Retrieval of surgical phase transitions using reinforcement learning

Abstract. In minimally invasive surgery, surgical workflow segmentation from video analysis is a well studied topic. The conventional approach defines it as a multi-class classification problem, where individual video frames are attributed a surgical phase label. We introduce a novel reinforcement learning formulation for offline phase transition retrieval. Instead of attempting to classify every video frame, we identify the timestamp of each phase transition. By construction, our model does not produce spurious and noisy phase transitions, but contiguous phase blocks. We investigate two different configurations of this model. The first does not require processing all frames in a video (only $< 60\%$ and $< 20\%$ of frames in 2 different applications), while producing results slightly under the state-of-the-art accuracy. The second configuration processes all video frames, and outperforms the state-of-the art at a comparable computational cost. We compare our method against the recent top-performing frame-based approaches TeCNO and Trans-SVNet on the public dataset Cholec80 and also on an in-house dataset of laparoscopic sacrocolpopexy. We perform both a frame-based (accuracy, precision, recall and F1-score) and an event-based (event ratio) evaluation of our algorithms.

Keywords: Surgical workflow segmentation · Machine Learning · Laparoscopic sacrocolpopexy · Reinforcement Learning.

1 Introduction

Surgical workflow analysis is an important component to standardise the timeline of a procedure. This is useful for quantifying surgical skills [7], training progression [9, 15], and can also provide contextual support for further computer analysis both offline for auditing and online for surgeon assistance and automation [14, 5, 16]. In the context of laparoscopy, where the main input is video, the current approaches for automated workflow analysis focus on frame-level multi-label classification. The majority of the state-of-the-art models can be decomposed into two components: feature extractor and feature classifier. The feature extractor normally is a Convolutional Neural Network (CNN) backbone converting images or batches of images (clips) into feature vectors. Most of the features extracted at this stage are spatial features or fine-level temporal features depending on the type of the input. Considering that long-term information in surgical video sequences aids the classification process, the following feature classifier predicts phases based on a temporally ordered sequence of extracted features. Following from natural language processing (NLP) and computer vision techniques,

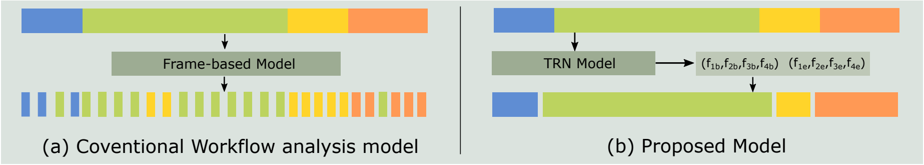

Frame-based Model
(a) Coventional Workflow analysis model
TRN Model
(f_{1b},f_{2b},f_{3b},f_{4b}) (f_{1e},f_{2e},f_{3e},f_{4e})
(b) Proposed Model

Fig. 1: Comparison of network architecture between (a) conventional model and (b) our proposed model with potential error illustration. The conventional model assigns labels for each individual frames and our proposed model predicts frame indices for the starts and end position of phases.

the architecture behind this feature classifier has evolved from Long Short-Term Memory (EndoNet, SVRCNet) [17, 8] , Temporal Convolution Network (TeCNO) [3], to Transformer (OperA, TransSV) [4, 6] in workflow analysis. Although these techniques have improved over the years, the main problem formulation remains unchanged in that phase labels are assigned to individual units of frames or clips.

These conventional models achieve now excellent performance on the popular Cholec80 benchmark [17], namely on frame-based evaluation metrics (accuracy, precision, recall, f1-score). However, small but frequent errors still occur throughout the classification of large videos, causing a high number of erroneous phase transitions, which make it very challenging to pinpoint exactly where one phase ends and another starts. To address this problem, we propose a novel methodology for surgical workflow segmentation. Rather than classifying individual frames in sequential order, we attempt to locate the phase transitions directly. Additionally, we employ reinforcement learning as a solution to this problem, since it has shown good capability in similar retrieval tasks [13, 11]. Our contributions can be summarised as follows:

- We propose a novel formulation for surgical workflow segmentation based on phase transition retrieval. This strictly enforces that surgical phases are continuous temporal intervals, and immune to frame-level noise.
- We propose Transition Retrieval Network (TRN) that actively searches for phase transitions using multi-agent reinforcement learning. We describe a range of TRN configurations that provide different trade-offs between accuracy and amount of video processed.
- We validate our method both on the public benchmark Cholec80 and on an in-house dataset of laparoscopic sacropolpopexy, where we demonstrate a single phase detection application.

2 Methods

We consider the task of segmenting the temporal phases of a surgical procedure from recorded video frames. The main feature of our proposed formulation can be visualised in Fig. 1. While previous work attempts to classify every frame of a video according to a surgical phase label, we attempt to predict the frame index of phase transitions. More specifically, for a surgical procedure with N different phases, our goal is to predict the frame indices where each phase starts $\{f_{1b}, f_{2b}...f_{Nb}\}$, and where each phase ends $\{f_{1e}, f_{2e}...f_{Ne}\}$. If we can assume that surgical phases are continuous intervals, as it is often the case, then our approach enforces this by design. This is unlike previous frame-based approaches where spurious transitions are unavoidable with noisy predictions. To solve this problem we propose the Transition Retrieval Network (TRN), which we described next.

2.1 Transition Retrieval Network (TRN)

Figure 2 shows the architecture of our TRN model. It has three main modules: an averaged ResNet feature extractor, a multi-agent network for transition retrieval, and a Gaussian composition operator to generate the final workflow segmentation result.

Averaged ResNet feature extractor We first train a standard ResNet50 encoder (outputs 2048 dimension vector) with supervised labels, in the same way as frame-based models. For a video clip of length K, features are averaged into a single vector. We use this to temporally down-sample the video through feature extraction. In this work we consider $K = 16$.

DQN Transition Retrieval We first discuss the segmentation of a single phase n. We treat it as a reinforcement learning problem with 2 discrete agents W_b and W_e, each being a Deep Q-Learning Network (DQN). These agents iteratively move a pair of search windows centered at frames f_{nb} and f_{ne}, with length L. The state of the agents s_k is represented by the $2L$ features within the search window, obtained with the averaged ResNet extractor. Based on their state, the agents generate actions $a_{kb} = W_b(s_k)$, and $a_{ke} = W_e(s_k)$, which move the search windows either one clip to the left or to the right within the entire video. During network training, we set a $+1$ reward for actions that move the search window center towards the groundtruth transition, and -1 otherwise. Therefore, we learn direction cues from image features inside the search windows. As our input to DQN is a sequence of feature vectors, a 3-layer LSTM of dimension 2048 is introduced to DQN architecture for encoding the temporal features into action decision process. The LSTMs are followed by 2 fully connected layer of dimension $20L$ and 50 respectively that maps temporal features to the final 2 Q-values of 'Right' and 'Left'. We implemented the standard DQN training framework for our netwrok. [12] At inference time, we let the agents explore the video until they converge to a fixed position (i. e. cycling between left and right actions). Two important characteristics of this solution should be highlighted: 1) we do not need to extract clip features from the entire video, just enough for the agent to reach the desired transition; 2) the agents need to be initialised at a certain position in the video, which we discuss later.

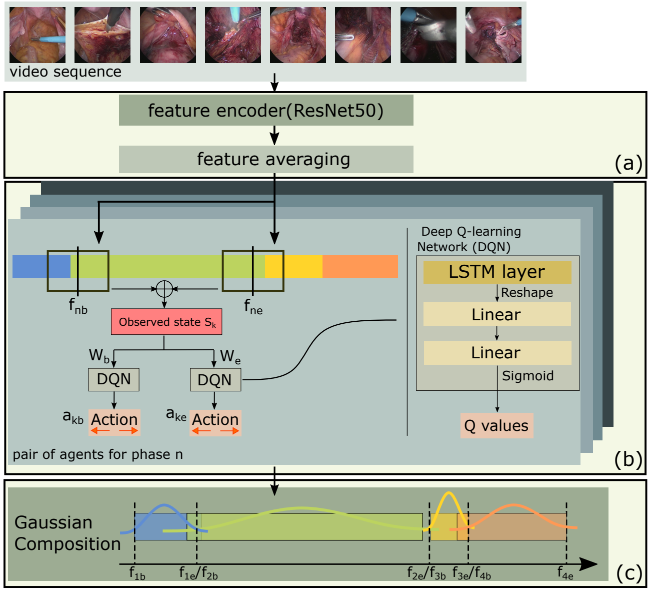


Fig. 2: TRN architecture with (a) averaged ResNet feature extractor, (b) multi-agent network for transition retrieval and (c) Gaussian composition operator

Agent initialization configurations: We propose two different approaches to initialise the agents: fixed initialization (FI) and, ResNet modified initialization (RMI). FI initializes the search windows based on the statistical distribution (frame index average) of each phase transition on the entire training data. With FI, TRN can make predictions without viewing the entire video and save computation time. On the other hand, RMI initialises the search windows based on the averaged-feature ResNet-50 predictions by averaging the indices of all possible transitions to generate an estimation. In this way, we are very likely to have more accurate initialization positions to FI configuration and yield better performance.

Merging different phases with Gaussian composition: So far, we have only explained how our DQN transition retrieval model segments a single phase. To generalise this, we start by running an independently trained DQN transition retrieval model for each phase. If we take the raw estimations of these phase transitions, we inevitably create overlapping phases, or time intervals with no phase allocated, due to errors in estimation. To address this, we perform a Gaussian composition of of the predicted phases. For each predicted pair of transitions f_{nb}, f_{ne}, we draw a Gaussian distribution centred at $\frac{f_{nb}+f_{ne}}{2}$, with standard deviation $\frac{|f_{nb}-f_{ne}|}{4}$. For each video clip, the final multi-class prediction corresponds to the phase with maximum distribution value.

2.2 Training details

The DQN model is trained in a multi-agent mode where W_b, W_e for a single phase are trained together. The input for individual DQNs in each agent shares a public state concatenated from the content of both search windows, allowing the agents to be able to aware information of others. The procedures of training the DQN are showing in pseudo code in algorithm 1. For one episode, videos are trained one by one and the maximum number of steps an agent can explore in a video is 200. For every steps the agents made, movement information (s_k, s_{k+1}, a_k, r_k) are stored in its replay memories, and sampled with a batch size of 128 in computing loss. [12] This loss is optimized with gradient descent algorithm, where α is the learning rate and $\nabla_{W_k}\mathcal{L}_k$ is the gradient of loss in the direction of the network parameters.

Algorithm 1 The procedures of training DQN

```
Initialize parameters of agents $W_b$ and $W_e$ as $W_{0b}$ and $W_{0e}$
Initialize individual replay memories for agents $W_b$ and $W_e$
for $episode \leftarrow 0$ to $episode_{MAX}$ do
    Initialize search window positions (FI or RMI)
    for $video \leftarrow 0$ to $range(videos)$ do
        for $k \leftarrow 0$ to 200 do
            $s_k \leftarrow$ read ResNet features in search window
            $a_{kb} \leftarrow W_{kb}(s_k)$ and $a_{ke} \leftarrow W_{ke}(s_k)$
            $s_{k+1} \leftarrow$ update search window position by $(a_{kb}, a_{ke})$ , read new features
            $r_{kb}, r_{ke} \leftarrow$ compare $s_k$ and $s_{k+1}$ with reward function
            Save $(s_k, s_{k+1}, a_k, r_{kb})$ and $(s_k, s_{k+1}, a_k, r_{ke})$ into agent memory
            Compute loss $(\mathcal{L}_{kb}, \mathcal{L}_{ke})$ from random 128 samples from each memory
            Optimize $W_{kb}$: $W_{k+1b} \leftarrow W_{kb} + \alpha\nabla_{W_{kb}}\mathcal{L}_{kb}$
            Optimize $W_{ke}$: $W_{k+1e} \leftarrow W_{ke} + \alpha\nabla_{W_{ke}}\mathcal{L}_{ke}$
        end for
    end for
end for
```

3 Experiment setup and Dataset Description

The proposed network is implemented in PyTorch using a single Tesla V100-DGXS-32GB GPU of an NVIDIA DGX station. For the ResNet-50 part, PyTorch default ImageNet pretrained parameters are loaded for transfer learning. The videos are subsampled to 2.4 fps, centre cropped, and resized into resolution 224*224 to match the input requirement of ResNet-50. We train both ResNet-50 and DQN with Adam [10] at a learning rate of 3e-4. For ResNet-50, we use a batch size of 100, where phases are sampled with equal probability. For DQN, the batch size is 128.

We tested the performance of TRN model on two datasets. Cholec80 is a publicly available benchamark that contains 80 videos of cholecystectomy surgeries [17] divided into 7 phases. We use 40 videos for training, 20 for validation and 20 for testing. We also provide results on an in-house dataset of laparoscopic sacrocolpopexy [1] containing 38 videos. It contains up to 8 phases (but only 5 in most cases), however, here we consider the simplified binary segmentation of the phases related to suturing a mesh implant (2 contiguous phases), given that suturing time is one of the most important indicators of the learning curve [2] in this procedure. We performed a 2-fold cross-validation with 20 videos for training, 8 for validation, and 10 for testing. For Sacrocolpopexy, we train our averaged ResNet extractor considering all phases, but train a single DQN transition retrieval for a suturing phase. We also do not require to apply Gaussian composition since we're interested in a single phase classification.

Evaluation metrics: We utilise the commonly utilised frame-based metrics for surgical workflow: macro-averaged (per phase) precision and recall, F1-score calculated through this precision and recall, and micro-averaged accuracy. Additionally, we also provide event-based metrics that look at accuracy of phase transitions. An event is defined as block of consecutive and equal phase labels, with a start time and a stop time. We define event ratio as $\frac{E_{gt}}{E_{det}}$ where E_{gt} is the number of ground truth events, and E_{det} is the number of detected events by each method. We define a second ratio based on the Ward metric [18] which allocates events into sub-categories as deletion(D), insertion(I′), merge(M, M′), fragmentation(F, F′), Fragmented and Merged(FM, FM′) and Correct(C) events. Here, we denote the Ward event ratio as $(\frac{C}{E_{gt}})$. For both of these ratios, values closer to 1 indicate better performance. Finally, whenever fixed initialisation (FI) is used, we also provide a coverage rate, indicating the average proportion of the videos that was processed to perform the segmentation. Lower values indicate fewer features need to be extracted and thus lower computation time.

4 Results and Discussion

We first provide an ablation of different configurations of our TRN model in Table 1, for Cholec80. It includes two search window sizes (21 and 41 clips) and two initialisations (FI, RMI). The observations are straightforward. Larger windows induce generally better f1-scores, and RMI outperforms FI. This means that heavier configurations, requiring more computations, lead to better accuracies. Particular choice of a TRN configuration would depend on a trade-off analysis between computational efficiency and frame-level accuracy.

Table 2 shows a comparison between TRN and state-of-the-art frame-based methods on both Cholec80 and Sacrocolpopexy. The utilised baselines are TeCNO [3], Trans-SVNet [6], which we implemented and trained ourselves. Instead of simple ResNet50, we use the same feature averaging process as the TRN for consistency.

Window size	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7	Overall F1-score
TRN21 FI	0.854	0.917	0.513	0.903	0.687	0.549	0.83	0.782
TRN41 FI	0.828	0.943	0.636	0.922	0.558	0.694	0.85	0.808
TRN21 RMI	0.852	0.942	0.619	0.939	0.727	0.747	0.837	0.830
TRN41 RMI	0.828	0.940	0.678	0.945	0.753	0.738	0.861	0.846

Table 1: TRN ablation in the Cholec80 dataset (F1-scores). The values per-phase are computed before Gaussian Composition, while the overall F1-score is for the complete TRN method.

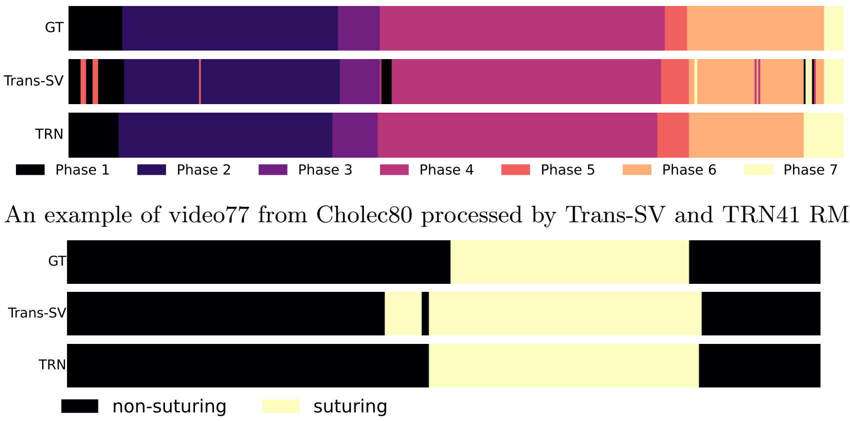
Dataset	Method	Accuracy	Precision	Recall	F1-Score	Event ratio	Ward Event Ratio	Coverage rate(%)
Cholec80	ResNet-50	79.7±7.5	73.5±8.4	78.5±8.9	0.756	0.120	0.375	full
	TeCNO	88.3±6.5	78.6±9.9	76.7±12.5	0.774	0.381	0.691	full
	Trans-SVNet	89.1±5.7	81.7±6.5	79.1±12.6	0.800	0.316	0.566	full
	TRN21 FI	85.3±9.6	78.1±11.1	78.9±13.5	0.782	1	0.934	57.6
	TRN41 FI	87.8±8.1	80.3±9.1	81.7±12.4	0.808	1	0.956	59.1
	TRN41 RMI	90.1±5.7	84.5±5.9	85.1±8.2	0.846	1	0.985	full
Sacrocol- -popexy	ResNet-50	92.5±3.8	94.9±2.8	84.5±8.4	0.892	0.029	0.016	full
	TeCNO	98.1±1.7	97.7±1.9	97.5±3.0	0.976	0.136	0.438	full
	Trans-SVNet	97.8±2.2	96.5±4.5	98.0±3.5	0.971	0.536	0.813	full
	TRN21 FI	89.8±6.2	88.6±11.7	85.3±11.1	0.860	0.971	0.875	14.6
	TRN81 FI	90.7±6.1	88.6±11.5	88.5±11.1	0.875	0.941	0.860	18.3

Table 2: Evaluation metric results summary of ResNet-50, our implementation of TeCNO and Trans-SV, and ablative selected TRN result on Cholec80 and Sacrocolpopexy.

For Cholec80, our full-coverage model (TRN41 RMI) surpasses the best baseline (Trans-SVNet) in all frame-based metrics, while having significantly better even-based metrics (event ratio, Ward event ratio). This can be explained by TRN's immunity to frame-level noisy predictions, which can be visualised on a sample test video in Fig. 3a. Remaining visualisations for all test data are provided in supplementary material.

Still for Cholec80, our partial-coverage models (TRN21/41 FI) have frame-based metrics below the state-of-the-art baselines, however, they have the advantage of performing segmentation by only processing below 60% of the video samples. The trade-off between coverage and accuracy can be observed. Additionally, TRN21/41 FI also have substantially better event-based metrics than frame-based methods due to its formulation.

For sacrocolpopexy, we display a case where our partial-coverage models (TRN21/41 FI) are at their best in terms of computational efficiency. These are very long procedures and we are interested in only the suturing phases, therefore, a huge proportion of the video can be ignored for a full segmentation. Our models slightly under perform all baselines in frame-based metrics, but achieve this result by only looking at under 20% of the videos on average.



(a) An example of video77 from Cholec80 processed by Trans-SV and TRN41 RMI

(b) An example video from Sacrocolpopexy processed by Trans-SV and TRN81 FI

Fig. 3: Color-coded ribbon illustration for two complete surgical videos from (a) Cholec80 and (b) Sacrocolpopexy processed by Trans-SV and TRN models.

5 Conclusion

In this work we propose a new formulation for surgical workflow analysis based on phase transition retrieval (instead of frame-based classification), and a new solution to this problem based on multi-agent reinforcement learning (TRN). This poses a number of advantages when compared to the conventional frame-based methods. Firstly, we avoid any frame-level noise in predictions, strictly enforcing phases to be continuous blocks. This can be useful in practice if, for example, we are interested in time-stamping phase transitions, or in detecting unusual surgical workflows (phases occur in a non-standard order), both of which are challenging to obtain from noisy frame-based classifications. In addition, our models with partial coverage (TRN21/41/81 FI) are able to significantly reduce the number of frames necessary to produce a complete segmentation result.

There are, however, some limitations. First, there may be scenarios where phases occur with an unknown number of repetitions, which would render our formulation unsuitable. Our TRN method is not suitable for real-time application, since it requires navigating the video in arbitrary temporal order. TRN may have scalability issues, since we need to train a different agent for each phase, which may be impractical if a very large number of phases is considered. This could potentially be alleviated by expanding the multi-agent framework to handle multiple phase transitions simultaneously. TRN is also sensitive to agent initialisation, and while we propose 2 working strategies (FI, RMI), they can potentially be further optimised.

References

1. Claerhout, F., Roovers, J.P., Lewi, P., Verguts, J., De Ridder, D., Deprest, J.: Implementation of laparoscopic sacrocolpopexy—a single centre's experience. International urogynecology journal **20**(9), 1119–1125 (2009)
2. Claerhout, F., Verguts, J., Werbrouck, E., Veldman, J., Lewi, P., Deprest, J.: Analysis of the learning process for laparoscopic sacrocolpopexy: identification of challenging steps. International urogynecology journal **25**(9), 1185–1191 (2014)
3. Czempiel, T., Paschali, M., Keicher, M., Simson, W., Feussner, H., Kim, S.T., Navab, N.: TeCNO: Surgical phase recognition with multi-stage temporal convolutional networks. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 343–352. Springer (2020)
4. Czempiel, T., Paschali, M., Ostler, D., Kim, S.T., Busam, B., Navab, N.: Opera: Attention-regularized transformers for surgical phase recognition. arXiv preprint arXiv:2103.03873 (2021)
5. DiPietro, R., Lea, C., Malpani, A., Ahmidi, N., Vedula, S.S., Lee, G.I., Lee, M.R., Hager, G.D.: Recognizing surgical activities with recurrent neural networks. In: International conference on medical image computing and computer-assisted intervention. pp. 551–558. Springer (2016)
6. Gao, X., Jin, Y., Long, Y., Dou, Q., Heng, P.A.: Trans-svnet: Accurate phase recognition from surgical videos via hybrid embedding aggregation transformer. arXiv preprint arXiv:2103.09712 (2021)
7. Goodman, E.D., Patel, K.K., Zhang, Y., Locke, W., Kennedy, C.J., Mehrotra, R., Ren, S., Guan, M., Downing, M., Chen, H.W., et al.: A real-time spatiotemporal ai model analyzes skill in open surgical videos. arXiv preprint arXiv:2112.07219 (2021)
8. Jin, Y., Dou, Q., Chen, H., Yu, L., Qin, J., Fu, C., Heng, P.: SV-RCNet: Workflow recognition from surgical videos using recurrent convolutional network. IEEE Transactions on Medical Imaging **37**(5), 1114–1126 (2018)
9. Kawka, M., Gall, T.M., Fang, C., Liu, R., Jiao, L.R.: Intraoperative video analysis and machine learning models will change the future of surgical training. Intelligent Surgery (2021)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
11. Lu, Y., Li, Y., Velipasalar, S.: Efficient human activity classification from egocentric videos incorporating actor-critic reinforcement learning. In: 2019 IEEE International Conference on Image Processing (ICIP). pp. 564–568 (2019). https://doi.org/10.1109/ICIP.2019.8803823
12. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. nature **518**(7540), 529–533 (2015)
13. Nikpour, B., Armanfard, N.: Joint selection using deep reinforcement learning for skeleton-based activity recognition. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 1056–1061 (2021). https://doi.org/10.1109/SMC52423.2021.9659047
14. Park, J., Park, C.H.: Recognition and prediction of surgical actions based on online robotic tool detection. IEEE Robotics and Automation Letters **6**(2), 2365–2372 (2021). https://doi.org/10.1109/LRA.2021.3060410
15. Rojas-Muñoz, E., Couperus, K., Wachs, J.: Daisi: Database for ai surgical instruction. arXiv preprint arXiv:2004.02809 (2020)

16. Sarikaya, D., Jannin, P.: Towards generalizable surgical activity recognition using spatial temporal graph convolutional networks. arXiv preprint arXiv:2001.03728 (2020)
17. Twinanda, A.P., Shehata, S., Mutter, D., Marescaux, J., De Mathelin, M., Padoy, N.: EndoNet: a deep architecture for recognition tasks on laparoscopic videos. IEEE transactions on medical imaging **36**(1), 86–97 (2016)
18. Ward, J.A., Lukowicz, P., Gellersen, H.W.: Performance metrics for activity recognition. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(1), 1–23 (2011)