Bounded Future MS-TCN++ for Surgical Gesture Recognition

Adam Goldbraikh[1](✉), Netanell Avisdris[2], Carla M. Pugh[3], and Shlomi Laufer[4]

[1] Applied Mathematics Department, Technion – Israel Institute of Technology, 3200003 Haifa, Israel
sgoadam@campus.technion.ac.il

[2] School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, Israel
netana03@cs.huji.ac.il

[3] School of Medicine Stanford, Stanford University, Stanford, CA, USA
cpugh@stanford.edu

[4] Faculty of Industrial Engineering and Management, Technion – Israel Institute of Technology, 3200003 Haifa, Israel
laufer@technion.ac.il

Abstract. In recent times there is a growing development of video-based applications for surgical purposes. Part of these applications can work offline after the end of the procedure, other applications must react immediately. However, there are cases where the response should be done during the procedure but some delay is acceptable. In the literature, the online-offline performance gap is known. Our goal in this study was to learn the performance-delay trade-off and design an MS-TCN++-based algorithm that can utilize this trade-off. To this aim, we used our open surgery simulation data-set containing 96 videos of 24 participants that perform a suturing task on a variable tissue simulator. In this study, we used video data captured from the side view. The Networks were trained to identify the performed surgical gestures. The naive approach is to reduce the MS-TCN++ depth, as a result, the receptive field is reduced, and also the number of required future frames is also reduced. We showed that this method is sub-optimal, mainly in the small delay cases. The second method was to limit the accessible future in each temporal convolution. This way, we have flexibility in the network design and as a result, we achieve significantly better performance than in the naive approach.

Keywords: Surgical simulation · Surgical gesture recognition · Online algorithms

1 Introduction

Surgical data science is an emerging scientific area [21,22]. It explores new ways to capture, organize and analyze data with the goal of improving the quality of

L. Karlinsky et al. (Eds.): ECCV 2022 Workshops, LNCS 13803, pp. 406–421, 2023.
https://doi.org/10.1007/978-3-031-25066-8_22

interventional healthcare. With the increased presence of video in the operating room, there is a growing interest in using computer vision and artificial intelligence (AI) to improve the quality, safety, and efficiency of the modern operating room.

A common approach for workflow analysis is to use a two-stage system. The first stage is a feature extractor such as I3D [4] or ResNet50 [5,29]. The next stage usually includes temporal filtering. The temporal filtering may include recurrent neural networks such as LSTM [6,33,35], and temporal convolutional networks (TCN) such as MS-TCN++ [19] or transformers [25,34].

Automatic workflow analysis of surgical videos has many potential applications. It may assist in an automatic surgical video summarizing [2,20], progress monitoring [26], and the prediction of remaining surgery duration [32]. The development of robotic scrub nurses also depends on the automatic analysis of surgical video data [14,30]. In addition, video data is used for the assessment of surgical skills [3,9,10] and identifying errors [15,23,24].

Traditionally systems are divided into causal and acausal. However, not all applications require this dichotomic strategy. For example, the prediction of the remaining surgery duration may tolerate some delay if this delay ensures a more stable and accurate estimation. On the other hand, a robotic scrub nurse will require real-time information. In general, any acausal system may be transformed into a causal system if a sufficient delay is allowed. Where in the extreme case, the delay would be the entire video. This study aims to find the optimal system, given a constraint on the amount of delay allowed.

Many studies use Multi-Stage Temporal Convolutional Networks (MS-TCN) for workflow analysis [19]. It has both causal and acausal implementations. The network's number of layers and structure defines the size of its receptive field. In the causal case, the receptive field depends on past data. On the other hand, in the acausal case, half of the receptive field depends on future data and half on the past. Assume a fixed amount of delay T is allowed. A naive approach would be to use an acausal network with a receptive field $2 \cdot T$. However, this limits the number of layers in the network and may provide sub-optimal results. In this study, we develop and assess a network with an asymmetric receptive field. Thus we may develop a network with all the required layers while holding the constraint on the delay time T. This network will be called Bounded Future MS-TCN++ (BF-MS-TCN++). We will compare this method to the naive approach that reduces the receptive field's size by changing the network's depth. The naive approach will be coined Reduced Receptive Field MS-TCN++ (RR-MS-TCN++). We perform gesture recognition using video from an open surgery simulator to evaluate our method.

The main contribution of our work is the development of an MS-TCN++ with a bounded future window, which makes it possible to improve the causal network performance by delaying the return of output at a predetermined time. In addition, we evaluated a causal and acausal video-based MS-TCN++ for gesture recognition on the open surgery suturing simulation data.

2 Related Work

Lea *et al.* [16] was the first to study TCN's ability to identify and segment human actions. Using TCN, they segmented several non-surgical data sets such as 50 Salads, GETA, MERL Shopping, and Georgia Tech Egocentric Activities. They implemented causal and acausal TCNs and compared their performance on the MERL data set. The acausal solution provided superior results. They also outperform a previous study that uses an LSTM as a causal system and a Bidirectional LSTM as an acausal system. In TeCNO [5] causal Multi-Stage Temporal Convolutional Networks (MS-TCN) were used for surgical phase recognition. Two data-sets of laparoscopic cholecystectomy were used for evaluation. The MS-TCN outperformed various state-of-the-art LSTM approaches. In another study [29], this work was expanded to a multi-task network and was used for step and phase recognition of gastric bypass procedures.

In one study, the performance of an acausal TCN was assessed. The analysis included both non-surgical action segmentation datasets as well as a dataset of a simulator for robotic surgery [18]. Zhang *et al.* [36] used a Convolutional Network to extract local temporal information and an MS-TCN to capture global temporal information. They used acausal implantation to perform Sleeve Gastrectomy surgical workflow recognition. Not all studies use a separate network for capturing temporal information. In Funke *et al.* [8] a 3D convolutional neural networks was used. In this study, they used the sliding window approach to evaluate different look-ahead times.

The use of TCN is not limited to video segmentation. It has been studied in the context of speech analysis as well [27]. In this context, the relationship between delay and accuracy has been assessed [28].

3 Methods

3.1 Dataset

Eleven medical students, one resident, and thirteen attending surgeons participated in the study. Their task was to place three interrupted instrument-tied sutures on two opposing pieces of the material. Various materials can simulate different types of tissues; for example, in this study, we used tissue paper to simulate a friable tissue and a rubber balloon to simulate an artery. The participants performed the task on each material twice. Thus, the data set contains 100 procedures, each approximately 2–6 min long. One surgeon was left-handed and was excluded from this study. Thus, this study includes a total of 96 procedures. Video data were captured in 30 frames per second, using two cameras, providing top and side views. In addition to video, kinematic data were collected using electromagnetic motion sensors (Ascension, trakSTAR). For this study, we only use the side-view camera. We perform a gesture recognition task, identifying the most subtle surgical activities within the surgical activity recognition task family. Six surgical gestures were defined: G0 - nonspecific motion, G1 - Needle passing, G2 - Pull the suture, G3 - Instrumental tie, G4 - Lay the knot G5 - Cut the suture. The video data were labeled using Behavioral Observation Research Interactive Software (BORIS) [7].

3.2 Architecture

MS-TCN++ [19] is a *temporal convolutional network* (TCN) designed for video data activity recognition. The input for this network is a vector of features extracted from the raw video using a CNN, such as I3D [4]. The video length is not predetermined. Let us assume that the video is given in 30 frames per second and contains T frames, namely, T is a parameter of a specific video. In the following sections, we will describe the different components of the MS-TCN++ and the modifications made for the BF-MS-TCN++. It should be noted that the naive approach, RR-MS-TCN++ has the same structure as the acausal MS-TCN++ and was coined with a unique name to emphasize the limitation on the receptive field size.

The architecture of MS-TCN++ is structured from two main modules: the prediction generator and the refinement. For the sake of simplicity, we will first describe the refinement module structure and then the prediction generator.

Refinement Module: As shown in Fig. 1, the refinement module includes several refinement stages, where the output of each stage is the input of the next. The refinement stage is a pure TCN that uses *dilated residual layers* (DRL). This allows the module to handle varying input lengths. To match the input dimensions of the stage with the number of feature maps, the input of the refinement stage passes through a 1×1 convolutional layer. Then these features are fed into several DRLs where the dilation factor is doubled in each layer. The dilation factor determines the distance between kernel elements, such that a dilation of 1 means that the kernel is dense. Formally, the dilation factor is defined as $\delta(\ell) = 2^{\ell-1} : \ell \in \{1, 2, \ldots L\}$, where ℓ is the layer number and L is the total number of layers in the stage. In MS-TCN++, the DRL is constructed from an acausal dilated temporal convolutional layer (DTCL), with a kernel size of 3, followed by ReLU activation and then a 1×1 convolutional layer. The input of the block is then added to the result by a standard residual connection, which is the layer's output. To get the prediction probabilities, the output of the last DRL passes through a prediction head which includes a 1×1 convolution, to adjust the number of channels to the number of classes, and is activated by a softmax.

Prediction Generation Module: The prediction generator consists of only one prediction generation stage. The general structure of this stage is similar to the refinement stage; however, instead of a DRLs, it has a *dual dilated residual layers* (DDRLs). Let's consider layer $\ell \in \{1, 2, \ldots, L\}$. The input of the DDRL is entered into two DTCLs, one with a dilation factor of $\delta_1(\ell) = 2^{\ell-1}$ and the other with a dilation factor of $\delta_2(\ell) = 2^{L-\ell}$. Then, the outputs of the two DTCLs are merged by concatenation of the feature in the channel dimension, followed by a

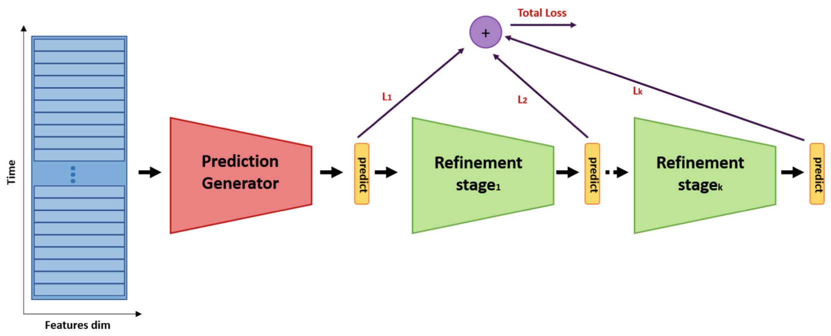

Fig. 1. General structure of MS-TCN++. The input is a vector of features (blue). It composes of multiple stages, where each stage predicts the frame segmentation. The first stage is the prediction generator (red) and the other are refinement stages (green), which can be any number ($k \geq 0$) of them. The loss is computed over all stages' predictions. (Color figure online)

1D convolutional layer to reduce the number of channels back to the constant number of feature maps. This output passes through a ReLU activation and an additional 1D convolutional layer before the residual connection. For a formal definition of MS-TCN++ stages and modules, see [19].

Future Window Size Analysis: As MS-TCN++ is a temporal convolutional network with different dilatation among the layers, analyzing the temporal dependence is not trivial. In order to determine the number of future frames involved in the output calculation, careful mathematical analysis is required. Calculating the number of future frames required is equivalent to the output delay of our system.

In the naive approach, RR-MS-TCN++, the number of layers of the network governs the receptive field and the future window. The BF-MS-TCN++ is based on the limitation of the accessible future frames in each temporal convolution; hence the name Bounded Future. This section aims to analyze both methods and calculate desired future window.

The input and the output of DRLs and DDRLs (i.e. (D)DRL) have the same dimensions of $N_f \times T$, where N_f represents the number of feature maps in the vector encoding the frame and T is the number of frames in the video. We assume that for every (D)DRL, the vector in the t index represents features that correspond to the frame number t. However, in the acausal case, the features of time t can be influenced by a future time point of the previous layer output. In MS-TCN++, (D)DRL has symmetrical DTCLs with a kernel size of 3. The layer's input ℓ is padded by $2 \times \delta(\ell)$ zero vectors to ensure the output dimensions are equal to the input dimensions. A symmetric convolution is created by padding the input with $\delta(\ell)$ number of zeros vectors before and after it. The result is that half of each layer's receptive field represents the past and the other half represents the future. To obtain a causal solution, all $2 \times \delta(\ell)$ zero vectors are added before the input. As a result, the entire receptive field is based on the past time. This method has been used in [5,16].

Let $S = \{PG, R, Total\}$ be a set of symbols, where PG represents relation to the prediction generator, R to the refinement stage, and $Total$ to the entire network. Let $L_s : s \in S$ be the number of (D)DRLs in some stage or in the entire network. The number of refinement stages in the network is N_R. Note that we assume that all refinement stages are identical, and that $L_{Total} = L_{PG} + N_R \cdot L_R$. Let the ordered set $\mathcal{L}_{total} = (1, 2, \ldots, L_{Total})$, where $\ell \in \mathcal{L}_{total}$ represents the index of ℓ^{th} (D)DRL in the order it appears in the network. Given integer x, $[x]$ denotes the set of integers satisfying $[x] = \{1, \cdots, x\}$.

Definition 1. *Let DTCL ϕ with dilation factor of $\delta(\phi)$. The* Direct Future Window *of a ϕ is $m \in [\delta(\phi)]$ if and only if the number of the padding vectors after the layers input is m and number of padding vectors before the vector is $2m - \delta(\phi)$. The function $DFW(\phi) = m$ gets a DTCL and returns it's direct future window.*

The definition of a Direct Future Window is shown in Fig. 2.

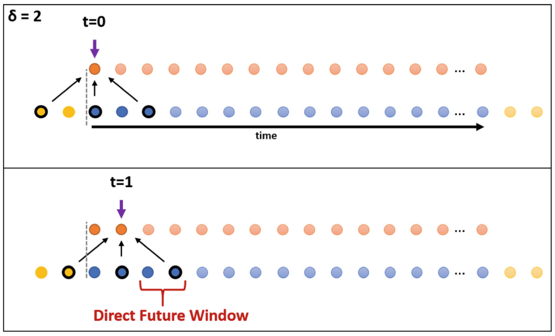

Fig. 2. Illustration of Direct Future Window of a dilated temporal convolutional layer (DTCL) with $\delta = 2$, for first (upper part) and second (lower part) timeframes. Blue dots denote input features, yellow dots denote padding needed for DTCL, and orange dots denotes the output of DTCL. (Color figure online)

Reduced Receptive Field MS-TCN++ Case: In the DRLs, in the refinement stages, the direct future window is equal to the direct future window of it's DTCL. Formally, let $\ell \in [L_R]$ a DRL, that contains a DTCL ϕ. The direct future window of this layer is given by $\mathcal{DFW}_R(\ell) = DFW(\phi)$, where the subscript R indicates an association with a refinement stage. However, each DDRL contains two different DTCLs ϕ_1 and ϕ_2, as described in Sect. 3.2. Consider a DDRL $\ell \in [L_{PG}]$. The direct future window of this layers is given by Eq. 1.

$$\mathcal{DFW}_{PG}(\ell) = \max\{DFW(\phi_1), DFW(\phi_2)\} \tag{1}$$

Definition 2. *The* Future Window *of layer $\ell \in \mathcal{L}_{total}$ defined as follows:*

$$FW(\ell) = \sum_{i \in [\ell]} \mathcal{DFW}(i)$$

Figure 3 shows how direct future windows are aggregated to the future window.

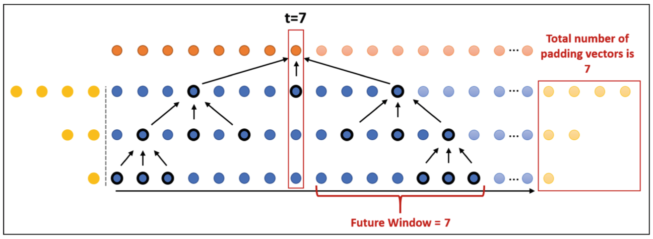

Fig. 3. Illustration of Future Window in a refinement stage with three dilated residual layers (DRLs).

Based on Definition 2, the future window of the RR-MS-TCN++ network is defined in Eq. 2.

$$FW^{RR}(L_{Total}) = \sum_{\ell \in [L_{PG}]} \max\{2^{\ell-1}, 2^{L_{PG}-\ell}\} + N_R \cdot \sum_{\ell \in [L_R]} 2^{\ell-1} \tag{2}$$

Note that superscript RR indicates that the network is RR-MS-TCN++. According to Definition 2, Eq. 2 is obtained by summing the prediction generator and refinement stages separately. In addition, the fact that in the prediction generator, for every DDRL $\ell \in [L_{PG}]$ there exists two DTCLs ϕ_1, ϕ_2 that satisfied $DFW(\phi_1) = 2^{\ell-1}$, and $DFW(\phi_2) = 2^{L_{PG}-\ell}$, yields that the direct future window of the DDRL is the maximum between these terms, as defined in Eq. 3. We take the maximum since the direct future window is determined by the furthest feature in the layer's input that participates in the calculation.

$$\mathcal{DFW}_{PG}(\ell) = \max\{2^{\ell-1}, 2^{L_{PG}-\ell}\} \tag{3}$$

and for some DRL $\ell \in [L_R]$ in the refinement $\mathcal{DFW}(\ell) = 2^{\ell-1}$.

Bounded Future MS-TCN++ Case: Let w_{max} be a bounding parameter that bounds the size of the direct future window. We determine that the direct future window of every DRL $\ell \in [L_R]$, in the refinement stage of the BF-MS-TCN++, is given by Eq. 4.

$$\mathcal{DFW}_R^{BF}(\ell) = \min\{w_{max}, \delta(\ell)\} = \min\{w_{max}, 2^{\ell-1}\} \tag{4}$$

The superscript BF indicates that the network is BF-MS-TCN++ and the subscript R indicates that this belongs to the refinement stage, where the replacement of R with PG indicates association with the prediction generator.

Figure 4 illustrates how the convolution's symmetry is broken in the case of $\delta(\ell) > w_{max}$.

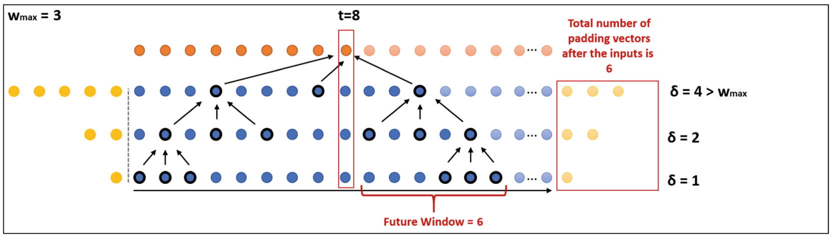

Fig. 4. Illustration of a Future Window and asymmetry in padding in a refinement stage of Bounded Future MS-TCN++ with three dilated residual layers (DRLs) and $w_{max} = 3$. In the last layer, since the dilation factor δ is larger than w_{max}, the number of padding vectors (yellow) before and after the sequence is different.

The future window of the refinement stage is given by Eq. 5.

$$FW_R^{BF}(L_R) = \sum_{\ell \in [L_R]} \mathcal{DFW}_R^{BF}(\ell) = \sum_{\ell \in [L_R]} \min\{w_{max}, 2^{\ell-1}\} \tag{5}$$

For the DDRL $\ell \in [L_{PG}]$ of the prediction generators, the direct future window is given by Eq. 6

$$\mathcal{DFW}_{PG}^{BF}(\ell) = \max\{min\{w_{max}, 2^{\ell-1}\}, min\{w_{max}, 2^{L_{PG}-\ell}\}\} \tag{6}$$

Hence the future window of the prediction generator for a causal network with delay is given by Eq. 7.

$$FW_{PG}^{BF}(L_{PG}) \sum_{\ell \in [L_{PG}]} \mathcal{DFW}_{PG}^{BF}(\ell) \tag{7}$$

$$= \sum_{\ell \in [L_{PG}]} \max\{\min\{w_{max}, \delta_1(\ell)\}, \min\{w_{max}, \delta_2(\ell)\}\}$$

This leads to Eq. 8 which presents the future window for the entire network.

$$FW^{BF}(L_{Total}) = \sum_{\ell \in [L_{PG}]} \max\{\min\{w_{max}, 2^{\ell-1}\}, \min\{w_{max}, 2^{L_{PG}-\ell}\}\} \tag{8}$$

$$+N_R \cdot \sum_{\ell \in [L_R]} \min\{w_{max}, 2^{\ell-1}\}$$

Note that a network with $w_{max} = 0$ is a causal network, which may have a fully online implementation.

3.3 Feature Extractor Implementation Details

As a first step, we trained a (2D) EfficientNetV2 medium [31] in a frame-wise manner; namely, the label of each frame is its gesture. The input was video frames with a resolution of 224×224 pixels. For each epoch, we sampled in a class-balanced manner 2400 frames, such that each gesture appears equally among the sampled frames. The frames were augmented with corner cropping, scale jittering, and random rotation. The network was trained for 100 epochs, with a batch size of 32. Cross-entropy loss was used. All the experiments were trained using an Adam optimizer, with an initial learning rate of 0.00025 that was multiplied by a factor of 0.2 after 50 epochs, and decay rates of $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The code of this part is based on the code provided by Funke *et al.* [8]. After the individual training of each split, the one before the last linear layer was extracted and used as a feature map for the MS-TCN++.

4 Experimental Setup and Results

4.1 Experimental Setup

In order to evaluate the effect of the delay on the performance of online algorithms we compare two methods, the naive **RR-MS-TCN++** and our **BF-MS-TCN++**. To this aim, we performed a hyperparameter search. For both networks, the number of refinement stages and the number of (D)DRLs inside the stages affect the total receptive field and hence the future window. The uniqueness of the BF-MS-TCN++ is that the future window can be limited by the bounding parameter W_{max} as well, regardless of the values of the other parameters. In our search, we forced the number of DRLs in the prediction generator to be equal to the number of DRLs in the refinement stages, namely $L_{PG} = L_R = L$.

To this end, for the RR-MS-TCN++ network, the number of DRLs L included in the search was in the range of $\{2, 3, 4, 5, 6, 8, 10\}$. The number of refinement stages N_R was in the range of $\{0, 1, 2, 3\}$. For the BF-MS-TCN++, the search included two grids. In the first grid, the values of w_{max} were in the range of $\{0, 1, 2, 3, 6, 7, 8, 10, 12, 13, 14, 15, 16, 17, 20\}$, where 0 represents a online algorithm. The number of DRLs L was in the range of $\{6, 8, 10\}$, and the number of refinement stages N_R was in the of $\{0, 1, 2, 3\}$. In the second grid, the values of w_{max} were in the range of $\{1, 3, 7, 10, 12, 15, 17\}$. The number of DRLs L was in the range of $\{2, 3, 4, 5\}$, and the number of refinement stages N_R was in the of $\{0, 1, 2, 3\}$. In total, we performed 320 experiments. The rest of the hyperparameters remained constant, where the learning rate was 0.001, dropout probability was 0.5, the number of feature maps was 128, batch size of 2 videos, the number of epochs was 40, and the loss function was the standard MS-TCN++ loss with hyperparameters $\tau^2 = 16$ and $\lambda = 1$. All experiments were trained with an Adam optimizer with the default parameters. Training and evaluation were done using a DGX cluster with 8 Nvidia A100 GPUs.

4.2 Evaluation Method

We evaluated the models using 5-fold cross-validation. All videos of a specific participant were in the same group (leave-n-users-out approach). The videos assigned to the fold served as the test set of that fold. The remaining participants' videos were divided into train and validation sets.

The validation set for fold $i \in [5]$ consists of 3 participants from fold $(i + 1) \mod 5$. Namely, for each fold, 12 videos were used as a validation set (3 participants $\times$ 4 repetitions). With this method, we create unique validation sets for each fold. The stopping point during training was determined based on the best F1@50 score on the validation set. The metrics were calculated for each video separately, so the reported results for each metric are the mean and the standard deviation across all 96 videos when they were in the test set.

4.3 Evaluation Metrics

We divide the evaluation metrics into two types: segmentation metrics and frame-wise metrics. The segmentation metrics contain F1@k where $k \in \{10, 25, 50\}$ [16], and edit distance [17]. While the frame-wise metrics included Macro-F1 [12] and Accuracy. We calculated each metric for each video and then averaged them across all videos.

F1@k: The intersection over union (IoU) between the predicted segments and the ground truth was calculated for each segment. If there is a ground truth segment with IoU greater than k, that ground truth segment is marked as true positive and is not available for future use. Otherwise, the predicted segment has been defined as a false positive. Based on these calculations, the F1 score was determined.

Segmental Edit Distance: The segmental edit distance is based on the Levenshtein distance, where the role of a single character is taken by segments of the activity. The segmental edit distance was calculated for all gesture segments in the video and normalized by dividing it by the maximum between the ground truth and prediction lengths.

Frame-Wise Metrics: We calculated the Accuracy and the multi-class F1-Macro scores as used in [11].

4.4 Experimental Studies

We performed four studies: (1) Baseline estimation; (2) Performance comparison; (3) Network hyperparameter importance; and (4) Competitive analysis.

(1) First, we try to estimate the best fully casual and acausal networks, which will serve as a baseline.
(2) In the *Performance comparison* study we compare directly between the performance of *Reduced Receptive Field MS-TCN++* and our *Bounded Future MS-TCN++*, with respect to Future Window size. We defined 12 Future Window intervals: $[0, 0.001, 0.125, 0.25, 0.5, 1, 2, 4, 8, 16, 32, 64, \infty]$ seconds. For each method separately, we divide the networks into these intervals considering the Future Window. A representative value of each interval was selected based on the highest results within each interval. Notice that a few intervals may be left empty.
(3) In the third study, we analyzed the *marginal performance* and *marginal importance* of the investigated hyperparameters on the F1@50 using the fANOVA method [13]. The receptive field is determined by the hyperparameters of our network structure. To understand the advantages and weaknesses of each of our methods, it's essential to assess the importance and trends of the hyperparameters.
(4) In the *Competitive analysis* study, we have two aims. First, it is to try to reveal what is the required delay to approach the best offline performance. Next, we aim to determine which method is more advantageous at which delay values. To this end, we need to define two new metrics: Global and local competitive ratio. In this paper, the *Competitive-Ratio*, inspired by its definition in theoretical analysis of online algorithms [1], will be the ratio between the performance of the causal (with delay) algorithm and the best acausal network. In addition, for each interval, we define the Local Competitive Ratio as the ratio between the best BF-MS-TCN++ and RR-MS-TCN++ networks in that interval.

4.5 Results

Table 1 lists the results of the baseline estimation study. The feature extractor performed relatively well in a frame-wise manner with an accuracy of 82.66 and F1-Macro of 79.46. Both the causal and acausal networks improve the accuracy and F1-Macro scores, however, the acausal network has a significant effect in both frame-wise metrics while in the causal case only the accuracy has been improved significantly. The Causal case has the best results in all metrics. Figure 5 shows a performance comparison study, where the trend is similar for all metrics, where BF-MS-TCN++ outperforms RR-MS-TCN++, especially for small delay values.

In the feature importance study (Fig. 6), w_{max} was found to have negligible importance. Other hyperparameters perform better when their values are increased. Lastly, the Competitive analysis results are illustrated in Fig. 7. In the global analysis (left) plot, it is evident that BF-MS-TCN++ has 80% of the performance of the best offline algorithm after only one second and 90% after $2\frac{1}{3}$ s. In the local analysis (right) plot, it is seen clearly that for future windows smaller than one second the BF-MS-TCN++ is significantly preferred over the RR-MS-TCN++.

Table 1. The Feature extractor EfficientNet v2, causal and acausal MS-TCN++ results on a gesture recognition task. Bold denotes best results for metric.

	F_1-Macro	Accuracy	Edit distance	F1@10	F1@25	F1@50
EfficientNet v2	79.46 ± 8.10	82.66 ± 6.03	–	–	–	–
Causal MS-TCN++	80.42 ± 8.67	85.04 ± 5.77	64.69 ± 12.36	74.30 ± 10.85	72.34 ± 12.04	64.96 ± 14.07
Acausal MS-TCN++	**83.85 ± 8.90**	**86.94 ± 6.50**	**84.65 ± 9.25**	**88.66 ± 7.79**	**87.13 ± 9.03**	**80.01 ± 13.21**

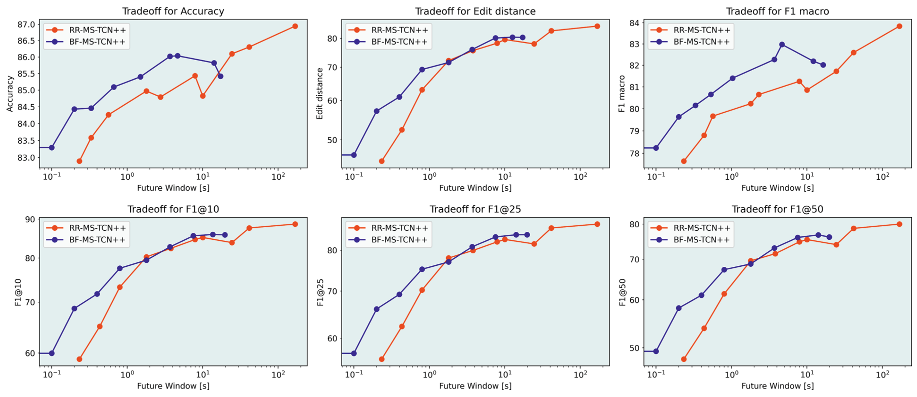

Fig. 5. performance-delay trade-off. Comparison of best BF-MS-TCN++ and RR-MS-TCN++ networks, in respect to future window size. The plots show the performance in terms of Accuracy, Edit distance, F1-Macro, and $F1@k,\ k \in \{10, 25, 50\}$.

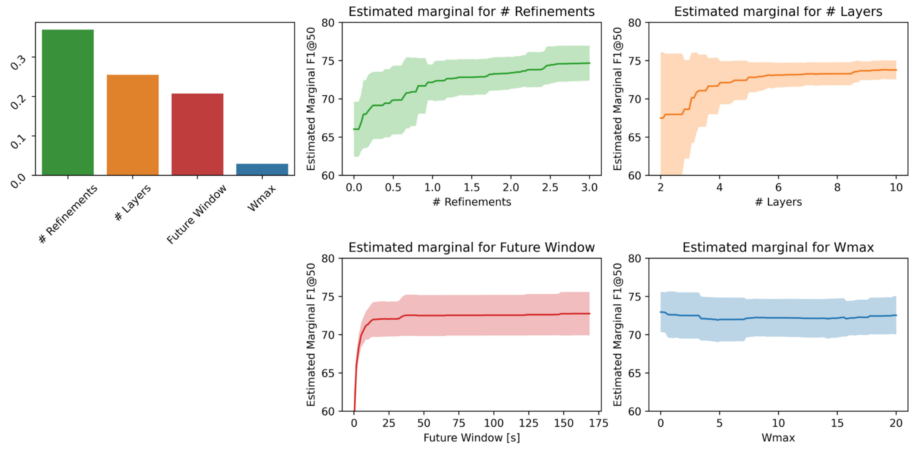

Fig. 6. fANOVA analysis of BF-MS-TCN++. Marginal importance (leftmost graph) and estimated marginal F1@50 on architecture hyperparameters: number of refinement stages, number of (D)DRLs in every stage, the Future Window of the network, and the value of w_{max}.

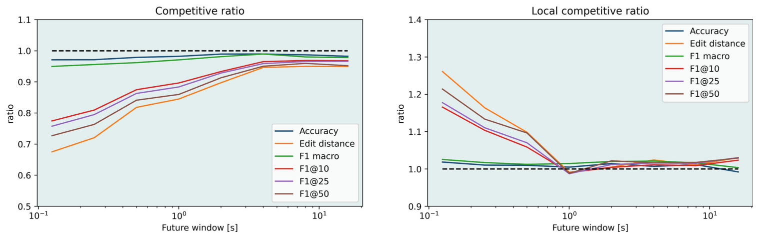

Fig. 7. Competitive ratio analysis between BF-MS-TCN++ and RR-MS-TCN++, for best performing network (left) and best performing with respect to future window (right) vs time [s] (X-Axis). Competitive ratio larger than 1 means that BF-MS-TCN++ performs better than RR-MS-TCN++. The dotted black line denotes the baseline (No competitive advantage).

5 Discussion and Conclusions

Automated workflow analysis may improve operating room efficiency and safety. Some applications can be used offline after the procedure has been completed, while other tasks require immediate responses without delay. Nevertheless, some applications require real-time yet may allow a delayed response, assuming it improves accuracy.

Studies showed that there is a performance gap between causal and acausal systems [36]. To choose the optimal network that compromises between delay and performance, it is necessary to investigate how the delay affects performance. Funke *et al.* [8] investigated the effect of delay on a 3D neural network. They found that adding delay improves the system's performance, primarily in segmentation metrics such as segmental edit distance and F1@10. Nevertheless, today these networks are considered relatively weak compared to the newer activity recognition networks that are typically based on transformers and temporal convolutional networks. In these algorithms, designing a future window is not trivial as in the 3D convolutional case.

In this work, we developed and analyzed different variations of the MS-TCN++, and studied the trade-off between delay and performance. This study sought to verify the intuition that adding a relatively small delay in the causal system's response, which usually operates in real-time, could also help bridge the causal-acausal gap in MS-TCN++. We examined this hypothesis in two methods. First, we tried to examine reducing the network's depth, thus the receptive field was reduced, with half of it being the future, namely Reduced Receptive field MS-TCN++, RR-MS-TCN++. We expected that this method would not work well for small future window sizes because reducing the window size in this method means a significant decrease in the number of layers or even eliminating of few refinement stages. Li *et al.* [19], showed these parameters have a crucial effect on performance. Therefore we developed the BF-MS-TCN++. This network involved applying a convolution in which the target point in the previous layer is not in the middle of the convolution's receptive field. This way, we bound the future window of the entire network even in large architectures with a small delay. With this method, different architectures can be implemented for approximately the same delay values. Thus in our analysis, several time intervals were defined for both methods, and we selected the network with the best results to represent each interval.

According to Fig. 5, the performance comparison study illustrates the trade-off between performance and delay, considering all metrics in both methods. As seen in Table 1, the gap between causal and acausal networks is much more prominent in the segmentation metrics than in the frame-wise metrics. Similarly, in Fig. 5 and Fig. 7, these metrics exhibit a stronger trade-off effect. Furthermore, BF-MS-TCN++ outperforms RR-MS-TCN++ especially when small delays are allowed. In the left image of Fig. 7, a future window of $2\frac{1}{3}$ s achieves more than 90% of the best offline network. Namely, getting close to an offline network's performance is possible even with a relatively small delay.

The fANOVA test showed, in Fig. 6, that the number of (D)DRLs and the number of refinement stages are the most important hyperparameters for optimizing performance, even more than the total delay. Where increasing these hyperparameter values tends to improve estimated marginal performance. Thus our BF-MS-TCN++, which allows the design of larger networks with that same delay factor, takes advantage of this fact. Another interesting finding is that the value of w_{max} barely affects the outcome (Fig. 6), whereas the total future delay plays a pivotal role. Thus, we conclude that minimizing the value of w_{max} for designing larger networks is an acceptable approach.

This study has a few limitations. First, it has been evaluated using only one data set. In addition, only data from surgical simulators were analyzed. Larger data sets, including data from the operating room, should be analyzed in the future.

The algorithms presented in this study are not limited to the surgical domain. Online with delay activity recognition is relevant to many other applications that evaluate human performance, even outside the surgical field. Therefore, this study lays the foundations for a broader study on the relationship between accuracy and delayed response in video-based human activity recognition.

References

1. Albers, S.: Online algorithms: a survey. Math. Program. **97**(1), 3–26 (2003)
2. Avellino, I., Nozari, S., Canlorbe, G., Jansen, Y.: Surgical video summarization: multifarious uses, summarization process and ad-hoc coordination. Proc. ACM Hum.-Comput. Interact. **5**(CSCW1), 1–23 (2021)
3. Basiev, K., Goldbraikh, A., Pugh, C.M., Laufer, S.: Open surgery tool classification and hand utilization using a multi-camera system. Int. J. Comput. Assisted Radiol. Surg. **17**, 1497–1505 (2022)
4. Carreira, J., Zisserman, A.: Quo Vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6299–6308 (2017)

5. Czempiel, T., et al.: TeCNO: surgical phase recognition with multi-stage temporal convolutional networks. In: Martel, A.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12263, pp. 343–352. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59716-0_33
6. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2625–2634 (2015)
7. Friard, O., Gamba, M.: Boris: a free, versatile open-source event-logging software for video/audio coding and live observations. Methods Ecol. Evol. **7**, 1325–1330 (2016). https://doi.org/10.1111/2041-210X.12584
8. Funke, I., Bodenstedt, S., Oehme, F., von Bechtolsheim, F., Weitz, J., Speidel, S.: Using 3D convolutional neural networks to learn spatiotemporal features for automatic surgical gesture recognition in video. In: Shen, D., et al. (eds.) MICCAI 2019. LNCS, vol. 11768, pp. 467–475. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32254-0_52
9. Funke, I., Mees, S.T., Weitz, J., Speidel, S.: Video-based surgical skill assessment using 3D convolutional neural networks. Int. J. Comput. Assist. Radiol. Surg. **14**(7), 1217–1225 (2019)
10. Goldbraikh, A., D'Angelo, A.L., Pugh, C.M., Laufer, S.: Video-based fully automatic assessment of open surgery suturing skills. Int. J. Comput. Assist. Radiol. Surg. **17**(3), 437–448 (2022)
11. Goldbraikh, A., Volk, T., Pugh, C.M., Laufer, S.: Using open surgery simulation kinematic data for tool and gesture recognition. Int. J. Comput. Assisted Radiol. Surg. **17**, 965–979 (2022)
12. Huang, C., et al.: Sample imbalance disease classification model based on association rule feature selection. Pattern Recogn. Lett. **133**, 280–286 (2020)
13. Hutter, F., Hoos, H., Leyton-Brown, K.: An efficient approach for assessing hyperparameter importance. In: International Conference on Machine Learning, pp. 754–762. PMLR (2014)
14. Jacob, M.G., Li, Y.T., Wachs, J.P.: A gesture driven robotic scrub nurse. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics, pp. 2039–2044. IEEE (2011)
15. Jung, J.J., Jüni, P., Lebovic, G., Grantcharov, T.: First-year analysis of the operating room black box study. Ann. Surg. **271**(1), 122–127 (2020)
16. Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks for action segmentation and detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 156–165 (2017)
17. Lea, C., Vidal, R., Hager, G.D.: Learning convolutional action primitives for fine-grained action recognition. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1642–1649. IEEE (2016)
18. Lea, C., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks: a unified approach to action segmentation. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 47–54. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_7
19. Li, S.J., AbuFarha, Y., Liu, Y., Cheng, M.M., Gall, J.: MS-TCN++: multi-stage temporal convolutional network for action segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 1 (2020). https://doi.org/10.1109/TPAMI.2020.3021756
20. Lux, M., Marques, O., Schöffmann, K., Böszörmenyi, L., Lajtai, G.: A novel tool for summarization of arthroscopic videos. Multimed. Tools Appl. **46**(2), 521–544 (2010)

21. Maier-Hein, L., et al.: Surgical data science-from concepts toward clinical translation. Med. Image Anal. **76**, 102306 (2022)
22. Maier-Hein, L., et al.: Surgical data science for next-generation interventions. Nat. Biomed. Eng. **1**(9), 691–696 (2017)
23. Mascagni, P., et al.: A computer vision platform to automatically locate critical events in surgical videos: documenting safety in laparoscopic cholecystectomy. Ann. Surg. **274**(1), e93–e95 (2021)
24. Mascagni, P., et al.: Artificial intelligence for surgical safety: automatic assessment of the critical view of safety in laparoscopic cholecystectomy using deep learning. Ann. Surg. **275**(5), 955–961 (2022)
25. Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3163–3172 (2021)
26. Padoy, N.: Machine and deep learning for workflow recognition during surgery. Minim. Invasive Ther. Allied Technol. **28**(2), 82–90 (2019)
27. Pandey, A., Wang, D.: TCNN: temporal convolutional neural network for real-time speech enhancement in the time domain. In: 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), ICASSP 2019, pp. 6875–6879. IEEE (2019)
28. Peddinti, V., Povey, D., Khudanpur, S.: A time delay neural network architecture for efficient modeling of long temporal contexts. In: Sixteenth Annual Conference of the International Speech Communication Association (2015)
29. Ramesh, S., et al.: Multi-task temporal 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). https://doi.org/10.1007/s11548-021-02388-z
30. Sun, X., Okamoto, J., Masamune, K., Muragaki, Y.: Robotic technology in operating rooms: a review. Curr. Robot. Rep. **2**(3), 333–341 (2021)
31. Tan, M., Le, Q.: EfficientNetV2: smaller models and faster training. In: International Conference on Machine Learning, pp. 10096–10106. PMLR (2021)
32. Twinanda, A.P., Yengera, G., Mutter, D., Marescaux, J., Padoy, N.: RSDNet: learning to predict remaining surgery duration from laparoscopic videos without manual annotations. IEEE Trans. Med. Imaging **38**(4), 1069–1078 (2018)
33. Ullah, A., Ahmad, J., Muhammad, K., Sajjad, M., Baik, S.W.: Action recognition in video sequences using deep bi-directional LSTM with CNN features. IEEE Access **6**, 1155–1166 (2017)
34. Yi, F., Wen, H., Jiang, T.: ASFormer: transformer for action segmentation. In: The British Machine Vision Conference (BMVC) (2021)
35. Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4694–4702 (2015)
36. Zhang, B., Ghanem, A., Simes, A., Choi, H., Yoo, A., Min, A.: Swnet: surgical workflow recognition with deep convolutional network. In: Medical Imaging with Deep Learning, pp. 855–869. PMLR (2021)