Supplementary Material
ENIGMA-51: Towards a Fine-Grained Understanding of Human Behavior in Industrial Scenarios

Francesco Ragusa [1,2], Rosario Leonardi[1], Michele Mazzamuto[1,2],
Claudia Bonanno[1,2], Rosario Scavo[1], Antonino Furnari[1,2], Giovanni Maria Farinella[1,2]

[1]FPV@IPLab - University of Catania, Italy
[2]Next Vision s.r.l. - Spinoff of the University of Catania, Italy

Abstract

This document is intended for the convenience of the reader and reports additional information about the collection and the annotations of the ENIGMA-51 dataset, as well as implementation details of the adopted baselines. This supplementary material is related to the following paper: ENIGMA-51: Towards a Fine-Grained Understanding of Human Behavior in Industrial Scenarios, IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2024. The reader is referred to the aforementioned manuscript for further information, and to our web page https://iplab.dmi.unict.it/ENIGMA-51 *to download the dataset.*

1. The ENIGMA-51 Dataset

The ENIGMA-51 dataset has been acquired in an industrial laboratory by 19 subjects who wore a Microsoft Hololens 2 providing audio instructions to follow to complete a repair procedure of electrical boards. The dataset is composed of 51 egocentric videos. Each video includes a complete repair procedure of an electrical board where movable objects (see Section 1.2.1) were placed in random positions on the working table. Each subject acquired at least one video for each electrical board (*high* and *low* voltage) obtaining a total of 51 videos. The dataset was divided into training, validation, and test sets. Each set contains videos acquired from different subjects, and there is no overlap between the subjects in any of the sets. Figure 1 shows the industrial laboratory in which the ENIGMA-51 dataset has been acquired.

1.1. Instruction for repair procedures

We designed two procedures composed of instructions that involve humans interacting with the objects of the laboratory to achieve the goal of repairing two different electrical boards. These procedures have been designed with the support of industrial experts with the aim of capturing realistic human-object interactions in a real industrial domain. Specifically, we designed a procedure for each electrical board: *High Voltage Repair* and *Low Voltage Repair*. Then, for each procedure we forced the use of one of the electric or standard screwdrivers and the soldering of one of the resistor/capacitor/transformer electrical components, obtaining four variants of each procedure. Each procedure is designed to allow the worker to interact with all the industrial objects and electric machinery present in the laboratory. With the provided instructions, we expected users to interact with movable objects (e.g., *"Take the soldering iron's probe"* or *"Place the electric board on the working area"*) and with fixed machinery (e.g., *''Press the two green buttons on panel A"* or *"Adjust the voltage knob of the power supply to set a voltage of 5 Volts"*). As example, Table 1 reports a complete repair procedure for the low-voltage electric board using the standard screwdriver and soldering the capacitor. To provide instructions to the user without the use of physical manuals, we developed an application for the Microsoft HoloLens 2.

1.1.1 HoloLens2 acquisition application

We developed an application for Microsoft HoloLens 2 using the Unit 3D graphic engine to provide instructions to the participants during the acquisition. In particular, the application helps the operators during the acquisition phase, providing audio instructions and showing images to facilitate complex operations (e.g., where to connect the oscillo-

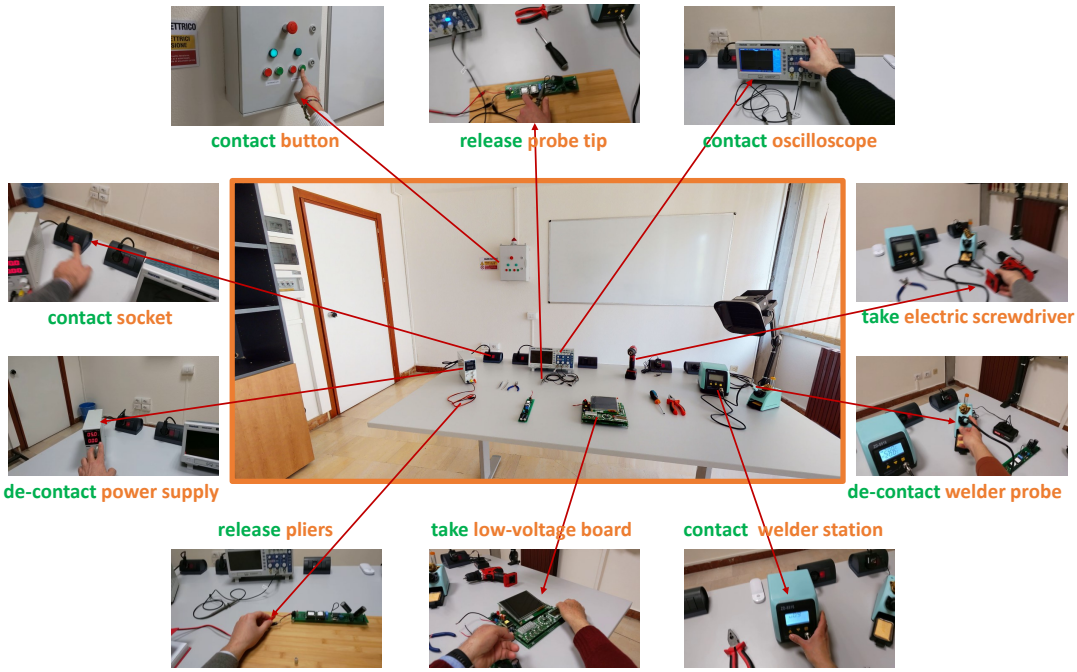

Figure 1. The ENIGMA-51 dataset has been acquired in an industrial laboratory. We show some interaction key frames with the related verb (in green) and the object involved in the interaction (in orange).

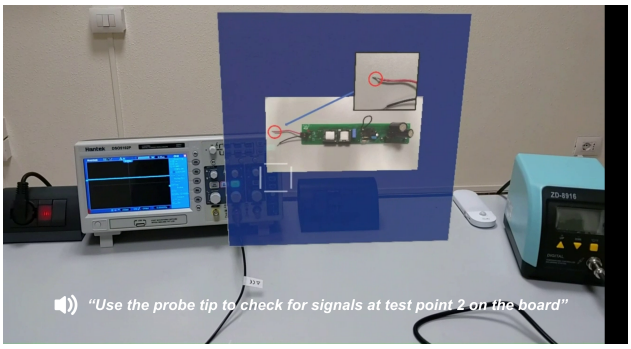

Figure 2. A screenshot captured from the developed application, during the acquisition phase.

scope ground clip). Figure 2 shows an example of the acquisition tool. The application integrates voice commands for the human-device interaction (e.g. the possibility to say "next" or "back" over the steps of the procedure). The audio guide describes the operations to be performed during the acquisitions. To create these audio tracks, a Python script has been created that utilizes the gTTS library for interacting with Google Translate APIs. This script takes an input text file divided into textual blocks and converts it into a set of MP3 audio tracks. After wearing the device, the operator will interact with the application using the following voice commands:

- **Forward**: to play the audio track for the next set of instructions.
- **Backward**: to play the audio track of the previous set of instructions.
- **Repeat**: to replay the audio track for the current set of instructions.
- **Record**: Using this command, the operator starts video recording (the application will play a sound for confirmation).
- **Stop**: Using this command, the operator stops the video recording (the application will play a sound for confirmation).

1.2. Data Annotation

1.2.1 Object Annotations

In our industrial setting, we considered both fixed (e.g., *oscilloscope*, *power suppl*) and movable objects (e.g., *screwdriver*, *electric boar*) present in the industrial laboratory. In particular, our object taxonomy is composed of 25 different objects: *power supply, power supply cables, oscilloscope, oscilloscope probe tip, oscilloscope ground clip, welder station, welder base, welder probe tip, electric screwdriver, electric screwdriver battery, battery connector, screwdriver,*

Step	Description	Step	Description	Step	Description
1	Sit at the workbench	41	Place the pliers on the workbench	81	Place the pliers on the workbench
2	Pronounce the voice command "Record" to start recording, and wait for the acoustic signal for confirmation	42	Lower the soldering iron temperature to the minimum (160°C) using the yellow "DOWN" button	82	Place the pliers on the workbench
3	Turn the lamp located on the workbench on and off while looking at it	43	Turn off the soldering iron using the socket switch	83	Turn off the soldering iron using the socket switch
4	Exit the laboratory	44	Fix the board to the workbench using the screwdriver	84	Connect the display to the board
5	Enter the laboratory and close the door	45	Observe the power supply	85	Turn on the power supply using the socket switch
6	Go to panel A and observe it for a moment	46	Adjust the current knob of the power supply until the green LED lights up	86	Adjust the power supply voltage knob to set a voltage of 5 Volts
7	Press the two green buttons on panel A	47	Connect the power supply cables to the board's power points	87	Connect the power supply cables to the board's power points
8	Head back to the workbench and sit down	48	Observe the board for a few seconds to verify the red LED turning on	88	Observe the power supply
9	Observe the low-voltage board for a while	49	Turn off the power supply using the socket switch	89	Turn off the power supply using the socket switch
10	Take the low-voltage board and place it on the work area	50	Set the current and voltage knobs of the power supply to 0	90	Set the current and voltage knobs of the power supply to 0
11	Remove the screws from the workbench using the screwdriver	51	Disconnect the clip and probe of the oscilloscope	91	Disconnect the power supply cables
12	Secure the board to the workbench using the screwdriver	52	Turn on the oscilloscope using the socket switch	92	Observe the board for a few seconds to verify the red LED turning on
13	Observe the power supply	53	Activate channel 2 of the oscilloscope using the "CH2 MENU" button with a blue outline	93	Turn on the oscilloscope using the socket switch
14	Turn on the power supply using the socket switch	54	Connect the ground clip of the probe to test point 1	94	Activate channel 2 of the oscilloscope using the "CH2 MENU" button
15	Adjust the current knob of the power supply until the green LED lights up	55	Use the probe tip to check for signals at test point 2 on the board	95	Connect the ground clip of the probe to test point 1
16	Adjust the voltage knob of the power supply to set a voltage of 5 Volts	56	Press the "Auto Set" button on the oscilloscope	96	Use the probe tip to check for signals at test point 2 on the board
17	Connect the power supply cables to the board's power points	57	Observe the oscilloscope's display	97	Press the "Auto Set" button on the oscilloscope
18	Observe the board for a few seconds to verify the red LED turning on	58	Rotate the "position" knob above the "CH2 MENU" button with a blue outline randomly	98	Observe the oscilloscope's display
19	Turn off the power supply using the socket switch	59	Repeat the previous three steps for the remaining test points (from number 3 to number 7)	99	Rotate the "position" knob above the "CH2 MENU" button with a blue outline randomly
20	Set the current and voltage knobs of the power supply to 0	60	Set the current and voltage knobs of the power supply to 0	100	Repeat the previous three steps for remaining test points (from number 3 to number 7)
21	Disconnect the power supply cables	61	Disconnect the ground clip and probe from the oscilloscope	101	Turn off the power supply using the socket switch
22	Remove the board from the workbench using the screwdriver	62	Deactivate channel 2 of the oscilloscope	102	Set the current and voltage knobs of the power supply to 0
23	Unscrew the 4 screws on the back of the board using the screwdriver	63	Turn off the oscilloscope using the socket switch	103	Disconnect the ground clip and probe from the oscilloscope
24	Remove the display from the board	64	Remove the board from the workbench using the screwdriver	104	Deactivate channel 2 of the oscilloscope
25	Observe the soldering iron	65	Observe the soldering iron	105	Turn off the oscilloscope using the socket switch
26	Turn on the soldering iron using the socket switch	66	Turn on the soldering iron using the socket switch	106	Remove the board from the workbench using the screwdriver
27	Set the soldering iron temperature to 200 degrees using the yellow "UP" button	67	Set the soldering iron temperature to 200 degrees using the yellow "UP" button	107	Observe the soldering iron
28	Grab the black capacitor on the board with pliers	68	Grab the black capacitor on the board with pliers	108	Turn on the soldering iron using the socket switch
29	Take the soldering iron's probe	69	Take the soldering iron's probe	109	Set the soldering iron temperature to 200 degrees using the yellow "UP" button
30	Touch the first pin of the black capacitor with the soldering iron's probe for 5 seconds	70	Touch the first pin of the black capacitor with the soldering iron's probe for 5 seconds	110	Head to panel A
31	Touch the second pin of the capacitor for 5 seconds with the soldering iron's probe	71	Touch the second pin of the capacitor for 5 seconds with the soldering iron's probe	111	Observe panel A for a moment
32	Place the pliers on the workbench	72	Place the pliers on the workbench	112	Press the two red buttons on panel A
33	Place the soldering iron's probe	73	Place the soldering iron's probe	113	Head to the door
34	Place the board vertically	74	Place the board vertically	114	Exit the laboratory
35	Grab the black capacitor on the board with pliers	75	Grab the black capacitor on the board with pliers	115	Enter the laboratory
36	Take the soldering iron's probe	76	Take the soldering iron's probe	116	Pronounce the voice command "Stop" to end the recording, hear an acoustic signal for confirmation
37	Touch the first pin of the black capacitor on the back of the board for 5 seconds	77	Touch the first pin of the black capacitor on the back of the board for 5 seconds		
38	Touch the second pin of the capacitor on the back of the board for 5 seconds	78	Touch the second pin of the capacitor on the back of the board for 5 seconds		
39	Place the soldering iron's probe	79	Place the soldering iron's probe		
40	Place the board on the work area	80	Place the board on the work area		

Table 1. Low Voltage Board Repair Procedure (Standard Screwdriver Version)

pliers, high voltage board, low voltage board, low voltage board screen, register, left red button, left green button, right red button, right green button, socket 1, socket 2, socket 3, and socket 4. Figure 3 reports the object class distribution grouping them into *fixed* and *movable* objects. The dataset has been labelled manually by a group of annotators that used the VGG Image Annotator [16] with a custom project (see Figure4).

1.2.2 Utterances

Classifying intents and entities within the industrial domain can be beneficial in the development of intelligent assistants that support workers during their interactions and en-

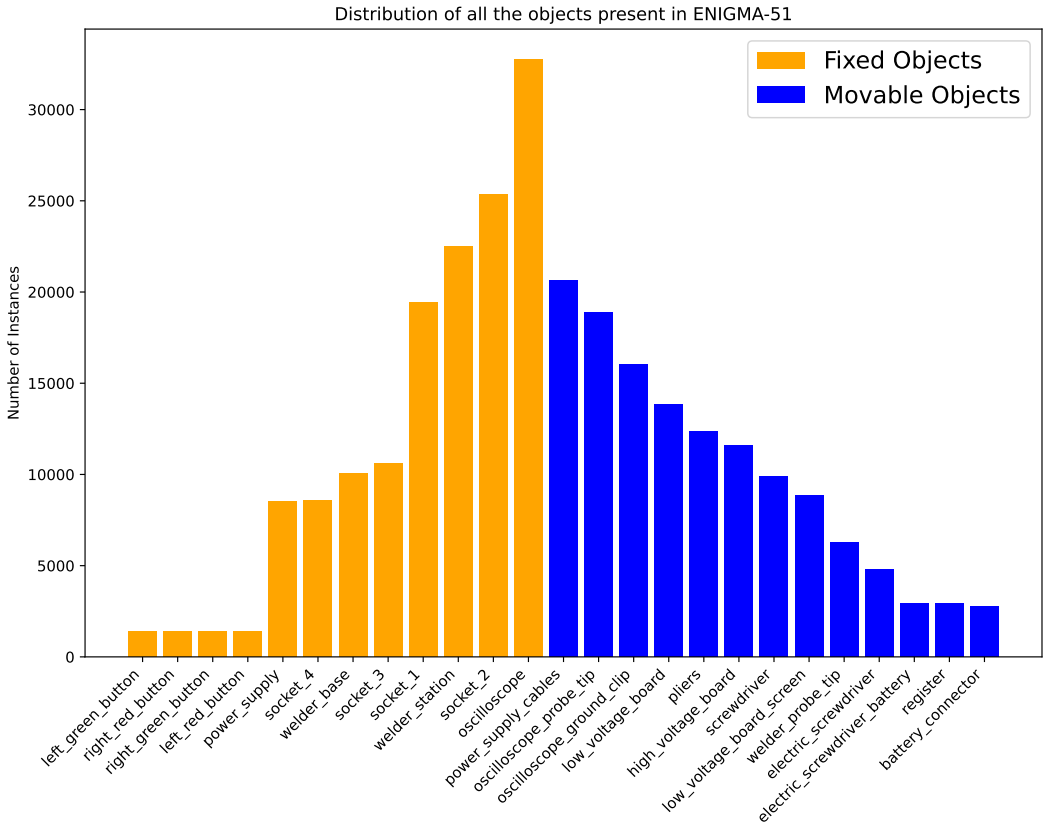

Figure 3. We report the object class distribution over the 51 videos of ENIGMA-51 grouping them into two categories: *fixed* (orange) and *movable* (blue).

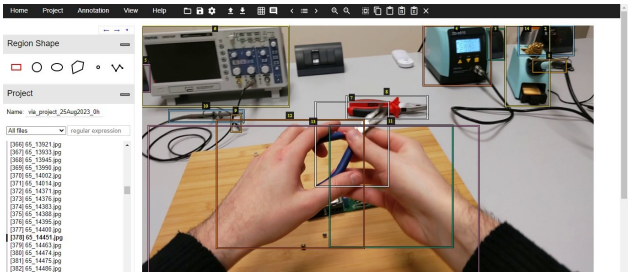

Figure 4. VGG Image Annotator tool.

sure enhanced workplace safety. Using the instructions that guided participants to acquire the ENIGMA-51 dataset, we obtained 265 textual utterances that simulate the kinds of questions a worker may have for a supervisor colleague as they carry out a procedure in an industrial setting. We manually labelled these utterances as "intents" (e.g. "object-instructions") considering a taxonomy of 24 classes and as "entities" (e.g. "object") considering 4 entity types. Table 2 reports the list of the 24 intent classes with an associated description. Each entity has been annotated using square brackets to denote its starting and ending characters in the text and round brackets to enclose the entity type. As a result, each entity is annotated following the [entity](type) form. Table 3 reports the list of the 4 considered entities.

To enrich this set of utterances, we generated similar utterances through the prompting of ChatGPT [11], obtaining 100 unique utterances for each intent. Due to the unique structure of "inform" utterances, which consist of only an entity and optionally an article, generating a set of 100 utterances was unfeasible; hence, a total of 10 utterances for the "inform" intent were produced. The "inform" intent is defined for conversations in which a worker's question cannot be adequately answered solely by performing slot filling on their initial utterance. This is often the case when some of the required entities for formulating an appropriate response are missing. For example, in the following conversation:

- Worker: *What's this object? I don't know how to use it.*
- Assistant: *Which object?*
- Worker: *The oscilloscope.*

The worker's first utterance falls under the "object-instructions" intent, whereas the worker's second utter-

Intent	Description
“greet”	Greet and start a conversation
“procedure-tutorial”	Ask a specific question about the ongoing procedure
“object-warnings”	Know if there are alerts for a specific object
“turn-object-on”	Turn on an object
“turn-object-off”	Turn off an object
“which-ppe-procedure”	Know which PPE is required to perform a specific procedure
“which-ppe-object”	Know which PPE is required to use a specific object
“object-instructions”	Know how to use a specific object
“is-object-on”	Find out if an object is turned on or off
“object-time”	Find out how long an object has been used
“where-board”	Know where a specific electronic board is located, or identify it on the working area
“board-detail”	Know the location of a component on an electronic board
“where-object”	Know where a specific object is located, or identify it on the working area
“object-detail”	Know the location of a component on an object
“start”	Start a procedure
“next”	Hear the next step in the ongoing procedure
“previous”	Hear the previous step in the ongoing procedure
“repeat”	Hear the current step in the ongoing procedure
“all-objects”	Know what objects are present in the laboratory
“ok-objects”	Know what objects can be used
“on-objects”	Know what objects are powered
“where-ppe”	Know where the PPEs are located
“inform”	Specify an entity
“out-of-scope”	This category includes all questions that are not relevant to the previous intents

Table 2. The 24 intent classes considered during our collection.

Entity	Example
“object”	[soldering iron](object)
“board’	[low voltage](board)
“component”	[display](component)
“procedure”	[repair](procedure)

Table 3. The table reports our entity taxonomy composed of 4 classes.

ance falls under the “inform” intent. Examples of utterances belonging to the inform intent include *“[high voltage](board) board [testing](procedure) procedure”* and *“[screwdriver](object)”*.

The used prompt for each intent, except for “inform” and “out-of-scope” intents, was the following: *“Imagine being an operator working inside an industrial laboratory. You can communicate with someone who knows the laboratory perfectly, including all the present objects and possible procedures that can be carried out. There are several intents you could have while operating within this industrial laboratory. This is one: <intent description>. Since you’ll have to communicate with the other person through text messages, try to avoid all forms of greeting and politeness. For this intent, imagine 100 unique sentences you would say to your interlocutor to express your intent and achieve the desired result.”* Please note that <intent description> was replaced with the description of each specific intent, using the descriptions listed in Table 2. Exceptions were made for the “inform” intent, for which we prompted the model to generate 10 unique sentences, and the “out-of-scope” intent, for which we used the following prompt: *“Imagine being an operator working inside an industrial laboratory. You can communicate with someone who knows the laboratory perfectly, including all the present objects and possible procedures that can be carried out. There are several intents you could have while operating within this industrial laboratory, which I will list below: <full list of intent descriptions>. Since you’ll have to communicate with the other person through text messages, try to avoid all forms of greeting and politeness. Knowing these intents, generate 100 unique sentences that are out of scope.”* Please note that <full list of intent descriptions> was replaced with the full list of intent descriptions listed in Table 2. Examples of the obtained utterances include: *“Provide [high voltage](board) board [repair](procedure) procedure tutorial now.”*, *“Quick status check: alerts for [battery charger](object)?”*, *“I require an image of the [display](component) that belongs to the [low voltage](board) board.”*, *“Where’s the PPE kept?”*, *“[high voltage](board)*

Figure 5. The acquired 3D models of the laboratory and some industrial objects within the set of ENIGMA-51.

board [testing](procedure) procedure" and *"I need help with my car's engine trouble; can you assist me?"* for the "procedure-tutorial" "object-warnings" "board-detail" "where-PPE", "inform", "out-of-scope" respectively.

As ChatGPT was not able to generate 100 unique utterances, we carried out additional duplicate filtering and re-prompted the model in order to generate more utterances, until we met the criteria of gathering 100 unique utterances for each intent. We hypothesize that the inability to generate a set of unique utterances is due to the many constraints expressed in our prompt, which on the other hand was designed to generate utterances that reflected the real ones collected in the same laboratory setting.

1.3. Additional Resources

1.3.1 3D models of the laboratory and objects

To enable the use of synthetic data to train scalable methods, we acquired the 3D models of the laboratory and all the 25 industrial objects. We used two different 3D scanners to create 3D models. Specifically, we used the structured-light 3D scanner Artec Eva[1] for scanning the objects, and the MatterPort[2] device to scan the industrial laboratory. Figure 5 illustrates the 3D models of the laboratory and some industrial objects within the set of ENIGMA-51. The 3D model of the laboratory weighs 30MB and covers an area of approximately 20 square meters, instead, the weight of the object's 3D model varies from 5 to 20 MB.

1.3.2 Hands and Objects Segmentation using SAM-HQ

SAM-HQ [7] is an advanced extension of the Segment Anything Model (SAM [8]), designed to enhance the segmentation of complex objects. SAM originally offered impressive scaling and zero-shot capabilities, but its mask prediction quality fell short, especially with intricate structures. To address this limitation, the authors of [7] proposed HQ-SAM, which retains SAM's promptable design, efficiency, and zero-shot generalizability while accurately segmenting

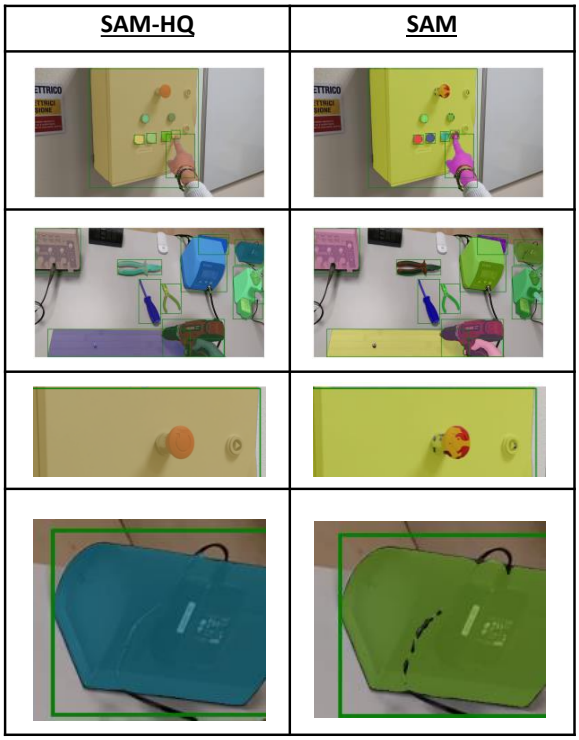

Figure 6. Comparison between SAM-HQ (left) and standard SAM (right). We reported also the bounding boxes (in green) used to generate segmentation masks.

any object. Considering the challenging nature of the industrial objects of the ENIGMA-51, we opted to use SAM-HQ as it proves to be a suitable solution for accurate segmentation. Figure 6 shows a comparison between SAM and SAM-HQ showing the better accuracy of the segmentation masks generated by SAM-HQ for wires and small buttons.

Implementation details: For the mask extraction, we used the SAM-HQ code provided in the official repository[3]. We used the bounding-box annotations from the ENIGMA-51 dataset to prompt SAM-HQ, which enabled the generation of the desired masks. The checkpoint file "sam_hq_vit_h.pth", pretrained on HQSeg-44K [7], was used for the model. During the inference phase, SAM-HQ generated a total of 55,427 hand masks and 270,519 object masks. The inference process required 6 hours using an NVIDIA A30 GPU. The semantic masks have been organized in structured JSON files, and they are released with the ENIGMA-51 dataset.

1.3.3 Hand keypoints using MMPose

Since the hands represent the channel with which humans interact with the objects, we extracted hand keypoints us-

[1] https://www.artec3d.com/portable-3d-scanners/artec-eva

[2] https://matterport.com/

[3] https://github.com/SysCV/sam-hq

ing the MMPose [3] framework with the aim of releasing pseudo-labels useful to study human-object interactions with the proposed ENIGMA-51 dataset. MMPose [3] is a useful open-source toolbox based on PyTorch, serving as part of the OpenMMLab project able to simultaneously detect the hands and localize their 2D keypoints.

Implementation detail: We used the code provided in the official repository[4]. Since MMPose requires an input hand box, we used our hand annotations. We employed the pretrained "onehand10k" model, which has been trained on images belonging to the Onehand10K [18] dataset with a resolution of 256x256. The model outputs keypoints for each hand, and each keypoint is associated with a confidence score ranging from 0 to 1. The confidence score allows us to filter out keypoints with lower accuracy. We saved all the extracted information in a JSON file. In total, we processed 30,747 left-hand bounding boxes and 24,680 right-hand bounding boxes using the MMPose framework. Figure 7 shows some examples of 2D hand keypoints extracted with MMPose.

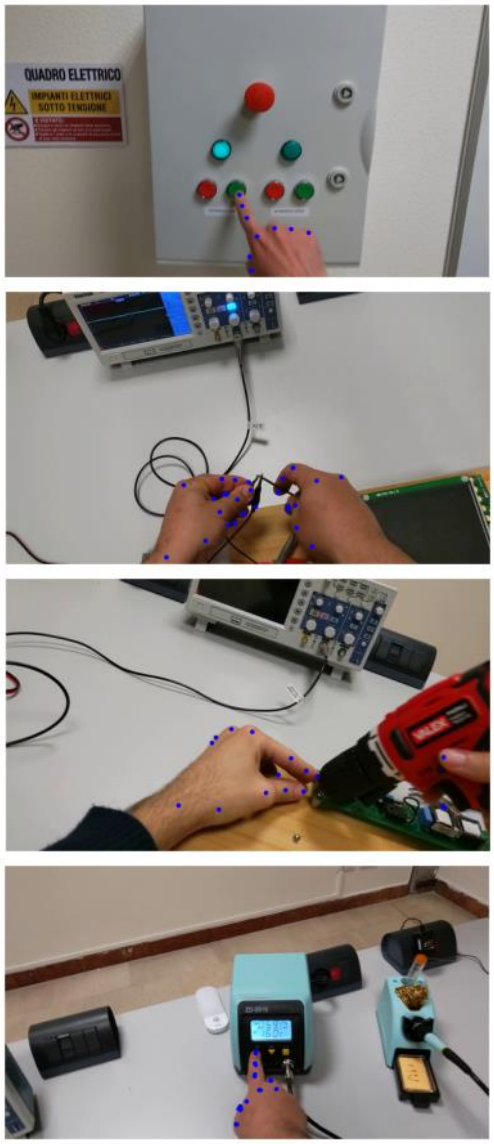

Figure 7. Hand Keypoints extracted with MMPose.

1.3.4 Features extraction using DINOv2

DINOv2 [12] is a family of foundation models that produce universal features suitable for both image-level visual tasks (such as image classification, instance retrieval, and video understanding) and pixel-level visual tasks (including depth estimation and semantic segmentation).

Implementation detail: We used the official implementation [5] with the publicly available dinov2_vitg14 pre-trained model. Image preprocessing involved a transformation pipeline consisting of resizing and centre cropping the images to a resolution of 224x224, followed by converting them to tensors and applying normalization with mean and standard deviation values of ImageNet. Each frame was then processed using the model, obtaining a tensor of size (1, 1536). The output tensors representing the extracted features were saved in *.npy* format and they will be released with the ENIGMA-51 dataset.

1.3.5 Features extraction using CLIP

To provide a set of features allowing further analysis and the study of downstream tasks with the ENIGMA-51 dataset, we exploited CLIP [13] to extract text-image representations. We also used these features to explore human-object interactions with foundational models trained with generic and diverse data and without domain-specific data. CLIP [13] is an advanced method for image representation learning from natural language supervision. It involves joint training of image and text encoders to predict correct pairings of (image, text) training examples. CLIP's architecture includes a simplified version of ConVIRT [20] trained from scratch, allowing for efficient and effective image representation learning.

Implementation details: We used the public implementation available to the following GitHub repository: https://github.com/moein-shariatnia/OpenAI-CLIP. The pretrained *ViT-L/14@336px* model has been used, and the images were processed through a specific preprocessing step composed of resizing, center cropping, tensor transformation, and normalization. The output of CLIP is a tensor of size (1, 768) which is saved in *.npy* format. All the extracted features will be released with the ENIGMA-51

[4] https://github.com/open-mmlab/mmpose
[5] https://github.com/facebookresearch/dinov2

dataset.

2. Benchmark and Baselines Details

2.1. Untrimmed Temporal Detection of Human-Object Interactions

Starting from the manually labeled timestamp of a key interaction, we defined the ground truth interaction temporal boundaries to employ our baseline based on ActionFormer [19]. We tested two different strategies to set the interaction temporal boundaries. The first consists of setting the start and end boundaries 15 frames before and after the labeled timestamp, respectively. Instead, in the second approach we set the action start at labeled interaction timestamp and we determined the action end empirically, allowing the model to observe hand movements after the labeled interaction timestamp for "take" and "release" actions. For "first contact" and "de-contact" interactions, the action end time was set 15 frames prior to the annotated timestamp.

The second approach demonstrated a better consistency in comparison to the first, despite yielding comparable mp-mAP scores during evaluations of "take" and "release" interactions (41.45% versus 42.27%). In particular, when applying a temporal threshold of 1 second, the second strategy yielded a p-mAP of 27.40%, distinctly outperforming the 11.25% achieved by the first. This observation highlights that although both methods produce comparable results, the second approach has an advantage in setting the action at the labeled timestamp. This ensures reliable performance even at lower time thresholds.

Implementation Details: We have used a Two-Stream (TS) network [17] to extract video features. Each video chunk is set to a size of 6, and there is no overlapping between adjacent chunks. With a video frame rate of 30, we get 5 chunks per second. For appearance features, we extract data from the Flatten 673 layer of ResNet-200 [5] from the central frame of each chunk. Motion features are extracted from the global pool layer of BN-Inception [6] from optical flow fields computed from the 6 consecutive frames within each chunk. Motion and appearance features are then concatenated. We used models pre-trained on ActivityNet to extract these feature vectors[6].

We tested different numbers of levels of feature pyramid and different regression ranges. We found reliable results when using 3 levels of the feature pyramid, respectively, with a regression range of [0, 2], [2, 5], [5, 10000]. We trained the model for 60 epochs using a learning rate of 0.0001, 5 warmup epochs, and a weight decay of 0.05, following a cosine scheduler. All the experiments were conducted using 4 Nvidia A30 graphics cards.

[6] https://github.com/yjxiong/anet2016-cuhk.

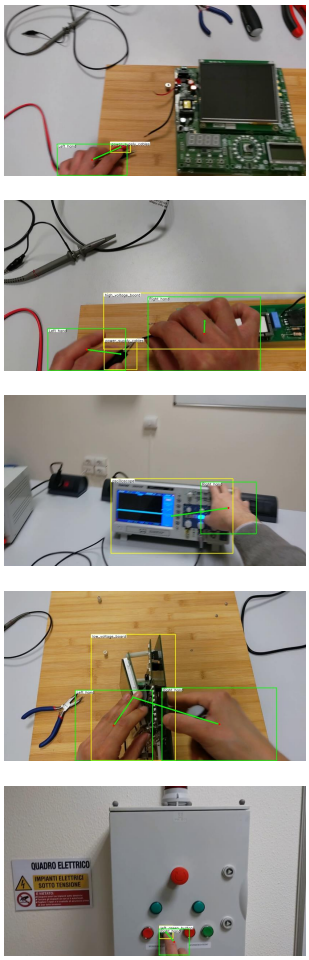

Figure 8. Qualitative results of the adopted baseline for the EHOI detection task.

2.2. Egocentric Human-Object Interaction Detection

To perform the experiments for the EHOI detection task using the baseline based on [9], we used a machine with a single *NVIDIA A30* GPU and an *Intel Xeon Silver 4310* CPU. We scaled all the images to a resolution of 1280x720 pixels. We trained the model using the *Stochastic Gradient Descent* (SGD) for 80,000 iterations, an initial learning rate of 0.001, which is decreased by a factor of 10 after 40,000 and 60,000 iterations, and a minibatch size of 4 images. Figure 8 shows qualitative results of the adopted baseline. These qualitative results provide insights into the importance of incorporating domain-specific data during the training phase to extract objects knowledge useful to provide services to workers in the industrial domain.

2.3. Short-Term Object Interaction Anticipation

We achieve the short-term object interaction anticipation task with our baseline based on [14]. Figure 9 shows some qualitative results. In the first row we reported correct predictions, while in the second one we reported wrong predictions. The predictions are represented with the green bounding boxes reporting the score, the noun and verb classes and the TTC, while the ground truth is shown in red with the name and verb classes and the TTC.

Implementation Details: At training time, to obtain high-resolution images and low resolution videos, we used the same parameters used in [14]. At test time, we feed to the networks still images of height $H = 800$ pixels and videos of height $h = 256$ pixels. The 2D backbone of the still branch is a ResNet-50 architecture. The weights of this backbone and the ones of the standard feature pyramid layer are initialized from a Faster R-CNN model [15] pre-trained on the COCO dataset [10]. The 3D network which composes the fast branch is an X3D-M model [4] pre-trained on Kinetics [2]. The model has been trained with a base learning rate of 0.001 and a weight decay of 0.0001. The learning rate is lowered by a factor of 10 after 15 and 30 epochs. The model is trained in half precision on four NVIDIA V100 GPUs with a batch size of 32.

2.4. Natural Language Understanding of Intents and Entities

We split our real dataset using an 80:20 ratio, with an identical test split employed across all experiments, which is uniformly distributed across intent and entity classes.

DIETClassifier [1] was adopted for intent and entity prediction. The model has been trained on an Intel Core i5 CPU for 100 epochs with a learning rate of 0.001 and a variable batch size which linearly increases for each epoch from 64 to 256.

Table 4 reports the results for the intent classification task (first four columns) and for the entity classification task (last four columns). Five different variations of the training set were explored: real data, real data + G10 data, real data + G50 data, real data + G100 data, and G100; and four different metrics were used for both intent and entity classification task: accuracy, precision, recall, and F1-score. The best results for the intent classification task have been obtained using only real data for training, with an accuracy, precision, recall, and F1-score of 0.867, 0.840, 0.867, and 0.844 respectively. However, using only generated data (G100) for training leads to poorer performances, with an accuracy, precision, recall, and F1-score of 0.584 (-0.283), 0.622 (-0.218), 0.584 (-0.283), 0.564 (-0.280) respectively. These results, in conjunction with the difficulties encountered during the prompting process, suggest that generated data does not fully reflect real utterances, and modern generative models may not accommodate all the constraints imposed by our specific context, thus affecting the model's performance. Exploring the use of combinations of real and generated data for training, we observe the best performances when the generated data is not predominant over the real data. In fact, we obtained an accuracy, precision, recall, F1-score of 0.830 (-0.037), 0.822 (-0.018), 0.830 (-0.037), and 0.815 (-0.029) respectively using real data and G10 for training, an accuracy, precision, recall, F1-score of 0.792 (-0.075), 0.788 (-0.052), 0.792 (-0.075), 0.773 (-0.071) respectively using real data and G50 for training, and an accuracy, precision, recall, F1-score of 0.792 (-0.075), 0.794 (-0.046), 0.792 (-0.075), 0.784 (-0.060) respectively using real data and G100. The performance deteriorates significantly with the addition of G10 to real data, subsequently showing a gradual decline as more generated data is introduced, ultimately stabilizing as G100 is added.

Regarding entity classification, the results show that generated data alone is able to represent entities as how they're encountered in the real setting. In fact, best results are attained using G100 alone for training with an accuracy, precision, recall, and F1-score of 1.0, 1.0, 1.0, 1.0 respectively. However, real data itself attains near-perfect performances, with an accuracy, precision, recall, and F1-score of 0.994 (-0.006), 0.965 (-0.035), 1.0 ($\pm$0), 0.981 (-0.019) respectively.

Figure 10 presents qualitative results for three distinct utterances for both intent and entity classification tasks. In the top row, we observe an utterance with an incorrect intent prediction ("object-instructions" instead of "object-warnings") with a confidence of 0.32. This utterance did not contain any entities, and the absence of entities was correctly predicted with a confidence of 0.98. These results suggest that our model exhibited uncertainty in determining the appropriate class for the utterance. This uncertainty could be attributed to the fact that both "object-instructions" and "object-warnings" often contain utterances formulated in a very similar manner. However, the model's capabilities concerning entity classification tend to be highly accurate with a significant confidence score. Moving to the middle row, we encounter an utterance with an incorrect intent prediction ("inform" instead of "where-object") with a confidence score of 0.94. This utterance contained an entity of the "object" type, and the presence and class of this entity were correctly predicted with a confidence of 0.91. These results suggest that our model exhibited a high level of confidence in classifying the utterance, despite its incorrect classification. This observation may indicate that this particular utterance shares significant similarities with those typically found in the "where-object" intent. Similarly, in this instance, the model's capabilities enable accurate entity classification with a high confidence score. Lastly, the bottom row showcases an utterance with a correct intent prediction ("which-PPE-procedure") with a confidence score

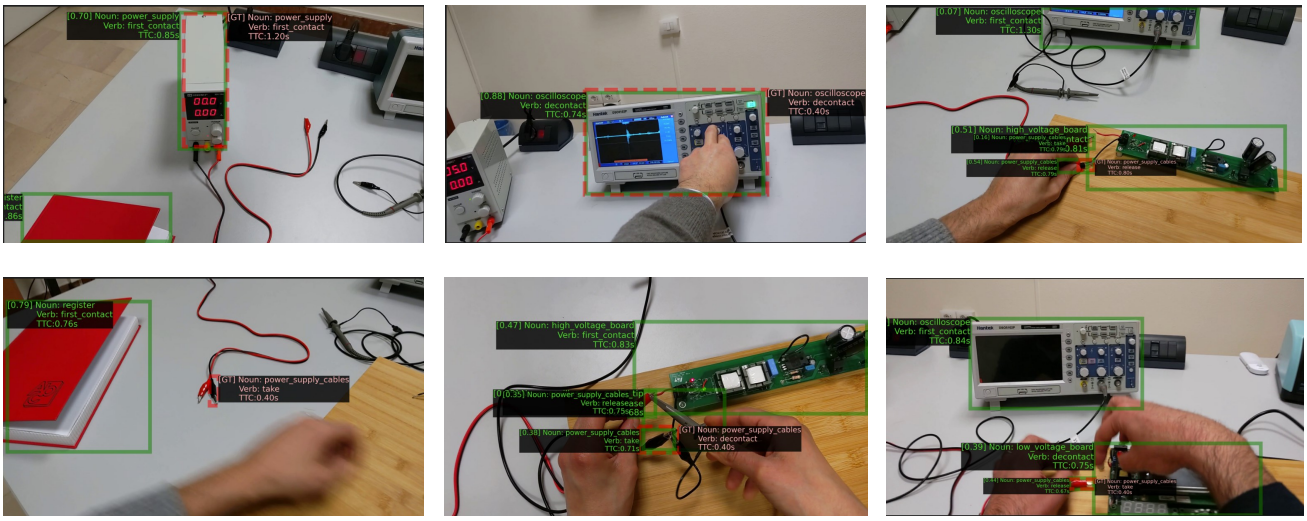

Figure 9. We reported qualitative results of our baseline based on StillFast [14] for the short-term object interaction anticipation task.

	GROUND TRUTH INTENT	PREDICTED INTENT	CONF.	GROUND TRUTH ENTITIES	PREDICTED ENTITIES	CONF.
should I know more about this object?	object-warnings	object-instructions	0.32	none	none	0.98
point me to the *panel*	where-object	inform	0.94	*panel*	*panel*	0.91
which PPE are required for the *repair* of the *low voltage* board?	which-PPE-procedure	which-PPE-procedure	0.91	*repair*, *low voltage*	*repair*, *low voltage*	0.74, 0.96

Figure 10. Qualitative results showing two incorrect intent predictions (first two rows) and a correct prediction (last row), alongside correct entity predictions.

	Intent				Entity			
Training	**Accuracy**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**
real	**0.867**	**0.840**	**0.867**	**0.844**	0.994	0.965	1.0	0.981
real+G10	0.830	0.822	0.830	0.815	1.0	1.0	1.0	1.0
real+G50	0.792	0.788	0.792	0.773	1.0	1.0	1.0	1.0
real+G100	0.792	0.794	0.792	0.784	1.0	1.0	1.0	1.0
G100	0.584	0.622	0.584	0.564	**1.0**	**1.0**	**1.0**	**1.0**

Table 4. Results for intents and entities classification considering different sets of training data.

of 0.91. This utterance featured entities of both "procedure" and "board" types, and the presence and classes of these entities were correctly predicted with confidence scores of 0.74 and 0.96, respectively. These results suggest that our model exhibited a high level of confidence in classifying the utterance, and its classification was indeed correct. Ultimately, in this latter scenario as well, the model's capabilities enable accurate entity classification, resulting in a confidence score ranging from moderate to high.

ACKNOWLEDGEMENTS

This research is supported by Next Vision[7] s.r.l., by MISE - PON I&C 2014-2020 - Progetto ENIGMA - CUP: B61B19000520008, and MIUR AIM Linea 1 - AIM1893589 - CUP: E64118002540007, and by the project FAIR – PNRR MUR Cod. PE0000013 - CUP: E63C22001940006.

References

[1] Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. Diet: Lightweight language understanding for dialogue systems. *arXiv preprint arXiv:2004.09936*, 2020. 9

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724—-4733, 2017. 9

[3] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020. 7

[4] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 200–210, 2020. 9

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 8

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 448–456. pmlr, 2015. 8

[7] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. *arXiv:2306.01567*, 2023. 6

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 6

[9] Rosario Leonardi, Francesco Ragusa, Antonino Furnari, and Giovanni Maria Farinella. Exploiting multimodal synthetic data for egocentric human-object interaction detection in an industrial scenario. *arXiv preprint arXiv:2306.12152*, 2023. 8

[10] T. Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2014. 9

[11] Openai chatgpt. https://openai.com/blog/chatgpt. 4

[12] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 7

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 7

[14] Francesco Ragusa, Giovanni Maria Farinella, and Antonino Furnari. Stillfast: An end-to-end approach for short-term object interaction anticipation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023. 9, 10

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Neural Information Processing Systems (NIPS)*, 28, 2015. 9

[16] Vgg image annotator. https://www.robots.ox.ac.uk/~vgg/software/via/. 3

[17] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018. 8

[18] Yangang Wang, Cong Peng, and Yebin Liu. Mask-pose cascaded cnn for 2d hand pose estimation from single color image. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(11):3258–3268, 2019. 7

[19] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 13664 of *LNCS*, pages 492–510, 2022. 8

[20] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. Contrastive learning of medical visual representations from paired images and text, 2022. 7

[7] Next Vision: https://www.nextvisionlab.it/