic planning. Since not all the knowledge required for automatizing a procedure is written in texts, we introduce the concept of surgical commonsense, showing how it relates to different autonomy levels. In the second part of the thesis, we analyzed surgical procedures from a lower granularity level, showing how each surgical gesture is associated with a given combination of kinematic data.*

Sommario

L'ultima generazione di sistemi robotici sta diventando sempre più autonoma grazie ai progressi tecnologici e all'intelligenza artificiale. Anche il settore medico, in particolare quello chirurgico, è interessato a queste tecnologie perché l'automazione si è rivelata vantaggiosa sia per chirurghi che per i pazienti. Sebbene la comunità scientifica sia attiva

in questa direzione, i robot chirurgici commerciali non operano ancora autonomamente a causa dei rischi legati al trattamento di pazienti umani. Si ritiene ancora più sicuro lasciare ai chirurghi umani le varie scelte e decisioni operative. Per diventare più autonomi e credibili, i robot devono dunque possedere un'intelligenza simile a quella umana, ed avere cioè spiccata capacità di ragionamento e di acquisizione di nuova conoscenza. Ricerche recenti dimostrano infatti che uno degli aspetti più critici nello sviluppo di sistemi autonomi è l'acquisizione e la gestione della conoscenza. In particolare, un robot chirurgico deve basare le sue azioni su una solida conoscenza chirurgica procedurale per operare in modo autonomo, sicuro ed esperto. Questa tesi esplora diverse possibilità per estrarre e gestire automaticamente la conoscenza: da dati testuali e da dati cinematici raccolti durante l'intervento. Nella prima parte di questa tesi, abbiamo studiato la possibilità di estrarre la conoscenza chirurgica procedurale dalle descrizioni di interventi già disponibili in libri di testo e articoli accademici nel dominio robotico-chirurgico, sfruttando modelli linguistici pre-addestrati basati sull'architettura neurale Transformer. Abbiamo sviluppato in particolare SURGICBERTA*, un modello linguistico pre-addestrato basato su RoBERTa per la comprensione della terminologia e del linguaggio chirurgico. In particolare, abbiamo usato* SURGICBERTA *per individuare frasi procedurali nei libri ed estrarre elementi procedurali da essi. Poi, con alcuni casi d'uso, abbiamo esplorato le possibilità di tradurre le informazioni estratte in regole logiche utilizzabili per la pianificazione robotica. Poiché non tutte le conoscenze necessarie per automatizzare una procedura sono descritte nei testi, abbiamo introdotto il concetto di commonsense chirurgico, mostrando come esso sia correlato a diversi livelli di autonomia. Nella seconda parte della tesi, abbiamo infine analizzato le procedure chirurgiche a un livello di granularità inferiore, mostrando come ogni gesto chirurgico sia associato a una determinata combinazione di dati cinematici.*

Ringraziamenti

Una tesi di dottorato deve contenere una descrizione lineare e organizzata delle attività svolte durante i tre anni di ricerca. Per sua natura ha come scopo quello di presentare solo ciò che ha funzionato, cioè i metodi che hanno portato a dei risultati scientificamente sensati. Ne va da se che questo modo di raccontare non può essere completo perché volutamente omette tutto il contesto. In particolare, nei capitoli di una dissertazione di dottorato non c'è spazio per enfatizzare che sono le *persone* ad aver permesso ai metodi di essere pensati, ai risultati di essere raggiunti e alle difficoltà di essere superate. In questa sezione vorrei ringraziare tutti coloro che provvidenzialmente ho incontrato e che mi hanno aiutato in questi tre anni.

Il mio primo grande grazie va a Paolo Fiorini che ha creduto in me fin dall'inizio e mi ha dato la libertà, le risorse e le rassicurazioni necessarie per esplorare una linea di ricerca completamente nuova. Lo ringrazio anche perché mi ha fatto conoscere Marco Rospocher che con dedizione, pazienza e costanza mi ha accompagnato in tutte le fasi del mio dottorato dandomi tutto l'aiuto tecnico e sostegno umano di cui avessi bisogno. Ringrazio Marco anche per avermi fatto incontrare Simone Paolo Ponzetto: il suo entusiasmo contagioso, assieme alla sua disponibilità e alle sue competenze tecniche hanno dato al mio dottorato una spinta nuova e hanno permesso che buona parte dei risultati di questa tesi venissero raggiunti. Sono davvero orgoglioso di aver lavorato con loro. Il mio augurio è pertanto che tutti i dottorandi vengano supportati e guidati come Paolo, Marco e Simone hanno fatto con me.

Un grazie particolare va inoltre a Chiara Ghidini e Myra Spiliopoulou per aver revisionato con attenzione questa tesi dandomi ottimi spunti per migliorarla e idee per sviluppi futuri. Ringrazio anche Carlo Combi per avermi sempre dato con disponibilità e attenzione ottimi suggerimenti nelle discussioni di fine anno.

Ringrazio poi tutti i membri del laboratorio Altair e i co-autori con cui ho lavorato in alcune fasi del mio dottorato, in particolare Diego.

Ringrazio quindi i colleghi con cui ho condiviso l'ufficio fin dall'inizio: senza l'amicizia di Filippo, Pietro e Federica le giornate passate seduto sulla scrivania sarebbero state molto più noiose e grigie. Ringrazio anche Sotaro, Tommaso, Chia-Chien, Florian, Pedro e Tornike, i colleghi che mi hanno accolto a Mannheim: mi hanno fatto sentire parte del gruppo fin dal primo giorno condividendo con me le loro idee e il loro tempo libero.

Il tempo del dottorato non lo si vive solo in ufficio, ma anche a casa e tra amici che non sono colleghi. Ringrazio quindi tutti i miei amici che in un momento o l'altro mi hanno aiutato, accompagnato o distratto con un po' di sano tempo libero: sarebbe impossibile citarli tutti ma hanno davvero avuto un ruolo fondamentale. Ringrazio poi la mia famiglia, mio papà Ivo, mia mamma Giuliana e mio fratello Luca per avermi supportato e sopportato nei momenti di tensione e stress. Ringrazio Francesca per esserci sempre stata, per avermi ascoltato, aiutato e per aver camminato sempre al mio fianco con amore e comprensione.

Infine ringrazio Dio per avermi donato la possibilità di fare questo dottorato e per avermi fatto incontrare tutte le persone di cui sopra.

A tutti voi che mi avete supportato io dedico questa tesi.

Contents

List of Tables

List of Figures

List of abbreviations used in this thesis

- ERC: European Research Council
- ARS: Autonomous Robotic Surgery
- AI: Artificial Intelligence
- NLP: Natural Language Processing
- NLU: Natural Language Understanding
- BoW: Bag Of Words
- IAA: Inter-Annotator Agreement
- SRL: Semantic Role Labeling
- ASP: Answer Set Programming
- RQ: Research Question
- NN: Neural Network
- ML: Machine Learning
- DL: Deep Learning
- LSTM: Long-short term memory
- Bi-LSTM: Bidirectional long-short term memory
- 1D-CNN: one-Dimensional Convolutional Neural-Network
- SVM: Support Vector Machine
- CRF: Conditional Random Field
- TF: Term-Frequency
- IDF: Inverse-Document-Frequency
- TF-IDF: Term-Frequency, Inverse-Document-Frequency
- RSPF: Robotic Surgery Procedural Framebank
- RSPB: Robotic Surgery Procedural Propositional Bank
- POS: Part-Of-Speech

- SPKS: Surgical Procedural Knowledge Sentences
- MRR: Mean Reciprocal Rank
- RR: Reciprocal Rank
- MRI: Magnetic Resonance Imaging
- PPMI: Positive Pointwise Mutual Information
- OOV: Out of Domain Vocabulary
- MLM: Master Left Manipulator
- MLML: Master Left Manipulator Left
- MLMR: Master Left Manipulator Right
- PSM: Patient-side Manipulator
- ECM: Endoscopic Camera Manipulator
- NSP: Next Sentence Prediction
- ReLU: Rectified Linear Unit
- AMR: Abstract Meaning Representation
- FLS: Fundamentals of Laparoscopic Surgery
- dVRK: da Vinci Research Kit
- ROI: Region of Interest
- AP: Attachment Point
- LTL: Linear Temporal Logic
- PDDL: Planning Domain Description Language
- MSE: Mean Square Error
- SSIM: Structural Similarity
- CSV: Comma-separated values
- MDI: Mean Decrease in Impurity
- SKF: Stratified k Fold

Part I

Introduction and background

This part presents the objectives and background knowledge required for a complete understanding of the other parts of this thesis. Chapter 1 presents the importance of procedural knowledge acquisition and management for developing autonomous surgical robots. After defining the concept of procedural knowledge and its granularity levels, two ways for its acquisition are presented: top-down approaches extract knowledge from textbooks, while bottom-up ones from kinematic and video data. The advantages and disadvantages of both approaches are discussed, and an overview of the state-of-the-art, which will be deepened in the next parts, is provided. Chapter 2 presents all the background technologies used in the next parts of this thesis.

1

Introduction

"All we have to decide is what to do with the time that is given us."

J.R.R Tolkien, The Lord of the Ring

Robotic systems are currently being used in a wide range of practical applications across multiple fields. Traditionally used in manufactury and assembly lines to perform repetitive actions without suffering from fatigue or in jobs that are too hazardous for humans, robots are now increasingly present in our daily life and in several domains. Among others, the use of robots has revolutionized the medical field, and in particular, the surgical domain as well: firstly adopted in orthopedics for knee [1, 2] and spine surgery [3], in the last few decades, robots have been increasingly adopted in laparoscopic surgery, in particular urology, gynecology, and general surgery [4]. Furthermore, thanks to the advancements in technology and artificial intelligence, the latest generation of robotic systems will become increasingly autonomous, thanks to higher decision-making skills. In accordance to these trends, also the robotic surgery community is dealing with automation aspects [5, 6].

Unlikely other fields where autonomous robots may be seen as a threat to workers, the majority of the surgical community recognizes the benefit of bringing autonomy in robotic surgery [7] for several reasons. First, surgeons often are overworked to high levels of fatigue that can cause hand tremors and attention reduction. In these situations, they may be less capable of performing precision tasks and, therefore, more prone to make errors. Autonomous robots are unaffected by these issues. Furthermore, especially in the hospitals of the more isolated cities, it is not always possible to recruit expert surgeons; having an autonomous robot (maybe remotely controlled by an expert) capable of operating with quality comparable to that of an experienced surgeon

can help to reduce the discrepancies and inequalities between operations performed in different geographical places. Moreover, an autonomous surgical robot may react faster than the surgeon to unexpected events, and autonomy may compensate for the time delay in remote telesurgery. This, combined with the greater dexterity of robotic systems facilitated by their wristed instruments, will further improve minimally-invasive procedures [8, 9, 10]. Finally, another compelling benefit is that surgeons will no longer need to be in the same room of the patient thus avoiding stray radiation from X-ray fluoroscopy devices [11, 12].

Because of the growing interest and benefits of bringing autonomy to robotic surgery, the scientific literature is discussing how the levels of autonomy can be defined. Following the taxonomy first presented for self-driving cars [13], an autonomous robotic surgical system can be classified into five levels of autonomy [14]: at autonomy level 0, the human performs all tasks and takes all decisions; at autonomy level 1 the robot provides dexterity and cognitive assistance during the task, sharing controls and actions with the human; at level 2, the robot is autonomous during specific tasks, i.e., trading control of the system with human at discrete times; at level 3 the robot generates task strategies, but the human has the final decisions over the proposed tasks; at level 4 the robot can make decisions on the complete surgical strategy, but under the supervision of a qualified doctor; finally, level 5 introduces the full autonomy, i.e., a robotic surgeon that can perform an entire procedure without supervision.

Nevertheless, at the moment, commercial robots only provide an autonomy level of 0 and do not perform any action in full autonomy, because of technological and legal reasons. This Ph.D. research is developed within the Autonomous Robotic Surgery (ARS) project[1], which aims at developing methodologies to enable the execution of surgical intervention by a robotic system in complete autonomy. ARS' research proved that to reach some level of autonomy, a robotic surgical system has to face different technological challenges. First, the anatomical environment in surgical procedures is composed of soft tissues that can deform due to the use of surgical instruments or physiological effects such as breathing or heartbeats. Additionally, tissue behavior is complex to model and can vary greatly among different patients, making it difficult to measure. Furthermore, the definition of a patient-specific intervention plan is challenging because it requires integrating notional knowledge (e.g. that contained in textbooks or pre-operative images) with the surgeons' way of reasoning and experience. The latter

[1] The ARS project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under grant agreement No. 742671.

can only be partially extracted from surgery video and kinematic data recorded during real interventions. Moreover, to address the uncertainties of the anatomical environment that can arise during surgery, it is important for a surgical robot to be able to adapt the patient-specific intervention plan during execution based on the current situation. This is because the anatomical environment may behave differently than what is expected from pre-operative knowledge. To accomplish this, incorporating strategies for real-time situation awareness, reasoning, and control into the robot is needed. Finally, since the operational environment is the human body, errors can be deadly. Consequently, to reach all the above requirements, such a robot has to be endowed with human-like intelligence that combines different reasoning capabilities with strong notional knowledge. The state of the art is indeed demonstrating that the real core of the research on autonomous systems is in knowledge and information acquisition and management [15]: to operate autonomously, safely, and expertly, a surgical robot must base its actions on solid *surgical knowledge*. Nowadays, in surgical robotics, the knowledge is manually encoded by domain experts in ontologies [16] or a pre-defined set of logical instructions [17]. The manual encoding of the prior domain knowledge in a logic formulation understandable by machines is a limitation and bottleneck in developing autonomous systems because it requires experts in surgery and computer science, who may not have the right competencies and are not used to work together. Furthermore, the manually encoded knowledge is static and it may not cover all complications and cases during surgery; thus, a way to automatically acquire knowledge from external resources is preferable. Since one of the main challenges an autonomous robotic surgical system has to face is the *automatic acquisition and management of surgical knowledge*, this thesis investigates different possibilities for automatically extracting surgical knowledge from existing resources — primarily free-text books, academic papers, and written tutorials, but also from kinematic data — to lay the foundations for tomorrow's knowledge-based surgical robot.

1.1 Surgical knowledge and its learning

There are two different types of surgical knowledge, both required for the autonomous execution of surgical intervention [18]: *procedural* and *non-procedural* knowledge. The *procedural knowledge* encodes instructions needed to perform the specific surgical intervention, being interventions on the body or the positioning of the robot. In general,

Table 1.1: Examples of procedural and non-procedural sentences in books. In the table, "P" means *Procedural*, while "!P" is for *Non-procedural*. These examples are taken from the dataset presented in 4.

Type	Sentence	Explanation
P	*The peritoneum is then incised.*	*Incision* of the peritoneum.
P	*Using a combination of blunt dissection and electrocautery, the posterior aspect of the pylorus and the proximal duodenum are gently elevated off of the retroperitoneum.*	*Elevation* of the retroperitoneum.
P	*Allis clamps are used to tension the ileal segment against the catheter along its antimesenteric edge.*	*Tension* of the ileal segment.
!P	*As a distinguishing feature, Gerota's fascia appears pale yellow, compared with the brighter yellow color of the mesentery.*	Descriptions of an anatomical feature.
!P	*Numerous descriptions of nerve sparing during RARP have been reported in the literature.*	Additional in-depth information.
!P	*Longer operative times were seen with robotic procedures.*	Information not directly useful to perform the procedure.

a procedure is an ordered sequence of actions linked together temporally and causally. An action may be activated when a certain pre-condition is satisfied and reaches its end state when a certain post-condition occurs. Usually, an action can be executed if accompanied by a set of semantic information, such as the *"agent"*, i.e., the one who acts; the *"patient"*, i.e., the one who undergoes the action; the *"instrument"*, which refers to the tool used for acting and the *"purpose"* describing the reason why the action is performed. In addition, other semantic information comprises *temporal* and *spatial* parameters. The *non-procedural knowledge* encodes instead anatomical knowledge and other ontological information. It does not include any indication of a specific surgeon's action. However, it describes anatomical aspects, exceptional events that can occur during surgery, and general indications that are not specific to a single intervention step. To clarify, Table 1.1 shows examples of procedural and non-procedural sentences taken from the dataset presented in Chapter 4 with the corresponding explanation of their content.

Fig. 1.1: Granularity axis. *Low-level information* is relative to video, image, and kinematic data.

This thesis only deals with *procedural knowledge* acquisition and management.

Procedural knowledge can be expressed according to different granularity levels presented in [19] and summarized in Figure 1.1. Each level represents the details provided during the description. In this classification, a procedure (e.g. partial nephrectomy) is composed of a sequence of main events, called phases, occurring in the procedure (e.g. tumor excision or final suture). Each phase is then composed of a set of steps, i.e. sequences of activities to achieve a surgical objective (e.g. the main steps of the final suture phase are the removal of the trocar, the extraction of the specimen, and the closure of the skin). Each activity is then composed of a sequence of motions, i.e. surgical movements involving only one hand trajectory (e.g. pulling the needle to close the suture using the right arm). Finally, low-level information is the raw data, i.e. kinematic and video captured during the surgery at a given frequency.

Depending on the granularity level at which information is to be extracted, two different approaches can be followed:

- *Top-down approach*: construct the execution flow of a surgical procedure by exploiting the notional knowledge available in ontologies or books. Starting from these resources, the goal is to develop a plan that a robot can execute.
- *Bottom-up approach*: starting from the data captured during the execution of a surgery (kinematics and video), develop methods to infer the surgical process. In this research domain, an important task is to define features useful for surgical gesture recognition: machine-learning algorithms use them to segment the intervention into phases and steps, deriving the surgical procedure.

The development of an autonomous robotic surgical system will require the integration of both the notional knowledge extracted from textbooks for understanding high-level instructions and the low-granularity actions, which can only be learned from data of actual interventions.

This thesis mainly deals with top-down approaches, particularly with the still unexplored possibility of extracting procedural surgical knowledge directly from written resources, such as textbooks, academic papers, and surgical guidelines. The bottom-up approaches are instead widely discussed in the literature, as shown later in this thesis. Anyway, the final part of the thesis is dedicated to it, where it is shown that suitable features from kinematic data captured during the execution of a task can help gesture understanding by machine learning techniques. In future work, we will explore the possibility of combining models extracted with top-down approaches with those obtained with bottom-up ones in a unique knowledge-based model because this is what a hu-

man surgeon does: the course of study for becoming a surgeon consists of the first part of theoretical study, in which the surgeon acquires the fundamental theoretical notions of the profession, and a final part of practice, in which the student, through a cycle of internships, integrates the theoretical knowledge learned with experience and observation of seniors.

1.2 Procedural knowledge extraction from text

Theoretical study occupies a predominant and substantial part of the study cycle of an apprentice surgeon. This is the reason why the literature is teeming with manuals, online resources, and academic papers of the highest quality used by universities around the world. Each book is written by expert surgeons and contains sections describing the pre- and post-procedure diagnoses, the procedure's name, a detailed description of the procedure, and other information. These texts are meant and written for the understanding of human readers and present the information in unstructured natural language. Having algorithms capable of understanding the surgical procedures written in natural language and capable of organizing the procedure content in a more structured and processable form would pave the way for developing intelligent surgical and clinical systems. If automatically processed by Natural Language Processing (NLP) techniques, this high-quality procedural information becomes valuable content that could be exploited in many clinical applications. For example, robots could automatically build or extend a proper surgical knowledge base, reasoning with it in realistic intervention scenarios. Humans could benefit from more structured knowledge in question-answering sessions, for example, in an early learning phase by medical students. However, so far, the extraction of procedural surgical knowledge directly from written resources such as textbooks, academic papers, or case reports has received little attention from the scientific community, as current trends mainly focus on the derivation of knowledge from kinematic and video data captured by endoscopic sensors and cameras during interventions [17, 20], or on the manual modeling of ontologies, e.g. [21].

Although not in the surgical domain nor with the purpose of automatizing surgical interventions, some works, e.g. [22, 23, 24, 25, 26, 27, 28] have explored the task of procedural knowledge extraction from text. These papers, which will be detailed in the other chapters of this thesis, propose approaches for extracting procedural knowledge

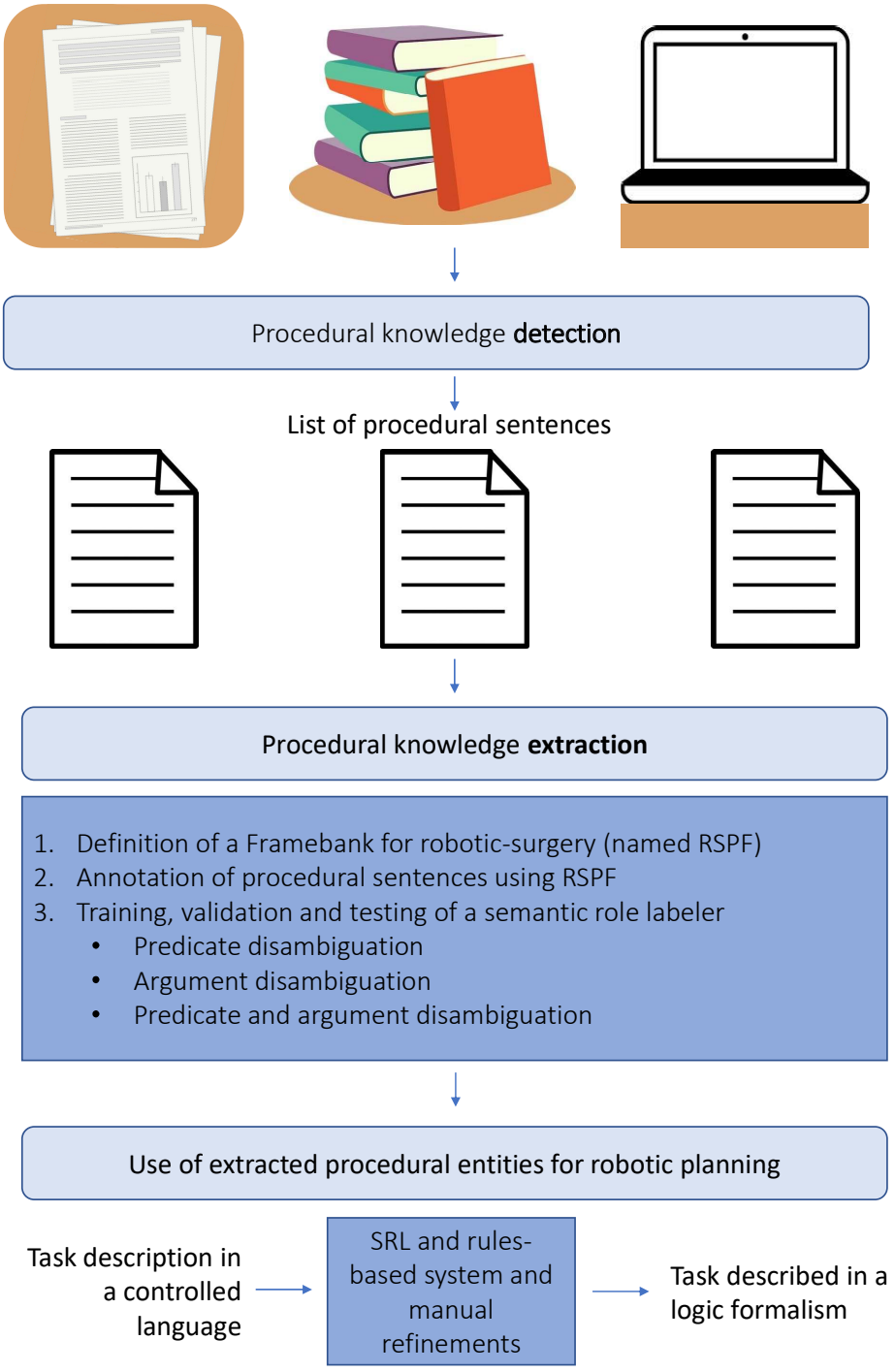


Fig. 1.2: Summary of the Part 1 of this thesis (Chapters 3-7). Part 2 is instead composed by the single Chapter 9.

in several domains, such as technical documentation, cooking recipes, maintenance manuals, repair guidelines, and instructions for the synthesis of nanomaterials. While all these works address the extraction of procedural knowledge from written text and are thus similar to our foreseen application, they deal with typologies of textual content substantially different from the description of a surgical procedure. They are different both from the terminological point of view as well as the structural one since these texts are structurally organized, frequently using numbered/bulleted lists. No established standard way to describe a surgical procedure instead exists. In addition, surgical interventions are mainly presented in a prose-like style. Furthermore, recently, researchers are starting to use natural language to generate or control a set of actionable instructions; some examples are [29, 30, 31]. The proposed tasks use, however, strong simplification of natural language adopted, and the main purpose is that of translating concepts in an actionable form rather than the understanding of complex procedural descriptions. Furthermore, they are not thought for the surgical domain.

While understanding very specialized literature written in free-text, i.e., without recurring to a controlled language [32], would be challenging for the traditional NLP methods, the advent of the transformer neural network architecture with the attention mechanism [33], and the pre-trained language models [34] have made this task feasible. Pre-trained language models have demonstrated remarkable performance in various downstream tasks, including machine translation, sentiment analysis, and text classification, outperforming traditional machine learning algorithms and rule-based methods thanks to their ability to learn complex linguistic patterns and contextual relationships from vast amounts of unlabeled text. However, these models are trained on general English data and may not perform as well in highly specialized domains such as scientific literature, law, or medicine. In such cases, domain adaptation techniques, such as fine-tuning or transfer learning, can be used to retrain the pre-trained models on domain-specific data, allowing them to capture the unique language patterns and terminologies of the specialized domain. Although domain adaptation techniques are available, they require a lot of time-consuming activities to find relevant information in surgery (i.e., defining a proper surgical framebank[2]) and to annotate the domain-specific texts that will be used as training material. Both the definition of a framebank and the annotation of surgical text are complex tasks as they demand the expertise of both surgical and linguistic professionals.

[2] The concept of framebank is defined in Section 2.6

This thesis fills these literature gaps by proposing different scientific contributions summarized in Figure 1.2. First, deep learning methods have been exploited to extract from books sentences containing procedural knowledge discarding those containing non-procedural ones. Then, to develop a model capable of understanding surgical language, this thesis defines a proper surgical framebank, adapting an existing general-English one to the robotic-surgery domain. The obtained surgical framebank is then used to annotate a corpus of as-is surgical sentences taken from surgical books and academic papers. The annotation step has been carried out by exploiting the Semantic Role Labeling (SRL) style using a semi-automatic technique based on post-editing and manual corrections. The annotated corpus obtained is then used to train, validate and test a deep learning, Transformer-based SRL model proving significant improvement in the surgical natural language understanding task compared with its vanilla model, i.e. the general English model not specialized for the surgical domain. In addition to the aforementioned supervised training, unsupervised learning was utilized on a substantial amount of raw text, resulting in the development of a new SRL model that exhibits enhanced comprehension of surgical literature. The language model obtained from the non-supervised learning step was then used to solve other NLP tasks, such as surgical terminology learning and ontological information inference. Finally, a pipeline based on SRL and some syntactic rules has been adopted to demonstrate how, within simple language constraints, it is possible to extract a logical template from sentences written in natural text. This logical template can then be easily translated to a logic planning formalism, such as Answer Set Programming (ASP)[35] without the need for significant manual revisions. As a result, the task of logicians is simplified because they no longer need to be surgical experts.

1.3 Procedural knowledge extraction from kinematics

In the bottom-up direction, the goal is to use low-level input information (mostly kinematics, video data or both together) acquired by sensors to recognize higher-level semantic knowledge, such as a list of surgical motions executed by the surgeon [19]. These motions implicitly contain expert human surgical knowledge because kinematic and video data is captured during the execution of interventions performed by experts and, consequently, can be used as the gold standard for teaching low-level robot movements. They allow an understanding of a surgical procedure with a lower level of granularity

than that described in textbooks, such as motions or actions. To extract this low-level knowledge, multi-modal deep-learning techniques have been exploited and applied to kinematic, video, or other data, such as system events [20, 36, 37, 38]. The most used methods are based on convolutional neural networks, LSTM, and other classic machine learning algorithms such as support vector machine and random forest, whose theory is discussed in 2.5.2. The most common dataset used to train and validate algorithms are JIGSAWS [39] or others ad-hoc developed [20, 36, 38, 40]. A crucial aspect of these algorithms is to find accurate features capable of describing each surgical motion, step, or phase [41, 42, 43]: this aspect will be analyzed in Chapter 9.

The bottom-up approaches have to face some challenges and practical issues. First, the literature is lacking freely available and realistic datasets, which are difficult to obtain due to patient privacy or commercial issues. Then, it is important to find significant features to use as input to the learning algorithms; in order for the features to be calculable, the datasets have to contain the relative information, and therefore the right choice of sensors must be made at the recording stage; however, some of this information (e.g. force data) is not always immediate to estimate from the available robotic tools or may be noisy. Finally, different approaches can lead to different models, and it is still unclear how to evaluate the differences.

This thesis proposes literature's improvements in features engineering, showing that adopting features based on joint robot orientations improves the understanding of the motions. Since no datasets containing information about robot joints were available in the literature, one ad-hoc was released.

1.4 Outline of the thesis

Chapter 2 presents all the background technologies used in the next parts of this thesis. Then, the thesis is split into two parts. The first one (Chapters 3-8) deals with procedural knowledge detection and extraction from robotic-surgery textbooks. The second one (Chapter 9) deals instead with procedural knowledge extraction from kinematic data. Finally, Chapter 10 summarizes the contributions of this thesis and proposes several possible future research directions. The content of each chapter is summarized in the introduction paragraph at the beginning of each part.

1.5 Contributions

The main contributions of this thesis are:

[C.01] The release of SPKS annotated textual resource for procedural surgical sentences detection (Chapter 4);

[C.02] The development of machine learning methods for procedural surgical sentences detection (Chapter 4);

[C.03] The development of SURGICBERTA, a pre-trained language model specific for surgical language (Chapter 3);

[C.04] The release of RSPF, a framebank specific for the robotic-surgery procedural language (Chapter 5);

[C.05] The annotation of a dataset of as-is textbooks sentences with the RSPF labels (Chapter 5);

[C.06] The development of deep learning methods to extract procedural surgical knowledge from the text (Chapter 6);

[C.07] The proposal of a pipeline for mapping natural language surgical procedures to a logic formalism and simulation (Chapter 7);

[C.08] The proposal of a taxonomy of different levels of surgical commonsense knowledge and links with the levels of autonomy (Chapter 8);

[C.09] Development of an annotated dataset for surgical gestures recognition containing joints-space orientation information (Chapter 9);

[C.10] Proposal of joints-space metrics for surgical gestures recognition (Chapter 9).

1.6 Publications

The main publications resulting from the thesis, with reference to the presented contributions, are:

- **Marco Bombieri**, Marco Rospocher, Simone Paolo Ponzetto and Paolo Fiorini. *The robotic-surgery propositional bank*. Language Resource and Evaluation. June 2023. **[C.05]** [44]
- Eleonora Tagliabue, **Marco Bombieri**, Paolo Fiorini and Diego Dall'Alba: *Robotic surgical systems need commonsense to achieve higher levels of autonomy*. Robotics and Automation Magazine (IEEE). May 2023. **[C.08]** [45]

- **Marco Bombieri**, Marco Rospocher, Simone Paolo Ponzetto and Paolo Fiorini. *Machine understanding surgical actions from intervention procedure textbooks*. Computers in Biology and Medicine. January 2023. **[C.06]** [46]
- Daniele Meli, **Marco Bombieri**, Diego Dall'Alba and Paolo Fiorini. *Inductive learning of surgical task knowledge from intra-operative expert feedback*. 9th Italian Workshop on Artificial Intelligence and Robotics (AIRO). December 2022. **[C.07]** [47]
- **Marco Bombieri**, Marco Rospocher, Simone Paolo Ponzetto and Paolo Fiorini. *The robotic surgery procedural framebank*. Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC). June 2022. **[C.04]** [48]
- **Marco Bombieri**, Marco Rospocher, Diego Dall'Alba and Paolo Fiorini. *Automatic detection of procedural knowledge in robotic-assisted surgical texts*. International Journal of Computer Assisted Radiology and Surgery. April 2021. **[C.01,C.02]** [18]
- **Marco Bombieri**, Diego Dall'Alba, Sanat Ramesh, Giovanni Menegozzo and Paolo Fiorini. *Joint-space metrics for automatic robotic surgical gestures classification*. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). October 2020. **[C.09,C.10]** [40]
- **Marco Bombieri**, Marco Rospocher, Simone Paolo Ponzetto, and Paolo Fiorini. *SurgicBERTa: A pre-trained language model for procedural surgical language*. Under revision in a journal. Submitted in March 2023. **[C.03]** [49]
- **Marco Bombieri**, Daniele Meli, Diego Dall'Alba, Marco Rospocher and Paolo Fiorini. *Mapping natural language procedures descriptions to linear temporal logic templates - An application in the robotic-surgery domain*. Under revision in a journal. Submitted in November 2022. **[C.07]** [50]

Publication contributed during the Ph.D. but not strictly related to the main topic of the thesis:

- Chia-Chien Hung, Tommaso Green, Robert Litschko, Tornike Tsereteli, Sotaro Takeshita, **Marco Bombieri**, Goran Glavas and Simone Paolo Ponzetto: *Data Augmentation with Specialized Models for Cross-lingual Open-retrieval Question Answering System*. Proceedings of the Workshop on Multilingual Information Access (MIA). July 2022. [51]

1.7 Released resources and models

- SPKS – annotated dataset for detecting procedural robotic-surgery sentences: `https://gitlab.com/altairLab/spks-dataset`
- RSPB – annotated dataset for procedural surgical SRL: `https://gitlab.com/altairLab/robotic-surgery-propositional-bank`
- SURGICBERTA– the language model for surgical language understanding: `https://gitlab.com/altairLab/surgicberta`
- SURGICBERTA $_{SRL}$ – SURGICBERTA fine-tuned for SRL: `https://gitlab.com/altairLab/surgical_srl`
- Dataset for surgical gestures recognition: `https://gitlab.com/altairLab/yeast-dataset`

2

Background

"If I have seen further, it is by standing on the shoulders of giants."

Isaac Newton

This thesis investigates the application of machine and deep learning techniques to text or kinematic data for surgical procedural knowledge extraction. This chapter aims at giving the thesis background by introducing all the technologies used in the research. In the first part, we briefly introduce the concept of machine and deep learning, focusing on the two main paradigms exploited in this thesis, i.e., supervised and unsupervised learning. The first requires the presence of annotated data to train the models, while the second requires the availability of a great amount of unlabeled data. Since no datasets were already available for the procedural robotic-surgery domain, we developed ad-hoc datasets by using semi-automatic techniques: therefore, this chapter also describes the techniques for data annotation and the quality metrics used to evaluate the results. Machine and deep learning techniques described in this chapter can, in our case, be applied both to textual and kinematic data. Then, the main part of the thesis deals with NLP techniques applied to texts of the surgical domain: the NLP state-of-the-art methods are nowadays mostly based on pre-trained large language models, and also the contributions of this thesis follow this trend. This chapter then introduces the language modeling techniques by comparing the recent pre-trained Transformer-based language models with the most traditional ones. This part will be propaedeutic for the chapter aimed at defining SURGICBERTA, the Transformer-based pre-trained language model specific for the procedural surgical language we developed. It is also needed for the understanding of the other chapters aimed at using SURGICBERTA and the other state-of-the-art models for procedural sentence detection and procedural knowledge

extraction. Then, since procedural sentence detection is tackled as a text classification task, this chapter defines this task by presenting some background knowledge used in the corresponding chapter. Finally, since the procedural knowledge extraction method is mostly based on Semantic Role Labeling (SRL), this chapter defines this task, the related language resources, and the methodological solutions.

2.1 An overview of machine learning

Machine learning is a subfield of Artificial Intelligence (AI) that focuses on developing algorithms and models that can automatically improve their performance on a specific task through experience. The main goal of machine learning is to enable computers to learn patterns and make predictions based on data without being explicitly programmed to do it [52]. As a subfield of machine learning, deep learning is based on artificial neural networks [53]. Deep learning algorithms use multiple layers of artificial neurons to process and transform information, allowing them to automatically extract high-level features from raw data and make predictions. Deep learning algorithms are particularly well-suited for tasks that involve large amounts of complex data, such as medical images or free text.

Applications in which the training data comprises examples of the input vectors and their corresponding target vectors are known as *supervised learning* problems. This is the most commonly used type of machine learning, where the algorithm is trained on a labeled dataset, and the goal is to learn a mapping from inputs to outputs based on this data. A typical example is that of sentiment analysis, where free-text reviews of a product are manually annotated with the *positive*, *neutral*, and *negative* labels. These labels correspond to the target vector and are used as training material for the model. After training, the obtained model can be used to recognize the customer's sentiment in reviews never seen before (i.e., on the test material).

On the other hand, *unsupervised learning* is used when the dataset is unlabeled, and the goal is to find patterns or relationships within the data. In more detail, the training data consists of a set of input vectors without any corresponding target values. The goal of such unsupervised learning problems may be to discover groups of similar examples within the data, which is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for visualization. In the context of

NLP, the techniques used to develop word embeddings (described later in this chapter) are examples of unsupervised learning. In these applications, the goal is to learn numerical representations of words by training a model on a large corpus of unlabeled text data by trying to predict the context of a word based on the surrounding words in a sentence; no manual annotated text is needed.

Alongside supervised and unsupervised learning, *reinforcement learning* is the third basic machine learning paradigm. Reinforcement learning is not used in this thesis but is mentioned just for completeness. This paradigm is concerned with how intelligent agents ought to take actions in an environment to maximize a "reward" through trial and error: no labeled data is required. A classic example is that of Tesauro et al. [54], where a neural network was used to learn to play backgammon to a high standard. In such an example, the network must learn to take a board position as input, along with the result of a dice throw, and produce a strong move as output. It is necessary to properly attribute the reward to all the moves that contributed to achieving victory, regardless of whether some of them were good and others were not as good.

The next subsections will present the main supervised and unsupervised learning issues of interest for this thesis.

2.2 Manual data annotation for supervised learning

Manual annotation is the labeling of data by human effort, then used for training supervised machine learning models. Gathering and annotating data are critical steps in developing supervised machine learning models. A list of best practices must be followed during these steps to obtain from training a robust and representative model that will provide high performance and generalization capabilities during testing to unseen data. These best practices are described below and can be applied to all types of data, such as video [55], or text [56].

2.2.1 Quality of the source data.

Collecting relevant and high-quality data is needed to ensure the highest performance of the supervised learning model. The data should be relevant to the problem to be solved and should accurately represent the real-world scenario. For some applications, this includes diversity in terms of demographic and cultural factors, as well as in terms of the distribution of the target variable. In particular, the data must be accurately

chosen from realistic sources, avoiding excessive simplifications that could make the trained model then ineffective in the real world. In the sentiment analysis scenario, for example, the data should be taken from realistic reviews and not generated in a contrived way.

Furthermore, data gathering should be done following ethical considerations, including, in some domains, data privacy and security and ensuring that the data was collected with the informed consent of the individuals involved: these considerations are particularly important when dealing with sensitive data, such as personal information or medical records. In other cases, the data should be subjected to copyright and then not freely usable and shareable, thus requiring agreements with the data owner. Finally, to avoid biased models that perform poorly on underrepresented classes, it would be preferable if the data were balanced, meaning that it equally represents all the different classes of the target variable. For example, if the model is trained on the sentiment analysis classification task, the data should include roughly equal numbers of positive, neutral, and negative examples. However, in some domains or applications is not always possible to have balanced data due to the scarcity of available resources: in this case, balancing techniques can be used [57].

2.2.2 Size of the source data.

Understanding how much data is needed to train a supervised learning model is another paramount aspect. Generally, more high-quality annotated data can help the model learn more complex patterns and generalize better to unseen data, but not always enough data is available, and usually, the annotation process is costly and, in some cases, such as in the medical domain, requires high-specialized personnel difficult to find. Furthermore, especially for domains, types of data, or tasks still unexplored, it is impossible to know in advance the amount of data required. A possible way is that of training the model on an increasing amount of data and observing the performance trend on the same test dataset. If the performance increases by adding more training material, it makes sense to annotate other data. Nonetheless, often a trade-off between costs and benefits has to be found.

2.2.3 Quality of the annotations.

When annotating data, it is important to ensure that the annotations are accurate, consistent, and of high quality. The annotators should be knowledgeable about the prob-

lem and trained to ensure consistent and accurate annotations. The annotations should also be checked for quality and consistency before being used for training.

The choice of annotators.

Having annotators who are knowledgeable about the domain and task is crucial for ensuring high-quality annotations. In particular, annotators who have a deep understanding of the domain are more likely to make accurate annotations. While general tasks such as sentiment analysis applied to commonly used product reviews by customers who have bought and tried them do not require specific background knowledge and can be efficiently and effectively carried out by crowdsourcing, other domains (e.g. medicine, engineering, or linguistics) and tasks (e.g. labeling of tumors in MRI images [58], annotation of data for natural disasters detection [59] or semantic role labeling [56]) require expert knowledge both for understanding the domain and the application task. In these cases, it is important to recruit a team of domain expert annotators.

Furthermore, having more annotators is generally considered better because it improves accuracy by mitigating the potential for bias or errors that a single annotator may introduce. This makes possible the quantification of the intra-annotator agreement, i.e., the agreement between the annotations made by a single annotator, as compared to the annotations made by other annotators for the same instances. Anyway, in some NLP applications, such as those related to abusive and offensive language, the utility of resorting to a single agreement between the annotators is debated: the same data can be labeled in one way by one annotator and oppositely by another annotator, depending on their opinions and cultural and demographic background. In some cases, both opinions may be considered correct. Consequently, both annotations should be considered true in the gold standard to avoid destroying any personal opinion, nuance, and rich linguistic knowledge by the agreement and harmonization processes: this concept is known as data perspectivism [60].

Annotation guidelines.

Once a team of annotators is recruited, it is important to define the annotation guidelines. Annotation guidelines are instructions or rules that define how data instances should be annotated for a specific task. The purpose is to reduce possible errors related to misinterpretation of the task. In particular, annotation guidelines should:

- clearly define the task, including the objectives and the scope of the annotation;

- provide clear and detailed instructions on how to annotate each data instance, including the definition of the labels and the labeling process;
- provide examples of annotated data instances for each label, helping the annotators to understand the annotation process;
- provide the criteria for quality control, such as inter-annotator agreement, and the procedures for ensuring the quality of the annotations;
- specify how annotators will receive feedback on their annotations and how they can ask for any doubts that may emerge during the annotation phase.

Annotation tools.

Using an annotation tool can be useful in data annotation for improving the quality of the annotations. The set of labels usable for the annotations can be encoded in the tool, thus helping to reduce noise in the annotations and improving the efficiency of the annotation process. Furthermore, annotation tools often provide built-in quality control features, such as the possibility to calculate intra-annotator agreement metrics described in the next paragraph. Commonly used tools for textual data annotation are, for example, Inception [61], and BRAT [62], while for video data are CVAT[1] and Vott[2].

Evaluating manual annotations.

To measure the quality of manual annotations and the agreement between annotators, the Inter-Annotator Agreement (IAA) has to be calculated. IAA provides an idea about how clear the annotation guidelines are, how uniformly the annotators interpret them, and how reproducible the annotation task is. It is thus a crucial step for both the validation and reproducibility of classification results. The most used metric for IAA when two annotators are involved is Cohen's Kappa [63], then generalized by Fleiss' kappa in a multi-annotators scenario [64].

Cohen's kappa (C_k) is a measure of the degree of agreement between two annotators beyond chance, considering the agreement that would be expected by chance alone. It is defined as follows:

$$C_k = \frac{p_o - p_c}{1 - p_c} \tag{2.1}$$

[1] Available at: `https://www.intel.com/content/www/us/en/developer/articles/technical/computer-vision-annotation-tool-a-universal-approach-to-data-annotation.html`

[2] Available at: `https://github.com/microsoft/VoTT`

In the formula, p_o is the proportion of units in which the annotators agreed, calculated as the number of agreed annotations divided by the total number of annotations. It represents the actual agreement rate between the annotators and reflects the extent to which they are consistent in their annotations; p_c is instead the proportion of units for which agreement is expected by chance. It is calculated by multiplying the marginal frequencies of each annotator, i.e., the proportions of annotations made by each annotator, and taking the sum over all categories.

The $p_o - p_c$ represents the proportion of the cases in which beyond-chance agreement occurred and is the numerator of the coefficient. The coefficient C_k is simply the proportion of chance-expected disagreements which do not occur, or it is the proportion of agreement after the chance agreement is removed from consideration. The C_k upper limit is +1.00, and its lower limit falls between zero and −1.00, depending on the distribution of judgments by the two annotators.

Fleiss' Kappa is a measure of IAA in data annotation tasks where multiple annotators label the same instances. Unlike Cohen's Kappa, which is calculated between two annotators, Fleiss' Kappa is used to measure IAA between more than two annotators. The formula for Fleiss' Kappa is as follows:

$$F_k = \frac{\overline{p_o} - p_c}{1 - p_c} \tag{2.2}$$

where $\overline{p_o}$ is the average agreement rate between the annotators, calculated as the sum of the agreement rates for each instance, divided by the total number of instances, and p_c is the expected agreement rate between the annotators, calculated as the sum of the products of the marginal frequencies of each annotator for each category, divided by the total number of annotations.

Both the C_k and F_k values (*kappa*) can be interpreted as follows [65]:

- $kappa < 0$: Less than change agreement
- $0.01 < kappa < 0.20$: Slight agreement
- $0.21 < kappa < 0.40$: Fair agreement
- $0.41 < kappa < 0.60$: Moderate agreement
- $0.61 < kappa < 0.80$: Substantial agreement
- $0.81 < kappa < 0.99$: Almost perfect agreement
- $kappa = 1.00$: Perfect agreement

While these metrics are widely adopted in state-of-the-art studies, their limits are also debated. The interested reader can find more information, for example, in [66].

2.3 Converting text to a numerical representation

For NLP machine learning approaches, text must be converted into a numerical representation. Different approaches have been proposed for this purpose and are summarized in the next sections.

2.3.1 Sparse vectors representation: the binary model and TF-IDF

The Bag Of Words (BoW) representation creates a vocabulary of all the unique words in the corpus and then represents each document as a vector of the frequency of each word in the vocabulary. In this approach, the histogram of the words within the text is checked, and each word count is considered a feature. Formally, given a collection of $|D|$ documents $D = \{d_1, d_2, \ldots, d_{|D|}\}$, each document d_j is represented as a vector x_j in a vocabulary V of size $|V|$, where $|V|$ is the number of unique words. The vocabulary can be obtained by taking the union of all words in the documents, and a word frequency matrix X is constructed, where x_{ij} is the frequency of word i in document j. Formally, the BoW representation of the $j-th$ document can be written as:

$$x_j = [x_{1j}, x_{2j}, \ldots, x_{|V|j}] \tag{2.3}$$

There are two main variants of the BoW model: the *binary model* and the *TF-IDF*. The first one represents each document as a binary vector, where each element of the vector indicates whether a word from the vocabulary is present or not in the document. The binary BoW representation is created by setting the value of each element in the vector to 1 if the corresponding word is present in the document and 0 otherwise. Consequently, in 2.3, $x_{ij} = 1$ if word i is present in document j, 0 otherwise.

The *Term Frequency-Inverse Document Frequency* (TF-IDF) is instead a weighting scheme that is often used in information retrieval and text mining to reflect the importance of a word in a document concerning an entire corpus of documents. TF-IDF extends the binary model by assigning a weight to each word in a document based on its term frequency (TF) and inverse document frequency (IDF). The term frequency

measures the number of times a word occurs in a document, while the inverse document frequency down-weights the importance of commonly occurring words and up-weights the importance of rare words. The TF of a word i in the document j is defined as:

$$TF_{i,j} = \frac{f_{i,j}}{n_j}$$

Where $f_{i,j}$ is the number of occurrences of the word i in the document j and n_j is the number of words in document j.

The IDF of a word i is instead defined as:

$$IDF_i = log\left(\frac{|D|}{n_i}\right)$$

where $|D|$ is the total number of documents in the corpus and n_i is the number of documents containing the word i. Finally, the TF-IDF weight of a word i in document j is given by:

$$TF - IDF_{ij} = TF_{ij} \times IDF_i$$

At this point, the same equation 2.3 can be used to define the TF-IDF representation of document j that can be obtained by computing the TF-IDF weights for all words in the vocabulary, resulting in a vector x_j of length $|V|$, where $|V|$ is the size of the vocabulary. In 2.3, x_{ij} is now the TF-IDF weight of word i in document j.

Despite not being used in this thesis, there are also alternative weighting functions to TF-IDF, like the *Positive Pointwise Mutual Information* (PPMI). PPMI draws on the intuition that the best way to weigh the association between two words is to ask how much more the two words co-occur in our corpus than we would have a priori expected them to appear by chance. The interested reader can find more details in [67, 68].

While widely used in some applications, the BoW models still have disadvantages. First, BoW only considers the frequency of words in a document, ignoring the order in which they appear and the context in which they are used, and does not capture the semantic relationships between words, such as synonymy, antonymy, or polysemy. This can badly affect the performance of NLP tasks. Finally, it has difficulty handling rare words specific to a particular domain or text, as they may not appear in the training data and will be excluded from the vocabulary. Furthermore, with a large vocabulary size, the BoW representation can become high-dimensional and sparse, making it

computationally expensive to process and store. A more sophisticated approach is to create a vocabulary of grouped words. This changes the scope of the vocabulary and allows the BoW to capture more meaning from the document. In this approach, each token is called a *N-gram*. For example, a 2-gram (more commonly called a bi-gram) is a two-word sequence of words, and a 3-gram (more commonly called a tri-gram) is a three-word sequence of words.

2.3.2 Dense vectors representation: Word2Vec, GloVe and FastText

As stated above, in the BoW representations, text documents are represented as sparse vectors, where each element in the vector corresponds to a word in the vocabulary, and the value of each element reflects the importance or frequency of the word in the document. However, since most words in a document are not used, these representations are very sparse, with most elements having a value of zero. This can lead to high-dimensional computationally demanding representations where algorithms perform poorly.

Dense vectors, instead provide a more compact representation of the data by representing each word as a dense vector in a lower-dimensional space, where the similarity between the vectors reflects the semantic similarity between the words. Dense vector representations are obtained using techniques such as Word2Vec [69, 70], GloVe [71], or FastText [72].

Word2vec embeddings are static embeddings, meaning the method learns one fixed embedding for each word in the vocabulary. The word2vec's intuition is that instead of counting how often each word w_1 occurs near w_2, a logistic regression classifier (refer to Section 2.5 for details on classifiers and the classification task) is trained on a binary prediction task asking if w_1 is likely to show up near w_2. The learned classifier weights are taken as the word embeddings. The running text is implicitly treated as training data for such a classifier, and thus this method is also called self-supervision.

Another very widely used static embedding model is GloVe [71], short for Global Vectors. GloVe is essentially a log-bilinear model with a weighted least-squares objective. The main intuition underlying the model is the simple observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning. The training objective of GloVe is to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. Owing to the fact that the logarithm of a ratio equals the difference of logarithms, this objective associates the

logarithm of ratios of co-occurrence probabilities with vector differences in the word vector space [71]. Because these ratios can encode some form of meaning, this information gets encoded as vector differences as well.

The word2vec and Glove embeddings can not directly deal with out-of-vocabulary (OOV), i.e., words that appear in a text corpus but were unseen in the training corpus.

To deal with these problems, FastText [72] uses subword models, i.e., it represents each word as itself plus a bag of constituent n-grams, with special boundary symbols < and > added to each word. For example, with $n = 3$ the word *surgery* would be represented by the sequence < *surgery* > plus the character n-grams: < *su; urg; rge; ery; ry* >. Then an embedding is learned for each constituent n-gram, and the word *surgery* is represented by the sum of all of the embeddings of its constituent n-grams. Unknown words can be presented only by the sum of the constituent n-grams. Furthermore, thanks to subword information for representing the meaning of a word, FastText can handle short texts more effectively than word2vec.

2.4 Language models

A problem related to the representational learning discussed in the previous section is language modeling since the process of representation learning and feature engineering often depends on the underlying language models. Language models are a type of statistical model that uses machine learning algorithms to learn patterns and relationships within text data. There are various types of language models, summarized in the next sections.

2.4.1 Classical Language Models

In its base formulation, the goal of a statistical language model is that of estimating the probability of a given sequence of words, $W = [w_1, w_2, \ldots, w_m]$, in the language. This can be represented as:

$$\begin{aligned} P(W) &= P(w_1, w_2, \ldots, w_m) = P(w_1) \cdot (w_2|w_1) \cdot (w_3|w_1, w_2) \cdot \ldots \cdot P(w_m|w_1, w_2, \ldots, w_{m-1}) \\ &= \prod_{i=1}^{n} P(w_i|w_1, w_2, \ldots, w_{i-1}) \end{aligned}$$

where m is the length of the sequence of words, w_i is the $i-th$ word in the sequence, and $P(w_i|w_1, w_2, \ldots, w_{i-1})$ is the conditional probability of the $i-th$ word given the previous words in the sequence. This represents the probability of observing word w_i in the sequence given the context of the previous words. Each of the terms $P(w_i|w_1, w_2, \ldots, w_{i-1})$ needs to be estimated directly from the dataset:

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) = \frac{P(w_1, \ldots, w_i)}{P(w_1, \ldots, w_{i-1})} = \frac{\text{Count}(w_1, \ldots, w_i)}{\text{Count}(w_1, \ldots, w_{i-1})}$$

Issues may arise for large values of the group size i. In such cases, the numerator and the denominator can be close to 0. To address this problem, the *short-memory assumption* can be used. According to it, only the last $n-1$ tokens are used to estimate the conditional probability of a token, which results in an n-gram model.

Mathematically, the short-memory assumption for the n-gram model can be written as follows:

$$P(w_i|w_1, \ldots, w_{i-1}) \approx P(w_i|w_{i-n+1}, \ldots, w_{i-1}) = \frac{\text{Count}(w_{i-n+1}, \ldots, w_i)}{\text{Count}(w_{i-n+1}, \ldots, w_{i-1})}$$

If $n = 2$, we are referring to a bi-gram model, whereas if $n = 3$, we are referring to a tri-gram model.

Language models are strictly related to word embeddings because can be used to develop them in several ways: the training process of a language model provides a way to learn word embeddings, by estimating the probabilities of words given the context information provided by the surrounding words in the sentence.

2.4.2 Transformer-based pre-trained language models

While classical language models have been state-of-the-art in NLP for several years, this thesis widely uses Transformer based pre-trained language models [34] that have revolutionized the NLP state of the art. BERT (Bidirectional Encoder Representations from Transformers) was the first paper using this neural architecture. BERT's key technical innovation is applying the self-attention model [33] to language modeling. Thanks to it, the obtained language models learn contextual relations between words (or sub-words) in a text: since one word can have different meanings in different contexts, attention allows the model to look at other positions in the input sequence for clues that can help lead to a better encoding for the current word. Unlike directional models, which read

the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word). A language model which is trained with the self-attention mechanism can have a deeper sense of language context and flow than single-direction language models [34].

From the architectural point of view, in its base form, a transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary (in contrast, e.g. to denoising autoencoders such as BART [73]). Figure 2.1 illustrates at a high-level the Transformer encoder.

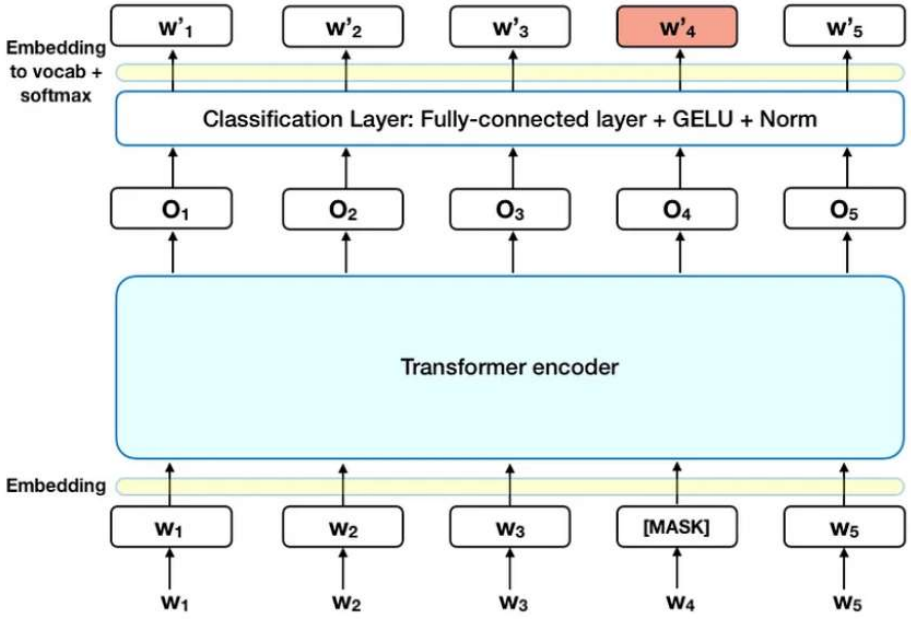


Fig. 2.1: The Transformer encoder: the input is a sequence of tokens, first embedded into vectors and then processed in the neural network. The output is a sequence of vectors of size H, in which each vector corresponds to an input token with the same index.

BERT also adopts a novel training technique named Masked Language Model (MLM), which allows bidirectional training. In the MLM task, a token w_t is replaced with $\langle mask \rangle$ and predicted using all past and future tokens:

$$\boldsymbol{W}_{\setminus t} := (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{t-1}, \boldsymbol{w}_{t+1}, \ldots, \boldsymbol{w}_{|W|})$$

During training with MLM, before feeding word sequences into the model, 15% of the words in each sequence is replaced with a $\langle mask \rangle$ token. The model then attempts to predict the original value of the masked words based on the context provided by the other non-masked words in the sequence. In technical terms, the prediction of the output words requires:

- Adding a classification layer on top of the encoder output.
- Multiplying the output vectors by the embedding matrix, transforming them into the vocabulary dimension.
- Calculating the probability of each word in the vocabulary with softmax.

In addition to MLM, the BERT training process adopts the Next Sentence Prediction (NSP) strategy. The model receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document. During training, 50% of the inputs are a pair in which the second sentence is the subsequent sentence in the original document, while in the other 50%, a random sentence from the corpus is chosen as the second sentence. When training the BERT model, MLM and NSP are trained together to minimize the combined loss function of the two strategies. BERT was trained on 800M words from BooksCorpus and 2,500M words from English Wikipedia.

From BERT, different variants have been proposed. One of the most famous and also used in this thesis is RoBERTa (short for "Robustly Optimized BERT Approach") [74]. It adopts the same BERT architecture while being trained on a larger dataset that goes over 160GB of uncompressed text, with sources ranging from the English language encyclopedic and news articles to literary works and web content. Representations learned by such models generally achieve strong performance across many tasks with datasets of varying sizes drawn from a variety of sources.

One key difference between RoBERTa and BERT is that RoBERTa was trained on a much larger dataset using a more effective training procedure. In particular, RoBERTa has improved BERT by:

- removing the NSP objective: the authors experimented with removing NSP loss, concluding that this removal slightly improves downstream task performance.

- training with bigger batch sizes and longer sequences: the large batches improve perplexity and accuracy on the masked language modeling objective; furthermore, large batches are also easier to parallelize via distributed parallel training.
- training via MLM with dynamic masking, i.e., a masking pattern is generated every time a sequence is fed to the model.

An interesting aspect of pre-trained language models (both BERT and RoBERTa) is that they can be fine-tuned for a large number of NLP tasks with a modest amount of training data and computational resources, achieving state-of-the-art results on many of them, such as sentiment analysis, textual entailment, and natural language inference, crucially also across languages [75]. This means that when the pre-training is complete, the obtained language model is saved as a set of parameters, which can then be loaded and fine-tuned on a smaller, task-specific dataset, simply adding standard layers on top of the architecture. The fine-tuning step involves updating the parameters of the pre-trained model to minimize a task-specific loss function.

2.4.3 Evaluating Language Models with Perplexity

As stated before, in its base formulation, the goal of a statistical language model is to estimate the probability that a particular word w appears after a sequence of observed words. The evaluation of language models is therefore based on statistically characterizing the likelihood of the presence of w after the observed sequence, and Perplexity (P) is one of the most common metrics adopted for this purpose.

Perplexity is defined as the exponentiated average negative log-likelihood of a sequence. For example, if we have a sequence $W = (w_1, \ldots, w_{|W|})$, the perplexity of W is:

$$P(W) = \exp\left\{-\frac{1}{|W|}\sum_{i=1}^{|W|}\log p_\theta(w_i|w_{<i})\right\} \tag{2.4}$$

Perplexity is not well defined for language models trained on MLM, such as BERT and RoBERTa. For these models, we can compute the perplexity from their *pseudo-log likelihood scores (PPL)* [76] instead, which corresponds to the sum of conditional log probabilities of each sentence token [77]. Formally, the pseudo-log likelihood scores (PPL) of a sentence $\boldsymbol{W} = (w_1, \ldots, w_{|\boldsymbol{W}|})$ under a language model with parameters Θ is defined as:

$$PPL(\boldsymbol{W}) := \sum_{t=1}^{|\boldsymbol{W}|}\log P_{\text{MLM}}(\boldsymbol{w}_t|\boldsymbol{W}_{\setminus t};\Theta)$$

where $P_{\text{MLM}}(\boldsymbol{w}_t|\boldsymbol{W}_{\setminus t};\Theta)$ is the conditional probability of token $\boldsymbol{w}_t$ given all past and future tokens $\boldsymbol{W}_{\setminus t} := (w_1,\ldots,w_{t-1},w_{t+1},\ldots,w_{|W|})$.

The (pseudo) *perplexity* PP of a masked language model [78] on a corpus of sentences $\mathbb{W}$, is then computed as:

$$PP(\mathbb{W}) := \exp\Big(-\frac{1}{N}\sum_{\boldsymbol{w}\in\mathbb{W}} PPL(\boldsymbol{W})\Big) \tag{2.5}$$

where N is the number of tokens in the corpus.

A lower perplexity value indicates that a model is making more confident and accurate predictions, thus indicating that the model has learned from the training data, and can well generalize to unseen data. A higher perplexity value indicates that a model makes less confident and less accurate predictions. This may be due to several factors, such as overfitting the training data, or a lack of data to learn from. For example, a model with a perplexity of 2 means that the model is on average twice as uncertain about the next word in the sequence compared to a model with a perplexity of 1.

2.5 Machine Learning for data classification

2.5.1 Definition

The first part of this thesis mainly deals with procedural sentence detection from surgical textbooks, academic papers, or online textual resources. As later explained in its dedicated chapter, we treated this task as a *text classification* problem. The final part of this thesis instead deals with surgical gesture classification, i.e., the task of recognizing the surgical gesture given its corresponding associated kinematic data. The goal of this section is thus to define the classification problem as a special kind of supervised learning task by explaining the main background technologies later used.

The goal of classification is to take a single observation, extract some useful features, and thereby classify the observation into one of a set of discrete classes. The task of supervised classification is to take an input x and a fixed set of output classes $Y = \{y_1, y_2, \ldots, y_m\}$ and return a predicted class $y \in Y$.

When the observation is a text, we face a *text classification* task. In this case, the task is that of assigning a label to an entire document or sentence. One of the most common examples is that of the already mentioned sentiment analysis, whose goal is the extraction of sentiment, i.e., the positive, neutral, or negative orientation that a writer

expresses toward some object that, according to the specific task, can be a movie, a book, a product or a person. In this case, the task is a ternary classification task because there are three classes to choose from. Another classic example is spam detection, the binary classification task of assigning an email the label *spam* or *not-spam*. Finally, one of the oldest tasks in text classification is assigning a library subject category or topic label to a text, an important sub-task of information retrieval. In this case, various sets of subject categories exist and therefore is a multi-class text classification task.

While rule-based approaches have been proposed in the past, nowadays, classification is mostly solved via supervised machine learning.

2.5.2 Main algorithms of data classification

Data classification is an instance of machine learning where specific algorithms and pre-trained models are used to cluster raw data into predefined categories. The most popular data classification algorithms are summarized below.

Logistic Regression

The Logistic Regression algorithm implements a linear equation with independent or explanatory variables to predict a response value.

If we have one explanatory variable x_1 and one response variable z, then the linear equation would take the form of:

$$z = \beta_0 + \beta_1 x_1$$

where the coefficients $\beta 0$ and $\beta 1$ are the parameters of the model. If there are multiple explanatory variables, then the above equation can be extended to:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

The predicted response value z is then converted into a probability value that lies between 0 and 1 thanks to the sigmoid function:

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

To map this probability value to a discrete class, a threshold value (also called decision boundary) has to be chosen. Generally, the decision boundary is set to 0.5.

Support Vector Machines

Support Vector Machine (SVM) is an approach for supervised machine learning classification.

Given a set of input data, it tries to determine which of two possible classes each data point belongs to. It does this by finding the optimal decision boundary. In SVM, the decision boundary is defined by a line (or hyperplane) that separates the two classes with the maximum margin. The margin is the distance between the hyperplane and the closest data points from each class, known as *support vectors*.

In linear SVM classifiers, the decision boundary is a straight line that separates the two classes. It is created by finding the line that maximizes the margin between the two classes, meaning that it tries to maximize the distance between the line and the closest data points from each class. The SVM algorithm optimizes this line by using a mathematical objective function, which considers the distances between the data points and the hyperplane.

Mathematically, the linear SVM classifier solves the following optimization problem:

$$\min_{w,b} \frac{1}{2}||w||^2 \; s.t. \; y_i(w^T x_i + b) \geq 1, \; i = 1, 2, \ldots, n \tag{2.6}$$

where w and b are the parameters of the hyperplane, x_i is the i-th feature vector, $y_i \in \{-1, 1\}$ is the corresponding label of the i-th instance, and n is the total number of instances in the training set. The constraint $y_i(w^T x_i + b) \geq 1$ ensures that all the instances are correctly classified, and the margin between the hyperplane and the closest points is at least 1. The objective function $\frac{1}{2}||w||^2$ encourages a simple and compact solution.

Once the decision boundary has been determined, new data points can be classified by measuring their distance from the boundary.

Random Forest Classifier

The random forest [79] consists of many individual decision trees that operate as an ensemble. Each tree in the random forest outputs a class prediction, and the class with the most votes becomes our model's prediction.

More formally, a random forest is an ensemble of different axis-parallel decision trees trained independently. In the random forest classifier, each non-leaf node is associated with a split function $f(x;\theta)$:

$$f(x;\theta) = \begin{cases} 1 \text{ if } x(\theta_1) < \theta_2 \\ 0 \text{ otherwise} \end{cases}$$

where $\theta_1 \in \{1,2,\ldots,d\}$ is the selected feature and $\theta_2 \in \mathbb{R}$ is a threshold. The outcome determines the child node to which x is routed. For instance, 0 may represent the left child node while 1 may represent the right child node. The leaf nodes of the tree either store class probability distributions or class labels based on the training samples they receive. During testing, for a test sample x, each tree returns a probability distribution $p_t(y|x)$ stored on the leaf node it falls into, and the class label is obtained via averaging.

Naive Bayes Classifier

Naive Bayes is a simple algorithm that classifies text based on the probability of the occurrence of events. This algorithm is based on the Bayes theorem, which helps in finding the conditional probabilities of events that occurred based on the probabilities of occurrence of each event. This model also requires a training dataset that contains a collection of sentences labeled with their respective classes. Using the Bayesian equation, the probability is calculated for each class with their respective sentences. Based on the probability value, the algorithm decides whether the sentence belongs to a question or statement class.

Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a deep learning architecture widely used in images, texts, and audio classification. It is composed of three main layers, named *convolutional layer*, *pooling layer*, and *fully-connected layer*.

The *convolutional layer* is the fundamental building block of a CNN, carrying out the majority of computations. The actors involved are the input data, a feature detector (also called a filter or kernel), and a feature map. The feature detector is an N-dimensional (N-D) array of weights that represents the N parts of the input. N is typically one for text or two for images. The filter is then applied to a portion of the input data, i.e., a dot product is computed between that portion and the filter. This dot product is then stored in an output array. The filter then shifts by a stride and the process is repeated until the entire input data has been traversed. The output resulting from this series of dot products is referred to as a feature map, activation map, or convolved feature. Multiple convolutional layers can be stacked after the initial one, each focusing on a different aspect of the input. The combination of these individual parts forms a

hierarchical structure of features, where each sub-layer addresses a distinct portion of the input and their collective representation captures higher-level patterns.

Pooling layers are responsible for reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation involves sweeping a filter across the entire input and applying an aggregation function to the values within the receptive field, thereby populating the output array. Although some information may be lost during the pooling process, it aids in reducing complexity, enhancing efficiency, and mitigating overfitting.

Finally, the *fully-connected layer* carries out the classification task based on the features extracted by the preceding layers and their various filters. It combines these extracted features to make predictions and assign class probabilities. To do it, a softmax activation function is generally applied. It takes a vector of real numbers as input and transforms them into a probability distribution over multiple classes, i.e., ensuring that the output values range between 0 and 1 and that they sum up to 1, making them interpretable as probabilities. The class with the highest probability is typically selected as the predicted class label.

Bi-LSTM

A Long Short-Term Memory (LSTM) is a deep learning architecture (recurrent neural network) capable of processing sequential data in a single direction, from the beginning to the end. Differently, a Bi-LSTM consists of two separate LSTMs that process the input sequence in opposite directions: one LSTM processes the sequence from the beginning to the end (the forward LSTM), and another LSTM processes the sequence from the end to the beginning (the backward LSTM). The outputs of these two LSTMs are concatenated or summed to provide a final output. Each LSTM network in a Bi-LSTM architecture comprises a series of repeating LSTM units or cells, each containing a cell state, an input gate, an output gate, and a forget gate. The cell state is responsible for storing and updating the memory of the network, while the gates control the flow of information into and out of the cell.

The network can access past and future information about each element by processing the sequence in both directions. This can be especially useful in speech recognition or natural language processing, where the context of a given element in the sequence is essential for determining its meaning.

2.5.3 Evaluation metrics

Standard metrics for evaluating data classification tasks include precision, recall, F1, and accuracy. Let *TP*, *TN*, *FP*, and *FN* denote the number of true positive, true negative, false positive, and false negative predictions, respectively, made by a classification model. Let *P* denote the number of actual positive cases in the data, and let *N* denote the number of actual negative cases in the data. Then, the following metrics can be defined:

Accuracy:

$$\frac{TP+TN}{P+N} \tag{2.7}$$

It measures the proportion of correct predictions made by the model.

Precision:

$$\frac{TP}{TP+FP} \tag{2.8}$$

It measures the proportion of true positive predictions out of all positive predictions made by the model.

Recall:

$$\frac{TP}{P} \tag{2.9}$$

It measures the proportion of true positive predictions from all actual positive cases in the data.

F1-score:

$$2 * \left(\frac{precision * recall}{precision + recall} \right) \tag{2.10}$$

It is the harmonic mean of precision and recall, providing a balanced measure of both metrics.

Micro, *macro*, and *weighted* metrics are used to compute evaluation metrics in the context of multi-class classification, which involves predicting multiple classes.

Micro metrics calculate the overall metric across all classes by summing up the corresponding values of true positive, false positive, and false negative across all classes. This results in a single evaluation score that reflects the overall performance of the model. Macro metrics calculate the average metric across all classes by averaging the corresponding values of precision, recall, or F1 score across all classes. This provides insight into how well the model performs for each class separately. Weighted metrics are similar to macro metrics, but they take into account the class imbalance by weighting the metrics by the number of samples in each class.

2.6 Semantic Role Labeling

A big part of this thesis deals with information extraction from procedural surgical texts. We treated this problem as a Semantic Role Labeling (SRL) task.

2.6.1 Definition

SRL is the task of labeling semantic arguments of predicates in sentences to identify *"Who"* does *"What"* to *"Whom"*, *"How"*, *"When"* and *"Where"*. Although there is not a universally adopted notation, in this thesis we refer to the following terminology:

- *framebank*: it is the lexical resource encoding different predicate's frames and roles.
- *predicate*: it is the action to which the various semantic arguments are connected;
- *frame*: it is the specific meaning that a predicate assumes in a given context; generally, each frame is accompanied by a list of expected semantic roles.
- *role*: it is the tag that is used to label the different arguments of a sentence; this tag is frame-specific but basically answers the question *who?* or *what?* or *whom?* or *how?* or *when?* or *where?*
- *argument*: it is the span of text that is labeled with a role.

The typical SRL task is composed of two sub-tasks:

1. Predicate identification and disambiguation: to identify each *predicate* in a sentence, assigning it the appropriate *frame*, i.e., the meaning it assumes in the given context, among the available ones for that lemma codified in the target lexical resource;
2. Argument identification and classification: to detect the *argument* spans or *argument* syntactic heads of a predicate, and to assign them the appropriate semantic *role* labels according to the target lexical resource.

To better clarify, one example follows. Given the sentence *"Yesterday Mary bought the book from John."*, in the predicate identification and disambiguation phase, SRL identifies that *"bought"* is the predicate and it has a meaning related to commerce. Then, in the argument identification and classification, the SRL has to identify that:

- *Yesterday* is the time reference of when the action is performed;
- *Mary* is the one who performs the action;
- *bought* is the action, i.e., the predicate identified and disambiguated in the previous step;

- *the book* is the object undergoing the action, in this case, the object bought;
- *John*, in this context, is the seller.

The way a sentence is labeled depends on the lexical resource used.

2.6.2 Lexical resources

The two most used lexical resources for SRL are PropBank [56] and FrameNet [80], and they use different typologies of semantic roles.

PropBank lexical resource

The Proposition Bank, generally referred to as PropBank, is a resource of sentences annotated with semantic roles. The English PropBank labels all the sentences in the Penn TreeBank; the Chinese PropBank labels sentences in the Penn Chinese TreeBank. Because of the difficulty of defining a universal set of thematic roles, the semantic roles in PropBank are defined concerning an individual verb sense. Each sense of each verb thus has a specific set of roles, which are given only numbers rather than names: Arg0, Arg1, Arg2, and so on. In general, Arg0 represents the one who performs the action, and Arg1 represents the one who is subjected to the action. The semantics of the other roles are less consistent, often being defined specifically for each verb. Nonetheless, there are some generalizations; the Arg2 is often the benefactive, instrument, attribute, or end state, the Arg3 the start point, the benefactive, instrument, or attribute, and the Arg4 the endpoint. Here are simplified PropBank entries for two of the senses of the verb *buy*:

- buy.01 - purchase
  - Arg0: buyer
  - Arg1: thing bought
  - Arg2: seller
  - Arg3: price paid
  - Arg4: benefactive
- buy.05 - accept as truth
  - Arg0: believer
  - Arg1: thing believed

Such PropBank entries are called frame files; the definitions in the frame file for each role (“buyer”, “thing bought”) are informal glosses intended to be read by humans rather than formal definitions.

PropBank also has many non-numbered arguments called ArgMs, (ArgM-TMP, ArgM-LOC, etc.) representing modification or adjunct meanings. These are relatively stable across predicates, so they are not listed with each frame file. Data labeled with these modifiers can be helpful in training systems to detect temporal, location, or directional modification across predicates. Some of the ArgMs include:

- TMP: when?
- LOC: where?
- DIR: where to/from?
- MNR: how?
- PRP: why?
- ADV: miscellaneous

The above example is instantiated in the PropBank scenario as follows. In the predicate identification and disambiguation phase, PropBank’s SRL identifies that *“bought"* is the predicate, and in this sentence, it has, among the different alternative senses for *“buy"* codified in PropBank, the meaning *buy.01 - purchase*. This means that semantic roles have to be chosen within the frame *buy.01 - purchase*. In the argument identification and classification phase, PropBank’s SRL produces the following output:

> “[**ArgM-TMP**: Yesterday] [**Arg0**: Mary] [**buy.01**: bought] [**Arg1**: the book] [**Arg2**: from John]."

The meaning of the labels is specified in the corresponding framebank.

Framenet lexical resource

The FrameNet project [80, 81] is another SRL project. Whereas roles in the PropBank project are specific to an individual verb, roles in the FrameNet project are specific to a coherent chunk of commonsense background information concerning a specific concept. For example, the concept *buyer*, *goods*, *money*, and *seller* are all linked to the same concept, in this case, the *commercial scenario*. The idea is then that of grouping words around a specific concept to which they are related and not around a specific verb. The same example as before:

> “[**Time**: Yesterday] [**Buyer**: Mary] [**bought**] [**Goods**: the book] [**Seller**: from John]."

Also in FrameBank, there exists the concept of core and non-core roles: the firsts are concept-specific, while the latter express more general properties of time and location and are more similar to PropBank's ArgM arguments.

Other lexical resources

Other lexical resources for SRL not used in this thesis are AMR (Abstract Meaning Representation) and VerbNet: the interested reader can find more details in [82, 83].

2.6.3 Main semantic role labeling models

SRL is traditionally performed with data-driven methods [84]. Traditional approaches were based on classifiers trained on manually-engineered textual features: e.g. [85] proposes a statistical classifier trained using various morpho-syntactic features (e.g. governing predicate, phrase type). Recent works on SRL leverage deep neural networks, shifting from feature engineering to architecture engineering. Several notable approaches suggest performing SRL in an end-to-end fashion, relying only on raw low-level input signals (characters/tokens) fed to advanced models, such as multi-layer recurrent networks [86]. More recently, approaches leveraging self-attention techniques [87] and Transformer-based architectures with pre-trained language models [88] have been proposed.

2.6.4 Evaluation metrics

The evaluation of SRL methods typically involves comparing the predicted roles for predicates and arguments to a set of manually annotated gold standard annotations. The used metrics are precision, recall, and F1-score. These are standard evaluation metrics for binary or multiclass classification tasks already discussed in Section 2.5. They evaluate the ability of the model to correctly identify and classify the semantic roles of predicates and their arguments.

Two common datasets used for evaluation are CoNLL-2005 [89] and CoNLL-2012 [90], both exploiting PropBank as frameBank.

2.7 Conclusions

This chapter has summarized all concepts and technologies that are used in the other parts of this thesis. It has introduced the concept of machine and deep learning, and

best practices to follow during the annotation process, which is a paramount step in supervised learning. Then, since the majority of the thesis deals with NLP techniques applied to texts, this chapter has compared traditional language modeling techniques with the recent pre-trained Transformer-based ones. This chapter has finally defined some tasks used in the other chapters of this thesis, namely the (textual or kinematic) data classification and semantic role labeling.

Part II

Procedural knowledge from surgical textbooks

This part presents our contributions to procedural robotic-surgery knowledge extraction from textbooks and academic papers. First, SURGICBERTA*, a novel pretrained language model for surgical language, is presented in Chapter 3.* SURGICBERTA *is then used in Chapter 4, together with other state-of-the-art deep learning methods, to detect in a text the sentences containing procedural knowledge discarding the others. Then, the task of robotic-surgery procedural knowledge understanding is covered by Chapters 5-8. In particular, Chapter 5 defines a proper surgical framebank, adapting an existing general-English framebank to the robotic-surgery domain. The obtained resource is used to annotate a corpus for SRL of as-is surgical sentences taken from surgical books and academic papers in Chapter 6. The resulting annotated corpus is then used to train, validate and test* SURGICBERTA *on the SRL task. Finally, Chapter 7 proposes a pipeline based on SRL and some syntactic rules to empirically demonstrate how, within simple language constraints, it is possible to extract a logical template from sentences written in natural language. Finally, since not all information needed to automate a task is expressed in textbooks, a mapping between commonsense knowledge and autonomy levels is proposed in Chapter 8 to guide future research directions.*

3

Developing a pre-trained language model for surgical language

"Larvatus prodeo [Masked, I go forward]"

René Descartes

3.1 Introduction

While a large number of domain-specific language models have been developed with an improved understanding of the semantic information in their field of expertise, to the best of our knowledge, a specialized model specific to the surgical language does not exist yet, even if the scientific community has shown a growing interest in the application of NLP in surgery, especially for the image captioning task [91, 92, 93, 94, 95]. As stated in the introduction, surgical literature is teeming with books, online resources, and academic papers of the highest quality used by universities worldwide.

This chapter introduces a new pre-trained language model, named SURGICBERTA, trained on a large quantity of surgical textual material. In more detail, this chapter describes:

1. the development of SURGICBERTA, a pre-trained language model specific for the understanding of procedural surgical language;
2. the intrinsic evaluation of SURGICBERTA with respect to the general-purpose model ROBERTA;
3. a preliminary extrinsic evaluation of SURGICBERTA with respect to ROBERTA, that is, comparing their performances when employed on different downstream tasks. SURGICBERTA will also be used in Chapter 4 and 6 for the procedural sentence detection and procedural knowledge extraction tasks, respectively.

The quantitative assessments are complemented with qualitative analysis on SURGICBERTA, showing that it contains a lot of surgical domain knowledge that could be useful to enrich existing state-of-the-art surgical knowledge bases. The evaluation indicates that SURGICBERTA better deals with surgical language than a state-of-the-art yet open-domain and general-purpose model such as ROBERTA, and therefore can be effectively exploited in many computer-assisted applications, specifically in the surgical domain.

The chapter is organized as follows: Section 3.2 revises relevant works in this area. Then, SURGICBERTA is presented in Section 3.3. The required textual data is collected, extracted, pre-processed, and used for the continuous training of ROBERTA on the MLM task with domain-specific text. Section 3.4 presents the intrinsic metrics and tasks used to evaluate SURGICBERTA. Section 3.4.4 reports and qualitatively discusses some examples of surgical domain knowledge contained in SURGICBERTA. Finally, 3.5 summarizes obtained results and proposes future works.

3.2 State of the art

Pre-trained language models in biomedicine. As stated in 2.4.2, transformer-based pre-trained language models can be easily fine-tuned for several downstream tasks, including those relevant to the biomedical domain. The first language models were built for general English, and thus, as stated in the papers cited in this section, they may not be particularly adequate to cover specific domains due to frequently missing domain words or expressions. To overcome this limit, there is the possibility to train from scratch a model specific to a given domain of interest, such as in [96, 97] where large models specific to the clinical domain are proposed. Developing such a model from scratch is very expensive for the computational resources and the training time required. For this reason, domain-adaptation techniques, such as the MLM described in Section 2.4.2, have been proposed and widely used in biomedicine, together with fine-tuning for various downstream tasks. In [98], domain-adaptation is used to obtain a cancer domain-specific language model for effectively extracting breast cancer phenotypes from electronic health records. The authors of [99] developed a pipeline for pre-trained neural models to classify patients as seizure-free and extract text containing their seizure frequency and date of last seizure from clinical notes. The first step of this pipeline is the unsupervised domain adaptation, using progress notes that were not selected for annotation. The obtained model has been fine-tuned for the classi-

fication and extraction tasks. Also [100] adopted a domain adaptation technique on clinical notes from the Medical Information Mart for Intensive Care III database [101] to extract clinically relevant information. In [102], causal precedence relations are recognized among the chemical interactions in the biomedical literature to understand the underlying biological mechanisms. However, detecting such causal relations can be challenging because annotating such causal relation detection datasets requires considerable expert knowledge and effort. To overcome this limitation, in-domain pretraining of neural models with knowledge distillation techniques have been adopted, showing that the neural models outperform previous baselines even with a small number of annotated data. In [103], a domain-adaptation strategy is adopted to encourage the model to learn features from the context to curate all validated antibiotic resistance genres, i.e. the ability of bacteria to survive and propagate in the presence of antibiotics, from scientific papers. In [104], a domain adaptation technique has been used to align large language models to new medical domains, showing that, after a proper adaptation step, they encode some clinical knowledge usable in question-answering applications. Finally, a domain adaptation technique has been adopted for biomedical domain adaptation in languages different than English, such as Spanish [105] and Chinese [106], showing the same improvement trend when compared to the corresponding base models.

However, due to terminological differences between biomedical domains, it is often difficult to use these models to gain benefits outside the goal they were trained on. Differences are also in the structure of the sentence: for example, EHRs and clinical notes are often structured and concise and may use not explained abbreviations. Academic articles or textbooks use instead a language that, although still highly technical, is more accessible and accompanied by background information. Therefore, it is generally accepted that model performance may degrade when evaluated on data with a different distribution [107]. Consequently, domain adaptation on relevant domain data is essential to improve performance in very specialized domains [108], and despite the availability of several biomedical language models, to the best of our knowledge, a pretrained surgical language model is still missing. Such a model is essential for mining surgical procedural knowledge from text and developing intelligent surgical systems.

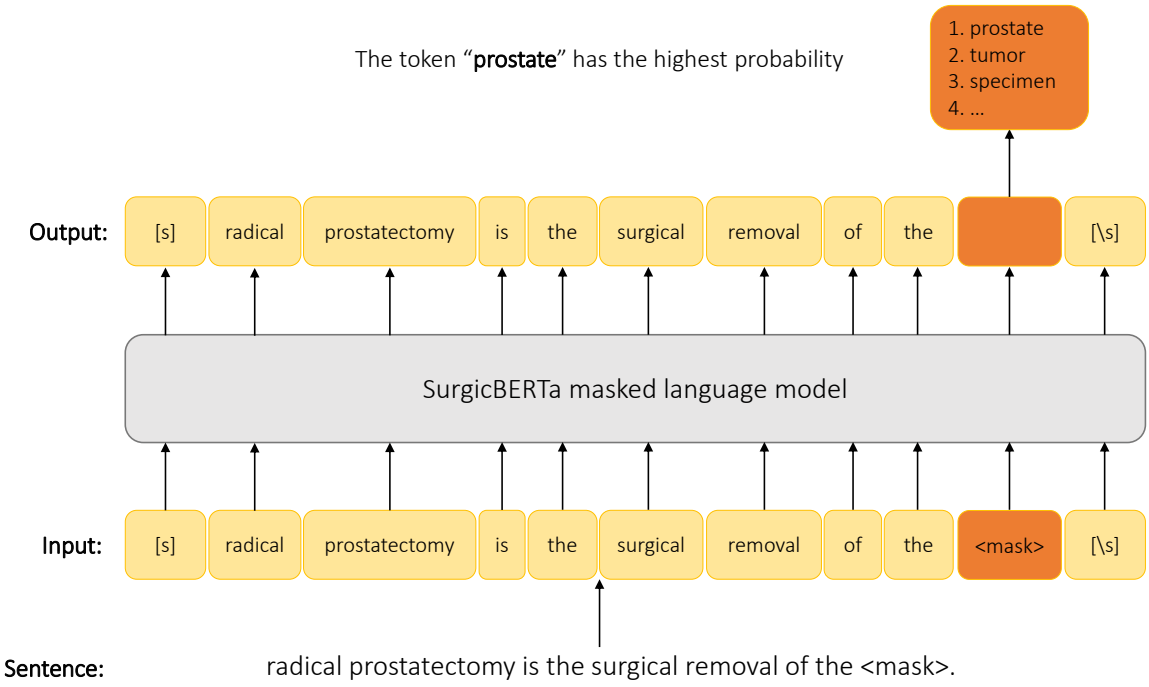


Fig. 3.1: MLM task used for adapting RoBERTa to the surgical domain. s and \s are special tokens denoting the sentence's beginning and end, respectively.

3.3 SurgicBERTa

This section describes the development of SurgicBERTa, the pre-trained language model for the surgical domain we released. SurgicBERTa has been developed on top of RoBERTa, the already available English pre-trained language model for the general domain described in Section 2.4.2.

Starting from RoBERTa, we develop a novel model specific to the surgical domain by continuously training RoBERTa for the MLM task on a large amount of surgical domain text. We recall that, in the MLM task, a token w_t is replaced with $\langle mask \rangle$ and predicted using all past and future tokens $\boldsymbol{W}_{\setminus t} := (\boldsymbol{w}_1, ..., \boldsymbol{w}_{t-1}, \boldsymbol{w}_{t+1}, ..., \boldsymbol{w}_{|W|})$. Figure 3.1 illustrates the MLM task used to obtain SurgicBERTa.

In more detail, 300K sentences from surgery books (7 million words) are selected. To obtain a surgical model as general as possible, the training sentences are selected from various books covering several heterogeneous surgical domains, from abdominal surgery to orthopedics, to eye surgery. We searched for surgery books written in En-

glish on the web pages of several publishing houses. As keywords, we used the name of the surgical macro-areas (e.g. general surgery, abdominal surgery, gynecology surgery, eye surgery). From the results, we downloaded the digital version only of the texts to which our universities have proper legal access. A very minimal pre-processing of the sentences is performed to remove URLs and bibliographic references.

In more detail, 15% of tokens are selected for possible replacement. Among those selected tokens, 80% are replaced with the special $\langle mask \rangle$ token, 10% are left unchanged, and 10% are replaced by a random token. The model is then trained to predict the initial masked tokens using cross-entropy loss. Following the RoBERTa approach, tokens are dynamically masked instead of fixing them statically for the whole dataset during preprocessing. This improves variability and makes the model more robust when training for multiple epochs.

3.4 Evaluation

We evaluate SurgicBERTa along several dimensions, comparing it with RoBERTa, the starting language model used for adaptation to the surgical domain. Section 3.4.1 presents the intrinsic evaluation, and Section 3.4.3 presents the two downstream tasks we use to evaluate SurgicBERTa, namely surgery and main anatomy link and surgical terminology acquisition. Chapter 4 will then use SurgicBERTa for the procedural/non-procedural surgical sentence classification, while Chapter 6 for the surgical information extraction.

3.4.1 Intrinsic evaluation

Evaluation metrics. In Section 2.4.3, equation 2.5 defined the pseudo-perplexity PP, the metric used to evaluate BERT-based language models. By computing PP on a test corpus for both RoBERTa and SurgicBERTa, we evaluate the model's ability to predict the unseen text from the corpus we used for evaluation and take this as an intrinsic evaluation metric of the quality of the two models in the surgical domain. The comparison is fair because RoBERTa and SurgicBERTa share the same tokenizer and vocabulary.

Other intrinsic metrics used in this chapter to evaluate RoBERTa and SurgicBERTa on the surgical domain are the accuracy of MLM computed on the masked words during the evaluation step and the evaluation loss. Accuracy measures how well our model predicts the masked words by comparing the model predictions with the proper values

Table 3.1: Perplexity, accuracy and evaluation loss. Bold values mark the better scores for each metric.

Pre-trained model	Perplexity	Accuracy	Evaluation Loss
RoBERTa	15.410	0.546	2.735
SurgicBERTa	**4.30**	**0.699**	**1.458**

in terms of percentage. Instead, the loss is a value that represents the summation of errors in a model. It measures how well or poorly the model is performing. If the errors are high, the loss will be high, and the model will not perform well.

Generally, the higher the accuracy in the evaluation dataset and the lower the evaluation loss, the better the model will perform.

Results and discussion. Table 3.1 reports perplexity, accuracy, and loss values of RoBERTa and SurgicBERTa obtained during the evaluation of the MLM task. SurgicBERTa has lower perplexity (−11.11), greater accuracy (+15.30%), and lower evaluation loss (−1.277) than RoBERTa. All obtained results intrinsically confirm that SurgicBERTa better deals with surgical language than RoBERTa.

3.4.2 Extrinsic Evaluation - Task 1

Task definition. The purpose of this task is to associate the name of the surgical procedure with the corresponding anatomical target or relevant feature to verify if the language models have learned this type of knowledge during training. For example, the *prostatectomy* has to be associated with *prostate*, *nephrectomy* with *kidney*, and *mastectomy* with *breast*. To evaluate our models on this task, we built a dataset consisting of the definition of 20 different surgical procedures. In particular, surgical procedures that can be performed with the aid of a robot have been chosen, together with other very frequent laparoscopic ones. The definitions are retrieved from the web or surgical manuals not used during the training of the language models. From them, the name of the corresponding anatomical target has been removed, and the models are asked to guess it. As evaluation metrics, we consider the ranking of the correct target word with respect to the others returned by the model, the probability that the model will select it, the Reciprocal Rank (RR), and the Mean Reciprocal Rank (MRR) [109]. MRR is a measure to evaluate systems that return a ranked list of answers to queries. In the case of this task, answers are words returned to fill the $\langle mask \rangle$, i.e. the anatomical part cor-

responding to the procedure description, and queries are the sentences describing the procedure. In more detail, for a single query, the RR is defined as $1/rank$, where *rank* is the position of the correct answer among the ones (sorted by probability, from the highest to the lowest) predicted by the model. For multiple queries $|Q|$, the MRR is the mean of the $|Q|$ RRs, i.e.:

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}RR_i \tag{3.1}$$

The vocabulary has not been restricted, i.e. a list of possible candidates to choose from has not been used so that models can return any word belonging to the vocabulary.

To better clarify with an example, consider the following sentence (i.e. query):

a sacrocolpopexy is a surgical procedure used to treat $\langle mask \rangle$ *organ prolapse.*

Models are asked to fill in the missing word with the correct one, which in the above example is *pelvic*. They will propose a list of possible candidates sorted by probability. For example, for the above sentence, RoBERTa and SurgicBERTa return the correct word *pelvic* in the third and first positions, respectively, thus obtaining both an RR of 0.33 and 1.0, respectively. The probability that RoBERTa will select the correct word is 0.043, while the one for SurgicBERTa is 0.33, which is significantly higher.

Results and discussion. This section summarizes the results of the above-described task, i.e. predicting the anatomical target given the name and a brief definition of the surgical intervention related to that target. On average, the correct target is returned by RoBERTa in position 2.35, while SurgicBERTa outperforms RoBERTa proposing the correct target in position 1.35. The MRR of RoBERTa is 0.731, while that of SurgicBERTa is 0.902. In more detail, 30% of the times SurgicBERTa performs substantially better than RoBERTa, while the contrary only holds in one case. The violin plots of Figure 3.2 summarize the obtained RRs on each query sentence: the one for SurgicBERTa is very wide at the top and skinny in the middle and the bottom, while the one of RoBERTa, albeit having a similar distribution, is much less wide at the top and has a median weight lower than that of SurgicBERTa. The shape of the distribution indicates that the RRs of SurgicBERTa are highly concentrated around the first quartile, meaning that the model predicts the proper anatomical target very well. In contrast, the RRs of RoBERTa are more evenly distributed across the entire range, highlighting

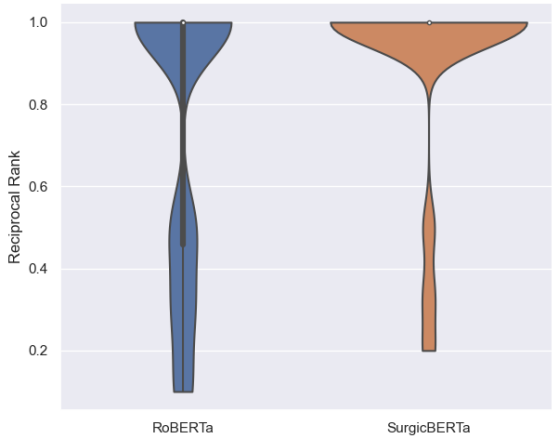


Fig. 3.2: Reciprocal rank of the predicted word in the task of predicting the anatomical target given the information of a surgical procedure (Extrinsic Evaluation - Task 1).

lower scores. Also this task confirms the benefit of having specialized RoBERTa for the surgical language.

3.4.3 Extrinsic Evaluation - Task 2

Task definition. This task is similar to the previous one but applied to a different dataset and therefore proposed for a different purpose: to verify whether SurgicBERTa masters the surgical language and can use it more appropriately than RoBERTa. In particular, a dataset of 50 surgical sentences was collected from different sources, i.e. surgical books, academic papers, and web pages not used during the MLM training. The sentences were randomly chosen from those that met the following requirements:

- the sentence has not been used to train SurgicBERTa;
- one of the following holds:
  - the sentence contains an expression commonly used in surgery. To define frequently used expressions, we have selected those typically abbreviated with an acronym in papers. In the sentences included in the dataset, the abbreviations have been substituted with the original expression, and the language models are asked to complete them correctly in the corresponding context;
  - the sentence contains a description of a surgical procedure. In the sentences inserted in the dataset, the verb describing the action is masked, and the language model is asked to guess it based on the context.

Table 3.2: Mean position, MRR, and mean probability on the task of surgical terminology acquisition (Extrinsic Evaluation - Task 2).

Pre-trained model	Mean position	MRR
RoBERTa	152.720	0.262
SurgicBERTa	**7.960**	**0.658**

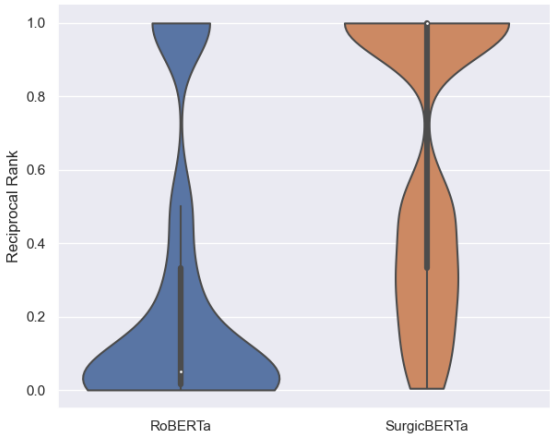


Fig. 3.3: Reciprocal rank of the predicted word in the task of surgical terminology acquisition (Extrinsic Evaluation - Task 2).

Since the configuration of the task is the same as the previous one, we used the same metrics adopted for it, i.e. the position in which the correct solution is proposed, the corresponding probability, the RR, and the MRR.

Results and discussion. Table 3.2 summarizes the obtained results for this task. SurgicBERTa substantially improves all proposed metrics: the SurgicBERTa mean position is 19.19 times better than the RoBERTa one; the MRR is improved by 0.396. 66% of the times SurgicBERTa improves the RRs when compared to RoBERTa. Only in two cases (out of 50) RoBERTa performs better than SurgicBERTa. The violin plots of Figure 3.3 illustrate the RRs of the two language models for each query: while the one for SurgicBERTa is wide at the top, the one for RoBERTa is wide at the bottom. Furthermore, SurgicBERTa has a median weight much higher than that of RoBERTa. This highlights the best accuracy of SurgicBERTa in learning surgical terminology. Also, this task confirms that SurgicBERTa better captures the surgical language.

Table 3.3: RoBERTa and SurgicBERTa most probable words for the most used surgical robots.

	RoBERTa		SurgicBERTa	
Rank	Word	Probability	Word	Probability
1	Braun	0.031	Zeus	0.261
2	Juno	0.027	Xi	0.111
3	Hawk	0.017	Si	0.055
4	Orion	0.016	robotic	0.035
5	MRI	0.016	S	0.030

3.4.4 Qualitative analysis

There is a lot of domain information implicit in pre-trained language models [110]. Adapting the domain through continual learning with MLM helps in capturing this knowledge. However, it is complicated to quantify this domain knowledge objectively and exhaustively due to the lack of any gold standard for the surgical domain. For this reason, this section proposes a qualitative analysis, providing examples of domain information stored in pre-trained language models.

To start with, RoBERTa and SurgicBERTa are asked to return the name of the most used surgical robot in the operating room. In particular, RoBERTa and SurgicBERTa are asked to substitute the $\langle mask \rangle$ in the following sentence with the most appropriate five words, ranking them in order of probability:

The most commonly used surgical robot is $\langle mask \rangle$.

Results are reported in Table 3.3. While to the best of our knowledge, none of the top five words returned by RoBERTa is the name of a surgical robot, *Zeus*[1], *Xi*[2], and *Si*[3] returned by SurgicBERTa are instead examples of surgical robots that have been used in operating theatres. This means that the continual MLM learning with domain text has captured this kind of information that is now available in the model.

As reported in Table 3.1, SurgicBERTa has a perplexity substantially lower than RoBERTa in the MLM task when applied to surgical literature. This intrinsically means that SurgicBERTa has learned the surgical language and thus also the composition

[1] https://en.wikipedia.org/wiki/ZEUS_robotic_surgical_system
[2] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6193435/
[3] https://www.davincisurgerycommunity.com/Systems_I_A/da_Vinci_Si_Si_e

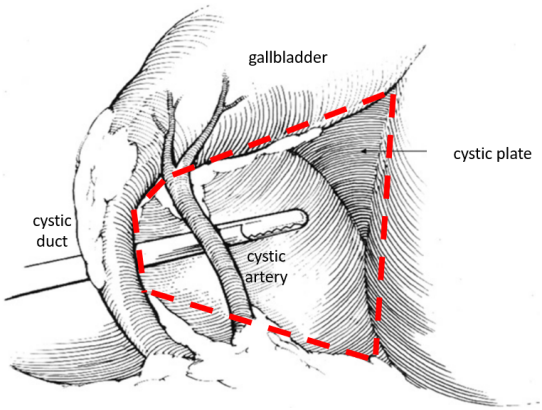


Fig. 3.4: Illustration of the critical view of safety method during a cholecystectomy.

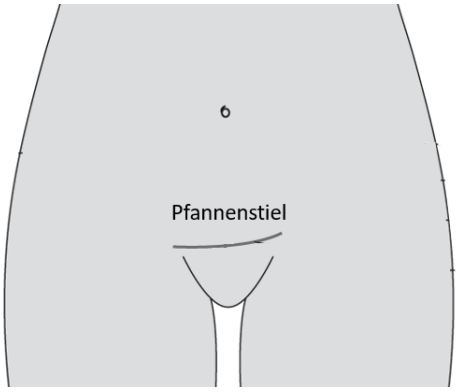


Fig. 3.5: Pfannenstiel incision to access the abdomen.

of well-known surgical expressions. Consider the following example highlights how SURGICBERTA has learned specialized domain terminology. In surgery, the expression *critical view of safety* refers to a method of secure identification in open cholecystectomy in which the cystic duct and artery are putatively identified, after which the gallbladder is taken off the cystic plate so that the gallbladder is attached only by the two cystic structures [111] as shown by Figure 3.4.

To verify if ROBERTA and SURGICBERTA know this information, they are asked to complete the following sentence:

During cholecystectomy, it is important to achieve the critical view of $\langle mask \rangle$.

SURGICBERTA returns the word *safety* as $1st$ result with a probability of 0.3428, while ROBERTA returns it only at $47th$ position with the probability of 0.0032.

This section ends with another example of domain knowledge available in SURGICBERTA. In surgery, a *Pfannenstiel incision* is a type of surgical incision that allows access to the abdomen (See Figure 3.5, adapted from [112].). The following test wants to investigate if pre-trained language models know this information:

The Pfannenstiel is a type of surgical incision that allows access to the $\langle mask \rangle$.

The correct word is *abdomen* and is retrieved by SURGICBERTA at the $1st$ position with probability 0.1267 and by ROBERTA at the $5th$ position with probability 0.0478, after the words *brain* (0.1969), *heart* (0.1488), *skin* (0.0713), and *vagina* (0.0542).

These qualitative examples show that in SURGICBERTA there is much surgical information that could be used, for instance, to enrich and complement the one codified in domain ontologies and knowledge bases.

3.5 Conclusions

This chapter proposed SURGICBERTA, a pre-trained language model fine-tuned for capturing surgical language and knowledge, i.e. the vocabulary and expertise provided in surgical books and academic papers.

The building process has been described, and the model has been evaluated both intrinsically by considering perplexity, accuracy, and evaluation loss during the MLM task and extrinsically by considering two downstream tasks. All the results confirm that SURGICBERTA deals with surgical language and knowledge more adequately than ROBERTA, a language model targeting general-domain English. Moreover, the potential of SURGICBERTA has been investigated qualitatively by showing some examples of surgical domain knowledge available in the model, which could complement other knowledge sources, e.g. state-of-the-art surgical knowledge bases. SURGICBERTA will also be used in the following chapters for procedural sentence detection and procedural knowledge extraction tasks.

4

Detecting sentences containing procedural knowledge in surgical textbooks

"The ability to simplify means to eliminate the unnecessary so that the necessary may speak."

Hans Hofmann

4.1 Introduction

This thesis's main objective is to develop a model able to automatically understand *procedural* written text of the robotic-surgical domain. In particular, we mainly target as-is textbooks and academic papers. Although the main goal of this kind of textual resource is to describe how to perform a given surgery by listing actions and the way they should be performed, the instruments to use, and spatial and temporal constraints to follow, they also contain *non-procedural* information, such as sentences introducing the history of the surgery, the number of surgeries of that type performed per year, the different typologies of patients, the related anatomy, positions and medium size of the organs, and other ontological information. This means that real-world documents usually describe surgical processes also including descriptive information, which is not directly useful to derive a workflow. This chapter tackles the problem of separating *procedural* from *non-procedural* sentences, a preliminary task towards the automatical understanding and extraction of procedural surgical sentences. Indeed, the overall research objective can be split into two main steps: first, *procedural sentences* i.e. sentences containing procedural knowledge, are recognized in a text; then, the recognized sentences are used to extract procedural surgical knowledge (objective of the Chapters 5-7).

This chapter presents our novel contribution to address the first of these steps, which, to our knowledge, has never been investigated before in surgery. We tackle this

problem as a sentence classification task by applying different machine learning (ML) and deep learning (DL) algorithms. In addition to consolidated ML approaches available in the literature and tested in other domains, we experiment with the FastText classifier (c.f., 2.3.2) since it demonstrated state-of-the-art performance in numerous text classification tasks. Moreover, we investigate the use of the subword-enriched word embeddings returned by FastText as features for a one-Dimensional Convolutional Neural-Network (1D-CNN) and a Bidirectional Long Short-Term Memory (Bi-LSTM) Neural-Network, described in 2.5. Finally, we test Transformers-based classification methods (c.f. 2.4.2), fine-tuning some pre-trained language models for the task.

To train and benchmark all these approaches, we introduce a novel surgical textual dataset, SPKS (Surgical Procedural Knowledge Sentences), consisting of sentences from surgical texts that we manually annotated as procedural or non-procedural. We presented this work in 2021 in [18]. At the end of 2022, we released SURGICBERTA, and, as part of its evaluation, we compared SURGICBERTA performance with that of its vanilla version, ROBERTA, on the same task to verify if MLM has led to benefits. Meanwhile, we have developed an extended version of the dataset, SPKSv1.1, containing more sentences and more procedures, and so we used it for SURGICBERTA and ROBERTA comparison. The results of the two experiments are so not directly comparable, but though out of the research question of the second evaluation, we also report the performance of the model that obtained the best results during the first evaluation.

In more detail, the following research questions are investigated during the first release of the dataset:

RQ1 Are the TF-IDF features fed to classic classification algorithms sufficient to detect procedural knowledge in surgical written texts? Is it necessary to resort to more sophisticated techniques of word embeddings and neural networks? Do more complex methods based on fine-tuning pre-trained language models outperform the other considered approaches?

RQ2 Do some dataset balancing techniques positively impact the performances of procedural sentence classification?

In the second evaluation, the following research question is instead investigated:

RQ3 As a continuation of the validation of Chapter 3, does SURGICBERTA better perform than ROBERTA also on the procedural robotic-surgery sentence detection task?

The contribution of this chapter is threefold:

- the proposal to address the detection of procedural knowledge in surgical texts as a sentence classification task;
- a novel, publicly-available, manually-annotated surgical textual dataset for benchmarking classification methods;
- a preliminary assessment on this dataset of various state-of-the-art classification methods.

4.2 State of the art

As stated before, to the best of our knowledge, works had yet to tackle the problem of detecting procedural sentences in surgical documents. However, approaches for detecting procedural sentences have been proposed in other domains and applied to typologies of textual content substantially different than the description of a surgical procedure, such as repair instructions [23, 25, 27], technical support documentation [22, 23, 26], instructions for nanomaterials' synthesis [24], cooking recipes [23, 27], and medical abstracts [113].

In [22], the authors tackle the problem of procedural knowledge detection in technical documentation as a classification task. They use a linear Support Vector Machine (SVM) exploiting both linguistic (usage of the imperative, declarative, conditional, or passive form) and structural (e.g. section/subsection organization, bulleted-list usage) features, showing that both of them contribute to improving performance.

The authors of [23] address the problem of identifying sentences mentioning actions in cooking recipes and maintenance manuals, exploiting a CNN fed with word embeddings. Classification ("relevant", "irrelevant") of recipe (for nanomaterials' synthesis) sentences is also investigated in [24], where the authors use a Naïve Bayes classifier fed with features such as word counts, TF-IDF (Term Frequency–Inverse Document Frequency) and N-grams.

In [25], the authors pursue the detection of repair instructions in user-generated text from automotive web communities. Various linguistic (bag-of-words, bag-of-bigrams, post length, readability index) and structural features (repair instructions are often provided as bulleted or numbered lists) are fed to several ML methods, from classical ones (e.g. Random Forest) to Neural-Networks (single and multilayer perceptrons).

In [26], an SVM is applied for detecting procedural sentences in technical support documentation, where procedures are typically described using lists. Besides tradi-

tional features, such as TF-IDF, the authors show the effectiveness of exploiting information on the list type, contextual features (e.g. sentences introducing a list), and the usage of imperatives.

The authors of [113] address the detection of procedural knowledge in MEDLINE abstracts. In their work, procedural knowledge is defined as a set of *unit procedures* (each consisting of a *Target*, *Action*, and *Method*) organized for solving a specific purpose. The proposed solution works in two steps. First, SVMs and Conditional Random Fields (CRFs) are combined for detecting sentences (purpose/solution) that may contain unit procedures, feeding them with content (unigrams and bigrams), position (sentence number in the abstract), neighbor (content features of nearby sentences) and ontological features (usage of terms from reference vocabularies). Then, sequence labeling with CRFs is performed to identify the components of unit procedures.

Finally, the authors of [27] address the extraction of procedural knowledge from structured instructional texts. First, they partition sentences into related segments, from which finite-state grammars are applied to extract procedural elements. Next, rule-based reasoning is applied to resolve certain types of omissions and ambiguities in instructions.

While all these works address the detection of procedural knowledge from written text, the proposed methods have been applied to typologies of textual content substantially different from the description of a surgical procedure. Troubleshooting and product documentation, cocking recipes, maintenance manuals, and repair instructions differ from descriptions of surgical procedures both on the terminological/language level and the structural one: typically, these kinds of texts are structurally organized, frequently using numbered/bulleted lists — a characteristic effectively exploited as a feature in many of the discussed approaches — while no established standard way to describe a surgical procedure exists. In addition, surgical interventions are mainly presented in a prose-like style. Indeed, the scenario where structural features cannot be exploited is considered more challenging to tackle (c.f. [26]).

Furthermore, all the approaches mentioned above have been applied to domains substantially different from the surgical one. In this regard, the closest work is [113]: however, MEDLINE abstracts are substantially different from intervention descriptions (e.g. MEDLINE abstracts are typically semantically divided into blocks such as Objective, Background, Methods), and the goal of the authors is to identify (a few) methodological sentences among an abstract text, while the goal of this task is to identify all

the sentences in an intervention procedure description that detail some surgical action performed.

4.3 Proposed procedural surgical sentences detection methods

In this section, we describe the collected dataset and the proposed methods. Our goal is not to propose a new state-of-the-art method for text classification but to assess whether the automatic classification of procedural knowledge in surgical written texts can be effectively solved with ML or DL text classification techniques.

4.3.1 Dataset

In order to train and test a supervised classification approach to automatically identify procedural sentences, a dataset of sentences labeled as procedural/non-procedural is needed. Given the lack of such a dataset in the literature, we manually constructed and annotated a new dataset, called SPKS (Surgical Procedural Knowledge Sentences)[1] composed by 1,958 sentences (37,022 words - 3999 unique words) from a recent surgical robotics book [114] and from some papers [115, 116, 117]. Different authors produced these documents and they vary significantly in the writing style: the procedure descriptions are essential and schematic in some cases, while longer sentences enriched with background information are used in others. The dataset consists of 20 descriptions of real-world procedures (taken as-is from the sources) from different surgical fields (urological, gynecological, gastrointestinal, and thoracic). Regarding the book [114], we have arbitrarily selected without lack of generality a few (among many) of the sections describing surgical procedures. The complete list of sections used is reported on the corresponding web page. More precisely, we have only annotated those chapters and sections that, given their name (e.g. “Operational steps”), are expected to describe the surgical intervention procedure, leaving out unrelated ones (e.g. “History of Robots and Robotic Surgery”). This is because our goal is to identify the sentences in a procedure description that detail some of the surgical actions performed, automatically cleaning out all those that are not relevant to build a procedural workflow. As we will show later in the dataset statistics, irrelevant sentences account for a substantial amount.

[1] Dataset web-page: `https://gitlab.com/altairLab/spks-dataset`

Each sentence in the selected procedure texts was manually annotated as *procedural* or *non-procedural*. As stated in 2.2.3, it is crucial to reduce labeling ambiguities by providing precise annotation guidelines. Since the same sentence may contain both procedural and non-procedural information, we provide the following definitions:

- *procedural*: a sentence describing at least one action by the robot or the human surgeon, being it an intervention on the body or the positioning of the robot;
- *non-procedural*: a sentence that does not indicate a specific surgeon action but describes anatomical aspects, exceptional events that can occur during surgery, and general indications that are not specific to a single step of the intervention.

To guide the annotation work, we also provided some examples, similar to those reported in Table 1.1. The actual annotation of the 1,958 sentences was performed by a single human annotator (M.Sc. with "C1" English language proficiency) with a 2-year experience in the robotic-surgical domain. The annotation of the whole dataset required approximately 65 working hours for the annotator. As frequently occurs with text classification tasks, the resulting annotated dataset is slightly unbalanced: ~64% of all the sentences are classified as procedural, while the remaining ~36% as non-procedural. Approximately one-third of the sentences in the collected text describing surgical intervention procedures do not describe concrete surgeon actions. Therefore, these sentences are not relevant for deriving the intervention workflow.

As manual annotation is a rather subjective process, performed in our case by a single annotator, in order to assess the general adherence of the annotations produced with respect to the presented guidelines, we performed an inter-annotation agreement analysis: 98 sentences, approximately 5% of the overall dataset, were randomly sampled, respecting the procedural/non-procedural balancing of the dataset, and a second expert (Ph.D. with "C1" English language proficiency, computer science background) was asked to annotate them following the same guidelines. We obtained a Kappa coefficient of 0.93 which, as stated in 2.2.3, indicates an almost perfect level of agreement between the two annotators.

At the end of 2022, we released an enlarged version of SPKS (SPKSv1.1), consisting of 2250 sentences annotated with the same strategies. Of them, ~ 68% are procedural, while ~ 32% are non-procedural. It contains descriptions of 28 robotic-surgery procedures.

4.3.2 Preprocessing the dataset

First, we tested different combinations of text normalization techniques in order to reduce the number of word forms in the original dataset and thus limit noisy features. In particular, we lowercased each word, we replaced each number with a fixed placeholder, we removed punctuation, leading/ending white spaces, and stopwords. We also experimented with combining these techniques with either lemmatization or stemming, but they turned out to be ineffective in our evaluation scenario.

4.3.3 Classifiers

We frame the problem of automatically detecting procedural sentences in surgical intervention texts as a sentence classification task. To better assess the feasibility of our approach, we experimented with and compared the performance of different text classifiers, ranging from classical machine learning to neural network methods and Transformer-based approaches.

Given the reduced size of the dataset, for each model, we applied the nested k*I-fold cross-validation protocol with k=10 and I=k-1=9. That is, the dataset is split into 10 sets. One by one, a set is selected as *test* set to assess the model performance, while the other 9 are used to fit the model (8 sets - a.k.a. *train* set) and determine the best hyperparameters[2] (1 set - a.k.a. *validation* set), until all possible combinations have been evaluated. The model performance is thus the average performance on the 10 test sets of the corresponding model trained and tuned (according to the best hyperparameters) on the remaining 9 sets. This technique ensures no data leakage can occur [119].

We first analyzed some widely used classical ML methods successfully applied for text classification and described in 2.5.2: namely, *Random Forest* (c.f. [120]); *Linear Support Vector Machine* (c.f. [121]); *Multinomial Naïve Bayes* (c.f. [79]) and *Logistic Regression* (c.f. [122]). These classifiers expect numerical feature vectors with a fixed size rather than the raw text of variable length (c.f. 2.3), and therefore sentences have to be appropriately pre-processed. Specifically, for each term of a sentence in our dataset, we calculate the TF-IDF measure described in 2.3.1: each sentence is then represented as a vector, where the components correspond to the most frequent terms of the dataset, and the value in the components is the TF-IDF measure for that term of the sentence. The classifiers are then trained using these vectors as features.

[2] For tuning hyperparameters, we either relied on built-in auto-tune functionalities (c.f., FastText) or the `HyperBand` algorithm [118].

We then decided to test the effectiveness of FastText (described in 2.3.2) for detecting procedural knowledge in written surgical text. In particular, we used the classifier presented in [123], i.e. a multinomial logistic regression method where each input sentence is encoded as a sentence vector, obtained by averaging the FastText word representations of all the words in the sentence. We used it because it has been widely used for numerous text classification tasks, such as mail classification [124] or toxic speech detection [125], and explicit content detection [126].

We also tested some neural-network classifiers, in particular, a 1-Dimension Convolutional Neural-Network (1D-CNN) and the Bi-LSTM that proved to be very effective in many different classification tasks and domains (e.g. [127, 128]). Given the possibility of building the FastText word embeddings separately from the FastText classifier, both neural approaches considered were fed with the same sentence vectors used to train and evaluate the FastText classifier. This also allows us to directly compare the efficient linear classifier implemented in FastText and the more advanced neural approaches.

Finally, we also tested BERT performance, fine-tuning it on the sentence classification task. As explained in 2.4.2, differently from the other word representations, word vectors in BERT are *contextualized*, meaning that the embedding of a word will be different according to the sentence in which it is used. Since BERT has been trained on general domain texts, which are substantially different from the robotic-surgery documents we are working with, we also decided to use ClinicalBERT [97], a language model pre-trained on clinical notes and Electronic Health Records (EHR). While still different from surgical procedure descriptions, these texts are certainly closer to the robotic-assisted surgery domain than those used for training BERT. Finally, in addition to the evaluation reported in [18], we also tested SURGICBERTA and ROBERTA on SPKSv1.1 to measure the benefit of MLM on this downstream task. To fine-tune BERT, ClinicalBERT, ROBERTA, and SURGICBERTA for procedural sentence classification in robotic-assisted surgical texts, we modified the base model to produce a classification output (procedural/non-procedural). This is achieved by adding a classification layer on top of the pre-trained models and then by training the entire model on our annotated dataset until the resulting end-to-end model is well-suited for our task. In detail, we use a single linear layer for the sentence classification part, similar to what is done in [97]. Note that some pre-processing of the dataset has to be performed to use its texts to fine-tune BERT, ClinicalBERT, ROBERTA, and SURGICBERTA, such as word tokenization and index mapping to the tokenizer vocabulary and fixed-length normalization (by truncation or padding) of all texts.

4.4 Results and Discussion

4.4.1 Evaluation on SPKS dataset (v1.0)

Table 4.1: Aggregated classification performance of the tested methods. "[bal]" indicates training on a 50-50 balanced dataset (upsampling).

Method	A	Macro			Weighted		
		P	R	F1	wP	wR	wF1
RandomForest	0.740	0.743	0.678	**0.686**	0.741	0.740	**0.721**
MultinomialNaïveBayes	0.737	0.785	0.655	0.657	0.767	0.737	0.701
LinearSVM	0.723	0.770	0.636	0.633	0.753	0.723	0.681
LogisticRegression	0.694	0.770	0.590	0.562	0.745	0.694	0.626
FastText	0.786	0.771	0.765	0.767	0.784	0.786	0.785
FastText[bal]	0.788	0.773	0.767	**0.770**	0.786	0.788	**0.787**
1D-CNN	0.829	0.816	0.828	0.820	0.835	0.829	0.831
1D-CNN[bal]	0.833	0.819	0.827	**0.823**	0.836	0.833	**0.834**
BiLSTM	0.867	0.857	0.856	0.857	0.867	0.867	0.867
BiLSTM[bal]	0.870	0.862	0.855	**0.859**	0.869	0.870	**0.869**
BERT	0.864	0.859	0.845	0.851	0.863	0.864	0.863
BERT[bal]	0.862	0.859	0.840	0.847	0.861	0.862	0.860
ClinicalBERT	0.872	0.866	0.856	**0.860**	0.871	0.871	**0.871**
ClinicalBERT[bal]	0.866	0.862	0.846	0.853	0.865	0.866	0.865

To address the research questions RQ1 and RQ2 presented in 4.1, we compare the prediction of the various classifiers against some gold annotations (i.e. a set of sentences annotated with a procedural/non-procedural label), using the metrics presented in 2.5.3, in particular the Macro-Averaged Metrics, i.e. Precision (P), Recall (R), F-Score (F1), the Weight-Averaged Metrics, i.e. w-Precision (wP), w-Recall (wR), w-F-Score (wF1); and, Accuracy (A).

In the first four rows of Table 4.1, we report the classification performance of the classical ML algorithms that exploit TF-IDF as features. The considered ML approaches based on TF-IDF have mediocre performance when used to solve this task. This could be due to the unbalanced dataset, which is difficult to handle with standard ML algorithms. Classical ML approaches are often biased towards the majority class (F1 scores

Table 4.2: Classification performance of the tested methods per class. "[bal]" indicates training on a 50-50 balanced dataset (upsampling).

Method	Procedural			Non-Procedural		
	P	R	F1	P	R	F1
RandomForest	0.738	0.913	0.816	0.747	0.443	0.556
MultinomialNaïveBayes	0.717	0.965	0.823	0.852	0.344	0.491
LinearSVM	0.706	0.964	0.815	0.835	0.308	0.450
LogisticRegression	0.678	0.981	0.802	0.861	0.199	0.323
FastText	0.821	0.846	0.833	0.720	0.683	0.701
FastText[bal]	0.824	0.846	0.835	0.722	0.689	0.705
1D-CNN	0.889	0.834	0.861	0.742	0.821	0.780
1D-CNN[bal]	0.881	0.851	0.866	0.758	0.803	0.780
BiLSTM	0.894	0.896	0.895	0.820	0.817	0.818
BiLSTM[bal]	0.887	0.910	0.898	0.837	0.801	0.819
BERT	0.875	0.916	0.895	0.843	0.775	0.808
BERT[bal]	0.867	0.922	0.894	0.850	0.757	0.801
ClinicalBERT	0.886	0.915	0.900	0.845	0.797	0.821
ClinicalBERT[bal]	0.874	0.922	0.897	0.851	0.871	0.809

on the procedural class are substantially higher than on the non-procedural one), not considering the data distribution. In the worst case, minority classes are treated as outliers and ignored. Moreover, TF-IDF cannot account for the similarity between the words in a document since each word is independently presented as an index. Among the considered ML algorithms, Random-Forest obtains the highest F1 scores.

The fifth row of Table 4.1 summarizes the performance of the FastText classifier. All scores demonstrate that FastText obtains much higher classification performance than the best-considered ML method (Ra-Fo). In particular, it improves 10.56% over Macro-F1 and 8.15% over Weighted-F1.

We then fed the FastText word embeddings learned on the dataset to a 1D-CNN and a Bi-LSTM. In our task, the adoption of more complex classification models allows us to substantially improve performance, as confirmed by the seventh and ninth rows of the Table. The 1D-CNN improves the Macro-F1 of 6.46% and the Weighted-F1 of 5.54%, and the Bi-LSTM contributes to improving the Macro-F1 of 10.50% and on the Weighted-F1 of 9.45% with respect to FastText performance. The downside is the computational

time: with the configurations used in the experiments, FastText is 8 times faster than the 1D-CNN and 40 times faster than the Bi-LSTM.

Finally, the eleventh and thirteenth rows of the Table show that it is possible to achieve high classification performance using transformer-based pre-trained language models. In particular, ClinicalBERT performs slightly better than Bi-LSTM (+ 0.12% of Macro-F1 and + 0.22% of Weighted-F1), while BERT performs slightly worse than Bi-LSTM. Computational-wise, fine-tuning transformers-based models on our dataset is 4 times slower than training Bi-LSTM.

We also investigated whether it is possible to boost classification performance by balancing the dataset. More precisely, we have applied standard random over-sampling techniques (i.e. the addition of a random set of copies of the minority class samples to the data) [57] to obtain a perfectly balanced (50% procedural / 50% non-procedural) training material, reassessing classification performance. Given the inadequate performance of classical ML algorithms, we have limited this analysis only to the three approaches that use subword word embeddings as features and to transformers-based methods. As shown in the rows of Table 4.1 tagged with `[bal]`, adopting upsampling techniques does not substantially improve classification results. Indeed, in the case of transformer-based models, balancing the dataset actually has some (limited) detrimental effects. While summarized by the results of Table 4.1, we reported the per-class classification performance for completeness in Table 4.2.

Answer to research questions RQ1 and RQ2

Based on the reported results, we can answer RQ1 by stating that the considered ML methods fed with TF-IDF features do not solve the problem satisfactorily. Using subword-enriched word embeddings fed to neural networks allows for substantially improved results, achieving overall good performance for the considered classification task (Bi-LSTM `wF1` = 0.869 with balancing). Concerning pre-trained language models, a further (marginal) improvement is observed exploiting ClinicalBERT, while fine-tuning the general-domain BERT leads to lower classification performance than Bi-LSTM, showing that, for the considered task, more advanced (yet computationally demanding) techniques do not necessarily produce better results. Overall, the results are satisfactory, confirming the feasibility of automatically detecting procedural sentences in surgical intervention descriptions. We believe there is room for improvement, for example, by enlarging the dataset. Moreover, we cannot positively answer RQ2, as we exper-

imentally observed that the adoption of upsampling techniques on the minority class does not substantially improve the performance for detecting procedural knowledge.

Considerations on the size of the dataset

We finally wondered if a dataset of 1,958 sentences is large enough for the optimal training of learning approaches for this task and if there is room to further improve the results by increasing the number of sentences in the dataset. To answer this question, we studied the evolution of the Macro-F1 when varying the size of the training dataset. Figure 4.1 (left) shows this analysis considering the FastText classifier. The curve tends to flatten out when reaching approximately 800 sentences in the Train dataset, thus possibly suggesting that adding more samples will unlikely yield substantially better performances. Figure 4.1 (center) shows the same analysis considering the Bi-LSTM classifier. The slope of the curve, especially approaching the total size of the training dataset, is constantly increasing and does not flatten out. Despite a less prominent increase rate, a similar trend is obtained for the same analysis on the classifier based on ClinicalBERT, shown in Figure 4.1 (right). These trends somehow suggest that by increasing the number of samples of the dataset, classification performances might be further improved for these two methods.

Interestingly, the Figure also shows that ClinicalBERT's fine-tuning on our dataset works very well, even for very limited-size datasets (F1>0.8 with just 400 samples).

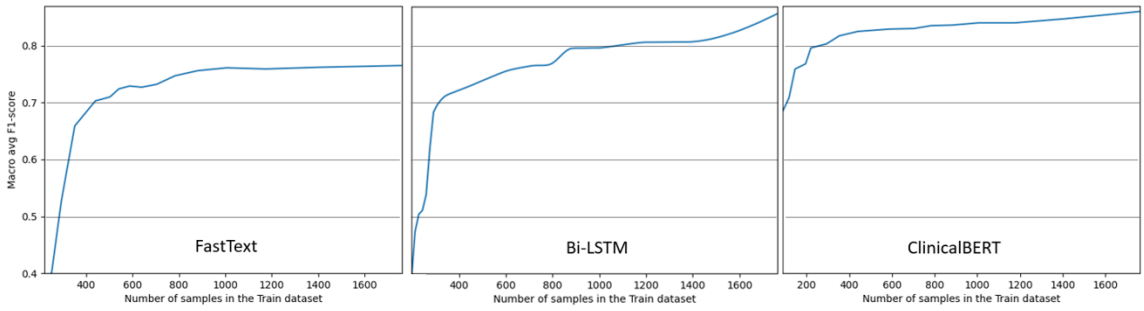


Fig. 4.1: The trend of Macro-F1 of the FastText, Bi-LSTM, and ClinicalBERT classifiers, obtained by varying the number of training samples.

Table 4.3: Aggregated classification performance of the tested methods in the second setting.

Method	A	Macro			Weighted		
		P	R	F1	wP	wR	wF1
ClinicalBERT	0.856	0.840	0.823	0.831	0.854	0.856	0.855
RoBERTa	0.872	0.860	0.841	0.849	0.871	0.872	0.871
SurgicBERTa	**0.886**	**0.880**	**0.853**	**0.864**	**0.885**	**0.886**	**0.884**

Table 4.4: Classification performance of the tested methods per class in the second setting.

Method	Procedural			Non-Procedural		
	P	R	F1	P	R	F1
ClinicalBERT	0.879	0.915	0.897	0.801	0.731	0.764
RoBERTa	0.889	0.928	0.908	0.831	0.753	0.790
SurgicBERTa	**0.894**	**0.945**	**0.919**	**0.865**	**0.762**	**0.810**

4.4.2 Evaluation related to the assessment of SurgicBERTa

To address the research question RQ3 presented in 4.1 and to continue the SurgicBERTa evaluation of Chapter 3, we compare the prediction obtained by SurgicBERTa with that obtained by its vanilla RoBERTa. Since for this evaluation the enlarged version of SPKS was used (SPKSv1.1), the results are not directly comparable to the ones reported in the previous section. To put the scores in perspective with the outcomes of the previous evaluation, we report also the performance of ClinicalBERT (the best performing model in the previous evaluation) on this extended dataset.

The second setting's procedural sentence detection task results have been reported in Tables 4.3 and 4.4, where higher results are in bold. SurgicBERTa improves all the performance metrics compared to RoBERTa and ClinicalBERT on both procedural and non-procedural classes. Overall, averaging the performances on both classes, SurgicBERTa improves the accuracy of 0.014, Macro-F1 of 0.033 and Weighted-F1 of 0.029 when compared with its base version RoBERTa, confirming the benefit of having a domain-specific language for surgical-related text classification. We can thus posi-

tively answer RQ3 because SURGICBERTA, obtained by ROBERTA with the MLM domain adaptation technique, achieves higher performance.

4.5 Conclusions

This chapter aimed to introduce and investigate the problem, never tackled before, of detecting procedural knowledge in written surgical intervention descriptions. In particular, we tested the effectiveness of various ML algorithms operating on TF-IDF features, observing their poor performance. Better scores are achieved using the linear classification algorithm implemented by FastText, which works on subword enriched words-embeddings and finally, using the embeddings returned by `FastText` as the input features of some neural networks (1D-CNN, Bi-LSTM). Finally, using ClinicalBERT to detect procedural sentences in robotic-surgical texts proved to be a good choice. From the experiments, it also emerged that balancing the number of class samples in the training dataset does not lead to a substantial performance boost. The second evaluation continues the extrinsic evaluation of SURGICBERTA presented in Chapter 3, confirming that SURGICBERTA better deals with surgical language-related tasks than its vanilla version ROBERTA.

The goal of this chapter was not to identify the best possible algorithm to tackle this problem nor to identify the highest classification scores achievable. The goal was, indeed, to provide a first assessment of the feasibility of the task using competitive methods. Indeed, we conjecture that the obtained results can still be improved. Concerning the dataset, enlarging it may be beneficial, also in light of the consideration reported toward the end of Section 4.4: to potentially speed up the annotation process, active learning could be worth investigating (i.e. collecting gold annotations by asking human evaluators to accept or correct the sentence classification predicted by the trained model). Furthermore, in this chapter, we tackled the procedural sentence detection task using information solely from the sentence to be classified. The integration of additional context-related (e.g. when a sentence is preceded by another "signaling" sentence or it appears in a bullet/numbered list) is worth investigating, in line with the recent work presented in [22].

This work is a preparatory activity toward extracting structured surgical intervention workflows from written procedural documents, a challenging and, to the best of

our knowledge, never investigated task in the surgical domain, which we address in the following chapters.

5

An annotated resource for procedural knowledge extraction in surgery

> The beginning of wisdom is the definition of terms.
>
> Sentence attributed to Socrates

5.1 Introduction

In the previous chapter, we proposed different methods to detect procedural sentences in as-is textbooks and academic papers. Recalling Figure 1.2, the extracted sentences are then sent to the second stage of the pipeline, dealing with the proper extraction of procedural elements, such as the actions to be performed, the agent executing them, the surgical instruments to use, spatial and temporal constraints to respect while performing a given action, the goal, and so on. The purpose of the second stage of the pipeline is to extract relational information based on actions and actors involved in them. To extract this kind of information, as described in 2.6, Semantic Role Labeling (SRL) techniques [85] have shown to be a promising and viable solution [129, 130]. These methods are based on shallow semantic parsing and produce predicate-argument structures of sentences. In most semantic theories, predicates are verbs, verbal nouns, and other verb forms. They are mainly based on PropBank [56] and FrameNet [131] lexical resources already described in 2.6.2. PropBank-based SRL methods are successfully used in numerous NLP applications, such as conversation analysis [132], video understanding [130], information extraction and ontology population [133], mining of event logs written in natural language [129] or automatic image captioning [134]. However, the performance of the current SRL systems on out-of-domain testing examples is often very poor [135]. This is because PropBank annotations focus

on general-purpose, newswire texts and do not fully cover specific domains, such as, in our case, the surgical one. Furthermore, a critical element in very specialized domains and in particular in the bio-medical one, is the lack of available dataset to train and validate models. Published papers often used private datasets, which are rarely shared, primarily due to patient privacy concerns [136], hindering the replication of the results. The most popular datasets and databases in the bio-medical NLP are MIMIC [101], the ones from i2b2 challenges (e.g. [137] for concept extraction), and the one from SemEval challenges (e.g. [138] for temporal relations extraction from clinical narratives). Unfortunately, none contains annotations of procedural surgical descriptions for semantic information extraction.

With this chapter, we aim to fill this gap and extend PropBank so that its frames are suitable for representing the semantic roles typically required in the procedural surgical domain.

This chapter's main contribution is the public release of a new linguistic resource that extends PropBank with frames describing actions and participants in the robotic-assisted surgical domain and releasing an annotated dataset to train and test automatic models. We named this resource *Robotic Surgery Propositional Bank* (RSPB). This material is essential for adapting SRL methods from other domains to the procedural robotic-surgical one.

5.2 State of the art

Researchers traditionally have built NLP lexical resources targeting general-domain English, which is syntactically and semantically different from domain-specific usage [139] as well as other languages [140, 141, 142]. Therefore, these resources cannot be directly exploited in very specific domains or with other languages, and different methods have been proposed to adapt them to specific needs. This section summarises some works that have adapted the general linguistic resources to a specific domain or languages other than English.

Many works on updating English frame banks have been carried out in various fields, such as the clinical [139, 143], the biomedical [144, 145], and other non-biomedical domains such as software analysis and cooking recipes [146, 147].

[139] has considered texts written in different laparoscopic cholecystectomy operational notes stating that the language is significantly different from general English and

existing semantic resources have limited coverage of the action verbs frequently occurring in operative notes. Based on these observations, the authors have surveyed the usage of each verb in the sample dataset to determine each verb's meanings and semantic arguments. In this way, they have extracted a set of differently used verbs, and, following the PropBank guidelines, they have defined specific frames for them. This work, however, has considered only surgical, non-robotic procedures taken only from gastrointestinal surgery notes that use more schematic language than descriptions taken from textbooks used in our work. Finally, no annotated dataset with these newly defined frames was provided, hindering the possibility of benchmarking available SRL tools on the considered domain.

[143] has annotated clinical narratives with layers of syntactic and semantic labels to facilitate advances in clinical NLP. Following PropBank guidelines, new frames have been defined. Although the dataset deal with a clinical language, this chapter considers a more specialized level, i.e. descriptions of robotic-surgical procedures, a restricted subset of the clinical domain considered by the paper (which includes disorders, physiology, chemicals and groups, and anatomical notions). Unfortunately, the related dataset is no longer freely accessible due to copyright issues [148].

[144] has presented a corpus of PropBank-style annotations for biomedical journal abstracts. The work has analyzed 30 biomedical verbs adding or modifying their meaning starting from general English resources. Then, a semi-automatic method was applied to annotate a collection of MEDLINE abstracts selected from the search results with the following keywords: human, blood cells, and transcription factors. First, predicate candidates were identified; then, an automatic tool was used to produce biomedical semantic roles; finally, the resulting annotations were manually corrected. In [145], a new resource that provides VerbNet-style [83] frames for biomedical verbs is released, together with the presentation of key differences between the general and biomedical domain, and the design choices made to accurately capture the meaning and properties of verbs used in biomedical texts. The conclusion is that leveraging a specialized VerbNet helps systems to improve verb classification and thus to tackle better challenging NLP tasks in biomedicine. The two previous works have dealt with a biomedical language that is still far from the procedural surgical one; moreover, the second one has dealt with VerbNet classes that are quite different from the PropBank frames adopted in this chapter.

Outside the medical domain, [147] has proposed a method for automatically extracting semantic information from software requirements specifications. First, fre-

quent verbs were selected from software requirement specification documents in the e-commerce domain to build the semantic frames for them. Then, selected sentences were annotated for using them as training material to benchmark different machine-learning methods.

[146] has proposed a new annotated dataset for extracting recipe information. The authors have defined ad-hoc entity types (action, food, tool, duration, temperature, condition clause, purpose clause, and others) and relation types following the methodology of PropBank. Then a corpus was annotated and used to benchmark a neural span-based model extracting entities and relationships.

Finally, [149] has applied a transformer-based SRL approach to map legislation from semi-free text to structured manually defined frames composed of fixed semantic roles. The domain is completely different from the one in our paper, but the approach bears some similarities.

Several works also propose PropBank language-specific lexicons for languages other than English, both for specialized or general domains. For example, [150] has performed an SRL task in Tamil Biomedicine texts, extracting domain-specific verbs and related semantic roles. [140, 141, 142] instead have built a general-domain PropBank specific for Turkish, Persian, and Russian, respectively. [151] has stated that despite the availability of SRL resources in different languages, building a single multilingual SRL labeler is almost impractical because of the differences in semantic labels and frame banks. To provide a possible solution to these issues, it has provided a family of auto-generated PropBanks for 23 languages from 8 language families, together with a small set of manually annotated sentences for Polish (100), Portuguese (3779), and English (16622), to enable the construction of SRL models for resource-poor languages by annotating the text in different languages with a layer of universal semantic role labeling annotation.

5.3 Building the Robotic-Surgery Procedural Propositional Bank

The Robotic Surgery Propositional Bank (RSPB) is an extension of PropBank [56] for the robotic-surgical domain. The standard PropBank is described in 2.6.2 and consists of:

- a *framebank*, i.e. a collection of *frames* (a.k.a., *meaning* or *senses*) for *lemmas* denoting predicates (*verbs* or *nominalized verbs*). Frames are specific to a given lemma,

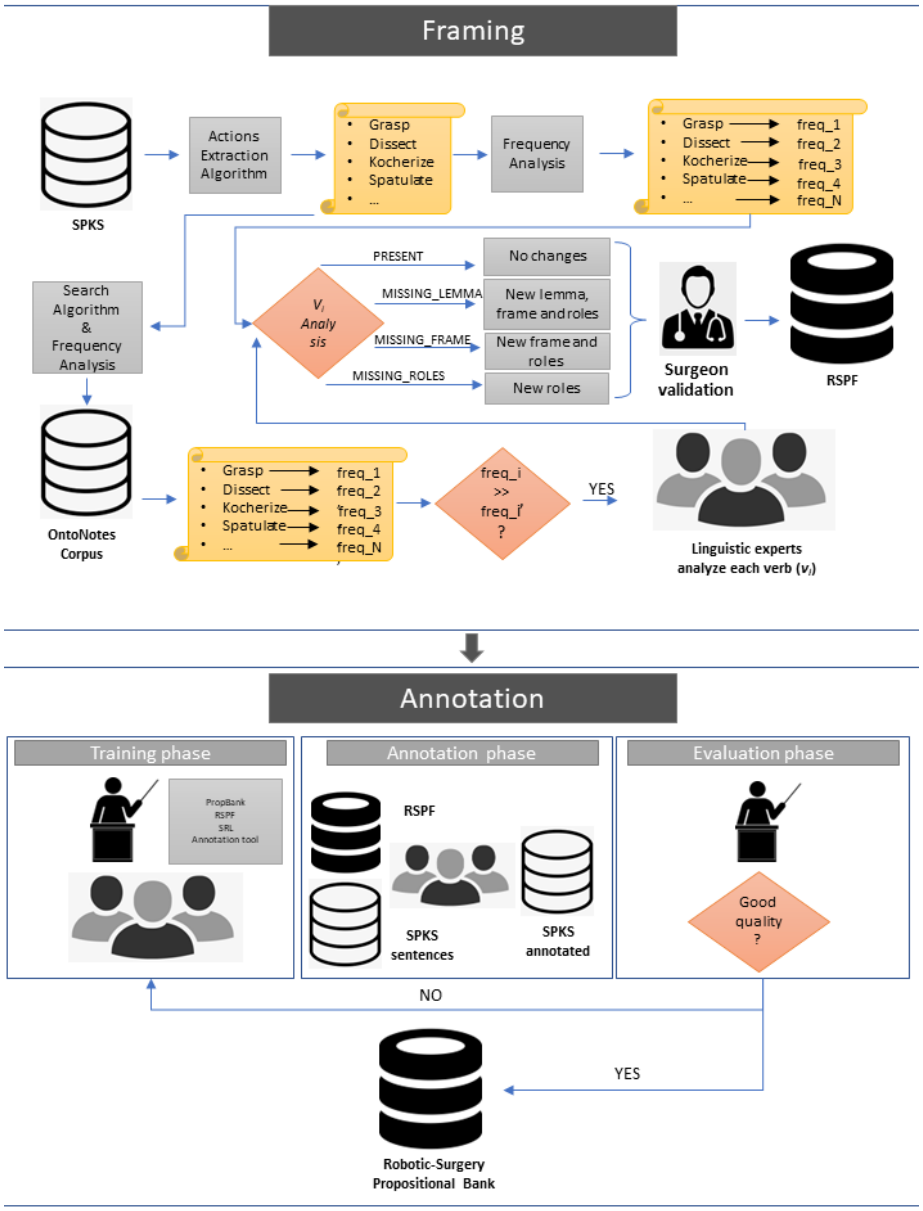


Fig. 5.1: High level diagram of the method described in Section 5.3.1 for the framing of surgical domain verbs and annotation.

and each lemma has one (*mono-sense* lemma) or more (*polysemous* lemma) associated frames. Moreover, each frame specifies its semantic *roles*, i.e. the different la-

bels that can be used to semantically characterize the arguments of the corresponding predicate;
- a corpus of text annotated (according to the framebank) with information about basic semantic propositions.

Following the steps described in [56], also the development of RSPB is divided into two parts, namely the creation of a lexicon of frames files (RSPF, i.e. Robotic Surgery Procedural Framebank) summarised in Section 5.3.1), and the annotated dataset with RSPF's labels, presented in 5.3.2.

Figure 5.1 shows a general overview of both steps: in the domain-verb framing process, some automatic methods extract lemmas describing actions from robotic-assisted surgical texts. How often they appear in the target domain (*freq_i*) is compared to how often they appear in OntoNotes [152] (*freq_i'*). If $freq\_i \gg freq\_i'$, i.e. the ratio between the two is higher than a given threshold, then the respective lemma is sent to a team of human linguistic experts that verify to which of the categories described in Section 5.3.1 the lemma belongs, modifying the corresponding frameset if necessary. The final frameset is validated by a clinician and publicly released. During the annotation step, a team of annotators is hired and trained. Annotation guidelines are written, and sentences to annotate are provided. Then, the annotation process is performed. Quality checks are periodically carried out and, if necessary, the training step is resumed.

5.3.1 The Robotic Surgery Procedural Framebank

Two strategies are applied for identifying procedural verbs and nouns used in the robotic-surgical domain, leveraging the available SPKS corpus. The first one deals with the detection of actions expressed by nominalized verbs, and it is based on keyword extraction. The second method is based on Part-Of-Speech (POS) tagging, and it is used to detect actions expressed by verbs. Their combination, together with additional low-frequency or missing candidates suggested by the clinician during the validation phase, offers broad coverage of the robotic-surgery actions for considered domains.

Adapting PropBank to the robotic-surgical domain

RSPF is an adaptation of PropBank's 3.1 version [56] to the robotic-surgical domain. By analyzing the semantic use of each lemma describing an action identified in the SPKS corpus with respect to the PropBank framebank, each candidate is assigned to one of the four categories described in Table 5.1.

Table 5.1: Categories to which each candidate lemmas is assigned.

Category	Description
PRESENT	The lemma is already present in PropBank, and there is a frame file that adequately describes the use of the predicate. For this lemma, PropBank already describes appropriate semantic roles as core entities.
MISSING_ROLE	The lemma is already present in PropBank, and there is a frame file that adequately describes the use of the predicate. This frame, however, does not include domain-specific semantic roles often used in the robotic-surgical domain.
MISSING_FRAME	The lemma is already present in PropBank, but a proper frame needs to be included, as the existing ones describe different meanings.
MISSING_LEMMA	The lemma is not present in PropBank.

If a lemma is assigned to the PRESENT class, no changes are needed since PropBank already covers the robotic-surgical usage (i.e. there is a frame for the lemma that perfectly describes that usage of the predicate).

If a lemma is assigned to the MISSING_ROLE class, some semantic roles important for the robotic-surgery domain are missing, and therefore they must be added. The lemma *to retract* is an example of an action belonging to this category. For it, PropBank offers the *"retract.01: to take back"* frame, which covers the specific meaning of the surgical domain. However, only two roles are proposed for it:

- **Arg0:** *taker back, agent*
- **Arg1:** *thing retracted*

The verb *to retract* is, however, used very often in the robotic-surgery domain, together with additional information that allows describing the action better: the instrument used for the retraction, the technique and/or manner, and the ending point or the indication of how much to retract.

A candidate lemma may be assigned to the MISSING_FRAME class for two different reasons: i) the usage of the lemma is semantically and entirely different from all the frames covered in PropBank, and there is no overlap between the existing and new semantic roles; ii) the meaning is not entirely new, but the existing frames are too broad to be helpful for the robotic-surgery domain, i.e. the new frame deals with a subset of the meaning captured by (some of) the old ones. An example of the first case is the verb *to*

grasp. For it, PropBank offers a single meaning *"grasp.01: to take hold of, comprehend"* with two semantic roles:

- **Arg0:** *grasper*
- **Arg1:** *thing grasped*

The robotic-surgical domain uses this lemma with a significantly different meaning, i.e. *"to clasp or embrace especially with the fingers or arms"*. For it, important information is also the grasper, the thing grasped, the instrument used for grasping, and important spatial indications for correct grasping. An example of the second case is the verb *to approximate*. For it, PropBank has the frame *"approximate.01: to be close or similar, cause to come near to or approach again"* with only two roles:

- **Arg0:** *entity coming close*
- **Arg1:** *entity coming close to*

It offers a broader meaning than the specialized one used in the robotic-surgery domain (*"to come near in position, to bring near"*), which is typically enriched with the following information: agent, entity coming close, entity coming close to, instrument and spatial indications.

Finally, an example of a lemma of class MISSING_LEMMA is the noun *"kocherization"*. In surgery, it refers to *"an operative maneuver to mobilize the duodenum before performing other procedures locally or before incising the duodenum"*. For it, important information is the agent, and the anatomical entity to be kocherized.

Collecting domain-specific lemmas

To extend PropBank to the procedural robotic-surgical domain, those verbs (or nominalized verbs) that are typical of the surgical domain must be identified. The SPKS dataset, presented in Chapter 4, is used to extract the domain actions of the procedural robotic-surgical domain. For the comparison with general English, we have instead considered the OntoNotes dataset. It is an extensively annotated dataset comprising various text genres such as news, conversational telephone speech, weblogs, usenet newsgroups, broadcast, and talk shows.

The two methods presented below extract domain-specific actions from SPKS. For each domain-specific predicate, it is necessary to check which of the categories described in Table 5.1 the lemma belongs and proceed with framing. Table 5.2 shows 10 examples of actions expressed by nouns identified by the first method and 10 examples

Table 5.2: (Left) Example of nominalized actions extracted using the first method with the indication of the verb they refer to ("—" means missing the corresponding verb) and the modification required. (Right) Example of domain lemmas extracted using the second method with the indication of the type of modification required.

Nominalized actions	Verbs
<Placement , Place , PRESENT >;	<Extraperitonealize , MISSING_LEMMA >
<Reflection , Reflect , MISSING_FRAME >;	<Resect , MISSING_ROLE >
<Retraction , Retract , MISSING_ROLE >;	<Spatulate , MISSING_LEMMA >
<Exposure , Expose , PRESENT >;	<Skeletonize , MISSING_LEMMA >
<Resection , Resect , MISSING_ROLE >;	<Kocherize , MISSING_LEMMA >
<Mobilization ,Mobilize , MISSING_FRAME >;	<Insufflate , MISSING_LEMMA >
<Traction , — , MISSING_LEMMA >;	<Redock , MISSING_LEMMA >
<Administration , Administer , PRESENT >;	<Detubularize , MISSING_LEMMA >
<Identification , Identify , MISSING_FRAME >;	<Grasp , MISSING_FRAME >
<Excision , Excise , MISSING_ROLE >	<Incise , MISSING_ROLE >

of verbs identified by the second method. For each of them, the indication of the type of modification requested on PropBank is reported. Finally, as frequency-based methods for extracting domain terminology may miss some particular terms rarely used in the text (thus ensuring high precision but low recall), the final list of extracted candidate verbs and nominalized verbs were also double-checked by the clinician in the validation phase, for a suggestion of possible missing domain-relevant verbs (and some examples of usage), thus improving the overall coverage of the domain. These additional verbs were then formalized in RSPF following the same framing process described.

Finding frame-evoking nouns

In medical English, actions can be frequently expressed using nouns rather than verbs. Below are two semantically equivalent sentences, where in the first, the concept is expressed using a verb, and in the second using a nominalized verb:

- At this point, the surgeon *sutures* the vein.
- At this point, a *suturation* of the vein is carried out.

For nouns, we addressed the task of domain action detection as a keyword extraction problem, i.e. identifying the lexical entities that best represent the domain according to

a reference corpus. In particular, we have adopted the unsupervised method proposed in [153].

From the output of the algorithm, only nominalized verbs are selected. Since the most common morphological process involved in nominalization is the derivation, which can be defined as the creation of a new lexeme by the addition of an affix (i.e. a bound grammatical morpheme) [154], obtained results are filtered keeping only those words ending with one of the following suffixes: “-sion", “-son", “-tion", “ness", “-ment", “-ery", “-ence", “-ance", “-ure", “-ize", “-ify". False positives are finally removed from the list by manual revision.

Finding frame-evoking verbs

For verbs, a simple approach that compares the frequency of terms used to describe actions between the SPKS and OntoNotes corpora is used. For each token of the domain text, its POS tag [155] and the number of its occurrences are calculated. Only the tokens whose POS tag denotes a verb (i.e. *VB*, *VBP*, *VBZ*, *VBD*, *VBN* or *VBG*) are retained. Lemmatization is then applied, and each token (e.g. *“cauterized"*, *“cauterizes"*, *“cauterizing"*) is associated with the corresponding lemma (resp., *“cauterize"*), aggregating number of occurrences appropriately. For each obtained lemma, the frequency with which it appears in domain sentences is then compared with the one the same lemma appears in OntoNotes. Finally, only those lemmas that are very frequent in the domain sentences and only rarely used in OntoNotes (i.e. in which the ratio between the two frequencies is higher than a given threshold empirically set) are considered as “in domain".

To clarify, this method identifies as “in domain" verbs like *“cauterize"*, *“detubolarize"* and *“extraperitonealize"*, because they are frequent in surgery and rarely used in general English, and therefore the ratio between the frequencies of these verbs in the two domains is very high. On the other hand, the method recognizes verbs such as *“need"*, *“aid"* and *“see"* as general English because they appear in the two corpora with similar frequencies.

Framing of domain-actions

The processes described allow to obtain a list of domain verbs and nominalized verbs associated with a list of SPKS sentences where they are used. Domain experts then an-

```
<roleset id="approximate.02" name="To come near in position, to bring near.">
  <note>Frames file for 'approximate' based on survey of sentences in
    the SPKS corpus.</note>
  <roles>
    <role descr="agent" f="" n="0"></role>
    <role descr="entity coming close" f="" n="1"></role>
    <role descr="entity coming close to" f="" n="2"></role>
    <role descr="instrument" f="" n="3"></role>
    <role descr="other useful spatial indications" f="" n="4"></role>
  </roles>
  <example name="approximate - surgery" src="" type="">
    <text>Next, the musculofascial plate of the rectourethralis is
      approximated to the posterior neobladder, approximately 2-cm
      posterior to the planned urethral aperture, using 3-0
      polyglactin suture.</text>
    <arg f="tmp" n="m">Next</arg>
    <arg f="" n="1">the musculofascial plate of the rectourethralis</arg>
    <rel f="">approximated</rel>
    <arg f="" n="2">to the posterior neobladder</arg>
    <arg f="" n="4">approximately 2-cm posterior to the planned urethral
      aperture</arg>
    <arg f="" n="3">using 3-0 polyglactin suture</arg>
  </example>
</roleset>
```

Fig. 5.2: XML file for the “approximate" lemma. It contains the number of the frame (*02*), with its informal definition (*to come near in position, to bring near*). It then enumerates a list of semantic core roles (numbered from 0 to 4) and provides an annotation example.

alyze each lemma and the respective sentences to understand which of the categories described in Table 5.1 the lemma belongs.

The framing was performed by three linguistic experts with a 3-year of experience in the robotic-surgical domain and validated by a clinician. All frames are collected in XML files. Figure 5.2 is an example of the corresponding XML file for the lemma *approximate*.

In the case MISSING_LEMMA, the lemma is not present in PropBank, and thus it is an unknown word for the resource. Domain experts, therefore, perform the following actions:

(i) they add the new lemma to the resource;
(ii) they add a new frame to the inserted lemma;
(iii) they provide a textual definition of the meaning of that lemma in the surgical domain taken from online medical dictionaries, in particular, Webster Dictionary[1] and The Free Medical Dictionary[2];
(iv) they add appropriate semantic core roles;
(v) they add at least one example of SRL-style annotation for the new frame

In the case MISSING_FRAME, the lemma is already in the resource but with inappropriate frames. In this case, domain experts perform only steps (ii)-(v).

In the case MISSING_ROLE, the lemma is already in the resource with an appropriate frame but with an inappropriate set of core roles. In this case, steps (iv-v) are performed.

Finally, in the case PRESENT, the lemma is already in the resource, with an appropriate frame and core roles. None of the previous steps are performed.

During step (iv), a role is considered as core if arguments playing that role occur with high frequency in the corpus' sentences that use that lemma (i.e. it is present in more than 50% of sentences where the lemma is used)[3] or, independently of its usage in the corpus, if it is considered fundamental by domain experts for interpreting and representing the action.

Framing effort

The framing step is quite expensive because it is carried out manually by personnel who must have expertise both in linguistics (SRL annotations in PropBank style) and in the robotic-surgical domain. The framing step took about 80 hours to be completed.

5.3.2 The Robotic Surgery Procedural Propositional Bank

This section presents the annotation process of sentences from the robotic-surgical domain according to the frames and roles defined in RSPF. SRL is traditionally framed as

[1] `https://www.merriam-webster.com/medical`
[2] `https://medical-dictionary.thefreedictionary.com`
[3] If for a lemma the associated sentences are less than 5, experts are instructed to retrieve additional examples through a web search.

either a dependency-based [156] or a span-based [89] labeling task. Given a predicate in a sentence, the difference between the two settings is in the formalism used to represent its arguments. Span-based SRL requires the identification and classification of the entire textual span of an argument, whereas dependency-based SRL is concerned with labeling only the head of the argument. In the dataset developed in this work, sentences are annotated in span-based fashion.

The team

A team of four people with different roles carried out the annotation process. In more detail, the team is composed by:

- Two annotators. They are bachelor's students of linguistics. During their studies, they have already encountered issues related to the semantic annotation of corpora and successfully passed the relevant exams. However, they never delved into PropBank-style annotation. They have excellent knowledge of the English language (C1 language level) but do not know the medical domain. They were involved in the project with a student collaboration contract of 150 hours each. They were exclusively concerned with the annotation work.
- The project leader. He is a Ph.D. candidate in computer science. He deals with NLP issues applied to medicine. He has the same English language level as the annotators. He was in charge of training, coordinating, and revising the annotation team by answering doubts, refining the guidelines based on annotation errors, and setting up the annotation tool.
- The surgeon. He responded to the doubts collected and presented by the project leader.

The two annotators annotated the total number of the sentences with the following proportions respecting the needs and timing of each: the first one annotated approximately 65% of the sentences while the second the remaining 35%. During the annotation, the project leader revised approximately 1/5 of their annotations to find recurring errors and improve the guidelines accordingly. The annotators processed and labeled a different number of sentences at the same time: this shows that the task, due to the high concentration and the fatigue load, lends itself to being carried out differently according to human characteristics and skills. Due to cost reduction strategy and financial possibilities, the surgeon was just involved in answering doubts instead of having him participate directly in the annotation process.

Text to annotate

The team annotated sentences of different surgical procedures taken from an extended version of the procedural part of the SPKS dataset. The sentences vary significantly in the writing style: the procedure descriptions are essential and schematic in some cases, while longer sentences enriched with background information are used in others. In total, we relied on 1,559 annotated sentences describing 28 surgical procedures of four different robotic-surgery sub-domains. All the sentences are, therefore, procedural in the sense described in [18]. Approximately 80% of the sentences are taken from robotic-surgery textbooks describing how-tos of surgical procedures, while 20% from academic papers or case reports dealing with academic research on surgical procedures or descriptions of real interventions on specific patients.

Training process

Despite having basic knowledge of linguistics and semantic roles annotation, the annotators did not know the PropBank style of annotating text spans. In the first step, during two workshops of one hour each, the project leader introduced the annotators to the project, the ultimate purpose of these annotations, PropBank and PropBank style SRL annotation, and the annotation tool.

At the end of these workshops, the annotators were asked to annotate 15 general English sentences of increasing complexity following the PropBank annotation guidelines. In the end, the annotation was evaluated by the project leader. The process was repeated with new sentences until a 90% inter-annotator agreement with the project leader was reached, following a similar approach to the one presented in [157].

Then, the project leader introduced RSPF to the annotators, focusing on the differences compared to PropBank. The same annotation experiments were conducted, but this time on surgical domain sentences instead of general English ones. Although the annotation guidelines are similar, this experiment was intended to measure the annotators' understanding of the surgical text. The project leader analyzed and discussed the errors of the annotators and refined the guidelines providing them with more explanations to fill the doubts until an 85% inter-annotator agreement with the project leader was reached.[4] Both arguments labeling and the choice of predicate's meaning were evaluated.

[4] We targeted a lower threshold for the agreement to balance the high specificity of the surgical domain and the annotation costs.

Then, during the actual annotation of the whole dataset, the project leader analyzed 20% of the sentences of the two annotators and organized weekly meetings with them to discuss possible mistakes and answer their doubts. The annotators were then asked to revise the labeling if needed be and to double-check the previous annotations in light of the new indications.

At the end of the dataset annotation process, 60 SPKS sentences were assigned to both annotators, which were asked to annotate them in parallel without confronting each other. The inter-annotator agreement on these annotations was finally calculated on predicates and argument labels (score reported and discussed later in this section).

The annotation tool and post-editing technique

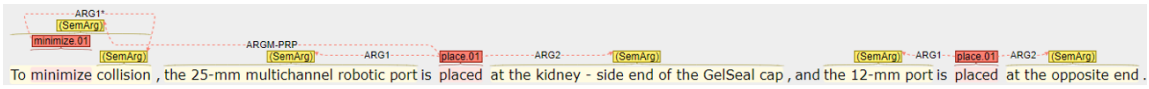
To reduce the annotation effort, a semi-automatic annotation approach was adopted. In a first step, the dataset was processed with a general English span-based SRL tool [158] for automatically obtaining PropBank annotations of the sentences in CoNLL-2012 format.

The annotations thus automatically obtained were uploaded on a server running Inception [61], a tool supporting SRL-style text labeling. Inception has been set up to allow user-friendly SRL annotation of the sentences. The annotators were asked to post-edit and revise the PropBank annotations according to RSPF and the guidelines. That is, instead of having to annotate the sentences from scratch manually, the annotators were asked to revise (i.e. adding missing annotations, deleting incorrect annotations, changing wrong PropBank frames and roles to appropriate RSPF ones) the automatically provided candidate annotations, so to reduce the annotation workload substantially.

Figure 5.3 shows an excerpt of the tool's graphical user interface with an example of annotation and the corresponding content in CoNLL-2012 format, which is directly readable by state-of-the-art SRL methods.

Annotation process and guidelines

The RSPB dataset follows the PropBank style of annotating predicates and semantic arguments (c.f. 2.6.2). Accordingly, similarly to PropBank, our corpus is a collection of sentences with verbs and nominalized verbs annotated with the corresponding frameset in RSPF, together with their related arguments labeled with semantic roles.




```
1_PN.conll 0  0          To   TO -          -  -  -  -  *          *  (ARGM-PRP*          *  -
1_PN.conll 0  1    minimize   VB -  minimize 01  -  -  *       (V*)          *          *  -
1_PN.conll 0  2   collision   NN -          -  -  -  -  *    (ARG1*)          *)         *  -
1_PN.conll 0  3           ,    , -          -  -  -  -  *          *          *          *  -
1_PN.conll 0  4         the   DT -          -  -  -  -  *          *      (ARG1*         *  -
1_PN.conll 0  5      25-mm  NNP -          -  -  -  -  *          *          *          *  -
1_PN.conll 0  6 multichannel  NNP -          -  -  -  -  *          *          *          *  -
1_PN.conll 0  7    robotic   JJ -          -  -  -  -  *          *          *          *  -
1_PN.conll 0  8       port   NN -          -  -  -  -  *          *          *)         *  -
1_PN.conll 0  9         is  VBZ -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 10     placed  VBN -  place 01  -  -  *          *          (V*)       *  -
1_PN.conll 0 11         at   IN -          -  -  -  -  *          *          (ARG2*     *  -
1_PN.conll 0 12        the   DT -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 13     kidney   NN -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 14          -  HYPH -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 15       side   NN -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 16        end   NN -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 17         of   IN -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 18        the   DT -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 19    GelSeal  NNP -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 20        cap   NN -          -  -  -  -  *          *          *)         *  -
1_PN.conll 0 21          ,    , -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 22        and   CC -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 23        the   DT -          -  -  -  -  *          *          *      (ARG1*  -
1_PN.conll 0 24     12-mm  NNP -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 25       port   NN -          -  -  -  -  *          *          *          *) -
1_PN.conll 0 26         is  VBZ -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 27     placed  VBN -  place 01  -  -  *          *          *          (V*)  -
1_PN.conll 0 28         at   IN -          -  -  -  -  *          *          *          (ARG2* -
1_PN.conll 0 29        the   DT -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 30   opposite   JJ -          -  -  -  -  *          *          *          *  -
1_PN.conll 0 31        end   NN -          -  -  -  -  *          *          *          *) -
1_PN.conll 0 32          .    . -          -  -  -  -  *          *          *          *  -
```

Fig. 5.3: On top, annotation example of one sentence through Inception tool graphical interface. In red are predicate annotations, while in yellow are arguments related to the corresponding predicate. The annotation is finally exported in CoNLL-2012 format, and it is directly processable by state-of-the-art SRL tools. Of the CoNLL-2012 fields, only the following columns have been annotated: the 3rd (identification number of the token), the 4th (list of tokens in the sentence), the 7th and 8th (predicate and corresponding frame number), and from the 12th to the second-last containing CoNLL-2012 annotations. In this sentence, three predicates are present: the first (minimize.01) is linked with only one argument (Arg1), the second (place.01) with two (ArgM-PRP, Arg1, and Arg2), and the third (place.01) with two (Arg1 and Arg2) whose meanings are contained in RSPF.

These semantic arguments are labeled according to some predefined categories (e.g. Arg0, Arg1, Arg2) whose specific meaning typically varies according to the predicate considered. The set of roles of each predicate is outlined in the corresponding RSPF frame that gives both semantic and syntactic information about each sense, together with correspondences between the number and semantics. Numbered arguments (e.g. Arg0, Arg1, Arg2) reflect either the arguments that are required for the valency of a predicate (e.g. agent, patient, benefactive) or those that occur with high frequency in actual usage (e.g. instrument, surgical technique, important spatial constraints) as explained in Section 5.3.1. In addition to numbered roles, RSPF also adopts the same modifiers of PropBank (e.g. ArgM-TMP, ArgM-PRP). The annotation of sentences with this information creates a dataset, which is then used as training and testing data in Chapter 6. However, for the annotations to be reliable, following a rigorous annotation process and precise guidelines is necessary (c.f., 2.2.3). Since our corpus is a specialization of PropBank to the surgical domain, it inherits a good part of the annotation guidelines from it.

The main tasks of the Robotic-Surgery Propositional Bank annotation are:

(i) to identify the predicates of the sentence if not already labeled by the automatic tool.
(ii) to choose a sense in RSPF for each predicate or verify if the one automatically assigned is correct;
(iii) to label core arguments for each predicate or verify if the labels automatically assigned are correct.
(iv) to label modifiers arguments if present or verify if the labels automatically assigned to them are correct.

For each sentence, step (i) is related to the predicate-level annotation. The annotators have to check the correctness of the automatically identified predicates and identify missing annotations (i.e. predicates not tagged as such by the automatic tool). If the algorithm has marked as a predicate a token that does not cover this role, it must be removed with all the annotations of the related arguments. This case is relatively rare since state-of-the-art algorithms tend to have a rather high ability to identify predicates. Examples that can sometimes mislead algorithms are those that contain highly-specialized domain expressions such as *running suture*, which in surgery indicates a particular technique for closing the deep portion of surgical defects under moderate tension, while an algorithm not trained in medical language could interpret it as *to run*

verb. Furthermore, the tool we used for automatically generating the candidate annotations also annotated modals and copulas. The annotators were asked to remove them (as verb) since for procedural knowledge extraction, i.e. the extraction of surgical *actions* and semantic arguments linked to them, we deemed them not relevant. Annotations of the modals have been kept however at the modifier argument level (with arguments ArgM-MOD) because they can be helpful to specify the obligatory nature of the corresponding action. Finally, in this step, some nominalized verbs, i.e. nouns that refer to actions, have been annotated as predicates. At this point, step (i) is finished, and annotators continue with step (ii).

Step (ii) is still related to the predicate level. At this point, the annotators have a list of predicates to disambiguate using the corresponding RSPF file. For most of the general English predicates, the automatic tool will have already proposed an appropriate sense which must only be verified by the annotators. If for it RSPF distinguishes two or more verb senses, annotators are asked to choose the one that best suits the context. Sometimes, the process is straightforward because RSPF has only one available sense. This is the case of mono-sense predicates, either specific to the surgical domain (e.g. *skeletonize, detubularize* or *kocherize*) or general English ones (e.g. *accomplish* or *avoid*). In other cases, the disambiguation is more complex because there are multiple senses in RSPF. Cases of this type can be further divided into two sub-categories:

- one of the lemma's senses is specifically used just in the surgical domain, while the other general English senses are rarely used in surgical procedural texts. An example is the lemma *grasp*, for which RSPF has two senses: *grasp.01: "to take hold of, comprehend"* clearly related to a general English usage; *grasp.02: "to clasp or embrace especially with the fingers or arms"* specific to the surgical use. In this case, the disambiguation is typically straightforward, as the general English sense is not or only rarely used in surgical text;
- the lemma has both general English and surgical-specific senses that may both occur in surgical procedural texts. An example is the lemma *follow*, for which RSPF has 9 senses. Although sense 09 *"move behind in the same direction"* has been added for surgical purposes, other general English senses are also used in surgical texts, for example, the 01. *"be subsequent, temporally or spatially"*. The disambiguation in these cases is more complex, and the annotators are asked to reflect well on the meaning of the sentence, comparing it with available examples in RSPF, and to discuss with the project leader if necessary.

During step (ii), occasionally, annotators may come across predicates that do not yet have an existing entry in RSPF.[5] In these cases, annotators are instructed to contact the project leader to describe the situation and report the corresponding dataset's sentence. The project leader analyzes the corresponding sentence and lemma, and then he decides whether to add this lemma to RSPF (because it is a lemma with a surgical sense that was not covered in the initial construction of RSPF) or to ignore the case (when the lemma is only a rarely used surgical slang). The project leader may also consult the surgeon to make an informed decision.

Once the correct meaning of a predicate has been identified, annotators proceed with step (iii), the argument-level annotation. While Arg0 is typically relative to the one who performs the action and Arg1 is typically relative to the one who undergoes it, for the other numbered arguments RSPF has to be checked more carefully. The annotators have to analyze all the arguments (both core and modifier) automatically identified by the tool, as well as possible arguments in the text not annotated by the automatic preprocessing, which are then added from scratch by the annotators. For core arguments, if the annotation label is incorrect, the most appropriate numbering must be inserted. For the arguments automatically labeled as a modifier, the annotators have to check if a more appropriate core role is available in the roleset of the frame and if so, to replace the modifier with it.

Since the tool used for obtaining the first draft of the annotations is trained on general English text, i.e. it does not know the RSPF-specific frames and roles added in the extension of PropBank, the case of spans annotated automatically with a modifier (for PropBank) instead of core role (for RPSF) is quite frequent. Two examples follow:

- In the sentence *"The proximal rectum is grasped using laparoscopic forceps."*, a PropBank-based SRL tool will likely recognize *"using a laparoscopic forceps"* as a generic ArgM-MNR entity while in RSPF, the instrument that should be used to grasp something is labeled as Arg2 of the sense 02 of the lemma *grasp*.
- In the sentence *"The gastric pouch is created using a perigastric technique."*, a PropBank based SRL tool will likely recognize *"using a perigrastric technique"* as a generic

[5] In some cases, this situation may occur due to some lemmatization error of the automatic SRL tool providing the candidate annotations, something that the annotators can easily fix by choosing the correct lemma and sense in RSPF.

ArgM-MNR entity, similarly to the previous example, while in RSPF, the surgical technique is identified with the core role Arg5 of the sense 01 of the lemma *create*.[6]

RSPF contains annotation examples to help annotators. In most cases, choosing a role is straightforward, given the verb-specific definition of the label in the frame files. However, it may be difficult to understand how to annotate a span of very specialized text in some cases. The annotators must decide between the available labels basing either on the explanations/examples provided in RSPF or by searching online for the meaning of unknown domain words. If the doubt persists, the project leader is consulted.

During step (iv), for modifier arguments not to be translated into an RSPF core role according to step (iii), annotators are asked to verify whether the annotations proposed by the automatic tool are consistent with the guidelines of the original PropBank and if not to correct them. RSPF does not add new modifier tags to PropBank, so no changes to its guidelines were necessary for these aspects.

Regarding which token to include in the span of the annotation (c.f., span boundaries) and corresponding exceptions, the same indications as in PropBank's guidelines are given to the annotators.

Inter-annotator Agreement

Agreement between the two annotators has been measured at the end of the process on a sample of 60 sentences annotated by both, using the kappa statistic (c.f., 2.2.3). The kappa statistic has been computed for predicates and arguments, obtaining the values 0.89 and 0.88, respectively. These values denote an almost perfect level of agreement between the annotators, reassuring of the adequacy of the annotation process and guidelines.

Annotation effort

The training and annotation process required a total of 450 hours. Each annotator was employed for 150 hours. In particular, the annotators were asked to annotate for a maximum of 1 hour per session to reduce errors due to fatigue or boredom from the repetitive task. The project leader coordinated the annotation work for another 150 hours.

[6] These two examples show one of the benefits of RSPF over PropBank, for the surgical domain: it allows to better discriminate, with specific core roles, instruments, and techniques, two substantially different entities in the surgical domain, which otherwise will be indistinguishably merged in the ArgM-MNR modifier role in PropBank.

In total, the whole process required 6 months to be carried out. Additional effort was required to set up the annotation tool and write down the guidelines' first version.

5.4 The Robotic-Surgery PropBank

Both the framebank and the dataset resulting from the annotation process described in Sections 5.3.1 and 5.3.2 are publicly available.[7] This section presents and discusses some statistics about them. In more detail, Section 5.4.1 presents RSPF, while 5.4.2 the annotated dataset.

5.4.1 The framebank (RSPF)

Table 5.3: Semantic type of the core roles added to modified lemmas.

Type	Description	Subtype	Number
Who and What	Core-role roles indicating who (or what) performs the action and who (or what) instead undergoes it. Often they respectively coincide with the robotic or the human operator and the anatomical part that is object of the action.	Agent	44
		Patient	46
How	Core-role arguments indicating how the action is performed by specifying the surgical technique or the manner to follow to carry out the action, or the instrument to use.	Manner or technique	36
		Instrument used	30
Spatial information	Core-role arguments specifying different kind of spatial information to know during the execution the corresponding action. These core-roles reply to questions "where?" or "through which passage or port?" or "starting from where?" or "ending where?" or final other frame-specific information such as orientation or spatial constraint to follow for safety reasons.	Where	22
		Through	9
		Starting point	2
		Ending point	4
		Other	32
Purpose	Core-role argument explicitly describing the purpose of the main action. It is inserted as core-role only if very frequently present in our sample sentences.	—	6
Other	Core-roles very specific to a particular lemma and thus not fitting in any of the above classes.	—	13

Using the method described in Section 5.3.1, 252 lemmas have been analyzed. At least one modification among those described in Table 5.1 has been requested in 109 cases. In particular, of the 252 analyzed lemmas, 24 belong to MISSING_LEMMA case, i.e.

[7] `https://gitlab.com/altairLab/robotic-surgery-propositional-bank`

new lemmas (verbs or nouns) that describe very specific actions of the surgical domain not yet present in the original PropBank have been added. 22 lemmas belong to MISSING_FRAME case, i.e. new senses have been added to existing lemmas describing meanings not already covered by PropBank. Finally, 63 lemmas suffer from MISSING_ROLE problem, and thus corresponding existing predicate's sense has been enriched with new semantic roles frequently used in robotic surgery. Considering the new lemmas added, new frames added to existing lemmas, and the new core roles added to existing frames, a total of 244 core roles have been inserted. Table 5.3 shows the semantic type of core roles added for the robotic-surgical domain lemmas. The table considers all core roles added, both in existing frames and in new frames: while the number of core roles added in the first row of the table is quite high, most of them are due to MISSING_LEMMA and MISSING_FRAME, i.e. from frames not yet present in the original PropBank.

The nature of the semantic roles inserted highlights that, in the surgical procedural language, it is of utmost importance to indicate for each action that describes an operation, who or what performs the action (Arg0), the anatomical part that undergoes the action (often Arg1), the instrument with which to perform the action, the surgical technique to adopt, the purpose, and a series of spatial information that helps locate the target anatomy within the human body. Overall, the number of newly introduced and modified lemmas and frames indicates that the extension of PropBank to cover the robotic-surgical domain is substantial and that procedural surgical language differs from general English in terms of both predicates used and the roles required.

5.4.2 The Annotated Dataset

Dataset-level statistics

Following the annotation process and guidelines described in Section 5.3.2, the first annotated dataset specific for SRL of robotic-surgery textbooks was obtained. 28 surgical operative descriptions have been annotated for 1,559 sentences and 32,448 tokens. The obtained dataset is composed of 12,202 annotations. Of them, 3,601 are predicate-level annotations, and 8,601 are argument-level annotations, both core, and modifier. Figure 5.4 shows more detail about this dataset's distribution of core and modifier arguments. A high percentage of the modifier arguments (left side of the figure) in the annotated dataset is covered by TMP. It provides temporal relationships between predicates, and thus, it is helpful to give a chronological order to the actions that must be performed for the correct execution of the robotic-surgical procedure. There are also many tokens

annotated with the MOD label: mostly tokens likes “can", “must", “might", “may", and “would" are annotated in this way. Specifying these arguments is helpful for extracting information on the mandatoriness of surgical actions or events that may occur in certain circumstances. Finally, other frequent modifier arguments are MNR, which enrich the corresponding predicate with generic information about how action should occur, and ADV, which in our dataset primarily identifies the span of texts containing conditional operators (if, then, else, or otherwise). Identifying spans of text tagged with this label is crucial for automatically reconstructing a workflow from text, i.e. to represent the surgical process in a more structured and schematic way, as confirmed in the use-case of Chapter 7. The remaining arguments describe spatial, purposeful, or other information not labeled with any core role.

The most frequent core arguments in the dataset (right side of the figure) is Arg1. Unlike the other core arguments, it has well-defined semantics. It plays the role of patient, i.e. the object that undergoes the action described by the predicate to which it belongs. Also, Arg0 has a well-defined semantic in most verbs (i.e. the agent who performs the action described in the corresponding predicate), but it is not so frequent in this dataset. This observation was also made in [139]: in most cases, the agent did not occur in sample sentences as most actions in procedural language are described in a passive voice, and the agent in operative notes or procedural textbooks that is typically the surgeon, the assistant, or the robot is omitted from the text. There is no well-defined semantics for the core arguments of higher numbers since it varies according to the frame considered. However, Arg2, Arg3, and Arg4 are also frequent and often associated with a surgical instrument, technique, or spatial information.

Procedure-level statistics

As stated before, 28 different robotic-surgery descriptions have been annotated. The average number of sentences per procedure is approximately 56. The shorter description is 10 sentences long, while the longer one comprises 123 sentences. These values are very different from that of other *procedural* descriptions. For example, in [159], a dataset of nano-material synthesis *procedural* descriptions is presented, and they reported 9 sentences per procedure on average. In [160], *procedural* corpora about kitchen and automotive domains were presented, and an average of 12 sentences per description was reported. This means that the robotic-surgical procedures described in

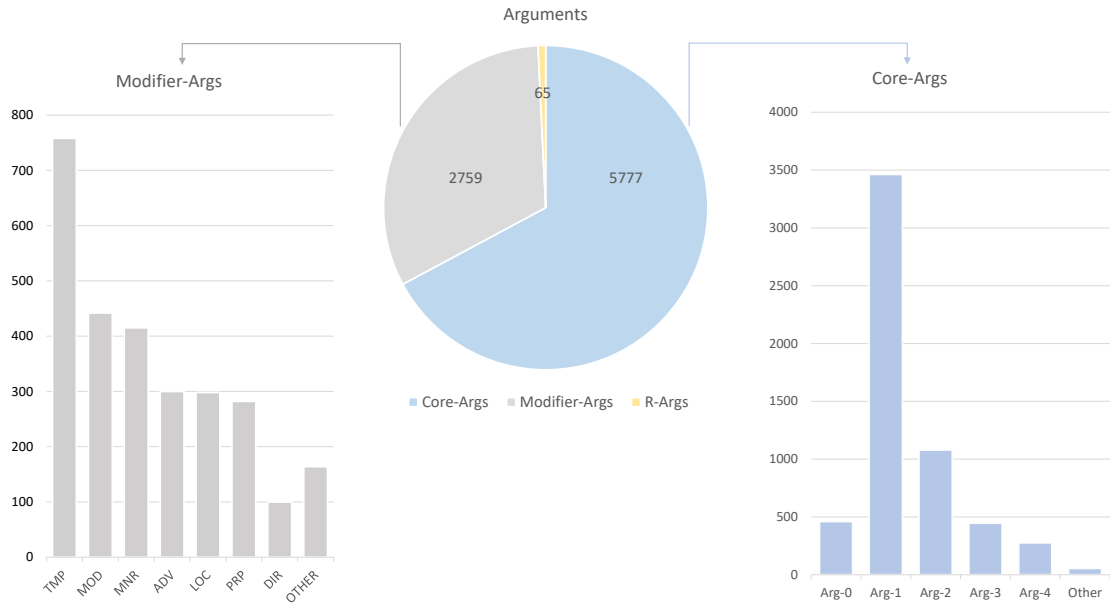


Fig. 5.4: Arguments-level annotations. The pie-chart in the center shows the distribution of semantic arguments between modifier, core and referent (i.e. core argument in cases of co-reference) in our annotated dataset. In total, annotated 5,777 core arguments, 2,759 modifier arguments and 65 referent of other core arguments.

textbooks can be much longer and more detailed than the procedural descriptions of other domains and sources, at least the ones considered so far in the literature.

Finally, our procedures have a mean of approximately 129 predicates (with a minimum of 21 and a maximum of 257) and are composed of a mean of 1,161 tokens (with a minimum of 201 and a maximum of 2,457).

Sentence-level statistics

A sentence of the robotic-surgery procedural domain has 2.31 predicates on average (minimum is 1, and the maximum is 9) and is composed of 5.52 arguments on average (minimum is 1 and the maximum is 20). Finally, it has 20.81 tokens on average (minimum is 5, and the maximum is 81). This last value can be compared with [159], where the authors observed that a sentence for nano-material synthesis has 26 tokens on average, and with [160], where the authors reported that a procedural sentence of the

kitchen or automotive domain is composed of 12 tokens on average. Also, this comparison suggests that depending on the domain, author, source, and purpose, more or less complex *procedural* sentences exist, and those from robotic-surgery textbooks tend to be among the most complex ones.

Predicate-level statistics

In total, this dataset uses 452 different predicate labels. Of them, 410 are used with only one sense, while 42 can have different meanings. In more detail, 100% of MISSING_LEMMA lemmas are mono-sense, meaning that there are not multiple surgical senses for very specialized domain lemmas; furthermore, approximately 70% of MISSING_FRAME lemmas and 85% of MISSING_ROLE lemmas are used with only one sense in our dataset. These statistics show that the procedural surgical language extensively uses mono-sense predicates. Furthermore, even for lemmas with multiple senses available in RSPF, one is typically used much more frequently in the robotic-surgery domain than all the other senses. More in detail, for each predicate p present in the dataset with at least two different meanings, denoting with α_p the frequency of the most common sense for the analyzed predicate with respect to the total number of occurrences in the dataset, we observe that α_p is on average 0.77: that is, the most frequent sense is used in almost 8 times out of 10 of the occurrences of that predicate in the dataset, confirming that also for polysemous RSPF lemmas, only one sense is mainly used in the dataset.

Table 5.4 shows examples of predicates, with the specification of the number of senses with which they appear in the dataset, together with the information on the most frequently used sense and the corresponding percentage of occurrence.

Finally, from a tenses point of view, approximately 56% of annotated predicates are in the passive or past tense, 25% are in a present or imperative tense, 14% in present participle or gerund form, and 5% in a nominalized form.

5.5 Conclusions

This chapter presented the first annotated resource for improving robotic-surgical NLP. The dataset consists of a corpus collecting sentences from textbooks and academic papers describing different robotic-surgical procedures that have been manually annotated in the PropBank-style exploiting an extension of its framebank. In detail, the construction of the dataset followed two steps: in the first one, a framebank specific to the

Table 5.4: Examples of 10 different predicates with the indication of the number of senses with which they appear in the dataset (S-text), the number of senses in the RSPF (S-RSPF), and the reference to the most frequently used sense with the corresponding percentage of occurrence.

Lemma	S-text	S-RSPF	Most frequent sense (% of occurrence)
Follow	4	9	[01] be subsequent, temporally or spatially (60.98)
Come	3	9	[01] motion (60.00)
Pass	3	11	[08] push through a passage (91.18)
Keep	3	6	[04] maintain some prepositional relationship (54.55)
Use	2	3	[01] to take advantage of, utilise (99.67)
Locate	2	2	[01] (cause to) be located in (66.67)
Introduce	2	3	[03] To put or place into something, to insert into (99.92)
Start	2	5	[01] Start (99.94)
Stop	1	3	[01] Stop, putting a stop to (100.00)
Enter	1	2	[01] Enter, go in (100.00)

surgical domain (RSPF) has been defined. In the second step, RSPF was applied to manually annotate sentences, taken without modification, from robotic-surgical texts. The annotation was performed at two levels: predicate level, where predicates are identified and disambiguated with respect to RSPF, and arguments level, where the same tasks are performed for the semantic arguments of each predicate. To perform the annotation, a team of collaborators with different roles has been engaged: two annotators, one project leader, and one clinician for final validation. The annotators were duly trained on PropBank, SRL, and RSPF, with theoretical workshops and an iterative training process. The resulting resource is used in the next Chapter to develop a SRL model specific for the procedural robotic-surgery domain.

6

Extracting procedural knowledge in surgical textbooks

"Who, What, Where, When, With what, Why, How."

The seven circumstances, associated with Hermagoras and Aristotle

6.1 Introduction

As stated in Chapter 1, extracting procedural robotic-surgical knowledge directly from textbooks is an opportunity towards the development of autonomous surgical robots that could automatically build or extend a proper surgical knowledge-base, reasoning with it in realistic intervention scenarios. Also humans could benefit from it for question answering applications, usable for example in an early learning phase by medical students.

In the previous chapter we presented a framebank and the corresponding dataset containing procedural robotic-surgery sentences annotated with semantic roles, named RSPB. In this chapter, we use RSPB to train an SRL model thus proposing a first benchmark on extracting detailed surgical actions from available robotic-surgery procedural textbooks and papers. Exploiting RoBERTa, BioMedRoBERTa and SurgicBERTa (c.f. Chapter 3) pre-trained language models, we first investigate a zero-shot scenario (i.e. the scenario where no additional SRL-annotated domain-specific data is used) and compare the obtained results with a full fine-tuning setting (i.e. the scenario where SRL-annotated domain-specific data is used). In the assessment, we explore different dataset splits (one in-domain and two out-of-domain) and we investigate also the effectiveness of the approach in a few-shot learning scenario (i.e. the scenario where only a portion of the SRL-annotated domain-specific sentences is used for training).

In more detail, we compare all the considered and contributed models in an extensive quantitative evaluation, concretely investigating the following research questions:

- RQ1: How well can general-English and bio-medical pre-trained language models perform SRL on surgical annotated texts without resorting to supervised learning (i.e. zero-shot learning)?
- RQ2: Does fine-tuning on surgical annotated texts substantially improve the performance with respect to the zero-shot setting using off-the-shelf models available in the literature?
- RQ3: How many annotated data are needed to attain substantial improvements via supervised learning for this task (i.e. few-shot learning)?
- RQ4: Does further unsupervised learning of pre-trained language models (as in SURGICBERTA) help to better understand surgical language?
- RQ5: Are the SRL models able to generalize over different surgical sub-domains?

Besides exploiting the standard evaluation measures for the SRL task, we also propose a new way for evaluating SRL systems, based on the joint disambiguation of arguments and predicates, i.e. on the correct disambiguation of semantic arguments with respect to the correct framing of the actual predicate. Results show that the fine-tuning of SURGICBERTA on the SRL task allows to achieve the highest performance on all splits and on all sub-tasks.

6.2 State of the art

While the field of biomedical NLP has a long history – see, among others, [161] for an overview and the proceedings of the long-standing ACL Workshop on Biomedical Language Processing [162] for up-to-date contributions – to the best of our knowledge, no works have tackled so far the problem of extracting procedural knowledge from surgical books or academic papers.

Nevertheless, the literature includes various approaches for extracting relevant information from medical or surgical operative notes using NLP or extracting procedural information from other non-surgical domains. Consequently, this section overviews relevant previous works in two different related areas: the first part discusses recent relevant applications of NLP techniques to the bio-medical and surgical domains; the second part presents papers dealing with the extraction of procedural knowledge from texts, considering also domains other than the bio-medical one.

Application of NLP techniques to bio-medical domains.

This paragraph summarises some recent and relevant applications of NLP techniques to bio-medical and surgical domains. In [163], the authors use logistic regression with unigrams and unique concept identifiers from the unified medical language system to automatically predict the severity of chest injury after trauma from clinical notes. [164] proposes rule-based NLP algorithms to automatically extract surgery-specific data elements (category of knee arthroplasty, laterality, constraint type, whether patella resurfacing was performed or not, and implant model numbers) from knee arthroplasty operative notes: the main objective was to decrease the need for costly manual chart review and to improve data quality using NLP techniques. In [165], they use information extraction techniques applied to operative notes to detect the presence of variables associated with periprosthetic joint infection, including the growth of cultured organisms, documentation of inflammation, presence of sinus tract, and purulence. In [166], the authors use an extreme gradient boosting NLP machine learning algorithm [167] for automated detection of incidental durotomies in free-text operative notes of patients undergoing lumbar spine surgery. The clinical goal is to automatically survey the incidental durotomy that could be potential implications for postoperative recovery, patient-reported outcomes, length of stay, and costs. In [113], the authors address the detection of procedural knowledge in MEDLINE abstracts. In their work, procedural knowledge is defined as a set of *unit procedures* (each consisting of a *Target*, *Action*, and *Method*) organized for solving a specific purpose. The proposed solution works in two steps. First, support vector machines and conditional random fields are combined for detecting sentences (purpose/solution) that may contain unit procedures, feeding them with content (unigrams and bigrams), position (sentence number in the abstract), neighbor (content features of nearby sentences) and ontological features (usage of terms from reference vocabularies). Then, sequence labeling with CRFs is performed to identify the components of unit procedures. In [168], the authors propose an NLP approach to automatically label right ventricular dysfunction size, and the function [169] from echocardiographic free text reports. In particular, manually annotated written reports were used to fine-tune a 12-layer BERT model pre-trained on a large dataset. The remaining written reports are used as test material. The extracted labels are finally used to annotate image data, training a 4-layer 3D convolutional neural network. In [170], NLP is used for adverse event detection from radiology reports and follow-up telephone call notes. In particular, hip dislocation after a primary total hip replacement

[171] is used as a case study. Radiology reports are manually labeled into three categories (current dislocation, evidence of previous dislocation, and no dislocation). In comparison, telephone notes are organized into two categories (evidence of previous dislocation and no dislocation). Then, the performance of different machine learning and deep learning models is compared. In [172] is observed that textual radiology reports contain relevant information for determining the likelihood of radiology signs of COVID-19 in the lungs. Machine Learning NLP approaches and SNOMED-CT reference terminology [173] are thus adopted to detect COVID-19-related disorders within radiology reports automatically.

These studies are examples of NLP applications in the medical domain. However, the texts' typology differs remarkably from ours: they mainly analyze medical notes, often written in highly structured language or abstracts, while we analyze free-text technical manuals or papers. Finally, the purpose is different: our goal is to lay the foundations for extracting a synthetic workflow by mining descriptions of surgical procedures abundantly available in the literature, while theirs is mainly focused on helping surgeons or assistants to analyze available data.

Procedural knowledge extraction.

More similar in terms of the overarching goal but more diverse in the application domain are the studies that, similarly to our work, propose approaches for extracting procedural knowledge for domains other than the biomedical one. In Chapter 4 we already presented some of these papers for the related problem of procedural sentence detection, but this paragraph explains their contributions to the procedural knowledge extraction one. In [22], the authors tackle the problem of procedural knowledge extraction in technical documentation as a multi-class classification task using Support Vector Machine with linguistic and structural features, but they do not extract sentence-level procedural entities, such as words expressing actions, agents, or instruments. The authors of [23] address the mining of cooking recipes and maintenance manuals, formalizing the task into the multi-grained text classification task: first, they detect procedural sentences, then they recognize their semantics (procedure begins or ends and successive, optional and concurrency relations), and finally they assign semantic roles (only action's executor, action name and direct object are considered) to words in a procedural sentence. They adopted a deep learning model that encodes BERT word vectors extracted from input sentences using a BI-LSTM to capture inherent clues in a sentence and a CNN to capture local *n*gram features. A multi-layer perceptron module

is finally used to perform the word-level predictions. Recipe for nanomaterials' synthesis has been mined in [24], where, after having classified each sentence as relevant or non-relevant, they adopt a rule-based parser to extract recipes, i.e. a set of specific actions that are applied to a set of recognized base materials during the synthesis of nanomaterials. In [25], the authors address the problem of extracting repair instructions in user-generated text from automotive web communities. In particular, they use *n*grams, domain-specific lexical features (e.g. text length, readability index, occurrences of enumerations and URLs), and syntactical features to feed several machine-learning methods. Their goal is to classify texts as containing repair instruction or not, and thus they do not deal with sentence-level procedural entity extraction. In [26], the authors extract procedural information in technical support documentation, where procedures are typically described using lists. They aim at extracting decision points within procedures, identifying blocks of instructions corresponding to these decision points, and mapping instructions within a decision block. To do it, they developed a manually annotated dataset and exploited parse-tree-based syntactical rules. Also, the authors of [27] address the extraction of procedural knowledge from structured instructional texts, exploiting finite-state grammars. In particular, they aimed to extract procedural entities such as conditions, actions verbs, agents, instruments, and temporal or spatial parameters. Furthermore, recent deep-learning-based NLP techniques have recently been applied to extract business processes from Standard Operating Procedure documents [28].

All these works address the extraction of procedural knowledge from written text and are thus similar to our foreseen application, They, however, deal with typologies of textual content substantially different from the description of a surgical procedure. Troubleshooting and product documentation, cocking recipes, maintenance manuals, and repair instructions differ significantly from descriptions of surgical procedures. They are different both from the terminological and structural points of view.

6.3 Method

We framed extracting procedural surgical knowledge from the text as an SRL problem since SRL is also applied for information extraction in various biomedical domains [174, 175, 176]. The related theory about SRL is described in 2.6. In this chapter, we use the PropBank-based SRL, exploiting the Robotic-Surgery Propositional Bank described in Chapter 5.

Figure 6.1 summarises the proposed approach.

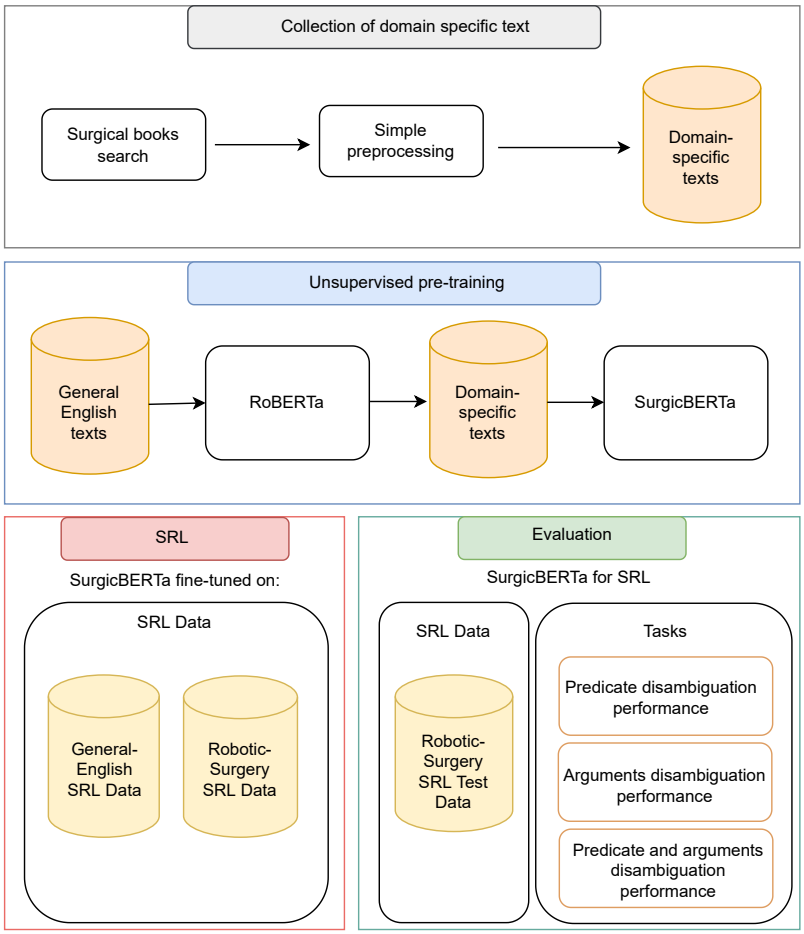


Fig. 6.1: Overview of our approach for procedural surgical knowledge extraction. The pipeline is composed of three stages (top to bottom): (i) the collection of surgical texts from the web and a simple pre-processing (grey box - Chapter 3). (ii) Using the data from the grey box, we adapt the RoBERTa pre-trained language model to the surgical domain, obtaining SurgicBERTa (right part of the blue box - Chapter 3); (iii) We then fine-tune SurgicBERTa in a supervised way on the downstream SRL task using general-English and RSPB datasets (red box - this chapter); SurgicBERTa thus learns the surgical SRL task. (iv) With the performance evaluation step (green box), we evaluate the obtained model on a further test dataset consisting of SRL-style annotated surgical sentences (this chapter).

We recall that the typical SRL task is composed of two sub-tasks. The first is the *predicate identification and disambiguation* task. It is aimed to identify each predicate in a sentence, assigning it the appropriate meaning (i.e. sense in RSPF) in the given context, among the available ones for that predicate lemma codified in the target lexical resource. The second is the *argument identification and classification* task. It aims to detect the argument spans or syntactic argument heads of a predicate and assign them the appropriate semantic role labels according to the target lexical resource.

As an example in the robotic-surgery domain, consider the following sentence, focusing on the verb *"grasp"*:

> *"Using the cadiere grasper (robot arm #3), grasp the soft tissues along the lesser curvature of the stomach to straighten out the lga perpendicular to the celiac axis."*

In the predicate identification and disambiguation phase, *"grasp"* is recognized as a predicate, assigning it the RSPF meaning of *grasp.02*: *"to clasp or embrace especially with the fingers or arms"*, rather than *grasp.01*: *"to take hold of, comprehend"*. Then, in the argument identification and classification phase, SRL produces the following output:

> "[**Arg2**: Using the cadiere grasper (robot arm #3)], [**grasp.02** grasp] [**Arg1**: the soft tissues] [**Arg3**: along the lesser curvature of the stomach to straighten out the lga perpendicular to the celiac axis]."

where, for the sense *grasp.02* in RSPF, Arg2 represents the *"instrument used for grasping"*, Arg1 is the *"thing grasped"*, and Arg3 identifies an *"important spatial indication for correct grasping"*.

Another example for the verb *"dissect"* follows:

> *"The lymphatic tissue is dissected off with meticulous hemostatic and lymphatic control, using bipolar electrocautery and hem-o-lok®clips, to improve visualization."*

is annotated as:

> "[**Arg-1**: The lymphatic tissue] is [**dissect.02** dissected] off [**Arg-3:** with meticulous hemostatic and lymphatic control], [**Arg-2:** using bipolar electrocautery and hem-o-lok® clips], [**ArgM-PRP**: to improve visualization.]"

where Arg-1 is the entity dissected, Arg-3 is the surgical technique, Arg-2 is the instrument, and ArgM-PRP is the modifier role for purpose.

6.3.1 The SRL neural architecture adopted

As stated in 2.6.3, SRL is traditionally performed with data-driven methods. Since recent approaches leverage self-attention techniques [87] and Transformer-based architectures with pre-trained language models [88], in this work, we also follow this trend and adopt a neural approach, thus addressing the SRL task in an end-to-end fashion while testing different pre-trained language models. The pre-trained language models considered in this chapter are the state-of-the-art RoBERTa (described in 2.4.2), BioMedRoBERTa [177] and SurgicBERTa, the one we contributed in Chapter 3. In particular, BioMedRoBERTa is obtained from RoBERTa via continuous pre-training on 2.68M full-text biomedical papers from S2ORC [178]. This amounts to 7.55B tokens and 47GB of data. With this configuration, we want to implicitly verify if the biomedical domain is similar to the surgical one and if we can obtain performance improvements by adopting a more accurate pre-trained language model than the general domain RoBERTa. All pre-trained language models use the same transformer-based architecture [33] and are trained with an MLM objective.

The word representations learned in the pre-trained models have then been reused for the SRL task through fine-tuning. In more detail, the SRL models used in this chapter are instantiated on top of the RoBERTa encoder (the same also used by BioMedRoBERTa and SurgicBERTa). At its core, the system is a standard BIO tagger whose objective is to assign a label of the form `B-X` (beginning of argument with role X), `I-X` (continuing of argument with role X) or `O` (not an argument) to the tokens of the sentence, with respect to the considered predicate. Figure 6.2 illustrates the neural architecture we use. First, we encode the input text using contextualized word embeddings for each token using the pre-trained language model; we then use linear transformations of the word embeddings to obtain a concatenated input for a two-layer ReLU, which is next input to a linear layer followed by softmax activation to produce a probability distribution over labels for each word (to avoid overfitting, a standard dropout layer [179] with probability 0.5 is used). To capture the sequential dependencies between labels, we use a standard CRF layer [180] to produce at testing the most probable label sequence using standard Viterbi decoding.

For training and validation, the CoNLL-2012 dataset [90], a large-scale ($\sim 318k$ annotated SRL predicates), multi-genre general-English corpus, is used to train and validate the *"zero-shot"* models, while RSPB dataset is used in combination with CoNLL-2012 for

the *"few shot"* and *"full fine-tuning"* models that will be described later. RSPB contains four different robotic-surgery domains:

- *Urology* - 51.51% of the sentences;
- *Gastrointestinal procedures* - 24.82% of the sentences;
- *Thoracic procedures* - 13.02% of the sentences;
- *Gynecology* - 10.65% of the sentences.

We therefore used the CoNLL-2012 dataset to make the architecture learn the standard SRL task and RSPB to specialize the model to understand better the surgical language and to perform the SRL task in the given surgical sub-domains.

Both datasets (CoNLL-2012 and RSPB) adhere to the PropBank way of annotating predicates and semantic arguments. Evaluation is carried out on different test splits of the robotic-surgery annotated dataset, detailed next in Section 6.3.2. Sentences of the test sets were never seen during the training and validation phase.

In all experiments, we inform the model about the tokens playing the predicate's role. Differently from [88], we do not use the gold frame sense since our purpose is also to evaluate the model's ability to disambiguate the predicate meaning correctly. The predicate disambiguation adopts a similar architecture.

6.3.2 Splits of the robotic-surgery annotated dataset

Due to the high computational costs needed for training, validating and testing the SRL models, we adopted the classical evaluation protocol of manually splitting the RSPB dataset into three components (train, validation, and test) instead of following a more computationally demanding cross-validation protocol. More in detail, we split the robotic-surgery annotated dataset into three different combinations:

- **BAL**: the split train-test-validation is balanced between different surgical domains. The procedures are split into train-test-validation, preserving the number of sentences per domain (thoracic, gynecological, urological, gastrointestinal). Then, 80% of sentences are used to train (10% of them are removed and used to validate the dataset) and 20% as a test dataset. A similar approach is also used by [56].
- **GYN**: train and validation datasets contain all the sentences of thoracic, gastrointestinal, and urological descriptions. The test dataset contains only sentences of the gynecological domain. No sentences describing gynecological surgeries were seen during the training and validation steps.

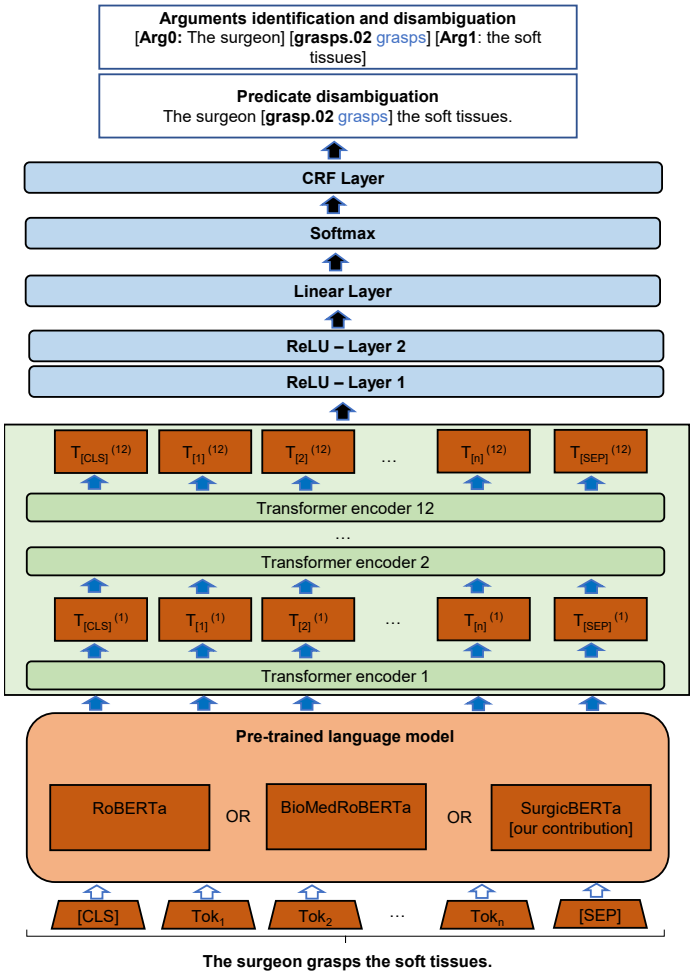


Fig. 6.2: The neural architecture used for SRL. Sentences are tokenized and each token is input to a pre-trained language model to produce a contextualized representation, which is then fed into ReLU layers and a linear layer. Next, a softmax layer produces a probability distribution over the labels. A CRF layer finally captures dependencies between labels by decoding the resulting representations into the most probable label sequence.

- **THO**: train and validation datasets contain all the sentences of gynecological, gastrointestinal, and urological descriptions. The test dataset contains only sentences of the thoracic domain. No sentences describing thoracic surgeries were seen during the training and validation steps.

The BAL split wants to investigate the ability of the method to learn a general surgical domain procedural language from a limited set of annotated sentences. The GYN and THO splits[1] aim instead to verify if the annotations are general across different surgical sub-domains, i.e. if the models trained on them perform comparably with the one trained with the BAL split. Table 6.1 summarizes some statistics about the splits.

Table 6.1: Statistics of the different splits. The numbers outside the parenthesis represent the percentage of the corresponding semantic argument in the respective train + validation or test datasets: the sum by columns of the numbers outside the parenthesis is 100. The parenthesis numbers represent the corresponding argument's split in the train + validation and test dataset. For each argument, the sum of the number in the parenthesis of train + validation and test datasets is 100. The same is for predicates (`Preds` in table).

Split	BAL		THO		GYN	
Arg	Train+Val (%)	Test (%)	Train+Val (%)	Test (%)	Train+Val (%)	Test (%)
PREDS	(80.20)	(19.80)	(87.46)	(12.54)	(89.28)	(10.71)
ARG-0	5.50 (81.90)	4.96 (18.10)	5.15 (81.74)	7.64 (18.26)	5.40 (87.26)	6.01 (12.74)
ARG-1	40.48 (80.21)	40.71 (19.79)	41.84 (87.22)	40.66 (12.78)	42.03 (89.31)	38.34 (10.69)
ARG-2	13.18 (83.56)	10.56 (16.44)	12.96 (87.52)	12.26 (12.48)	13.05 (89.87)	11.21 (10.13)
ARG-3	4.94 (75.95)	6.37 (24.05)	5.50 (87.26)	5.34 (12.74)	5.69 (91.93)	3.80 (8.07)
ARG-4	2.81 (69.29)	5.07 (30.71)	3.67 (90.43)	2.58 (9.57)	3.72 (93.40)	2.00 (6.60)
ARG-5	0.54 (82.22)	0.47 (17.78)	0.55 (89.13)	0.44 (10.87)	0.49 (80.43)	0.90 (19.57)
ARG-6	0.07 (55.56)	0.24 (44.44)	0.13 (0.90)	0.09 (0.10)	0.11 (0.89)	0.10 (0.11)
ARGM	32.48 (80.71)	31.62 (19.29)	30.20 (86.60)	30.99 (13.40)	29.51 (85.66)	37.64 (14.34)

6.3.3 Fine-tuning of language models on the SRL downstream task

Using the neural architecture and the language models of Section 6.3.1, the general-English CoNLL-2012, and RSPB, we trained 18 different models, six for each split. In par-

[1] Although any of the sub-domains in the SPKS dataset could have been chosen as a test set while training on the others, given the relatively small size of the robotic-surgery annotated dataset, we opted for testing on these two domains as they are the smaller ones and thus maximize the size of the available material (from the other domains) used for training.

ticular, for each one, we fine-tuned RoBERTa, BioMedRoBERTa, and SurgicBERTa on two different scenarios:

- **Zero-Shot**: we fine-tuned the language models only on CoNLL-2012 annotated data (train and validation sets), i.e. on non-surgical data. We then evaluated the obtained models on the surgical test set for the various splits;
- **Full Fine-Tuning**: starting from the fine-tuned models of the Zero-Shot scenario, we continued to fine-tune them on the train and validation sets for the different splits of the robotic-surgery annotated dataset. We then evaluated the resulting models on the corresponding surgical test set according to the split, the same used in the Zero-Shot scenario.

Transformer-based language models are known for their capability to achieve high scores also when fine-tuned with a limited amount of task-specific training material (**Few-Shot learning** [181]). This capability is beneficial in situations of a scarcity of annotated data due to few resources or costly content annotation, such as the robotic-surgical one. We thus decided to run some experiments to assess whether this also holds for the surgical SRL task. We created various subsets of the train and validation splits for the BAL scenario of the robotic-surgical annotated dataset, having a number of sentences that are respectively of 0%, 1%, 5%, 10%, 25%, 50% and 100% of the initial training and validation sets. We then trained the SurgicBERTa model on these different subsets, validating them on the same reference test set.

Following the guidelines provided by the authors of [88], we performed the fine-tuning of the models on the downstream SRL task in two stages, with the following suggested configurations:

- stage 1: fine-tuning using cross-entropy loss for 30 epochs with learning rate 3×10^{-5};
- stage 2: further fine-tuning using the combined loss for additional 5 epochs with a lower learning rate (1×10^{-5}.)

Details on the loss functions used can be found in [88].

6.3.4 Evaluation methodology

Performance is evaluated according to three different dimensions:

- *argument identification and disambiguation*: the capability of assigning the correct semantic role label to the predicate arguments mentioned in the text, after

identifying it. This is the traditional dimension used for benchmarking SRL tools [84, 88, 182], adopted also in the CoNLL-2012 Shared Task evaluation (and corresponding script);

- *predicate disambiguation*: the capability of assigning the correct RSPF frame (i.e. meaning) to the predicate in the text. In our domain setting, this evaluation is particularly useful to assess if the models are capable to discriminate the domain-specific usage of some verbs with respect to their general-English usage;
- *predicate-argument disambiguation*: the capability of assigning the correct semantic role label to the predicate arguments as well as to assign the correct sense (i.e. frame) to the corresponding predicate.

The first two dimensions correspond to the two standard SRL sub-tasks, while the third one aims at combining the correctness of both dimensions. To the best of our knowledge, the assessment of this combined predicate-argument disambiguation performance was not addressed in previous works and evaluation campaigns, although we deem it particularly relevant for assessing SRL performance, especially for Propbank-style annotations: indeed, as arguments are defined in RSPF (and PropBank) according to predicate senses (i.e. different senses of the same predicate have different semantic roles), if a tool correctly predicts the role label (e.g. Arg-1) for the argument but fails to disambiguate the sense of the corresponding predicate (e.g. proposing dissect.02 instead of correct dissect.01), it fails in predicting the actual semantic arguments for that predicate, as it predicted a semantic role but for a different predicate sense. Note that these cases are not handled by the standard CoNLL-2012 argument disambiguation, for which the role assigned to an argument is correct independently of the disambiguated sense of the corresponding predicate.

In practice, the evaluation compares the annotations made on the sentence with the gold ones. Namely, for each token of the sentence, the predicted annotation is compared with the gold one. For the first dimension, only the labels of the arguments are considered, while in the second dimension, only the labels of predicates are used. Finally, for the third dimension, the comparison is performed on enriched labels derived from the raw ones as follows: the argument label on each token (both gold and predicted) is concatenated with the label of the corresponding predicate sense so that the same annotation contains both information on the role of the argument and the predicate sense to which that role refers. Then, for each dimension, performance is com-

puted with standard metrics for classification tasks, i.e. precision, recall, and F1-score described in 2.6.4.

6.3.5 Computational aspects

All models are computed using one NVIDIA RTX A6000 GPU, with 48 GB of GPU memory, with the (one-time) MLM training required for building SURGICBERTA taking approximately 8 hours.

Since the compared models share the same SRL neural architecture and vary in the language model used, we observed no significant difference in the time required for fine-tuning them on the annotated dataset. Indeed, each model has required approximately 20 hours for this step. Although the training time is substantial, once the models have been trained, getting the annotations automatically on the test sentences is extremely fast, taking approximately 15 seconds on the largest test split, consisting of approximately 400 sentences (i.e. roughly 0.04 seconds per sentence): exploiting already available models, the extraction of surgical actions and related semantic information from a sentence is almost instantaneous.

6.4 Results

In this section, we report and discuss the results obtained using the methods described in Section 6.3. Each score reported in the section is the average over three distinct runs of the considered method.

6.4.1 Argument disambiguation

We first evaluate the obtained models on the traditional argument disambiguation task. The results are reported in Table 6.2.

The results show that having annotated domain data available is essential to improve the arguments' disambiguation performance. In fact, fine-tuning the language models with some domain data allows us to significantly increase considered metrics on all splits. By focusing on the F1 metric of the BAL split, moving from a zero-shot scenario to a full fine-tuning one, we improve the performance of 0.061 for ROBERTA, of 0.065 for BIOMEDROBERTA and of 0.063 for SURGICBERTA. Similar considerations hold for precision (ROBERTA +0.057; BIOMEDROBERTA +0.061 and SURGICBERTA +0.054) and

Table 6.2: Performance (overall) on the arguments-disambiguation task for BAL, THO and GYN splits. FFT means *Full Fine-Tuning* scenario, while ZS stands for *Zero-Shot* scenario. The best scores are highlighted in bold.

SPLIT	BAL			THO			GYN		
MODEL	P	R	F1	P	R	F1	P	R	F1
RoBERTa$_{ZS}$	0.714	0.688	0.701	0.692	0.677	0.685	0.775	0.767	0.771
BioMedRoBERTa$_{ZS}$	0.718	**0.696**	0.707	0.708	0.684	0.696	0.788	0.777	**0.782**
SurgicBERTa$_{ZS}$	**0.724**	**0.696**	**0.710**	**0.726**	**0.700**	**0.713**	**0.827**	**0.781**	0.775
RoBERTa$_{FFT}$	0.771	0.752	0.762	0.753	0.744	0.748	0.799	0.781	0.790
BioMedRoBERTa$_{FFT}$	**0.779**	0.764	0.772	0.756	0.738	0.747	0.798	0.794	0.796
SurgicBERTa$_{FFT}$	0.778	**0.768**	**0.773**	**0.759**	**0.749**	**0.753**	**0.813**	**0.796**	**0.804**

recall (RoBERTa +0.064; BioMedRoBERTa +0.065 and SurgicBERTa +0.072). These results confirm that using domain annotated data helps the models to both improve the proportion of positive identifications that was actually correct and the proportion of actual positives that were identified correctly. This is in line with what was expected: being RSPF an extension of PropBank for the surgical domain, the CoNLL-2012 dataset, the only SRL training material used for the zero-shot models, does not contain annotated examples for some of the labels of RSPF (the ones in RSPF but not in PropBank), and thus it won't be able to predict them on the test set, where some of these labels are likely to occur. Furthermore, the domain annotated data is fundamental to accurately understanding the surgical procedural language which often has different needs than those of general-English [48].

Similar considerations also apply to the performance on the THO split: the full fine-tuning improves precision (RoBERTa +0.061; BioMedRoBERTa +0.048 and SurgicBERTa +0.033), recall (RoBERTa +0.067; BioMedRoBERTa +0.054 and SurgicBERTa +0.049) and F1-score (RoBERTa +0.063; BioMedRoBERTa +0.051 and SurgicBERTa +0.040) for all considered models. The improvement between zero-shot and full-fine tuning is comparable to that observed for the BAL split. Full fine-tuning typically im-

proves the performance over zero-shot learning also on the GYN split, although the improvement is somehow restrained with respect to the other two splits: precision (RoBERTa +0.024; BioMedRoBERTa +0.010 and SurgicBERTa −0.014), recall (RoBERTa +0.014; BioMedRoBERTa +0.017 and SurgicBERTa +0.016) and F1-score (RoBERTa +0.019; BioMedRoBERTa +0.014 and SurgicBERTa +0.029). This minor improvement may be due to the presence of fewer sentences in the GYN split that require annotation using the RSPF specializations (i.e. those labels in RSPF but not in PropBank): this is somehow confirmed by the significantly higher values obtained with zero-shot on GYN than on the other two splits. We can thus answer RQ1 and RQ2: injecting domain sentences in the training step helps to substantially improve performance in all compared scenarios (RQ2), also when leveraging general-English and biomedical models (RQ1), whose zero-shot scores are lower than the full fine-tuned ones. Also, RQ5 has a positive answer since the improvement from zero-shot to full fine-tuning is comparable between the different splits, showing that the models perform reasonably well when tested on surgical sub-domains not seen during training.

Note that SurgicBERTa achieves the best results in both the zero-shot and full fine-tuning scenarios for almost all metrics of all splits.[2] This confirms that using unsupervised domain adaptation techniques such as MLM can improve performance even in the presence of few or no annotated data. It is interesting to note that SurgicBERTa also improves performance compared to BioMedRoBERTa, which has been specialized in biomedical domain texts. This means that the procedural robotic-surgical domain, which is a specialized subset of the biomedical one, uses a "distinct" language that deserves appropriate, specialized training resources to be adequately covered by language models. We can thus positively answer RQ4.

Table 6.3 goes deeper into the analysis and compares the fine-grained performance, argument-by-argument, by the baseline model (i.e. RoBERTa in the zero-shot scenario - RoBERTa$_{ZS}$) with those obtained by the best model for the BAL split (i.e. SurgicBERTa in a full fine-tuning scenario - SurgicBERTa$_{FFT}$). The detailed results show that full-fine tuning for the BAL split improves the disambiguation of almost all core and modifier arguments. The most substantial improvements are among the core numbered arguments (i.e. Arg-N with N $\in$ 0..6. Quite often, especially for N $\geq$ 3, these are the ones not present in the standard PropBank but introduced in RSPF, and therefore are very specialized arguments of the surgical domain never seen in CoNLL-2012 data.

[2] The only exception is in the zero-shot scenario for the F1 metric of the GYN split, where BioMedRoBERTa attains a slightly better score.

Table 6.3: A fine-grained comparison between a baseline model and the best model for the argument disambiguation task. Best F1 scores are highlighted in bold.

MODEL	RoBERTa$_{ZS}$			SurgicBERTa$_{FFT}$		
ARGUMENT	P	R	F1	P	R	F1
Arg-0	0.696	0.655	0.675	0.879	0.691	**0.773**
Arg-1	0.903	0.890	0.896	0.911	0.926	**0.919**
Arg-2	0.647	0.553	0.596	0.671	0.603	**0.635**
Arg-3	0.000	0.000	0.000	0.554	0.380	**0.451**
Arg-4	0.000	0.000	0.000	0.614	0.628	**0.621**
Arg-5	0.000	0.000	0.000	0.000	0.000	0.000
Arg-6	0.000	0.000	0.000	1.000	0.250	**0.400**
ArgM-ADJ	0.000	0.000	0.000	0.000	0.000	0.000
ArgM-ADV	0.564	0.585	**0.574**	0.553	0.491	0.520
ArgM-CAU	0.500	1.000	**0.667**	0.500	1.000	**0.667**
ArgM-DIR	0.154	0.240	0.188	0.292	0.280	**0.286**
ArgM-DIS	0.500	0.286	0.364	0.429	0.429	**0.429**
ArgM-EXT	0.500	1.000	**0.667**	0.500	1.000	**0.667**
ArgM-GOL	0.000	0.000	0.000	0.000	0.000	0.000
ArgM-LOC	0.381	0.500	0.432	0.436	0.578	**0.514**
ArgM-MNR	0.267	0.722	0.390	0.544	0.681	**0.605**
ArgM-MOD	0.988	0.976	0.982	0.988	1.000	**0.994**
ArgM-NEG	1.000	1.000	**1.000**	1.000	1.000	**1.000**
ArgM-PNC	0.000	0.000	0.000	0.000	0.000	0.000
ArgM-PRD	0.000	0.000	0.000	0.000	0.000	0.000
ArgM-PRP	0.754	0.754	**0.754**	0.708	0.807	**0.754**
ArgM-TMP	0.827	0.865	0.845	0.865	0.865	**0.865**
R-Arg0	1.000	1.000	**1.000**	1.000	1.000	**1.000**
R-Arg1	0.857	1.000	**0.923**	0.857	1.000	**0.923**
R-Arg2	1.000	1.000	**1.000**	0.000	0.000	0.741
R-ArgM-LOC	1.000	1.000	**1.000**	0.500	1.000	0.667

This, again, answers RQ1, since although the zero-shot scenario with RoBERTa obtains acceptable results, using more specific language models and annotated data allows for improved performance.

6.4.2 Predicate and Predicate-argument disambiguation

Table 6.4 shows the results of the other two dimensions considered in our analysis, i.e. predicate disambiguation and predicate-argument disambiguation. For predicate disambiguation, as the used SRL tool was configured to work with gold predicate mentions (i.e. having an oracle that predicts whether a token denotes a predicate or not),[3] for predicate disambiguation we only report the accuracy score, as in this setting, by definition, P=R=F1=Acc.

Table 6.4: Performance (overall) on the predicate disambiguation and predicate-argument disambiguation tasks for BAL, THO, and GYN splits. The best scores are highlighted in bold.

SPLIT	BAL				THO				GYN			
TASK	Pred	Pred-Args			Pred	Pred-Args			Pred	Pred-Args		
MODEL	Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1
RoBERTa$_{ZS}$	0.731	0.544	0.525	0.534	0.769	0.555	0.543	0.549	**0.835**	**0.649**	0.642	**0.645**
BioMedRoBERTa$_{ZS}$	**0.748**	0.560	0.543	0.551	**0.777**	**0.573**	**0.555**	**0.564**	0.810	0.641	0.632	0.636
SurgicBERTa$_{ZS}$	0.735	**0.565**	**0.544**	**0.555**	0.732	0.559	0.540	0.549	0.827	0.646	**0.643**	**0.645**
RoBERTa$_{FFT}$	0.907	0.706	0.689	0.697	**0.910**	0.680	0.672	0.676	0.930	0.745	0.729	0.737
BioMedRoBERTa$_{FFT}$	0.897	0.707	0.694	0.700	0.887	0.669	0.653	0.661	0.935	0.752	**0.748**	**0.750**
SurgicBERTa$_{FFT}$	**0.925**	**0.737**	**0.727**	**0.732**	**0.910**	**0.690**	**0.680**	**0.685**	**0.938**	**0.756**	0.741	0.749

Similar considerations as the one reported for argument disambiguation also hold for these two assessments: using domain annotations allows for improving the performance of the models. The improvements are comparable and very noticeable for the BAL and THO splits, while they are less substantial in the GYN split. Also for the predicate disambiguation and the predicate-argument disambiguation tasks, using a domain language model (i.e. SurgicBERTa) often improves performance. The most

[3] Note that this is by no means a limitation of the comparison conducted in our work as: (i) predicates can be easily spotted via part-pf-speech tagging, considering only the tokens labeled as Verb, or Proper Nouns having specific suffixes (e.g. -ize, -ation); and (ii), this applies for all the models considered in the assessment.

substantial improvements are achieved within the full-fine tuning scenario. Again, this confirms the trends of the data observed on argument disambiguation, thus confirming the answers for RQ1, RQ2, RQ4, and RQ5.

Furthermore, note that the scores for argument disambiguation in Table 6.2 are substantially lower than the ones for predicate-argument disambiguation reported in Table 6.4. For example, SURGICBERTA$_{FFT}$ obtains an F1 of 0.773 for argument disambiguation in BAL split and only a 0.732 (i.e. −0.041) in predicate-arguments disambiguation. The difference in the scores between argument disambiguation and predicate-arguments disambiguation is even larger in the Zero-Shot scenario (e.g. 0.701 vs. 0.534 for ROBERTA$_{ZS}$. That is, in many cases, while the argument label proposed by the models may be correct per se (i.e. ignoring the predicate to which the argument refers), it actually denotes the argument label for a wrong predicate sense, and therefore a wrong argument label in the end, since argument labels are predicate-sense specific in resources such as PropBank. This further confirms the relevance of considering the proposed joint predicate-argument disambiguation performance in SRL evaluations, in addition to the standard (and independent) argument disambiguation and predicate disambiguation.

6.4.3 Few-shot Learning

Finally, Figure 6.3 shows the few-shot learning curve of the SURGICBERTA model, obtained by varying the number of training (and validation) sentences. This assessment allows us to address RQ3.

The curve shows that if the number of added domain annotations is too small, a detrimental effect is obtained for all the analyzed metrics (P, R, and F1). However, with at least 15% of the training material (approximately 190 sentences), the performance constantly grows as annotations are added. Indeed, the curve shows a positive trend also when using all the available domain annotated material (i.e. full fine-tuning), thus suggesting that further improvements are likely by injecting additional annotated examples. However, as stated in Chapter 5, the data annotation for the SRL task in the surgical domain is quite demanding, requiring both linguistic and surgical skills, and its cost is not negligible. This analysis answers RQ3.

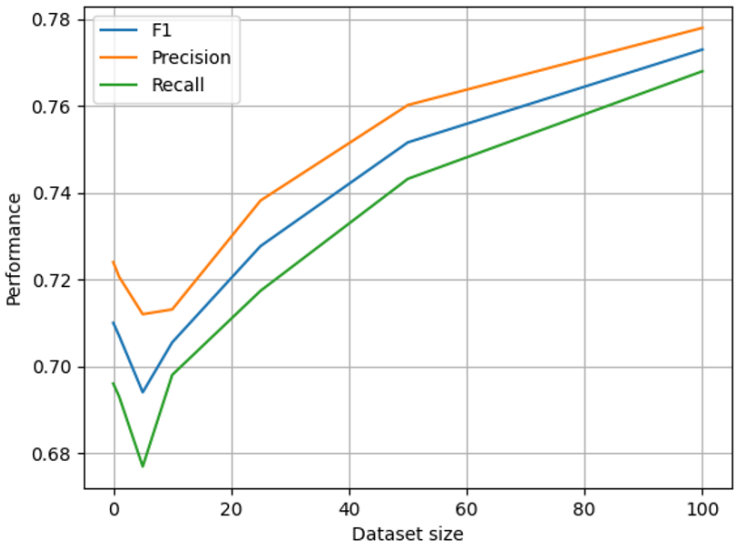


Fig. 6.3: Few-Shot performance of SURGICBERTA model by varying the number of training (and validation) domain sentences.

6.5 Conclusions

In this chapter, we tackled the problem of automatically extracting procedural surgical knowledge from available surgical text materials, such as textbooks and academic papers. Given a text, the goal is to extract structured information about the surgical actions described, the agents performing them, the anatomical parts involved, the tools used, and so on. We proposed to frame the problem as an SRL task and to apply a state-of-the-art approach based on Transformer-based language models. In detail, we experimented with different models: ROBERTA (general-English), BIOMEDROBERTA (biomedical domain), and SURGICBERTA, the pre-trained language model we presented in Chapter 3. We assessed the performance of the models in different, classical scenarios: the zero-shot scenario, where no domain-specific SRL training data is used, and the full fine-tuning scenario, where the models are additionally trained with SRL annotated sentences according to the predicate and roles defined in RSPF, a recently proposed PropBank-style resource covering the typical actions (and related information) of the surgical domain.

Results show that: (i) existing state-of-the-art tools, trained on general-English data, have low performance in extracting structured procedural content in robotic-surgery

procedural texts; (ii) exploiting language models unsupervisely trained on domain-related (BIOMEDROBERTA) or domain-specific data (SURGICBERTA) helps to improve the SRL performance even in the zero-shot scenario; (iii) supervised training with domain-specific SRL data substantially improves the performance of all models on all the SRL evaluation dimensions investigated, i.e. predicate disambiguation, argument disambiguation, and predicate-argument disambiguation. This suggests that for adapting general SRL methods to unexplored, specific domains like the surgical one, some domain-specific SRL manual annotation like the one performed in Chapter 5 is necessary.

7

Towards robotic-surgery task planning from text

> The expert in anything was once a beginner.
>
> Helen Hayes

7.1 Introduction

In the previous chapters, we developed language models to extract complex surgical procedures from as-is textbooks. This chapter aims to introduce a use case and show how, after having defined some constraints on the language, the SRL method, together with other rule-based methods, can help the engineer write the logic rules needed to define the plan for the robot. In particular, we propose AUTOMATE (l**A**ng**U**age **T**o l**O**gic te**M**pl**ATE**s), a pipeline that helps the translation of natural language instructions to linear temporal logic (LTL). However, this chapter uses a controlled language and a general English language model. We made this choice for two reasons. First, state of the art in autonomous surgical robotics mainly uses two tasks as benchmarks: peg transfer and tissue retraction, presented later. Although performed with surgical robots, they are still simplified tasks whose description does not require particularly complex surgical expressions. Nonetheless, as the surgical robotics community moves to more realistic and complex benchmarks requiring more specialized surgical language, the models developed in the previous chapters will allow for more in-depth language understanding. Secondly, we started this use case when RSPB, SURGICBERTA, and its fine-tuned version on the SRL task were not yet available.

The purposes of this chapter are to empirically show that SRL technology is a fundamental tool to extract logical entities from procedural natural language text and to

highlight the technological deficiencies for achieving a completely automated translation. As future work, beyond the purpose of this thesis, we will test the SRL fine-tuned release of SURGICBERTA for task planning.

7.2 Surgical language analysis

As stated before, this chapter uses controlled language, i.e. we impose constraints on how a procedure can be expressed. Nevertheless, to impose suitable language constraints, we analyze in this section the linguistic and stylistic properties of texts describing robotic-surgery procedures. Specifically, we are interested in three aspects that are essential to extract useful task knowledge for autonomous execution:
- the *description of robot setup* relevant to identify *agents of the task*, i.e. surgical instruments;
- the *action representations*, i.e. how operations of the procedure are expressed in domain language;
- the *causal and temporal flow* of actions, e.g. conditions for specific operations and temporal duration.
For our analysis, we considered the resources used to develop the SPKS dataset described in Chapter 4.

7.2.1 Robot setup

Analyzing the structure of surgical manuals used in creating the SPKS dataset, we noted that an initial part of the description is often dedicated to the robot's setup, instruments' docking, and patient positioning. These parts are often described in a separate paragraph, indicated by titles such as *Port Placement and Instruments*, *Robotic Setup and Patient Positioning*, or similar. The setup of the robot is often described with the incremental numbering of *arms* (e.g. first and second arm), indicating which instruments are mounted on them, using verbs such as *equip*, *place*, *install*, *use*, *mount* and *attach*. The procedural description is instead preceded by evoking titles such as *Procedural Details*, *Key Operative Steps*, or *Surgical Technique*. In these sections, however, frequent are also non-procedural sentences describing properties of specific anatomical parts or other considerations not necessary for actual task execution.

Table 7.1: Common linguistic styles usable to express the action of grasping anatomical tissue using a particular surgical instrument. In (Id. 1), the action is expressed in the present tense, and the human is the agent; In (Id. 2), the action is expressed in the passive tense, and the human is the agent; In (Id. 3), the action introduced by the verb to use and human is the agent; In (Id. 4), the action is expressed in the imperative tense, and the agent is not specified; In (Id. 5), the action is expressed in the present tense, and the instrument is the agent;

Id.	Example
1	The surgeon grasps the tissue with forceps
2	The tissue is grasped by the surgeon with forceps
3	The surgeon uses the forceps to grasp the tissue
4	To grasp the tissue with forceps
5	Forceps grasps the tissue

7.2.2 Action representation

In surgery, actions are typically expressed using different styles and verbal tenses. Table 7.1 reports several examples of the action of grasping an anatomical tissue using a particular surgical instrument, expressed following different styles. For the verbal tense, we note that the same procedural action can be described using the *imperative* (Id. 4), *passive* (Id. 2), or *present* (Ids. 1, 3 and 5) form. *Modal* verbs and *phrasal* verbs are also frequent. Furthermore, verbs as *use, employ* and synonyms are often used in procedural texts to introduce the main action that must be performed (Id. 3). The main action is accompanied by a list of procedural entities, such as the *agent* performing the action, the *target anatomical part* affected by the action, and the *surgical instrument* used to carry out the action. These entities may be either expressed explicitly, or taken for granted (e.g. with pronouns or references to previous sentences). The agent may either be a human operator (surgeon, operating room assistant, nurse) (Ids. 1, 2, and 3) or coincide with the surgical instrument (Id. 5).

7.2.3 Causal and temporal flows

We are also interested in analyzing how the following causal and temporal relations are expressed in surgical expert-written texts:
- *conditions* required for facts to become true, or actions to become feasible;

- *temporal sequences* between facts and actions;
- *loop iterations* defining perduration of (sequences of) actions and facts or stop conditions of the corresponding action.

Conditions are mostly expressed with statements containing *if* / *otherwise* words (≈ 85%) or with expressions as *in case* / *otherwise* (≈ 15%). Temporal sequences are mostly conveyed with sentences containing *then* (≈ 67%), *when* (≈ 6%), *after* (≈ 8%), *before* (≈ 5%), *once* (≈ 13%) words; moreover, when two actions appear in consecutive sentences, they are often assumed to be part of a temporal sequence. Finally, loop iterations are almost exclusively expressed in sentences containing the *until* preposition and accompanied by expressions such as *continue to* or *repeat* (or synonyms) *action until* some condition occurs.

7.2.4 Language variability

One difficulty that emerged during the analysis is the use of alternative forms to describe the same concept. Synonyms can be used for actions (e.g. *"move"*, *"go"* and *"approach"* are used with the same meaning), or different expressions can refer to the same anatomical parts (e.g. *"renal tissue"*, *"kidney tissue"*). The synonym management could be tricky in surgical texts since their detection requires not available adequate domain-specific lexical resources. In this work, we adopt a standard general state-of-the-art solution based on WordNet [183], leaving more advanced techniques for future works.

7.2.5 Language constraints for surgical texts

Based on the observations made in Sections 7.2.1-7.2.4, we propose some language constraints which preserve most of the generality of the surgical domain language yet favouring the processing with NLP tools. For temporal and causal relations, we allow only expressions whose frequency in SPKS dataset is ≥ 10%, in order to reach a tradeoff between language generality and NLP performance. The following choices are made for our benchmark texts:

- The elements of the robot are described with incremental numbering (e.g. *first arm* and *second arm*);
- The *robotic setup* is described in the first paragraph; docking of instruments to the robot can be only described by verbs such as *equip*, *place*, *install*, *use*, *mount* and *attach*;

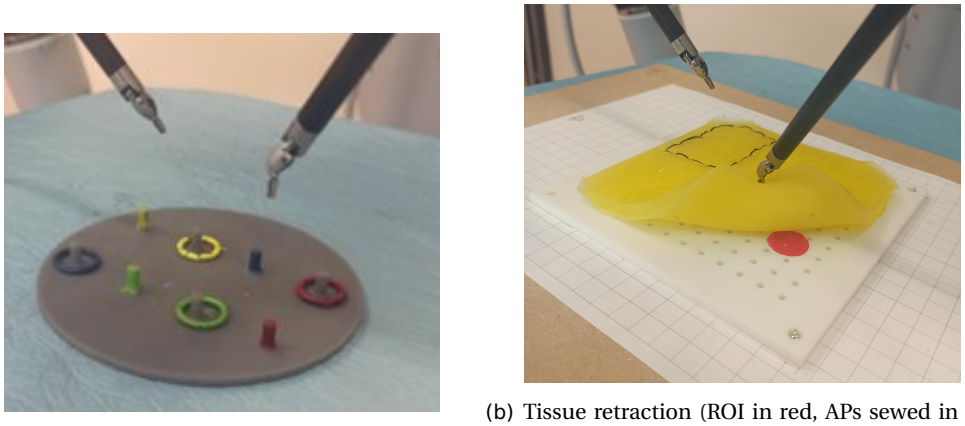
- *verbs* are expressed in active or passive form, at present or imperative tense. Verbs such as use and synonyms are allowed to introduce the main action;
- The list of usable *instruments* is a-priori known (e.g. described in state-of-the-art ontologies such as [184]);
- *instruments* may or may not coincide with *agents*, i.e. with those who perform the action;
- *conditions* can be only expressed with *if* / *otherwise* and *in case* / *otherwise* statements;
- *temporal sequences* can only contain *then* and *once* connectors;
- *loop iterations* can only be expressed with *until* / *repeat* constructs; the action to be repeated must be explicitly indicated and has to coincide with a verb already mentioned before in the text;
- The use of *synonyms* is limited to those statically recognized by the state-of-the-art resources, such as WordNet [183];
- Standard logic connectors (e.g. *and, or*) are allowed in texts to specify more or alternative actions in the same sentence.

7.3 Benchmark tasks

This section describes the surgical training tasks chosen for experimental validation, presenting texts written by experts in the domain used as validation. The texts follow the language constraints defined in the previous section. For both tasks, we consider two different ways of writing the procedural description: one written from the point of view of the autonomous robot (i.e. *the robot as the agent*) and one from the point of view of a surgeon using the surgical robot (i.e. *the surgeon as agent*). Besides being a reference for the surgical robotic community, these tasks also represent concrete examples of robotic manipulation tasks. Thus they are relevant both to validate the potential generality of the AUTOMATE pipeline and interesting for the more generic problem of robotic and process automation from texts.

7.3.1 Peg transfer

The peg transfer (Figure 7.1a) is a training task from the Fundamentals of Laparoscopic Surgery (FLS) [185], recognized as a benchmark for performance assessment in autonomous robotic surgery [186]. The setup consists of three patient-side *arms* of the



(a) Peg transfer.

(b) Tissue retraction (ROI in red, APs sewed in top left.

Fig. 7.1: The setup for the benchmark surgical training tasks with dVRK.

DaVinci Research Kit (dVRK), the research version of the DaVinci surgical robot, two equipped with graspers (first arm and second arm), and one for the camera. The grasping arms operate on a peg base with up to four colored rings, with the goal of placing them on the same-colored pegs. Several constraints influence the workflow of execution. In particular, rings can be picked only from the closest arm, and can be placed on a peg only by the closest arm to it. As a consequence, one single arm can pick and place a ring (e.g. the right with the red ring in Figure 7.1a), or transfer from one arm to the other may be needed (e.g. the blue ring in Figure 7.1a). Furthermore, rings may be initially placed on grey pegs; thus, extraction may be needed before moving them to the peg or transfer point.

Text descriptions

PEG TRANSFER - Robot as agent
The setup has three arms. The first and second arms are equipped with grippers, while the third arm has a camera mounted on it for vision. 4 rings of different colors (red, green, blue, yellow) are placed on a base with 4 colored pegs and 4 grey pegs. First, the camera identifies the rings. Then, the first and second arm open the grippers. The camera selects one colored ring in the scene. If the ring is close to first arm, the first arm attains it; otherwise, the second arm reaches the ring. Then, the gripper grasps the ring. Once the ring is on a peg, the arm raises it. Then, if the peg with the same ring's color is close to the arm, the arm reaches it; otherwise, it transfers the ring to the other arm. If the gripper is at the peg, the ring is placed on the peg. Then, the arm opens the gripper and goes to home position. The camera selects a ring to grasp and the procedure repeats until all visible rings are not on the same-colored pegs.

PEG TRANSFER - Surgeon as agent
The setup has three arms. The first and second arms are equipped with grippers, while the third arm has a camera mounted on it for vision. 4 rings of different colors (red, green, blue, yellow) are placed on a base with 4 colored pegs and 4 grey pegs. First, the surgeon identifies rings via camera. Once rings are detected, the surgeon opens grippers of first and second arms and use camera to select one colored ring in the scene. If the ring is close to first arm, the surgeon uses first arm to reach it; otherwise, the second arm is used to reach the ring. Once reached, the surgeon employs the grippers to grasp the ring. If the ring is on a peg, with help of the arm the surgeon raises it. Then, if the peg with the same ring's color is close to the arm, they use grippers to reach it; otherwise, the ring is transferred to the other arm. If the gripper is at the peg, the surgeon places the ring on the peg, then opens the gripper and moves the arms to home position. The surgeon finally uses third arm to identify a ring to grasp and the procedure repeats until all rings are not on the same-colored pegs.

7.3.2 Tissue retraction

Tissue retraction (Figure 7.1b) is a benchmark task for evaluating the performance of autonomous surgical systems [187]. The robotic setup is the same as peg transfer. The goal is to reveal a (red in figure) Region Of Interest (ROI), e.g. a tumor, hidden below a (rectangular) flap of soft (e.g. adipose) tissue (yellow in figure). The ROI can be exposed by grasping and pulling the tissue with one arm. Candidate grasping points are equally spaced on the tissue surface, discretizing it in a $N \times N$ grid of sub-flaps and considering their centroids as possible targets for arms. For safe manipulation, the set of candidate grasping points is restricted, excluding ones that lie within sub-flaps containing Attachment Points (APs), where the tissue is anchored to surrounding anatomies. Given a randomly selected grasping point, the closest arm executes the task. Pulling does not always ensure task completion. In fact, an arm can pull up to a pre-defined extent, depending on workspace constraints imposed, e.g. by the anatomy of the patient. Furthermore, pulling must be interrupted in case the force exerted by the arm is too high, in order to avoid tissue damage. If pulling is not successful, the robotic tool can move the tissue away from the camera, in order to fold it and ease ROI exposure. In case this action fails (e.g. due to high force on tissue) or is not successful, a new grasping point is selected.

Text description

TISSUE RETRACTION - Robot as agent The setup consists of three robotic arms. First arm and second arm are equipped with grippers, while third arm holds a camera for vision. A flap of adipose tissue is attached to surrounding anatomies at some points (APs), and covers a region of interest (ROI). The camera identifies the APs. First and second arms open the grippers. The camera selects a point on the tissue if it is far from APs. In case the point is close to first arm, the point is reached by first arm; otherwise, the second arm reaches the point. Then, the gripper grasps the tissue and raises it up. The arm lifts the tissue until a maximum height is reached, or maximum force is reached, or the ROI is visible. If the ROI is not visible in case of raising, the gripper goes towards the centre of tissue, horizontally. If the ROI is still not visible, the arm opens the gripper and goes upwards, the third arm selects a different grasping point and the procedure is repeated.
TISSUE RETRACTION - Surgeon as agent The setup consists of three robotic arms. First arm and second arm are equipped with grippers, while third arm holds a camera for vision. A flap of adipose tissue is attached to surrounding anatomies at some points (APs), and covers a region of interest (ROI). The surgeon uses camera to identify APs. Then, the surgeon opens first and second arm grippers. The surgeon exploits camera in order to select a point on the tissue if it is far from APs. If the point is close to first arm, the first arm is used to reach it; otherwise, the surgeon uses the second arm to reach the point. Then, the surgeon grasps the tissue with gripper and raises it. Using the arm, the surgeon lifts the tissue until a maximum height is reached, or maximum force is reached, or the ROI is visible. If the ROI is not visible in case of raising, the surgeon moves the gripper towards the centre of tissue horizontally. If the ROI is still not visible, the surgeon opens the gripper and moves it upwards, use the camera arm to select a different grasping point and the procedure is repeated.

7.4 AUTOMATE pipeline

A schematic representation of the proposed pipeline is shown in Figure 7.2. The first step is *filtering procedural knowledge* from textual resources to select only robot setup information and procedural sentences. These are processed by a SRL module and some filtering rules that highlight the *main action* of each sentence together *with procedural entities* such as agent, object, instrument and causal / temporal information. The output of this module is then automatically translated to *LTL formalism*.

Before describing single steps in more detail, supported by clarifying examples from our benchmark tasks, we explain how synonyms and natural language variability are managed through the whole pipeline.

7.4.1 Synonyms and natural language variability

A complex feature of natural language is the extensive use of synonyms and alternative forms to express similar concepts. A module for reducing the language variability based

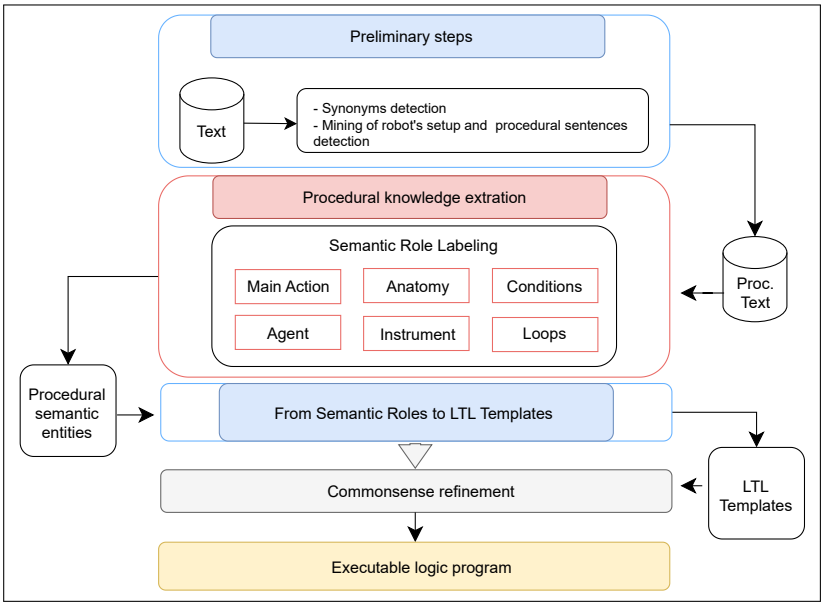


Fig. 7.2: Overview of the proposed AUTOMATE approach for automatic translation of procedural texts to executable ASP rules.

on the resolution of synonyms is fundamental to restructure the natural language description of a procedure. In this chapter, we exploit WordNet's synsets [183] to identify and cluster synonyms. Wordnet is a large lexical database of English verbs, adjectives and adverbs, organized into sets of cognitive synonyms (synsets).

For instance, in peg transfer text with the robot as agent, consider the sentence:

> If the ring is close to first arm, the first arm attains it; otherwise, the second arm reaches the ring.

Verbs *reach* and *attain* belong to the same synset in WordNet, so they are recognized as synonyms for the same action. WordNet is designed for general English and therefore may fail to fully cover all the terms used in surgical language: we leave a WordNet specialization to the domain as a future work. Anyway, this section is meant to emphasize that a module of reduction of natural language variability through resolution of synonyms is fundamental to structure the natural language description of a procedure.

7.4.2 Identifying robot setup and procedural sentences

Before pruning non-procedural sentences from texts, we need to extract robotic setup information. As explained in Section 7.2.5, this knowledge is contained in the first para-

graph of surgical text descriptions and we assume to know in advance the list of all instrument's names: this is not a strict requirement since surgical instruments are common to most procedures, and ontologies, as the one presented in [184], already contain general surgical knowledge, in particular about surgical instruments. We then build a semantic connection between arms and instruments, searching for verbs listed in Section 7.2.5 (e.g. equip, place, install, use and synonyms). In this way, arms and instruments can be used interchangeably in the procedural description.

As an example, consider the text for the peg transfer task, with the surgeon as an agent:

> The setup has three arms.
> The first and second arms are equipped with grippers, while the third arm has a camera mounted on it for vision.

Assuming *camera* and *grippers* to be known as surgical instruments, they are linked to *third arm* and *first and second arm*, respectively, after identifying the words *equipped* and *mounted* as setup-evoking terms. This means that in descriptions, we can use, for example, *third arm* and *camera* with the same meaning.

Afterward, we can exploit the methodology proposed in Chapter 4 to select only procedural sentences in surgical texts. For instance, consider the following sentence from the text for tissue retraction with the robot as an agent:

> A flap of adipose tissue is attached to surrounding anatomies at some points (APs), and covers a region of interest (ROI).

The sentence describes the anatomical setting and, being classified as non-procedural, it is not processed further.

7.4.3 Procedural knowledge extraction

For each identified procedural sentence, we extract actions and relevant semantic information, which represent the procedural knowledge needed for LTL template extraction. To this purpose, we exploit PropBank-based SRL described in 2.6.2 and some rules defined on the language constraints presented in Section 7.2.5.

Selecting main action

Given a procedural sentence, we first use Part-Of-Speech (POS) algorithm [155] to identify verbs (hence potential actions). However, multiple verbs may occur in a sentence.

For instance, consider the following excerpt from the text for peg transfer, with the surgeon as agent:

> If the ring is close to first arm, the surgeon uses first arm to reach it; [...]

Three verbs are identified in this sentence, i.e. *is*, *uses* and *reach*. However, only *reach* is the main action. In order to identify it, we proceed as follows: first, we exclude *-ing* forms, modals and auxiliaries (e.g. *is*), as well as verbs such as *use* and its synonyms, which only introduce main task actions; we then exclude all the candidate verbs that appear in a span of text that SRL has labeled with semantic roles referred to causal and temporal relations (c.f., next section). This approach is robust with respect to different verbal forms, e.g. passive verbs.

Identifying semantic roles

Semantic roles from SRL must be matched to the relevant meanings for task procedural description, i.e.:

- the *agent* who performs the action
- the *object* (e.g. anatomical part) undergoing the action
- the *instrument* (or *robotic arm*) used to perform the action
- *causal and temporal relations* such as conditions, temporal sequences and loops

Assuming the language constraints from Section 7.2.5, we combine the spotting of connectors (e.g. *if, until*) and semantic roles to detect causal / temporal information. In particular, for detecting conditions, spans of text labeled by the SRL with ArgM-ADV (adverbial modifiers) or ArgM-DIS (discourse Markers) and respecting the form presented in Section 7.2.5 (i.e. containing *if* / *otherwise* or *in case* words) are selected. A similar approach is adopted for loops and temporal sequences, by selecting spans of text labeled with ArgM-TMP (temporal markers) and containing connectors (i) *then* and *once* for temporal sequences and (ii) *repeat* / *until* for loops.

In order to identify agents, objects and instruments, we need a different strategy for *robot-as-agent* and *surgeon-as-agent* scenarios. In the first scenario, the robot' arm or instrument plays the role of *agent* and thus span of texts labeled with Arg0 are selected. Instead, the *object* of the action always plays the role of proto-patient, and thus spans of text labeled as Arg1 are selected. For instance, in the tissue retraction text, the sentence *The camera selects a point on the tissue if it is far from APs.* is labeled as:

> [Arg0: The camera] [V: selects] [Arg1: a point on the tissue] [ArgM-ADV: if it is far from APs].

Thus, by looking at the arguments labeled with Arg0 (*camera*) and Arg1 (*point on the tissue*), the *agent* and the *object* of the action (*to select*) are identified respectively.

When the surgeon is the agent, the situation is more complex because the instrument/arm is not the agent (Arg0) and, if mentioned, it occurs in other roles of the corresponding action. In particular, the instrument/arm can be contained either in a core argument (Arg2 or Arg3) or a non-core argument (ArgM-MNR), depending on the specific verb. For instance, in the peg transfer scenario, the sentence *First, the surgeon identifies rings via camera* is annotated via SRL as:

[ArgM-TMP: First], [Arg0: the surgeon] [V: identifies] [Arg1: rings] [ArgM-MNR: via camera].

The span *via camera* is labeled as an ArgM-MNR.

In this case, we can rely on available resources (e.g. a knowledge base or ontology such as [184]) listing surgical instruments and we search the sentence for mentions of these instruments within arguments labeled as Arg-MNR, Arg2, or Arg3 by SRL. In the considered example, *camera* is a candidate instrument, it is labeled as ArgM-MNR, and thus is recognized as the instrument for the *identify* action.

The use of SRL also for instrument detection has to be necessarily performed since not all mentions of medical instruments in a sentence refer to the actual usage of an instrument to perform the main action. In particular, it has to be applied after identifying the candidate arguments possibly containing the instrument. For instance, the sentence *The surgeon uses the first arm to grasp scissors* mentions *scissors*, a surgical instrument, but without referring to its usage to perform the action herein described (*to grasp*). Indeed, SRL returns the following annotations:

[Arg0: The surgeon] uses the first arm to [V: grasp] [Arg1: scissors].

That is, *scissors* is correctly recognized as the *thing grasped* (Arg1 of *grasp*), and not as the instrument used to perform the grasping (i.e, Arg2, Arg3, or ArgM-MNR).

7.4.4 From SRL to LTL relations

The output of SRL highlights relevant semantic information about task knowledge. This section shows how this information is automatically translated to LTL logic templates representing procedural relations for the task. Consider the following example sentences from the texts of our benchmark tasks (the same considerations apply also to the other sentences of the benchmark tasks):

(A 1): In case the point is close to first arm, point is reached by first arm; otherwise, the second arm reaches point.

(A 2): Then, the gripper grasps the tissue.

(A 3): The arm lifts the tissue until a maximum height is reached, or maximum force is reached, or the ROI is visible.

The output of SRL and filtering steps described before is, respectively:

(B 1): [ArgM-ADV: In case the point is close to first arm,] [Arg1: the point] is [V: reached] [Arg0: by first arm;] [ArgM-ADV: otherwise] [Arg0: the second arm] [V: reaches] [Arg1: the point.]

(B 2): [ArgM-TMP: Then,] [Arg0: the gripper] [V: grasps] [Arg1: the tissue.]

(B 3): [Arg0: The arm] [V: lifts] [Arg1: the tissue] [ArgM-TMP: until a maximum height is reached, or maximum force is reached, or the ROI is visible.]

In order to highlight relevant semantic entities (verb, agents, objects, and temporal/causal information), these can be re-written in a more convenient predicate form as follows, with the verb as the name of the predicate with ordered arguments *agent - object - additional information*:

(C 1):
reach(the first arm, the point, ADV: in case the point is close to first arm)
reach(the second arm, a point on the tissue, ADV:otherwise)

(C 2):
grasp(first arm, the tissue, TMP:then)

(C 3):
raise(first arm, the tissue, TMP:until a maximum height is reached, or maximum force is reached,
or the ROI is visible.

Annotations *(C 1)* correctly split sentence *(A 1)* into two main *reach* actions, having either *first arm* or *second arm* as agent and *a point on the tissue* as object / anatomical target. SRL correctly identifies the conditions *if* / *otherwise* as ADV roles. Sentence *(A 2)* leads to annotation *(C 2)*, where the temporal relation *then* is also marked. The agent *gripper* is automatically translated to *first arm*, following instruments recognition from the robot's setup description (see Section 7.4.2). Finally, in sentence *(A 3)*, the *until* temporal relation is recognized. Furthermore, the main action *lift* is automatically translated to the synonym *raise*, which is arbitrarily chosen as the representative lemma of the synset in WordNet (see Section 7.4.1) containing both *lift* and *raise*.

Given the SRL output, we translate logic / temporal connectors defined in the language constraints to corresponding LTL operators. This can be done automatically, assuming such an injective map exists. We obtain the following LTL rule templates:

(D 1):
`reach(first arm, a point on the tissue)` ← `the point is close to first arm`
`reach(second arm, a point on the tissue)` ← ¬ `the point is close to first arm`

(D 2):
∘ `grasp(first arm, tissue)`

(D 3):
`raise(first arm, tissue)` $\mathcal{U}$ (`maximum height is reached` ∨
`maximum force is reached` ∨ `the ROI is visible`)

where $\mathcal{U}$ denotes *until* operator, ∘ is the *next* operator, ← is the logic *implication*, ∨ is logic *disjunction* and ¬ is logic *negation*.

7.4.5 From LTL templates to executable logic program

LTL templates encode task actions, agents, relevant objects / anatomical parts, and temporal/causal relations, which determine the flow of execution. However, they must be translated to the syntax of a specific logic program, in order to actually implement an autonomous task planner for robotics.

A logic program represents a domain of interest with a *signature* and *axioms*. The signature is the alphabet of the domain, defining its relevant attributes. Attributes may be *statics*, i.e. domain attributes whose values do not change over time, or *fluents*, i.e. time-dependent domain attributes. Attributes may be *terms*, *atoms*, predicates of terms (e.g. `atom(`t_1`, ..., `t_n`)` is an atom with terms $t_{1,...,n}$ as arguments), and their classical or default negations (respectively, ¬`a`, meaning that `a` is false, or `not a`, meaning that `a` is not known to be true). Values of terms are *constants* (either integers, Booleans or strings). A term whose value is assigned is *ground*, and an atom is ground if its terms are ground.

Axioms are logical relations between attributes. A causal rule `h` ← b_1`, ...,` b_n defines preconditions $b_{1,...,n}$ (*body* of the rule) for the *head* `h`. A rule with an empty head defines a constraint, meaning that body atoms cannot be ground concurrently. Axioms can also represent temporal relations between atoms, thanks to the definition of an explicit time variable `t` for fluents[1]. For instance, `a(t)` ← `b(t-1)` is equivalent to the *next* operator in LTL, meaning that `a` occurs at the subsequent time step with respect to `b`; similarly, `a(t)` ← `a(t-1), ¬ b(t)` encodes LTL *release* operator, meaning that `a` keeps holding until `b` does not.

[1] There exist logic programming frameworks, e.g. *telingo* [188], which implement LTL operators. However, we consider the explicit temporal variable definition to adhere to ASP syntax, which is more popular in the robotics and AI community.

In the classical representation of task knowledge, statics typically define agents and invariant environmental resources, while fluents are actions and dynamic environmental features. Axioms encode task specification, capturing the causal relations between the agents and the environment, following the pattern *precondition* → *action* → *effect* with *constraints* proposed in the Planning Domain Definition Language (PDDL) [189]. As an example, consider LTL relation *(D 1)*, defining the precondition for *reaching* action. Assuming that *the point is close to first arm* is encoded with an atom `close(first arm, tissue point)`, being `first arm` and `tissue point` constant values for domain variables `Agent` and `Object`, respectively, we can write the following axiom:

$$\texttt{reach(Agent, Object, t)} \longleftarrow \texttt{close(Agent, Object, t).}$$

LTL templates can then be automatically implemented in the formalism of a logic program with the following steps:

- implementation of LTL operators in the specific logic programming syntax;
- definition of variables and atoms, with variables lifting information retrieved from SRL (e.g. *reach(Agent, Object)*, which can be lifted from *reach(first arm, tissue)*).

Representing operators is trivial since they are implemented in any logic program. The second step, however, requires yet missing information. In particular, underlined parts in LTL templates *(D 1 - D 3)* represent low-level concepts and variables, which at the moment need to be encoded by a logic programming expert manually. They mostly represent commonsense knowledge (e.g. *the point is close to first arm*, associated with the notion of *spatial distance*, or ROI visibility and maximum force measurement), which are not related to the specific procedure, but rather to the generic surgical and robotic domains. Hence, similarly to instrument knowledge (see Section 7.4.2), we can assume these concepts are stored and retrieved from clinical domain [190] or even general-scope ontologies [191], in order to make the translation to an executable logic program possible.

7.5 Application of AUTOMATE

In this section, we empirically test the utility of AUTOMATE. In other words, we verify that the extracted task knowledge is correct and general enough to compute suitable

plans for our benchmark robotic tasks, given any possible initial environmental context. This requires:

1. implementation of LTL templates into a specific logic programming language for autonomous planning;
2. implementation of low-level routines for robotic motion planning/control and perception, needed for environmental context evaluation and instantiations of LTL predicates;
3. an environment to replicate the benchmark tasks and the robot.

To address the first requirement, we implement LTL templates in the formalism of ASP, specifically with Clingo 5 [192] software for ASP representation and solving (i.e. plan computation). For the second requirement, we adopt the framework for integrated planning and execution of surgical robotic tasks proposed in [193], including the dVRK and vision sensors. For the third requirement, we create simulation environments for the peg transfer and tissue retraction.

We empirically evaluate the quality of the extracted task specifications in terms of *planning success* and *planning computational performance*.

The planning success measures the percentage of successful generation of task plans in a set of random environmental contexts. It is important that random contexts are representative of the variability of the task, hence lead to the generation of multiple workflows of execution. Hence, we randomize relevant variables for our two benchmark tasks, for a total of 100 different contexts for each task.

The computational performance is calculated as the time required by Clingo solver to find a first plan (i.e. neglecting possible re-planning in case bad events occur), given some initial environmental context. We evaluate this metric for increasing size of each domain of interest, which affects the number of variables/atoms to be grounded by ASP. For each complexity class of context configuration, we replicate 20 executions to calculate the mean and variance of Clingo's planning time. In this way, we can analyze the evolution of computational effort as task complexity increases.

For both metrics, we compare the performance of the extracted ASP program with the ASP task description written by an expert in both of the domain and the logic syntax, following PDDL-like classical representation. Standard task description from PDDL does not always match LTL formulas extracted from texts with NLP. For example, consider relation *D 2*. It specifies that *grasp* shall occur after the previous action (*reach*). This can be encoded as:

```
grasp(Agent, Object, t) ← reach(Agent, Object, t-1).
```

In classical PDDL representation, a specific effect for *reach* should be defined, which then acts as a precondition to *reach*. An example is provided below:

```
at(Agent, Object, t) ← reach(Agent, Object, t-1).
grasp(Agent, Object, t) ← at(Agent, Object, t).
```

where `at(Agent, Object, t)` is a fluent representing the location of an arm with respect to an object.

7.5.1 Peg transfer

In the peg transfer domain, the simulation is implemented in CoppeliaSim.[2] The perception module is in charge of identifying locations of rings, pegs, and robotic arms, in order to instantiate LTL predicates properly (e.g. distances between rings and arms) [193].

Planning success

Domain variables which influence the workflow of execution, hence are relevant for assessing planning success, are:

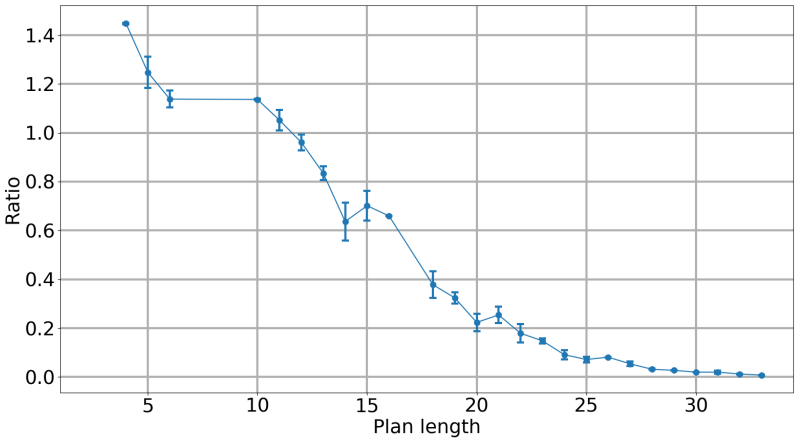
- number of visible rings (affecting the number of required actions to complete the task successfully);
- placement of rings on the pegs, which requires extraction before bringing them to the pegs;
- relative positions of rings with respect to arms, affecting reachability conditions and thus possibly requires a transfer between arms before placement on pegs.

Hence, we generate random scenarios as follows:

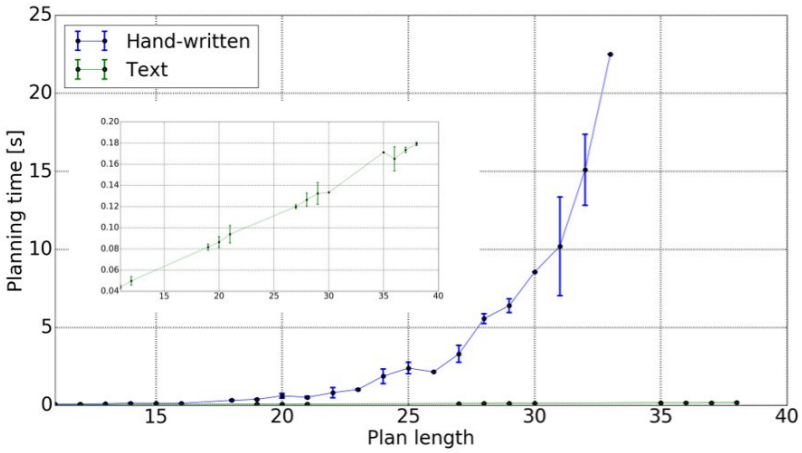
- 19 scenarios present only 1 ring, 30 scenarios 2 rings, 22 scenarios 3 rings, and 29 scenarios 4 rings[3];
- 84 / 100 scenarios present at least one ring on a peg, so they require extraction;
- 80 / 100 scenarios require transferring of rings between arms.

[2] `https://www.coppeliarobotics.com/`

[3] The maximum number of rings in the scene is set as of FLS specifications [185].



(a)



(b)

Fig. 7.3: **On top**, the ratio between planning times with ASP program from text and hand-written ASP program, for 100 random initial configurations (sorted and clustered by plan length) of peg transfer task. **In the bottom**, mean and standard planning times for the two ASP programs vs. plan length. In the box, focus on the results for ASP encoding extracted by AUTOMATE (same units as the main plot).

The task is considered successful when all visible rings are placed on the same-colored pegs. In all scenarios, a 100% success rate is achieved with both ASP programs.

Computational performance

The complexity classes for the ring transfer are defined by the same variables randomized for planning success evaluation. In fact, the number of rings and actions to be executed for each of them affects the plan length, hence the number of atoms to be grounded by Clingo. Hence, we consider the same scenarios as above and arrange them by plan length, reporting mean and standard deviation for each cluster of scenarios with the same plan length.

In particular, Figure 7.3a shows the ratio between the planning time with the ASP program extracted from text and the expert-written one. Except for shorter plans, the ratio is < 1 when plan length is > 12, meaning that extracted ASP task knowledge is more efficient for the solver. Moreover, the ratio decreases significantly for longer plans.

This is even more evident in Figure 7.3b, showing the planning time for both ASP programs against the plan length. As the plan length increases, the computational performance of the ASP program extracted from text scales linearly with the length of the plan, thus significantly better than the hand-written program with quadratic progression. This happens because of the different ASP representation, with the classical PDDL-like formalization possibly having more axioms as explained at the beginning of Section 7.5. In fact, Clingo computes plan grounding atoms iteratively, starting from initial conditions and propagating through axioms. Hence, more or longer axioms require more computational time.

Notice that the two ASP programs generate plans with different lengths, though under the same initial configurations. This depends on a slightly different action representation. For instance, in the text description, there are actions as *selecting a target ring with camera* which are captured by SRL and then converted to LTL / ASP predicates. However, such actions are not properly moving actions, so they do not affect the workflow of execution. Hence they are not encoded by the expert writing ASP program from scratch.

7.5.2 Tissue retraction

For tissue retraction, we assume that the grasping points may be selected in a discretized set, obtained as follows:

1. the rectangular tissue flap is discretized as a $N \times N$ grid;
2. candidate grasping points are centroids of cells in the grid.

At LTL / ASP level, a variable for the candidate grasping point is added, with a unique identifier for each point in $\{1,...,N^2\}$. We use a simulation within the Sofa framework[4] to emulate soft tissue deformation via finite element methods. The simulated perception module is in charge of identifying locations of grasping points and APs on the tissue and measuring ROI final visibility. In this way, it is possible to ground LTL / ASP predicates and reason on task knowledge to compute a plan, which is then executed by the motion planning and control module.

Planning success

Variables that affect the workflow of execution are:

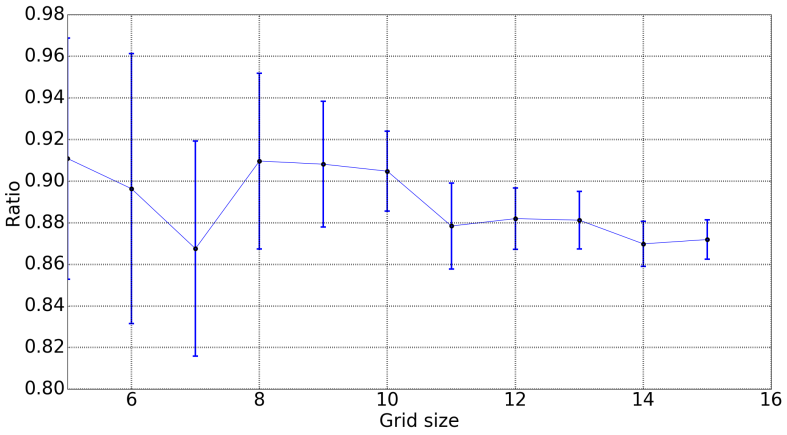
- initial grasping and pulling of the tissue may not be sufficient to expose ROI, so re-planning (either further pulling or moving away from the camera) may be useful;
- a different arm (PSM) may be needed to grasp the tissue, depending on the chosen grasping point, which depends on APs locations (recall that arms should not operate close to APs).

Hence, we generate random contexts with fixed $N = 5$ grid discretization, specifically 35 / 100 requiring re-planning and 67 / 100 requiring usage of the first arm. The task is considered to be successfully executed if the final ROI exposure percentage is $> 70\%$. When the hand-written ASP program is implemented for task planning, the planning success rate is 98%, against 94% with the ASP program extracted from text. However, the mean and standard deviation of ROI exposure with hand-written ASP program and extracted from text (considering only successful task executions) are respectively $92.26\% \pm 9.53\%$ (100% median) and $97.47\% \pm 6.90\%$ (100% median). Overall, extracted ASP encoding has similar performance on planning success with respect to expert-written one.

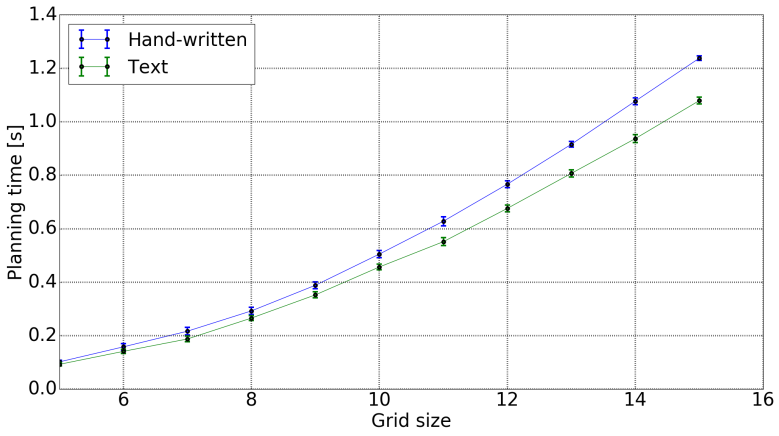
Computational performance

The planning time with Clingo depends mainly on the number of candidate grasping points, i.e. the grid discretization parameter N since it increases the number of ASP variables. On the contrary, the specific location of ROI does not affect the computational complexity since we only evaluate the initial planning time, i.e. neglecting any re-planning occurrence. Then, we consider different $N \times N$ grid discretizations of the tissue flap, with $N \in \{5,...,15\}$, and randomize 20 different locations of APs and ROI for

[4] `https://www.sofa-framework.org/`



(a)



(b)

Fig. 7.4: **On top**, the ratio (mean ± standard deviation) between planning times with ASP program from text and hand-written ASP program for tissue retraction task, for different size N of grid discretization of the tissue (100 initial configurations per size). **In the bottom**, mean and standard planning times for the two ASP programs vs. grid size.

each of them. In other words, a complexity class of scenarios for the tissue retraction task is represented by the value of N.

In Figure 7.4a we show the ratio between planning times obtained with the ASP program extracted from text and the hand-written one. The ratio is always < 1, meaning

that ASP axioms extracted from text are more efficient than hand-written ones. The ratio does not significantly vary for different tissue discretizations, while the absolute discrepancy between planning times for the two ASP programs increases (Figure 7.4b). Thus, extracted ASP task knowledge is still slightly more efficient for the ASP solver.

7.6 Discussion

This chapter empirically shows that it is possible to extract procedural information with NLP techniques, in the form of LTL relations, from text written by a domain expert. The extracted knowledge can then be easily translated into any logic programming formalism for autonomous planning. A solver for the task planning problem (Clingo in our experiments) is then able to compute a suitable plan, given any random initial context, with a similar percentage of success as a logic program written by an expert both of the domain and the logic paradigm. Furthermore, we found that the logic program extracted from text is computationally more convenient for the solver since less time is needed for plan computation, thanks to the more efficient formalization with respect to PDDL-like specifications in the *precondition-action-effect* paradigm. However, one of the goals of this chapter is also to highlight the issues that still remain to be addressed.

In this chapter, we applied a general-English model on a controlled language, built on purpose to be correctly interpreted by state-of-the-art tools. Moving to completely unconstrained natural language, interpretation problems are likely to increase, especially in a domain such as surgery, where the plan must be certified before being performed. In that case, the use of SURGICBERTA presented in Chapter 3 will be necessary to understand the procedural surgical language better. Another similar problem is related to the specific domain lexicon that is sometimes used, which is completely unknown to state-of-the-art resources. For example, one limitation concerns synonyms and alternative expressions for the same concept. Often general-English resources (such as WordNet used in this chapter) are not effective in dealing with the surgical domain, and further research should be carried out to enrich them with surgical terminology and expressions manually. For example, the verb *excise* can sometimes be used with the same meaning as *remove*, but this does not emerge from WordNet.

Furthermore, in order to automate the entire process without requiring human intervention, we should deal with the intrinsic incompleteness of natural language descriptions that often leave some knowledge unsaid. This is the case, for instance, of spe-

cific encoding of environmental fluents (see Section 7.4.4) that have to be filled manually: at the moment, this pipeline significantly helps the programmer to write a logic program encoding effective task representation, for example, it can identify actions, agents, relevant environmental/domain entities as target objects and anatomies, and semantic roles which can be directly mapped to LTL syntax. Hence, the programmer shall only convert LTL expressions to a specific logic program syntax (e.g. ASP), with no required awareness of the specific task/surgical procedure. Anyway, encoding of environmental information, instead, is mostly related to commonsense concepts (e.g. spatial information) or the considered domain at large (e.g. surgery) rather than the specific task. The concept of commonsense is analyzed in Chapter 8.

Another important aspect to be considered is the difference between classical PDDL-style task knowledge representation and text-extracted one. Experimental results have evidenced that ASP programs extracted from the text are not only adequate to represent the domains of the two tasks considered in this chapter, but they are also more efficient, i.e. the planning time required by state-of-the-art Clingo solver is reduced (the amount of the improvement depends on the specific task domain). This is probably related to the different complexity of axioms between classical and extracted representations.

A final point to be discussed is *completeness* of LTL relations extracted from text, i.e. their adequacy to represent *all possible task occurrencies* (i.e. initial configurations) and guarantee successful task completion. Procedural texts usually describe a limited range of possible scenarios. Unexpected events and conditions (originated, e.g. by the uncertainty of pre-operative information, intra-operative anomalies, and patient-specific clinical situation) may not be described. This also happens in the training tasks studied in this chapter. For instance, consider the peg transfer task. In the analyzed initial configurations of rings on the peg base, we have allowed rings to be placed either on grey or no pegs. However, rings may also be placed on colored pegs. An example is shown in Figure 7.5. The blue and red rings occupy red and blue pegs, respectively, and they are all reachable only by first (right) arm. Obviously, we could enrich the description by adding a final paragraph in the text where all well-known exceptions are described, similarly to what is done for the robot setup. However, when the plan is not complete, human intervention must always be requested.

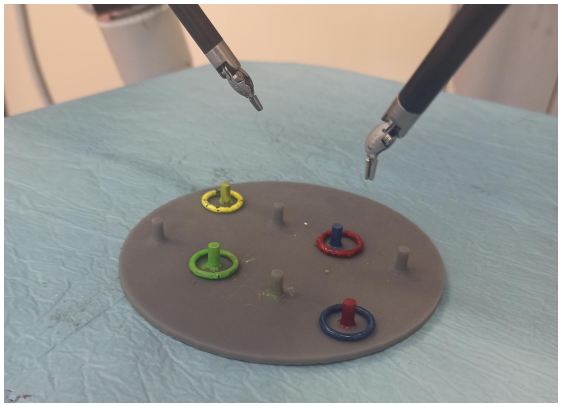


Fig. 7.5: An anomalous condition for peg transfer task, where colored pegs are occupied.

7.7 Conclusion

In this chapter, we have presented AUTOMATE, a pipeline exploiting SRL for procedural task knowledge extraction from text. We empirically showed that starting from specifications in controlled language, it is possible to exploit SRL techniques for extracting task actions (with relevant semantic information as agents and targets/objects) and LTL templates from textual procedural descriptions. In the context of two benchmark surgical training tasks, which also exemplify the more general domain of robotic manipulation, we have then translated extracted knowledge to the state-of-the-art logic programming ASP formalism, and compared it with respect to classical PDDL-like representations written by domain and logic experts. Experiments in randomized simulations of the scenarios have shown the improved performance in terms of planning time when considering text-extracted ASP representations, and the suitability to solve the task planning problem successfully. Anyway, this work is a preliminary step towards the exploitation and understanding of the knowledge contained in domain-specific manuals and has as its main goal that of highlighting research directions open for investigation. One of the main limitations is that, for extracting meaningful robotic domain-specific procedural knowledge and allowing a higher language variability, an adaption of WordNet (or similar resources) to surgical English is necessary. While in simple procedures written in a controlled language, the translation is feasible, state-of-the-art algorithms and resources will probably face more challenges in more complex scenarios. Another main issue is the lack of commonsense knowledge in procedural descriptions.

Work needs to be done on the representation of commonsense in surgery. The next Chapter goes further on the problem of commonsense knowledge management.

8

The need for commonsense knowledge in autonomous surgical robots

> How much do we know at any time? Much more, or so I believe, than we know we know."
>
> Agatha Christie, *The Moving Finger*

8.1 Introduction

Chapters 3-7 have shown that automatic extraction of surgical procedures from surgical textbooks is a challenging but not impossible task. One main issue is that textbooks do not include the large amount of implicit knowledge that humans use during surgical tasks. Indeed, while executing a surgical procedure, surgeons do not only rely on their specific medical knowledge but also on a set of skills that are “obvious" to them and allow them to intuitively evaluate and react to the intervention evolution. Such skills belong to what is usually called commonsense, which is essential to carry out an intervention. While general commonsense refers to all the basic concepts about the world and belongs to all human beings (e.g. the fact that a needle must be inserted from tip to eye), we believe that field-specific commonsense is developed depending on individual experiences within a field of expertise. In surgery, field-specific commonsense is the “glue" knowledge that is not explicitly described in surgical manuals, but it is acquired during the long surgical training. For example, a textbook does not explicitly describe how the needle should be held nor how it should be inserted in the human body, but this knowledge is known by domain experts. Understanding how to describe, represent and learn this knowledge is paramount to developing autonomous robotics surgical systems. The importance and challenges of commonsense have been discussed

in other fields [194, 195], but to the best of our knowledge this aspect has not yet been addressed in robotic surgery.

8.2 Commonsense and surgery

In the following, we refer to the granularity classification of surgical procedures in phases, steps, actions, and motions proposed by Lalys et al. [19] and presented in Chapter 1. Based on these definitions and in-depth discussions with surgeons, we propose a preliminary classification of surgical knowledge into four levels:

1. *Procedural knowledge* is the description of the sequence of phases required to perform a procedure, as can be learned from surgical manuals (c.f. 1.1). It does provide general information about the specific steps, but it does not specify the parameters of each step, i.e. the physical quantities that instantiate individual actions and motions, such as the motion velocity or the force to be applied.
2. *Surgical commonsense* is a field-specific commonsense and encompasses all those skills surgeons acquire while experiencing (assisting and performing) a specific procedure multiple times. It allows for defining the sequence of elementary actions needed to perform the surgical task and intuitively setting their parameters. It also includes the capability to interpret surgical situations and thoroughly understand correlations, causes, and consequences of actions and thus select the best surgical technique for each specific situation.
3. *Medical commonsense* is another subset of field-specific commonsense, that is not specific to a single surgical procedure and is acquired during medical studies. For example, it includes the knowledge of basic anatomical concepts (positions and functions of organs), the high-level understanding of how surgical actions impact anatomy, the evolution of the patient's prognosis after surgery and medications.
4. *General commonsense* is commonsense that surgeons have as human beings. It represents the basic knowledge of the world, is acquired while experiencing everyday situations, and helps infer the meaning and behavior of things.

This classification is not always crisp, because it depends on the context. The same concept can refer to one or to another type of commonsense. For example, knowing the effects of a complex surgical action on the internal organs requires surgical knowledge and experience (and thus surgical commonsense) that may not be required in simpler and standard cases, when only medical commonsense may be sufficient.

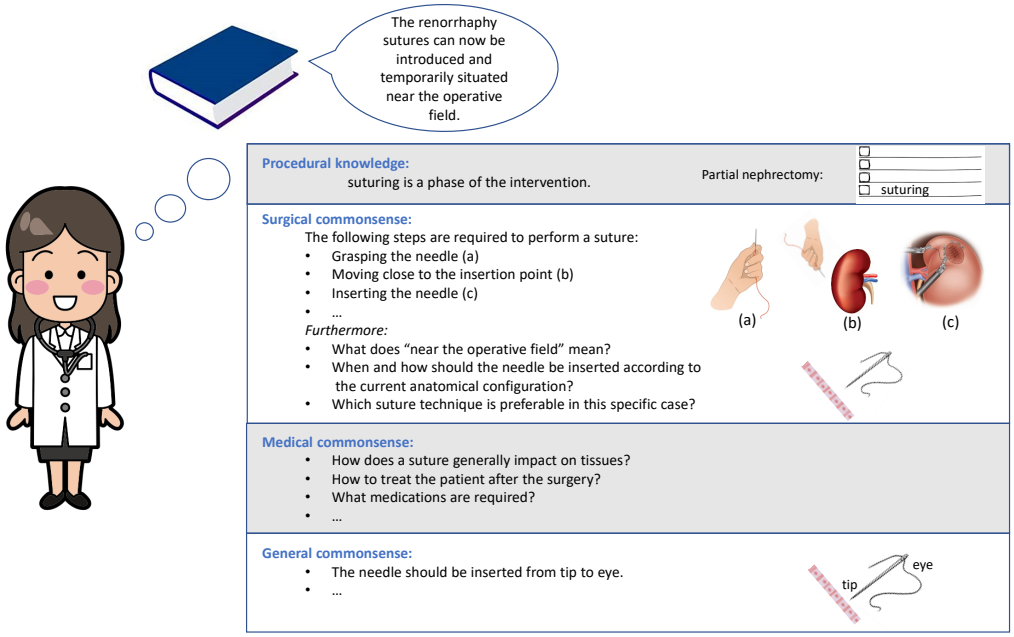


Fig. 8.1: Procedural knowledge enriched with surgical, medical and general commonsense.

To clarify the proposed classification, we introduce Fig. 8.1 that describes the commonsense knowledge required during the suturing phase of a partial nephrectomy intervention. This classification can be adapted to other surgical phases or procedures.

The sentence at the top is taken from a surgical textbook [114] and represents the *procedural knowledge* of the intervention. However, it does neither describe how to assess the conditions of a good suture, nor how to select the best surgical approach to connect the tissues, nor it lists the individual steps and actions needed to accomplish it (*surgical commonsense*). Furthermore, it neither describes the effect that generally a suture causes on tissue nor how to pharmacologically treat the patient after the surgery (*medical commonsense*), nor it provides the implicit general knowledge about objects involved in the suture, e.g. a needle is needed, and it must be grasped, inserted and extracted (*general commonsense*).

Despite these missing details, surgeons are able to perform the intervention after reading a manual. This is possible because they can leverage their broader background that glues information together. Surgical textbooks alone are not sufficient to fully describe an intervention, and an autonomous surgical robot must acquire the same level

of knowledge mentioned above to perform the surgical task and to properly handle the situations occurring.

8.3 Mapping commonsense skills to autonomy levels in surgery.

The different levels of autonomy proposed by Yang et al. [14] and summarized in Chapter 1 are obviously associated with different levels of knowledge, and we propose the classification schematized in Fig. 8.2. In particular, the sophisticated capabilities required to reach high autonomy levels are implicitly connected to the breadth of the required commonsense knowledge.

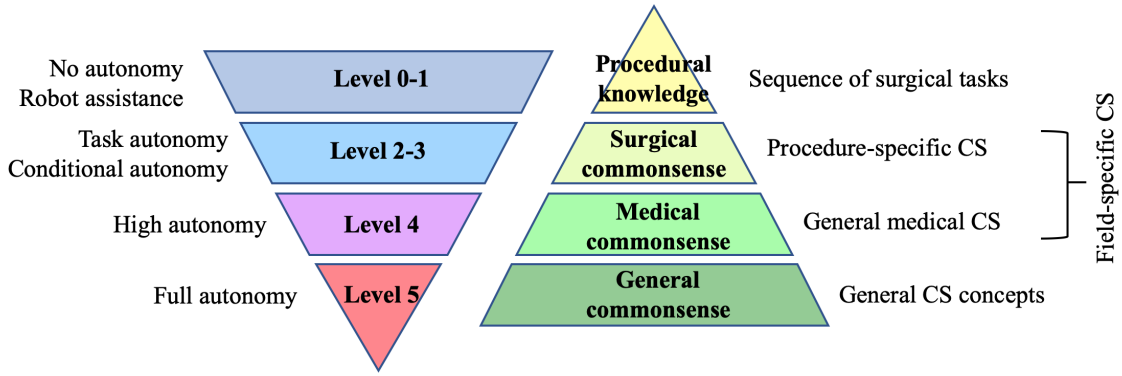


Fig. 8.2: Mapping between autonomy levels and the required knowledge levels. Higher autonomy requires broader knowledge. “CS" stands for “commonsense".

Current surgical robots are teleoperated systems with some assistive function (level of autonomy 0 and 1), and have no knowledge of the steps to be performed but simply monitor some working variables, such as current levels, maximum speed, and electrical noise. The specific phases and steps of the intervention are determined by the surgeon who directly controls the surgical actions.

However, even when autonomous functions are limited (levels 0 -1) a certain amount of commonsense knowledge is implicitly included in the control algorithms and data analysis methods. For example, autonomy level 0 has specific actions that include e.g. tremor suppression and maintaining orientation during clutching. At level 1, the robot

provides dexterity and cognitive support to the human for specific actions or tasks, but not for strategic decisions (plans) or actions, and all commonsense reasoning is provided by the surgeon.

When a surgical robot is able to execute some individual actions (levels of autonomy 2 and 3), it would require the presence of *surgical commonsense*. In fact, the robot would need to perform some basic reasoning on the patient-specific pathology and anatomy, with the surgeon ready to intervene if needed. Referring to Fig. 8.1, the robot would have to interpret the "near the operative field", and decide where to insert the needle given the patient's anatomy. Furthermore, in the example above, the robot would have to know that a suture needs a *surgical* needle, which has to be grasped with a specific orientation for optimal insertion. It has to control motion parameters (force and velocity) during the insertion and it has to know how to close the suture. Some concepts of *medical commonsense* would be needed to give the autonomous robotic surgical system more autonomy and connect multiple phases of an intervention. For example, the autonomous robotic surgical system needs to know the human anatomy and how each step impacts it. However, it is not always easy to identify a sharp division between *surgical commonsense* and *medical commonsense*, since a combination of both is necessary to specialize a well-defined textbook procedure into the real intervention.

To achieve higher autonomy, an autonomous robotic surgical system must be able to make autonomous decisions related to the complete procedure. It will generate, select, execute, and monitor a surgical plan, adapt it to different anatomies, and react to unexpected situations. Enhanced sensing, situation awareness, and reasoning technologies are key to achieving such capabilities [5], together with a broader commonsense to properly assess and react to the situation. While *medical commonsense* would allow reaching high autonomy (level 4), fully autonomous systems (level 5) would need the integration of *general commonsense* because decisions would be required that go outside the medical field, such as lifting and moving objects, turning on and off devices and lights, understanding human emotions.

8.4 Conclusion

As long as surgical robots maintain an assistive role, they can rely on the commonsense of the operating surgeons. As soon as they become aids or surgical colleagues, they must be able to perform commonsense reasoning on their own, making it crucial

to understand how to deal with this kind of knowledge. The map between commonsense types and autonomy levels proposed in this chapter aims to make the problem more tractable and encourage researchers to fill the scientific and technological gaps related to commonsense knowledge and reasoning.

Data-driven deep learning algorithms, such as large language models and generative language models are a promising approach to embedding commonsense into a process since data implicitly encode common sense knowledge. However, this commonsense is neither explicitly formalized nor identified, thus process reasoning is not directly explainable to the user, and it violates also the upcoming regulations on using artificial intelligence methods in safety-critical systems. For these reasons, we feel that commonsense in surgical robotics will be an interesting research direction to be investigated, as discussed in 10.2.

Part III

Procedural knowledge from kinematic data

While the previous part dealt with procedural knowledge extraction from text, this part presents our contribution to procedural knowledge extraction from kinematic data recorded during the execution of a surgical task performed by an expert surgeon. In particular, a set of novel joint-space orientation-based features are designed and used with supervised machine learning to recognize surgical gestures, i.e., procedural knowledge at a lower granularity level.

9

Procedural knowledge understanding from kinematic data

"Teaching is the highest form of understanding."

Aristotle

9.1 Introduction

In the previous part of this thesis, we investigated the possibility of automatically understanding the robotic-surgery literature by exploiting pre-trained Transformer-based language models. These approaches aim to extract a plan from textual descriptions. Books, academic papers, and operative surgical guidelines are written by domain experts and thus are highly reliable. Nevertheless, as stated in Chapter 8, not all operative information is written but has to be learned from experience: much is unsaid and falls within the sphere of surgical, medical, and general commonsense. One example of commonsense knowledge not explicitly explained in textbooks is the guidelines for performing surgical gestures, i.e. the best kinematic movements of the intervention in a low-level granularity. We define "surgical competence" as the skill level required to safely perform a surgical procedure [196]. The surgical competence, in a specific phase of the intervention, comprises factors such as the velocity with which to move the instruments, the instruments' motion radius, the force to be applied in lifting or pulling a tissue, the optimal orientation of the instruments to avoid collisions with other objects (other operating instruments or anatomy), or the position of the endoscopic camera which allows the best view of the operative scene. These elements cannot be learned from textbooks but only from practical experience. Numerous studies show that the surgeon's expertise directly impacts the patient's post-operation health status [15, 196].

The same principle will be applied to autonomous robots: to guarantee a safe intervention, they must know how to map high-level textual instructions or concepts with optimal kinematic movements. Numerous practical curricula exist related to robotic surgery training aimed at teaching the best kinematic movements in a given situation. However, standardized training protocol has yet to be defined and validated [197]. In this chapter, we propose qualitative metrics to describe how a gesture, i.e. low-level kinematic surgical movements, must be performed, a precursor task, together with that of gesture recognition for achieving the goal of objective evaluation of surgical skills [196]. This chapter, therefore, considers the problem of the surgical gesture classification task. From spans of kinematic data, the goal is to label them with the proper textual description by choosing the best set of features able to describe the movement. The list of labels is a-priori defined.

State-of-the-art research has proposed and validated metrics based on the motion of instruments to use as features for automatically classifying surgical gestures [41]. Most have been derived from those developed for standard minimally invasive surgery. They are based only on Cartesian space motion analysis, mainly end-effector velocities and accelerations or total distance [42]. Other proposed metrics are based on tools orientation [41] or forces applied during gripping of interaction with the environment [43]. However, the computation of these metrics requires access to low-level surgical robotics kinematic data, which in the past was challenging to obtain due to manufacturer trade secrets and user/patient privacy. This fact limited the development and exploitation of such advanced metrics. The limited number of publicly available datasets for classifying surgical gesture metrics confirms this difficulty. This situation has recently changed thanks to the introduction of research platforms, such as DaVinci Research Kit (dVRK) [198] and Raven II [199], which enables the acquisition of low-level kinematics and status variables.

In this chapter, we present a series of orientation-based metrics that can be used together with the more traditional Cartesian-based metrics to objectively indicate how a surgical gesture should be performed. These metrics, calculated in Cartesian and joint space, are used in this work as input features to an automatic classification algorithm. Since existing open source datasets, such as JIGSAWS [39], only present information in the Cartesian space, we also introduce a novel surgical dataset containing information in both spaces, Cartesian and joints. This dataset lets us objectively describe how a surgical gesture should be performed. The experimental results show that applying metrics in the joint space significantly improves the automatic classification results compared

to those obtained by applying the metrics to the Cartesian space only, as described by [41].

9.2 Method

In 9.2.1, the novel dataset is described, together with temporal and spacial calibrations techniques used. The novel joints-space orientation metrics are presented in 9.2.2 while 9.2.3 presents the evaluation strategy.

9.2.1 The new dataset

The structure of the DaVinci slave robotic manipulator used to do experimental validation is described in Figure 9.1. It can be divided into two parts: the *Base Unit*, i.e. the first three joints of the robot, and the *Instrument Unit*, i.e. the last three. The acquired dataset consists of 42 suturing trials performed by a single expert user. The dominant right-hand expert has over 50 hours of experience with the DaVinci surgical robotic system. Trials are divided into two sets: the first 20 use a different phantom position than the last 20, where the phantom is turned 45 degrees clockwise. Trials 21 and 42 contain not the suturing task but two helpful procedures for spatial calibration. Figure 9.2 illustrates the training phantom used for the experimental validation and the standard reference system defined. Each trial is executed in a different section of the phantom, represented by letters A, B, C, and D. The trials follow lettering ordering (clockwise ordering starting from the vertical section). Each trial consists of a 3-pass suturing task executed with a 1/2 circle suture needle following reference points present in the phantom.

Each trial consists of 10 Comma Separated Value (CSV) files containing raw kinematic data (time, position, and orientation) and two videos reproducing the surgical scene captured by two endoscopic cameras. Files contained in the dataset are listed below:

- `(ECM|PSM1|PSM2|MTML|MTMR) _position_Cartesian_current.csv`: they contain temporal information about position and orientation in the cartesian 3D space of the ECM, PSM1, PSM2, MTML, and MTMR respectively [1]

[1] ECM: Endoscopic Camera Manipulator; PSM: Patient Side Manipulator; MTML: Master Tool Manipulator Left; MTMR: Master Tool Manipulator Right.

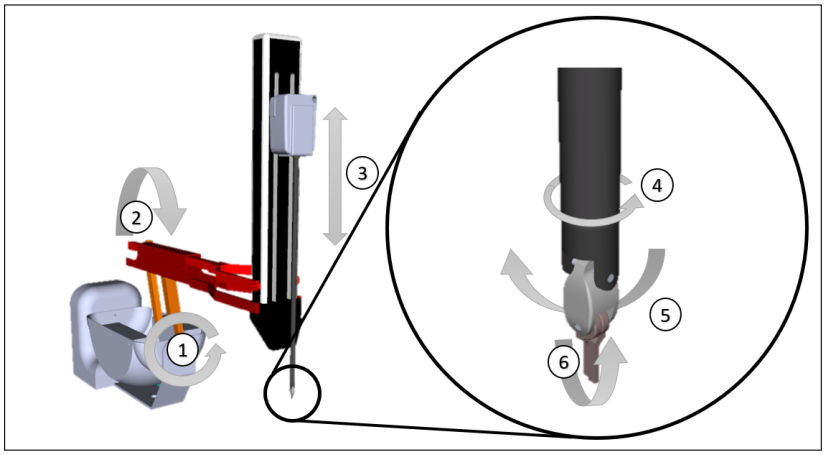


Fig. 9.1: Schematic representation of the considered robotic manipulator with joints configuration: Base unit (left) and Instrument unit (right).

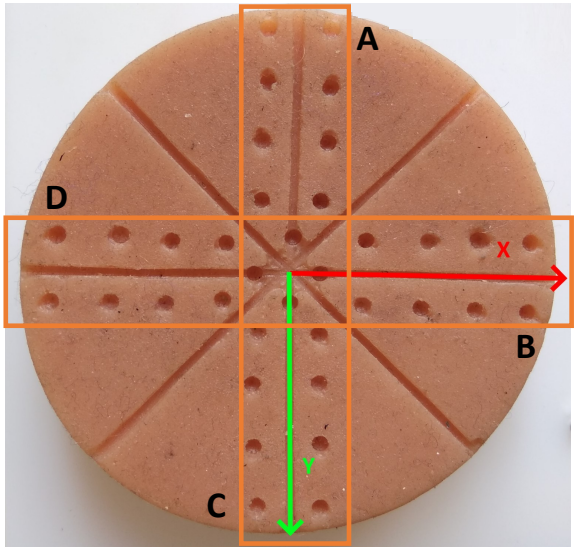


Fig. 9.2: Axes of the phantom (z-axis outward). Orange boxes represent the different sections of the phantom used for performing the different trials. See text for more details.

- `(ECM|PSM1|PSM2|MTML|MTMR) _state_joint_current.csv`: contain temporal information about position, velocity, and effort of each joint for ECM, PSM1, PSM2, MTML, and MTMR respectively. joints are: *outer-yaw, outer-pitch, insertion, outer-roll, outer-wrist-pitch e outer-wrist-yaw*.

The Cartesian position and the orientation of MTM and PSM are calculated starting from the joint angles using direct kinematics. Velocities in the joint space are computed directly by the dVRK robot controller. Acceleration and jerks are instead calculated with numerical derivation using a finite element method. Finally, we performed a spatial and temporal calibration of the robot's kinematics to have a single reference system representing all the robot's components.

Temporal Calibration

During the surgical task, MTM, PSM, and ECM kinematics were captured, and videos were recorded using two cameras (Left and Right): kinematics are gathered with a frequency of 100Hz (every 10 ms), but the video frames are updated at 25Hz (40 ms). Therefore, kinematics need to be synchronized to the video frames. Consequently, the description at a specific time t obtained by kinematic analysis is inconsistent with the same time t by video analysis. To synchronize the kinematics with video frames temporally is necessary to map each frame to the corresponding kinematic representation.

To solve the problem, we calculated the instant in which each task starts by analyzing both kinematics (obtaining t_{0_kin}) and video frames (obtaining t_{0_video}). For kinematics, we calculate the Euclidean distance in 3D space traveled by PSMs between two successive updates, assuming that without any movement, kinematics always return approximately the same value. The value will not be precisely zero due to noise in the sensors. When at time t the returned distance is higher than an experimentally estimated threshold, we assume that $t = t_{0\_kin}$. For videos, we did similarity analysis between adjacent frames to detect movement; the time when the first motion is visible from the video is t_{0_video}. For similarity analysis, we explored two alternatives: MSE (Mean Square Error) [200] metric and SSIM (Structural Similarity) [200] metric. In this application, the first metric provided more accurate results; thus, we used it in the final synchronization. [201, 202, 203] prove the effectiveness of the MSE metric for motion detection. Given t_{0_kin} and t_{0_video} the desynchronization between kinematics and video is expressed as:

$$\Delta t\_psm = t_{0\_kin} - t_{0\_video}$$

Then, the initial synchronized time is:

$$t_{0\_synch} = t_{0\_video} + \Delta t\_psm$$

A further problem arises: as mentioned above kinematics sampling rate is greater than the frames sampling rate, so t_{0_synch} found is just a fictitious timestamp that will not exactly match any real timestamp: we decided to associate it with the nearest (but not future) timestamp. Figure 9.3 illustrates the temporal calibration problem.

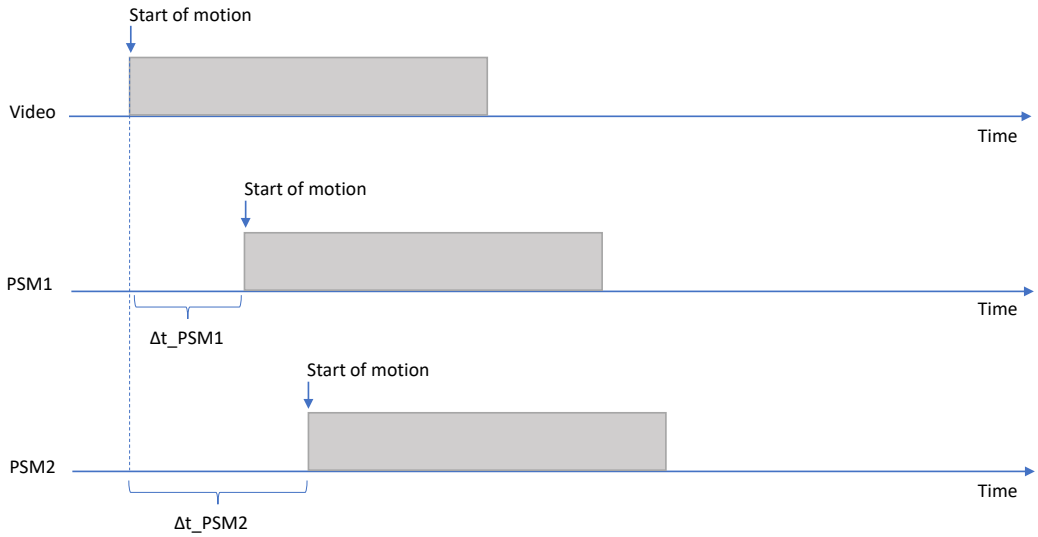


Fig. 9.3: Diagram representing the temporal calibration problem with different data streams and temporal offset considered in this work.

Spatial Calibration

PSM1 and PSM2 have independent spatial reference frames; the task is to rotate and translate frames to uniform them in a unique reference frame named `world`. After that, `world` has to be mapped into the `camera`'s reference space to have a uniform spatial view of the scene. The `world` is the phantom where the suturing task is executed; on it, nine fiducials (see Figure 9.4) are selected, and for each, the Cartesian position is measured. The goal is to align the two sets of 3D points by finding the best rotation and translation. In particular, as shape and size are preserved during rotation and translation, a Euclidean transformation is used, as described in [43, 204].

With this procedure, all PSMs have been mapped to `world`; then, `world` has to be mapped to the `camera` space. The goal is to find the pose of an object having a calibrated camera, locations of n 3D points of an object, and the corresponding 2D projections. This problem is known in the literature as *Perspective-n-Point problem* [205], and we solved it using functions provided by the OpenCV library.

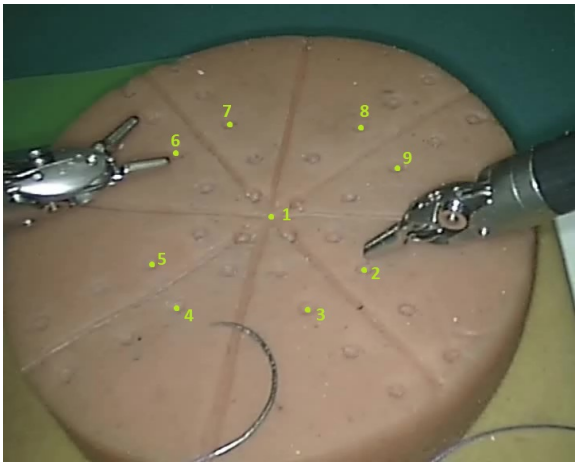


Fig. 9.4: Right camera endoscope image extracted from trial 22 showing the rotated positioning of the phantom. Numbered points indicate ordered fiducials used for spatial calibration

9.2.2 Metrics

Given the above dataset, we manually divided it into different temporal segments corresponding to surgical actions or gestures. We followed the guidelines in the literature (JIGSAWS convention [39]), to which extensions were made to consider using both hands. Table 9.1 summarizes the elementary sub-operations with corresponding annotation labels (in bold our extension to JIGSAWS nomenclature).

For each segment, the metrics below described are computed. They are an extension of those introduced in [206] to consider both the Cartesian and the joint spaces.

The first straightforward metric is *Task Time*, defined as:

$$T = t_{I_{i+1}} - t_{I_i} \tag{9.1}$$

Table 9.1: Labels used for gesture annotation in the proposed dataset. Bold font indicates additional label introduced with respect to original JIGSAWS convention

Gesture	**Description**
G1	Reaching for needle with right hand
G2	Positioning needle with right hand
G3	Pushing needle through tissue with right hand
G4	Transferring needle from left to right
G5	Moving to center with needle in grip
G6	Pulling suture with left hand
G7	Pulling suture with right hand
G8	Orienting needle with right hand
G12	Reaching for needle with left hand
G15	Pulling suture with both hands
G20	Positioning needle with left hand
G21	Pushing needle through tissue with left hand
G22	Transferring needle from right to left
G23	Orienting needle with left hand

where $t_{I_{i+1}}$ and t_{I_i} are the timestamps corresponding to the beginning of segment number i and $i+1$, respectively.

Total distance traveled between two successive frames is defined as:

$$\Delta d_{i,i+1} = \|[\Delta x_i, \Delta y_i, \Delta z_i]\| \tag{9.2}$$

where Δx_i, Δy_i, Δz_i are the differences between frame i and $i+1$ with respect to x, y, z positions, respectively and $\|\cdot\|$ is the Euclidean norm. Using $\Delta d_{j,j+1}$ we can define *Path Length* as:

$$P = \sum_{j=1}^{N-1} \Delta d_{j,j+1} \tag{9.3}$$

where N is the number of samples composing the gesture.

In the joint space, the *Angular Displacement joint* (ADIJ) metric for the joint k is defined as:

$$ADIJ_k = \sum_{i=1}^{N-1} |\Delta\Theta_{i+1,i}| \tag{9.4}$$

The angle $\Delta\Theta_{i+1,i}$ represents the orientation change between pairs of consecutive samples for the joint k.

In the joint space, the *Time Angular Displacement joint* (TADJ) for the joint k is defined as:

$$TADJ_k = \frac{1}{T} \sum_{i=1}^{N-1} |\Delta\Theta_{i,i+1}| \tag{9.5}$$

where T is the duration of the surgical action and k is the joint number.
Finally, the metric *Rate Of Change Joint* (ROCJ) is defined as:

$$ROCJ_k = \frac{1}{N-1} \sum_{i=1}^{N-1} \omega_i \tag{9.6}$$

where ω_i is the angular speed of the joint number k in the frame i and N is the number of samples.

In addition, joint data contains information about joint effort (e.g. joint motor current, joint force, or torque) as a scalar value τ_i. We therefore introduce the *mean effort* (MEJ), which represents how the joint k interacts with the environment:

$$MEJ_k = \frac{1}{N-1} \sum_{i=1}^{N-1} \tau_i \tag{9.7}$$

It is possible to extend these formulas to consider the average value of metrics on all joints of the base/instrument part of the robot described in Figure 9.3. Let M_k be a metric calculated on a general joint k (M_k can be $ADIJ$, $TADJ$, $ROCJ$ or MEJ). The value of this metric on joints of the base unit is expressible as:

$$M_{base\_unit} = \frac{1}{3} \sum_{k=1}^{3} M_k \tag{9.8}$$

while the value of this metric on joints of the instrument unit is expressible as:

$$M_{inst\_unit} = \frac{1}{3} \sum_{k=4}^{6} M_k \tag{9.9}$$

The dataset we contributed is the first one containing information on the joint space, thus making the calculation of joint space metrics feasible.

9.2.3 Automatic classification

To further investigate the ability of the previously described metrics to discriminate gestures, we trained a model classifier that, given a set of metrics, automatically recognizes the gesture. We use the random forest classifier described in 2.5 implemented from the `sklean` library for the following reasons [207, 208]:

- it shows the highest performance when applied on the general dataset without hyper tuning parameters that were outside the scope of this article [209];
- it has improved explainability since it is possible to characterize the Mean Decrease Impurity (MDI) to state the variable's importance [210];

Due to the high imbalances in the number of gestures reported in Figure 9.5 we validate our classifier using Stratified k Fold(SKF) methodology. In SKF cross-validation, the folds are created in a way that they contain approximately the same proportion of labels as the original dataset. Since the lowest number of samples for a gesture is $n = 5$, we create 5 test and train sets. This was done to maintain nearly 20% of samples for testing and guarantee the presence of each gesture in the splits. Performance is evaluated by using the metrics defined in 2.5.3.

9.3 Results and discussions

9.3.1 Temporal and Spatial calibration

The mean temporal desynchronization between video frames and PSMs is $140ms$. The mean error for spatial calibration is 1.5 mm for PSM1 and 1.6 mm for PSM2.

9.3.2 Automatic classification of surgical gestures

This section reports the results obtained from the automatic classification of surgical gestures using the metrics presented in the section 9.2.2 as features. Specifically, we analyzed the contribution of pose (distance and acceleration), orientation (cartesian space and joint space), and effort (joint space) metrics to gesture classification, observing changes in accuracy performance.

Fig. 9.5: Histogram for gestures distribution, considering only gesture labels annotated in the proposed dataset.

Table 9.2: Average classification accuracy, considering the different subset of metrics considered for the experimental evaluation

Features	**Average Accuracy**
Joint space	83.01%
Cartesian space	75.27%
All metrics	**86.51%**

In table 9.2, we report the mean accuracy obtained using Stratified k Fold ($k = 5$) methodology on three metrics groups. The highest average accuracy, with a score of 86.51%, is obtained using cartesian and joint space. In Figure 9.6, we show that few features are highly discriminative for the dataset since the average accuracy increases quickly. The ten most important metrics used for automatic classification are reported below:

1. distance on z-axis of PSM1;
2. acceleration on z-axis of PSM1;
3. distance on z-axis of PSM2;

Table 9.3: Precision, recall and F-Score of considered surgical gestures

Gestures	Cartesian space			Joints space			Cartesian + Joint		
	P	R	F1	P	R	F1	P	R	F1
G1	0.715	0.643	0.664	0.786	**0.836**	0.799	**0.856**	**0.836**	**0.836**
G2	**0.903**	**0.920**	**0.904**	0.810	0.900	0.849	0.876	0.900	0.881
G3	0.853	0.880	0.860	0.833	0.9200	0.873	**0.860**	**0.920**	**0.887**
G4	0.000	0.000	0.000	0.000	0.0000	0.000	0.000	0.000	0.000
G5	0.650	0.646	0.634	0.675	0.8143	0.726	**0.784**	**0.836**	**0.791**
G6	0.895	0.943	0.912	**0.949**	**0.971**	**0.960**	0.909	**0.971**	0.937
G7	0.805	0.946	0.866	**0.883**	0.975	0.925	**0.883**	**1.00**	**0.937**
G8	0.627	0.652	0.622	0.799	0.724	0.745	**0.844**	**0.843**	**0.836**
G12	0.785	0.757	0.758	0.893	**0.893**	**0.889**	**0.910**	0.868	0.886
G15	0.733	**0.550**	**0.625**	0.500	0.250	0.327	**0.833**	0.417	0.534
G20	0.914	0.750	0.759	**0.950**	0.800	0.855	0.920	**0.950**	**0.927**
G21	**1.00**	0.850	0.880	**1.000**	**0.900**	**0.933**	**1.000**	**0.900**	**0.933**
G22	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
G23	0.581	0.686	0.611	0.721	0.771	0.722	**0.792**	**0.857**	**0.794**

4. acceleration on z-axis of PSM2;
5. **mean effort on sixth joint of PSM2**;
6. **mean effort on sixth joint of PSM1**;
7. **mean effort on PSM2's Instrument Unit**;
8. **angular Displacement of PSM2's fifth joint**;
9. rate of change of PSM1;
10. **angular Displacement of PSM2's forth joint**.

Five (shown in bold) out of the ten most important features are from joint space and six of them are orientation-based features.

Furthermore, the joint space analysis (made possible by the dataset we introduced) improves the classification quality. Also, the joints' sub-division in two classes (base unit and instrument unit) allows for obtaining important features for the classification. Table 9.3 reports Precision, Recall, and F-Score for each gesture and three groups of metrics: cartesian space metrics, joints space metrics, and combined analysis of cartesian and joint metrics. Table 9.3 shows that some gestures have better metric scores in the proposed joint space, and most of the gestures are better recognized in the com-

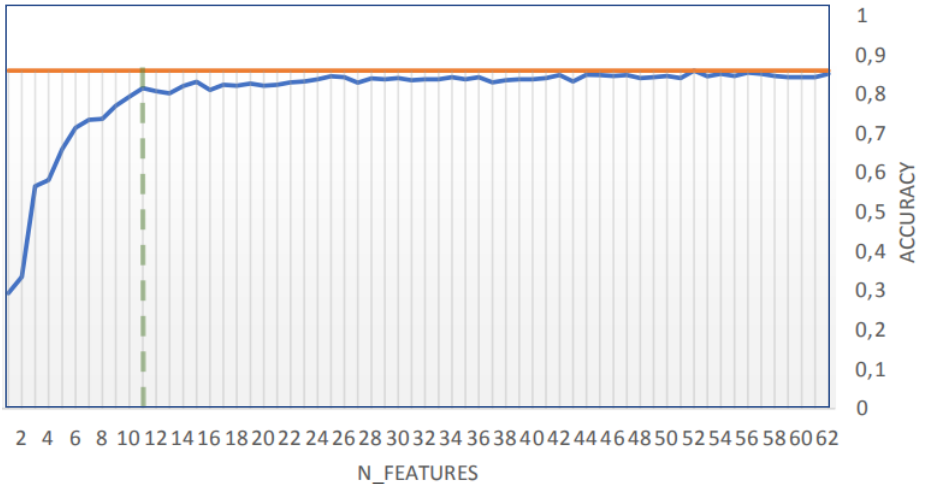


Fig. 9.6: Accuracy curve considering an increasing number of features.

bined cartesian and joint space. Our hypothesis that joint space helps to capture more information and improves the performance of the model is validated: our proposed orientation-based metrics in joint space are a good set of features that can be used in developing an automated classification of surgical gestures.

The classification method, however, fails for some gestures, like, for example, for G4 (*Transferring needle from left to right*) and G22 (*Transferring needle from right to left*). The reason is that the number of occurrences of such gestures is very low: 4 occurrences for G4 and six occurrences for G22. Another reason could be related to the intrinsic similarity of G4 and G22: from the kinematic variable, it is difficult to estimate which instrument is holding the needle and therefore distinguish between these two gestures.

9.4 Conclusion

With the work presented in this chapter, we addressed the lack in the literature about using orientation-based metrics in cartesian and joint space to objectively describe the characteristics of a surgical gesture composing the surgery. The effectiveness of the metrics has been validated using them as features for an automatic classification algorithm. Some surgical sub-operations can be well described using metrics based on cartesian position, while others can be described using orientation-based metrics instead. We have also introduced a new dataset containing cartesian and joint space

information. It was used to make experimental validation of the proposed metrics. In conclusion, surgical competence, which refers to the skill level required to perform a surgical procedure safely, involves factors such as the speed of instrument movement, range of action, force applied, optimal instrument orientation, and camera position. Practical experience is required to learn how to perform surgical gestures expertly, and it cannot be learned from textbooks. The combination of notional and practical knowledge will allow the development of autonomous robotics surgical systems, as suggested in Chapter 8.

Part IV

Final remarks

This part summarizes the contributions of the first and second parts of this thesis and discusses obtained results. Finally, it proposes related future research directions.

10

Conclusions, limitations and future works

"End? No, the journey doesn't end here"

Gandalf in "The Lord of the Ring",
J.R.R. Tolkien

10.1 Conclusions

The development of autonomous robotic surgical systems is a rapidly growing research field that holds tremendous promise for the future of medicine. Research on autonomous robots has demonstrated that, nowadays, the real core of the research is in automatic knowledge acquisition and management. For these systems to operate safely and expertly, they must be built on a solid foundation of surgical knowledge. Actually, domain experts manually encode this knowledge in ontologies or pre-defined logical instructions. This manual process is time-consuming and requires surgery and computer science expertise, making it a bottleneck in developing autonomous systems. Furthermore, this manually encoded knowledge may not cover all possible complications and cases that can arise during surgery but which are instead documented in textbooks and in surgical case reports. To overcome these limitations, this thesis investigated automatic knowledge acquisition and management possibilities. This involves extracting knowledge from external resources, such as free-text books, academic papers, written tutorials, and kinematic data captured during surgical procedures.

The first part of this thesis is dedicated to procedural knowledge extraction from as-is textbooks and academic papers describing how to perform a given surgery. First, in Chapter 3 we pre-trained a novel language model on surgical literature, named SURGICBERTA. Then, this model was used, together with other state-of-the-art approaches to

detect sentences containing procedural information in robotic surgery in Chapter 4. To do so, we framed the problem as a sentence classification task, solved via supervised machine learning. To train and test those models, we collected a novel dataset containing sentences of the robotic-surgery domain, manually annotated as procedural or non-procedural, accordingly to their content. Then, once filtered out a set of procedural sentences and discarded all the others, we faced the problem of procedural knowledge extraction from them in Chapters 5-7. We solved this problem by exploiting language models and fine-tuning them for the SRL task. In particular, since no training and testing material were available, we developed an ad-hoc dataset in Chapter 5: following the PropBank way of annotating semantic information, we organized the work in two parts: the first is defining important information in robotic surgery to annotate surgical sentences, while the second is the actual annotation. Exploiting the obtained dataset and fine-tuning the pre-trained SURGICBERTA language model developed in Chapter 3, we obtained an SRL model able to recognize and discriminate surgical content in procedural sentences in Chapter 6. We finally exploited SRL in some simple use cases to empirically investigate the possibility of translating natural language descriptions, under some language constraints, into logical rules in Chapter 7. Reflections made on the first part's chapters convinced us that some information needed to automatize a robotic-surgery intervention is missing in textbooks because it belongs to what we name commonsense knowledge that can only be acquired by practical experience. To stimulate future research in this direction, we so proposed a mapping between the levels of autonomous surgical systems and the kind of required knowledge in Chapter 8.

Finally, in the second part of this thesis, we investigated the possibility of acquiring knowledge directly from kinematic data, in particular, to recognize how a surgical action expertly executed can be described with joint-space orientation-based metrics.

We believe this thesis has the potential to pave the way for several research directions as it is the first to propose using the information written in textbooks to automate the surgical operative process. Obviously, there is still much to be done: limitations and possible future works are discussed in Section 10.2.

10.2 Limitations and future works

This section briefly discusses the main limitations of this thesis and presents future work projects to improve or enrich the current analysis.

Chapter 3 presented SURGICBERTA, the pre-trained language model for surgical language. In developing it, we collected books and academic papers from sources that our universities have free access to. Anyway, by doing so, we may therefore have excluded some valid resources. Furthermore, the choice of various texts is not balanced by surgical domain and other subdomains may not have been included at all: testing SURGICBERTA in surgical domains different from the ones it was trained could strengthen the analysis. Also, SURGICBERTA is currently trained only with English-language texts, but a lot of written material is also available in other languages: including them may increase its usability. In conclusion, enlarging and enriching the dataset used to develop SURGICBERTA in these ways could increase model performance. Chapter 4 presented a method for detecting procedural sentences in surgical books and academic papers. The method assumes that procedural information is present only in sentences describing at least one action to perform. However, this could be simplistic because some procedural details can also be implicitly present in sentences describing the background or the features of the procedure. The detection of procedural knowledge hidden in other types of sentences would therefore require further research. Also here, enlarging the dataset with novel surgical domains may be useful to strengthen the analysis in future work. Chapters 5 and 6 presented an annotated dataset and a method for extracting procedural entities from sentences. The proposed dataset and the described method also work at the sentence level, following the standard SRL approach. While this is correct in most cases, there may be procedural elements of the same action described in different sentences. The proposed dataset could therefore be enriched by also annotating these cases and the method could be perfected to identify and link them correctly. Furthermore, the same considerations done for the dataset used to develop SURGICBERTA can be applied: extending the dataset and the models in a multilingual scenario would be undoubtedly helpful in making these tools more accessible and usable by anyone. Chapter 7 presented a preliminary pipeline for extracting procedural workflows from robotic-surgery procedural descriptions. This method could be improved by using the models proposed in previous chapters, by further investigating the language variability, and by exploring similar works in completely different domains, such as business processes [28]: this could create an interesting synergy between the two fields. Furthermore, developing a surgical WordNet containing surgical synsets or enriching RSPF with surgical synonyms can also be helpful to deal with language variability. Chapter 8 presented one of the problems related to procedural knowledge extraction from written texts: the lack of commonsense-related information, often not explicit in

written material thought for humans: this presents the problem and possible solutions that can be explored with future research. Among all, exploiting large language models, as demonstrated in [211] can be a viable solution. These models can be prompted to extract information, similarly to what we did in Chapter 3 with the qualitative evaluation. Commonsense knowledge can also be inferred from kinematic data captured during the surgery, so research can be conducted to extend that proposed in Chapter 9. While commonsense skills can be learned implicitly from kinematic or video data, the problem of confirming the appropriateness and completeness of the available datasets remains. In fact, robust learning is feasible only by having large amounts of available data, which is difficult to obtain in the surgical domain. Finally, Chapter 9 presented one contribution in the bottom-up direction. The analysis could be enriched by adopting recent deep-learning methods, thus avoiding an explicit feature engineering task. Furthermore, the combination of the methods described in the first part with this contribution remains an open task.

Additional related chapter-independent future research directions follow. As illustrated in Figure 1.2, the thesis's first part is composed of two stages: the first is related to procedural sentence detection, while the second deals with procedural elements understanding from procedural sentences. The system is now developed as a two-stage pipeline, tackling the two tasks separately. This requires the training and testing of two different models. To overcome this issue, we plan to develop a model to detect procedural sentences and extract procedural elements (with SRL) from them in an end-to-end fashion. This will simplify both the training and final evaluation, helping to avoid the propagation of errors from the first to the second stages.

Another related research direction will be comparing how different surgeons with different expertise levels write the same procedure, e.g. with greater or lesser detail. Developed models and resources can simplify the translation of natural surgical language to logical language, independently from the surgeon's expertise and background. Using these resources to translate equivalent descriptions of the same procedure into a unique, simplified form can also help to certify the obtained logical plan and limit the number of security assertions to develop. At the same time, we plan to develop a text editor that assists the surgeon in writing procedural descriptions providing in real-time the interpretation the model gave of the given sentence and eventually suggesting changes. In this phase, realistic intervention descriptions will be used, requiring the use of SURGICBERTA. While in this thesis, we used pre-trained language models for information extraction, thus only using the encoder part of the Transformer architecture, an

interesting application will be to use also the decoder to generate output text. This application is especially interesting after the release of large language models such as GPT [212], GLaM [213], LaMDA [214], Gopher [215], Megatron-Turing NLG [216], and PaLM [217]. In the context of this thesis, the encoder part could be used to extract information from written materials, and the decoder part to produce the corresponding logic plan.

Knowledge-based approaches can also be implemented relying on ontologies, which provide a suitable way to represent entities and relations. Surgical ontologies like OntoSPM [184] could be extended to integrate, or aligned with, broader knowledge bases that include general commonsense knowledge and reasoning, like CyC and ConceptNet [218]. A possible path would be investigating ways to merge knowledge bases that are developed independently for different purposes[219], thus obtaining a representation that includes procedural knowledge, surgery-specific commonsense, and general commonsense.

Furthermore, combining bottom-up with top-down approaches in a unique knowledge based model will be extremely interesting and crucial for obtaining higher levels of autonomy in robotic surgery.

Finally, this thesis lays the groundwork for extracting procedural knowledge from texts in a format that autonomous surgical robots could understand and use to define an intervention plan. A high-level limitation is, however, that its practical validation remains nowadays challenging because there are still no robots capable of performing similar operations autonomously due to technological and ethical reasons. For this point, all related future research must therefore be multidisciplinary to address the complexity of the topic in the best possible way.

References

[1] Mako. https://www.stryker.com/us/en/portfolios/orthopaedics/joint-replacement/mako-robotic-arm-assisted-surgery.html, 2023. Accessed: 2023-02-15.

[2] Rosa. https://www.zimmerbiomet.com/en/products-and-solutions/specialties/knee/rosa--knee-system.html, 2023. Accessed: 2023-02-15.

[3] Mazorx. https://www.medtronic.com/it-it/operatori-sanitari/therapies-procedures/spinal-orthopaedic/spine-robotics.html, 2023. Accessed: 2023-02-15.

[4] Da-vinci. https://www.intuitive.com/en-us, 2023. Accessed: 2023-02-15.

[5] Aleks Attanasio, Bruno Scaglioni, Elena De Momi, Paolo Fiorini, and Pietro Valdastri. Autonomy in surgical robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:651–679, 2021.

[6] Claudia D'Ettorre, Andrea Mariani, Agostino Stilli, Ferdinando Rodriguez y Baena, Pietro Valdastri, Anton Deguet, Peter Kazanzides, Russell H. Taylor, Gregory S. Fischer, Simon P. DiMaio, Arianna Menciassi, and Danail Stoyanov. Accelerating surgical robotics research: A review of 10 years with the da vinci research kit. *IEEE Robotics Autom. Mag.*, 28(4):56–78, 2021. doi: 10.1109/MRA.2021.3101646. URL https://doi.org/10.1109/MRA.2021.3101646.

[7] Michael Yip and Nikhil Das. Robot autonomy for surgery. *The Encyclopedia of Medical Robotics*, pages 281–313, 2018.

[8] R. H. Taylor, A. Menciassi, G. Fichtinger, and P. Dario. *Medical Robotics and Computer-Integrated Surgery*. Springer, 2008.

[9] J. Heemskerk, R. Zandebergen, G. Maessen, W. Greve, and N. Bouvy. Advantages of advanced laparoscopic systems. *Surgical Endoscopy*, pages 730–733, 2006. doi:

doi:10.1007/s00464-005-0456-3.

[10] H. Kang and J. Wen. Robotic assistants aid surgeons during minimally invasive procedures. *IEEE Engineering in Medicine and Biology Conference*, pages 94–104, February 2001.

[11] K. Chun, B. Schmidt, B. Kokturk, R. Tilz, A. Furnkranz, M. Konstantinidou, E. Wissner, A. Metzner, F. Ouyang, and K. Kuck. Catheter ablation: New developments in robotics. *Herz Kardiovaskuläre Erkrankungen*, page 586–589, 2008.

[12] B. Davies. A review of robotics in surgery. *In Proceedings of the Institution of Mechanical Engineers*, page 129–140, 2000.

[13] SAE international. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles, 2016.

[14] Guang-Zhong Yang, James Cambias, Kevin Cleary, Eric Daimler, James Drake, Pierre E Dupont, Nobuhiko Hata, Peter Kazanzides, Sylvain Martel, Rajni V Patel, et al. Medical robotics—regulatory, ethical, and legal considerations for increasing levels of autonomy, 2017.

[15] Paolo Fiorini, Kenneth Y. Goldberg, Yunhui Liu, and Russell H. Taylor. Concepts and trends in autonomy for robot-assisted surgery. *Proc. IEEE*, 110(7): 993–1011, 2022. doi: 10.1109/JPROC.2022.3176828. URL https://doi.org/10.1109/JPROC.2022.3176828.

[16] Michele Ginesi, Daniele Meli, Hirenkumar Nakawala, Andrea Roberti, and Paolo Fiorini. A knowledge-based framework for task automation in surgery. In *19th International Conference on Advanced Robotics, ICAR 2019, Belo Horizonte, Brazil, December 2-6, 2019*, pages 37–42. IEEE, 2019. doi: 10.1109/ICAR46387.2019.8981619. URL https://doi.org/10.1109/ICAR46387.2019.8981619.

[17] Daniele Meli, Mohan Sridharan, and Paolo Fiorini. Inductive learning of answer set programs for autonomous surgical task planning. *Machine Learning*, pages 1–25, 2021.

[18] Marco Bombieri, Marco Rospocher, Diego Dall'Alba, and Paolo Fiorini. Automatic detection of procedural knowledge in robotic-assisted surgical texts. *International Journal of Computer Assisted Radiology and Surgery*, 16(8):1287 – 1295, 2021. ISSN 18616410. doi: 10.1007/s11548-021-02370-9.

[19] Florent Lalys and Pierre Jannin. Surgical process modelling: a review. *International journal of computer assisted radiology and surgery*, 9(3):495–511, 2014.

[20] Sanat Ramesh, Diego Dall'Alba, Cristians Gonzalez, Tong Yu, Pietro Mascagni, Didier Mutter, Jacques Marescaux, Paolo Fiorini, and Nicolas Padoy. Multi-task tem-

poral convolutional networks for joint recognition of surgical phases and steps in gastric bypass procedures. *Int. J. Comput. Assist. Radiol. Surg.*, 16(7):1111–1119, 2021. doi: 10.1007/s11548-021-02388-z. URL https://doi.org/10.1007/s11548-021-02388-z.

[21] Bernard Gibaud, Germain Forestier, Carolin Feldmann, Giancarlo Ferrigno, Paulo Gonçalves, Tamas Haidegger, Chantal Julliard, Darko Katić, Hannes Kenngott, Lena Maier-Hein, Keno März, Elena De Momi, Dénes Nagy, Hirenkumar Nakawala, Juliane Neumann, Thomas Neumuth, Javier Balderrama, Stefanie Speidel, Martin Wagner, and Pierre Jannin. Toward a standard ontology of surgical process models. *International Journal of Computer Assisted Radiology and Surgery*, 07 2018.

[22] Shivali Agarwal, Shubham Atreja, and Vikas Agarwal. Extracting procedural knowledge from technical documents. *arXiv preprint arXiv:2010.10156*, 2020.

[23] Chen Qian, Lijie Wen, Akhil Kumar, Leilei Lin, Li Lin, Zan Zong, Shu'ang Li, and Jianmin Wang. An approach for process model extraction by multi-grained text classification. In Schahram Dustdar, Eric Yu, Camille Salinesi, Dominique Rieu, and Vik Pant, editors, *Advanced Information Systems Engineering*, pages 268–282, Cham, 2020. Springer International Publishing.

[24] H. Yang, C. A. Aguirre, M. F. De La Torre, D. Christensen, L. Bobadilla, E. Davich, J. Roth, L. Luo, Y. Theis, A. Lam, T. Y. Han, D. Buttler, and W. H. Hsu. Pipelines for procedural information extraction from scientific literature: Towards recipes using machine learning and data science. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 41–46, 2019.

[25] Thiemo Wambsganss and Hansjörg Fromm. Mining user-generated repair instructions from automotive web communities. In *HICSS*, 2019.

[26] Abhirut Gupta, Abhay Khosla, Gautam Singh, and Gargi Dasgupta. Mining procedures from technical support documents. *arXiv:1805.09780*, 2018.

[27] Ziqi Zhang, Philip Webster, Victoria Uren, Andrea Varga, and Fabio Ciravegna. Automatically extracting procedural knowledge from instructional texts using natural language processing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 520–527, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA).

[28] Patrizio Bellan, Mauro Dragoni, and Chiara Ghidini. Extracting business process entities and relations from text using pre-trained language models and in-

context learning. In João Paulo A. Almeida, Dimka Karastoyanova, Giancarlo Guizzardi, Marco Montali, Fabrizio Maria Maggi, and Claudenir M. Fonseca, editors, *Enterprise Design, Operations, and Computing - 26th International Conference, EDOC 2022, Bozen-Bolzano, Italy, October 3-7, 2022, Proceedings*, volume 13585 of *Lecture Notes in Computer Science*, pages 182–199. Springer, 2022. doi: 10.1007/978-3-031-17604-3_11.

[29] Wenlong Huang, P. Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *ArXiv*, abs/2201.07207, 2022.

[30] Prasoon Goyal, Raymond J. Mooney, and Scott Niekum. Zero-shot task adaptation using natural language. *ArXiv*, abs/2106.02972, 2021.

[31] Shuyan Zhou, Pengcheng Yin, and Graham Neubig. Hierarchical control of situated agents through natural language. *ArXiv*, abs/2109.08214, 2021.

[32] Tobias Kuhn. A Survey and Classification of Controlled Natural Languages. *Computational Linguistics*, 40(1):121–170, 03 2014.

[33] Xiaopeng Zhang, Haoyu Yang, and Evangeline F. Y. Young. Attentional transfer is all you need: Technology-aware layout pattern generation. In *58th ACM/IEEE Design Automation Conference, DAC 2021, San Francisco, CA, USA, December 5-9, 2021*, pages 169–174. IEEE, 2021.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 10 2018.

[35] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012. doi: 10.2200/S00457ED1V01Y201211AIM019.

[36] Yidan Qin, Sahba Aghajani Pedram, Seyedshams Feyzabadi, Max Allan, A. Jonathan McLeod, Joel W. Burdick, and Mahdi Azizian. Temporal segmentation of surgical sub-tasks through deep learning with multiple data sources. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 371–377, 2020. doi: 10.1109/ICRA40945.2020.9196560.

[37] Yidan Qin, Seyedshams Feyzabadi, Max Allan, Joel W. Burdick, and Mahdi Azizian. davincinet: Joint prediction of motion and surgical state in robot-assisted

surgery. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2921–2928, 2020. doi: 10.1109/IROS45743.2020.9340723.

[38] Yidan Qin, Max Allan, Joel W. Burdick, and Mahdi Azizian. Autonomous hierarchical surgical state estimation during robot-assisted surgery through deep neural networks. *IEEE Robotics and Automation Letters*, 6(4):6220–6227, 2021. doi: 10.1109/LRA.2021.3091728.

[39] Narges Ahmidi, Lingling Tao, Shahin Sefati, Yixin Gao, Colin Lea, Benjamín Béjar, Luca Zappella, Sanjeev Khudanpur, René Vidal, and Gregory Hager. A dataset and benchmarks for segmentation and recognition of gestures in robotic surgery. *IEEE Transactions on Biomedical Engineering*, PP:1–1, 01 2017. doi: 10.1109/TBME.2016.2647680.

[40] Marco Bombieri, Diego Dall'Alba, Sanat Ramesh, Giovanni Menegozzo, Caitlin Schneider, and Paolo Fiorini. Joints-space metrics for automatic robotic surgical gestures classification. *IEEE International Conference on Intelligent Robots and Systems*, page 3061 – 3066, 2020. doi: 10.1109/IROS45743.2020.9341094.

[41] Yarden Sharon, Thomas S. Lendvay, and Ilana Nisky. Instrument orientation-based metrics for surgical skill evaluation in robot-assisted and open needle driving. *CoRR*, abs/1709.09452, 2017.

[42] Carol Reiley, Henry Lin, David Yuh, and Gregory Hager. Review of methods for objective surgical skill evaluation. *Surgical endoscopy*, 25:356–66, 02 2011. doi: 10.1007/s00464-010-1190-z.

[43] Paul Besl and H.D. McKay. A method for registration of 3-d shapes. ieee trans pattern anal mach intell. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14:239–256, 03 1992. doi: 10.1109/34.121791.

[44] Marco Bombieri, Marco Rospocher, Simone Paolo Ponzetto, and Paolo Fiorini. The robotic-surgery propositional bank. *Language Resources and Evaluation*, June 2023. doi: 10.1007/s10579-023-09668-x.

[45] Eleonora Tagliabue, Marco Bombieri, Paolo Fiorini, and Diego Dall'Alba. Robotic surgical systems need commonsense to achieve higher levels of autonomy. *IEEE Robotics and Automation Magazine*, 2023.

[46] Marco Bombieri, Marco Rospocher, Simone Paolo Ponzetto, and Paolo Fiorini. Machine understanding surgical actions from intervention procedure textbooks. *Computers in Biology and Medicine*, 152, 2023. doi: 10.1016/j.compbiomed.2022.106415.

[47] Marco Bombieri, Daniele Meli, Diego Dall'Alba, and Paolo Fiorini. Inductive learning of surgical task knowledge from intra-operative expert feedback. In *9th Italian Workshop on Artificial Intelligence and Robotics (AIRO)*, 2022.

[48] Marco Bombieri, Marco Rospocher, Simone Paolo Ponzetto, and Paolo Fiorini. The Robotic Surgery Procedural Framebank. In *Proceedings of the Thirteenth International Conference on Language Resources and Evaluation (LREC 2022)*, pages 3950–3959, Marseille, France, June 21-23, 2022 2022. European Language Resources Association (ELRA). ISBN 979-10-95546-72-6.

[49] Marco Bombieri, Marco Rospocher, Simone Paolo Ponzetto, and Paolo Fiorini. SurgicBERTa: A pre-trained language model for procedural surgical language. *Under review*, under review.

[50] Marco Bombieri, Daniele Meli, Diego Dall'Alba, Marco Rospocher, and Paolo Fiorini. Mapping natural language procedures descriptions to linear temporal logic templates - an application in the robotic-surgery domain. *Under review*, under review.

[51] Chia-Chien Hung, Tommaso Green, Robert Litschko, Tornike Tsereteli, Sotaro Takeshita, Marco Bombieri, Goran Glavaš, and Simone Paolo Ponzetto. Data augmentation with specialized models for cross-lingual open-retrieval question answering system. In *MIA 2022 - Workshop on Multilingual Information Access, Proceedings of the Workshop*, page 77 – 90, 2022.

[52] Christopher M. Bishop. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007. ISBN 9780387310732. URL https://www.worldcat.org/oclc/71008143.

[53] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[54] Gerald Tesauro. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Comput.*, 6(2):215–219, 1994. doi: 10.1162/neco.1994.6.2.215. URL https://doi.org/10.1162/neco.1994.6.2.215.

[55] Ozanan R. Meireles, Guy Rosman, Maria S. Altieri, Lawrence Carin, Gregory Hager, Amin Madani, Nicolas Padoy, Carla M. Pugh, Patricia Sylla, Thomas M. Ward, Daniel A. Hashimoto, Yutong Ban, Fillipo Filicori, Pietro Mascagni, John Mellinger, Christopher Schlacta, Stefanie Speidel, Thorsten Juergens, Pablo Garcia-Kilroy, Dotan Asselman, Jordan Bohnen, Rachel Ballantyne Draelos, Hans Fuchs, Ricardo Henao, Duygu Sarikaya, Christopher Boyle, Danyal Fer, Zhen Li, Arvind Ramadorai, Danail Stoyanov, Andrew Yoo, Cristians Gonzalez, Dmitry

Oleynikov, Janey Pratt, Danny Scott, Swaroop Vedula, Elan Witkowski, Takayuki Shimizu, Mark Tousignant, Dan Azagury, Flavien Bridault, Brian Dunkin, Teodor Grantcharov, Pierre Jannin, Anand Malpani, Silvana Perretta, Steven Schwaitzberg, Anthony Jarc, Kurt Landfors, Amit Mahadik, and Holly Nguyen. Sages consensus recommendations on an annotation framework for surgical video. *Surgical Endoscopy*, 35(9):4918 – 4929, 2021. doi: 10.1007/s00464-021-08578-9.

[56] Martha Palmer, Paul R. Kingsbury, and Daniel Gildea. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguistics*, 31(1):71–106, 2005. doi: 10.1162/0891201053630264.

[57] Gustavo Batista, Ronaldo Prati, and Maria-Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6:20–29, 06 2004.

[58] Soheila Saeedi, Sorayya Rezayi, Hamidreza Keshavarz, and Sharareh R. Niakan Kalhori. Mri-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques. *BMC Medical Informatics Decis. Mak.*, 23(1):16, 2023. doi: 10.1186/s12911-023-02114-6. URL `https://doi.org/10.1186/s12911-023-02114-6`.

[59] Alexander L. Pyayt, Ilya I. Mokhov, Bernhard Lang, Valeria V. Krzhizhanovskaya, and Robert J. Meijer. Machine learning methods for environmental monitoring and flood protection. *World Academy of Science, Engineering and Technology*, 78: 118 – 123, 2011.

[60] Valerio Basile, Federico Cabitza, Andrea Campagner, and Michael Fell. Toward a perspectivist turn in ground truthing for predictive computing. *CoRR*, abs/2109.04270, 2021. URL `https://arxiv.org/abs/2109.04270`.

[61] Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. The inception platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9. Association for Computational Linguistics, June 2018. URL `http://tubiblio.ulb.tu-darmstadt.de/106270/`. Event Title: The 27th International Conference on Computational Linguistics (COLING 2018).

[62] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations Session at EACL 2012*, Avignon, France, April 2012. Association for Computational Linguistics.

[63] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. doi: 10.1177/001316446002000104.

[64] Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378 – 382, 1971. doi: 10.1037/h0031619.

[65] Anthony J. Viera and Joanne M. Garrett. Understanding interobserver agreement: the kappa statistic. *Family Medicine*, 37.5:360–363, 2005.

[66] Rosario Delgado and Xavier-Andoni Tibau. Why cohen's kappa should be avoided as performance measure in classification. *PLoS ONE*, 14(9), 2019. doi: 10.1371/journal.pone.0222916.

[67] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting on Association for Computational Linguistics*, ACL '89, page 76–83, USA, 1989. Association for Computational Linguistics. doi: 10.3115/981623.981633. URL `https://doi.org/10.3115/981623.981633`.

[68] Ido Dagan, Shaul Marcus, and Shaul Markovitch. Contextual word similarity and estimation from sparse data. In *Proceedings of the 31st Annual Meeting on Association for Computational Linguistics*, ACL '93, page 164–171, USA, 1993. Association for Computational Linguistics. doi: 10.3115/981574.981596. URL `https://doi.org/10.3115/981574.981596`.

[69] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, pages 1–12, 2013.

[70] Tomás Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura, editors, *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048. ISCA, 2010.

[71] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162.

[72] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–

146, 2017. doi: 10.1162/tacl_a_00051.

[73] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. URL https://arxiv.org/abs/1910.13461.

[74] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[75] Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. From zero to hero: On the limitations of zero-shot cross-lingual transfer with multilingual transformers, 2020. URL https://arxiv.org/abs/2005.00633.

[76] Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-2304. URL https://aclanthology.org/W19-2304.

[77] Joonbo Shin, Yoonhyung Lee, and Kyomin Jung. Effective sentence scoring method using bert for speech recognition. In Wee Sun Lee and Taiji Suzuki, editors, *Proceedings of The Eleventh Asian Conference on Machine Learning*, volume 101 of *Proceedings of Machine Learning Research*, pages 1081–1093. PMLR, 17–19 Nov 2019. URL https://proceedings.mlr.press/v101/shin19a.html.

[78] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Masked language model scoring. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2699–2712. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.240. URL https://doi.org/10.18653/v1/2020.acl-main.240.

[79] Muhammad Abbas, Kamran Ali, Saleem Memon, Abdul Jamali, Saleemullah Memon, and Anees Ahmed. Multinomial naive bayes classification model for sentiment analysis. *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, 19(3):62–67, 03 2019.

[80] C. J Fillmore and C. F. Baker. A frames approach to semantic analysis. *The Oxford Handbook of Linguistic Analysis*, pages 313–340, 2009.

[81] Charles J. Fillmore, Christopher R. Johnson, and Miriam R.L. Petruck. Background to framenet. *International Journal of Lexicography*, 16(3):235 – 250, 2003. doi: 10.1093/ijl/16.3.235. URL https://www.scopus.com/inward/

record.uri?eid=2-s2.0-23844488601&doi=10.1093%2fijl%2f16.3.235&partnerID=40&md5=be5baf2bcdfe17f7ef3f1fafcc91d303. Cited by: 446.

[82] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In Stefanie Dipper, Maria Liakata, and Antonio Pareja-Lora, editors, *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, LAW-ID@ACL 2013, August 8-9, 2013, Sofia, Bulgaria*, pages 178–186. The Association for Computer Linguistics, 2013. URL https://aclanthology.org/W13-2322/.

[83] Karin Kipper Schuler. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. PhD thesis, University of Pennsylvania, 2006.

[84] Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. Semantic role labeling: An introduction to the special issue. *Comput. Linguistics*, 34(2):145–159, 2008. doi: 10.1162/coli.2008.34.2.145.

[85] Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Comput. Linguistics*, 28(3):245–288, 2002. doi: 10.1162/089120102760275983.

[86] Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1044.

[87] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5027–5038. Association for Computational Linguistics, 2018.

[88] Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online, July 2020. Association for Computational Linguistics.

[89] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In Ido Dagan and Daniel Gildea, editors, *Proceedings of the Ninth Conference on Computational Natural Language Learning, CoNLL*

2005, Ann Arbor, Michigan, USA, June 29-30, 2005, pages 152–164. ACL, 2005. URL https://aclanthology.org/W05-0620/.

[90] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. Towards robust linguistic analysis using ontonotes. In Julia Hockenmaier and Sebastian Riedel, editors, *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 143–152. ACL, 2013.

[91] Mengya Xu, Mobarakol Islam, Chwee Ming Lim, and Hongliang Ren. Learning domain adaptation with model calibration for surgical report generation in robotic surgery. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12350–12356, 2021. doi: 10.1109/ICRA48506.2021.9561569.

[92] Chen Lin, Shuai Zheng, Zhizhe Liu, Youru Li, Zhenfeng Zhu, and Yao Zhao. SGT: scene graph-guided transformer for surgical report generation. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention - MICCAI 2022 - 25th International Conference, Singapore, September 18-22, 2022, Proceedings, Part VII*, volume 13437 of *Lecture Notes in Computer Science*, pages 507–518. Springer, 2022. doi: 10.1007/978-3-031-16449-1_48. URL https://doi.org/10.1007/978-3-031-16449-1_48.

[93] Mengya Xu, Mobarakol Islam, Chwee Ming Lim, and Hongliang Ren. Class-incremental domain adaptation with smoothing and calibration for surgical report generation. In Marleen de Bruijne, Philippe C. Cattin, Stéphane Cotin, Nicolas Padoy, Stefanie Speidel, Yefeng Zheng, and Caroline Essert, editors, *Medical Image Computing and Computer Assisted Intervention - MICCAI 2021 - 24th International Conference, Strasbourg, France, September 27 - October 1, 2021, Proceedings, Part IV*, volume 12904 of *Lecture Notes in Computer Science*, pages 269–278. Springer, 2021. doi: 10.1007/978-3-030-87202-1_26. URL https://doi.org/10.1007/978-3-030-87202-1_26.

[94] Lalithkumar Seenivasan, Mobarakol Islam, Adithya K. Krishna, and Hongliang Ren. Surgical-vqa: Visual question answering in surgical scenes using transformer. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention - MICCAI 2022 - 25th International Conference, Singapore, September 18-22, 2022, Proceedings, Part VII*, volume 13437 of *Lecture Notes in Computer Science*, pages 33–

43. Springer, 2022. doi: 10.1007/978-3-031-16449-1_4. URL https://doi.org/10.1007/978-3-031-16449-1_4.

[95] Mengya Xu, Mobarakol Islam, and Hongliang Ren. Rethinking surgical captioning: End-to-end window-based MLP transformer using patches. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention - MICCAI 2022 - 25th International Conference, Singapore, September 18-22, 2022, Proceedings, Part VII*, volume 13437 of *Lecture Notes in Computer Science*, pages 376–386. Springer, 2022. doi: 10.1007/978-3-031-16449-1_36. URL https://doi.org/10.1007/978-3-031-16449-1_36.

[96] Xi Yang, Aokun Chen, Nima PourNejatian, Hoo Chang Shin, Kaleb E. Smith, Christopher Parisien, Colin Compas, Cheryl Martin, Anthony B. Costa, Mona G. Flores, Ying Zhang, Tanja Magoc, Christopher A. Harle, Gloria Lipori, Duane A. Mitchell, William R. Hogan, Elizabeth A. Shenkman, Jiang Bian, and Yonghui Wu. A large language model for electronic health records. *npj Digital Medicine*, 5 (1):194, Dec 2022. ISSN 2398-6352. doi: 10.1038/s41746-022-00742-2. URL https://doi.org/10.1038/s41746-022-00742-2.

[97] Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[98] Sicheng Zhou, Nan Wang, Liwei Wang, Hongfang Liu, and Rui Zhang. CancerBERT: a cancer domain-specific language model for extracting breast cancer phenotypes from electronic health records. *Journal of the American Medical Informatics Association*, 03 2022. ISSN 1527-974X.

[99] Kevin Xie, Ryan S Gallagher, Erin C Conrad, Chadric O Garrick, Steven N Baldassano, John M Bernabei, Peter D Galer, Nina J Ghosn, Adam S Greenblatt, Tara Jennings, Alana Kornspun, Catherine V Kulick-Soper, Jal M Panchal, Akash R Pattnaik, Brittany H Scheid, Danmeng Wei, Micah Weitzman, Ramya Muthukrishnan, Joongwon Kim, Brian Litt, Colin A Ellis, and Dan Roth. Extracting seizure frequency from epilepsy clinic notes: a machine reading approach to natural language processing. *Journal of the American Medical Informatics Association*, 29(5): 873–881, 02 2022. ISSN 1527-974X.

[100] Xi Yang, Jiang Bian, William R Hogan, and Yonghui Wu. Clinical concept extraction using transformers. *Journal of the American Medical Informatics Association*, 27(12):1935–1942, 10 2020.

[101] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li-Wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3, 2016. doi: 10.1038/sdata.2016.35.

[102] Zhengzhong Liang, Enrique Noriega-Atala, Clayton Morrison, and Mihai Surdeanu. Low resource causal event detection from biomedical literature. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[103] Sidhant Chandak, Liqing Zhang, Connor Brown, and Lifu Huang. Towards automatic curation of antibiotic resistance genes via statement extraction from scientific papers: A benchmark dataset and models. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[104] Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. Large language models encode clinical knowledge, 2022. URL `https://arxiv.org/abs/2212.13138`.

[105] Casimiro Pio Carrino, Joan Llop, Marc Pàmies, Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Joaquín Silveira-Ocampo, Alfonso Valencia, Aitor Gonzalez-Agirre, and Marta Villegas. Pretrained biomedical language models for clinical NLP in Spanish. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 193–199, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.bionlp-1.19. URL `https://aclanthology.org/2022.bionlp-1.19`.

[106] Liang Yao, Zhe Jin, Chengsheng Mao, Yin Zhang, and Yuan Luo. Traditional Chinese medicine clinical records classification with BERT and domain specific corpora. *Journal of the American Medical Informatics Association*, 26(12):1632–1636, 09 2019.

[107] Sunghwan Sohn, Yanshan Wang, Chung-Il Wi, Elizabeth A Krusemark, Euijung Ryu, Mir H Ali, Young J Juhn, and Hongfang Liu. Clinical documentation variations and NLP system portability: a case study in asthma birth cohorts across institutions. *Journal of the American Medical Informatics Association*, 25(3):353–359, 11 2017.

[108] Oliver J Bear Don't Walk IV, Tony Sun, Adler Perotte, and Noémie Elhadad. Clinically relevant pretraining is all you need. *Journal of the American Medical Informatics Association*, 28(9):1970–1976, 06 2021. ISSN 1527-974X.

[109] Ellen M. Voorhees. The TREC-8 question answering track report. In Ellen M. Voorhees and Donna K. Harman, editors, *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*, volume 500-246 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 1999. URL `http://trec.nist.gov/pubs/trec8/papers/qa_report.pdf`.

[110] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL `https://aclanthology.org/D19-1250`.

[111] S. M. Strasberg, M. Hertl, and N. J. Soper. An analysis of the problem of biliary injury during laparoscopic cholecystectomy. *Surgery Gynecology and Obstetrics*, 180(1):101–125, 1995. ISSN 1072-7515.

[112] Kahkashan Jeelani. *Surgical Anatomy of the Female Pelvis and Abdominal Wall*, page 8–14. Cambridge University Press, 2020. doi: 10.1017/9781108644396.002.

[113] Sa-kwang Song, Heung-seon Oh, Sung Hyon Myaeng, Sung-pil Choi, Hong-woo Chun, Yun-soo Choi, and Chang-hoo Jeong. Procedural knowledge extraction on medline abstracts. In Ning Zhong, Vic Callaghan, Ali A. Ghorbani, and Bin Hu, editors, *Active Media Technology*, pages 345–354, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[114] Yuman Fong, Yanghee Woo, Woo Hyung, Clayton Lau, and Vivian Strong. *The SAGES Atlas of Robotic Surgery*. Springer International Publishing, 01 2018. ISBN 978-3-319-91043-7.

[115] Inderpal S. Sarkaria and Nabil P. Rizk. Robotic-assisted minimally invasive esophagectomy: The ivor lewis approach. *Thoracic Surgery Clinics*, 24(2):211 – 222, 2014. Robotic Surgery.

[116] Alessandro Pardolesi, Luca Bertolaccini, Jury Brandolini, Filippo Gallina, Pierluigi Novellis, Giulia Veronesi, and Piergiorgio Solli. Four arms robotic-assisted pulmonary resection-right lower/middle lobectomy: How to do it. *Journal of Thoracic Disease*, 10:476–481, 01 2018.

[117] Michael A. Savitt, Guangquiang Gao, Anthony P. Furnary, Jeffrey Swanson, Hugh L. Gately, and John R. Handy. Application of robotic-assisted techniques to the surgical evaluation and treatment of the anterior mediastinum. *The Annals of Thoracic Surgery*, 79(2):450 – 455, 2005.

[118] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

[119] Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010.

[120] Kanish Shah, Henil Patel, Devanshi Sanghvi, and Manan Shah. A comparative analysis of logistic regression, random forest and knn models for the text classification. *Augmented Human Research*, 5, 12 2020.

[121] Pooja Saigal and Vaibhav Khanna. Multi-category news classification using support vector machine based classifiers. *SN Applied Sciences*, 2, 02 2020.

[122] W. P. Ramadhan, S. T. M. T. Astri Novianty, and S. T. M. T. Casi Setianingsih. Sentiment analysis using multinomial logistic regression. In *2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC)*, pages 46–49, 2017.

[123] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[124] R. Tahsin, M. H. Mozumder, S. A. Shahriyar, and M. A. Salim Mollah. A novel approach for e-mail classification using fasttext. In *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 1392–1395, 2020.

[125] A. G. D'Sa, I. Illina, and D. Fohr. Bert and fasttext embeddings for automatic detection of toxic speech. In *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)*, pages 1–5, 2020.

[126] Marco Rospocher. Explicit song lyrics detection with subword-enriched word embeddings. *Expert Systems with Applications*, 163:113749, 07 2020.

[127] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[128] Gang Liu and Jiabao Guo. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337, 02 2019.

[129] Adrian Rebmann and Han van der Aa. Extracting semantic process information from the natural language in event logs. In Marcello La Rosa, Shazia Sadiq, and Ernest Teniente, editors, *Advanced Information Systems Engineering*, pages 57–74, Cham, 2021. Springer International Publishing. ISBN 978-3-030-79382-1.

[130] Arka Sadhu, Tanmay Gupta, Mark Yatskar, Ram Nevatia, and Aniruddha Kembhavi. Visual semantic role labeling for video understanding. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 5589–5600. Computer Vision Foundation / IEEE, 2021.

[131] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In Christian Boitet and Pete Whitelock, editors, *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL '98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference*, pages 86–90. Morgan Kaufmann Publishers / ACL, 1998. doi: 10.3115/980845.980860. URL https://aclanthology.org/P98-1013/.

[132] Kun Xu, Han Wu, Linfeng Song, Haisong Zhang, Linqi Song, and Dong Yu. Conversational semantic role labeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2465–2475, 2021. doi: 10.1109/TASLP.2021.3074014.

[133] Llio Humphreys, Guido Boella, Leendert van der Torre, Livio Robaldo, Luigi Di Caro, Sepideh Ghanavati, and Robert Muthuri. Populating legal ontologies using semantic role labeling. *Artif. Intell. Law*, 29(2):171–211, 2021. doi: 10.1007/s10506-020-09271-3. URL https://doi.org/10.1007/s10506-020-09271-3.

[134] Long Chen, Zhihong Jiang, Jun Xiao, and Wei Liu. Human-like controllable image captioning with verb-specific semantic roles. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 16846–16856. Computer Vision Foundation / IEEE, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Chen_

Human-Like_Controllable_Image_Captioning_With_Verb-Specific_Semantic_Roles_CVPR_2021_paper.html.

[135] Quynh Thi Ngoc Do, Steven Bethard, and Marie-Francine Moens. Domain adaptation in semantic role labeling using a neural language model and linguistic resources. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23 (11):1812–1823, 2015. doi: 10.1109/TASLP.2015.2449072.

[136] Stephen Wu, Kirk Roberts, Surabhi Datta, Jingcheng Du, Zongcheng Ji, Yuqi Si, Sarvesh Soni, Qiong Wang, Qiang Wei, Yang Xiang, Bo Zhao, and Hua Xu. Deep learning in clinical natural language processing: a methodical review. *J. Am. Medical Informatics Assoc.*, 27(3):457–470, 2020. doi: 10.1093/jamia/ocz200.

[137] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 06 2011. ISSN 1067-5027. doi: 10.1136/amiajnl-2011-000203.

[138] Steven Bethard, Leon Derczynski, Guergana Savova, James Pustejovsky, and Marc Verhagen. SemEval-2015 task 6: Clinical TempEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 806–814, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-2136.

[139] Yan Wang, Serguei Pakhomov, and Genevieve B. Melton. Predicate argument structure frames for modeling information in operative notes. In *MEDINFO 2013 - Proceedings of the 14th World Congress on Medical and Health Informatics*, number 1-2 in Studies in Health Technology and Informatics, pages 783–787. IOS Press, 2013. ISBN 9781614992882. doi: 10.3233/978-1-61499-289-9-783. 14th World Congress on Medical and Health Informatics, MEDINFO 2013 ; Conference date: 20-08-2013 Through 23-08-2013.

[140] Sarah R. Moeller, Irina Wagner, Martha Palmer, Kathryn Conger, and Skatje Myers. The russian propbank. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 5995–6002. European Language Resources Association, 2020.

[141] Neslihan Kara, Deniz Baran Aslan, Büsra Marsan, Özge Bakay, Koray Ak, and O. T. Yildiz. Tropbank: Turkish propbank v2.0. In *LREC*, 2020.

[142] Azadeh Mirzaei and Amirsaeid Moloodi. Persian proposition bank. In *LREC*, 2016.

[143] Daniel Albright, Arrick Lanfranchi, Anwen Fredriksen, William F. Styler IV, Colin Warner, Jena D. Hwang, Jinho D. Choi, Dmitriy Dligach, Rodney D.

Nielsen, James H. Martin, Wayne H. Ward, Martha Palmer, and Guergana K. Savova. Towards comprehensive syntactic and semantic annotations of the clinical narrative. *J. Am. Medical Informatics Assoc.*, 20(5):922–930, 2013. doi: 10.1136/amiajnl-2012-001317. URL https://doi.org/10.1136/amiajnl-2012-001317.

[144] Wen-Chi Chou, Richard Tzong-Han Tsai, Ying-Shan Su, Wei Ku, Ting-Yi Sung, and Wen-Lian Hsu. A semi-automatic method for annotating a biomedical Proposition Bank. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, pages 5–12, Sydney, Australia, July 2006. Association for Computational Linguistics. URL https://aclanthology.org/W06-0602.

[145] Olga Majewska, Charlotte Collins, Simon Baker, Jari Björne, Susan Windisch Brown, Anna Korhonen, and Martha Palmer. Bioverbnet: a large semantic-syntactic classification of verbs in biomedicine. *J. Biomed. Semant.*, 12(1):12, 2021. doi: 10.1186/s13326-021-00247-z. URL https://doi.org/10.1186/s13326-021-00247-z.

[146] Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester, and Chris Develder. Recipe instruction semantics corpus (risec): Resolving semantic structure and zero anaphora in recipes. In Kam-Fai Wong, Kevin Knight, and Hua Wu, editors, *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 821–826. Association for Computational Linguistics, 2020. URL https://aclanthology.org/2020.aacl-main.82/.

[147] Yinglin Wang. Semantic information extraction for software requirements using semantic role labeling. In *2015 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pages 332–337, 2015. doi: 10.1109/PIC.2015.7489864.

[148] Yifan Peng, Zizhao Zhang, Xiaosong Wang, Lin Yang, and Le Lu. Chapter 5 - text mining and deep learning for disease classification. 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 109–135. Academic Press, 2020. ISBN 978-0-12-816176-0. doi: https://doi.org/10.1016/B978-0-12-816176-0.00010-7. URL https://www.sciencedirect.com/science/article/pii/B9780128161760000107.

[149] Roos Bakker, Romy A.N. van Drie, Maaike de Boer, Robert van Doesburg, and Tom van Engers. Semantic role labelling for dutch law texts. In *Proceedings of the Language Resources and Evaluation Conference*, pages 448–457, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.47.

[150] J. Betina Antony, N. R. Rejin Paul, and G. S. Mahalakshmi. Entity and verb semantic role labelling for tamil biomedicine. In Purushothama B. R., Veena Thenkanidiyoor, Rajendra Prasath, and Odelu Vanga, editors, *Mining Intelligence and Knowledge Exploration*, pages 72–83, Cham, 2020. Springer International Publishing. ISBN 978-3-030-66187-8.

[151] Ishan Jindal, Alexandre Rademaker, MichaÅ, Ulewicz, Ha Linh, Huyen Nguyen, Khoi-Nguyen Tran, Huaiyu Zhu, and Yunyao Li. Universal proposition bank 2.0. In *Proceedings of the Language Resources and Evaluation Conference*, pages 1700–1711, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.181.

[152] Sa-kwang Song, Yun-soo Choi, Heung-seon Oh, Sung-Hyon Myaeng, Sung-Pil Choi, Hong-Woo Chun, Chang-Hoo Jeong, and Won-Kyung Sung. Feasibility study for procedural knowledge extraction in biomedical documents. In Mohamed Vall Mohamed Salem, Khaled Shaalan, Farhad Oroumchian, Azadeh Shakery, and Halim Khelalfa, editors, *Information Retrieval Technology*, pages 519–528, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-25631-8.

[153] Ricardo Campos, Vı'tor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289, 2020. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2019.09.013. URL https://www.sciencedirect.com/science/article/pii/S0020025519308588.

[154] Rossella Varvara. *Verbs as nouns: empirical investigations on event-denoting nominalizations*. PhD thesis, University of Trento, 05 2017.

[155] Steven Bird. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72, Sydney, Australia, July 2006. Association for Computational Linguistics. doi: 10.3115/1225403.1225421. URL https://aclanthology.org/P06-4018.

[156] Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The conll 2008 shared task on joint parsing of syntactic and seman-

tic dependencies. In Alexander Clark and Kristina Toutanova, editors, *Proceedings of the Twelfth Conference on Computational Natural Language Learning, CoNLL 2008, Manchester, UK, August 16-17, 2008*, pages 159–177. ACL, 2008. URL https://aclanthology.org/W08-2121/.

[157] Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA, June 2006. Association for Computational Linguistics. URL https://aclanthology.org/N06-2015.

[158] Peng Shi and Jimmy Lin. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255, 2019.

[159] Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4007. URL https://aclanthology.org/W19-4007.

[160] Ziqi Zhang, Philip Webster, Victoria Uren, Andrea Varga, and Fabio Ciravegna. Automatically extracting procedural knowledge from instructional texts using natural language processing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 520–527, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/244_Paper.pdf.

[161] Kevin Bretonnel Cohen and Dina Demner-Fushman. *Biomedical Natural Language Processing*. John Benjamins, 2014.

[162] Dina Demner-Fushman, Kevin Bretonnel Cohen, Sophia Ananiadou, and Junichi Tsujii, editors. *Proceedings of the 21st Workshop on Biomedical Language Processing*, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[163] Sujay Kulshrestha, Dmitriy Dligach, Cara Joyce, Marshall S. Baker, Richard Gonzalez, Ann P. O'Rourke, Joshua M. Glazer, Anne Stey, Jacqueline M. Kruser, Matthew M. Churpek, and Majid Afshar. Prediction of severe chest injury using natural language processing from the electronic health record. *Injury*, 52(2):205–212, 2021. ISSN 0020-1383. doi: https://doi.org/10.1016/j.injury.2020.10.094.

[164] Elham Sagheb, Taghi Ramazanian, Ahmad P. Tafti, Sunyang Fu, Walter K. Kremers, Daniel J. Berry, David G. Lewallen, Sunghwan Sohn, and Hilal Maradit Kremers. Use of natural language processing algorithms to identify common data elements in operative notes for knee arthroplasty. *The Journal of Arthroplasty*, 36(3):922–926, 2021. ISSN 0883-5403. doi: https://doi.org/10.1016/j.arth.2020.09.029.

[165] Sunyang Fu, Cody C. Wyles, Douglas R. Osmon, Martha L. Carvour, Elham Sagheb, Taghi Ramazanian, Walter K. Kremers, David G. Lewallen, Daniel J. Berry, Sunghwan Sohn, and Hilal Maradit Kremers. Automated detection of periprosthetic joint infections and data elements using natural language processing. *The Journal of Arthroplasty*, 36(2):688–692, 2021. ISSN 0883-5403. doi: https://doi.org/10.1016/j.arth.2020.07.076.

[166] Aditya V. Karhade, Michiel E.R. Bongers, Olivier Q. Groot, Erick R. Kazarian, Thomas D. Cha, Harold A. Fogel, Stuart H. Hershman, Daniel G. Tobert, Andrew J. Schoenfeld, Christopher M. Bono, James D. Kang, Mitchel B. Harris, and Joseph H. Schwab. Natural language processing for automated detection of incidental durotomy. *The Spine Journal*, 20(5):695–700, 2020. ISSN 1529-9430. doi: https://doi.org/10.1016/j.spinee.2019.12.006.

[167] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785.

[168] Eva Hagberg, David Hagerman, Richard Johansson, Nasser Hosseini, Jan Liu, Elin Björnsson, Jennifer Alvén, and Ola Hjelmgren. Semi-supervised learning with natural language processing for right ventricle classification in echocardiography—a scalable approach. *Computers in Biology and Medicine*, 143:105282, 2022. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2022.105282.

[169] Leonardo A.M. Zornoff, Hicham Skali, Marc A. Pfeffer, Martin St. John Sutton, Jean L. Rouleau, Gervasio A. Lamas, Ted Plappert, Jacques R. Rouleau, Lemuel A. Moyé, Sandra J. Lewis, Eugene Braunwald, and Scott D. Solomon. Right ventricular dysfunction and risk of heart failure and mortality after myocardial infarction. *Journal of the American College of Cardiology*, 39(9):1450–1455, 2002. ISSN 0735-1097. doi: https://doi.org/10.1016/S0735-1097(02)01804-1.

[170] Alireza Borjali, Martin Magnéli, David Shin, Henrik Malchau, Orhun K. Muratoglu, and Kartik M. Varadarajan. Natural language processing with deep learn-

ing for medical adverse event detection from free-text medical narratives: A case study of detecting total hip replacement dislocation. *Computers in Biology and Medicine*, 129:104140, 2021. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2020.104140.

[171] Javad Parvizi, Elizabeth Picinic, and Peter F Sharkey. Revision total hip arthroplasty for instability: surgical techniques and principles. *Instructional course lectures*, 58:183 – 191, 2009.

[172] Pilar López-Úbeda, Manuel Carlos Díaz-Galiano, Teodoro Martín-Noguerol, Antonio Luna, L. Alfonso Ureña-López, and M. Teresa Martín-Valdivia. Covid-19 detection in radiological text reports integrating entity recognition. *Computers in Biology and Medicine*, 127, 2020. ISSN 00104825. doi: 10.1016/j.compbiomed.2020.104066. Cited by: 21; All Open Access, Bronze Open Access, Green Open Access.

[173] Kent A. Spackman, Keith E. Campbell, and Roger A. Côté. SNOMED RT: a reference terminology for health care. In *AMIA 1997, American Medical Informatics Association Annual Symposium, Nashville, TN, USA, October 25-29, 1997*, pages 640–644. AMIA, 1997.

[174] Thorsten Barnickel, Jason Weston, Ronan Collobert, Hans-Werner Mewes, and Volker Stümpflen. Large scale application of neural network based semantic role labeling for automated relation extraction from biomedical texts. *PLOS ONE*, 4 (7):1–6, 07 2009.

[175] Steven Bethard, Zhiyong Lu, James H Martin, and Lawrence Hunter. Semantic role labeling for protein transport predicates. *BMC bioinformatics*, 9(1):1–15, 2008.

[176] Fabian Eckert and Mariana Neves. Semantic role labeling tools for biomedical question answering: a study of selected tools on the BioASQ datasets. In *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, pages 11–21, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5302.

[177] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740.

[178] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S. Weld. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, 2020.

[179] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[180] Nils Reimers and Iryna Gurevych. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1035.

[181] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, pages 1877–1901, 2020.

[182] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In Ido Dagan and Daniel Gildea, editors, *Proceedings of the Ninth Conference on Computational Natural Language Learning, CoNLL 2005, Ann Arbor, Michigan, USA, June 29-30, 2005*, pages 152–164. ACL, 2005.

[183] Dagobert Soergel. *WordNet. An Electronic Lexical Database*. MIT Press, 10 1998.

[184] Bernard Gibaud, Germain Forestier, Carolin Feldmann, Giancarlo Ferrigno, Paulo Gonçalves, Tamás Haidegger, Chantal Julliard, Darko Katić, Hannes Kenngott, Lena Maier-Hein, et al. Toward a standard ontology of surgical process models. *International journal of computer assisted radiology and surgery*, 13(9):1397–1408, 2018.

[185] Nathaniel J Soper and Gerald M Fried. The fundamentals of laparoscopic surgery: its time has come. *Bull Am Coll Surg*, 93(9):30–32, 2008.

[186] Tamás D Nagy and Tamás P Haidegger. Towards standard approaches for the evaluation of autonomous surgical subtask execution. In *2021 IEEE 25th International Conference on Intelligent Engineering Systems (INES)*, pages 000067–000074. IEEE, 2021.

[187] E. Tagliabue, D. Meli, D. Dall'alba, and P. Fiorini. Deliberation in autonomous robotic surgery: a framework for handling anatomical uncertainty. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 11080–11086, 2022.

[188] Pedro Cabalar, Roland Kaminski, Philip Morkisch, and Torsten Schaub. telingo= asp+ time. In *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 256–269. Springer, 2019.

[189] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. Pddl-the planning domain definition language, 1998.

[190] David Pérez-Rey, Victor Maojo, Miguel García-Remesal, Raúl Alonso-Calvo, Holger Billhardt, Fernando Martín-Sánchez, and A Sousa. Ontofusion: Ontology-based integration of genomic and clinical databases. *Computers in biology and medicine*, 36(7-8):712–730, 2006.

[191] Séverin Lemaignan, Raquel Ros, Lorenz Mösenlechner, Rachid Alami, and Michael Beetz. Oro, a knowledge management platform for cognitive architectures in robotics. In *2010 IEEE/RSJ International conference on intelligent robots and systems*, pages 3548–3553. IEEE, 2010.

[192] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Philipp Wanko. Theory solving made easy with clingo 5. In *Technical Communications of the 32nd International Conference on Logic Programming (ICLP 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[193] Michele Ginesi, Daniele Meli, Andrea Roberti, Nicola Sansonetto, and Paolo Fiorini. Autonomous task planning and situation awareness in robotic surgery. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3144–3150. IEEE, 2020.

[194] Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103, 2015.

[195] Niket Tandon, Aparna S Varde, and Gerard de Melo. Commonsense knowledge in machine intelligence. *ACM SIGMOD Record*, 46(4):49–52, 2018.

[196] Robert DiPietro, Narges Ahmidi, Anand Malpani, Madeleine Waldram, Gyusung Lee, Mija Lee, Sunil Vedula, and Gregory Hager. Segmenting and classifying activities in robot-assisted surgery with recurrent neural networks. *International Journal of Computer Assisted Radiology and Surgery*, 14:1–16, 04 2019. doi: 10.1007/s11548-019-01953-x.

[197] Jian Chen, Nathan Cheng, Giovanni Cacciamani, Paul Oh, Michael Lin-Brande, Daphne Remulla, Inderbir Gill, and Andrew Hung. Objective assessment of robotic surgical technical skill: A systematic review. *The Journal of Urology*, 201, 07 2018. doi: 10.1016/j.juro.2018.06.078.

[198] Intuitive Surgical Inc. `"http://www.intuitive-foundation.org/dvrk/"`, 2023. Accessed: 2023-02-15.

[199] Yangming Li, Blake Hannaford, and Jacob Rosen. The raven open surgical robotic platforms: A review and prospect. *Acta Polytechnica Hungarica*, 16(8):9 – 27, 2019. doi: 10.12700/APH.16.8.2019.8.2.

[200] Morris DeGroot and Mark Schervish. *Probability and Statistics*. Pearson, 01 2011.

[201] Daxing Qian, Ximing Pei, and Xiangkun Li. The application of equivalent mean square error method in scalable video perceptual quality. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 227 LNICST:3 – 7, 2018. doi: 10.1007/978-3-319-73447-7_1.

[202] Etienne Kerre and Mike Nachtegael. Fuzzy techniques in image processing. *Physica-Verlag*, 52, 01 2000.

[203] Birmohan Singh, Dalwinder Singh, Gurwinder Singh, Neeraj Sharma, and Vicky Kumar. Motion detection for video surveillance. *2014 International Conference on Signal Propagation and Computer Technology, ICSPCT 2014*, pages 578–584, 07 2014. doi: 10.1109/ICSPCT.2014.6884919.

[204] K.S. Arun, T.S. Huang, and Steven Blostein. Least-squares fitting of two 3-d point sets. ieee t pattern anal. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-9:698 – 700, 10 1987. doi: 10.1109/TPAMI.1987.4767965.

[205] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513 – 523, 1988.

[206] Francesca Fallucchi, Maria Teresa Pazienza, and Fabio Massimo Zanzotto. Generic ontology learners on application domains. *Proceedings of the 7th International Conference on Language Resources and Evaluation, LREC 2010*, page 378 – 385, 2010.

[207] Gilles Louppe. *Understanding Random Forests: From Theory to Practice*. PhD thesis, University of Liège, 10 2014.

[208] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, Vincent Dubourg, I. Vanderplas, A. Passos, D. Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-iearn: Machine learning in python. *Journal of Machine Learning Research*, 12, 01 2011.

[209] Manuel Fernandez-Delgado, E. Cernadas, S. Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15:3133–3181, 10 2014.

[210] Gilles Louppe, Louis Wehenkel, Antonio Sutera, and Pierre Geurts. Understanding variable importances in forests of randomized trees. *Advances in Neural Information Processing Systems*, 2013.

[211] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language models of code are few-shot commonsense learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1384–1403, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.90.

[212] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[213] Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire

Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR, 17–23 Jul 2022.

[214] Aaron Daniel Cohen, Adam Roberts, Alejandra Molina, Alena Butryna, Alicia Jin, Apoorv Kulshreshtha, Ben Hutchinson, Ben Zevenbergen, Blaise Hilary Aguera-Arcas, Chung ching Chang, Claire Cui, Cosmo Du, Daniel De Freitas Adiwardana, Dehao Chen, Dmitry (Dima) Lepikhin, Ed H. Chi, Erin Hoffman-John, Heng-Tze Cheng, Hongrae Lee, Igor Krivokon, James Qin, Jamie Hall, Joe Fenton, Johnny Soraker, Kathy Meier-Hellstern, Kristen Olson, Lora Mois Aroyo, Maarten Paul Bosma, Marc Joseph Pickett, Marcelo Amorim Menegali, Marian Croak, Mark Díaz, Matthew Lamm, Maxim Krikun, Meredith Ringel Morris, Noam Shazeer, Quoc V. Le, Rachel Bernstein, Ravi Rajakumar, Ray Kurzweil, Romal Thoppilan, Steven Zheng, Taylor Bos, Toju Duke, Tulsee Doshi, Vincent Y. Zhao, Vinodkumar Prabhakaran, Will Rusch, YaGuang Li, Yanping Huang, Yanqi Zhou, Yuanzhong Xu, and Zhifeng Chen. Lamda: Language models for dialog applications. *ArXiv*, 2022.

[215] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John F. J. Mellor, Irina Higgins, Antonia Creswell, Nathan McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, L. Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, N. K. Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Tobias Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew G. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem W. Ayoub, Jeff Stanway, L. L. Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scal-

ing language models: Methods, analysis & insights from training gopher. *ArXiv*, abs/2112.11446, 2021.

[216] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zheng, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. Using deepspeed and megatron to train megatron-turing NLG 530b, A large-scale generative language model. *CoRR*, abs/2201.11990, 2022.

[217] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

[218] Hugo Liu and Push Singh. Commonsense reasoning in and over natural language. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 293–306. Springer, 2004.

[219] Marios Daoutis, Silvia Coradeshi, and Amy Loutfi. Grounding commonsense knowledge in intelligent systems. *Journal of Ambient Intelligence and Smart Environments*, 1(4):311–321, 2009.