**DVRA / MultimodalRapportVirtualHuman - Model design ideas, proposals, thoughts**

[very useful] *“A Recipe for Training Neural Networks”* [*http://karpathy.github.io/2019/04/25/recipe/*](http://karpathy.github.io/2019/04/25/recipe/)

INPUT

* Audio of speaker
  + Use raw
* Video of speaker
  + Extract facial landmarks & head pose
    - Or higher level: FACS, nods/shakes ...

AUDIO MODEL

* Google’s VGGish

OUTPUT - animated virtual human

* 2 independent channels of responses:
* Audio
  + None
  + Agreement “Uhm / ok”
  + Disappointment “oh no”
  + Laugh “hm, hm”
  + **Take turn**
* Behaviour / visual
  + None
  + Head nod
  + Head shake
  + eye-gaze
  + ? lean fwd / bwd
  + Facial:
    - surprise / happy (smile/laugh) / thinking / confusion

**Predictions/outputs**

* ML-based
  + Attend - gaze
  + Smile T/F
  + Nod with and without paraverbals
* Rule-based
  + Turn taking - based on pause / use of another model
  + Over speaking - as in VRA2

1. **VH = listener only**

* Trained on: Speaker AV -> Listener AV

1. **VH = interviewer (when listening as above; + detects when to ask predefined Qs)**

* Trained on: Speaker AV -> Listener AV

**Points**

Look at data - distributions, cooccurrences

Quasi-monolog - questions asked, Interviewer setting

MFCC sensitive to microphone, so look at prosody, energy and pitch [see 1st VRA paper for speech features]

Verbal - language from ASR

Look at NVB and SEWA databases for head nod detection - if you need to train your own head nod detector!!! -> to generate labels for the main project!

Input head rotations x,y,z directly to the final model

Face / head - AU 6+12, headpose, eyegaze - average of 2 eyes

* from OpenFace, ask Ed, detected (vita4vets - attention measure)

Kinect head pose vs Openface - ask Kalin

Try GRUs for head nods

**Can check self-disclosure paper ICMI 2019**

* VGGish and MFCC for audio
* ResNet video frames
* BERT for text
* Fusion ...

**Multi-task learning**

* Overview: “An Overview of Multi-Task Learningin Deep Neural Networks” <https://arxiv.org/pdf/1706.05098.pdf>
* Related work
  + “Learning Off-line vs. On-line Models of Interactive Multimodal Behaviors with Recurrent Neural Networks” <https://hal.archives-ouvertes.fr/hal-01609535/file/dan_PRL2017_R2_v2.pdf>
    - jointly model speech, gaze and gestures of two subjects involved in a collaborative task
    - predict the instructor’s co-verbal gestures and region of interest fixated by the instructor’s gaze given his verbal activity and the interlocutor’s gestures
    - use multi-task LSTM
    - perform both on-line and off-line prediction using LSTM and BiLSTM models respectively
    - (for evaluation) perform coordination histogram to capture global coordination patterns between different modalities given synchronous streams of discrete events
* Train NN to do more tasks at the same time
* Multiple outputs in last layer
* Sum up losses
* I.e. multilabel classification?
* Earlier features in the NN can be shared = power of multi-task learning
* Robust to missing labels for certain tasks/outputs (sum losses only for tasks whose labels are present)
* **Pros:** 
  + Benefit from having shared lower-level features
  + Share knowledge from data between the tasks
  + Usually good, if amount of data for
* **Cons:** 
  + Worse than having separate NNs only if you can’t have sufficiently big NN
  + <https://www.youtube.com/watch?v=UdXfsAr4Gjw>
* Example
  + <https://blog.manash.me/multi-task-learning-in-keras-implementation-of-multi-task-classification-loss-f1d42da5c3f6>

**Hierarchical classification**

* <https://www.kdnuggets.com/2018/03/hierarchical-classification.html>
* <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=7C0EC326B47A60B9159A6BF00B4AD105?doi=10.1.1.183.302&rep=rep1&type=pdf>
* Options
  + Flat classifier - ignores class hierarchy
    - Discriminate btw all classes (leaves), then assign classes going up the tree hierarchy
  + Local classifier
    - Binary classifier per node
    - Multi-class classifier per parent node
    - Multi-class classifier per level of the tree hierarchy
  + Global classifier / big-bang
    - Single complex model
    - Many methods
* => Not needed here

**Reinforcement learning**

* Inverse RL to estimate reward function

then direct sparse RL methods to train simulator

* INVERSE RL
  + Based on execution traces - infer reward f()
  + unsupervised/semisupervised
  + main problems:
    - For most observations of behavior there are many fitting reward functions. The set of solutions often contains many degenerate solutions, i.e. assigning zero reward to all states.
    - The IRL algorithms assume that the observed behavior is optimal. This is a strong assumption, arguably too strong when we talk about human demonstrations.
* Or Apprenticeship Learning