EventFormer: AU Event Transformer for Facial Action Unit Event Detection

Yingjie Chen
chenyingjie@pku.edu.cn
Jiarui Zhang
zjr954@pku.edu.cn
Tao Wang†
wangtao@pku.edu.cn
Yun Liang
ericlyun@pku.edu.cn

School of Computer Science
Peking University
Beijing, China

Abstract

Facial action units (AUs) play an indispensable role in human emotion analysis. We observe that although AU-based high-level emotion analysis is urgently needed by real-world applications, frame-level AU results provided by previous works cannot be directly used for such analysis. Moreover, as AUs are dynamic processes, the utilization of global temporal information is important but has been gravely ignored in the literature. To this end, we propose EventFormer for AU event detection, which is the first work directly detecting AU events from a video sequence by viewing AU event detection as a multiple class-specific sets prediction problem. Extensive experiments conducted on a commonly used AU benchmark dataset show the superiority of EventFormer under suitable metrics.

1 Introduction

Facial expression, as the most expressive emotional signal, plays an essential role in human emotion analysis. According to Facial Action Coding System (FACS) [13], facial action units (AUs) refer to a set of facial muscle movements and are the basic components of almost all facial behaviors. The increasing need for user emotion analysis in application scenarios, such as online education and remote interview, leads to rapid growth in the field of AU analysis in recent years. With the prosperity of deep learning, two mainstream tasks of AU analysis, AU recognition [5, 6, 10, 27, 29, 32] and AU intensity estimation [2, 14, 15, 25, 26], have seen great improvements in recent years, both of which aim to estimate AU occurrence state or intensity for a given frame, *i.e.*, frame-level AU results.

However, when it comes to high-level emotion analysis, frame-level analysis results are not enough for the need of some real-world applications [11, 22], due to the lack of various sequence-level information for further analysis, such as the occurrence frequencies, durations, and chronological order of AU events, each of which is a temporal segment containing one AU's complete temporal evolution, as shown in Fig. 1. For example, in public places

† Corresponding author.

such as airports, unnatural facial expressions or fleeting panics act as key information for the discrimination of a suspicious passenger. In this case, sequence-level AU event results are required to capture such abnormal emotions and based on which, warnings will be sent to officers for a further inspection of the person. Furthermore, sequence-level AU event results are also important to distinguish spontaneous and pretended facial expressions. For *happiness*, the identification depends heavily on the overlapping situation of events labeled AU6 and AU12, in which case not only their durations but also chronological order matters.

Some works [7, 12] have made attempts to generate AU event results based on frame-level or unit-level AU results via a series of postprocessing steps, but they suffer from the lacking of global temporal information and are highly dependent on hyper-parameters for postprocessing. To this end, we design an AU **Event** Trans**Former** (**EventFormer**) architecture to directly detect AU events from a video sequence by utilizing the benefit of global temporal information. EventFormer takes a video sequence as input and detects AU events for each AU class directly and simultaneously by viewing AU event detection as a *multiple class-specific sets prediction problem*.

Specifically, first, a region-aware AU feature encoder is used for extracting fine-grained AU features as frame embedding. Then, an event transformer encoder-decoder module is used to generate event embeddings by learning global dependencies among frames in a video sequence. Through self-attention mechanism, all frame embeddings are fed to transformer architecture simultaneously, and each frame can interact with others directly. In this way, a global view is maintained. After that, classification branch and regression branch are applied to each event embedding for event validity prediction and boundary regression, respectively.

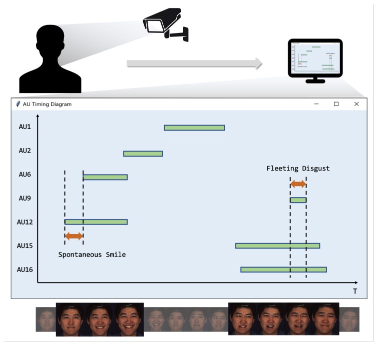

Figure 1: An illustration of the potential application scenario of direct sequence-level AU event detection. AU event provides overlapping as well as chronological information between different AUs, which is more suitable for further emotion analysis.

Unlike methods such as [19] modeling local temporal relations among several frames via RNN [17] or methods such as [12, 23] using local temporal information by extracting unit-level features, our EventFormer takes the whole video sequence as input and model global temporal relations through the mechanism of transformer, which allows each frame to access all the other frames simultaneously. And instead of generating AU events based on frame-level results through postprocessing which highly depends on manually selected hyper-parameters, EventFormer detects AU events in a direct way, and thus is able to alleviate discontinuous results. The key contributions of our work are listed as:

- To the best of our knowledge, this is the first work that directly detect AU events from a video sequence, which are more critical and practical for real-world applications.
- We propose EventFormer for AU event detection, taking advantage of the mechanism of transformer to maintain a temporal global view and alleviate discontinuous results.
- Extensive experiments conducted on a commonly used AU benchmark dataset, BP4D, show the superiority of our method under suitable metrics for AU event detection.

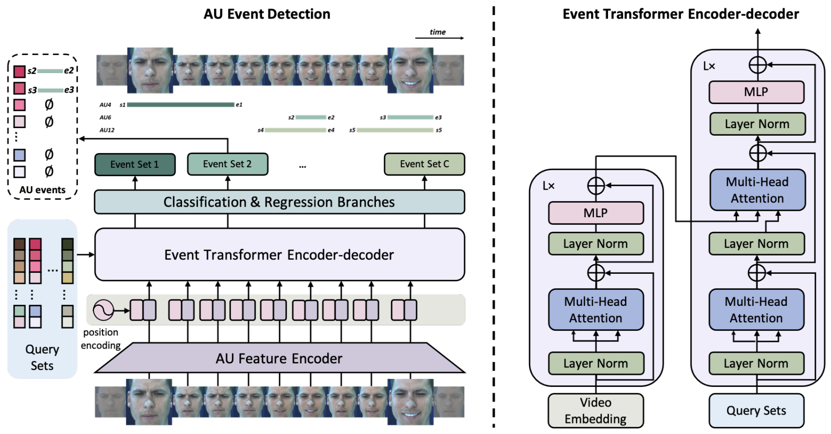

Figure 2: Architecture of EventFormer. EventFormer takes a video sequence as input, and AU feature encoder is first applied to the input to generate frame embeddings. Then position embeddings are concatenated to each frame embedding. Event transformer encoder models global relationships among frames via self-attention mechanism to enhance frame embeddings. After that, event transformer decoder takes encoder output and sets of queries, *i.e. Query Sets*, as input, and outputs aggregated event embeddings for each query. Then the output event embeddings are passed to two branches to obtain the final *Event Sets*.

2 Related Work

In recent years, AU analysis tasks have drawn increasing attention as fundamental tasks in the field of affective computing. Conventional methods [23, 24] mainly design hand-crafted features as the input of a classifier for AU recognition. With the development of deep learning, methods [8, 9, 14, 19, 34] have raised the performance of AU analysis to a new height. Since AUs are dynamic processes, methods such as [19] employ LSTM to model local temporal relations, but longer the video sequence, weaker the relationships between temporally far apart frames. To obtain AU event results, Ding *et al.* [12] proposed a method that extracts unit-level features, predicts event-related scores and generates AU events via a series of post-processing steps. In contrast, we propose EventFromer to model global temporal relations among frames in a video sequence and detect AU events in a direct way.

Transformer [28] has attracted increasing research interest in computer vision tasks. Self-attention mechanism as the core of transformer allows the model to aggregate information from the whole input sequence with much less memory consumption and computing time compared to RNNs. However, it is not until works [4, 30] succeeded that the architecture has been proved effective and efficient in computer vision tasks. Our EventFormer makes full use of the mechanism to model global temporal information.

3 EventFormer for AU Event Detection

3.1 Problem Definition

Given an input video sequence $\mathcal{I} = \{I_t\}_{t=1}^{T}$ with T frames recording facial actions, where I_t is the t^{th} frame in $\mathcal{I}$. The annotations of $\mathcal{I}$ are composed of a set of ground-truth AU events $\Phi_g = \{\phi_i = (s_i^g, e_i^g, c_i^g) | 0 \leq s_i^g < e_i^g \leq T, c_i^g \in \{1, 2, \dots, C\}\}_{i=1}^{M}$, where M is the number of ground-truth AU events in the video sequence $\mathcal{I}$, C is the number of AU classes, and s_i^{g}, e_i^{g}, c_i^{g} are the start time, end time, and AU class label of AU events ϕ_i, respectively. AU event detection aims to detect a set of events $\Phi_{\mathrm{p}} = \{\varphi_i = (s_i^{\mathrm{p}}, e_i^{\mathrm{p}}, c_i^{\mathrm{p}}) | 0 \le s_i^{\mathrm{p}} < e_i^{\mathrm{p}} \le T, c_i^{\mathrm{p}} \in \{1, 2, \dots, C\}\}_{i=1}^{N}$ which match Φ_{g} precisely and exhaustively. During training, the set of ground-truth AU events Φ_{g} is used as supervision for detected events Φ_{p}, and during inference, the Φ_{p} can be simply filtered as results.

3.2 Multiple Class-specific Sets Prediction

Due to AU co-occurrence relationships, AU events in different AU classes can have near-identical or exactly identical temporal boundaries. Inspired by DETR [4] which views object detection as a single set prediction problem, we view AU event detection as a multiple class-specific sets prediction problem. Instead of predicting a class-agnostic set with events of all classes, we predict multiple class-specific sets, each contains events for a specific AU class.

As noted above, AU event detection aims to detect class-specific sets of AU events, $\Phi_{\mathrm{p}} = \{\varphi_i = (s_i^{\mathrm{p}}, e_i^{\mathrm{p}}, c_i^{\mathrm{p}})\}_{i=1}^{N}$, from $\mathcal{I}$. If we bind AU class labels to events and search for a permutation of Φ_{p} to match Φ_{g} directly, some class mismatches caused by the multi-label property of AU event detection are hard to solve. For example, events with identical temporal boundaries but different AU labels in Φ_{g} can match with any permutation of the corresponding detected events in Φ_{p} during training, which causes unstable training and makes the class labels hard to learn. To alleviate the instability issue, we split Φ_{g} into C disjoint class-specific sets $\{\Phi_{\mathrm{g}}^c\}_{c=1}^{C}$ such that $\Phi_{\mathrm{g}} = \bigcup_{c=1}^{C} \Phi_{\mathrm{g}}^c$, where $\Phi_{\mathrm{g}}^c = \{\phi_i^c = (s_i^{\mathrm{g}}, e_i^{\mathrm{g}}, c_i^{\mathrm{g}}) | \forall \phi_i \text{ s.t. } c_i^{\mathrm{g}} = c\}_{i=1}^{N_c}$, and N_c is the number of ground-truth events belonging to class c. In this way, the problem turns into predicting several class-specific sets Φ_{p}^c, ($\Phi_{\mathrm{p}} = \bigcup_{c=1}^{C} \Phi_{\mathrm{p}}^c$), one for each AU class, *i.e.* a multiple class-specific sets prediction problem.

For the implementation of EventFormer, assuming N_0 is a number larger than any N_c, we pad Φ_{g}^c with $\varnothing$ (no event) to make the set $\tilde{\Phi}_{\mathrm{g}}^c = \{\tilde{\phi}_i^c = (s_i^{\mathrm{g}}, e_i^{\mathrm{g}}, v_i^{\mathrm{g}})\}_{i=1}^{N_0}$ with a fixed number of events, where $v_i^{\mathrm{g}} \in \{0, 1\}$, $v_i^{\mathrm{g}} = 0$ represents $\tilde{\phi}_i^c$ is not a valid event, *i.e.* $\varnothing$ for padding, and $v_i^{\mathrm{g}} = 1$ represents $\tilde{\phi}_i^c$ is a valid event. We denote $\tilde{\Phi}_{\mathrm{p}}^c = \{\tilde{\varphi}_i^c = (s_i^{\mathrm{p}}, e_i^{\mathrm{p}}, v_i^{\mathrm{p}})\}_{i=1}^{N_0}$ as the set of N_0 detected events for class c, called *Event Set c*. EventFormer takes a video sequence $\mathcal{I}$ and a union of C *Query Sets* $\Phi_{\mathrm{q}} = \bigcup_{c=1}^{C} \Phi_{\mathrm{q}}^c$ as inputs, and outputs a union of C corresponding *Event Sets* $\tilde{\Phi}_{\mathrm{p}} = \bigcup_{c=1}^{C} \tilde{\Phi}_{\mathrm{p}}^c$.

3.3 EventFormer Architecture

As shown in Fig. 2, our EventFormer mainly consists of three parts.

AU Feature Encoder Compared to coarse-grained body actions, AUs only cause subtle appearance changes on several local facial regions, which puts high demands on the discriminability of frame embeddings. Thus, we design a region-aware AU feature encoder to extract local features for each AU separately to preserve more detailed information. Each frame $I_t \in \mathbb{R}^{3 \times H \times W}$ in $\mathcal{I}$ is fed to a backbone network to extract $F^{\mathrm{global}} \in \mathbb{R}^{d \times H_0 \times W_0}$ as global feature. And C spatial attention layers [35] are applied to the global feature to extract local features $f^{\mathrm{local}} \in \mathbb{R}^{d}$ for each AU. Then, the concatenated local features $F^{\mathrm{local}} \in \mathbb{R}^{(d \times C)}$ are mapped to $E_t \in \mathbb{R}^{d_{\mathrm{m}}}$ via a linear layer as frame embedding for I_t.

Event Transformer Encoder-decoder Module We start from the original transformer encoder-decoder architecture [28] and design an event transformer encoder-decoder module specially for AU event detection. Our event transformer encoder-decoder consists of L encoder layers and L decoder layers. To ensure a stable training period, LayerNorm [1] is applied before Multi-head Attention and Multi-layer Perceptron, according to [31]. To maintain positional information in time dimension, positional encoding is employed to generate positional embeddings $\mathcal{P} = \{P_t\}_{t=1}^{T} \in \mathbb{R}^{T\times d_{\mathrm{m}}}$, corresponding to frame embeddings $\mathcal{E} = \{E_t\}_{t=1}^{T} \in \mathbb{R}^{T\times d_{\mathrm{m}}}$. The transformer encoder takes the concatenated frame embeddings and positional embeddings, *i.e.* video embedding, as input and outputs refined frame embeddings by enabling interaction among frames via self-attention mechanism. Event transformer decoder takes event queries $Q \in \mathbb{R}^{(CN_0)\times d_{\mathrm{m}}}$ as input, which can be regarded as the union of *C Query Sets*, $\Phi_{\mathrm{q}} = \bigcup_{c=1}^{C} \Phi_{\mathrm{q}}^{c}$, where $\Phi_{\mathrm{q}}^{c} = \{q_i^c\}_{i=1}^{N_0}$. Each query $q_i^c \in \mathbb{R}^{d_{\mathrm{m}}}$ is a learned positional embedding, which differs from each other. Queries first interact with each other through self-attention to alleviate event redundancy, and then interact with the encoder output, *i.e.* the refined frame embeddings as keys and values, to aggregate frame embeddings relative to each potential event as event embeddings $D \in \mathbb{R}^{(CN_0)\times d_{\mathrm{m}}}$.

Classification and Regression Branches The output event embeddings D of the event transformer encoder-decoder module are further fed into classification branch and regression branch separately. For each feature vector $d_i \in \mathbb{R}^{d_{\mathrm{m}}}$ in D representing a potential event φ_i, the regression branch aims to estimate the start time s_i and duration l_i of the event, and $e_i = min(T, s_i + l_i)$. The classification branch aims to estimate a one-hot vector $\hat{p}_i \in \mathbb{R}^2$ that denotes the probabilities of the value of v_i. We use a linear layer to output the classification probability $\hat{p}_i$ and two linear layers to regress s_i and l_i. After that, by reorganizing events in order, $\tilde{\Phi}_{\mathrm{p}} = \bigcup_{c=1}^{C} \tilde{\Phi}_{\mathrm{p}}^{c}$ is obtained.

4 Training and Inference of EventFormer

To train EventFormer, a multiple class-specific sets matching cost is introduced for class-specific bipartite matching between each pair of $\tilde{\Phi}_{\mathrm{g}}^{c}$ and $\tilde{\Phi}_{\mathrm{p}}^{c}$. After the matching for each AU class, a multiple class-specific sets prediction loss can be computed for back-propagation.

4.1 Multiple Class-specific Sets Matching Cost

Due to the disorder of events in one set, the loss function designed for multiple class-specific sets prediction should be invariant by a permutation of the detected events with identical class labels, *i.e.* in one *Event Set*. We apply a loss based on Hungarian algorithm [18], to find a bipartite matching between ground-truth events and detected ones for each class.

A permutation of N_0 elements $\sigma \in \Omega_{N_0}$ is searched by finding a bipartite matching between $\tilde{\Phi}_{\mathrm{g}}^{c}$ and $\tilde{\Phi}_{\mathrm{p}}^{c}$ for each class that minimizes the total matching cost, as shown in Eq. 1:

$$\hat{\sigma} = \arg\min_{\sigma\in\Omega_{N_0}} \textstyle\sum_{i=1}^{N_0} \mathcal{L}_{\mathrm{match}}(\tilde{\phi}_i^c, \tilde{\varphi}_{\sigma(i)}^c), \tag{1}$$

where $\mathcal{L}_{\mathrm{match}}(\tilde{\phi}_i^c, \tilde{\varphi}_{\sigma(i)f}^c)$ is a pair-wise matching cost between ground-truth $\tilde{\phi}_i^c$ and a detected event $\tilde{\varphi}_{\sigma(i)}^c$ with matching index $\sigma(i)$. The loss function of matching is designed to minimize the distance between matched pairs and maximize the validity of matched events at the same time. We define $\mathcal{L}_{\mathrm{match}}(\tilde{\phi}_i^c, \tilde{\varphi}_{\sigma(i)}^c)$ as

$$\mathbb{1}_{\{v_i^{\mathrm{g}}=1\}} \left(\lambda_{\mathrm{bound}}\mathcal{L}_{\mathrm{bound}}(\tilde{\phi}_i^c, \tilde{\varphi}_{\sigma(i)}^c) - \lambda_{\mathrm{valid}}\hat{p}_{\sigma(i)}^c[v_i^{\mathrm{g}}]\right), \tag{2}$$

where $\hat{p}_{\sigma(i)}^c \in \mathbb{R}^2$ denotes the probabilities of the value of $v_{\sigma(i)}^{p}$ indicating whether $\tilde{\varphi}_{\sigma(i)}^c$ is a valid event, *i.e.* the probabilities of $v_{\sigma(i)}^{\mathrm{p}} \in [0,1]$, $\hat{p}_{\sigma(i)}^c[v_i^{\mathrm{g}}]$ denotes the probability of

$v^{\mathrm{p}}_{\sigma(i)} = v^{\mathrm{g}}_i$, and λ_{bound} and λ_{valid} are for balancing. The boundary loss $\mathcal{L}_{\mathrm{bound}}$ measures the similarity between a pair of matched ground-truth event and detected event. L1 loss measures the numerical difference of the regression results, and tIoU loss measures the overlapping area ratio of matched pairs. Both of them are used, considering pairs of ground-truth event and detected event may have a minor difference in terms of L1 but a huge difference in terms of tIoU. Thus, a linear combination of tIoU loss (Eq. 3) and L1 loss is used as our boundary loss $\mathcal{L}_{\mathrm{bound}}$, as shown in Eq. 4.

$$\begin{aligned} T_{\cap} &= max(0, min(e_1, e_2) - max(s_1, s_2)), \\ \mathcal{L}_{\mathrm{tIoU}}((s_1,e_1),(s_2,e_2)) &= \frac{T_{\cap}}{(e_1 - s_1) + (e_2 - s_2) + T_{\cap}}. \end{aligned} \tag{3}$$

$$\begin{aligned} \mathcal{L}_{\mathrm{bound}}(\tilde{\phi}^c_i, \tilde{\varphi}^c_{\sigma(i)}) = \lambda_{\mathrm{tIoU}} \mathcal{L}_{\mathrm{tIoU}}((s^{\mathrm{g}}_i, e^{\mathrm{g}}_i), (s^{\mathrm{p}}_{\sigma(i)}, e^{\mathrm{p}}_{\sigma(i)})) \\ + \lambda_{\mathrm{L1}}(\|s^{\mathrm{g}}_i - s^{\mathrm{p}}_{\sigma(i)}\| + \|e^{\mathrm{g}}_i - e^{\mathrm{p}}_{\sigma(i)}\|), \end{aligned} \tag{4}$$

where λ_{tIoU} and λ_{L1} are for balancing.

4.2 Multiple Class-specific Sets Prediction Loss

After finding a bipartite matching minimizing the matching cost, the loss function can be computed. A combination of $\mathcal{L}_{\mathrm{bound}}$ and $\mathcal{L}_{\mathrm{class}}$ (Eq. 5) forms the event detection loss $\mathcal{L}$, as shown in Eq. 6.

$$\mathcal{L}_{\mathrm{class}}(p^c_i, \hat{p}^c_{\sigma(i)}) = -\sum\nolimits_{v \in (0,1)} p^c_i(v) \log(\hat{p}^c_{\sigma(i)}(v)). \tag{5}$$

$$\mathcal{L} = \sum\nolimits_{c=1}^{C} \sum\nolimits_{i=1}^{N_0} (\mathbb{1}_{\{v^{\mathrm{g}}_i = 1\}} \mathcal{L}_{\mathrm{bound}}(\tilde{\phi}^c_i, \tilde{\varphi}^c_{\sigma(i)}) + \lambda_{\mathrm{class}} \mathcal{L}_{\mathrm{class}}(p^c_i, \hat{p}^c_{\sigma(i)})), \tag{6}$$

where $p^c_i \in \mathbb{R}^2$ is the one-hot encoding of v^g_i for $\tilde{\phi}^c_i$, and λ_{class} is for balancing. It is worth mentioning that all the detected events are involved in the calculation of $\mathcal{L}_{\mathrm{class}}$, but only the matched events in *Event Sets* are involved in the calculation of $\mathcal{L}_{\mathrm{bound}}$.

In the inference stage, bipartite matching is disabled. By given a threshold τ, we can simply filter out events with $\hat{p}_i$ lower than the threshold and preserve more valid events. For each *Event Set* c, each preserved event $\tilde{\varphi}^c_i$ is assigned with an AU class label c to form one final detected event $\varphi_i = (s^{\mathrm{p}}_i, e^{\mathrm{p}}_i, c^{\mathrm{p}}_i)$ with $c^{\mathrm{p}}_i = c$ in Φ_{p}. In this way, the set of final AU events Φ_{p} can be easily obtained.

5 Experiments

5.1 Experimental Settings

Datasets & Metrics Extensive experiments are conducted on a commonly used benchmark dataset, BP4D [33]. In BP4D, 328 videos of 41 participants are taken, including 23 women and 18 men. Each frame is annotated by certificated FACS coders with binary AU occurrence labels. We consider 12 emotion-related AUs on BP4D. To construct the training data, we use a sliding window with length T to truncate all the videos into several equal length video sequences, and there is an overlap of $T/2$. Based on binary AU occurrence labels, ground-truth AU events Φ_{g} are obtained for each video sequence. Following the common protocol mentioned in [34], subject-exclusive 3-fold cross-validation is conducted for all experiments.

Considering that AU event detection shares some similarities with temporal action detection [20], we select several suitable metrics for AU event detection drawing on those used

Backbone	Scheme	mAP@0.3	mAP@0.4	mAP@0.5	mAP@0.6	mAP@0.7	AR@10	AR@50	AR@100	AUC
RN18	Frame2Event [36]	10.03	8.97	8.07	7.17	6.22	3.83	24.15	60.15	27.34
	Unit2Event [2]	22.09	19.52	16.71	14.53	12.54	48.22	**73.83**	**74.51**	**68.17**
	EventFormer	**33.82**	**29.09**	**24.34**	**20.29**	**16.39**	**51.46**	62.27	68.34	60.06
RN34	Frame2Event [36]	14.33	12.20	10.53	8.80	7.23	3.93	18.58	41.37	20.09
	Unit2Event [2]	24.04	21.18	18.15	15.84	13.77	49.09	**73.79**	**75.85**	**68.49**
	EventFormer	**36.92**	**32.06**	**26.87**	**22.37**	**18.18**	**52.83**	64.95	69.62	62.12
RN50	Frame2Event [36]	16.56	14.32	12.50	10.63	8.88	4.16	18.94	40.36	20.30
	Unit2Event [2]	24.40	22.01	19.36	17.03	14.67	50.24	**73.46**	**76.15**	**68.24**
	EventFormer	**41.41**	**35.79**	**30.10**	**25.00**	**20.32**	**53.59**	66.72	72.31	63.76

Table 1: Comparison among schemes on BP4D in terms of mAP@tIoU, AR@AN, AUC.

in that task. The goal of AU event detection task is to detect AU events with not only high precision but also acceptable recall. Thus, we consider Mean Average Precision(mAP) and Average Recall with an average number of events (AR@AN) at different tIoU thresholds α. α is set to $[0.3 : 0.1 : 0.7]$ for mAP and $[0.5 : 0.05 : 0.95]$ for AR@AN. We also report Area under the AR *vs*. AN curve (AUC) for evaluation.

Implementation Details Facial images are aligned, cropped, and resized to 256×256. RN50 [16] without the last linear layer is used as backbone in AU feature encoder. Empirically, we set local feature dimension $d = 512$, $H_0 = W_0 = 16$, embedding dimension $d_{\mathrm{m}} = 256$, and the number of encoder/decoder layers L is set to 6. Other hyper-parameters λ_{bound}, λ_{valid}, λ_{tIoU}, λ_{L1} and λ_{class} are set to 5, 1, 2, 5, 1, respectively. The number of queries N_0 in each *Query Set* is set to 100, and τ is set to 0.5. We train EventFormer with AdamW [21] optimizer setting transformer's learning rate to 10^{-4}, AU feature encoder's learning rate to 10^{-5}, and weight decay to 10^{-4}. Batch size is set to 8 and the number of training epochs is set to 100. All models are trained on two NVIDIA Tesla V100 GPUs.

5.2 Comparison among Schemes

We classify AU event detection schemes into three categories, which use frame-level (Frame2Event), unit-level (Unit2Event), and video-level (Video2Event) results to detect AU events respectively. To demonstrate the effectiveness of EventFormer (Video2Event), two methods following other schemes are selected for comparison. For a fair comparison, all methods apply the same AU feature encoder pre-trained using AU occurrence labels.

Comparison to Frame2Event Frame2Event scheme collects frame-level AU results and converts them to AU events through postprocessing such as Temporal Actionness Grouping (TAG) [36], which involves two hyper-parameters, *water level* γ and *union threshold* τ. We use the same AU feature encoder and an MLP as classifier to obtain frame-level AU results. Based on the predicted AU occurrence probabilities, candidate events are generated under several combinations of γ and τ. Specifically, we sample γ and τ within the range of 0.5 and 0.95 with a step of 0.05. The confidence score of each candidate event (s,e,c) is computed by averaging the probabilities of class c within segment (s,e). Soft-NMS [3] is employed to select N_0 events for each class from the candidate events.

As shown in Table 1, EventFormer outperforms Frame2Event method by a large margin in terms of mAP given any tIoU threshold, regardless of what backbone is used. Especially, EventFormer achieves a performance gain of 24.85% in mAP@0.3 than Frame2Event method using RN50 as backbone, which shows the superiority of EventFormer. We also notice that Frame2Event method obtains a pretty low AR given a small AN, which is because the prediction jitters due to the lack of a global view make it hard to use a set of fixed hyper-parameters to balance the trade-off between AP and AR. Such limitation reflects the necessity of maintaining a global view and detecting events directly.

Comparison to Unit2Event We choose AUPro [7] on behalf of Unit2Event scheme, which extracts unit-level features and predicts event-related scores to generate AU events. AUPro estimates the start and end probabilities, P_s and P_e, for each time position exhaustively, and generates an action completeness map P_c consisting of the completeness score for any event (s,e), and the final confidence score for an event (s,e) is calculated as $P_s(s) \times P_e(e) \times P_c(s,e)$. Since the method only predicts class-agnostic events, we simply make it generates C sets of P_s, P_e, and P_c, one set for each class. We adopt Soft-NMS to select N_0 events out of T^2 detected events for each class.

Hyper-parameters	Values	mAP@0.5	AUC
#Queries in *Query Set* N_0	10	**31.78**	34.77
	50	31.33	45.77
	100	30.03	62.32
	200	25.04	**63.68**
Embedding Dimension d_m	128	**30.22**	59.65
	256	30.03	**62.32**
	512	24.82	61.20
	1024	22.95	59.33
#Encoder/decoder layers L	3	29.39	58.67
	4	29.58	59.51
	5	**30.31**	62.05
	6	30.03	**62.32**

Table 2: Sensitivity to hyper-parameters.

As shown in Table 1, EventFormer outperforms Unit2Event method with any backbone in terms of mAP given any tIoU. Specifically, EventFormer achieves a performance gain of 17.01% in mAP@0.3 than Unit2Event method with RN50 as backbone. Since Unit2Event method generates events exhaustively, it is supposed to obtain better results in terms of AR. Although EventFormer performs a little bit worse than Unit2Event method in terms of AR given a large AN, it outperforms it in AR@10 by 3.35%, which indicates that events generated by EventFormer are of better quality.

5.3 Sensitivity Analysis

Table 2 shows sensitivity analysis of hyper-parameters, including the number of queries in *Query Set* N_0, embedding dimension d_m and the number of layers L of encoder and decoder. As the number of queries in *Query Set* increases, mAP decreases while AUC increases, due to the trade-off between mAP and AR. We notice that the mAP does not decrease a lot from $N_0 = 10$ to $N_0 = 100$, and AUC increases much slower when $N_0 > 100$. Thus, we choose $N_0 = 100$ for EventFormer. As for d_m, AUC reaches its peak when we set d_m to 256, while at the same time, mAP is also around its best score. As for the number of layers L, we notice that EventFormer achieves better performance with L increasing. For the balance between computing complexity and model performance, we choose $L = 6$ for EventFormer.

5.4 Class-agnostic Set *vs*. Class-specific Sets

To show the superiority of viewing AU event detection as a multiple class-specific sets prediction problem instead of a single class-agnostic set prediction problem, We implement a class-agnostic version of EventFormer for comparison, which generates events with AU class labels directly and applies bipartite matching once between the ground-truth events and detected ones of all classes. From Fig. 3 we can see that the class-agnostic version obtains poor results on AU2, AU15, AU23, and AU24, of which the mAP@0.5 and AR@100 are near zero. The results variance among AU classes is huge for the class-agnostic version, while the class-specific version achieves relatively balanced results. We attribute the superiority to the binding between sets and AU classes, which is essential for stabilizing training and alleviating the variance of the results among AU classes caused by data imbalance problem.

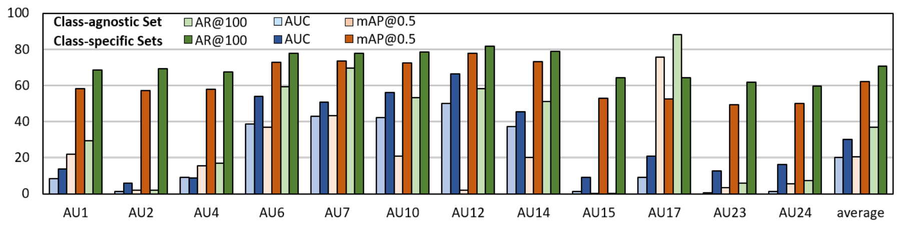

Figure 3: A comparison between multiple class-specific sets prediction and single class-agnostic set prediction.

5.5 Qualitative Results

Visualization of Attention in EventFormer To better understand how the attention mechanism takes effect in EventFormer, we visualize the attention weights of the last layer of transformer decoder in Fig. 4. The brightest parts of the attention show that the cross-attention of event transformer decoder tends to focus on the embeddings of frames where the states of AUs change. The results also show that EventFormer could capture subtle and transient appearance changes that occur in a very short duration ($\leq$5 frames) and detect an event successfully, as shown in Fig. 4(a).

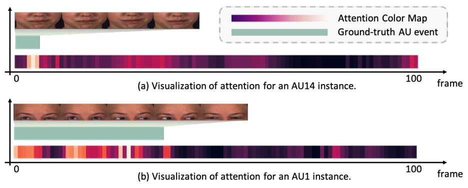

Figure 4: Visualization of attention.

Visualization of Detected AU Events Fig. 5 shows AU events detected by (a) Frame2Event method and (b) EventFormer. AU events detected by EventFormer are of better quality, while AU events detected by Frame2Event method contains several false positive events with a very short duration. There is a false positive event of AU2 (Outer Brow Raiser) in Fig 5(b), and the visualized frames corresponding to this period show a process of the subject opening her eyes, during which wrinkles appeared above her eyebrow, misleading EventFormer to detect an AU2 event.

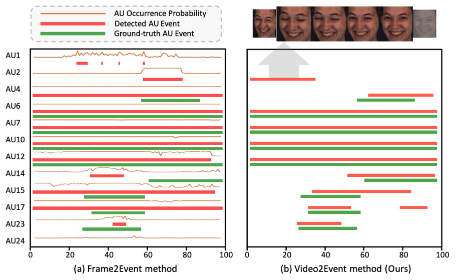

Figure 5: Visualization of detected AU events.

6 Conclusion

This paper focuses on making full use of global temporal information to directly detect AU events from a whole video sequence, which are more practical and critical in some real-world application scenarios. We propose EventFormer for AU event detection, which maintains a temporal global view to alleviate discontinuous results. Future work will focus on high-level emotion analysis based on AU events, such as detecting unnatural facial expressions.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] T. Baltrušaitis, L. Li, and L. Morency. Local-global ranking for facial expression intensity estimation. In *ACII*, pages 111–118, 2017. doi: 10.1109/ACII.2017.8273587.

[3] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Soft-nms – improving object detection with one line of code. In *ICCV*, Oct 2017.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020.

[5] Yingjie Chen, Diqi Chen, Yizhou Wang, Tao Wang, and Yun Liang. Cafgraph: Context-aware facial multi-graph representation for facial action unit recognition. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1029–1037, 2021.

[6] Yingjie Chen, Han Wu, Tao Wang, Yizhou Wang, and Yun Liang. Cross-modal representation learning for lightweight and accurate facial action unit detection. *IEEE Robotics and Automation Letters*, 6(4):7619–7626, 2021.

[7] Yingjie Chen, Jiarui Zhang, Diqi Chen, Tao Wang, Yizhou Wang, and Yun Liang. Aupro: Multi-label facial action unit proposal generation for sequence-level analysis. In *ICONIP*, pages 88–99. Springer, 2021.

[8] Yingjie Chen, Chong Chen, Xiao Luo, Jianqiang Huang, Xian-Sheng Hua, Tao Wang, and Yun Liang. Pursuing knowledge consistency: Supervised hierarchical contrastive learning for facial action unit recognition. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 111–119, 2022.

[9] Yingjie Chen, Diqi Chen, Tao Wang, Yizhou Wang, and Yun Liang. Causal intervention for subject-deconfounded facial action unit recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 374–382, 2022.

[10] UmurAybars Ciftci, Xing Zhang, and Lijun Tin. Partially occluded facial action recognition and interaction in virtual reality applications. In *ICME*, pages 715–720. IEEE, 2017.

[11] Jeffrey F Cohn and Karen Schmidt. The timing of facial motion in posed and spontaneous smiles. In *Active Media Technology*, pages 57–69. World Scientific, 2003.

[12] Xiaoyu Ding, Wen-Sheng Chu, Fernando De la Torre, Jeffery F Cohn, and Qiao Wang. Facial action unit event detection by cascade of tasks. In *ICCV*, pages 2400–2407, 2013.

[13] P. Ekman and W. Friesen. *Facial action coding system: A technique for the measurement of facial movement*. 1978.

[14] Yachun Fan, Jie Shen, Housen Cheng, and Feng Tian. Joint facial action unit intensity prediction and region localisation. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2020.

[15] Yingruo Fan, Jacqueline Lam, and Victor Li. Facial action unit intensity estimation via semantic correspondence learning with dynamic graph convolution. In *AAAI*, volume 34, pages 12701–12708, 2020.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. LSTM can solve hard long time lag problems. In *NIPS*, 1996.

[18] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[19] Wei Li, Farnaz Abtahi, and Zhigang Zhu. Action unit detection with region adaptation, multi-labeling learning and optimal temporal fusing. In *CVPR*, 2017.

[20] Chuming Lin, Jian Li, Yabiao Wang, Ying Tai, Donghao Luo, Zhipeng Cui, Chengjie Wang, Jilin Li, Feiyue Huang, and Rongrong Ji. Fast learning of temporal action proposal via dense boundary generator. In *AAAI*, volume 34, pages 11499–11506, 2020.

[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[22] Karen L Schmidt, Zara Ambadar, Jeffrey F Cohn, and L Ian Reed. Movement differences between deliberate and spontaneous facial expressions: Zygomaticus major action in smiling. *Journal of nonverbal behavior*, 30(1):37–52, 2006.

[23] T. Simon, M. H. Nguyen, F. De La Torre, and J. F. Cohn. Action unit detection with segment-based svms. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2737–2744, 2010. doi: 10.1109/CVPR.2010.5539998.

[24] Tomas Simon, Minh Hoai Nguyen, Fernando De La Torre, and Jeffrey F Cohn. Action unit detection with segment-based svms. In *CVPR*, pages 2737–2744. IEEE, 2010.

[25] Tengfei Song, Zijun Cui, Yuru Wang, Wenming Zheng, and Qiang Ji. Dynamic probabilistic graph convolution for facial action unit intensity estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4845–4854, 2021.

[26] Xinhui Song, Tianyang Shi, Zunlei Feng, Mingli Song, Jackie Lin, Chuanjie Lin, Changjie Fan, and Yi Yuan. Unsupervised learning facial parameter regressor for action unit intensity estimation via differentiable renderer. In *ACM MM*, pages 2842–2851, 2020.

[27] Praveen Tirupattur, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. Modeling multi-label action dependencies for temporal action localization. In *CVPR*, pages 1460–1470, 2021.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[29] Can Wang and Shangfei Wang. Personalized multiple facial action unit recognition through generative adversarial recognition network. In *ACM MM*, pages 302–310, 2018.

[30] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020.

[31] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *ICML*, pages 10524–10533. PMLR, 2020.

[32] Huiyuan Yang and Lijun Yin. Re-net: A relation embedded deep model for au occurrence and intensity estimation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[33] Xing Zhang, Lijun Yin, Jeffrey F Cohn, Shaun Canavan, Michael Reale, Andy Horowitz, Peng Liu, and Jeffrey M Girard. Bp4d-spontaneous: A high-resolution spontaneous 3d dynamic facial expression database. *Image and Vision Computing*, 32(10): 692–706, 2014.

[34] Kaili Zhao, Wen-Sheng Chu, and Honggang Zhang. Deep region and multi-label learning for facial action unit detection. In *CVPR*, 2016.

[35] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In *CVPR*, pages 3085–3094, 2019.

[36] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *ICCV*, Oct 2017.