MOMA-LRG

Release 0.1

The MOMA team

Dec 28, 2022

CONTENTS

The **Multi-Object, Multi-Actor Dataset with Language Refined Graphs (MOMA-LRG)** is an novel benchmark designed to develop highly general and interpretable video understanding models.

The dataset is a research project from the Stanford Vision and Learning Lab.

Note: The dataset and its API are currently under active development. For the source code, please navigate to this GitHub repository.

CHAPTER

ONE

GETTING STARTED

1.1 Installation

To install the MOMA API, first clone the repository by running

```
git clone https://github.com/StanfordVL/moma.git
```

Install the API and its dependencies by running

```
cd moma
pip install .
pip install -r requirements.txt
```

1.2 Retrieving the dataset

The dataset will can be downloaded by following the instructions from the MOMA website.

Download the dataset into a directory titled `dir_moma` with the structure below. The anns directory requires roughly 1.8GB of space and the video directory requires 436 GB.

```
$ tree dir_moma
.
├── anns
│   ├── anns.json
│   ├── split_std.json
│   ├── split_fs.json
│   ├── clips.json
│   └── taxonomy
└── videos
    ├── all
    ├── raw
    ├── activity_fr
    ├── activity
    ├── sub_activity_fr
    ├── sub_activity
    ├── interaction
    ├── interaction_frames
    └── interaction_video
```

CHAPTER

TWO

EXAMPLE USAGE

2.1 Creating a MOMA Object

MOMA-LRG exports a simple to use API that allows users to access the dataset via an easy-to-use interface.

To begin, create a MOMA-LRG object by passing in the path to the MOMA-LRG dataset as follows:

```
import momaapi
dir_moma = "my/moma/directory"
moma = momaapi.MOMA(dir_moma)
```

Getting access to the underlying data can be done via calling methods on the `moma` object.

2.2 Few-shot experiments

MOMA-LRG was designed to provide an abstraction to learn highly generalizable video representations. As a result, the MOMA-LRG API supports a few-shot paradigm where different splits have non-overlapping activity classes and sub-activity classes. This is in contrast to the standard evaluation paradigm, where different splits share the same sets of activity classes and sub-activity classes.

To evaluate on few-shot, create a MOMA object by running:

```
moma = momaapi.MOMA(dir_moma, paradigm='few-shot')
```

The interface is the same as in the standard paradigm.

2.3 Working with the data

After creating a MOMA object, you can interface with the dataset through a very simple API. Let's say that you wanted to retrieve the annotations for all videos containing the activity class `"basketball game"` in the validation set. You could run

```
ids_act = moma.get_ids_act(split="val", cnames_act=["basketball game"])
anns_act = moma.get_anns_act(ids_act)
```

`anns_act` now contains a list of Activity annotations, each containing metadata on a different instance of a basketball game.

CHAPTER

THREE

MOMA

class momaapi.moma.**MOMA**(*dir_moma: str*, *paradigm: typing_extensions.Literal[standard, few - shot] = 'standard'*, *reset_cache: bool = False*)

Class to interface with the MOMA-LRG dataset. Initialization requires passing in a directory containing the MOMA-LRG dataset.

The MOMA object can be used for few-shot experiments, which reduces the number of classes and examples, or used in the standard paradigm.

The following conventions are used throughout the documentation as shorthand:

- `act`: activity
- `sact`: sub-activity
- `hoi`: higher-order interaction
- `entity`: entity
- `ia`: intransitive action
- `ta`: transitive action
- `att`: attribute
- `rel`: relationship
- `ann`: annotation
- `id`: instance ID
- `cname`: class name
- `cid`: class ID

Parameters

- **dir_moma** (*str*) – directory containing the MOMA dataset
- **paradigm** (*Literal['standard', 'few-shot']*) – the experiment configuration, which is either `'standard'` or `'few-shot'`
- **reset_cache** (*bool*) – flag that indicates whether to reset cached data
- **taxonomy** (Taxonomy) – a Taxonomy object containing information about the dataset taxonomy
- **lookup** (Lookup) – a Lookup object containing information about class IDs and class names
- **statistics** – a Statistics object that can generate dataset-level statics
- **num_classes** (*int*) – the number of classes contained in the MOMA object

`get_anns_act`(*ids_act: list*) → list

Given activity instance IDs, return their annotations

Parameters
`ids_act` – activity instance IDs

Returns
annotations for the given activity instance IDs

Return type
list

`get_anns_hoi`(*ids_hoi: list*) → list

Given higher-order interaction instance IDs, return their annotations

Parameters
`ids_hoi` – higher-order interaction instance IDs

Returns
annotations for the given higher-order interaction instance IDs

Return type
list

`get_anns_sact`(*ids_sact: list*) → list

Given sub-activity instance IDs, return their annotations

Parameters
`ids_sact` – sub-activity instance IDs

Returns
annotations for the given sub-activity instance IDs

Return type
list

`get_cids`(*kind: typing_extensions.Literal[act, sact, actor, object, ia, ta, att, rel]*, *threshold: int*, *split: typing_extensions.Literal[train, val, test, either, all, combined]*) → list

Parameters

- `kind` (*Literal['act', 'sact', 'actor', 'object', 'ia', 'ta', 'att', 'rel']*) – the kind of annotations needed to be retrieved
- `threshold` (*int*) – exclude classes with fewer than this number of total instances
- `split` (*Literal['train', 'val', 'test', 'either', 'all', 'combined']*) – the split to be used for the retrieval. Here, `train` refers to the training set, `val` refers to the validation set, and `test` refers to the test set. `either` will exclude a class if the smallest number of instances in across splits is less than the threshold, *all* will exclude a class if the largest number of instances in across splits is less than the threshold, and `combined` will exclude a class if the smallest number of instances in across splits is less than the threshold

Returns
a list of class IDs

Return type
List[int]

`get_clips`(*ids_hoi: list*) → list

Given higher-order interaction instance IDs, return their clips

Parameters
ids_hoi – higher-order interaction instance IDs

Returns
clips for the given higher-order interaction instance IDs

Return type
list

get_cnames(*cids_act: Optional[list] = None*, *cids_sact: Optional[list] = None*, *cids_actor: Optional[list] = None*, *cids_object: Optional[list] = None*, *cids_ia: Optional[list] = None*, *cids_ta: Optional[list] = None*, *cids_att: Optional[list] = None*, *cids_rel: Optional[list] = None*) → list

Returns the associated class names given the class IDs.

Parameters

- **cids_act** (*Optional[List[int]]*) – a list of class IDs of activities
- **cids_sact** (*Optional[List[int]]*) – a list of class IDs of sub-activities
- **cids_actor** (*Optional[List[int]]*) – a list of class IDs of actors
- **cids_object** (*Optional[List[int]]*) – a list of class IDs of objects
- **cids_ia** (*Optional[List[int]]*) – a list of class IDs of intransitive actions
- **cids_ta** (*Optional[List[int]]*) – a list of class IDs of transitive actions
- **cids_att** (*Optional[List[int]]*) – a list of class IDs of attributes
- **cids_rel** (*Optional[List[int]]*) – a list of class IDs of relationships

Returns
a list of class names

Return type
List[str]

get_ids_act(*split: Optional[str] = None*, *cnames_act: Optional[list] = None*, *ids_sact: Optional[list] = None*, *ids_hoi: Optional[list] = None*) → list

Get the unique activity instance IDs that satisfy certain conditions

Parameters

- **split** (Union['train', 'val', 'test', 'either', 'all', 'combined']) – get activity IDs that belong to the given dataset split
- **cnames_act** (*list*) – get activity IDs that belong to the given activity classes
- **ids_sact** (*list*) – get activity IDs for given sub-activity IDs
- **ids_hoi** (*list*) – get activity IDs for given higher-order interaction IDs [ids_hoi]

Returns
a list of activity IDs

Return type
list

get_ids_hoi(*split: Optional[str] = None*, *ids_act: Optional[list] = None*, *ids_sact: Optional[list] = None*, *cnames_actor: Optional[list] = None*, *cnames_object: Optional[list] = None*, *cnames_ia: Optional[list] = None*, *cnames_ta: Optional[list] = None*, *cnames_att: Optional[list] = None*, *cnames_rel: Optional[list] = None*) → list

Get the unique higher-order interaction instance IDs that satisfy certain conditions dataset split

Parameters

- **split** (Union['train', 'val', 'test', 'either', 'all', 'combined']) – get higher-order interaction IDs [ids_hoi] that belong to the given dataset split
- **ids_act** (*list*) – get higher-order interaction IDs [ids_hoi] for given activity IDs [ids_act]
- **ids_sact** (*list*) – get higher-order interaction IDs [ids_hoi] for given sub-activity IDs [ids_sact]
- **cnames_actor** (*list*) – get higher-order interaction IDs [ids_hoi] for given actor class names [cnames_actor]
- **cnames_object** (*list*) – get higher-order interaction IDs [ids_hoi] for given object class names [cnames_object]
- **cnames_ia** (*list*) – get higher-order interaction IDs [ids_hoi] for given intransitive action class names [cnames_ia]
- **cnames_ta** (*list*) – get higher-order interaction IDs [ids_hoi] for given transitive action class names [cnames_ta]
- **cnames_att** (*list*) – get higher-order interaction IDs [ids_hoi] for given attribute class names [cnames_att]
- **cnames_rel** (*list*) – get higher-order interaction IDs [ids_hoi] for given relationship class names [cnames_rel]

get_ids_sact(*split: Optional[str] = None*, *cnames_sact: Optional[list] = None*, *ids_act: Optional[list] = None*, *ids_hoi: Optional[list] = None*, *cnames_actor: Optional[list] = None*, *cnames_object: Optional[list] = None*, *cnames_ia: Optional[list] = None*, *cnames_ta: Optional[list] = None*, *cnames_att: Optional[list] = None*, *cnames_rel: Optional[list] = None*) → list

Get the unique sub-activity instance IDs that satisfy certain conditions dataset split

Parameters

- **split** (Union['train', 'val', 'test', 'either', 'all', 'combined']) – get sub-activity IDs [ids_sact] that belong to the given dataset split
- **cnames_sact** (*list*) – get sub-activity IDs [ids_sact] for given sub-activity class names [cnames_sact]
- **ids_act** (*list*) – get sub-activity IDs [ids_sact] for given activity IDs [ids_act]
- **ids_hoi** (*list*) – get sub-activity IDs [ids_sact] for given higher-order interaction IDs [ids_hoi]
- **cnames_actor** (*list*) – get sub-activity IDs [ids_sact] for given actor class names [cnames_actor]
- **cnames_object** (*list*) – get sub-activity IDs [ids_sact] for given object class names [cnames_object]
- **cnames_ia** (*list*) – get sub-activity IDs [ids_sact] for given intransitive action class names [cnames_ia]
- **cnames_ta** (*list*) – get sub-activity IDs [ids_sact] for given transitive action class names [cnames_ta]
- **cnames_att** (*list*) – get sub-activity IDs [ids_sact] for given attribute class names [cnames_att]
- **cnames_rel** (*list*) – get sub-activity IDs [ids_sact] for given relationship class names [cnames_rel]

Returns
a list of sub-activity IDs

Return type
list

`get_metadata`(*ids_act: list*) → list

Get the metadata for the given activity IDs. The metadata returned is that associated with the raw videos that contain instances of the activity IDs.

Parameters
ids_act – get metadata for the given activity IDs

Returns
video metadata for the given activity ID

Return type
list

`get_paths`(*ids_act: Optional[list] = None*, *ids_sact: Optional[list] = None*, *ids_hoi: Optional[list] = None*, *id_hoi_clip: Optional[str] = None*, *full_res: bool = False*, *sanity_check: bool = True*) → list

Given activity, sub-activity, higher-order interaction, or clip IDs, return the paths to the videos.

Parameters

- **ids_act** (*list*) – activity instance IDs
- **ids_sact** (*list*) – sub-activity instance IDs
- **ids_hoi** (*list*) – higher-order interaction instance IDs
- **id_hoi_clip** (*str*) – clip ID
- **full_res** (*bool*) – return full-resolution videos
- **sanity_check** (*bool*) – check that the video exists

Returns
paths to the videos

Return type
list

`is_sact`(*id_act: int*, *time: int*, *absolute: bool = False*) → bool

Checks whether a certain time in an activity has a sub-activity.

Parameters

- **id_act** (*int*) – activity ID
- **time** (*int*) – time in the activity
- **absolute** (*bool*) – relative to the full video if `True` or relative to the activity video if `False`

`map_cids`(*split: typing_extensions.Literal[train, val, test, either, all, combined]*, *cids_act_contiguous: Optional[list] = None*, *cids_act: Optional[list] = None*, *cids_sact_contiguous: Optional[list] = None*, *cids_sact: Optional[list] = None*) → list

Map class IDs between standard class IDs and split-specific contiguous class IDs. **For the few-shot paradigm only**.

Parameters

- **split** (*Literal['train', 'val', 'test', 'either', 'all', 'combined']*) – the dataset split to use
- **cids_act_contiguous** (*Optional[List[int]]*) – a list of contiguous class IDs in the activity set
- **cids_act** (*Optional[List[int]]*) – a list of class IDs in the activity set
- **cids_sact_contiguous** (*Optional[List[int]]*) – a list of contiguous class IDs in the sub-activity set
- **cids_sact** (*Optional[List[int]]*) – a list of class IDs in the sub-activity set

Returns
mapping between standard class IDs and split-specific contiguous IDs

sort(*ids_sact: Optional[list] = None*, *ids_hoi: Optional[list] = None*, *sanity_check: bool = True*)

Given a list of sub-activity or higher-order interaction instance IDs, return them in sorted order by when they occured in the video.

Parameters

- **ids_sact** (*list*) – sub-activity instance IDs
- **ids_hoi** (*list*) – higher-order interaction instance IDs
- **sanity_check** (*bool*) – check that the video exists

Returns
sorted IDs

Return type
list

CHAPTER

FOUR

ANNOTATION STRUCTURE

class momaapi.data.ann.**AAct**(*info*, *entities*, *ias*, *tas*, *atts*, *rels*)

Class for an atomic action annotation. Atomic actions are unary predicates that *actors* perform.

Variables

- **id_entity** – Entity ID
- **kind_entity** – type of the entity
- **cname_entity** – Entity class name
- **cid_entity** – Entity class ID
- **start** – start time of the atomic action in seconds, relative to the start of the activity video
- **end** – end time of the atomic action in seconds, relative to the start of the activity video

class momaapi.data.ann.**Act**(*ann*, *taxonomy*)

Class for an activity annotation. An **activity** is the coarsest level of annotation, consisting of a series of subactivities that are decomposed into smaller subactivities.

Variables

- **cname** – Activity class name
- **cid** – Activity class ID
- **start** – Start time of the activity in seconds
- **end** – End time of the activity in seconds
- **ids_sact** – List of sub-activity IDs

class momaapi.data.ann.**BBox**(*ann*)

Bounding box in the form of [x, y, w, h]. These are utilized to localize entities.

Variables

- **x** – x-coordinate of the top-left corner of the bounding box
- **y** – y-coordinate of the top-left corner of the bounding box
- **w** – width of the bounding box
- **h** – height of the bounding box

class momaapi.data.ann.**Clip**(*ann*, *neighbors*)

A clip corresponds to a 1 second/5 frames video clip centered at the higher-order interaction - <1 second/5 frames if exceeds the raw video boundary - Currently, only clips from the test set have been generated

class momaapi.data.ann.**Entity**(*ann*, *kind*, *taxonomy*)

Class of an annotation of an entity. Entities are the building blocks of interactions. They are either human actors or inhuman objects.

Variables

- **id** – entity ID
- **kind** – kind of the entity, either "actor" or "object"
- **cname** – class name of the entity
- **cid** – class ID of the entity
- **bbox** – bounding box of the entity

class momaapi.data.ann.**HOI**(*ann*, *taxonomy_actor*, *taxonomy_object*, *taxonomy_ia*, *taxonomy_ta*, *taxonomy_att*, *taxonomy_rel*)

Class for a higher order interaction. A **higher-order interaction**, abbreviated as HOI, is a predicate involving *two or more entities*.

Variables

- **id** – HOI annotation ID
- **time** – time of the HOI annotation in seconds, relative to the start of the activity video
- **actors** – list of actor entities involved in the interaction
- **ias** – list of intransitive actions occuring between actors
- **tas** – list of transitive actions occuring between actors
- **atts** – list of attributes that the actor has
- **rels** – list of relationships between entities in the interaction

class momaapi.data.ann.**Metadatum**(*ann*)

Metadata class for a video. The metadata contains information for videos in the MOMA-LRG dataset, the properties of which are detailed below.

Variables

- **id** – Activity ID
- **fname** – File name of the video
- **num_frames** – Number of frames in the video
- **width** – Width of the video resolution
- **height** – Height of the video resolution
- **duration** – Duration of the video in seconds

get_fid(*time*)

Get the frame ID given a timestamp in seconds :param time: Timestamp in seconds :type time: float

class momaapi.data.ann.**Predicate**(*ann*, *kind*, *taxonomy*)

Predicate class, representing unary and binary predicates. **Predicates** are of the form `[src] (cid) [trg]`, where `src` refers to the "source entity" performing the action and `trg` to the "target entity" who is affected by the source entity.

Variables

- **kind** – kind of the predicate

- **cname** – class name of the predicate
- **id_src** – ID of the source entity
- **id_trg** – ID of the target entity

class momaapi.data.ann.**SAct**(*ann*, *scale_factor*, *taxonomy_sact*, *taxonomy_actor*, *taxonomy_object*, *taxonomy_ia*, *taxonomy_ta*, *taxonomy_att*, *taxonomy_rel*)

Class for a sub-activity class annotation. A **subactivity** is a finer grained level of annotation which refers to a step taken as part of an activity. It is temporallly localized within the activity (that is, it has a start and end time in seconds that are *relative to the start of the activity*).

Variables

- **cname** – Sub-activity class name
- **cid** – Sub-activity class ID
- **start** – Start time of the sub-activity in seconds, relative to the start of the activity video
- **end** – End time of the sub-activity in seconds, relative to the start of the activity video
- **ids_hoi** – List of higher-order interactions
- **times** – Times of higher order interactions inside the video

CHAPTER

FIVE

TAXONOMY

class momaapi.taxonomy.**Taxonomy**(*dir_moma*)

The MOMA taxonomy object is a dictionary that contains information about the MOMA hierarchy. This typically should not be used, but contains information about different levels of the MOMA hierarchy for each split of the dataset.

Printing the Taxonomy can be done via

```
from momaapi import MOMA
moma = MOMA(dir_moma)
print(moma.taxonomy)
```

CHAPTER

SIX

LOOKUP

class momaapi.lookup.**Lookup**(*dir_moma*, *taxonomy*, *reset_cache*)

Lookup utility class to help lookup annotations.

map_id(*kind*, *id_act=None*, *id_sact=None*, *id_hoi=None*)

Maps instance IDs across the MOMA hierarchy. Usage:

- **Convert an `id_act` into `ids_sact` (one-to-many):**
  `map_id(id_act=id_act, kind='sact')`
- **Convert an `id_act` into `ids_hoi` (one-to-many):**
  `map_id(id_act=id_act, kind='hoi')`
- **Convert an `id_sact` into `id_act` (one-to-one):**
  `map_id(id_sact=id_sact, kind='act')`
- **Convert an `id_sact` into `ids_hoi` (one-to-many):**
  `map_id(id_sact=id_sact, kind='hoi')`
- **Convert an `id_hoi` into `id_sact` (one-to-one):**
  `map_id(id_hoi=id_hoi, kind='sact')`
- **Convert an `id_hoi` into `id_act` (one-to-one):**
  `map_id(id_hoi=id_hoi, kind='act')`

retrieve(*kind*, *key=None*)

Accesses the value given a key. There are several different ways to retrieve:

- **Convert a `split` into `ids_act` (one-to-many):**
  `retrieve(kind='id_act', key=split)`
- **Convert an `id_act` into an `ann_act`, metadatum (one-to-one):**
  `retrieve(kind='ann_act' or 'metadatum', key=id_act)`
- **Convert an `id_sact` into an `ann_sact` (one-to-one):**
  `retrieve(kind='ann_sact', key=id_sact)`
- **Convert an `id_hoi` into an `ann_hoi` or a `clip` (one-to-one):**
  `retrieve(kind='ann_hoi' or 'clip', key=id_hoi)`

Parameters

kind (*Literal["paradigms","splits","ids_act","ids_sact","ids_hoi","anns_act","metadata","anns_sact","anns_hoi","clips",]*) – indicates the type of retrieval that is used

PYTHON MODULE INDEX

m

INDEX