GUI-ASFormer: Detecting Keyframes in GUI Videos

Hao-Chun Shih, haochuns@umich.edu Lei Lei, leilea@umich.edu Pei-Chi Huang, peichi@umich.edu

Te-Hsiu Tsai, tedtsai@umich.edu Yu-Hsin Huang, yuhsinhu@umich.edu

***Abstract*—This work focuses on automating action segmentation in GUI (Graphical User Interface) videos, where actions involve tasks like clicking, scrolling, and typing, rather than physical movements. Traditional action segmentation relies heavily on human manual labeling, which is time-consuming and difficult to scale. Although several automated approaches exist for keyframe extraction, they typically underperform with GUI videos due to the unique characteristics of GUI videos: 1) GUI items are often small and difficult to detect, 2) actions are highly localized, and 3) action durations vary significantly. To tackle these challenges, we adapt the ASFormer architecture with several GUI-specific enhancements. We introduce a two-stream encoder that extracts both global and fine-grained visual features from low- and high-resolution frames, a frame differencing block to highlight subtle temporal changes, and a multiscale temporal convolution module to capture both short- and long-term dependencies. Extensive experiments demonstrated the effectiveness of our model. Code is available at: https://github.com/Peggy1210/eecs545-final-project**

***Keywords*—Action Segmentation, GUI Video Understanding, Vision Transformer**

I. Introduction

Action segmentation is a crucial task in video analysis, enabling the identification of meaningful events within a continuous video stream. While approaches primarily focus on human activities in real-world environments, an emerging area of interest is action segmentation in GUI (Graphical User Interface) videos, where actions involve interactions with software interfaces rather than physical movements. Accurate segmentation of these interactions enhances content clarity, improves user comprehension, and supports a variety of applications, including GUI tutorial summarization [1], automated task guidance, and training GUI-based robotic agents [2], [3], thereby enhancing automation in various domains.

In GUI videos, interactions typically include actions such as clicking, scrolling, dragging, keyboard input, etc.; keyframes refer to the most representative frames within an action sequence. Traditionally, this process has relied on human labeling, which is labor-intensive and difficult to scale, and gesture-matching techniques [4] often fail due to the increasingly high dimensionality and complexity of modern GUI videos.

With the rise of deep learning, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) were adopted for video understanding, but fall short in modeling long-range dependencies. Transformer-based architectures have since emerged as the state-of-the-art for video segmentation, providing improved handling of temporal context, boundary accuracy, and computational scalability. However, based on our background research, few existing models are specifically designed for GUI video segmenting. In particular, state-of-the-art models often underperform in this domain due to the unique characteristics of GUI videos. We identify three challenges in segmenting GUI videos:

1. **Dispersed and fine-grained visual elements:** Unlike real-world videos, where a central object dominates the frame, GUI elements are often small in size and scattered across the screen. Also, some elements, such as text, contain rich details. Meaningful representations are required to preserve this information.
2. **Subtle visual changes:** Many interactions, such as clicking a button, selecting a menu item, or entering text, result in only subtle visual differences between consecutive frames. These changes lack strong visual cues and are hard to detect in pixel-level comparisons.
3. **Varied action durations:** GUI actions vary significantly in temporal length. For example, a click may last less than a second, whereas scrolling can span several seconds. This poses difficulties for models not designed to handle multi-scale temporal dynamics.

To address these challenges, we propose a novel action segmentation framework for GUI videos. Particularly, we employ vision transformers (ViTs) to extract video representation to capture spatial detail and preserve fine-grained interface elements. For the second concern, we combine both low- and high-resolution features to add information to the prediction process, supplemented by a frame differencing module to highlight subtle frame-wise changes. Lastly, we introduce a multiscale convolution mechanism that captures long- and short-term dependencies. Empirical results demonstrate that each architectural component improves segmentation performance, and the overall system outperforms our baseline model. Our contribution is twofold:

- We demonstrate the effectiveness of transformer-based representations in capturing fine-grained spatial details critical to GUI video segmentation.
- We enhance the segmentation architecture to better detect subtle and temporally diverse actions specific to GUI interaction patterns.

II. Methodology

In this paper, we propose GUI-ASFormer, focusing on two key aspects: 1) Enhanced spatial understanding of GUI frames, and 2) Improved temporal modeling of user interactions. The overall architecture is shown in Figure 1.

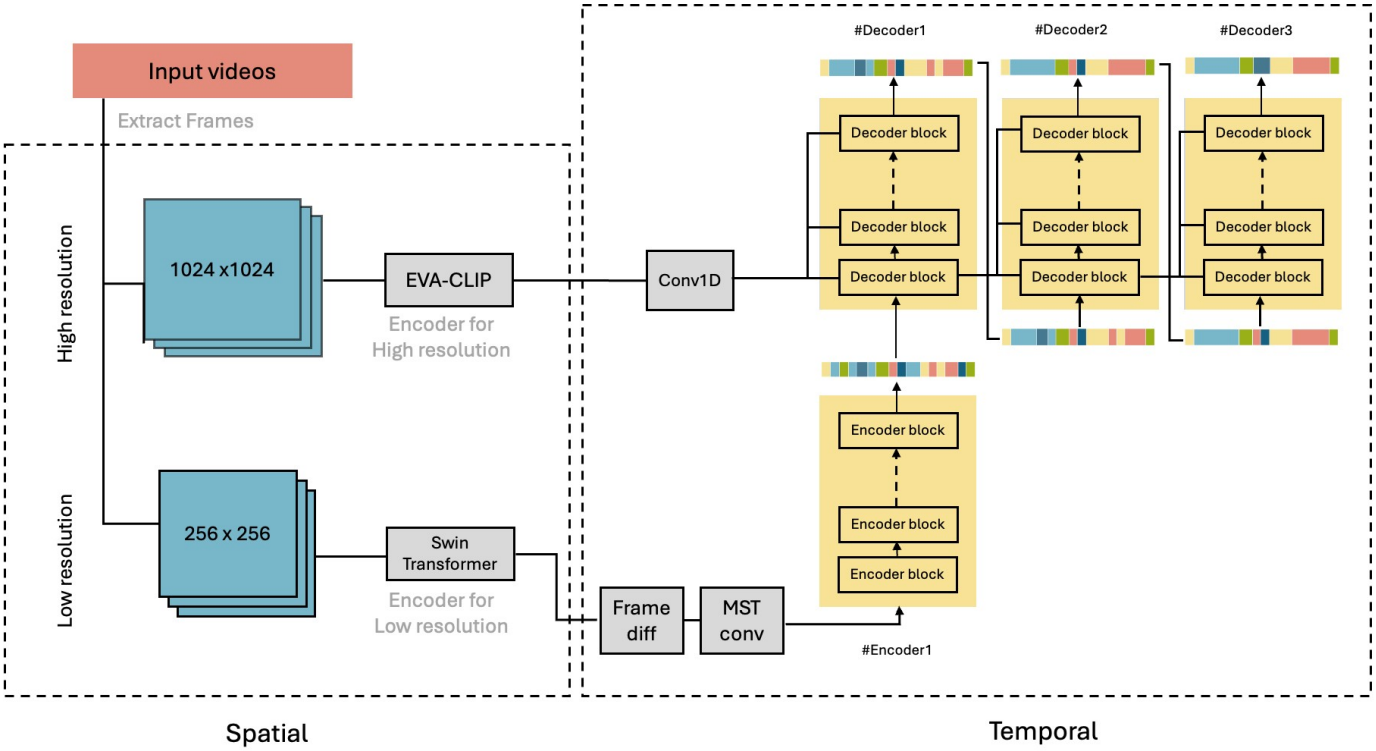

Fig. 1. The architecture of GUI-ASFormer.

A. *Video Feature Extraction*

We extract frame-wise features and stack them into a unified video representation. To effectively capture both global and fine-grained GUI details, we introduce two vision transformer models: Swin Transformer [5] for low-resolution features and EVA2-CLIP-L [6] for high-resolution features.

Low-Resolution Features. We use pretrained Swin Transformer ("swinv2_base_window8_256") with an additional spatial attention layer to extract low-resolution (256×256 pixels) frame images. Swin Transformer builds upon the Vision Transformer (ViT) architecture, introducing hierarchical feature maps and shifted window attention to enhance computational efficiency and multi-scale feature learning.

In the context of action segmentation, GUI actions often exhibit subtle temporal variations and occur at multiple spatial scales. The hierarchical design of Swin Transformer enables the model to capture both local and global information, while the shifted window mechanism increases the receptive field across layers without incurring high computational costs. This architecture is particularly effective for modeling the subtle, region-specific changes common in GUI interactions.

The output of Swin Transformer for each frame is an 8×8 spatial grid of feature embeddings. To aggregate these regional features into a single frame-level embedding, we apply a spatial attention mechanism followed by a CNN fusion layer. The spatial attention layer assigns importance weights to different regions, enabling the model to focus on the most informative parts of the GUI interface. The CNN-based fusion then aggregates these weighted features into a global embedding at dimension D_{low}.

This process enhances the model's ability to capture which parts of the GUI are most critical for downstream action segmentation, especially when fine details (such as buttons or text fields) are key to identifying user actions.

High-Resolution Features. Although Swin Transformer provides strong feature representations, it has fixed input dimensions in pre-trained settings and may not fully preserve fine details in GUI frames when resized to lower resolutions. Unlike conventional image recognition, low-resolution images barely meet the demands of GUI videos, as small screen elements can become too vague to recognize after resizing.

To address this, we adopt EVA2-CLIP-L [6], a smaller and efficient vision transformer model (149M parameters), to extract high-resolution features. Input frames are resized to 1024×1024 pixels and passed through EVA2-CLIP-L to obtain detailed frame-wise embeddings. To align with the dimensionality required by subsequent modules, the high-resolution output is projected through a fixed-weight linear layer, producing frame-level features with dimension D_{high}.

The embeddings from all frames in a video are stacked along the temporal axis. As a result, each video is represented by two sequences: $X_{low} \in \mathbb{R}^{D_{low} \times T}$ for low-resolution features and $X_{high} \in \mathbb{R}^{D_{high} \times T}$ for high-resolution features, where T denotes the number of frames in the video.

B. Action Segmentation Model

The goal of this stage is to assign an action label to each frame based on the extracted visual features. We adopt ASFormer as the backbone for temporal modeling, due to its strong performance and architecture specifically designed for action segmentation tasks.

ASFormer introduces three key innovations to address the challenges of limited training data and long video sequences: (1) temporal convolutions within each encoder block to enforce local connectivity and provide inductive bias; (2) a hierarchical attention pattern that progressively expands receptive fields to capture both short- and long-range dependencies; and (3) a multi-stage decoder with cross-attention to iteratively refine predictions. These design choices enable ASFormer to achieve state-of-the-art performance on public benchmarks.

We use low-resolution embeddings extracted from Swin Transformer as the primary input to ASFormer. Before entering the encoder, these embeddings are first processed through a frame differencing block to highlight subtle temporal changes. Specifically, the difference between consecutive frames is computed as:

$$f'_t = f_t - f_{t-1}$$

, where f_t denotes the frame feature at time t. For consistency, we pad a zero vector at the first frame ($t = 0$) to maintain the original sequence length.

A multiscale temporal convolution block (MST conv) follows immediately after to extract temporal patterns at multiple scales. This module consists of $n_{branches}$ parallel 1D convolution branches, each with different kernel sizes (e.g., 3, 5, 7) and dilation rates (e.g., 1, 2, 4). The outputs from all branches are concatenated along the channel dimension, forming a feature of size $(n_{branches} \cdot D_{low}) \times T$. A final 1×1 1D convolution is applied to project the concatenated features back to the original dimension $D_{low} \times T$. This design allows the model to jointly capture local, mid-range, and long-range temporal patterns while maintaining computational efficiency.

The encoder generates initial frame-wise action predictions. Each encoder block contains a dilated temporal convolution feed-forward layer and a single-head self-attention layer, each followed by a residual connection, instance normalization, and ReLU activation (Figure 2). The feed-forward layer uses dilated convolutions instead of point-wise fully connected layers to introduce local inductive bias, which is beneficial for modeling the temporal continuity of actions across frames. To efficiently handle long video sequences, the encoder applies a hierarchical attention pattern, where each self-attention layer is restricted to a local window of size $w = 2^i$ at encoder layer i. The dilation rate of the temporal convolution layer is also doubled correspondingly. This reduces the memory complexity from $O(J \times T^2)$ in a standard Transformer to approximately $(w - \varepsilon) \times 2^J \times T$, where J is the number of encoder layers and ε is a small constant.

In later decoding stages, high-resolution embeddings extracted from EVA2-CLIP-L are projected through a CNN layer and used to refine the low-resolution predictions through cross-attention. Each decoder contains a fully connected layer followed by a series of decoder blocks (Figure 2). Similar to the encoder, the decoder structure consists of a temporal convolution and a hierarchical cross-attention layer with progressively expanding local windows. In cross-attention, the query Q and key K are derived from the concatenation of the encoder output and the previous decoder output, while the value V is derived solely from the previous decoder output. This design allows external high-resolution information to guide attention without disturbing the internal prediction space, ensuring stable refinement [7].

The multi-stage decoder structure, where the output of each decoder stage is passed to the next, further improves the refinement as shown in Figure 1. The cross-attention mechanism allows for bringing in external information to guide the refinement process. A weighted residual connection is used between the feed-forward and cross-attention outputs:

$$out = feed\_forward(x)$$
$$out = \alpha \times cross\_attention(out) + out$$

, where α controls the influence of cross-attention. We initialize $\alpha = 1$ for the first decoder and exponentially decay it across subsequent stages, progressively reducing dependence on external features to mitigate error accumulation.

In our approach, we combine two streams of embeddings rather than relying solely on high-resolution features. High-resolution models are typically heavier, making it computationally impractical to generate embeddings directly from high-resolution frames. To address this, we introduce EVA2-CLIP-L, a relatively lightweight encoder optimized for efficient coarse representation extraction without excessive complexity. However, these coarse embeddings alone are not sufficient for accurate video segmentation, as they lack the detailed temporal structure needed to precisely localize actions. Therefore, we use low-resolution embeddings as the primary input and incorporate high-resolution embeddings through cross-attention refinement. The two streams offer complementary information: low-resolution features capture robust global structure, while high-resolution features refine fine-grained local details.

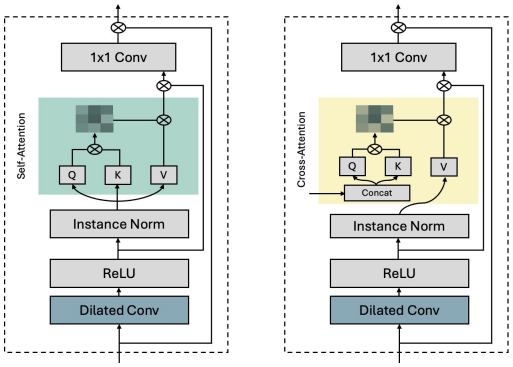

Fig. 2. Encoder (left) and decoder (right) architecture.

C. Training Objectives

The loss function is a combination of classification loss and a smoothing loss. The classification loss is a cross-entropy loss computed over all frames:

$$\mathcal{L}_{cls} = \frac{1}{T}\sum_{t}(-log\ y_{t,c})$$

, where $y_{t,c}$ denotes the predicted probability for the ground-truth label c at time step t. The smoothing loss is defined as the mean squared error (MSE) over the log-probabilities to reduce the over-segmentation errors:

$$\mathcal{L}_{T-MSE} = \frac{1}{TC}\sum_{t,c}(log\ y_{t,c} - log\ y_{t-1,c})^2$$

The loss function for a single stage $\mathcal{L}_s$ is a combination of the classification and smoothing losses, weighted by a hyperparameter λ. To train the complete model, we minimize the sum of losses over all stages:

$$\mathcal{L} = \sum_{s}\mathcal{L}_s = \sum_{s}\mathcal{L}_{cls} + \lambda\mathcal{L}_{T-MSE}$$

III. Related Work

A. Video Representation

Traditional video segmentation relied on handcrafted features like optical flow and SIFT (Scale-Invariant Feature Transform) [8]. While effective for capturing local motion and key points, these approaches struggled to handle complex scene variations and dynamic changes. With increasing video complexity, deep learning approaches such as Recurrent Neural Networks (RNNs) (e.g., LSTMs and GRUs), and 3D CNNs (e.g., I3D [9]) were adopted to jointly model spatial and temporal information. However, these models usually faced efficiency and scalability challenges.

Recently, transformer-based architectures, particularly Vision Transformers (ViT), have been introduced for video feature extraction with enhanced interpretation of frame-wise information. ViT treats the image as a sequence of patches, using self-attention to capture long-range dependencies and global context, making it effective for extracting rich visual representations. However, ViT's quadratic complexity with respect to input size makes it computationally expensive, especially for high-resolution video. Swin Transformer [5] addresses this issue by introducing hierarchical embeddings and a shifted window attention mechanism, reducing computational overhead while preserving spatial relationships. This is particularly advantageous for GUI video segmentation, where screen changes are often subtle and spatial variations are small.

While Swin Transformer improves efficiency, it requires resizing input images to relatively low resolutions, typically to 192×192 or 256×256, which can result in the loss of critical fine details. This is especially problematic in GUI videos where subtle interface changes are important for accurate segmentation. To address this, models like BLIP-2 [9] and ViT-H/14 [10] support high-resolution inputs through larger patch sizes and global attention. However, these approaches usually come with significant computational costs and memory usage. To strike a balance between performance and efficiency, EVA-CLIP [6] offers a better solution, supporting flexible input sizes while preserving fine-grained visual details. As a more lightweight model, it is well-suited for GUI video segmentation, where both spatial precision and efficiency are critical. This flexibility allows for higher resolution without the heavy computational costs seen in other models.

B. Action Segmentation Model

Early models used motion cues (e.g., frame differencing, optical flow [4], [11]), probabilistic frameworks (e.g., CRFs [12]–[14], HMMs [15], [16]), or frame-wise prediction using RNN-based models [17]–[21]. Although effective, these methods faced challenges in capturing long-range dependencies and achieving efficient sequence modeling.

Temporal Convolutional Networks (TCNs) [22] addressed some of these limitations by enabling parallel temporal modeling through convolutional layers, providing a foundation for modern action segmentation architectures. More recently, transformer-based models have been introduced, leveraging self-attention mechanisms originally developed for natural language processing to capture dependencies across frames more effectively. Unlike traditional frame-wise classifiers, transformers model long video sequences with improved efficiency and global context understanding.

ASFormer [23] extends the concept of multi-stage TCNs [7] by integrating them with transformer architectures, reducing temporal complexity while achieving accurate frame-level segmentation. Building upon ASFormer, several recent models have further advanced the field. FACT [24] introduces a cross-attention mechanism to facilitate bidirectional information exchange between frames and action representations. UVAST [25] incorporates self-supervised learning, making it applicable to both supervised and unsupervised segmentation tasks. BAFormer [26] focuses on explicitly modeling action boundaries, improving temporal precision, and mitigating over-segmentation and under-segmentation errors. These models represent the current state of the art in action segmentation, addressing critical challenges related to computational efficiency, accuracy, and boundary detection.

C. GUI Video Segmentation

Based on our research, a few segmentation models have been specifically fine-tuned for GUI videos. Video2Action [4] proposed a segmentation framework that identifies and localizes actions by focusing on frame-to-frame changes and applying topological methods for precise action localization. It effectively captures subtle visual transitions, which is crucial for GUI scenarios where user interactions often involve minimal changes. Its topological approach better captures spatial-temporal relations than general-purpose models, which often miss subtle GUI transitions. However, its range of recognizable actions is limited to only tap, scroll, and backward gestures, restricting its applicability to more complex GUI interactions.

D. Our Approaches

Our approach utilized the well-established action segmentation model ASFormer on GUI videos. The architecture serves as a strong backbone model and thus can be easily adapted. Specifically, we added two stream transformer-based encoders, combining Swin Transformer and EVA2-CLIP-L to capture both global and fine-grained visual features. Additionally, we enhance the model with a frame differencing block to highlight subtle temporal changes and a multiscale temporal convolution block to better capture short- and long-term dependencies. This GUI-specific adaptation differentiates our approach from general-purpose video segmentation techniques and enables more precise automation for UI analysis and testing.

IV. Experiment Results

A. Dataset

The GUI-world dataset comprises a diverse collection of GUI videos, including content from iOS, Android, XR, websites, and other platforms. In this study, we focus on the website subset of GUI-world. Compared to mobile interfaces such as iOS and Android, website pages typically feature more detectable visual elements, making them more suitable for our segmentation objectives.

Following preprocessing, we select videos with durations between 5 and 50 seconds, resulting in 2,031 videos. We define actions as user interactions involving either mouse or keyboard inputs. A total of seven types of interactions are identified, as detailed in Table I.

TABLE I
Action Types in GUI-World Website Videos

Mouse	Keyboard
scroll	input
hover	delete
drag	enter
click	

B. Measurements

We evaluate model performance using frame-wise accuracy, segmental F1 score at overlapping thresholds of 10%, 25%, and 50% (denoted as F1@{10, 25, 50}), and segmental edit distance. Frame-wise accuracy measures the percentage of correctly labeled frames. The segmental F1 score uses an intersection-over-union (IoU) threshold to determine overlap between predicted and ground-truth segments. Segmental edit distance quantifies the minimum number of edit operations (insertions, deletions, substitutions) needed to align predicted and ground-truth segmentations.

Although frame-wise accuracy is widely used for action segmentation, it is biased toward longer actions and underpenalizes over-segmentation errors. Therefore, following prior work [22], we primarily adopt the segmental F1 score as the main metric for evaluating segmentation quality.

C. Experiment Settings

We extract frame images from videos at 30 frames per second (FPS) and resize them to 256×256 pixels and 1024×1024 to match the input dimensions required by the Swin Transformer and EVA2-CLIP-L models, respectively. The number of channels is configured to 2048 ($D_{low} = D_{high} = 2048$).

To facilitate training and evaluation, we partition the dataset into five non-overlapping subsets, each containing approximately 400 videos. Each subset is split into 80% training and 20% testing videos. In this section, we use split 1 for testing. Although we do not perform full cross-validation, this splitting strategy provides a consistent and manageable structure for experimentation, as shown in Section V. We train the model for 120 epochs using the Adam optimizer, with an initial learning rate of 0.0005.

D. Effect of Number of Blocks

The number of blocks J in the encoder and decoder is an important hyperparameter. Increasing J expands the receptive field and improves the model's ability to capture long-range temporal dependencies, but also results in higher memory consumption. As shown in Table II, performance improves steadily up to $J = 10$. Beyond this point, all metrics decline noticeably, suggesting that deeper architectures may lead to overfitting or optimization instability. Additionally, GPU memory usage increases significantly with larger J. To balance segmentation performance and computational cost, we adopt $J = 10$ for all subsequent experiments.

TABLE II
Comparison of Different Block Number J

J	Params (M)	F1@{10,20,50}			Edit	Acc
1	75.878	11.71	6.94	2.70	13.92	46.45
4	76.176	55.08	47.67	30.39	51.78	57.29
6	76.375	64.45	58.22	38.88	61.16	61.91
8	76.574	65.51	60.33	40.51	**61.96**	**61.86**
10	76.773	**66.36**	**61.14**	**41.36**	61.81	61.83
12	76.972	62.12	56.06	38.09	55.91	61.46

E. Effect of Number of Stages

We evaluate the effect of using a multi-stage architecture, as shown in Table III. The table compares the prediction performance of a single-encoder model to multi-stage models with varying numbers of decoder stages. Although all stages achieve comparable frame-wise accuracy, the quality of the predictions differs significantly. From the low edit distance and F1 scores, it is clear that the encoder-only predictions produce substantial over-segmentation errors. Introducing a multi-stage architecture significantly reduces these errors and improves the F1 scores. This effect is clearly visible with three or four decoder stages, which gives a huge boost to the performance. Adding a fifth stage provides only marginal improvements, likely due to saturation.

The refinement effect of the multi-stage architecture is also illustrated in the qualitative results shown in Figure 3, where each additional stage leads to progressively more accurate predictions. However, after the third stage, improvements become negligible. Based on these observations, we adopt a three stage decoder architecture for all experiments.

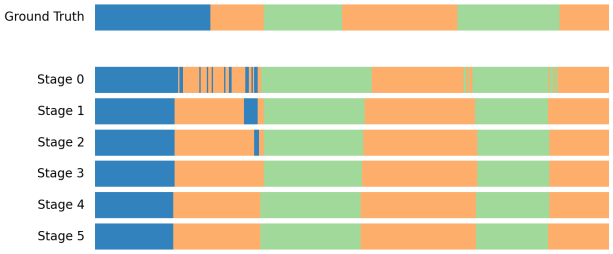

Fig. 3. **Segmentation results at each stage of a five-decoder architecture.** Each bar represents a different action segment within a video. At Stage 0, the model exhibits substantial over-segmentation, producing many fragmented segments. This is progressively corrected in Stage 1 and Stage 2. By Stage 3, the predicted segments closely align with the ground truth. Stages 4 and 5 provide minimal additional improvement, indicating that further refinement beyond Stage 3 yields diminishing returns.

TABLE III
COMPARISON OF PREDICTION WITH DIFFERENT NUMBER OF DECODERS

	F1@{10,20,50}			Edit	Acc
Encoder	16.10	14.28	8.75	15.57	58.28
Encoder + 1 Decoder	56.49	49.95	33.70	53.89	61.05
Encoder + 2 Decoder	61.49	55.58	35.45	56.58	61.17
Encoder + 3 Decoder	62.64	57.05	36.91	58.78	61.16
Encoder + 4 Decoder	**64.44**	59.14	38.20	60.79	61.22
Encoder + 5 Decoder	64.27	**59.40**	**38.28**	**60.92**	**61.23**

F. Ablation Study

We perform an ablation study to analyze the contribution of each component to overall performance, as shown in Table IV. Each component incrementally improves segmentation quality across multiple metrics. First, we replace the I3D backbone with Swin Transformer for feature extraction. While I3D yields relatively higher frame-wise accuracy, it is less meaningful for GUI videos, as I3D captures actions across every 8 frames and may overestimate for short actions. In contrast, Swin Transformer leads to more precise predictions, as reflected in the improved F1 scores and edit distance.

Adding a high-resolution branch significantly improves segmentation performance, especially for F1@25 and F1@50, indicating that fine-grained actions, which may be overlooked in coarse predictions, are better captured with high-resolution features. Incorporating the frame differencing block gives a huge boost in overall performance. Finally, introducing a multi-scale temporal convolution further enhances short-range temporal segmentation quality as indicated by improvements in F1@10 and F1@25. Qualitative results illustrating these improvements are provided in Figure 4.

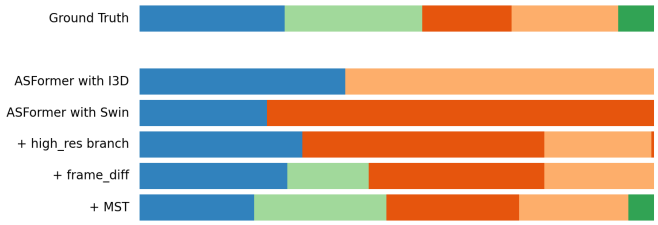

Fig. 4. **Ablative segmentation results.** The baseline model using I3D overlooks short actions due to its coarse temporal resolution. Replacing I3D with Swin Transformer produces more plausible predictions, but some fine-grained actions are still missed. Adding a high-resolution branch provides finer spatial detail, enabling the model to capture additional actions. Incorporating a frame differencing block allows the model to detect subtle transitions, further improving segmentation granularity. Finally, the multi-scale temporal convolution (MST) effectively captures short actions that were previously neglected or overshadowed by longer segments. The resulting prediction closely aligns with the ground truth.

TABLE IV
COMPARISON OF ABLATION STUDY RESULTS

	Params (M)	F1@{10,20,50}			Edit	Acc
ASFormer with I3D	1.132	52.02	45.69	29.88	45.54	53.73
ASFormer with Swin	1.132	57.31	50.29	30.41	54.53	54.46
+ *high-res branch*	1.267	58.23	53.75	34.70	57.71	58.63
+ *frame-diff*	1.267	64.57	56.95	41.02	61.08	60.75
+ *MST*	76.773	66.36	61.14	41.36	61.81	61.83

We compare the proposed GUI-ASFormer with several ASFormer variants in Table V. All models were trained using Swin-based features to ensure consistency. The result shows that our approach achieves significant enhancement across all metrics. Especially, we improve the segmentation quality as indicated by the enhancement in F1 scores. This underscores the contribution of the high-resolution branch, frame differencing module, and multi-scale temporal convolution.

Notably, MSTCN, a convolution-only architecture, performs comparably to Transformer-based models. This may be due to the dataset's limited set of recognizable actions and shorter video lengths, reducing the advantage of Transformer architectures. However, as noted in the original ASFormer paper, Transformers generally outperform convolutional models, particularly on longer videos, suggesting that further benefits may emerge in future studies on more complex GUI recordings.

TABLE V
COMPARISON WITH THE SOTA ON GUI-WORLD WEBSITE VIDEOS

	Params (M)	F1@{10,20,50}			Edit	Acc
MSTCN with Swin	0.798	56.85	47.87	29.92	55.34	53.36
ASFormer with Swin	1.132	57.31	50.29	30.41	54.53	54.46
FACT with Swin	10.418	57.32	49.93	34.13	49.89	56.40
UVAST with Swin	1.102	58.25	50.50	30.50	53.63	49.23
GUI-ASFormer	76.773	**66.36**	**61.14**	**41.36**	**61.81**	**61.83**

V. Discussion

A. Performance on Different Splits

Since we did not perform cross-validation due to computational limitations, we instead evaluated the generalizability of our model by providing an overall test result for each split, as shown in Table VI. Each split achieved consistent results, with slight differences observed across certain metrics and datasets.

In the original ASFormer settings, the model is known to perform effectively on small datasets, which explains the high performance we achieved even when training on limited subsets. While our model introduces additional parameters that could increase the risk of overfitting, the evaluation results suggest that such effects are minimal. The low variance across splits indicates that our model maintains robustness and generalizability despite being trained on limited data.

TABLE VI
Performance in Each Split

	F1@{10,20,50}			Edit	Acc
Split 1	66.36	61.14	41.36	61.81	61.83
Split 2	66.35	60.37	40.25	62.01	62.24
Split 3	66.67	60.86	41.29	62.64	65.12
Split 4	67.45	60.60	41.33	66.62	63.14
Split 5	66.88	60.38	40.04	61.69	61.99
Overall	**66.74**	**60.47**	**40.85**	**62.95**	**62.87**

B. Effect of Video Lengths

Our primary experiments focus on videos ranging from 5 to 50 seconds. To further analyze the effect of video length, we separately evaluated the model's performance on short videos (5–10 seconds) and long videos (30–50 seconds). We sampled approximately 100 training videos and 30 testing videos for each group. The results show that models trained on longer videos achieve substantially higher performance across all metrics (Table VII). Possible reasons include: (1) longer videos introduce greater variability, featuring more distinct actions and transitions, and (2) models trained on short videos are more prone to overfitting due to limited temporal diversity.

However, training on long videos is computationally much more expensive, requiring more than three times the training time compared to short videos. Given computational constraints and the desire to cover both short and long interactions typical in GUI videos, our main experiments include videos spanning from 5 to 50 seconds.

TABLE VII
Comparison of Training on Long/Short Videos

	F1@{10,20,50}			Edit	Acc
Short Videos	43.90	36.93	19.51	47.21	40.36
Long Videos	65.52	58.19	40.95	62.52	72.84

C. Alternative Designs and Limitations

Early fusion is a widely adopted technique in multimodal learning, where features from different modalities are concatenated at an early stage for joint processing. We implemented this using two separate encoders for from high- and low-resolution streams, with their outputs concatenated to generate initial predictions before being refined through multistage decoding. The resulting average F1 scores are [53.87, 46.47, 32.92] and 60.11% accuracy, all of which are notably lower than the performance achieved by the cross-attention-based fusion method. These results suggest that early fusion, in most cases, struggles to effectively integrate multimodal features and cannot dynamically focus on salient information critical for the task.

We also replaced the multiscale temporal convolution block with a multiscale temporal attention module. However, the design added ˜315M parameters, resulting in increased computational overhead. Given the relatively small size of the dataset and the short length of many videos, this model exhibited signs of overfitting, ultimately underperforming compared to the convolution-based approach.

D. Video Language Model Performance

Recent work has introduced strong vision-language models (VLMs), such as Video-LLaMA [27], which may identify key frames of GUI videos with appropriate prompting. Additionally, GUI grounding agents have also shown potential for detecting action transitions via element-wise differences. For example, SmolVLM [28] is a compact VLM specifically designed for multimodal tasks and supports multi-frame input. We conducted a zero-shot evaluation using SmolVLM on a limited set of videos. However, the model demonstrated poor performance, achieving only 35%–40% accuracy. Furthermore, SmolVLM is not well-suited for handling very long input sequences (e.g., around 300 frames simultaneously), limiting its practicality for our task. While VLMs exhibit strong generalization across diverse multimodal tasks, they are inefficient when applied to domain-specific problems like GUI action segmentation and lack the ability to model fine-grained temporal dynamics effectively.

E. Future Work

Building upon the findings of this work, we identify several directions for future research:

GUI-Based Embedding. GUI videos often contain subtle and context-dependent elements, including structured layouts and embedded text. While our current approach focuses on frame-level image features, these representations may suffer from compression and information loss. Future work can explore advanced embedding methods such as Screen2Vec [5], Pix2Code [6], or GUIBERT, which utilize both visual input and structural metadata (e.g., raw HTML or layout trees). These methods can provide more detailed and semantically rich representations of GUI frames, potentially enhancing the model's ability to capture fine-grained action boundaries in GUI video segmentation.

End-to-End Segmentation Model. Several prior approaches aim for unified video segmentation pipelines that directly map video frames to action predictions. Due to computational constraints, our current design separates processing into lightweight spatial and temporal components. In the future, we plan to unify these components into a single, end-to-end model that can jointly learn spatiotemporal features. This integration may further improve accuracy and generalization across diverse GUI video datasets.

VI. Conclusion

In this work, we address the challenges of action segmentation in GUI videos, where subtle interactions and fine-grained interface elements make traditional video segmentation approaches less effective. We adapt the ASFormer framework with GUI-specific enhancements, including a two-stream encoder for low- and high-resolution feature extraction, a frame differencing module to highlight subtle temporal changes, and a multiscale temporal convolution block to capture both short- and long-term dependencies. Our results demonstrate that each component contributes to the overall performance, offering a step forward in automating GUI understanding.

VII. Contributions

All authors contributed equally to this project. Specifically,:

- **Lei Lei** and **Yu-Hsin Huang** focused on feature extraction from GUI videos.
- **Hao-Chun Shih**, **Pei-Chi Huang**, and **Te-Hsiu Tsai** worked on implementing the models.
- All members actively participated in brainstorming, conducting paper research, and writing the report.

VIII. Acknowledgement

This project is completed as part of EECS545: Machine Learning at the University of Michigan. We would like to thank Professor Honglak Lee, the course staff for their support, and Yeda Song for helping define the research direction..

References

[1] T. Tuna, M. Joshi, V. Varghese, R. Deshpande, J. Subhlok, and R. Verma, "Topic based segmentation of classroom videos," in *2015 IEEE Frontiers in Education Conference (FIE)*, 2015, pp. 1–9.

[2] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su, "Mind2web: Towards a generalist agent for the web," 2023. [Online]. Available: https://arxiv.org/abs/2306.06070

[3] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried, U. Alon, and G. Neubig, "Webarena: A realistic web environment for building autonomous agents," 2024. [Online]. Available: https://arxiv.org/abs/2307.13854

[4] S. Feng, C. Chen, and Z. Xing, "Video2action: Reducing human interactions in action annotation of app tutorial videos," 2023. [Online]. Available: https://arxiv.org/abs/2308.03252

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9992–10 002.

[6] Q. Sun, Y. Fang, L. Wu, X. Wang, and Y. Cao, "Eva-clip: Improved training techniques for clip at scale," 2023. [Online]. Available: https://arxiv.org/abs/2303.15389

[7] Y. A. Farha and J. Gall, "Ms-tcn: Multi-stage temporal convolutional network for action segmentation," 2019. [Online]. Available: https://arxiv.org/abs/1903.01945

[8] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157 vol.2.

[9] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," 2023. [Online]. Available: https://arxiv.org/abs/2301.12597

[10] B. N. Patro and V. S. Agneeswaran, "Efficiency 360: Efficient vision transformers," 2023. [Online]. Available: https://arxiv.org/abs/2302.08374

[11] S. Sowmyayani, V. Vivek, A. P, and T. S, "Frame differencing based temporal feature extraction in human action recognition," *Journal of Computational Analysis and Applications (JoCAAA)*, vol. 33, no. 05, p. 549–557, Sep. 2024. [Online]. Available: https://eudoxuspress.com/index.php/pub/article/view/569

[12] H. Pirsiavash and D. Ramanan, "Parsing videos of actions with segmental grammars," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 612–619.

[13] Q. Shi, L. Wang, L. Cheng, and A. Smola, "Discriminative human action segmentation and recognition using semi-markov model," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[14] L. Tao, L. Zappella, G. D. Hager, and R. Vidal, "Surgical gesture segmentation and recognition," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 339–346.

[15] K. Tang, L. Fei-Fei, and D. Koller, "Learning latent temporal structure for complex event detection," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1250–1257.

[16] H. Kuehne, J. Gall, and T. Serre, "An end-to-end generative framework for video segmentation and recognition," 2016. [Online]. Available: https://arxiv.org/abs/1509.01947

[17] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2017.

[18] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," 2015. [Online]. Available: https://arxiv.org/abs/1503.08909

[19] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1961–1970.

[20] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3156–3164.

[21] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei, "Every moment counts: Dense detailed labeling of actions in complex videos," 2017. [Online]. Available: https://arxiv.org/abs/1507.05738

[22] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," 2016. [Online]. Available: https://arxiv.org/abs/1611.05267

[23] F. Yi, H. Wen, and T. Jiang, "Asformer: Transformer for action segmentation," 2021. [Online]. Available: https://arxiv.org/abs/2110.08568

[24] Z. Lu and E. Elhamifar, "Fact: Frame-action cross-attention temporal modeling for efficient action segmentation," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 18 175–18 185.

[25] N. Behrmann, S. A. Golestaneh, Z. Kolter, J. Gall, and M. Noroozi, "Unified fully and timestamp supervised temporal action segmentation via sequence to sequence translation," 2022. [Online]. Available: https://arxiv.org/abs/2209.00638

[26] P. Wang, Y. Lin, E. Blasch, J. Wei, and H. Ling, "Efficient temporal action segmentation via boundary-aware query voting," 2024. [Online]. Available: https://arxiv.org/abs/2405.15995

[27] H. Zhang, X. Li, and L. Bing, "Video-llama: An instruction-tuned audio-visual language model for video understanding," 2023. [Online]. Available: https://arxiv.org/abs/2306.02858

[28] A. Marafioti, O. Zohar, M. Farré, M. Noyan, E. Bakouch, P. Cuenca, C. Zakka, L. Ben Allal, A. Lozhkov, N. Tazi, V. Srivastav, J. Lochner, H. Larcher, M. Morlon, L. Tunstall, L. von Werra, and T. Wolf, "Smolvlm: Redefining small and efficient multimodal models," 2025.