Report on Head Gesture Detector (HGD)

[Please, also see the README.md for a brief overview of the Head Gesture Detector files.]

Contents

Overview

First, we experimented only with the nod detection using only the **vra1** dataset. Later, we have used 4 datasets (**vra1**, **hatice2010**, **sewa**, **nvb**) to perform cross-dataset testing (train a model on one dataset and test on each dataset). Finally, we used the combination (**4comb**) of all these 4 datasets to train the final Head Gesture Detector (HGD) and we further evaluated the HGD on the **ccdb** dataset.

Datasets

We compared and used several datasets suitable for head gesture detection from a video. The comparisons of various datasets are summarized in the document deep-virtual-rapport-agent/notes/Datasets.docx
The scripts for **analysis** of the datasets (vra1, …) can be found in the folder deep-virtual-rapport-agent/data_analysis
(Please, see the associated README.md files.)

Further notes on the employed / other related datasets follow:

vra1

- ICT audio-visual dataset of dyadic human interactions.
- Videos of **listeners** only.
- Head gesture annotations of listeners: **nod** and only few tilts (do not seem reliable => not used).
- For some annotation files the listener videos are missing and likewise for some listener videos the annotations are missing. So we excluded the excessive annotations and 1 excessive video manually (see the folders dvra_datasets/vra1/excessive_listener_nod_annotations and dvra_datasets/vra1/excessive_listener_videos)
  - => 45 video-annotation pairs left
- Had to do some manual data cleaning of annotations, so that only one tier name “Head Nods” appears in EAF files, if annotation is present.
- When generating the sequenced/segmented dataset, we ignored the beginnings of data until the beep (given by deep-virtual-rapport-agent/data_analysis/offsetListenerNods.txt).
- The speaker offsets (given by deep-virtual-rapport-agent/data_analysis/offsetSpeakerAudio.txt) were not relevant for head gesture detection, but they would need to be used if generating dataset for the development of the Rapport Model.

vra2

- Dataset by ICT.
- Looking at VRA2 data, we conclude it is not very useful for head nod detection, since the study participants **only watched** speakers' videos from VRA1 dataset and gave feedback by pressing space when they **would** nod/make brief sound (explanation in lixing/AAMAS_10/experiment_explanation.docx). This is however not a natural dyadic scenario that we wanted to learn from.

nvb

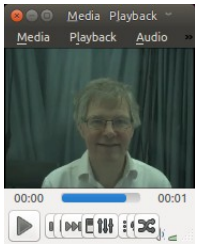
- From the original annotations, we used the files *Lia&Richard_merged_LiaNodSmile/Raw*_Final.eaf* (the other types of merged annotations by 2 people contained conflicting annotations).
- We had to further fix the Elan TIER_IDs in:
  - Raw31M0_Final.eaf: <TIER DEFAULT_LOCALE="en" LINGUISTIC_TYPE_REF="default-lt" TIER_ID="Head Tilt ">
  - Raw03F0_Final.eaf: <TIER DEFAULT_LOCALE="en" LINGUISTIC_TYPE_REF="default-lt" TIER_ID="Head Nods">

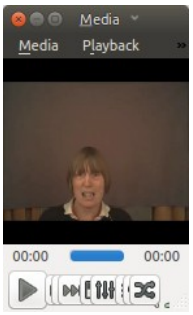
hatice2010

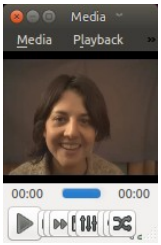
- We had to manually identify subjects in this dataset.
- The discovered mapping (recording_name → subject_id) is provided below and implemented in deep-virtual-rapport-agent/datasets_scripts/hatice2010/annotate_features.ipynb and then used in deep-virtual-rapport-agent/datasets_scripts/4comb/generate_4comb_dataset.ipynb

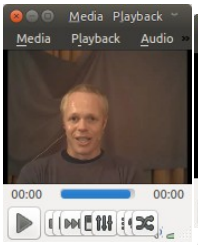
- **NOD** (11 different session names, 8 different subjects)

2008.12.05.16.03.15_Operator_AV_lowQ_sf1260_ef1349	1
2008.12.05.16.03.15_User_AV_lowQ_sf23785_ef23820	2
2008.12.14.14.47.07_Operator_AV_lowQ_sf1540-ef1580	1
2008.12.14.14.47.07_User_AV_lowQ_sf11278_ef11310	2
2008.12.19.11.03.11_User_AV_lowQ_sf16724_ef16772	3
2009.01.06.14.53.49_User_AV_lowQ_sf20635_ef20710	4
2009.01.28.15.35.20_Operator_AV_lowQ_sf1843_ef1895	5
2009.01.28.15.35.20_User_AV_lowQ_sf1160_ef1180	1
2009.05.22.15.17.45_User_AV_lowQ_sf4632_ef4680	6
2009.05.25.11.23.09_User_AV_lowQ_sf3705_ef3758	7
2009.05.26.10.19.53_User_AV_lowQ_sf823_ef878	8

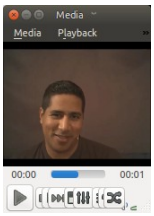

1


2


3

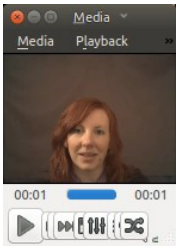

4

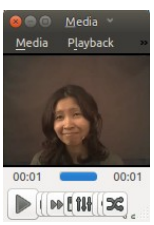

5


6

- **SHAKE** (same subjects as above, that is 8 different subjects)

2008.12.05.16.03.15_Operator_AV_lowQ_sf41794_ef41860	1
2008.12.05.16.03.15_User_AV_lowQ_sf19080_ef19202	2
2008.12.14.14.47.07_Operator_AV_lowQ_sf51928-ef51990	1
2008.12.14.14.47.07_User_AV_lowQ_sf17670_ef17700	2
2008.12.19.11.03.11_Operator_AV_lowQ_sf28930-ef28965	1
2008.12.19.11.03.11_User_AV_lowQ_sf11840_ef11864	3

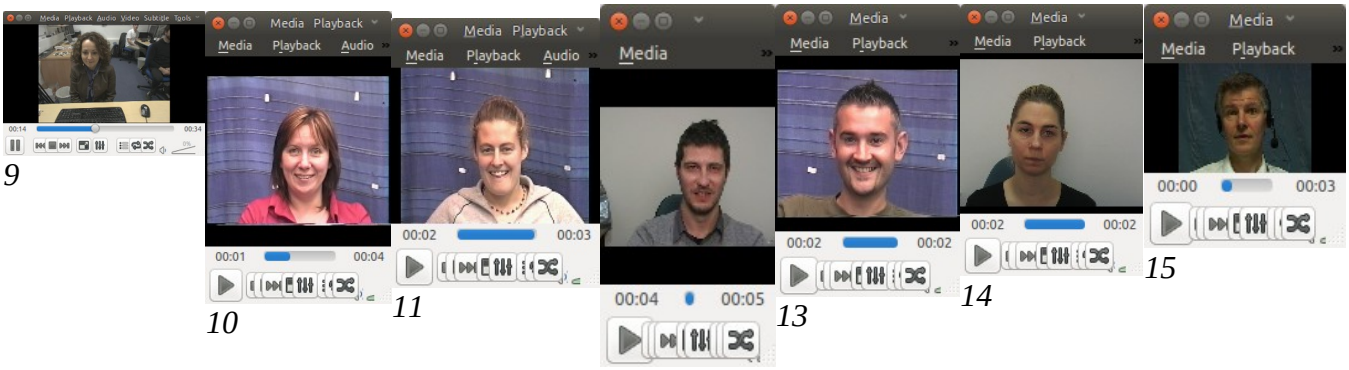

7


8

2009.01.06.12.41.42_Operator_AV_lowQ_sf7335_ef7426	1
2009.01.06.14.53.49_Operator_AV_lowQ_sf15730_ef15790	1
2009.01.06.14.53.49_User_AV_lowQ_sf28497_ef28555	4
2009.01.28.15.35.20_Operator_AV_lowQ_sf21778_ef21878	5
2009.01.28.15.35.20_User_AV_lowQ_sf24478_ef24537	1
2009.01.30.12.00.35_Operator_AV_lowQ_sf15181_ef15205	5
2009.01.30.12.00.35_User_AV_lowQ_sf2790_ef2806	2
2009.05.22.15.17.45_User_AV_lowQ_sf52935_ef52985	6
2009.05.25.11.23.09_User_AV_lowQ_sf12002_ef12073	7
2009.05.26.10.19.53_User_AV_lowQ_sf24281_ef24350	8

- **OTHER** (below are the first 4 characters of 7 types of session names, 7 additional subjects)

2008... <=> Hatice	9
Alex...	10
Alis...	11
anto...	12
Liam...	13
maja...	14
rodA...	15



9 *10* *11* *12* *13* *14* *15*

4comb

- Combination of all 4 main datasets used for the development of the final Head Gesture Detector
  - **vra1**: 1 recording corresponds to 1 subject (45 subjects in total)
  - **sewa**: 538 recordings from 275 subjects
  - **hatice2010**: 283 recordings from 15 subjects
  - **nvb**: 1 recording corresponds to 1 subject (38 subjects in total)
    - [checked semi-automatically, manually comparing extracted frames]
- Not all head gestures (nod, shake, tilt) were available for each dataset. See below what head gesture data came from a particular dataset.
  - vra1: nod
  - hatice2010: nod, shake
  - sewa: nod, shake
  - nvb: nod, shake, tilt

ccdb

- We extracted features from all 50 videos from all sessions in the CCDb dataset.
  - 8 sessions (16 videos) have at most 1 annotation at a time - ***basic set***
    - From 6 different subjects P1-P6.
    - This basic set was used for cross-dataset testing of the developed HGD.
      - The extracted OpenFace vision features were annotated with the provided head gesture ground-truths and compared with the extracted OpenFace vision features annotated using the developed HGD.
  - 17 sessions (34 videos) can have 2 distinct annotations at a time (by annotators A1/A4) - ***extended set***
- We also checked the ccdb TIER_IDs of the Elan annotations - they were ok.
- We noted that the ground-truths overlap (e.g. one frame is annotated as nod and shake at the same time).

MultiLis Corpus

- Very relevant, suggested by LP. Morency.
- See the paper: "The MultiLis Corpus – Dealing with Individual Differences in Nonverbal Listening Behavior"
- We contacted Prof Dirk Heylen (d.k.j.heylen@utwente.nl). He replied that he will provide the dataset, but we have not received it yet.

FIPCO

- It is the **largest** dataset we found, and it contains nod, shake and tilt annotations (see deep-virtual-rapport-agent/notes/Datasets.docx for comparison).
- Paper: "Head Gesture Recognition in Spontaneous Human Conversations: A Benchmark" (http://www.cbi.gatech.edu/fpv2016/abstracts/hand_gesture_recognition_abstract.pdf)
  - We contacted Prof Yang Wu (yangwu@rsc.naist.jp). He provided us with the following:
    - ZFace features
    - Annotations
  - Later, we should be also able to get the original videos.
  - We further asked the following questions about the provided data:
    1.) What are the other three features 4-6 extracted by Z-Face?
    => “4-6 are the x,y,z coordinates of the mean position of the facial landmarks.”
    2.) Does the annotation -1000 mean the same as 0?
    => “For all items, "-1000" (default value) means unlabeled. You can treated the same as '0'. As originally I thought that some annotators might choose '0' and some other may just ignore if no annotation is needed for that frame, I added this default value. It just means no annotation for its value.”
- We also contacted the author Mohit Sharma (mohits1@andrew.cmu.edu) of the work “Recognizing Visual Signatures of Spontaneous Head Gestures” that uses the FIPCO dataset and also evaluates the developed head gesture detectors on the CCDb dataset.
  - We can thus compare our HGD with theirs, performing a cross-dataset evaluation of our final HGD on the CCDb dataset.
  - However, they evaluated only on the *basic set* of the CCDb. They did not use the *extended set* where there might be 2 distinct annotations at a time (by annotators A1/A4).
    => for comparison with them, just use the *basic set* from CCDb
  - They shared with us their code (deep-virtual-rapport-agent/head_gesture_detector/multi_scale_head_gesture-master) https://github.com/mohitsharma0690/multi_scale_head_gesture

Data preprocessing

The scripts for **preprocessing** of the datasets (vra1, …) can be found in the folder deep-virtual-rapport-agent/datasets_scripts (Please, see the associated README.md files for dataset-specific preprocessing pipelines and more details.)

In general, the preprocessing pipeline is:

extract vision features → *annotate features* → *generate sequenced/segmented dataset*
(using OpenFace) (with ground-true head gestures)

Vision features extraction

We used OpenFace to get per-frame pose features, namely, we focused on x, y, z head translations and rotations (pose_Tx, pose_Ty, pose_Tz, pose_Rx, pose_Ry, pose_Rz). Details on how these vision features were finally used are provided in the section *Generating sequences/segments*.

The feature extraction scripts can be found as deep-virtual-rapport-agent/datasets_scripts/*/extract_features.sh
For vra1 dataset, we checked that the missed frames by OpenFace do not present a significant fraction (deep-virtual-rapport-agent/data_analysis/vra1_stats.ipynb).

Annotating vision features

We annotated the extracted OpenFace vision features from the datasets (**vra1**, **hatice2010**, **sewa**, **nvb, ccdb**) with the provided ground-true head gestures.

The deep-virtual-rapport-agent/datasets_scripts/*/annotate_features.ipynb scripts perform the following:

- Resample the csv files of OpenFace features to common frame rate (30 Hz).
- Add 1st and 2nd differences of x, y, and z head translations and rotations (prepending zeros at the beginning to keep the same number of frames). The added columns were named 'diff_ pose_Tx', 'diff2_ pose_Tx', ...
- Annotate frames with ground-true head gestures from the set {nod, shake, tilt} depending on the dataset, adding a new column for each kind of head gesture. We used binary labels 0 / 1 that denote that a head gesture is / is not present in a given frame.
- Save new resampled and annotated dataframes as csv files.

Generating sequences/segments

Having vision features annotated with head gestures, we generated final datasets (**vra1**, **hatice2010**, **sewa**, **nvb, 4comb, ccdb**) of sequences/segments of length WINDOW_SIZE past frames. The terms window size and sequence/segment length refer to the same concept.
Using the scripts deep-virtual-rapport-agent/datasets_scripts/*/generate_dataset.ipynb, we generated the datasets as follows:

- *Note*: for each head gesture we generated a separate segmented dataset, since we aimed at developing 3 separate detection models (for nod, shake, and tilt).
- We generated datasets with 2 sets of vision features:
  - Set named *6f:*
  - 'diff_ pose_Tx',
  - 'diff_ pose_Ty',
  - 'diff_ pose_Tz',
  -
  - 'diff_ pose_Rx',
  - 'diff_ pose_Ry',
  - 'diff_ pose_Rz',
  -
  - Set named 12*f:* *[used for the final HGD]*
  - 'diff_ pose_Tx',
  - 'diff_ pose_Ty',
  - 'diff_ pose_Tz',

    -
    - 'diff2_ pose_Tx',
    - 'diff2_ pose_Ty',
    - 'diff2_ pose_Tz',
    -
    - 'diff_ pose_Rx',
    - 'diff_ pose_Ry',
    - 'diff_ pose_Rz',
    -
    - 'diff2_ pose_Rx',
    - 'diff2_ pose_Ry',
    - 'diff2_ pose_Rz',
- Dataset split (for individual datasets; for the combined 4comb dataset it was different, see below)
    - Each csv file was randomly split into train and validation partitions.
        - Validation set size: 15%.
        - The **train-validation** split was **subject-dependent**, since the data from one subject appear in both the train and validation partitions.
    - For each dataset, we also saved a test partition that contains all the segments for cross-dataset testing.
    - In case of the **ccdb** dataset, only the test partition was generated and later used for cross-dataset testing of the final Head Gesture Detector trained on the **4comb** dataset.
- Segmentation details:
    - Dataset was segmented into same-length sequences of the length WINDOW_SIZE = 32 frames.
    - Feature segments were pre-padded with MASK_VALUE-s and label segments with 0 (not a nod/shake/tilt).
    - Since we address the head gesture detection task as a seq2seq problem, we obtain several predictions for the same frame. Therefore, to correctly compute the validation and test metrics we used majority (and another) voting schemes. For this, we also had to save chunk sizes (counts of adjacent frames that belong to the same dataset partition for each dataframe). These chunk sizes/lengths were saved as arrays of integers within the segmented dataset files using the keys “train_len”, “val_len”, and “test_len”.
- All generated sequences/segments were saved in one output file per head gesture per dataset, using the naming convention *{dataset_name}_{head_gesture}_{window_size}ws_{len(selected_features)}f.npz*

- In case of the **4comb** dataset, the procedure differed in the following:
    - The **train-validation-test** split (0.7 : 0.15 : 0.15) was **subject-independent** (i.e., data from one subject must appear only in one partition).
    - We also experimented with data augmentations
        - In general, we came up with the following types of (random/pre-defined) augmentations:
            - [not implemented] Stretch a head gesture feature timeseries in time.
            - [not implemented] Flip a head gesture feature timeseries along the time axis.
            - Flip a head gesture feature timeseries along the feature axis.
                - Specifically, we performed all 2^3 = 8 pre-defined augmentations of rotation features timeseries.
                - This was named an augmentation A1. The suffix “_A1” in a segmented dataset name denotes that the dataset was augmented using this augmentation method A1.
        - For specific details on various approaches to the augmentation of the third type please see deep-virtual-rapport-agent/datasets_scripts/4comb/generate_4comb_dataset.ipynb
        - However, the augmentation didn’t seem to improve the performance (it rather deteriorated the performance).
    - The print outputs from the generate_4comb_dataset.ipynb script were also saved in deep-virtual-rapport-agent/notes/results_head_gesture_detection/log_generate_4comb_dataset.docx

Models and training

Using the Keras library, we implemented machine learning models to learn the mapping from a vision feature sequence to a sequence of a particular head gesture binary labels (i.e., seq2seq prediction). In particular, we set up the models as follows:

- One recurrent layer of *N_GRU* GRU units returning whole sequences
  - with dropout of 0.1, and no recurrent dropout
- Time-distributed output layer
  - with the sigmoid activation function

Since we performed seq2seq prediction on segments/sequences that are usually shorter than whole recordings, we had to fuse multiple predictions (for the same frame) from adjacent output segments/sequences. Specifically, we used 2 voting strategies:

- **Last voting strategy**
  - The frame's label was set to the last frame prediction of the segment that ends on this frame.
- **Majority voting strategy**
  - The frame's label was set to the most common frame prediction considering predictions (for this frame) from all segments that involved this frame.
  - Ties were resolved using the last voting strategy.

We further post-process the sequences of fused binary head gesture labels as follows:

- **Smoothed** [filename infix *_S_*]
  - The sequences of fused binary head gesture labels were smoothed using the median filter with the kernel size of 9 frames.
  - These predictions were meant for final use of the HGD model to annotate unseen data (e.g., from a potential user study).
- **Non-smoothed** [filename infix *_nonS_*]
  - No smoothing was applied to the sequences of fused binary head gesture labels.
  - These predictions were used for validation and can be used for model comparison with other works (e.g., comparing with the work "Recognizing Visual Signatures of Spontaneous Head Gestures" on the ccdb dataset).

As expected, the smoothed predictions result in better evaluation metrics.

For details on voting strategies and smoothing, please see the script deep-virtual-rapport-agent/head_gesture_detector/utils.py with the corresponding helper functions.

The default training and validation parameters we used were as follows:

- Batch size: 128
- Number of training epochs: 100
- Adam optimizer.
- Binary cross-entropy loss and temporal sample weighting mode.
- We monitored the validation balanced accuracy metric (computed using the last voting strategy) to decide whether to update the best model checkpoint and stop training, using early stopping with a patience of 10 epochs. (In case of the final training on the whole 4comb dataset (see **TS3** below), we did not use early stopping and trained for a fixed number (100) of epochs, since the validation metrics oscillated a lot.)

We trained a **separate HGD model for each head gesture** (nod, shake, and tilt) and in 3 different settings:

1. **[TS1]** Train and validate a HGD model **on one dataset** to detect one head gesture.
   - We used the datasets (vra1, hatice2010, sewa, nvb) so that we trained and validated on one dataset and later tested on each of these datasets (cross-dataset evaluation).
   - *Note:* this setting uses **subject-dependent** dataset splits.
   - Not all head gestures (nod, shake, tilt) were available for each dataset, see the possible combinations below:
   - vra1: nod
   - hatice2010: nod, shake
   - sewa: nod, shake
   - nvb: nod, shake, tilt
   - So far, for TS1, only the nod head gestures were used
     => 4 **nod** HGD models were trained.

- Script: deep-virtual-rapport-agent/head_gesture_detector/train_hgd.py

- *Experiments:*
  - We compared validation metrics obtained using various model architectures (4, 8, 16, and 32 GRU units). The detailed results can be found in the document deep-virtual-rapport-agent/notes/results_head_gesture_detection/TR1_nod_validation.docx
  - We noticed that the last voting strategy results in slightly better evaluation metrics than the majority voting strategy. Reasoning/explanation: the prediction from last frame of a segment is better informed than intermediate predictions within other segments. We thus made our further decisions and comparisons using the last voting strategy.

2. **[TS2]** Train and validate a HGD model **on the 4comb dataset** to detect one head gesture.
   - We used the train and val partitions of the 4comb dataset for training and validation, whereas the test set was untouched and later used to evaluate the developed Head Gesture Detector on the same dataset.
   - *Note:* this setting uses **subject-independent** dataset split.
   - Used with all head gestures (nod, shake, tilt)
     => 3 HGD models were trained.
   - Script: deep-virtual-rapport-agent/head_gesture_detector/train_4comb_hgd.py

   - *Experiments:*
     - We compared validation metrics obtained using various model architectures (in terms of the number of GRU units) and found the optimal numbers of GRU units to be 16, 8, and 16 for nod, shake, and tilt head gesture detection models respectively. The detailed results can be found in the documents deep-virtual-rapport-agent/notes/results_head_gesture_detection/TR2_4comb_{nod, shake, tilt}_validation&testing.docx

3. **[TS3]** Train and validate a final HGD model **on the whole 4comb dataset** to detect one head gesture.
   - In this case, the test set was also used for training (train = train + test, val = val).
   - The developed final Head Gesture Detector can be evaluated on other datasets (e.g., cross-dataset testing on ccdb) and used for general prediction of head gestures (see the *Final Head Gesture Detector* section below).
   - *Note:* this setting uses **subject-independent** dataset split.
   - Used with all head gestures (nod, shake, tilt)
     => 3 HGD models were trained (using the best model architectures found in TS2 during validation)
   - Script: deep-virtual-rapport-agent/head_gesture_detector/train_final_4comb_hgd.py

   - *Experiments:*
     - We compared validation metrics obtained with and without data augmentation (A1) and found that this augmentation didn't improve the performance (it rather deteriorated the performance). The detailed results can be found in the document deep-virtual-rapport-agent/notes/results_head_gesture_detection/TR3_final_4comb_validation.docx
     - After training the nod HGD for 100 epochs, it seemed that it might not be enough. We thus trained another nod model for 200 epochs (deep-virtual-rapport-agent/head_gesture_detector/checkpoints/final_4comb/final_200epochs_4comb_nod_32ws_12f_16u.{hdf5, pkl}). However, 200 epochs gave boost of only 0.0003 in the "last voting strategy" validation bacc metric, so we did not use this model further.

Model checkpoints (.hdf) and training history (.pkl) files are saved to the deep-virtual-rapport-agent/head_gesture_detector/checkpoints folder. The files related to training settings TS1, TS2, and TS3 can be found in the folders ./checkpoints, ./checkpoints/4comb, and ./checkpoints/final_4comb respectively. The best final nod, shake, and tilt HGD models can be found in the folder ./checkpoints/final_4comb/best
Filenames naming convention:

{dataset_name}_{head_gesture}_{window_size}ws_{number_of_features}f_{model_architecture}.{hdf5,pkl}

Training history files contain training parameters and train and validation evaluation metrics.

The training history of HGD models from every training setting (TS1, TS2, TS3) can be viewed using the script deep-virtual-rapport-agent/head_gesture_detector/show_training_history.ipynb (and the script deep-virtual-rapport-agent/head_gesture_detector/utils.py contains several related helper functions). This was used to compare training and validation curves for various window sizes, number of features, model architectures, and evaluation metrics. (It also shows the training history when using the augmentation A1 with the training setting TS3.)

Final Head Gesture Detector

The final nod, shake, and tilt Head Gesture Detector was developed using the training setting TS3 (model checkpoints can be found in the folder ./checkpoints/final_4comb/best). This final HGD annotates frames of given csv files of vision features (obtained using OpenFace) with nod, shake, and tilt head gestures. We save the predictions from each model independently and also as one fused label per frame (0/1/2/3 denote none/nod/shake/tilt head gesture classes respectively). Specifically, each output csv file differs from the corresponding input csv file in the following:

- All columns are resampled to 30 FPS.
- 12 new columns of **difference features**:
  - Prefix "diff_" denotes first-order differences.
  - Prefix "diff2_" denotes second-order differences.
- 6 new columns of **predictions from 3 independent binary head gesture classifiers** (nod/shake/tilt):
  - Suffix "_prob" denotes probability of a positive (head gesture occurred) class.
  - Binary output 1 denotes a positive (head gesture occurred) class.
    - These binary predictions are smoothed with a median filter with a kernel size of 9 frames (this smoothing may cause that probabilities for some frames are not in agreement with associated binary labels).
    - Columns with suffix "_NS" denote non-smoothed binary outputs.
- 4 new columns of **fused predictions** from the independent 3 binary classifiers, so that only one unique head gesture is predicted at a time:
  - Labels 0/1/2/3 denote none/nod/shake/tilt head gesture classes respectively.
  - Columns with suffix "_NS" denote fused predictions of non-smoothed binary outputs (other columns are based on smoothed binary outputs).
  - Columns starting with "head_gesture_max_probab" contain labels fused using the **max probability fusion**:
    - If exactly one of the 3 classifiers detects a head gesture, assign this head gesture class.
    - If multiple classifiers detect a head gesture, assign the highest probability class.
      - Tie resolution: if there are multiple max probabilities, assign none class.
    - Otherwise, assign none class.
  - Columns starting with "head_gesture_unique" contain labels fused using the **unique fusion**:
    - If exactly one of the 3 classifiers detects a head gesture, assign this head gesture class.
    - Otherwise, assign none class.

Usage:

- The script to run the final HGD is deep-virtual-rapport-agent/head_gesture_detector/hgd_annotate_frames.ipynb

So far the HGD was run on:

- CCDb dataset (to evaluate and compare the performance of the HGD with previous works)
- Mimicry DB (for the main project to develop rapport models)
- IPD data from Qintian Li (summer intern, 2019)

Annotating videos with head gestures

- The annotated vision features csvs output from the HGD and the original videos (used to obtain the vision features) can be used to create annotated videos with in-video annotations of head gestures, namely, head nod, head shake, and head tilt. See the script deep-virtual-rapport-agent/head_gesture_detector/hgd_annotate_videos.py
- This allowed us to qualitatively evaluate the performance of the developed HGD.

Figure 1 shows the whole data pipeline of the developed Head Gesture Detector (HGD).

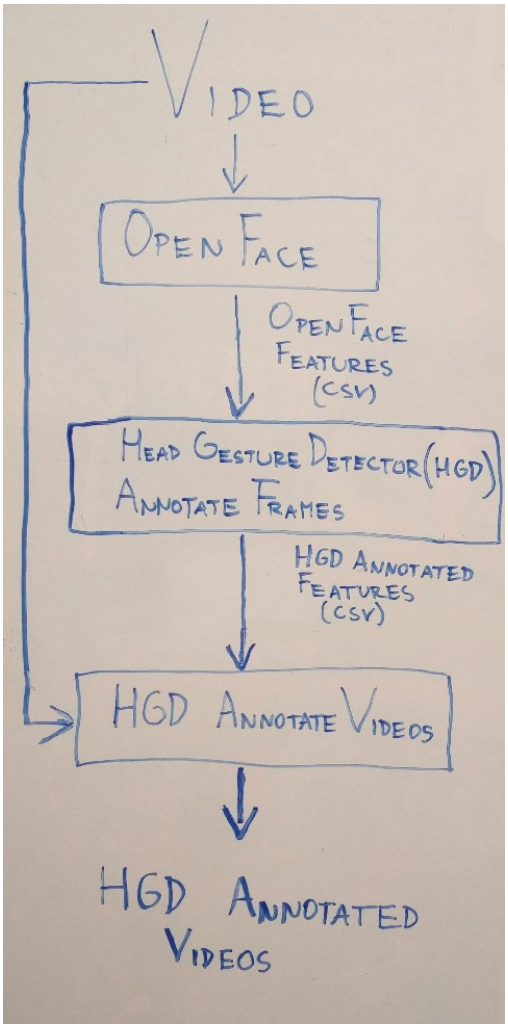


Figure 1: Data pipeline of the developed Head Gesture Detector (HGD).

Testing

To evaluate the developed models, we calculated the following metrics: **precision**, **recall**, **f1**, and **bacc** (balanced accuracy). These metrics were also used during training for validation.

We tested the HGD models according to the 3 training settings. In each case we saved the test results as one .pkl file per head gesture in the folders ./test_results/test_results_{S,nonS}, ./test_results/test_results_4comb_{S,nonS}, and ./test_results/test_results_final_4comb_{S,nonS} for the training settings TS1, TS2, and TS3 respectively.
Filenames naming convention:
test_results_{dataset_name}_{S,nonS}_{head_gesture}_{window_size}ws_{number_of_features}f_{model_architecture}.pkl

1. **[TS1]** Each model trained on one of the vra1, hatice2010, sewa, and nvb datasets was tested on each of these datasets. So far, we performed 4 x 3 = 12 nod only **cross-dataset** tests. The script to perform testing and save the results as .pkl files: deep-virtual-rapport-agent/head_gesture_detector/evaluate_hgd. ipynb

Results:

- The summary of testing results obtained using various model architectures (4, 8, 16, and 32 GRU units) with non-smoothed and smoothed predictions can be found in the document deep-virtual-rapport-agent/notes/results_head_gesture_detection/TR1_nod_testing.docx

2. **[TS2]** Nod, shake, and tilt detector models trained on the **4comb** dataset were tested on the test partitions of the **same** dataset. The script to perform testing and save the results as .pkl files: deep-virtual-rapport-agent/head_gesture_detector/evaluate_4comb_hgd.ipynb

   Results:

   - The summary of testing results obtained using the best model architectures (16, 8, and 16 GRU units for nod, shake, and tilt head gesture detection models respectively) with non-smoothed and smoothed predictions can be found in the documents deep-virtual-rapport-agent/notes/results_head_gesture_detection/ TR2_4comb_{nod, shake, tilt}_validation&testing.docx

3. **[TS3] Final** nod, shake, and tilt detector models trained on the **whole 4comb** dataset were tested on other datasets, namely, **cross-dataset** testing on the **ccdb** dataset.
   - We performed testing of the nod, shake, and tilt models **independently** and saved the results as .pkl files using the script deep-virtual-rapport-agent/head_gesture_detector/evaluate_final_4comb_hgd. ipynb (uses the segmented ccdb datasets).
   - We also performed testing of the **fused** 4-class (none/nod/shake/tilt) HGD model, also showing confusion matrices, using the script deep-virtual-rapport-agent/head_gesture_detector/evaluate_final_4comb_hgd. ipynb (uses the ccdb vision features csvs annotated by the final HGD).

     In order to compare our HGD with the work "Recognizing Visual Signatures of Spontaneous Head Gestures" that also evaluates their head gesture detector on the CCDb dataset, we had to fuse the ccdb ground-truth annotations since there were frames annotated with 2 or more head gestures at a time (e.g. nod and shake at the same time). *Note:* The work "Recognizing Visual Signatures of Spontaneous Head Gestures" does not mention how they fused the ccdb ground-truth annotations!
     First, we performed an analysis (evaluate_final_4comb_hgd. ipynb) of ground truth annotation overlaps in the basic set of the ccdb dataset and concluded that the ground truth annotation overlaps do not present a significant fraction, and so we can neglect/skip such frames. We were then able to compare our fused 4-class predictions with the 4-class ccdb ground-truth annotations. (Earlier we also tried other approaches to fuse the ground-truth annotations when a frame is annotated with 2 or more head gestures at a time: (1) use the none label, or (2) impose priorities nod > shake > tilt in such cases.)

   Results:

   - The summary of testing results on the ccdb dataset obtained using the best model architectures (16, 8, and 16 GRU units for nod, shake, and tilt head gesture detection models respectively) trained with and without data augmentation (A1), with non-smoothed and smoothed predictions can be found in the document deep-virtual-rapport-agent/notes/results_head_gesture_detection/ TR3_final_4comb_ccdb_testing.docx
     It provides the results when testing nod, shake, and tilt models independently as well as a final fused 4-class (none/nod/shake/tilt) HGD model.

Initial experiments with vra1 dataset

First, we experimented only with the **nod** detection using only the **vra1** dataset. The related scripts and checkpoints can be found in the folder deep-virtual-rapport-agent/head_gesture_detector/_early_vra1_nod_only and notes and results in the folder deep-virtual-rapport-agent/notes/results_head_gesture_detection/_early_vra1_nod_only

The terms “cleaned” and “non-cleaned” denotes whether the beginnings of recordings from the vra1 dataset, prior to the beep, were skipped or not skipped respectively.

Further notes from these experiments follow:

- Prior to balancing the loss, we also tried random under-sampling of the majority class (“head gesture not present”).
  - This meant that we could use accuracy as an evaluation metric.
  - We under-sampled the majority class (0 / not a nod) in each data partition:
    - original # examples: train:val:test = 33984:7018:5806 = 0.73:0.15:0.12
  - We did not perform oversampling of the minority class, as we had about 6-times more data from the negative class and synthesizing so many new data is very likely to generate a biased dataset.

- Prior to using the padding and masking in generating the sequences/segments, we created segments from intervals only where all original WINDOW_SIZE frames could be used, i.e., we did not exploit all the information at the start/end of recordings. We were assuming that the major head gestures do not tend to occur immediately at the start/end of recordings.

- Initially, we were predicting the nod head gesture for a single frame from features from a sequence of frames. This was later changed to the seq2seq prediction.

- Initially, we were using features from past and also from future frames to predict a head gesture for the current frame. However, such an approach is not well-suited for real-time head gesture detection.

- We investigated if **head translations** by OpenFace are informative features for nod detection. Specifically, we compared the performance of models using 2 different sets of features:
  - 3 features (1st order differences of 3 head rotations)
  - 6 features (1st order differences of 3 head rotations and 3 head translations)

  We found that **translation features are informative** (performance drop by ~6-10% when excluded).
  The results from this comparison can be found in the document
  deep-virtual-rapport-agent/notes/results_head_gesture_detection/_early_vra1_nod_only/
  WithOrWithoutTranslationFeatures.docx

- We investigated if **2nd order differences** of head rotations and translations by OpenFace are informative for nod detection. Specifically, we compared the performance of models using 2 different sets of features:
  - 6 features (1st order differences of 3 head rotations and 3 head translations)
  - 12 features (above 6 features with their 2nd order differences added)

  Training settings: 128 GRU unis, 32 batch size, max 100 epochs, early stopping on validation loss with a patience of 10 epochs. We found that in case of 12 features the training converges faster. However, the differences were very small. Reasoning: having the same model complexity and fewer features results in slower convergence => 2nd order differences **are informative**.

- We considered various dataset splits:
  - Subject-independent split into 9-folds
    - Split dataset
      - subject-independent split into 9-folds
        - train+val partition (40 subjects) and test partition (5 subjects)
    - Segment dataset
      - In, each fold:
        - each recording (from train+val partition) is randomly split into train and val partition (15%), then padded and masked (both features and labels) and segmented
        - each recording (from test partition) is padded and masked (both features and labels) and segmented

        - One output dataset file is generated for each fold.
    - Subject-dependent split
        - split each recording into 3 parts (train/val/test) randomly
    - Subject-independent 45-fold split
        - various feature sets 6 vs 12 features
        - various window sizes
        - various GRU architectures (e.g. 64-64u denotes 2 layers of 64 GRU units)
        - Extensive results (including the visualization of results by subject) can be found in the documents deep-virtual-rapport-agent/notes/results_head_gesture_detection/_early_vra1_nod_only/45fold_*.docx and a summary in the document deep-virtual-rapport-agent/notes/results_head_gesture_detection/_early_vra1_nod_only/Summary_45fold.docx

- We experimented with batch normalization
    - BN0 – only after input layer
    - BN1 – after input and first GRU layer
        - not clear if better than BN0 (comparing based on val_loss decrease in first epochs)
    - BN2 – after input and both GRU layers
        - not clear if better than BN0 (comparing based on val_loss decrease in first epochs)
    - BN3 – after input and last dense layer (before sigmoid activation)
        - worse than BN0 (comparing based on val_loss decrease in first epochs)

  => Batch normalization helps if applied after the masking layer only.

- Analysis of nod data from the vra1 dataset: deep-virtual-rapport-agent/notes/results_head_gesture_detection/_early_vra1_nod_only/NodsAnalysis.docx

Future work & ideas to try

- Data
    - Add the CCDb dataset to the 4comb dataset (creating a new 5comb dataset) and train new HGD on more data. To use the *extended set* of the CCDb dataset we can perform union/intersection of annotations and compare performance of different approaches.
    - Add facial landmarks to vision features (similarly to "Recognizing Visual Signatures of Spontaneous Head Gestures"). Or add action units to vision features (e.g., the work "Evaluating Models of Speaker Head Nods for Virtual Agents" states in the Introduction "… head movements are also influenced by our emotions …" which implies that using action units to predict head gestures is reasonable and might be beneficial).
    - Perform further data augmentation types (as suggested in the *Generating sequences/segments* section) and use fit_generator so that the augmentations are generated on-the-fly. However, in case of the stretch/squeeze augmentation along the time axis, it is probably not possible to perform on-the-fly augmentation.

- Model
    - Try generative models such as CVAE (for a recurrent version see *"Anticipating many futures: Online human motion prediction and synthesis for human-robot collaboration"*).
    - Experiment with Temporal Convolutional Network (TCN) classifier (https://github.com/locuslab/TCN).
    - Try to add Gaussian noise layer.
    - Develop 3 detectors at different scales, setting the window size to 16, 32, 64 frames => (533ms, 1067ms, 2133ms). These scales can be supported by the head gesture duration distributions summarized in the document deep-virtual-rapport-agent/notes/Datasets.docx The predictions from these scale-specific detectors will need to be fused afterwards.
    - If we have more data, use multiple recurrent GRU/LSTM layers. The reasoning why it might be better:
        - https://machinelearningmastery.com/stacked-long-short-term-memory-networks/

        - Given that LSTMs operate on sequence data, it means that the addition of layers adds levels of abstraction of input observations over time. In effect, chunking observations over time or representing the problem at different time scales.
          "… building a deep RNN by stacking multiple recurrent hidden states on top of each other. This approach potentially allows the hidden state at each level to operate at different timescales." — How to Construct Deep Recurrent Neural Networks, 2013
        - Stacked LSTMs or Deep LSTMs were introduced by Graves, et al. in their application of LSTMs to speech recognition, beating a benchmark on a challenging standard problem.
          "… RNNs are inherently deep in time, since their hidden state is a function of all previous hidden states. The question that inspired this paper was whether RNNs could also benefit from depth in space; that is from stacking multiple recurrent hidden layers on top of each other, just as feedforward layers are stacked in conventional deep networks. " — Speech Recognition With Deep Recurrent Neural Networks, 2013

- Evaluation
    - Compare the obtained results from testing on the ccdb with the work "Recognizing Visual Signatures of Spontaneous Head Gestures" that uses the FIPCO dataset and also evaluates the developed head gesture detectors on the CCDb dataset. However, they evaluated only on the *basic set* of the CCDb. They did not use the *extended set* where there might be 2 distinct annotations at a time (by annotators A1/A4). So when comparing to their work, we need to use only the *basic set* from CCDb, as we have currently performed.