Here's a structured **report template** comparing different **lip-syncing approaches** for a **VRM model** using **custom video/image and audio**. This report evaluates multiple methods, including **First-order-model, Wav2Lip, D-ID API, and Unity Unreal engine (pro method) DeepFaceLive**,discussing their steps, pros/cons, and outputs.

📄 REPORT: Avatar model Approaches

📌 Project Title: Choosing the Best Lip-Sync Approach for VRM Models

---

1️⃣ Introduction

Lip-syncing technology allows facial animations to match spoken words, making it essential for **avatars, VRM models, and AI-generated videos**. Various techniques exist, from AI-driven solutions like **D-ID API** to open-source alternatives such as **Wav2Lip and DeepFaceLive**.

This report explores **three** different methods, detailing their workflow, advantages, disadvantages, and a comparative analysis to determine the best approach.

2️⃣ Approaches for Lip-Syncing

1. First-order-model

🔍 **Overview:**
FOMM is an AI-driven animation model that can generate motion from a single image using a driving video (i.e., face movements).

📌 Steps to Implement:

1. **Load the checkpoints from repo** to the code.
2. **Provide an source image** to generate the avatar.
3. **Also provide the reference video** to generate a lip-synced video.
4. **Download the processed video** once completed.

✅ **Pros:**
✔ **Full-face animation, not just lip-syncing**.
✔ Works **offline**.
✔ Can animate **any static image**.

❌ **Cons:**
✖ **Needs a driving video** for animation.
✖ Less accurate for **precise lip movements**.

✖ **checkpoint path vox.pth file is corrupted**.

🔹 2 Wav2Lip (Local AI Model)

Trail Code : https://github.com/shardsnaik/AI-Avatar_/blob/main/wav2lip_trail.ipynb

🔍 **Overview:**
Wav2Lip is an **open-source** deep learning model that enables precise lip-syncing by analyzing audio waveforms and matching facial movements.

📌 **Steps to Implement:**

1. Install Wav2Lip using `pip install git+https://github.com/Rudrabha/Wav2Lip.git`.
2. Provide a **video (face)** and **audio file**.
3. Run `python inference.py --checkpoint_path checkpoints/wav2lip_gan.pth --face video.mp4 --audio audio.wav`.

✅ **Pros:**
✔ **Works offline** (No API needed).
✔ **Highly accurate lip movements**.
✔ Open-source and **free**.

❌ **Cons:**
✖ **Requires a powerful GPU** for real-time processing.
✖ Setup is **complex**,

✖ works good only for zoomed target face (less accurate for fully body picture)

✖ **No built-in facial expressions**, only lip movement.

📌 **Sample Output:**

- Lip-synced **MP4 file** with frame-perfect mouth movements.


- **Failing to full size image**

  Sample output 👇👇

  https://drive.google.com/file/d/1PIrsoulhmOdUqWqFET0xDyV17CYXlHe6/view?usp=sharing

  Note : It can be adjusted by hard coding but chance of crash in different env

---

🔹 3 D-ID API (Cloud-Based AI)

Trail Code: _https://github.com/shardsnaik/AI-Avatar_/blob/main/d-id_trail.ipynb

🔍 **Overview:**
D-ID provides an **AI-based API** that can lip-sync an image or video using custom audio. It is a cloud-based solution requiring internet access.

📌 **Steps to Implement:**

5. **Upload an image or video** to D-ID's cloud storage.
6. **Provide an audio file** or generate text-to-speech (TTS).
7. **Send an API request** to generate a lip-synced video.
8. **Download the processed video** once completed.

✅ **Pros:**
✔ No need for local GPU processing.
✔ High-quality and **realistic facial animations**.
✔ Supports **TTS & different voice providers** (Google, Microsoft, ElevenLabs).
✔ **Easy API integration** into websites & apps.

❌ **Cons:**
✖ Requires **API key** and has **usage costs**.
✖ Limited control over **expressions & movements**.
✖ **Internet-dependent** processing.

📌 **Sample Output:**

Through Api call:

https://drive.google.com/file/d/1VJ816toUyf1XW4xQjCcKo1vJGsczB6Lh/view?usp=drive_link

Through Website Interface (d-id)

https://drive.google.com/file/d/13bUK3rCZatV20T_VyvAwYi39wFZy8F20/view?usp=sharing

---

🔹 4. DeepFaceLive (Real-Time Lip-Syncing)

🔍 **Overview:**
DeepFaceLive enables **real-time** lip-syncing using **deepfake AI** and webcam-based tracking.

📌 **Steps to Implement:**

1. Install DeepFaceLive on a Windows/Linux machine.
2. Load a **VRM avatar or video face**.
3. Select **audio input (live speech or pre-recorded audio)**.
4. Apply **real-time facial tracking and lip-syncing**.

✅ **Pros:**
✔ **Real-time processing** (useful for streaming & live avatars).
✔ Supports **VRM models & custom avatars**.
✔ Can be **combined with VR chat & metaverse applications**.

❌ **Cons:**
✖ **Needs a high-end GPU** for smooth results.
✖ **Setup is complex** (requires deepfake model training).
✖ Not ideal for **pre-recorded, high-quality** outputs.

📌 **Sample Output:**

- Real-time lip-synced avatar via webcam/microphone.
- **Best for**: **Live streaming, VTubing, & virtual events**.

🔹 5. Using Unity Unreal engine (Production Grade Approach)

🔍 **Overview:**
To create a production-grade real-time lip-syncing avatar like this one, you'll need several components working together::

📌 **Steps to Implement:**

1. ****3D Character Model**

- Create a high-quality 3D model using software like Blender, Maya, or ZBrush

- Design detailed facial topology with focus on mouth and facial muscles

- Implement proper rigging with blend shapes/morph targets specifically for speech

2. **Facial Animation System**

- Develop phoneme-based viseme sets (visual representations of phonetic sounds)

- Implement blend shape/morph target controllers for lip movements

- Implement audio-to-viseme mapping algorithms

3. **Real-Time Audio Processing Pipeline**

- Build real-time audio analysis system for phoneme detection

- Select a real-time rendering engine (Unity, Unreal Engine)

- Implement proper lighting and shading for photorealistic appearance

Assumption of Pros and Cons

✅ **Pros:**
✔ **Real-time processing** (useful for streaming & live avatars) with Production Grade result.
✔ Supports **VRM models & custom avatars**.
✔ Can be **combined with VR chat & Real time audio phoneme**

✔ Easy to maintain by connecting various pipeline together

✔ Additionally **integrate ml, methods to learn from past data.**

❌ **Cons:**
✖ **Needs investment, time team and high-end GPU** for smooth results.
✖ **Setup is complex** (requires deepfake model training).
✖ Need good **3D designed model**

✖ **Need blend shape setup using c-sharpe/C# or by unity to make high realistic lip sync to avatar**

📌 **Sample Output:**

3️⃣ Comparison of Approaches

Method	Processing Type	Internet Required?	GPU Required?	Ease of Use	Accuracy	Best For
D-ID API	Cloud-based AI	✅ Yes	❌ No	⭐⭐⭐⭐	⭐⭐⭐⭐	Pre-recorded, high-quality avatar videos
Wav2Lip	Local AI	❌ No	✅ Yes	⭐⭐⭐	⭐⭐⭐⭐⭐	Offline, frame-perfect lip sync
DeepFaceLive	Real-time	❌ No	✅ Yes	⭐⭐	⭐⭐⭐	Live streaming & VTubing
Unity Unreal engine	Both real-time and cloud bases	Flxible	✅ Yes	⭐⭐⭐	⭐⭐⭐⭐⭐	Best for Real world, Production grade taks

4️⃣ Conclusion & Recommendation

After evaluating all approaches, **each method serves a different purpose**:

✔ **Use D-ID API** if you need **quick AI-generated videos with minimal setup**.
✔ **Use Wav2Lip** for **offline & high-quality AI-driven lip-syncing for only face**.
✔ **Use DeepFaceLive** if you require **real-time VTubing or streaming avatars**.

✔ **Use Unity Method** if you require **real-time High-quality for production** .