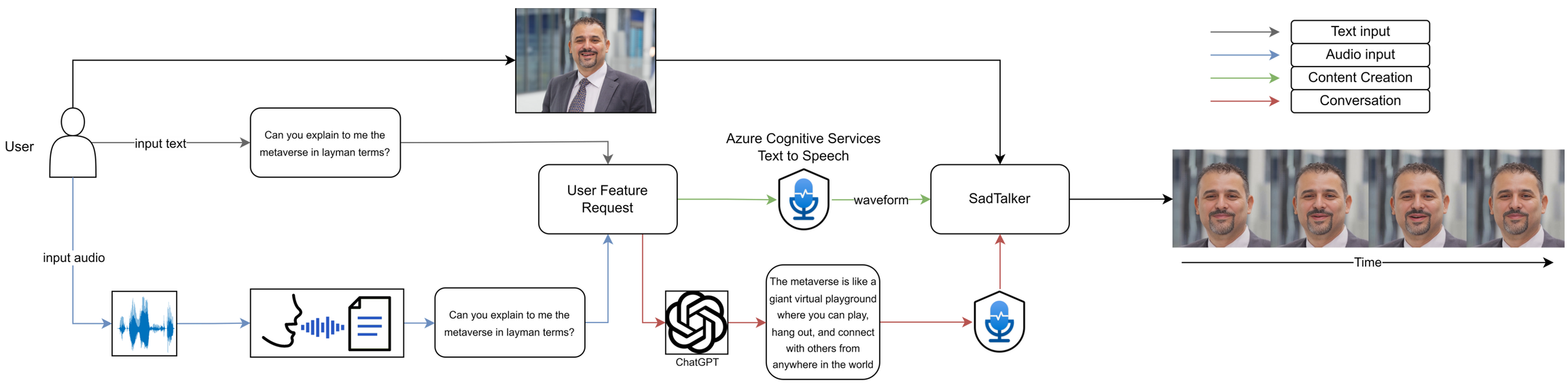
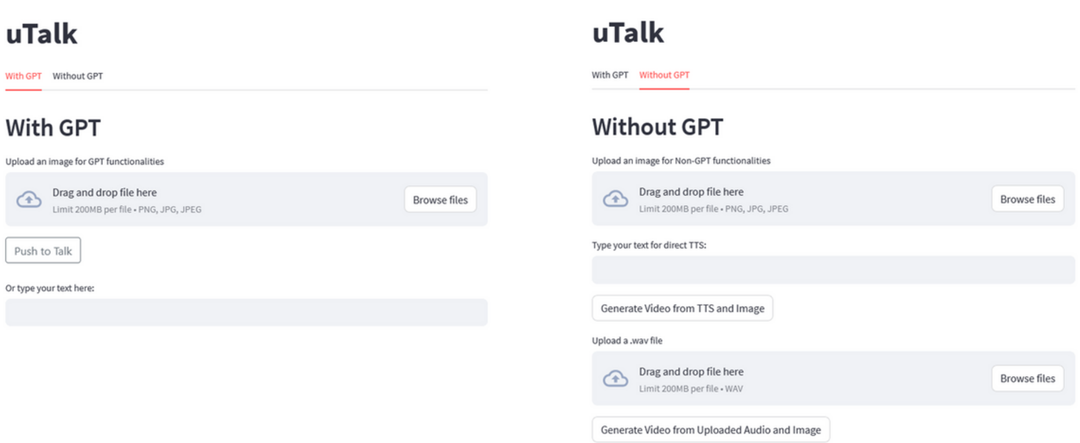
uTalk: Bridging the Gap Between Humans and AI

Hussam Azzuni * §, Sharim Jamal * §, Abdulmotaleb Elsaddik * †
* Mohamed Bin Zayed University of Artificial Intelligence
† University of Ottawa
§ indicates equal contribution

System Architecture



User Interface



Contribution

- Framework with an integrated and optimized SadTalker and APIs for Human-Computer interactive avatar.
- Optimizing the run-time of SadTalker by **27.69%**
- User Interface (UI) offering audio and text inputs for conversing with AI or creating content.
- Providing FPS adjustment feature for video generation.
- Applying the context of the two previous questions and answers enhances the user experience.

Motivation

- **Responding to Industry Needs:** Addressing the growing demand for efficient, AI-driven solutions in various sectors.
- **Revolution in Digital Interactions:** Enhancing Human-Computer Interaction through LLMs and Visual models.
- **Bridging Technology and Accessibility:** Ensuring the benefits of AI are accessible and user-friendly for a broader audience.

Applications

- **Customer Service:** Enhance customer interactions with AI-powered chatbots and digital twins.
- **Innovative Content Creation:** Empower users to generate unique content using AI assistance.
- **Language and Accessibility Tools:** Break down language barriers and improve communication with translation and accessibility features.

Modifications

Removing Redundant code:
- **Tqdm library:** Only used to monitor progress.
- **Intermediate values:** Saving and loading across the entire system.

Adjustable FPS:
- **Custom FPS:** Reducing FPS without affecting the user experience.

Enhancing facexlib efficiency:
- **Input size:** "FaceRestoreHelper" to 256x256, instead of 512x512.
- **Gaussian noise:** Removed.

Integration of SadTalker with Streamlit:
- **Modular operations:** Modularizing the framework.
- **Two distinct modules:** Initialization, and generation modules:
  - Crucial to integrate these modules into Streamlit.
  - Streamlit caching to preload models when Streamlit is loaded.
  - Reduces run-time as the system is more resource-efficient.

Results

Model	Modifications				FPS	Run-time (seconds)
	Tqdm Removal	Facexlib Optimization	Removed Intermediate Values	Replace Mimsave		
SadTalker	X	X	X	X	25	40.637 ± 0.320
Proposed mod1	✓	X	X	X	25	39.930 ± 0.116 (**-1.74%**)
Proposed mod2	✓	✓	X	X	25	31.182 ± 0.526 (**-23.27%**)
Proposed mod3	✓	✓	✓	X	25	31.438 ± 0.579 (**-22.64%**)
Proposed mod4 (uTalk)	✓	✓	✓	✓	25	29.385 ± 0.284 (**-27.69%**)

FPS	Mean Score (SD)	Min	Max	FPS	Mean Score (SD)	Min	Max
16	2.83 (1.10)	1	5	21	3.14 (1.03)	1	5
17	2.21 (1.21)	1	5	22	2.97 (1.15)	1	5
18	3.48 (0.87)	1	5	23	3.71 (1.15)	1	5
19	2.83 (1.10)	1	5	24	3.62 (1.08)	1	5
20	3.66 (1.01)	1	5	-	-	-	-

Subjective Study: Opinion of participants on the smoothness of 9 AI-generated videos with varying FPS (16 -> 24 FPS).
Participants: 29 participants
Conclusions:
- Videos with less than 18FPS are perceived as less smooth.
- Videos with 20, 23, and 24 FPS performed well.
- Videos with 17, and 22 FPS received lower scores, possibly due to generated head movements.

Model	FPS	Run-time (seconds)	Model	FPS	Run-time (seconds)
Experiment 1	25	29.385 ± 0.284	Experiment 6	20	25.041 ± 0.104
Experiment 2	24	28.525 ± 0.199	Experiment 7	19	24.196 ± 0.217
Experiment 3	23	27.833 ± 0.137	Experiment 8	18	23.026 ± 0.131
Experiment 4	22	26.842 ± 0.189	Experiment 9	17	22.134 ± 0.131
Experiment 5	21	25.899 ± 0.301	Experiment 10	16	21.241 ± 0.137

Ablation Study: Effect of video FPS on inference run-time.
Conclusions: Obtain a 14.88% speedup compared to 25FPS.