Visual Acoustic Discriminative Relevance (VADR)

Demo for DAIS ITA Annual Fall Meeting 2019

Version: 1.1

Dave Braines, Harri Taylor, Liam Hiley, Alun Preece

Last updated: 20th Aug 2019

[System details 3](#_Toc16510894)

[Terminology 3](#_Toc16510895)

[Event types 3](#_Toc16510896)

[Activities 4](#_Toc16510897)

[Trigger words and narrative 4](#_Toc16510898)

[Demonstration roles 5](#_Toc16510899)

[System components 5](#_Toc16510900)

[Hardware and feed components 5](#_Toc16510901)

[Analytics components 5](#_Toc16510902)

[Service components 6](#_Toc16510903)

[Wireframe layout 7](#_Toc16510904)

[Demonstration details 8](#_Toc16510905)

[References 9](#_Toc16510906)

[Appendices 10](#_Toc16510907)

[Appendix A – Activity 101 classes 10](#_Toc16510908)

[Appendix B – Speech commands dataset classes 11](#_Toc16510909)

[Appendix C – Example JSON response formats 12](#_Toc16510910)

[Video-explanation service JSON 12](#_Toc16510911)

[Inference-explanation service JSON 12](#_Toc16510912)

[Audio-explanation service JSON 12](#_Toc16510913)

[Audio-speech-to-text service JSON 12](#_Toc16510914)

[List-events service JSON 12](#_Toc16510915)

[Appendix D – Logical inference rules 12](#_Toc16510916)

From Alun: My first impression is that the terminology of “event” and “activity” is a bit confusing /distracting- I don’t see why we need both terms? Could we have something like “raw event” and “target event” (instead of what you’ve called “activity”) or even “raw activity” and “target activity”? I’m inclined to prefer the latter (“raw activity” / “target activity”) because the field tends to be known as “activity recognition”.

From Liam: Here are some things I've noticed so far:

* You've referred to the UCF-101 dataset as Activity 101.
* I thought we were obscuring the link between AR classes and our events (i.e. PizzaTossing -> Hooray)? I don't necessarily have an argument for doing this other than participants nitpicking what activities "should" be mapped.
* RE your comment in References, I definitely think we should list key papers, lest we get bogged down on all the moving parts in this haha.
* Building on this, would it be useful to have a short entry in this document vaguely defining DR?
* Here's the UCF-101 link for Appendices: [https://www.crcv.ucf.edu/data/UCF101.php](https://urldefense.proofpoint.com/v2/url?u=https-3A__www.crcv.ucf.edu_data_UCF101.php&d=DwMGaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=swmOA_Pyl4SqADw3EEEeejy_9cZb8HHfh1LHvuMN9CU&m=sd5EgjKsA0RW5AiMPBCNENdBicQR363oXIx3pd5AeKk&s=f6gMG6QaaPp5kyLVASKUR7GO2iimm4OHBBGfiNY1pag&e=)
* In response to Alun's suggestion, I view the activity as the identifier for the live stream the model is constantly updating, which is chosen from a predefined vocabulary. We then detect events when the explanation for the activity meets some criteria we define.

Sorry if I've missed anything! Looks good so far, and I love the wireframe. If I could also ask quickly, at the moment I can't get a val. acc. higher than 75% for the model. Do you want me to pursue this as a main concern, as it's probably not worth looking for our activities until we've settled on a model right? Alun, this is also the model I'll be using for AAAI, so I'd appreciate your guidance on this.

From Harri: Update for today from me: preliminary audio recognition model (VGG-16) is getting 50% accuracy within 30 mins of training (on all 32 words). I am yet to augment the data with background noise. As soon as we know what words we are definitely classifying for I will lock that in, as the model can be trained on a much more restrictive vocabulary for more reliable results (in the SOTA paper 98% accuracy is achieved using a 12:20 class:unknown split).

# System details

*(note to co-authors: in this version of the document all names, components, diagrams, storylines etc are able to be easily changed. It’s important that we get things right as early as possible so if you see terminology or descriptions that you don’t like or don’t feel are right please do feedback right away. It’s better that we iterate through the design and get good agreement rather than build things and then share our concerns later ☺)*

The VADR (Visual Acoustic Discriminative Relevance) system shows subsymbolic multi-modal (video and audio) analytics fused with symbolic reasoning to provide a variety of explanations for detected events. The events are detected in the video stream and have a correlated audio stream which is analysed to provide contextual information for the detected event. Human participants are guided to perform certain actions in from of a live video feed upon hearing certain trigger words within a narrative text read by one of the experimenters. The live video feed is being constantly analysed for real-time event detection and should detect the actions being made by the human participants. Such detection causes system events which integrate with symbolic reasoning to potentially infer additional information. The events and any inferences are shown in a simple user interface. For any detected events the user interface can be used to obtain an explanation of the events and any inferences, with the explanations spanning the sub-symbolic and symbolic layers.

This document describes the system that is required to achieve the outcome described above. This document focuses specifically on the demonstration of the VADR system for the Annual Fall Meeting (AFM) in September 2019, but by building for this demonstration the VADR system will be able to be used for other purposes in the future.

Note: In this document some text is highlighted *like this*. The highlight indicates that the term is an exact reference to some component within the system and the corresponding definition can be found elsewhere within the document.

## Terminology

There are a number of key terms that are used within the system, but which do not explicitly relate to system components. These are defined in the sub-sections below:

### Event types

Within the demonstration there are a number of event types. To ensure accurate discussion, development and demonstration it is important that we have agreed names for these different event types and use them consistently. The proposed event type names and descriptions are:

* *Raw event*  
  Detected in the sub-symbolic parts of the system, from the *live video feed* by the *Video-DR* analytics. All generated *raw events* correspond to pre-defined activities in the Activity 101 dataset (See “Appendix A – Activity 101 classes” for details).
* *Mapped event*  
  Inferred in the symbolic parts of the system, by the *Infer-mapped-event* service, when a *raw event* is detected which has been mapped to a predefined *activity*.
* *Interesting event*  
  Also inferred in the symbolic parts of the system. Fuses information from video and audio analytics using symbolic rules. When a *mapped event* is detected with a corresponding spoken word from the *Audio-speech-to-text* analytics above a pre-defined frequency threshold. i.e. *“Hooray” mapped event was detected >3 times after the word “go” was spoken*.

### Activities

In addition to these event types, there is also the concept of an *activity*. They represent the mapping between the standard Activity 101 detectable *raw events* and the real-world behaviours that we have pre-defined our system to monitor for (which do not necessarily map to the existing Activity 101 classes). In our demonstration environment we have identified two *activities* that we will instruct the *participants* to perform. These are: “Hooray” and “Calm down” and have been mapped to appropriate detectable *raw events*. The system can support any number of *activities* and any number of mappings for each activity to *raw events*.

Exact mappings of *activities* to *raw events* will be determined during the development of the system based on the most reliably detectable *raw events* that can be easily communicated to the *participants*. Candidate *activity* names and *raw event* mappings are:

* Activity: “Hooray”
  + Raw events: PizzaToss, Diving, FrisbeeCatch
* Activity: “Calm down”
  + Raw events: BalanceBeam, Drumming, Rowing

### Trigger words and narrative

The *narrative* is a simple short story that is read to the *participant* during the demo. The *narrative* contains some pre-defined *trigger words* that have been communicated to the *participant* to show them how to react when they hear them. This is a simple visual form that shows the body shape they need to make and the *trigger words* that tell them when to do so, e.g. via a printed sheet that they are shown such as in Figure 1.

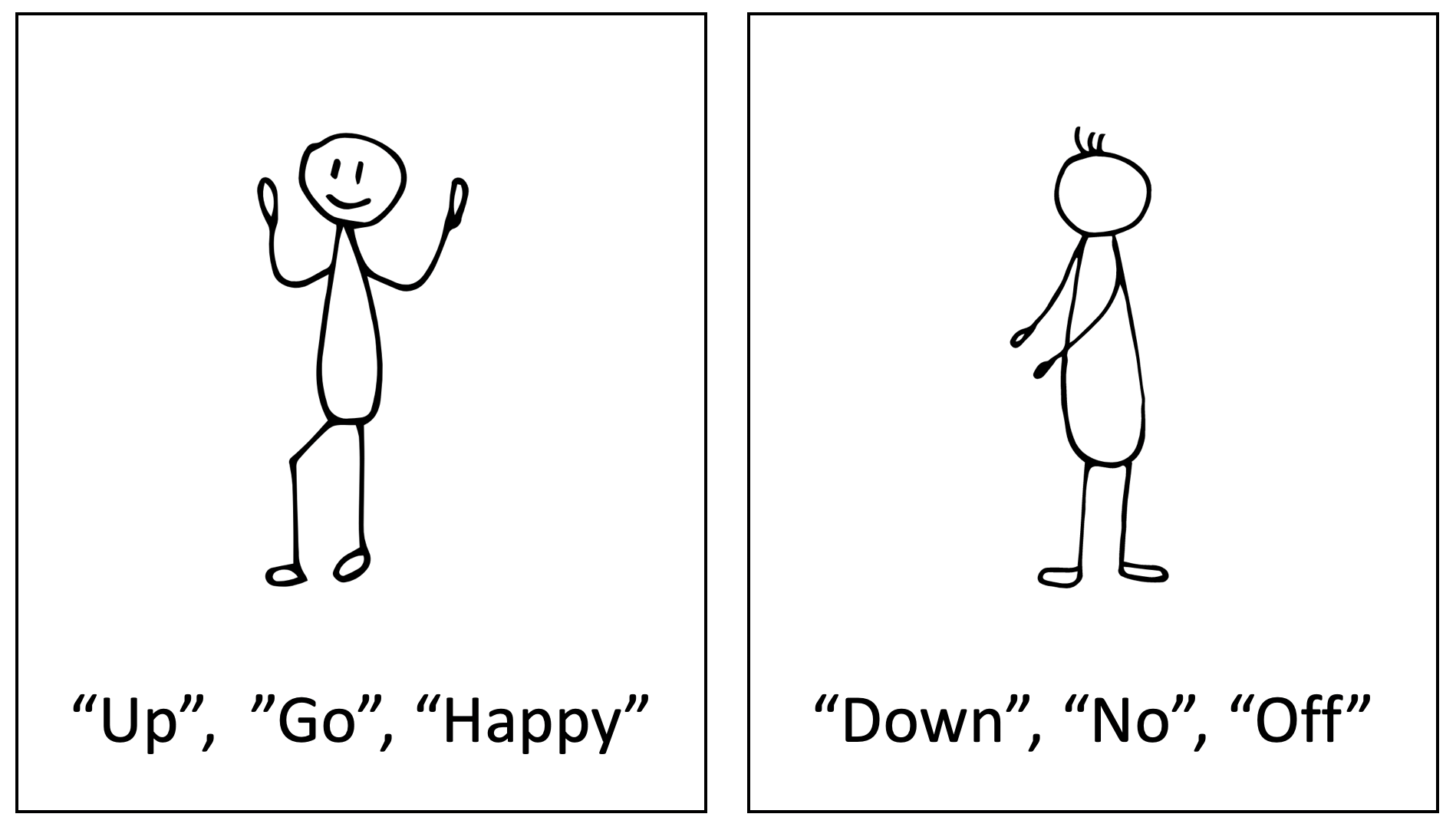


Figure - Example card showing participants body shapes and trigger words

The potential narrative to be read during the demo is: “I was walking around the conference and I decided to walk up the stairs. I was pretty happy to find lots of people. In the evening I will take some time off before I come back down tomorrow.”

### Demonstration roles

When giving the demonstration there are a number of roles for the demonstration team and the audience:

* *Participant* – someone chosen from the audience to be the subject of the demo.
* *Experimenter 1* – overall explainer of the demo, showing the user interface and explaining the system.
* *Experimenter 2* – reads the narrative text that drives the *participant* behaviour.

# System components

This section describes in more detail each of the components within the system. For interface components (such as web services) the full details are given in this document. For the underlying components (such as hardware or analytics capabilities) there is no need to provide full internal details in this document as it is up to each researcher to simply contribute those working components into the system.

## Hardware and feed components

* *Webcam*  
  A typical webcam that is set up against a plain background with a suitable place for the *participant* to stand and make their body shapes.
* *Microphone*  
  A typical microphone set up close to *experimenter 2*. Testing should be done in a noisy environment to check that reliable audio analytics can be performed.
* *Live video feed*From the *webcam*, this feed needs to be available to support video analytics.
* *Live audio feed*From the *microphone*, this feed needs to be available to support audio analytics. Audio analytics is done only when requesting explanations so the audio feed must be streamed to a file to support later explanation requests.

## Analytics components

* *Video-DR*  
  Real-time video analytics of the *live video feed*, applying DR techniques to identify and provide explanations for *raw events* as they occur.
* *Audio-DR*  
  Audio analytics, applying DR techniques to identify and provide explanations for *trigger words* within the audio stream. Invoked for a specific segment of the audio stream when an explanation is requested.
* *Audio-speech-to-text*Speech-to-text analytics of a particular audio segment, undertaken when an explanation is requested.
* *Infer-mapped-event*  
  Symbolic analytics that is invoked whenever *raw events* are detected. Uses logical inference rules to infer *mapped events* if the detected *raw event* is mapped to a pre-defined *activity*. See “Appendix D – Logical inference rules” for details.
* *Infer-interesting-event*  
  Symbolic analytics that is invoked whenever *mapped events* are detected. Uses logical inference rules to infer *interesting events* if a *mapped event* is correlated above a certain threshold to words detected in the audio input (via the *audio-speech-to-text* service). It is important to note that whilst the participant is told the *trigger words*, the system has no pre-define knowledge of these, and the logical inference rules will match to any correlated words that are consistently detected. See “Appendix D – Logical inference rules” for details.

## Service components

* *Stream-video-DR-output*  
  Input parameters: None  
  Processing: simply apply the *Video-DR* analytics to the incoming *live video feed*.  
  Output: A live video stream (modified from the input live stream by the application of the video-DR analytics, i.e. showing the Deep Taylor video explanation mode).
* *Video-explanation*  
  Input parameters: Event id  
  Processing: For a given event retrieve the video segment and the raw event that was detected.  
  Output: JSON description of the video explanation for the detected event. See Appendix C - “Video-explanation service JSON” for details.
* *Inference-explanation*  
  Input parameters: *tbc*  
  Processing: *tbc*  
  Output: JSON description of the inferences and their explanation(s) for the detected event. See Appendix C - “Inference-explanation service JSON” for details.
* *Audio-explanation*  
  Input parameters: timestamp  
  Processing: *tbc*  
  Output: JSON description of the audio explanation for the detected event. See Appendix C - “Audio-explanation service JSON” for details.
* *Audio-speech-to-text*Input parameters: *tbc*  
  Processing: *tbc*  
  Output: JSON output of the speech-to-text processing for the detected event. See Appendix C - “Audio-speech-to-text service JSON” for details.
* *List-events*Input parameters: timestamp (optional)  
  Processing: Lists all *raw events*, *mapped events* and *interesting events* that have been detected by the system since the specified optional timestamp. These events are detected by various analytics and inference components and recorded within the system (e.g. as files) and this service simply provides a mechanism for getting the details of them all in order to update the *user interface*.  
  Output: JSON output of the speech-to-text processing for the detected event. See Appendix C - “Audio-speech-to-text service JSON” for details.

## Wireframe layout

The core user interface for the demo will be very simple and is illustrated in Figure 2.

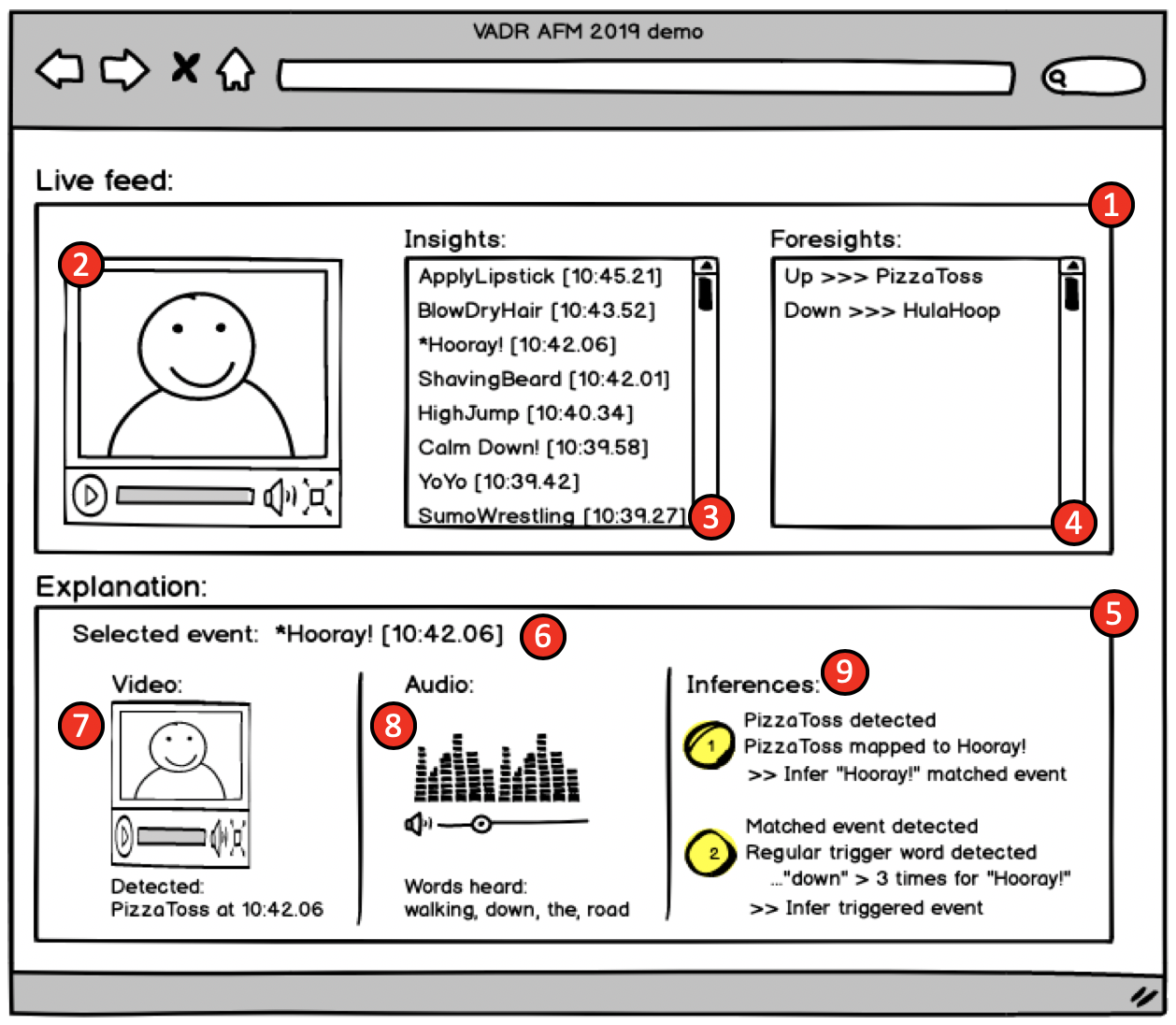


Figure - Simple wireframe for the VA-DR AFM 2019 demo

The wireframe shows the main controls that need to be present in order to explain the story for the demo. Key items or areas are indicated with red numbered circles and are described in the text below using “(n)” notation to relate the description to the numbered circle in the figure. The layout is not important and can be changed during implementation, however each control is key to the demonstration. The role and meaning of each control are best explained via a simple narrative:

* The *live feed section* (1):
  + Is initially only populated with the live *stream-video-DR-output* service feed into the *main video control* (2).
  + The *insights list* (3) and *foresights list* (4) are initially empty.
  + The *insights list* (3) is dynamically populated as *raw events* are detected as part of the *video-DR* analytics processing and *mapped events* are inferred as part of the *infer-mapped-event* analytics processing. Mapped events are shown in the list preceded by “\*”. This is achieved by polling the *list-events* service every few seconds.
  + The *foresights list* (4) is dynamically populated as *interesting events* are detected as part of the *infer-interesting-event* analytics processing. This is also achieved by polling the *list-events* service every few seconds.
* The *explanation section* (5):
  + Is initially empty. It is not populated until the user clicks on an entry in the *insights list* (3) within the *live feed section* (1).
  + There is an alternative mode for the *explanation section* (5), when the user clicks on a link in the *foresights list* (4). In this case only the *inferences list* (9) is shown, and the explanation given is the logical inference rules that led to the foresight inference, e.g. that three or more separate mapped events led to the foresight conclusion that there is an interesting event. It is possible that this could be shown in a separate part of the user interface instead of using the standard *explanations section* (5). This would also allow the three or more events that led to the conclusion to be clickable and each then shown in the *explanation section* (5) as they are clicked on.

## Demonstration details

The demonstration should ideally fit within a 10 minute or less time-frame. It will be run many times on an ad-hoc basis based on the open-ended format of the demonstration session. Some specific demonstration times may be pre-arranged, e.g. with senior invited guests or with peer reviewers.

The approximate steps for the demo are:

1. *Participant* is selected and shown the guidance card (See Figure 1) and asked to stand in front of the *webcam* and told what to do.
2. *Experimenter 1* shows the *user interface* (See Figure 2) to the audience, indicating the live feed, referring to the system diagram on the poster and explains what will happen.
3. *Experimenter 2* reads into the *microphone* the narrative text which includes some of the *trigger words*.
4. *Participant* reacts to the *trigger words* by making the required body shapes at the appropriate times.
5. The system should detect the *raw events*, adding them to the list in the *user interface*.
6. Some *mapped events* may be inferred, updating the list in the *user interface* accordingly.
7. Some *interesting events* may also be inferred, updating the list in the *user interface* accordingly.
8. *Experimenter 1* then shows the audience the various events (*raw*, *mapped* or *interesting*) in the *user interface*, clicking on one or more to show each explanation.
9. The team engage with the audience as needed, with more detailed discussion about the system components, overall design, symbolic/subsymbolic integration, military relevance etc.

# References

*tbc*

# Appendices

All information, code and data relating to the VADR environment and the specific demo for AFM 2019 is held in github, here: [https://github.com/dais-ita/vadr-demo/tree/afm-2019](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_dais-2Dita_vadr-2Ddemo_tree_afm-2D2019&d=DwMGaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=swmOA_Pyl4SqADw3EEEeejy_9cZb8HHfh1LHvuMN9CU&m=-ej1kz-dfMbBa0HaGbJSeApGynCCiOEcNxUuGPGW5tk&s=qGv2909yxlq7VZKdj_zAo6bGQejumjiYvGGlzdAcyRc&e=)

Each of the appendices below gives more detailed information about some aspect of the system.

## Appendix A – UCF101 classes

UCF101 is a dataset of 101 Human Action classes, based on over 13,000 video clips covering 27 hours of activity from publicly available videos.

The table below lists the 101 classes that are known to the activity classifier:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Activity Name** |  | **#** | **Activity Name** |
| 1 | ApplyEyeMakeup |  | 51 | LongJump |
| 2 | ApplyLipstick |  | 52 | Lunges |
| 3 | Archery |  | 53 | MilitaryParade |
| 4 | BabyCrawling |  | 54 | Mixing |
| 5 | BalanceBeam |  | 55 | MoppingFloor |
| 6 | BandMarching |  | 56 | Nunchucks |
| 7 | BaseballPitch |  | 57 | ParallelBars |
| 8 | Basketball |  | 58 | PizzaTossing |
| 9 | BasketballDunk |  | 59 | PlayingCello |
| 10 | BenchPress |  | 60 | PlayingDaf |
| 11 | Biking |  | 61 | PlayingDhol |
| 12 | Billiards |  | 62 | PlayingFlute |
| 13 | BlowDryHair |  | 63 | PlayingGuitar |
| 14 | BlowingCandles |  | 64 | PlayingPiano |
| 15 | BodyWeightSquats |  | 65 | PlayingSitar |
| 16 | Bowling |  | 66 | PlayingTabla |
| 17 | BoxingPunchingBag |  | 67 | PlayingViolin |
| 18 | BoxingSpeedBag |  | 68 | PoleVault |
| 19 | BreastStroke |  | 69 | PommelHorse |
| 20 | BrushingTeeth |  | 70 | PullUps |
| 21 | CleanAndJerk |  | 71 | Punch |
| 22 | CliffDiving |  | 72 | PushUps |
| 23 | CricketBowling |  | 73 | Rafting |
| 24 | CricketShot |  | 74 | RockClimbingIndoor |
| 25 | CuttingInKitchen |  | 75 | RopeClimbing |
| 26 | Diving |  | 76 | Rowing |
| 27 | Drumming |  | 77 | SalsaSpin |
| 28 | Fencing |  | 78 | ShavingBeard |
| 29 | FieldHockeyPenalty |  | 79 | Shotput |
| 30 | FloorGymnastics |  | 80 | SkateBoarding |
| 31 | FrisbeeCatch |  | 81 | Skiing |
| 32 | FrontCrawl |  | 82 | Skijet |
| 33 | GolfSwing |  | 83 | SkyDiving |
| 34 | Haircut |  | 84 | SoccerJuggling |
| 35 | Hammering |  | 85 | SoccerPenalty |
| 36 | HammerThrow |  | 86 | StillRings |
| 37 | HandstandPushups |  | 87 | SumoWrestling |
| 38 | HandstandWalking |  | 88 | Surfing |
| 39 | HeadMassage |  | 89 | Swing |
| 40 | HighJump |  | 90 | TableTennisShot |
| 41 | HorseRace |  | 91 | TaiChi |
| 42 | HorseRiding |  | 92 | TennisSwing |
| 43 | iHoop |  | 93 | ThrowDiscus |
| 44 | IceDancing |  | 94 | TrampolineJumping |
| 45 | JavelinThrow |  | 95 | Typing |
| 46 | JugglingBalls |  | 96 | UnevenBars |
| 47 | JumpingJack |  | 97 | VolleyballSpiking |
| 48 | JumpRope |  | 98 | WalkingWithDog |
| 49 | Kayaking |  | 99 | WallPushups |
| 50 | Knitting |  | 100 | WritingOnBoard |
|  |  |  | 101 | YoYo |

## Appendix B – Speech commands dataset classes

The table below lists the 35 speech command words and the number of instances included in the training data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Word** | **Number of Utterances** |  | **Word** | **Number of Utterances** |
| Backward | 1,664 |  | No | 3,941 |
| Bed | 2,014 |  | Off | 3,745 |
| Bird | 2,064 |  | On | 3,845 |
| Cat | 2,031 |  | One | 3,890 |
| Dog | 2,128 |  | Right | 3,778 |
| Down | 3,917 |  | Seven | 3,998 |
| Eight | 3,787 |  | Sheila | 2,022 |
| Five | 4,052 |  | Six | 3,860 |
| Follow | 1,579 |  | Stop | 3,872 |
| Forward | 1,557 |  | Three | 3,727 |
| Four | 3,728 |  | Tree | 1,759 |
| Go | 3,880 |  | Two | 3,880 |
| Happy | 2,054 |  | Up | 3,723 |
| House | 2,113 |  | Visual | 1,592 |
| Learn | 1,575 |  | Wow | 2,123 |
| Left | 3,801 |  | Yes | 4,044 |
| Marvin | 2,100 |  | Zero | 4,052 |
| Nine | 3,934 |  |  |  |

## Appendix C – Example JSON response formats

Details of the various web services required for this demonstration are given in this section. For ease of explanation they are given as specific JSON examples with any supporting explanations after the example as required.

### Video-explanation service JSON

*tbc*

### Inference-explanation service JSON

*tbc*

### Audio-explanation service JSON

*tbc*

### Audio-speech-to-text service JSON

*tbc*

### List-events service JSON

*tbc*

## Appendix D – Logical inference rules

*tbc*

(end of document)