OXOLO

SADtalker repository modifications

Anastasios Tsourtis

Deliverables

- Repository: https://github.com/tasoskrhs/SadTalker

  (branches: **main** & **Task_5__60fps** (task 5)).

- Collab notebook: SADTalker_demo.ipynb
- Shared gDrive folder:

  https://drive.google.com/drive/folders/1ClacjLCZoA2XLxWhFQBX3HzgRFKF-SIX?usp=sharing
- Notes/Ideas in this presentation.

Pipeline high-level overview: preprocess

1. Input video/image. First frame is kept for video
   Modification: `source_image_flag`=False → all video frames are kept.

2. Crop image/1st frame and subsequent frames according to detected head mask (preprocess.py/crop()). Get landmarks for first frame (68,2) → paper fig. B12.

3. Save (color modified) cropped frame for later. Modification: flag in step 1. requires a `break` so that the first cropped image is not overwritten.

4. Get landmarks for all (cropped) frames and then extract 3DMM i.e. (257) texture, gamma… + translation **t** & scale **s** of face
   KEEP: {expression, rotation, transformation, translation} → (70+3)

Pipeline high-level overview: audio

1. (generate_batch.py) Process audio file in 16KHz and hardcode to 25fps → 640Hz/frame.

   Depending on the audio length calculate audio_frames. Get mels based on spectrum. Modification: instead of replicating the 3dmm coeffs based on first frame only → use 3DMM coeffs of all frames (70-D ie coeffs w/out translation)

   Then subsample to 25 fps according to the (hardcoded) audio extraction before… If video is shorter, I replicate last frames.

2. (test_audio2coeff.py) provide 3dmm coeffs to generate:
   i) expression predictions from audio (window of **10 frames ahead** based on: audio & 3DMM coeffs (64-D coresp. to {exp, angle})) → (audio_frames, 64)
   ii) pose from audio: use Decoder of CVAE for 32-mel embedded audio sequences. (audio_frames, 6)

Pipeline high-level overview: Face Renderer

1. (generate_facerenderer_batch.py) replicate 3DMM coeffs around a semantic radius of ±13 for middle frames and 27 repeats for the 1st frame.
   Modification: use available 3DMM coeffs for first 27 frames instead of repeat.

   Expression coeffs are multiplied by a constant, and since the output head is "still" we overwrite translation and pose by the reference 1st frame.

2. (make_animation.py)
   Get canonical key_points based on face2vid detector model and the 1st image → `kp_canonical`
   Get mapping based on 3DMM coeffs → `he_source`
   Get transformation based on canonical keypoints and mapping
   Iterate over generated 3DMM coeffs (based on audio) and a use `OcclusionAwareSPADEGenerator` model with inputs: their mapping & transformation in order to get **predictions** on each frame (image: 256x256x3).

Pipeline high-level overview: Face Renderer

(2. cont') Modification: changed pose 3DMM coeffs to account for all reference ones (audio2pose.py).

3. (animate.py) Resize generated frames and compile video.
Append audio to video and add background based on initial crop pixels.

Modification: (animate.py & generate_batch.py) Change to 60 frames (hardcoded) instead of 25.

Notes on Task 4 (not carried out): Eyes Region

Landmarks (out of 68) corresponding to: {left eye, right eye, left eyebrow, right eyebrow} → {[36:41], [42:47], [17:21], [22:26]}.

My initial idea would be to overwrite these coefficients by the reference frame (either 1st or all frames) instead of using the predictions of the cVAE model.