December 2024

Surgical Gesture Recognition in Robot-Assisted Surgery using Machine Learning Methods on Kinematic Data

Alexandros DIMITRIADIS [a,1], George MOUSTRIS [b] and Costas TZAFESTAS [a,b]
[a] *National Technical University of Athens (NTUA), School of Electrical and Computer Engineering*
[b] *Athena Research Center, Institute of Robotics*
ORCiD ID: George Moustris https://orcid.org/0000-0002-0454-0755, Costas Tzafestas https://orcid.org/0000-0003-1545-9191

Abstract. This work focuses on training machine learning models to recognize gestures during robot-assisted surgical procedures in real-time, using exclusively kinematic data from the patient-side manipulators. The JIGSAWS dataset, specifically the suturing tasks, serves as the evaluation benchmark. We experimented with various neural network architectures, using an LSTM architecture as the baseline approach. To further enhance performance, two hybrid approaches are proposed in this work: the first one combining an LSTM with a Conditional Random Field (CRF) and the second one integrating an attention layer. An extensive experimental study was conducted to evaluate and optimize the performance of the different approaches, and identify areas for improvement. A thorough comparative analysis of the results shows that the proposed hybrid approaches, in particular the one combining an attention layer, can improve recognition rate, as compared to relevant state-of-the-art, with accuracy of 81.56%. This study lays the foundation for further research in the field, focusing on advancing real-time surgical gesture recognition as a means to develop tools that can provide intraoperative monitoring and assistance to the surgeon.

Keywords. Surgical Gesture Recognition, Robotic Surgery, JIGSAWS, LSTM, Kinematic Data, Real-time, Machine Learning, Hybrid Model, Attention Mechanism, Self Attention, CRF

1. Introduction

The remarkable progress in the development of artificial intelligence and machine learning techniques has fostered a deeper integration of technology across diverse sectors, driving significant advancements. The recent surge in data science and deep learning has not gone unnoticed by the healthcare industry. Medical experts increasingly recognize the potential for a wider incorporation of robotics and machine learning in healthcare [1], which could considerably improve patient care and streamline surgical workflows.

[1] Corresponding Author: Alexandros Dimitriadis, alathdim@gmail.com

Clinical evidence has indicated that increased precision and accuracy of surgical robots can enhance the operative outcomes of surgery, showing improvements like reduced blood loss and shortest operation duration [2]. Moreover, surgical robots enable teleoperation, allowing surgeons to perform procedures remotely, expanding access to specialized care. The usage of surgical robots can provide a variety of valuable data, which can be utilized by deep learning models and contribute to our understanding of surgical tasks, assess the execution, provide assistance during the procedure, and generally advance towards the automation of surgery. Kinematic and visual data from surgical robots have been harvested and organized in datasets, such as JIGSAWS [3], which contains recordings of surgical tasks performed on benchtop models using the da Vinci Surgical System, MISAW [4], which includes sequences of micro-surgical anastomosis on artificial blood vessels using a master-slave robotic platform [5] and RARP-45 [6], which leverages data from in-vivo Radical Prostatectomies, also from the da Vinci system. The development of such datasets facilitates systematic training, evaluation, and benchmarking of diverse methodologies, thereby advancing the field of surgical robotics. An integral part of this process is the segmentation of surgical tasks in atomic gestures and the ability to recognize them.

Automated surgical gesture recognition seeks to identify and categorize meaningful, fundamental, atomic action units within surgical tasks that comprise a surgical intervention [7], and it is a crucial component of analysis of surgical tasks, technical skill assessment, intraoperative assistance and, ultimately, robotic automation.

Surgical activity is a complex interplay of dexterous human actions and as such, is susceptible to variability arising from factors like different surgical styles, patient anatomy or the occurrence of unexpected intra-operative events. This inherent variability in surgical procedures, coupled with the intricacy of the end-effector trajectories further compound the challenge of analyzing and segmenting surgical motion patterns [6]. The integration of multiple data sources, including kinematic data and video recordings, can provide valuable contextual information, such as instrument interactions, to enhance the understanding and analysis of surgical procedures. Recent advances in deep learning, particularly in the domain of computer vision and sequential data processing, have enabled the development of sophisticated models capable of processing and interpreting these multimodal data streams. One promising approach to address the challenges of surgical gesture recognition involves the development of hybrid models that combine the strengths of traditional machine learning techniques with the power of deep learning.

Although the clinical adoption of computer-assisted or autonomous robotic surgery is still not widespread, a significant body of research has been published on gesture recognition, presenting substantial results. In this work, we present a comprehensive study on the application of machine learning techniques for surgical gesture recognition. Theoretical foundations are laid out, discussing relevant machine learning concepts focusing on techniques suitable for sequential data. Various approaches are presented and evaluated, aiming to delve deeper into the challenges and limitations of current models and their underlying weaknesses.

2. Theoretical Foundations of Machine Learning in Gesture Recognition

2.1. Deep Learning

Deep learning is a subfield of machine learning that leverages the power of *artificial neural networks*. These architectures typically feature a hierarchical arrangement of stacked layers. Each layer processes the data through nonlinear transformations to generate an abstract representation. The lower layers capture simpler, more fundamental, information which the higher layers then use to construct increasingly complex representations. These networks generally include an input layer, an output layer, and multiple hidden layers in between.

An important method of performing deep learning is the *Supervised Learning* paradigm. In supervised machine learning, models are trained on *labeled* datasets, where the input's set of measurable properties called *features*, is paired with their corresponding target outputs. Given these input features, the model generates a probability distribution over the output space. The model's parameters are learned through an iterative optimization process, typically using gradient-based methods, to minimize a specified *loss function* that expresses the deviation between the predicted and actual outputs [8].

Optimization is a crucial part of deep learning, involving the process of searching for an objective function's *optimal value* [9]. Since the goal is to minimize the discrepancy between model predictions and the ground truth labels, the training process can be expressed as the optimization task of finding the *minimum* of the loss function. *Gradient descent* (GD) iteratively adjusts parameters in the direction of the negative gradient, pointing towards the steepest descent.

2.2. Backpropagation

Each neuron receives input signals from other neurons, which are then weighted and summed, along with a bias term. This weighted sum, known as the pre-activation, determines the neuron's output:

$$z_i(x) = b_i + \sum_{j=1}^{n} W_{ij} x_j \tag{1}$$

where:

$z_i(x)$ = preactivation
b_i = bias term
W_{ij} = weight connecting neuron j to neuron i
x_j = input

The weights and biases of the neurons are the *trainable properties* of the model, and they are iteratively adjusted in order to improve the model's overall performance [10]. Usually the weights and biases are initialized randomly and subsequently adjusted using *backpropagation*, which optimizes model parameters by calculating the gradient of the loss function with respect to these parameters.

During the forward pass, input data is conveyed through multiple layers, with each neuron applying its weights and biases and passing the result through an activation func-

tion. The loss function quantifies the difference between the final output and the ground truth. Backpropagation algorithm, utilizing the chain rule, transmits the error backwards through the network, layer by layer, and after calculating the gradient of the loss function, the weights and biases are appropriately adjusted in order to minimize the loss.

2.3. Long Short Term Memory

Recurrent Neural Networks (RNNs) are distinguished by their architecture, which contains *feedback loops*, enabling them to process sequential data effectively. These cyclical connections give the network the ability to incorporate past information into the calculations of the current outputs, creating a form of memory. This memory mechanism empowers RNNs to learn and utilize temporal dependencies within the data.

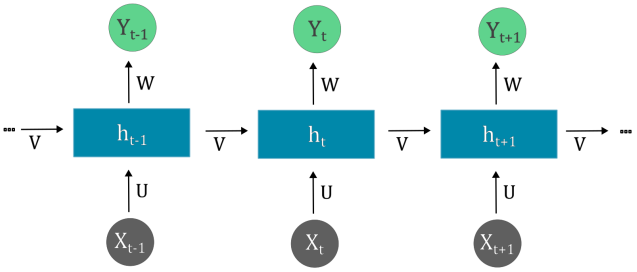

Figure 1. Basic RNN architecture unfolded

By unfolding the RNN as presented in Figure 1, we can visualize its sequential nature. The chain-like structure allows the network to capture temporal dependencies within data sequences, making it well-suited for tasks like gesture recognition, where the evolution of the feature values through time is crucial for accurate interpretation.

RNNs, due to their unique architecture, are susceptible to the vanishing and exploding gradient problems. These issues arise from the repeated application of derivatives during backpropagation, leading to either diminishing or amplifying gradients. This can hinder the network's ability to learn long-term dependencies, especially in deep architectures.

Long Short Term Memory (LSTM) is an RNN variant, that was designed to address the vanishing gradient problem [11]. It achieves this by employing a cell state and three gates: the input, forget, and output gates. These gates collaboratively determine which information to retain or discard from the cell state and which to output.

As illustrated in Figure 2, each LSTM cell receives data from three sources at each time step:

- **Input Vector ($\mathbf{x_t}$):** The current input data
- **Previous hidden state ($\mathbf{h_{t-1}}$):** The output from the previous time step
- **Previous cell state ($\mathbf{c_{t-1}}$):** The long term memory

These inputs are processed by the three primary gates:

1. **Forget Gate:**
   - Determines which information from the previous cell state should be discarded

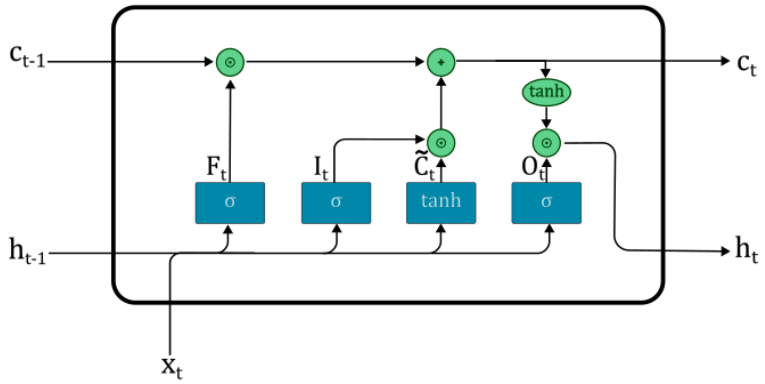

Figure 2. LSTM memory cell architecture

- Calculate a forget gate value (F_t) using a sigmoid activation function:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2}$$

2. **Input Gate:**
   - Responsible for dictating which parts of the new input information should be stored in the cell state.
   - Calculates an input gate value (I_t) using a sigmoid activation function:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{3}$$

   - The potential new value for the cell state, which is evaluated by the input gate, derives from the candidate memory, which applies a *tanh* activation function to a combination of the current input and the previous hidden state.

$$\tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{4}$$

3. **Cell State Update:**
   - Updates the cell state (C_t) based on the forget gate, input gate, and candidate memory:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{c}_t \tag{5}$$

   where $\odot$ represents element-wise multiplication.

4. **Output Gate:**
   - Determines which parts of the cell state should be output as the current hidden state.
   - Calculates an output gate value (O_t) using a sigmoid activation function:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{6}$$

- Using the gate's result, the new hidden state (h_t) is calculated:

$$h_t = o_t \odot \tanh(C_t) \tag{7}$$

The gate mechanism in LSTM enables the network to selectively retain relevant information and forget less useful parts, allowing it to learn complex patterns over large datasets of sequential data. Additionally, by mitigating the gradient vanishing problem, the LSTM network is able to store information over extended periods and identify longer temporal dependencies, which is important in the gesture classification task where the accuracy of the prediction depends on a sequence of movements over time.

2.4. Conditional Random Fields

Probabilistic Graphical Models (PGMs) use graph structures in order to represent complex probabilistic relationships between variables. The nodes of the graph correspond to random variables and the edges represent the probabilistic dependencies between the variables they connect [12].

Markov Random Fields (MRFs), a subset of PGMs, are *undirected* graphical models that embody the Markov property. This property stipulates that a node's conditional probability distribution is solely dependent upon its immediate neighbors in the graph. In other words, a node's future state is independent of its past states, given its current state and the states of its neighboring nodes. This characteristic enables MRFs to efficiently model complex dependencies between random variables

Conditional Random Fields (CRFs) are a class of statistical modeling techniques that have gained significant traction in machine learning, particularly for tasks involving sequential data processing. Built upon the foundation of MRFs, CRFs offer a robust framework for modeling the conditional probability of a sequence of labels given a sequence of observations. *Discriminative models* are a broad category of statistical models focused on modeling the conditional probability of the output given the input, $P(y|x)$, in contrast to *generative models* whose aim is to model the joint probability distribution of the input and output variables, $P(x,y)$. They are primarily concerned with predicting the correct output category for a given input, rather than modeling the underlying probability distribution of the data. Thus, discriminative models are well-suited for classification tasks.

In the classification context, X represents a sequence of data points while Y represents the corresponding sequence of labels. Although X and Y are jointly distributed, a discriminative modeling approach focuses on the conditional probability distribution $P(y|x)$. This model directly estimates the probability of a specific label sequence Y given a particular data sequence X. Since CRFs are a variant or MRFs, they obey the Markov property, which implies that a label is conditionally independent of other labels, given its neighboring labels and the global input. Formally, this can be expressed as follows:

Let $Y = \{Y_v : v \in V\}$ be a set of random variables, where Y_v is associated with vertex v in the graph $G = (V,E)$. (X,Y) is a CRF if, when conditioned on X, the random variables Y satisfy the pairwise Markov property with respect to the graph G:

$$P(Y_v|X,\{Y_w : w \neq v\}) = P(Y_v|X,\{Y_w : w \sim v\})$$

where $w \sim v$ denotes that vertices w and v are adjacent (i.e., there is an edge between them in G). [13]

The conditional probability of the output sequence given the input sequence is described by the following equation:

$$P(y|x) = \frac{\exp\left(\sum_i w_i f_i(y,x)\right)}{Z(x)}$$

where:

x, y = input and output sequences
$f_i(y,x)$ = feature functions that compute the contribution of each feature to the overall score of the output sequence
w_i = weights associated with feature functions
$Z(X)$ = normalization factor

1

2.5. Attention Mechanisms

Attention mechanisms are a machine learning technique, that empowers neural networks with the ability to selectively focus on relevant parts of the input data. Central to this approach is the identification and weighting of input features. By assigning higher weights to relevant features, we are effectively guiding the model to concentrate on the most informative aspects of the input. The versatility of attention mechanisms has led to their widespread adoption in various domains, including natural language processing, image recognition, and machine translation [14].

Although the attention mechanism has a similar purpose to the LSTM gate mechanism, which is to isolate relevant information of the input sequence, they differ in their approach of selective focus and memory management. LSTMs encode in fixed memory cells all parts of past information that was chosen to be retained, while attention mechanisms dynamically assign weights to different parts of the input, based on the current output and the entire sequence, ensuring that the model can adapt its focus to the specific task. Additionally, while LSTM process data sequentially, attention mechanisms process the entire sequence simultaneously, enabling parallel computation. This approach speeds up the learning process and enhances its efficiency, especially when working with extensive sequential data.

Self attention is a fundamental attention mechanism that attempts to discover and model relationships between different parts of a single input sequence. It uses a unique way of encoding information by using three vectors: **Query**($\mathbf{Q}$), which describes what kind of information this element is seeking to acquire from other elements of the sequence, **Key**($\mathbf{K}$), which represents the relevance of this element to a given query and **Value**($\mathbf{V}$), which contains the actual data that the element holds.

Multi-head attention extends the concept of self attention by applying the attention mechanism concurrently in multiple instances. Instead of learning a single set of transformations, multi-head attention has the ability to learn multiple, independent groups of transformations called heads. The idea behind this implementation is that each head can

focus on specific patterns and capture different kinds of relationships between data items. These separate representations are combined in order to form the final output.

Let $\mathbf{X}$ be the input embedding matrix. Each head independently learns its own linear transformations to project input embeddings into different *query*, *key*, and *value* subspaces (Eq. 8a, 8b, 8c). This allows each head to specialize in different aspects of the input data.

$$\mathbf{Q_i} = \mathbf{W_Q^i} \cdot \mathbf{X} \tag{8a}$$

$$\mathbf{K_i} = \mathbf{W_K^i} \cdot \mathbf{X} \tag{8b}$$

$$\mathbf{V_i} = \mathbf{W_V^i} \cdot \mathbf{X} \tag{8c}$$

where $\mathbf{W_Q^i}, \mathbf{W_K^i}, \mathbf{W_V^i}$ are projection matrices for the i_{th} head. For each head, the attention mechanism is applied as described in the standard self-attention, viz.

$$\text{attention}(\mathbf{Q_i}, \mathbf{K_i}, \mathbf{V_i}) = \text{softmax}\left(\frac{\mathbf{Q_i}\mathbf{K_i}^T}{\sqrt{d_k}}\right) V_i \tag{9}$$

The attention outputs from all heads are concatenated and are then projected back to the original dimension. In matrix form, the entire multi-head attention operation is represented as:

$$\text{multi\_head}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{W}_O \cdot \text{concat}(\text{head}_1(\mathbf{Q}, \mathbf{K}, \mathbf{V}), \ldots, \text{head}_h(\mathbf{Q}, \mathbf{K}, \mathbf{V})) \tag{10}$$

where $\mathbf{W}_O$ is the final projection matrix and $\text{head}_i(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ is the output of the i^{th} attention head.

3. Related Work

3.1. Probabilistic Model approaches

The literature contains numerous works that present a range of approaches to gesture classification. Earlier work focused on kinematic data and tried to tackle the problem using probabilistic models. One of the first approaches for automatic gesture recognition was presented in 2006 [15]. Kinematic data were collected from the da Vinci Surgical System, including joint positions and velocities from the master-side and patient-side manipulators (MSM and PSM). Following the observation that surgical motions change gradually, feature vectors from neighboring time steps were concatenated to form a new "super" feature vector in order to improve classification performance. Data were normalized to ensure consistency cross different measurements, and their dimensionality was reduced using Linear Discriminant Analysis. A Bayes classifier then determined the most probable gesture at each time.

A subsequent study [16] utilized kinematic data from a bench-top surgical training task to develop an automatic skill assessment model for minimally invasive surgery. A

Hidden Markov Model (HMM) for automatic gesture recognition was trained with kinematic data from both the PSM and MSM of the da Vinci robotic system, captured during suturing tasks performed by surgeons with varying levels of expertise. By grouping consecutive frames into blocks and applying Linear Discriminant Analysis (LDA) high-dimensional kinematic data were projected onto a lower-dimensional space. Each gesture was represented by a separate HMM, which modeled the temporal sequence of kinematic features. The parameters of these HMMs were estimated using the Baum-Welch algorithm. During recognition, the Viterbi algorithm found the most likely sequence of gestures given the input sequence.

Tao et al. [17] gathered data from a similar experimental setup for skill evaluation and proposed a sparse Hidden Markov Model (S-HMM) approach, which used a sparse linear combination of elements from a dictionary of motion words to model the observations. The authors concluded that Gaussian distributions were insufficient for high-dimensional data due to the large number of parameters that were needed to be learned. To address this, they introduced an HMM variation that employed multiple subspaces to model the observations from each gesture, enforcing sparsity constraints on the latent variables. This approach allowed for a more robust parameter learning process and a more accurate representation of the data.

In 2013, one of the few surgical gesture recognition models that could also utilize visual data (alongside [18] and [19]) at the time was proposed [20]. It included a combined Markov/semi-Markov conditional random field (MsM-CRF). This model integrated Conditional Random Fields (CRFs) and Semi-Conditional Random Fields (Semi-CRFs) for gesture recognition. CRFs modeled frame-level dependencies, assigning label probabilities to individual frames. Semi-CRFs assigned labels to segments, which were sequences of frames where a single gesture was executed. Both models were graphical and represented their fundamental time units (frames and segments, respectively) as nodes, connected by edges to model dependencies. For the kinematic data, each model's output was generated using a Support Vector Machine (SVM) classifier, which classified frames and segments. In the case of video data, spatio-temporal features were extracted and organized into a visual dictionary through K-means clustering. Frame and segment representations were then created using histograms of these visual words, which were subsequently used to train SVMs.

3.2. Recurrent Neural Networks

A key limitation of many probabilistic model approaches is that they are restricted to modeling frame-to-frame or segment-to-segment transitions [7]. In contrast, recent efforts have sought to overcome this constraint by utilizing deep learning techniques that have the inherent ability to capture complex long-term dependencies in sequential data. Recurrent Neural Networks are known for their ability to learn temporal characteristics, facilitating their widespread use. In most cases, the preferred RNN variant is the Long Short Term Memory network.

DiPietro et al. [21] trained RNNs for joint segmentation and classification of surgical activities using kinematic data from suturing trials of the JISGAWS [3] and MISTIC-SL datasets. Previous research primarily concentrated only on recognizing discrete gestures. In contrast, this study focused on recognizing maneuvers, which represent higher-level activity segments such as knot-tying. The performance of a forward LSTM and a bidi-

rectional LSTM model were compared, with the bidirectional LSTM showing significant improvement in accuracy and edit distance. However, it is crucial to acknowledge that the bidirectional nature of this model, necessitates access to the entire sequence before generating a prediction, which limits their suitability for real-time applications. In an extension of this work [22] the usage of more RNN variants is explored. Following analysis and optimization of the hyperparameters of each model, the performance was further improved. The proposed architectures included gated recurrent units (GRUs), which are a simpler alternative to LSTM and mixed history RNNs (MIST RNNs), a type of RNNs, that incorporates an attention mechanism that selectively combines hidden states from the distant past to maintain a long-term memory, instead of gates. Each of the examined RNN was enhanced by techniques such as *regularization*, usage of *multiple layers*, and *bidirectional processing*. Simple RNNs were outperformed by the more sophisticated variants, probably due to their inherent vulnerability to the vanishing gradient problem. While MIST RNNs mitigate vanishing gradients, their lack of an inherent bias towards smooth predictions, unlike LSTMs and GRUs, leads to competitive error rates but inferior edit distances. It is apparent from the results of these studies that RNNs can significantly outperform non-RNN methods in both maneuver and gesture recognition. Similarly. The authors in [23] compared the performance of a traditional RNN with an LSTM trained on data acquired by an instrumented obstetrical forceps coupled with the BirthSIM simulator. Although their examined task involved a much smaller number of possible gestures than a suturing task (namely *four*), the LSTM superiority once again highlighted the importance of capturing long term-relationships in surgical gesture recognition.

3.3. Temporal Convolution Networks

Several studies have focused on processing only video data, e.g. [24], where a unified approach to action segmentation using Temporal Convolutional Networks (TCN) was presented. The authors explored the usage of diverse techniques like adding skip connections between layers, and using different patterns of convolutional layers, and evaluated their methods in different datasets containing both surgical and non-surgical tasks. TCNs leverage dilated convolutions to efficiently capture both short-term and long-term dependencies in time series data. Their hierarchical structure with multiple layers of convolutions also enables them to learn complex temporal patterns from video data. The authors observed that certain layers of convolutional filters within the TCN appear to have learned to detect temporal shifts in the input data, demonstrating that TCNs, despite their convolutional nature, can effectively capture temporal dependencies in a manner similar to more traditional temporal models like RNNs or CRFs and, additionally, avoid over-segmentation.

The idea of processing surgical video data through a TCN is extended in [25] with TeCNO, a multi-stage TCN. In order to maintain temporal resolution and reduce the number of parameters, it exclusively uses temporal convolutional layers instead of pooling and fully connected ones. The raw video data were transformed into features in a frame-by-frame fashion using the ResNet50 [26]. The proposed architecture consisted of a 1×1 convolutional layer to match the input feature dimension, followed by a stack of dilated residual layers with increasing dilation factor that widens the receptive field of the model allowing it to capture long-range dependencies.

3.4. Multi-modal architectures

Recently, some multi-modal architectures have been presented that utilize a combination of vision and kinematics, yielding very promising results. The availability of datasets like [3] and [27], with synchronized kinematic and video data, captured at the same rate, proved crucial for many of these studies.

The unified model Fusion-KV [28] demonstrates robustness in complex and realistic surgical scenarios, including dry-lab, cadaveric, and in-vivo experiments. The proposed model consists of four single-source state estimation models based on vision, kinematics and system events. These are combined using a fusion model to make comprehensive inferences. The *vision-based* state estimation model uses a hybrid CNN-TCN approach. The *kinematic* based estimator is also a hybrid model, combining LSTM and TCN models designed to capture both short-term and long-term data feature evolution. Lastly, the *event-based* method evaluates multiple classification algorithms, including Adaboost, decision trees, Random Forests (RF), SVM and others, with the top-scoring models averaged for the final state estimation. Since different states are more easily recognized by certain types of data, to leverage the strengths of all models a weighted voting method was used to combine the prediction vectors from all three estimators.

Keshara Weerasinghe et al. [29] introduced a novel transformer-based framework for real-time recognition and prediction of surgical gestures and trajectories, based on an adaptation of the original transformer model presented in [14] for Natural Language Processing. The model integrates multi-modal information, encompassing kinematic data and video streams. The proposed pipeline comprises three distinct stages; feature extraction and transformation, gesture recognition, and gesture/trajectory prediction, receiving input data through an observation window with one second width. Gesture recognition is achieved through a dedicated transformer encoder, while a separate transformer model is employed for simultaneous gesture and trajectory prediction. During the initial stage, a range of feature subsets derived from the PSM captures in the JIGSAWS dataset were evaluated for kinematic data input. For video data, widely used feature extractors, including ResNet50, were explored, targeting not only phase recognition but also the extraction of supplementary information, such as tool segmentation. The authors utilized the information inferred from the video data by incorporating a state vector that represents surgical context. This vector encodes interactions between surgical instruments and objects within the surgical scene. This contextual information can help disambiguate gestures which might appear similar.

The gesture recognition stage employs a multi-headed attention mechanism to generate a feature vector, which is subsequently processed by a fully connected layer to produce the gesture output vector. For predicting future gestures and trajectories, based on recognized gestures and other input features, a multi-layer transformer decoder is utilized. The decoder's outputs are further refined through linear transformations to produce precise trajectory coordinates..

Despite the numerous approaches and significant advancements in gesture recognition, there remains considerable room for improvement. Probabilistic models often struggle with capturing complex temporal dependencies, and bidirectional variations are unsuitable for real-time applications. Conversely, LSTM networks have proven to be robust solutions. Our research focuses on examining the limitations of LSTM-based architectures that leverage kinematic data and seeks to enhance their capabilities by incorporat-

ing additional techniques. These innovations aim to provide more accurate and efficient real-time surgical gesture recognition.

4. Data Collection and Preprocessing

4.1. Dataset

For the training and evaluation of our methods, we have utilized the JIGSAWS dataset. The JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) is a prominent benchmark dataset in the field of robotic surgery, extensively utilized for research in robotics, computer vision and machine learning. This dataset is pivotal for analyzing surgical activities, treating them as dexterous human motions, with the ultimate goal of enhancing the effectiveness and safety of surgical procedures. JIGSAWS is primarily focused on two critical areas of study: **surgical activity recognition** and **skill assessment**. This dataset was selected due to its widespread adoption, enabling direct comparison with existing research and its rich feature set.

The dataset consists of annotated synchronized captures of kinematic and video data from three basic surgical tasks performed using the Da Vinci Robotic Surgical System. Each task is repeated 5 times by 8 surgeons with varying level of experience in the usage of robotic surgical systems. The operators are index by letters “B”, “C”, “D”, “E”, “F”, “G”, “H”. For gesture recognition, the suturing task is most commonly used, due to its diverse range of gestures and its relevance to real life scenarios. The suturing activity is simulated by sequentially inserting and withdrawing a needle through marked entry and exit points on a soft material using two tooltips to manipulate the needle as shown in Figure 3.

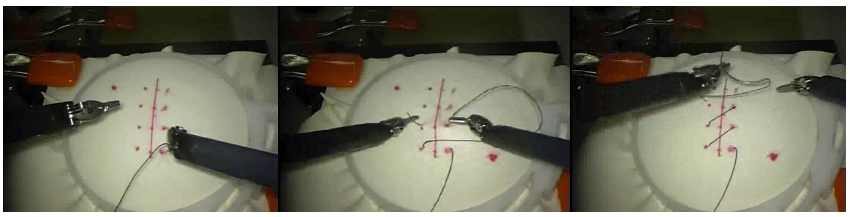

Figure 3. Suturing task footage from the JIGSAWS dataset

The set of kinematic data includes various variables from both the Master Tool Manipulators and Patient Side Manipulators captured at 30 Hz, though only the latter are pertinent to our task. Each manipulator’s kinematic data comprises 19 variables, including Cartesian positions, linear and angular velocities, the rotation matrix and the gripper angle. A critical step of designing a gesture recognition model is exploring the optimal subset of the available kinematic data to use as input.

4.2. Data annotation

Each time step in the captures is assigned a label, representing a corresponding gesture. Although there is no standardized method for defining these gestures, they are gener-

ally considered as fundamental, deliberate actions with meaningful outcomes, varying depending on the context. The JIGSAWS dataset defines a vocabulary of 15 gestures. Ten of those can be seen in Table 1, used during the suturing task. The annotation was performed manually with the assistance of surgeons. As such, the task is subjective and prone to misrepresentations. The annotation was further reviewed in [7], where 12 mistakes were identified and corrected. Consequently, we will be utilizing those revised annotations in our study.

Table 1. Suturing task gestures

Label	Gesture Description
G1	Reaching for needle with right hand
G2	Positioning needle
G3	Pushing needle through tissue
G4	Transferring needle from left to right
G5	Moving to center with needle in grip
G6	Pulling suture with left hand
G8	Orienting needle
G9	Using right hand to help tighten suture
G10	Loosening more suture
G11	Dropping suture at the end and moving to end points

4.3. Prepocessing

Before being fed into a machine learning model, kinematic data is typically transformed into a format that aligns with the model's architecture, ensuring more effective utilization. A crucial step in this process is *normalization*, the process of rescaling the features to a common range, thereby ensuring that they have comparable magnitudes. This standardization facilitates the convergence of gradient-based optimization algorithms and enhances their efficiency. In this study, each trial was standardized independently by computing the mean and standard deviation of each feature column. The normalized value z_i is obtained by Eq. 11.

$$z_i = \frac{x_i - \mu}{\sigma} \tag{11}$$

Another important step when incorporating categorical data into an RNN model, such as gesture labels, involves *one-hot encoding*. This process entails assigning a unique index to each label and subsequently transforming it into a binary vector. The length of this vector corresponds to the total number of distinct labels, with a single element set to 1, representing the presence of the specific label, while all other elements are set to 0.

Due to the sequential nature of the data, each trial can have a different duration, preventing direct input into the model. To address this, a *sliding window* approach is employed. A fixed-width window is moved across the dataset with a fixed step, mimicking real-time data arrival and processing. To handle the initial portions of the sequence, padding values (-1) are introduced and should later be masked by the model.

4.4. Training, Validation, and Test Set Division

Before starting the training of the model, the dataset is split in three parts: the **training set**, the **validation set** and the **test set**. To ensure unbiased evaluation of the model's performance, a portion of the dataset should be initially isolated as the *test set*. It serves as a benchmark to assess the model's ability to generalize to new unseen data. The remaining data is then partitioned into two subsets. The *training set*, which constitutes the majority, at 85%, and consist of the data that the network is actually trained on, and the *validation set*, which is used during the training but only to evaluate the loss and any other metrics at the end of each epoch.

A significant challenge that is often observed during the preparation of the dataset, is the presence of an imbalanced class distribution. As evident from Fig.4, which depicts the frequency of each gesture within the dataset, some gestures like G10 ("Loosening more suture") are sparsely represented in the distribution.

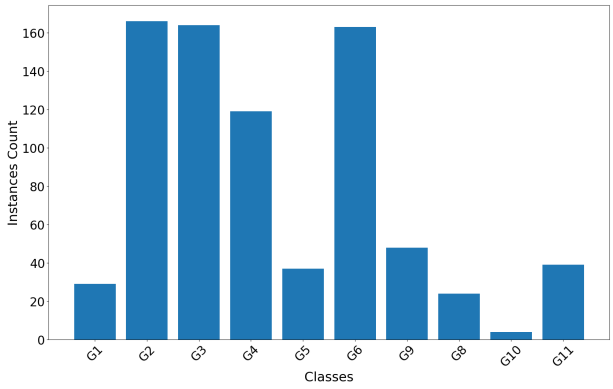

Figure 4. Number of instances per label

To mitigate this issue, we employed stratified sampling during the dataset splitting process. Stratification ensures that each class is proportionally represented in the training, validation and test sets, thereby improving the model's ability to learn, enhancing its overall performance.

5. Machine Learning Models for Gesture Recognition

5.1. LSTM architecture

We begin by establishing a strong baseline using an LSTM-based network, a well-suited architecture for sequential data, given its ability to capture long-term relationships.

The proposed architecture, as illustrated in Fig. 5, consists of multiple stacked layers, starting with the *input layer* that receives data sequences through the sliding window, followed by a *masking layer*, configured to exclude the value we used for padding the sliding window at preprocess (see Section 4.3).

The first LSTM layer consists of 256 units and also applies *L2 regularization*, a technique commonly used in machine learning in order to prevent over-fitting. This tech-

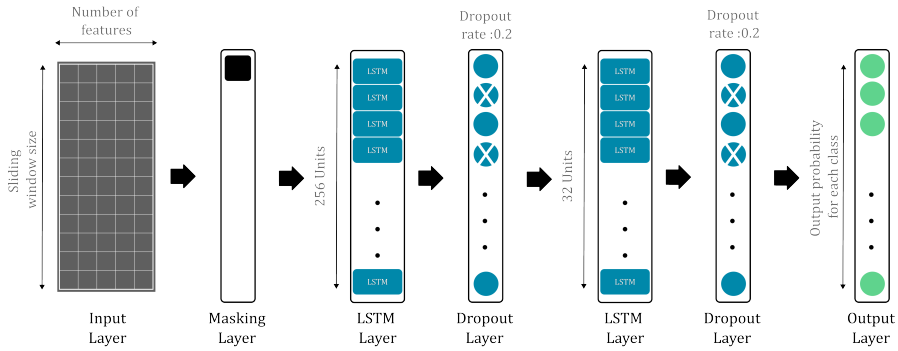

Figure 5. LSTM based model architecture

nique, also known as "ridge regression", penalizes large weights by adding a term proportional to the squared sum of their coefficients, as shown in eq. 13. This encourages smaller weights, leading to better generalization by reducing the model's susceptibility to noise in the training data. The second LSTM layer, with 32 units, similarly employs L2 regularization.

Following each LSTM layer, a *dropout layer* is introduced. By randomly deactivating a subset of neurons, "dropout" further enhances generalization by preventing excessive reliance on specific connections.

In multi-class classification problems, the network's output typically takes the form of a probability distribution vector, with a dimension equal to the number of classes. Each element represents the model's estimated probability assigned to each class, given the current input. The output layer uses the softmax activation function (Eq. 12) to ensure that all output probabilities are in the range [0,1] and sum to 1.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{12}$$

where:

x_i = input value for the i-th class (often the output of a linear layer)
K = total number of classes

Hyperband [30] was also utilized to optimize key hyperparameters for our LSTM model, such as the number of units, L2 regularization, and dropout rate. Hyperband is a sophisticated hyperparameter optimization technique that efficiently allocates computational resources by combining random search and successive halving. It effectively balances exploration (trying many different configurations) and exploitation (focusing on promising configurations) , accelerating the discovery of optimal hyperparameters while minimizing computational costs.

$$L_2 = \lambda \sum_{i=1}^{n} w_i^2 \tag{13}$$

where:

λ = regularization coefficient.
w_i = i-th weight in the model
n = total number of weights in the model

The network's loss function is the "Categorical Cross-Entropy", commonly used with the Softmax output activation function. This loss function measures the difference between the predicted and actual probability distributions. The total loss is calculated by summing the loss for each class as shown in Eq. 14.

$$L = -\sum_{i=1}^{10} t_i log(p_i) \tag{14}$$

where i is an iterator over the classes, t_i is the ground truth and p_i is the probability distribution produced by the Softmax.

The optimization algorithm used is "AdamW", a variant of the Adaptive Moment Estimation (Adam) algorithm. Adam is an adaptive learning rate optimization algorithm based on stochastic gradient descent and Root Mean Square Propagation (RMSProp). It incorporates momentum, calculated as an exponentially weighted moving average of past gradients, to accelerate convergence, assigning bigger weights to recent gradients while still considering the history of past gradients.

5.2. Feature engineering

Building on the foundation provided by the LSTM network, we sought to further enhance our model's performance by integrating additional techniques. These enhancements aim to capture more intricate patterns and dependencies within the sequential data.

One of the techniques considered to enhance the base model's performance is *feature engineering*, which involves transforming raw data into meaningful features. As presented in [29], video data are used to extract a state vector representing the interactions between surgical instruments and objects in the surgical scene. Since our work is focused solely on kinematic data, a state variable was generated by the combination of two existing kinematic features; the *left* and *right* gripper angles. These created a categorical context variable called the "Joint gripper state", with four possible values, viz.:

1. "Both closed"
2. "Left open - Right closed"
3. "Left closed - Right open"
4. "Both open"

5.3. Hybrid LSTM-CRF

In many areas of machine learning, especially natural language processing, researchers frequently attempt to improve the performance of RNNs by integrating them with statistical models. This has led to various hybrid models, such as those combining LSTM with HMMs [31,32] or CRFs [33].

Analysis of the dataset's transition matrix revealed a sparse distribution, indicating limited permissible state transitions (Figure 6). Most state pairs have zero transition probability, while a few allowed transitions exhibit a dominant probability compared to others.

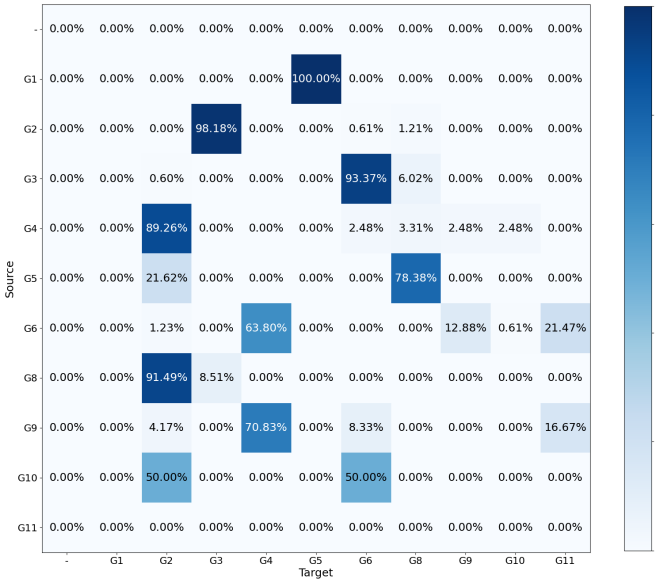

Figure 6. Transition probability matrix of the ground truth.

Since LSTMs excel at capturing long-term dependencies in sequential data, we combined them with a CRF to potentially exploit these unique transition characteristics. This combination allows the model to learn complementary patterns: LSTMs capture long-term context while CRFs attempt to model local label dependencies.

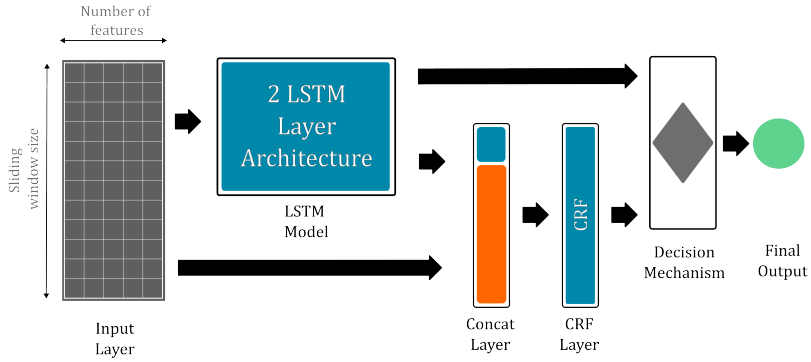

Figure 7. LSTM-CRF model architecture

In our proposed architecture (Figure 7), the CRF follows the previously described 2-layer LSTM network (Figure 5). We concatenate the LSTM's previous gesture pre-

diction with the original input and feed it to the CRF. This aims to refine the LSTM's output by incorporating transition information. In order to refine the LSTM output without compromising its reliability, we implemented a selective decision mechanism. This mechanism utilizes the CRF exclusively when the LSTM's prediction confidence falls below a predefined threshold. Specifically, the maximum probability of the LSTM's output sequence is compared to the threshold. If this probability is below the threshold, indicating low LSTM confidence, the CRF's prediction is selected as the final output. Through empirical evaluation, a threshold of 77% was determined to be adequate for this approach.

The CRF employs both L1 and L2 regularization with a coefficient of 0.1 for each. We utilize the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) optimization algorithm for efficient minimization of the objective function. L-BFGS uses an approximation of the inverse Hessian matrix of the objective function to find the search direction. The CRF training process is limited to a maximum of 100 iterations.

5.4. Hybrid LSTM-Attention

The second approach considered in this work to extend the baseline LSTM model and enhance its overall performance is based on the integration of of a MultiHead attention mechanism. This mechanism can give the model the unique ability to focus simultaneously on different parts of the data sequence and to model an estimation of the importance of each part.

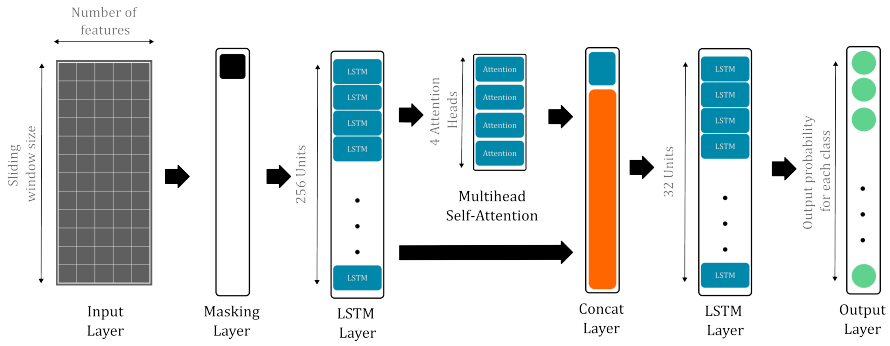

Figure 8. LSTM-Attention model architecture

Following a trial-and-error approach, we experimented with various configurations of the model's building blocks and found that placing the 4-head attention mechanism between the two LSTM layers was the most effective, resulting in the model illustrated in Fig. 8. The input sequence is processed by the first LSTM layer, which captures temporal dependencies and generates a sequence of hidden states. These hidden states are then refined by a multihead attention mechanism to model more complex dependencies within the data. The same sequence is used as query and value vector, resulting in *Self Attention*. The produced output is concatenated with the output of the LSTM, which is then fed into the second LSTM layer for further processing.

6. Experimental Results and Analysis

For model evaluation, we employed the Leave-One-Out Cross-Validation (LOOCV) method. This approach is the standard evaluation protocol for the JIGSAWS dataset, as its trials are pre-segmented by user. Given eight users, LOOCV provides an unbiased estimate of model performance while facilitating comparisons with existing work. In LOOCV, each user's trial is sequentially withheld for evaluation, while the model is trained on the remaining data. This process is repeated for all users, and the final performance metrics are averaged across all iterations. An analysis of each model is presented in the following.

6.1. LSTM Analysis

Our LSTM network was trained using exclusively kinematic data of the PSM. The dataset provides a range of kinematic variables. We initially utilized 14 of these variables; seven corresponding to each of the two tool tips:

- linear position (x, y, z)
- linear velocity (x', y', z')
- gripper angle (θ)

Following, six more variables were then added; three for each tool tip:

- rotational velocity (α', β', γ')

Lastly, the "Joint Gripper State" variable was added (see Section 5.2). The "Accuracy" and "F1 Score" results of the three feature configurations of the LSTM model, is presented in Table 2. We notice that the inclusion of the angular velocities and the Joint Gripper state plays a positive role in boosting the model's performance.

Table 2. Feature selection ablation study on base LSTM model

Features	Accuracy (%)	F1 score
LSTM with 14 features	78.82	0.597
LSTM with 20 features	79.47	0.6
LSTM with 20 features + Joint Gripper State	80.58	0.606

We have also conducted a hyperparameter sensitivity analysis to understand how *sliding window* size and *delay* influence our model's behavior. The results are presented in Tables 3 and 4 respectively. Varying the *window size* revealed a trade-off between capturing sufficient context and increased complexity. A larger window improves accuracy by providing more contextual information, but excessively large windows introduce computational burden and might introduce noise. Similarly, a larger delay might improve accuracy by allowing more thorough analysis but at the cost of responsiveness. To balance accuracy and real-time requirements, we allowed a maximum delay of *one* second.

Further analysis of the LSTM network performance revealed significant variations in accuracy across different users. Notably, user-specific gestures such as "Using right hand to help tighten the suture" (G9), posed a challenge. When G9 was prevalent in some trials and underrepresented in others, the model struggled to generalize, exhibiting de-

Table 3. Sliding window size analysis.

Sliding Window Size	Accuracy (%)	F1 score
3.2 sec	76.7	0.5855
6.4 sec	77.12	0.5891
12.8 sec	79.26	0.588
25.6 sec	75.99	0.5337

Table 4. Delay analysis.

Delay	Accuracy (%)	F1 score
0 sec	74.96	0.5697
0.2 sec	74.5	0.5738
0.5 sec	76.49	0.5867
1 sec	79.26	0.588

creased accuracy and confidence. See for example Fig. 9 where a trial of user B, with multiple instances of gesture G9, is compared to a trial of user F, with no usage of gesture G9. This suggests that class imbalance, particularly when concentrated within specific evaluation folds, hinders the network's ability to learn robust representations and accurately recognize underrepresented gestures. Furthermore, reducing the feature set from 20 to 14 exacerbated this issue, further emphasizing the impact of data scarcity on model performance.

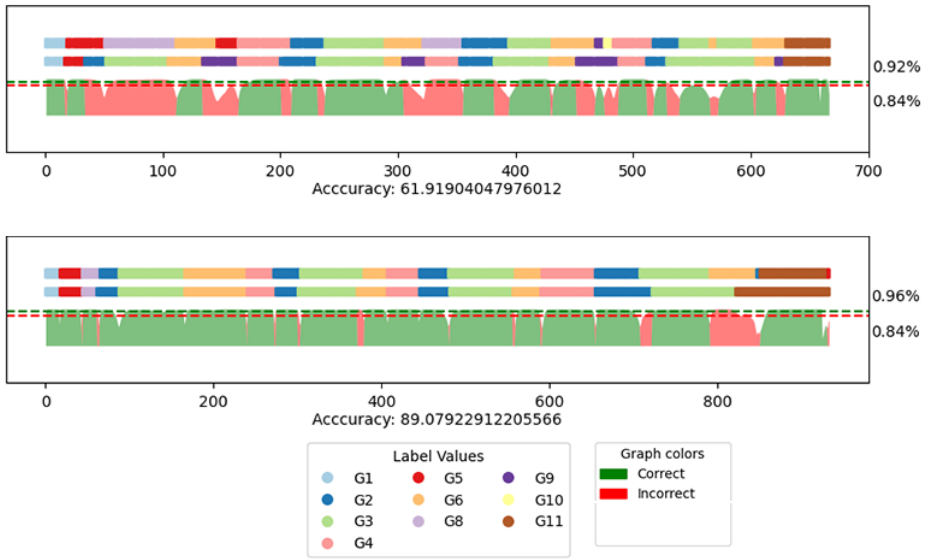

Figure 9. Top: LSTM Results for a user B trial with frequent usage of G9. Bottom: LSTM Results for a user F trial with high accuracy. Each subplot contains from top to bottom: LSTM output, ground truth , confidence curve.

By directly comparing the accuracy per class to the number of instances per class in Fig.(10) we can confirm that in most cases there is a strong correlation between the representation of the class and its recognition rate.

Another insightful visualization of the network's performance is the confusion matrix, shown in Fig. 11. Notably, it does not reveal any pairs of gestures that are consistently misclassified as each other in both directions. The minority classes (G9, G10, and G11) display a high level of misclassification across all classes. This pattern suggests that the network struggles to establish clear decision boundaries for these underrepresented classes, leading to unpredictable and seemingly random outputs.

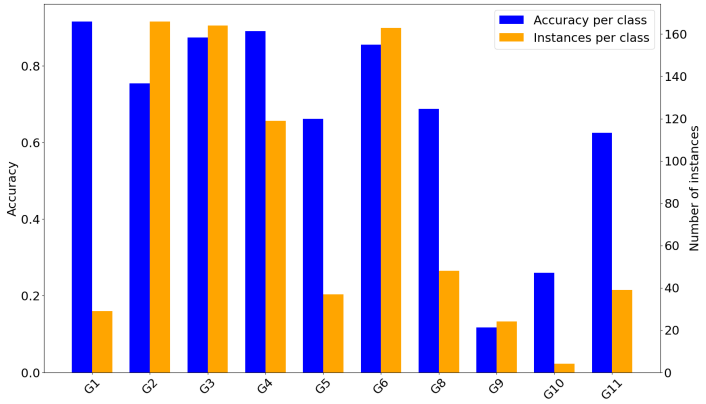

Figure 10. Left Axis: Accuracy per class - Right Axis: number of instances per class side-by-side. Results obtained from LSTM model on JIGSAWS dataset

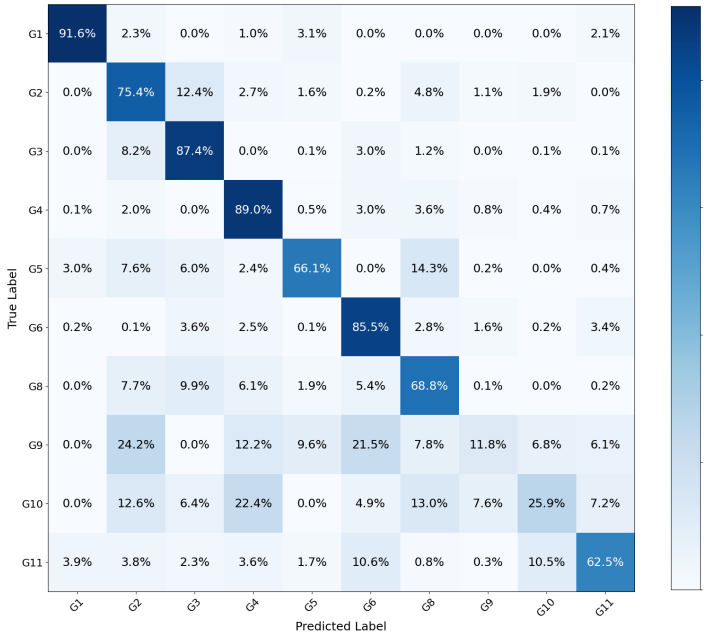

Figure 11. Confusion matrix for the LSTM model on the JIGSAWS dataset

6.2. Hybrid LSTM-CRF analysis

The LSTM-CRF hybrid model was introduced to leverage the strong transition probabilities observed in the dataset (see Section 5.3). The comparison to the corresponding matrix of the LSTM output (Fig. 12) reveals a small number of illegal transitions. The addition of the CRF attempts to correct some of the LSTMs misclassifications by utilizing the transition patterns between consecutive labels.

While the CRF model alone exhibited limitations, including over-segmentation (e.g., trial 1 in Fig. 13), using it in conjunction with an LSTM model such as the one introduced

Figure 12. Left: Transition probability matrix of the ground truth. Right: Transition probability matrix of the LSTM model

Table 5. Results on the JIGSAWS dataset with kinematic data that can be used in real time

Method	Accuracy (%)	year
Skip-Chain CRF [34]	80.29	2015
Forward LSTM [21]	80.5	2016
Our work		
Forward LSTM - 14 Features	**78.82**	2024
Forward LSTM - 20 Features	**79.47**	2024
Forward LSTM - 20 Features + Gripper State	**80.58**	2024
LSTM with self attention	**81.56**	2024

in Section 5.1, could potentially produce better results (e.g., trial 2 in Fig. 13). However, a simple threshold-based decision mechanism, as the one explained in 5.3, proved insufficient.

6.3. Hybrid LSTM-Attention analysis

In our final effort to enhance the model, we incorporated an attention layer, as described in Section 5.4. This addition improved the model's generalization performance, including its handling of trials featuring minority gestures. This architecture produced the best results out of all three approaches we tried, bearing an average accuracy of **81.56%**

6.4. Comparison with state of the art

Although numerous approaches have been proposed in the field of gesture recognition, many of which are evaluated on the *suturing task* of the JIGSAWS dataset, not all align with the criteria set in this work. Specifically, we focus on methods that rely *exclusively on kinematic data* and propose an architecture *suitable for online use*. When considering only studies that adhere to these constraints, our network outperforms the state of the art, achieving an accuracy of **81.56%**, as presented in Table 5.

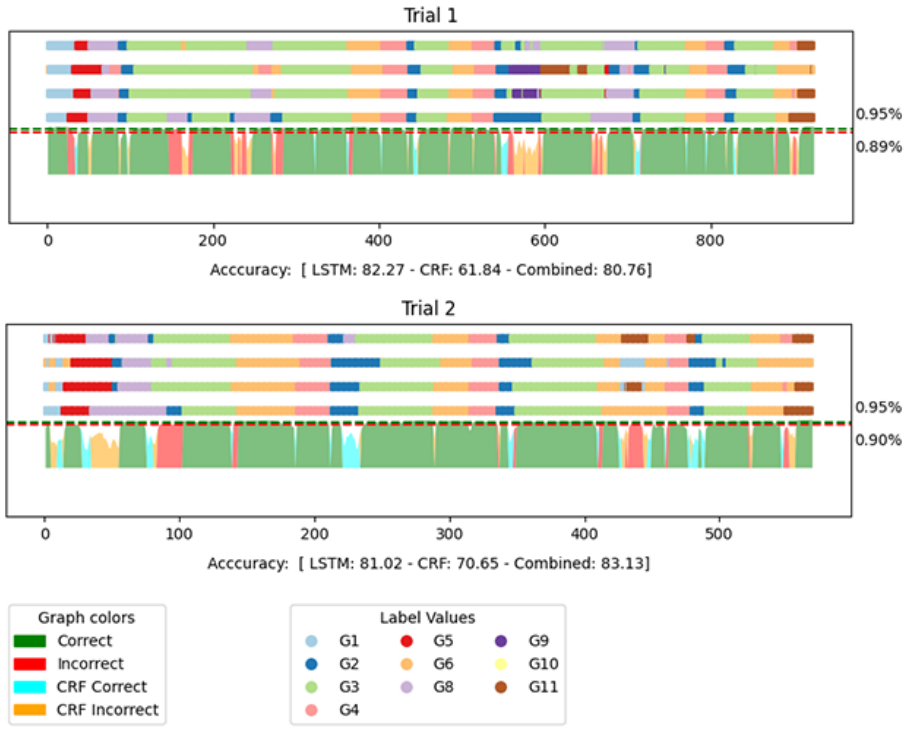

Figure 13. Results of "user B" using the Hybrid LSTM-CRF model. Each subplot contains from top to bottom: LSTM output, CRF output, Hybrid LSMT-CRF, output, ground truth - confidence curve.

7. Conclusion and Future Directions

In this work, we investigated the task of real-time gesture recognition during surgical procedures using machine learning techniques. We focused on suturing tasks within the JIGSAWS dataset and aimed to achieve state-of-the-art performance under specific constraints; (a) the model operated with a maximum 1-second delay using a sliding window, and (b) it was trained exclusively on kinematic data. Our initial approach employed an LSTM network, exploring variations such as *single* and *double* layers. We systematically enhanced the model through techniques like *hyperparameter tuning*, *dropout* and *L1/L2 regularization*. Recognizing the underperformance of classes with limited representation, we implemented stratification in the data splitting and introduced a higher penalty for misclassifying the underrepresented classes. We also enriched the feature set by incorporating appropriate variables and engineering a new feature based on the discrete gripper angle states.

To further boost performance, we explored two hybrid approaches: (i) combining LSTM predictions with kinematic data as input to a Conditional Random Field, leveraging the sparse transition structure of the data, and (ii) integrating a multi-head self attention mechanism at different points within the LSTM architecture (LSTM-Attention). Under the constraints imposed in this work, this hybrid LSTM-Attention architecture

achieved an accuracy of 81.56%, slightly outperforming relevant state-of-the-art approaches.

Building on the work presented in this study, several directions for future research can be proposed. Although we explored the idea of combining LSTM with CRF, there is room for improvement in the way the two model outputs are merged. A potential solution could be a more complex voting mechanism with separate class weights, based on the success rate for each class prediction by each model. This idea could be expanded in other hybrid models that might further enhance the LSTM abilities, such as models based on Transformers. Even though our model relied exclusively on kinematic data, future research could extend this work by incorporating visual information. This addition would enable the exploration and evaluation of a broader range of machine learning models. Additionally, the performance of the examined models can be assessed in a wider range of publicly available surgical datasets (see [1] for a comprehensive list) . Finally, the gesture recognition network could be integrated into a larger system that monitors surgical procedures and provides relevant information or assistance.

In conclusion, the work presented in this study lays a solid foundation for advancing real-time gesture recognition in surgical procedures. While several improvements and extensions have been proposed, the potential for enhancing model accuracy and applicability remains vast. The integration of the gesture recognition network into comprehensive surgical monitoring systems will constitute key steps in bringing this research closer to real-world applications. We envision that such advancements will make a significant contribution towards shaping the future of surgical robotics and assistive technologies.

References

[1] Maier-Hein L, Eisenmann M, Sarikaya D, März K, Collins T, Malpani A, et al. Surgical data science – from concepts toward clinical translation. Medical Image Analysis. 2022;76:102306. Available from: https://www.sciencedirect.com/science/article/pii/S1361841521003510.

[2] O'Toole M, Bouazza-Marouf K, Kerr D, Gooroochurn M, Vloeberghs M. A methodology for design and appraisal of surgical robotic systems. Robotica. 2010;28:297 310.

[3] Gao Y, Vedula SS, Reiley CE, Ahmidi N, Varadarajan B, Lin HC, et al. JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) : A Surgical Activity Dataset for Human Motion Modeling. In: MICCAI workshop: M2cai. vol. 3; 2014. Available from: https://api.semanticscholar.org/CorpusID:16185857.

[4] Huaulmé A, Sarikaya D, Le Mut K, Despinoy F, Long Y, Dou Q, et al. MIcro-surgical anastomose workflow recognition challenge report. Computer Methods and Programs in Biomedicine. 2021;212:106452. Available from: https://www.sciencedirect.com/science/article/pii/S0169260721005265.

[5] Mitsuishi M, Morita A, Sugita N, Sora S, Mochizuki R, Tanimoto K, et al. Master–slave robotic platform and its feasibility study for micro-neurosurgery. The International Journal of Medical Robotics and Computer Assisted Surgery. 2013;9(2):180-9. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/rcs.1434.

[6] van Amsterdam B, Funke I, Edwards E, Speidel S, Collins J, Sridhar A, et al. Gesture Recognition in Robotic Surgery With Multimodal Attention. IEEE Transactions on Medical Imaging. 2022;41(7):1677-87.

[7] van Amsterdam B, Clarkson MJ, Stoyanov D. Multi-Task Recurrent Neural Network for Surgical Gesture Recognition and Progress Prediction. In: 2020 IEEE International Conference on Robotics and Automation (ICRA); 2020. p. 1380-6.

[8] Campesato O. Artificial Intelligence, Machine Learning, and Deep Learning. Mercury Learning & Information; 2020. Available from: https://www.merclearning.com/titles/Artificial%20Intelligence-Machine-Learning-and-Deep-Learning.html.

[9] Goodfellow I, Bengio Y, Courville A. Deep Learning. Cambridge, Mass: The MIT Press; 2016.

[10] Roberts DA, Yaida S. The Principles of Deep Learning Theory: An Effective Theory Approach to Understanding Neural Networks. Cambridge: Cambridge University Press; 2022. Available from: https://www.cambridge.org/core/books/principles-of-deep-learning-theory/3E566F65026D6896DC814A8C31EF3B4C.

[11] Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. Physica D: Nonlinear Phenomena. 2020;404:132306. Available from: https://www.sciencedirect.com/science/article/pii/S0167278919305974.

[12] Airoldi EM. Getting Started in Probabilistic Graphical Models. PLoS Computational Biology. 2007;3(12):e252. Available from: http://dx.doi.org/10.1371/journal.pcbi.0030252.

[13] Lafferty JD, McCallum A, Pereira FCN. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of the Eighteenth International Conference on Machine Learning. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2001. p. 282–289.

[14] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All you Need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc.; 2017. Available from: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[15] Henry C Lin DY Izhak Shafran, Hager GD. Towards automatic skill evaluation: Detection and segmentation of robot-assisted surgical motions. Computer Aided Surgery. 2006;11(5):220-30. PMID: 17127647. Available from: https://doi.org/10.3109/10929080600989189.

[16] Varadarajan B, Reiley C, Lin H, Khudanpur S, Hager G. Data-Derived Models for Segmentation with Application to Surgical Assessment and Training. In: Yang GZ, editor. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. p. 426-34.

[17] Tao L, Elhamifar E, Khudanpur S, Hager GD, Vidal R. Sparse Hidden Markov Models for Surgical Gesture Classification and Skill Evaluation. In: Abolmaesumi P, Joskowicz L, Navab N, Jannin P, editors. Information Processing in Computer-Assisted Interventions. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012. p. 167-77.

[18] Zappella L, Béjar B, Hager G, Vidal R. Surgical gesture classification from video and kinematic data. Medical Image Analysis. 2013;17(7):732-45. Special Issue on the 2012 Conference on Medical Image Computing and Computer Assisted Intervention. Available from: https://www.sciencedirect.com/science/article/pii/S1361841513000522.

[19] Béjar Haro B, Zappella L, Vidal R. Surgical Gesture Classification from Video Data. In: Ayache N, Delingette H, Golland P, Mori K, editors. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012. p. 34-41.

[20] Tao L, Zappella L, Hager GD, Vidal R. Surgical Gesture Segmentation and Recognition. In: Mori K, Sakuma I, Sato Y, Barillot C, Navab N, editors. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 339-46.

[21] DiPietro R, Lea C, Malpani A, Ahmidi N, Vedula SS, Lee GI, et al. Recognizing Surgical Activities with Recurrent Neural Networks. In: Ourselin S, Joskowicz L, Sabuncu MR, Unal G, Wells W, editors. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016. Cham: Springer International Publishing; 2016. p. 551-8.

[22] DiPietro R, Ahmidi N, Malpani A, Waldram M, Lee G, Lee M, et al. Segmenting and classifying activities in robot-assisted surgery with recurrent neural networks. International Journal of Computer Assisted Radiology and Surgery. 2019;14:1-16.

[23] Cifuentes J, Boulanger P, Pham MT, Prieto F, Moreau R. Gesture Classification Using LSTM Recurrent Neural Networks. In: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC); 2019. p. 6864-7.

[24] Lea C, Vidal R, Reiter A, Hager GD. Temporal Convolutional Networks: A Unified Approach to Action Segmentation. In: Hua G, Jégou H, editors. Computer Vision – ECCV 2016 Workshops. Cham: Springer International Publishing; 2016. p. 47-54.

[25] Czempiel T, Paschali M, Keicher M, Simson W, Feussner H, Kim ST, et al. TeCNO: Surgical Phase Recognition with Multi-stage Temporal Convolutional Networks. In: Martel AL, Abolmaesumi P, Stoyanov D, Mateus D, Zuluaga MA, Zhou SK, et al., editors. Medical Image Computing and Computer

Assisted Intervention – MICCAI 2020. Cham: Springer International Publishing; 2020. p. 343-52.

[26] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016. p. 770-8.

[27] Heilbron FC, Escorcia V, Ghanem B, Niebles JC. ActivityNet: A large-scale video benchmark for human activity understanding. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2015. p. 961-70.

[28] Qin Y, Pedram SA, Feyzabadi S, Allan M, McLeod AJ, Burdick JW, et al. Temporal Segmentation of Surgical Sub-tasks through Deep Learning with Multiple Data Sources. In: 2020 IEEE International Conference on Robotics and Automation (ICRA); 2020. p. 371-7.

[29] Weerasinghe K, Reza Roodabeh SH, Hutchinson K, Alemzadeh H. Multimodal Transformers for Real-Time Surgical Activity Prediction. In: 2024 IEEE International Conference on Robotics and Automation (ICRA); 2024. p. 13323-30.

[30] Li L, Jamieson K, DeSalvo G, Rostamizadeh A, Talwalkar A. Hyperband: a novel bandit-based approach to hyperparameter optimization. J Mach Learn Res. 2017 Jan;18(1):6765–6816.

[31] Krakovna V, Doshi-Velez F. Increasing the Interpretability of Recurrent Neural Networks Using Hidden Markov Models. In: ICML Workshop on Human Interpretability (WHI 2016); 2016. Available from: `https://scholar.harvard.edu/files/finale/files/increasing_the_interpretability_of_recurrent_neural_networks_using_hidden_markov_models.pdf`.

[32] Hechavarria AA, Shafiq MO. Modeling and Predicting Online Learning Activities of Students: An HMM-LSTM based Hybrid Solution. In: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA); 2021. p. 682-7.

[33] Zavala RMR, Martínez P, Segura-Bedmar I. A Hybrid Bi-LSTM-CRF model for Knowledge Recognition from eHealth documents. In: TASS@SEPLN; 2018. Available from: `https://ceur-ws.org/Vol-2172/p6_hybrid_bi_lstm_tass2018.pdf`.

[34] Lea C, Hager GD, Vidal R. An Improved Model for Segmentation and Recognition of Fine-Grained Activities with Application to Surgical Training Tasks. In: 2015 IEEE Winter Conference on Applications of Computer Vision; 2015. p. 1123-9.