# Interactive Video Editing using Machine Learning Techniques

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Masters of Science*
*in*
*Computer Science and Engineering by Research*

by

Anchit Gupta
20171041
anchit.gupta@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
November, 2022

International Institute of Information Technology
Hyderabad, India

# CERTIFICATE

It is certified that the work contained in this thesis, titled "Interactive Video Editing using Machine Learning Techniques" by Anchit Gupta, has been carried out under my supervision and is not submitted elsewhere for a degree.

| | |
|---|---|
| Date | Adviser: Prof. C V Jawahar |
| Date | Adviser: Prof. Vinay P Namboodiri |

To, my family and friends

# Acknowledgments

I would like to express my deepest gratitude to my advisors, Prof. C V Jawahar, Prof. Vinay P. Namboodiri, and Prof. Makarand Tapaswi, for their constant guidance and support throughout my Master's journey. Prof. Jawahar always emphasized to see the "bigger picture" of the problem and stressed the importance of being able to explain the problem to others in a clear and simple manner, which often made me think more deeply about the problem statement. Prof. Vinay provided many valuable insights and exciting ideas for solving the problem. Having these discussions and receiving guidance has allowed me to become a better researcher. It is a gift I will always cherish.

It would be remiss not to thank Rudrabha, my project partner and friend, for constantly helping and guiding me throughout my Master's, from explaining numerous concepts to helping me with code implementation or helping me regarding Ada. Furthermore, I would like to thank my CVIT lab mates and coauthors Faizan, Sindhu, Zeeshan, Madhav, Ravi, Seshadri, Soumya, Aditya, and Bipasha for making my time in the lab memorable and fun. In addition to having fun and laughter, we also had interesting discussions about one another's projects. I am also grateful to Rohitha and Varun for helping with administrative tasks. I also thank Language and Content Editing Support for proofreading my papers and thesis.

The acknowledgments would be incomplete without thanking my friends Neel, Pulkit, Anush, Manan, and the "Babbar Shers," who not only helped me with my research work but made my five years of college experience one to cherish.

Finally, I would like to express my gratitude to my parents and sister for their selfless support and affection.

# Abstract

There is no doubt that videos are today's most popular content consumption method. With the rise of the streaming giants such as YouTube, Netflix, etc., video content is accessible to more people. Naturally, video content creation has also increased to cater to the rising demand. In order to reach out to a wider audience, the creators dub their content. An important aspect of dubbing is not only changing the speech but also lip synchronizing the speaker in the video. Talking-face video generation works have achieved state-of-the-art results in synthesizing videos with accurate lip synchronization. However, most of the previous works deal with low-resolution talking-face videos (up to $256 \times 256$ pixels), thus, generating extremely high-resolution videos still remains a challenge. Also, with advancements in internet and camera tech more and more number of people are able to create video content and that too in ultra high resolution such as 4K ($3840 \times 2160$). In this thesis, we take a giant leap and propose a novel method to synthesize talking-face videos at resolutions as high as $4$K! Our task presents several key challenges: (i) Scaling the existing methods to such high resolutions is resource-constrained, both in terms of compute and the availability of very high-resolution datasets, (ii) The synthesized videos need to be spatially and temporally coherent. The sheer number of pixels that the model needs to generate while maintaining the temporal consistency at the video level makes this task non-trivial and has never been attempted in literature. We propose to train the lip-sync generator in a compact Vector Quantized (VQ) space for the first time to address these issues. Our core idea to encode the faces in a compact $16 \times 16$ representation allows us to model high-resolution videos. In our framework, we learn the lip movements in the quantized space on the newly collected 4K Talking Faces (4KTF) dataset. Our approach is speaker agnostic and can handle various languages and voices. We benchmark our technique against several competitive works and show that we can achieve a remarkable $64$-times more pixels than the current state-of-the-art!

Now, how to edit videos using the above algorithm or any other deep learning algorithm? To do so, the person has to download the source code of the required method and run the code manually. How amazing would it be if people could use the deep learning techniques in video editors with a click of a single button? In this thesis, we also propose a video editor based on OpenShot with several state-of-the-art facial video editing algorithms as added functionalities. Our editor provides an easy-to-use interface to apply modern lip-syncing algorithms interactively. Apart from lip-syncing, the editor also uses audio and facial re-enactment to generate expressive talking faces. The manual control improves the overall experience of video editing without missing out on the benefits of modern synthetic video generation

algorithms. This control enables us to lip-sync complex dubbed movie scenes, interviews, television shows, and other visual content. Furthermore, our editor provides features that automatically translate lectures from spoken content, lip-sync of the professor, and background content like slides. While doing so, we also tackle the critical aspect of synchronizing background content with the translated speech. We qualitatively evaluate the usefulness of the proposed editor by conducting human evaluations. Our evaluations show a clear improvement in the efficiency of using human editors and an improved video generation quality.

# Contents

# List of Figures

# List of Tables

*Chapter 1*

# Introduction

Video is the most common form of entertainment for the majority of the people. With affordable smartphones and internet facilities, people have access to the video content library of the world. Many of these videos contain a speaker talking, termed ”talking face videos,” like movies / TV shows, interviews, educational lectures, etc. While most of the videos are produced in English, many people aren't native English speakers. People will only appreciate the content when they will understand the language of the content.

**How often do people want to see videos in their regional/native language?** While most of the video content is produced in English, almost 80% of the world does not have English as their native language [1]. According to a study, even in India, around 93% of YouTube viewers prefer watching content in the Indic language, [2]. This study illustrates how important it is to produce video content in languages other than English. However it is not feasible to recreate the already existing content in English to desired language, and this approach to translating video content will not be scalable.

**Ways to make the video content accessible to more people?** The current methods of making video content accessible to more people is by providing subtitles in the native language or dubbing the content. The former first requires getting the transcript of the video and then translating the transcript to the desired language. Given how much video content is available, this task cannot be performed at a large scale in multiple languages by only a human. Algorithms such as Automatic Speech Recognition (ASR)[2, 26, 4, 13, 52] and Neural Machine Translation (NMT) [49, 36, 37, 35] are there to help perform these tasks automatically, but these require human intervention as these are not correct all the time. Although adding subtitles will help people understand the content, the overall experience of hearing subtitles in one language and viewing the content in another is sup-par. The latter method, Dubbing, is taken a step further, and the original audio is replaced with the dubbed audio. The process of dubbing a video presents numerous challenges

- Getting the transcription and translation discussed above is one of them

[1] https://blog.google/technology/area-120/aloud/

[2] https://www.businessworld.in/article/Over-20M-Viewers-In-India-Streaming-Youtube-On-TVs-Consumption-Of-Indic-Language-Content-Rises/16-09-2021-404909/

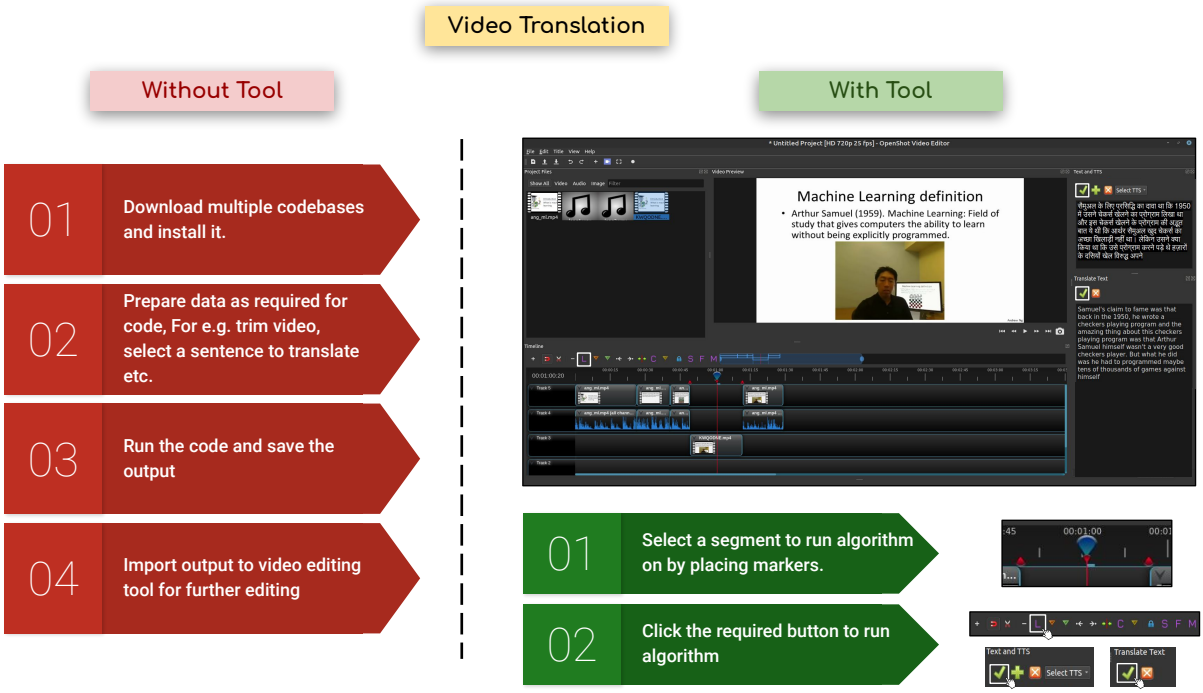


**Figure 1.1** Shows the steps involved in translating the video with and without using our video editor tool.

- Not everyone can speak all languages. If the content is to be dubbed in different languages, only someone who knows the language can speak the translated transcript, making dubbing harder.
- Often, the length of translated audio and the original audio mismatch because of the difference in languages. Audio and video content can go out of sync if not done carefully.

To overcome the problems in dubbing, one can use Text to Speech (TTS)[43, 38, 42, 41, 25] models for the narration of the translated transcripts and for content synchronisation, one can either alter the speed of audio/video or use ML algorithms to generate synthetic video frames to fill the gaps due to length mismatch (discussed in 2.4.4).

**Given all the above-mentioned ML algorithms to help us in the process of translating video content, how does one use these algorithms to translate?** To use these algorithms, the person must first download and install the code.These algorithms usually run on the entire video instead of a portion of it. Translating a video involves splitting the video into chunks rather than processing the whole thing at once. To do this, the person must trim the video and audio to the required length first and then run the code. Although these steps may seem trivial to someone with programming and machine learning knowledge, it can be very difficult for those without these skills to utilize them.

Despite dubbing the video, the viewing experience remains incomplete. The unsynchronized lips and the translated audio makes the dubbed video look unnatural even though we can understand what is

said. However, there exist many deep learning works which can generate lip-synchronized videos that are in sync with given audio like [40, 39], generate synthetic talking head videos using only a source image and an audio [69] or a driving video [46]. Although all these methods help a great deal, they also have similar shortcomings. In addition, those methods generate outputs in resolutions only up to 256x256 pixels, which does not work well with videos produced today, given the advances in camera technology. Now, even smartphones are able to record 4K videos ($3840 \times 2160$).

## 1.1 Contributions

The contributions of the thesis are as follows:

- We propose a video editor tool with various SoTA deep learning algorithms integrated into it.
  - We replace downloading and running the code using the traditional command line with just a few button clicks.
  - We provide an option for the user to stay in the loop in the automatic translation of the videos. The SoTA ASR and NMT are yet to match human capabilities, thus, the user can correct any mistakes these algorithms make.
  - We simplify the process of using talking head generation algorithms. These algorithms typically work on an entire video rather than just a segment. In our video editor, users can select video segments by placing markers at the beginning and ending timestamps and then edit the segments with the desired algorithm. This is the usual way to edit videos - by breaking a large video into chunks and editing each separately.
  - Our tool enables everyone to use machine learning algorithms (by making it intuitive to use) to edit videos, even if they are unfamiliar with machine learning or programming.
  - We conduct a user study to measure our tool's effectiveness in reducing the manual effort to edit and improve the quality of videos.
  - By using our tool, it will be possible to translate videos at scale since anyone can use it, and less time is required to edit and translate the video.
- We propose a model which can precisely lipsync the modern high-resolution videos.
  - We collect the first-ever 4K talking face dataset. We have 140 videos in the dataset amounting to 30 hours of data.
  - We obtain a novel quantized generative pipeline that decodes *images and face meshes*. The generated images in the generative pipeline are used to learn accurate lip-synchronization using appropriate discriminators in the quantized latent space. The face mesh is used to obtain ultra-high resolution image generation.

- Overall our generated faces contain $64$-times more pixels than the current $96 \times 96$ output from Wav2Lip [39]
- Our model is speaker-independent and preserves the pose and expression of the speaker.
- We show various applications where our model can be used.
- We also explicitly design an invisible watermarking scheme to address the ethical concerns and reduce the potential misuse of our model.

We discuss more about contributions in detail in 2 and 3.

## 1.2 Organization of Thesis

The rest of the thesis is organized as follows.

- In Chapter 2, we discuss how state-of-the-art deep learning algorithms can be interactively used in an open source video editor - OpenShot.
- In Chapter 3, we discuss the issues in the current lipsync approaches and how it is not usable on the high-resolution video content produced these days. We design an architecture where we learn lipsync in a quantized space that is much smaller in size compared to the image space, making it possible to generate higher quality outputs.
- Chapter 4 presents the concluding thoughts.

*Chapter 2*

# Intelligent Video Editing: Incorporating Modern Talking Face Generation Algorithms in a Video Editor

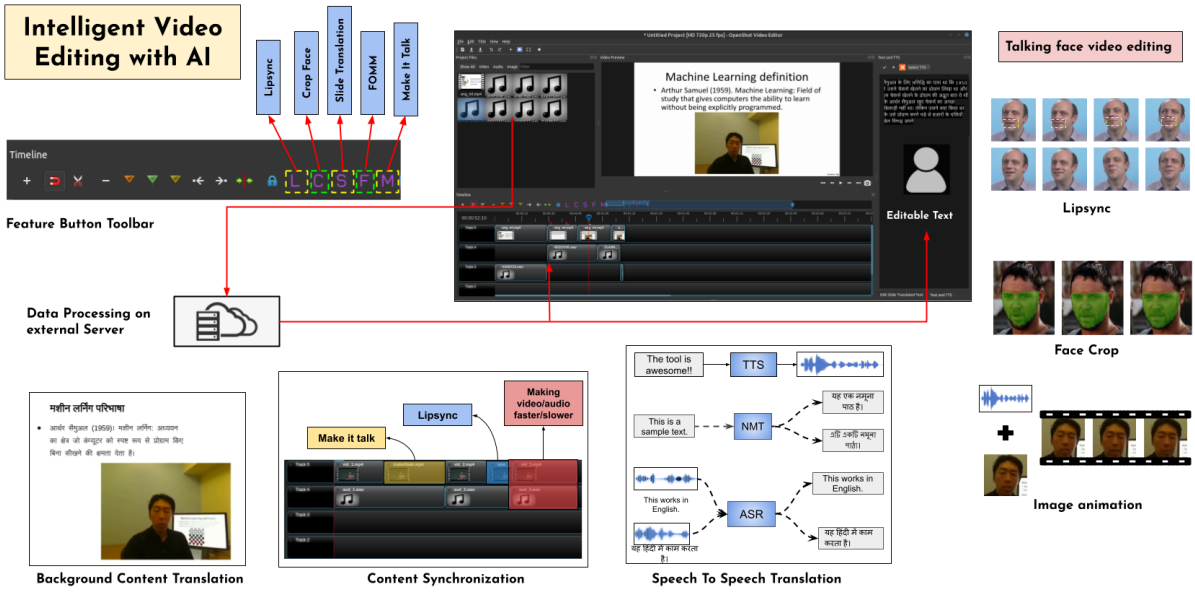


**Figure 2.1** Our video editor aims to bring together the latest advancements in talking face generation algorithms in an interactive manner. The tool provides ample manual control over several aspects of these algorithms leading to better quality video generation. Our tool can be used extensively to translate lectures and dub movie scenes into various languages.

[0]Demo video link for the paper: https://youtu.be/AhnSAVl9f8w

## 2.1 Introduction

In today's world, humans are more connected than ever due to social networking phenomena leading to tremendous opportunities for information sharing. Advancement in internet connectivity has led to a massive increase in the consumption of online video materials. These include movies, talk shows, educational lectures, and vlogs. Most of this content is present in languages like English, making the content less popular among the local populace across countries.

Dubbing is the most popular method to make the content available for consumption in different languages, but it produces unnatural visual content due to out-of-sync lip movements. Our work introduces the users to an easy-to-use framework built on an open-source video editing tool OpenShot [32], to reduce the manual effort while correcting out-of-sync lip movements of complex movie scenes. We add video editing features to lip-sync the speaker's lip movements according to the dubbed speech. In the presence of multiple speakers, our tool allows the human editor to select the speaker to lip-sync manually. We avoid artifacts by using facial key-point detection to replace any remnant out-of-sync lip movements. We allow the users to manually remove any undesirable frame or chunk of the video according to their discretion. Additional features for the generation of expressive talking faces from only audio and facial reenactment are present in the tool, providing users with creative ways to edit videos. The manual control in the tool helps in both the experience of editing and generating better quality results.

On the unavailability of dubbed speech, a pipeline to generate translated speech was proposed in LipGAN [40]. However, the automatic pipeline had severe limitations due to a lack of manual control. This motivated us to bring together the best from translation and talking face features in a tool that incorporates the automatic features of a face-to-face translation system with controllable manual editing. To translate the original speech, we use an automatic-speech-recognizer(ASR) [52] followed by neural-machine-translation(NMT) systems [36] to get the translated text. Speech is generated using the text-to-speech algorithm [25] from translated text. The easy-to-use interfaces allow the user to correct transcription and translation errors.

Out-of-sync problems arise when replacing the original speech with the automatically translated speech. This out-of-sync content often gets cumulated when translating long videos, and automatic techniques like [40] prove inadequate in such scenarios. We, therefore, propose various methods of synchronizing content, discussed in Section 2.4.4, helping to preserve the quality of the lecture. We also use this opportunity to include an additional feature overlooked in LipGAN [40] by automatically translating the background text. Similar to other features, the background translation also allows for manual intervention to improve the accuracy of the translated content. Our tool can help create translated versions of famous courses in Indian languages, making an ever-lasting impact.

We do extensive human evaluations to prove the effectiveness of the tool. Demo video[1] clearly explain the process of editing videos using the tool and showcasing multiple results.

[1] https://youtu.be/AhnSAVl9f8w

## 2.2 Current Deep Learning based tools

Text2Vid [64] is a text-based tool for editing talking-head videos by manipulating the wording of the speech and non-verbal aspects of the performance by inserting mouth gestures like "smile" or changing the overall performance style. DeepFaceLab [34] and FaceSwap [14] are open-source deep learning-based video editing tools for achieving photorealistic face-swapping results. Yanderify [15] is a wrapper around first-order-model [46] that exposes a simple user interface for anyone with any level of technical skill.

The above works focus majorly on one problem, which makes them unsuitable for editing videos for translation. We present a video editing tool with multiple SOTA deep learning features available with one click of a button, allowing easy and high-quality video editing.

## 2.3 Talking Face Video Editing

Visual presentation of a talking person requires generating image frames showing the speaker in various views while pronouncing various phonemes. The creation of realistic content synthetically is a very active field. While most of these works have been published, and often the implementations are also publicly available, these works are far from being widely used in real-world applications. The generation of a talking face consists of many sub-problems that are attempted separately without connecting the sub-problems. We observe that the automated versions are unsuitable for video editing due to the editor's lack of control. For starters, the standalone algorithms work on entire videos. However, in real-world situations, these algorithms need to be used often in short temporal segments of a larger video. The algorithms may also work on spatial crops of a video instead of the whole frames. In such cases, the larger video needs pre-processing, which involves cropping and trimming the portion of interest and processing it. The editor then needs to insert the processed output back into the original video. To avoid using external pre-processing options, which are tedious and hard to integrate, we provide the option to the user to select the segment which needs to be processed interactively. We also provide extensive manual controls for trimming, cropping, and splitting videos. Similar options are also provided for the audio to select portions of audio to be fed into the required algorithms. We provide multiple state-of-the-art features in our editor to generate talking face features and generate realistic results by using them with our tool. Please note that these features can be accessed interactively via the GUI interface.

### 2.3.1 Generating Talking Heads from Audio

Generation of expressive talking-head videos from a single facial image with audio as the only guiding input is a challenging problem that can transform film-making and video editing in many ways. Algorithms like [67, 50, 24, 48] have tried to tackle this problem. The main challenges include synchronizing audio and facial movements and generating multiple talking heads conveying different personal-

ities. Finally, such algorithms need to generate lip-synced talking faces along with the complete set of head motion, lip motion, expressions only with audio, and an identity image as input.

We choose the current state-of-the-art MakeItTalk [69] because of its ability to disentangle the speech content and speaker identity information in the input audio signal. It is then used to generate lip movements, facial movements corresponding to the expression, and the rest of the head poses. This disentanglement leads to plausible results and is considered one of the best systems around for this problem. Furthermore, the system was publicly released and is based on the PyTorch [33] library making it easier to integrate.

We add this feature to our tool for various reasons. In translation systems, there is often a mismatch of length between the audio and video. When audio length exceeds the video's length, makeittalk can be used to fill the required gap by generating a synthetic video of the speaker. Furthermore, the same can be extended to education lectures where the professor is not visible in the slide while teaching. This improves the user experience by a huge margin. Using MakeItTalk for video editing can lower the camera recording time and potentially reduce studio costs.

The user can select the audio segment in our tool by placing the markers over start and end positions. The user can place the cursor at the frame to select the images used for this feature. After the source image and audio are selected, the user needs to click on the *MakeItTalk* button, and the video generated gets displayed in the tool. The generated video is also stored temporarily. The user can place the generated video at the required timestamp and then export the final video.

### 2.3.2 Lip-syncing Talking Face Videos to a given Audio

Lip-syncing talking faces for given audio requires modifying lip movements without changing other characteristics like head pose and emotion. Algorithms like [10, 40, 39] are designed to morph the lip movements of speakers according to a guiding speech. These algorithms input a speech segment and a talking face video segment of any identity to morph the speaker's lip movements, matching the input speech with high accuracy.

We choose Wav2Lip [39] due to its higher quality and superior performance. It is significantly more accurate than the previous works on this problem that can handle any arbitrary speech and identity. Furthermore, Wav2Lip works for videos in the wild, making it highly usable in various situations. One of the prominent use-cases of Wav2Lip is in lip-syncing dubbed movies and translated lectures. However, the automatic code-base again proves inadequate for editing complex movie scenes, and manual control becomes necessary.

**2.3.2.0.1 Which portions of the video to lip-sync?** Lip-sync is required only at the places where a face is present in the video. To select that portion of the video, the user can mark the starting and ending timestamp of the video by placing the markers. The user can lip synchronize the segment by clicking the *Wav2Lip* button. Like the MakeItTalk feature, the generated output is stored temporarily and can replace the non-lip synchronized segment.

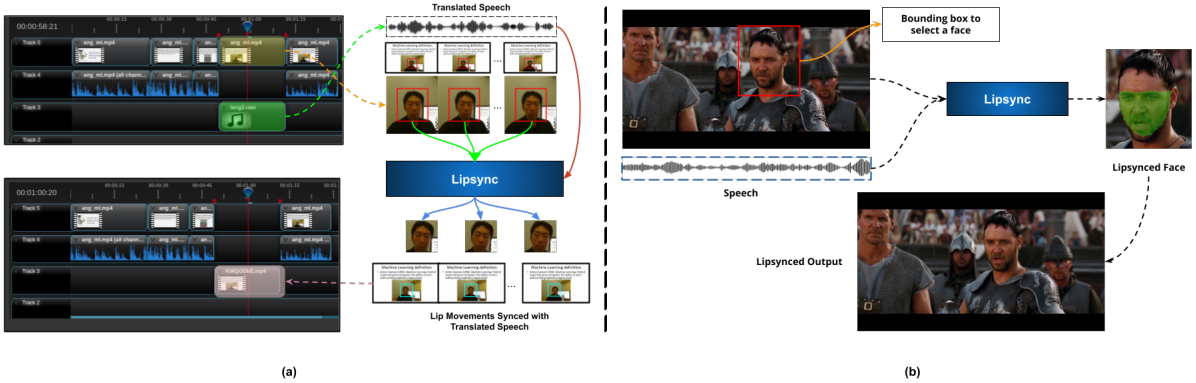


**Figure 2.2** Demonstration of the Lip Sync feature in the tool. Figure (a) shows the video segment selected by the user to lip synchronize. Figure (b) shows the selection of a particular face in a frame to lip synchronize and the replacement of lip synchronized face region encased by the facial key points in the original frame.

**2.3.2.0.2 Which face to lip-sync?** The Wav2Lip [39] lacks a mechanism to pick out a single face from multiple faces that can be lip-synced. To tackle this problem, the user can click the *Select a face* button. The user can select a bounding box around the required face in the video frame, generating better outputs and decreasing inference time.

**2.3.2.0.3 Pasting the resulting face crops** The existing algorithms replace rectangular regions around the face, which works decently in most cases. Still, the artifacts are significantly visible when used for movies, and the output quality is reduced. To solve this, we detect the face region using facial keypoint detection and replace only that portion of the face in the video as illustrated in Figure 2.2, producing lesser artifacts and higher quality output.

### 2.3.3 Face Re-enactment to a Guiding Video

The task of face re-enactment consists of synthesizing videos automatically by extracting appearance from a source image and motion patterns from a driving video. Works like [19, 7, 8] have tried to tackle this problem; however, they all depend on pre-trained models to extract the information like keypoint locations.

Monkey-Net by [45] was the first object-agnostic model for image animation but suffered from poorly modeled object appearance transformation. This issue was tackled in FOMM [46], which proposes using a set of self-learned key points and local affine transformations to model complex motions and claims to outperform state-of-the-art image animation methods significantly.

We choose to include [46] in our tool for numerous reasons. The generation of visual content by animating objects in still images serves many applications in movie production. The translation of

educational lectures serves as an essential application of content-synchronization (discussed in Section 2.4.4). In cases where translated audio is longer than its corresponding video, this feature can be used to generate video frames to sync the audio and video. The synchronization is required for a short segment of video where the mismatch is present. Still, the standalone algorithm takes an entire video, and hence the processing of a short segment becomes a tedious task. In our tool, we tackle this by allowing the user to select a driving video segment which provides the motion. The user selects this video segment by placing two markers, one at the start and the other at the end. The user also places the cursor at the frame, which is selected as the source image. After the selection is made, the user can click on the *FOMM* button to generate the result. After the result is generated, it is displayed in the tool, and the user can easily drag this segment to fill the gap between the audio and video segments.

### 2.3.4 Editing a dubbed movie using Wav2Lip

OTT content these days is the most significant source for entertainment, with every significant streaming platform having a plethora of options in many languages. However, a user is not free to consume whatever content they prefer because of the language barrier. Two heavily used methods to address this issue are to either view subbed content or dubbed content. The subbed content relies heavily on the user's ability to continue reading the dialogues while also focusing on the video, which is not ideal. In the dubbed content, the unsynchronized lip movements make the content unnatural. To address these issues, we show how our tool can be used to generate translated content with lip sync.

Figure 2.3 shows a movie clip where multiple faces are present with a single speaker speaking in presence of background music. Wav2lip [39] is unable to produce quality results in such cases mainly because of two reasons:

- In the presence of multiple faces, there is no mechanism to select a face to be lip synchronized.
- The presence of background music makes it harder to generate accurate lip movements.

The user can select the speaker by drawing a bounding box around the facial region to overcome the first problem. Wav2lip [39] then generates the lip movements for the speaker selected by the user. This selection reduces the search area for face detection resulting in faster and accurate inferences. For the second problem, the user has an option to separate dialogues from background music [57]. Using only the separated dialogue as speech input for Wav2lip [39] generates more accurate lip movements.

## 2.4 Translating Lectures from Language $L_A$ to Language $L_B$ with a Human in the Loop

Given a lecture video in language $L_A$, we aim to translate the lecture in language $L_B$. Our tool provides an interface consisting of different modules using which a user can easily translate the lecture

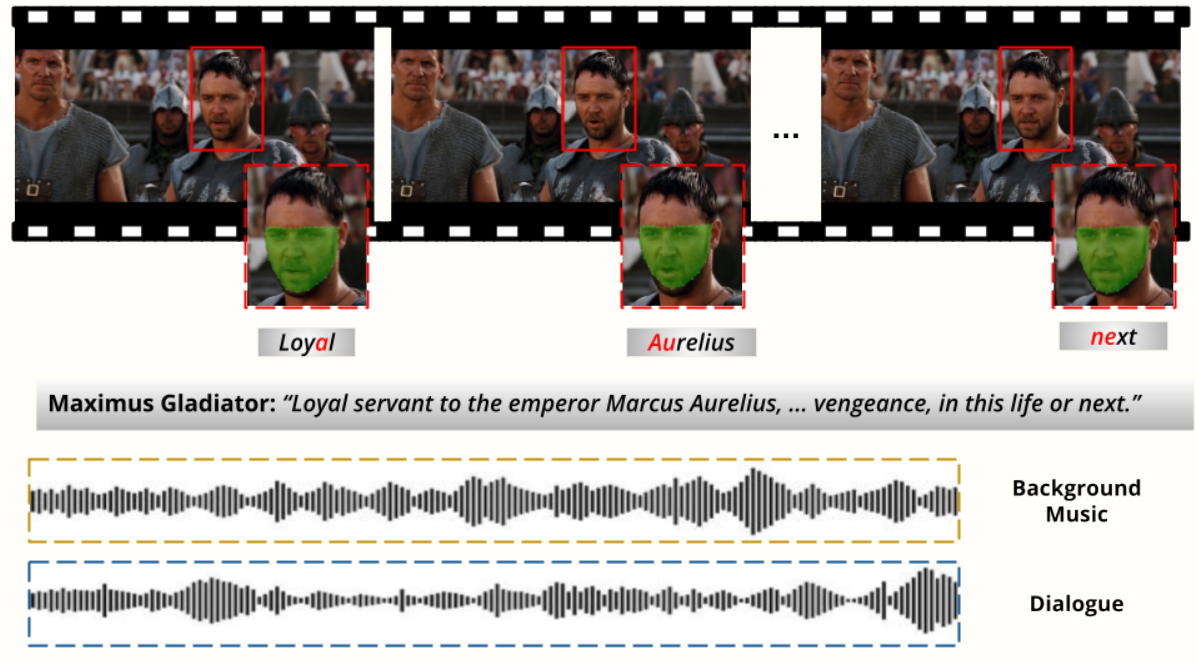


**Figure 2.3** Demonstration of the lipsync module working with the tool. The lipsync module processes the region in the bounding box drawn by the user. For better results, we only replace the facial region and not the entire bounding box enclosed by the facial key points to reduce the artifacts. Separation of background music from the audio results in clearer audio, lip-syncing using which improve results.

from one language to another. We begin with the translation of speech which follows the steps discussed by lipgan. However, we improve the automatic system by including manual control and editing capabilities, as shown in Figure 2.4, in a systematic manner that leads to better quality translation. The length of the translated speech often varies from the video length, leading to out-of-sync content, corrected using the tool as discussed in Section 2.4.4. The additional challenge of background content translation for lectures having slides is discussed in Section 2.4.5.

### 2.4.1 Automatic Speech Recognition

Automatic speech recognition (ASR) is the capability that enables machines to process human speech into a written format. Speech recognition is an essential module in video translations. There has been significant research in the area of automatic speech recognition. We use popular works on Automatic Speech Recognition (ASR)[2, 26, 4, 13, 52] in our tool to obtain the corresponding text from speech.

We directly use transcripts when they are available (often present in YouTube) for a lecture and provide the user with an option in the tool to use Google's ASR [2] or Amazon Transcribe [26] when the transcripts are unavailable. While these services are highly accurate and dependable, they are accessible


**Figure 2.4** Block diagram of speech to speech translation. The transcript in language $L_A$ is obtained from the speech using the ASR module, and then the text generated is translated into language $L_B$ using the NMT module, which is editable by the user. The speech in language $L_B$ is obtained from the translated transcript using the TTS module.

only for a trial period; hence we also have integrated an open-source ASR [52] into the tool. SileroModels can handle four languages: English, German, Spanish and Ukrainian. SileroModels works reasonably well on any 'in the wild' audio with sufficient SNR and reports WER of 11.5 on LibriSpeech test set. We also provide users the option to edit the text manually. This helps the user correct transcription errors caused by the ASRs and correct even spoken language mistakes by the original speaker. We generate the output from the ASR of the user's choice and display it in an editable text box. The editor can modify the textual annotations, which is then fed to the NMT module discussed in Section 2.4.2.

### 2.4.2 Neural Machine Translation

Neural Machine Translation(NMT) is an end-to-end learning approach that involves translating text from one language to another. The use of neural networks for NMT was first introduced in Sequence to sequence learning with neural networks [49] using an end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. This was followed by the attention mechanism used in RNNsearch [6], which improved the performance of such systems further. The rise of transformers [56] has lead to massive gains since their introduction in 2017. This seminal work led to the meteoric rise of NMT, and soon hundreds of languages were covered. The Indian languages were also explored widely during this phase. Works like [36, 37, 35] aimed to train models on major Indian lan-

guages and open-sourced them. Our editor includes Amazon Translate [1] and Google Translate [62] for translating English into other Latin languages. For translating Indian languages, we use the state-of-the-art system developed by Jerin [35]. However, it is widely known that the best NMT systems are still far from matching human capability and thus require much manual correction. This is especially true when dealing with lectures that contain keywords that cannot be translated.

We acknowledge this wide gap between the state-of-the-art NMT models and a human translator. We, therefore, provide the user with an option to edit the results from our NMT module. Similar to the ASR outputs, the NMT outputs are also displayed in an editable text box (Figure 2.4). The user can correct errors and ensure that the correct translations are passed to the text-to-speech module for generating the translated speech. The user can also optionally add delimiters like "comma" and "full-stop" to improve the naturalness of the generated speech.

### 2.4.3 Text-to-speech

Text-to-speech (TTS) involves generating speech from text inputs. The field of text-to-speech has seen tremendous advancement in recent years. Several works [43, 38, 42, 41] have tried to tackle synthesizing speech with neural networks. Tacotron2 [43], a neural text-to-spectrogram conversion model uses Griffin-Lim for spectrogram-to-waveform synthesis. Deepvoice3 [38] uses a fully convolutional character-to-spectrogram architecture enabling parallel computation for faster training that competitive architectures using recurrent cells lack. These works produce high-quality speech on the LJSpeech [21] dataset and are comparable to the original human voice. But when trained on Indian regional languages, the results produced are far worse and deemed unusable. Another issue is the sub-par speech quality of the above-mentioned TTS while synthesizing results on longer sentences.

Thus in our tool, we use GlowTTS [25], a flow-based generative model for parallel TTS that internally learns to align text and speech by leveraging the properties of flows and dynamic programming. Flow-based TTS' are robust and perform considerably better on longer sentences. We also find that it generalizes well on Indian languages compared to [38, 43]. The user can synthesize the speech directly from the translated text written in the text box by selecting the desired TTS model and clicking the *TTS* button. We also provide a stand-alone option to synthesize speech by loading the text from a file.

### 2.4.4 Content synchronization

While translating or dubbing a video, the speech and video often go out of sync, resulting in losing out on the experience of the original lecture. This causes a bigger problem in educational lectures as most educational lectures use slides to explain various concepts. Furthermore, if the speech and video are out of sync, the translated speech may end up getting overlaid to the wrong slides, causing significant issues in the viewer's understanding.

Editing a longer video is a tedious task; we segment the video into smaller chunks to make it simpler. Following this, each segment can be synchronized independently. In the end, the user can combine all

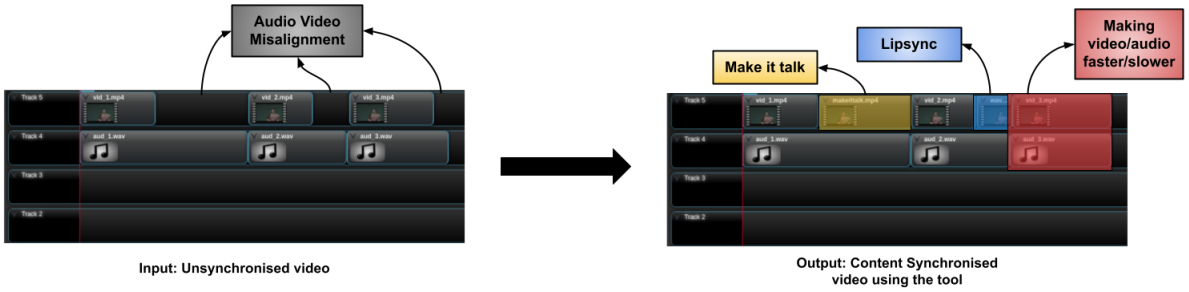


**Figure 2.5** illustrates the different features like generation of talking heads from audio, lipsync and speed manipulation of audio/ video present in the tool used to synchronize a misaligned video.

the segments to generate the final result. The whole procedure is done interactively. The different features used for content synchronization of the audio/video segments have been illustrated in Figure 2.5.

- The user can manually place the audio or video at any timestamp to correct the sync.
- The user can alter the audio or video length by manipulating the speed of that segment to match the length of the corresponding audio or video segment.
- Using Section 2.3.1, 2.3.2, and 2.3.3 the user can generate synthetic video frames to fill the gap caused due to differences in audio and video length.

### 2.4.5 Background Content Translation

Background content translation involves translating the content present in the video in language $L_A$ to language $L_B$. It is an essential step in video translation. We use a combination of OCR and NMT modules for this task. OCR, or optical character recognition, is one of the earliest addressed computer vision tasks. Deep learning techniques [53, 22, 60, 5] have significantly improved the accuracy and are used for industrial applications. Pytesseract [29] is an open-source python wrapper for Google's Tesseract-OCR [53] Engine used for this task.

We include LayoutParser [44], a python library, in the tool, which is built on top of Pytesseract. LayoutParser provides an additional benefit of "line detection" over Pytesseract's word-level detection, which helps improve the content translation.

Educational lectures heavily make use of slides for teaching. To explain a single slide, the instructor may take several seconds or even minutes, keeping the slide constant for the whole period. We exploit this prior in our tool for achieving background content translation. The user can select the video segment's starting and ending timestamp of constant slide/background in the editor. The user then gets the translations of the content into a text box by clicking on the *Background Content Translation* button.

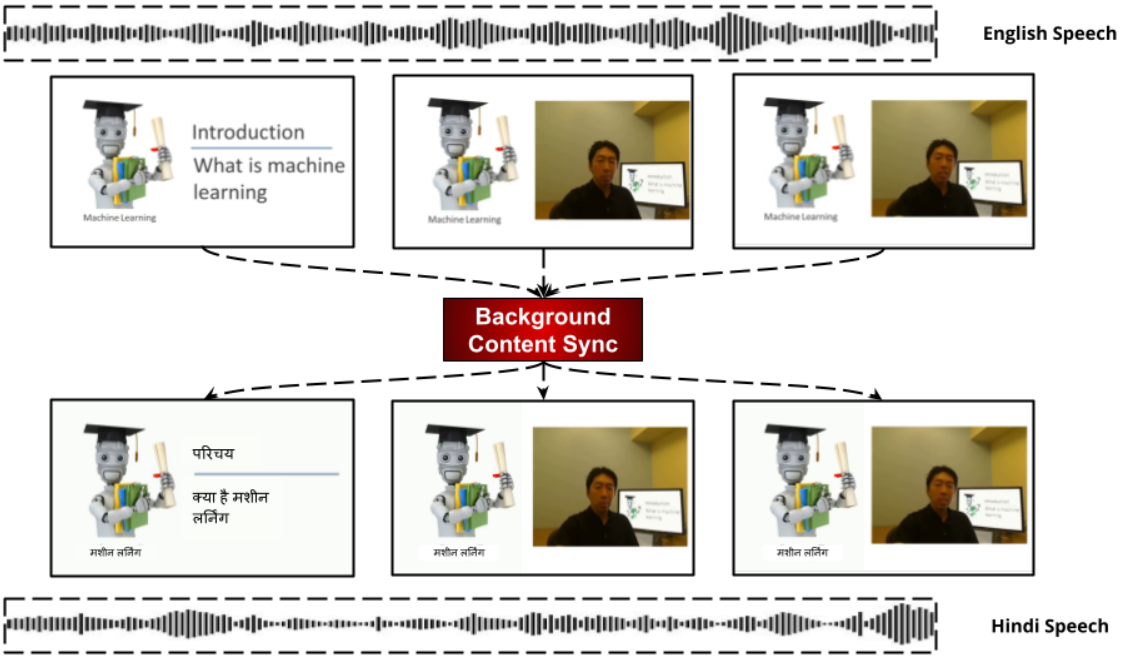


**Figure 2.6** This figure shows the use of the Background Content Synchronization feature, using which the content of the slide in language $L_A$ is translated into language $L_B$.

The automatic translation is obtained from Section 2.4.2 and is improved by human translators via editing the translated text. After the user finalizes the text, the user can click the 'Overlay on slide' button to replace the translated text into the slide. This feature achieves results as shown in Figure 2.6.

### 2.4.6 Combining the Steps to Translate a Lecture

Translating a lecture by segmenting it into multiple smaller chunks is easier than entirely translating at once. Our tool significantly reduces the manual effort required to translate the lecture by enabling the user to process chunks individually.

The first step in translating a lecture from language $L_A$ to $L_B$ is to translate the speech from $L_A$ to $L_B$. The ASR module present in the tool helps get the speech's transcript in language $L_A$. The transcript is then translated into $L_B$ by the NMT module. With manual control, the user gets to correct the errors in automatic translations. The translated text is then converted into $L_B$ speech by using the TTS module. The output obtained from the TTS module in language $L_B$ is often different in length than the original speech in language $L_A$. This difference can result in misalignment of the audio and video segment of the lecture. To avoid such misalignment, the user can use MakeItTalk, FOMM, and speed manipulation features. With this, the user can achieve translated audio-video synchronized content.

The user can also translate the lecture slides' content wherever present, from language $L_A$ to $L_B$ using Section 2.4.5 with manual control to correct the translation errors due to automatic translation.

The final step in the translation is to lip synchronize the lecture with the output speech using the Lipsync feature.

## 2.5 System Details

OpenShot [32] is a python based tool implemented in the PyQt library. We have added the above-discussed features in the tool, accessible to the user using a button click. For every feature, the audio and video segments selected by the user are extracted from the original video and are then processed. The tool takes the same amount of time to generate the outputs as the original works. The generated outputs are temporarily stored and displayed in the tool for that particular session, which can replace the original video segment or add to fill the gaps due to audio and video duration mismatch. The user can specify the quality and fps for saving the video. The time required to save the video is proportional to the duration and quality of the video, where an hour-long 720p video takes nearly 2 to 3 minutes to export. To make the process of installation easier for the user, we provide an alternative where the computation takes place on a remote server so that the user does not have to worry about the required deep learning packages.

## 2.6 Evaluations

In this section, we conduct two types of experiments to understand the impact of our tool on both the human editors and viewers who consume the final videos. We first understand how far our tool helps humans to edit videos in Section 2.6.1, followed by a comprehensive user study on 25 participants to rate the quality of the generated outputs through manual editing.

### 2.6.1 Comparing Manual Effort

This section compares the time taken to edit the video with and without using our tool. Five video segments of 10 minutes were edited by 5 participants, (a) with and (b) without using our tool. We recorded the time taken in both cases for (i) Audio translation (ii) Background Content Translation (iii) Content Synchronization. The average time (in minutes) for these tasks is reported in Table 2.1.

While using method (b), users spent maximum time on editing tasks like splitting videos into smaller segments, processing each segment using automatic algorithms, and loading the processed segments back into the tool. Since these standalone algorithms provide no manual control, there is no option to correct the mistakes that occur at different stages of translating the lecture, resulting in inaccuracies. Method (a) provides an easy-to-use interface to correct the automatic translations, significantly reducing the time to get correct translations versus when not using our tool. The entire video translation process takes a huge amount of effort and time, which reduces considerably using our tool.

| Method | S2S | Video Editing | BG. Translation | Total |
|---|---|---|---|---|
| External | $33.7 \pm 3.76$ | $48.1 \pm 2.81$ | $17.3 \pm 2.58$ | $99.1 \pm 5.61$ |
| **Our Editor** | $6.8 \pm 0.94$ | $11.9 \pm 1.05$ | $6.2 \pm 0.3$ | $24.9 \pm 1.68$ |

**Table 2.1** present the comparison of the time taken to translate lecture videos using our tool and stand-alone automatic systems + external editors. The time are mentioned in minutes. S2S: Speech-to-Speech translation, Video Editing: Using talking face features and general cropping/trimming actions, BG Translation: Translating the background content. The last column denotes the total time taken on average by the editors.

### 2.6.2 User Study to Rate the Quality of Results

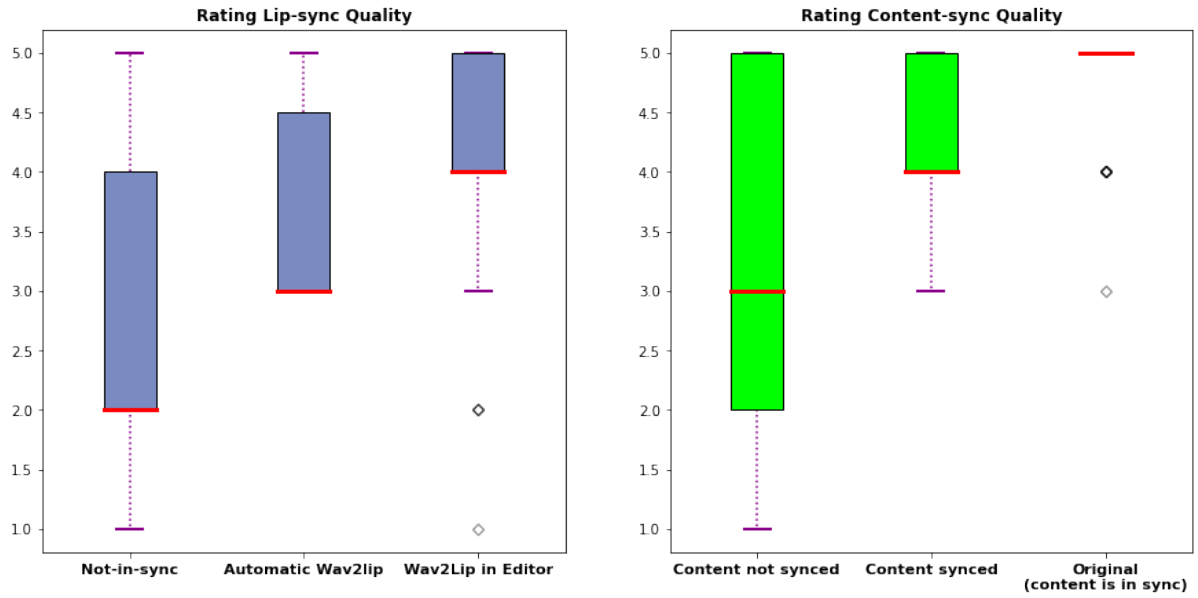


**Figure 2.7** The boxplots show the distribution of values recorded from the experiments conducted. The left plot records the lipsync quality, and the right plot records the content synchronization quality.

Users can use the tool to generate visual content for direct human consumption. Hence, we subject the results generated from the tool directly to human evaluation. The participants are asked to evaluate the quality of lipsync in lectures and dubbed movies. The participants were also asked to rate the translated lectures in terms of content sync. For both these tasks, we ask the users to rate between 1 to 5 where 1 is the lowest rating, and 5 is the best possible. 25 users participated in our study. Male:Female ratio was $\approx 1$, and the participants were in the $21 - 30$ age range.

We compare results from our editor with those from the automatic Wav2lip [39] algorithm and the out-of-sync original video. As observed from Figure 2.7 (box plot on the left), the median user rating for the output from our tool was $4$ out of $5$, while automatic Wav2Lip had a median rating of $3$. The not-in-sync output was rated the lowest.

For the second task, we used multiple videos to compare the original video, translated video by simply overlaying the translated speech, and translated video using our tool. As observed in figure 2.7 (box plot on the right), the median rating of our tool was $4$ out of $5$, while simply overlaying the translated text was rated at a median of $3$, and the original videos received a median score of $5$.

Overall, we get conclusive evidence that our tool improves the time required for editing a video and reduces the manual effort while improving the results.

## 2.7 Conclusion

In this paper, we present a version of the OpenShot video editor with multiple new functionalities. We incorporate a speech-to-speech translation pipeline where automatically generated results can be corrected manually. We also include several state-of-the-art talking face generation techniques in our editor, editing complex dubbed movie scenes and translating lectures into a more straightforward task. Our tool can translate background content on the slides and ensure the synchronization between the speech and the background content is preserved using systematic steps. We believe our editor opens up a new paradigm in video editing that allows interactive use of the latest published research and improves the overall user experience.

*Chapter 3*

# Towards Generating Ultra-High Resolution Talking-Face Videos with Lip Synchronisation

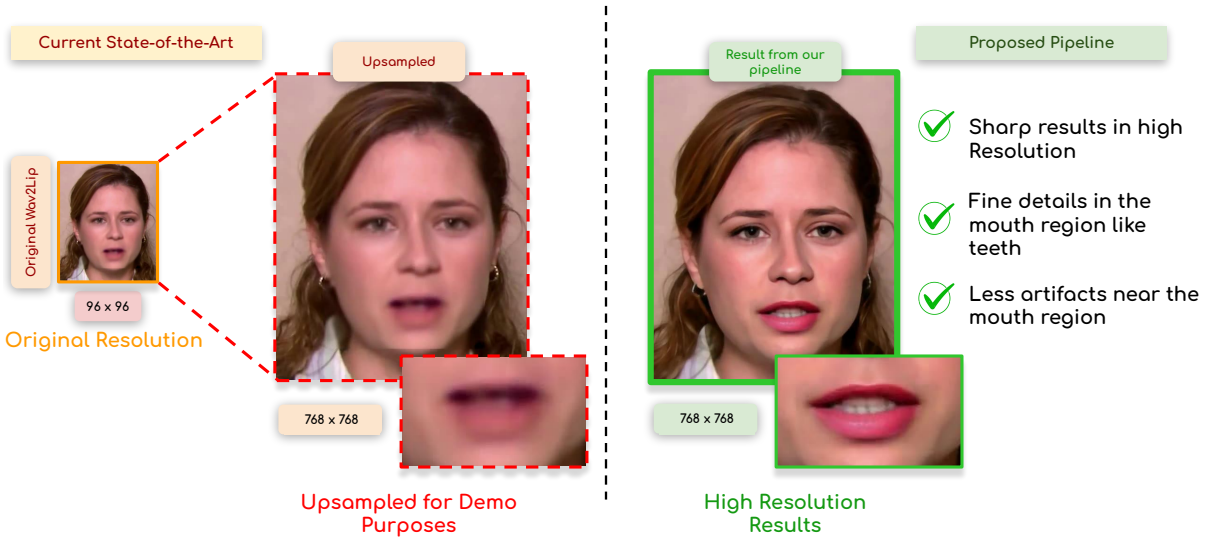


**Figure 3.1** We propose the first talking-face generation network, which can lip-sync any identity at ultra-high resolutions like 4K. Our model captures fine-grained details of the lip region, including color, texture, and essential features like teeth. While the current state-of-the-art model Wav2Lip [39] generates faces at $96 \times 96$ pixels (left part), our proposed method synthesizes $64$ times more pixels, rendering realistic, high-quality results at $768 \times 768$ pixels.

[0]Demo video for the paper: https://youtu.be/l3Lbxz3J_WU

## 3.1 Introduction

When was the last time we watched a video? For many of us, it will be well within 24 hours! In fact, for the majority of people, videos are the most common form of entertainment [1]. The rise of streaming platforms like YouTube and Over-The-Top (OTT) media platforms like Netflix has made video production more accessible to the masses. Such is the impact that over 200 thousand minutes of videos are streamed solely on Netflix every day! Video conferencing is yet another area that has seen a massive influx of users. According to a recent report [2], a video conferencing platform like Zoom enables over 300 million daily meetings amounting to 3.3 trillion minutes per year! Recently, due to the COVID-19 pandemic, the need for online lectures is gaining tremendous user attention. News reading, video calling, vlogs, marketing videos and often a large part of movie scenes contain videos of the speaker. These videos are termed as "talking-face videos". As the overall video content grows, the critical component of talking-face videos continues to grow exponentially. Due to the advancement in internet services and camera technology, most of the videos today are captured and streamed in extremely high-resolution. Resolutions like $3840 \times 2160$ (4K) and $7680 \times 4320$ (8K) are considered to be mainstream and an important requirement for the entertainment industry.

With this growth in international video content, the ability to consistently dub the generated video content based on audio is a new multimedia application. This technology can be used to enable seamless watching of video content in other languages as well as applications such as anonymizing avatars for video conferencing, gaming, and other multimedia applications. However, the major challenge for audio-based visual dubbing has been the lack of scalability of audio-based lip-synchronization approaches. These either failed to generalize easily to multiple identities or, if suited for numerous identities, were unable to generalize to high-quality, high-resolution visual dubbing. Our work aims to solve this challenge comprehensively by enabling high-resolution lip-sync for any identity to a given speech. Before delving into the details, we start by surveying the major branches of current approaches for lip-syncing talking-faces to a given speech.

### 3.1.1 Speaker-Specific Lip-Sync Models

Audio-driven talking-face generation has witnessed tremendous progress in recent years. The first works [50, 27] in this space dealt with large amounts of data of a specific speaker (e.g., President Obama) and trained deep neural networks to learn the speaker attributes. These works showed that it is possible to learn phoneme-viseme correspondence through a neural network. Follow-up works continued to deal with speaker-specific approaches [18, 54, 28, 47] with an aim to reduce the amount of speaker-specific data required for training. While the initial models were trained with over 20 hours of Obama's speeches, the latest approaches only need a few minutes of data per-speaker to generate high-quality

[1] https://www.business2community.com/content-marketing/top-reasons-why-videos-are-a-must-in-your-content-strategy-plan-02130432

[2] https://backlinko.com/zoom-users

results. The basic idea of all the speaker-specific approaches is to train two separate modules. The first module learns correspondence between lip shapes and speech, while a second renderer module generates the final video. Generally, this renderer is trained in a speaker-specific fashion. It is important to note that even though the data required for isolated speakers has significantly reduced over the years, these models still fail to perform for unseen speakers and even for seen speakers with a significantly changed appearance. Further, they also fail to handle dynamic environments like movie scenes consisting of large head motions and lighting variations.

### 3.1.2 Speaker-Agnostic Lip-Sync Models

To learn the lip synchronization for in-the-wild speakers, speaker-agnostic works started gaining importance. These works [10, 40, 39] train on large datasets like LRS2 [11] containing thousands of identities to learn speaker-agnostic characteristics. They can handle unseen identities without requiring additional fine-tuning on speaker-specific data. They also work for various languages, poses, and voices. The current state-of-the-art, Wav2Lip [39] is well known for generating accurate lip-sync for videos of any identity in any language. Wav2Lip uses a standard encoder-decoder architecture that takes the target pose and target speech as input and generates a lip-synced face. A pre-trained lip-sync expert discriminator is used as a critique that penalizes the network for inaccurate lip shapes. However, Wav2Lip generates videos with a resolution of $96 \times 96$ pixels - making it practically unusable in professional videos that often require 4K resolution. We summarize the capabilities of the current models and compare them with our proposed method in Table 3.1.

| **Method** | **Unseen IDs?** | **In-the-wild?** | **High Res.** |
|---|---|---|---|
| Synth. Obama [50] | × | × | ✓ |
| ObamaNet [27] | × | × | ✓ |
| Neural Puppetry [54] | × | × | ✓ |
| LipGAN [40] | ✓ | × | × |
| Wav2Lip [39] | ✓ | ✓ | × |
| **Ours** | ✓ | ✓ | ✓ |

**Table 3.1** Comparison of different lip-sync models. Our model handles the most challenging cases in this space.

Please note that we differ from audio-based talking Head generation works [69, 66, 61], where the aim is to generate the head movements along with lips from speech. Similarly, face re-enactment works [46, 59, 68] use a driving video to transfer the head motion to a source identity. In our case, we only morph lip movements to be in sync with a target speech without altering expressions or head motion, thus we exclude these works in our comparison.

### 3.1.3 Why not train Wav2Lip in ultra-high resolution?

As Wav2Lip [39] is the current state-of-the-art in lip synchronization, the most straight-forward and a natural question that arises is: "can we directly extend Wav2Lip to generate and lip-sync ultra-high resolution videos?" There are two major ways of achieving this: (i) Training Wav2Lip at higher resolutions (like 4K) and (ii) Using state-of-the super-resolution (SR) techniques on top of the current Wav2Lip generations. We observe that using either of these strategies results in sub-optimal generations. There are several key reasons to this. First, the lip-sync expert from Wav2Lip does not converge on high-resolution data from datasets like AVSpeech [16] or our proposed 4KTF dataset. We believe this is directly related to the increased number of pixels that the network deals with, increasing the overall variability. The encoder-decoder structure of Wav2Lip also faces similar issues and does not generate effective outputs. Another major challenge to deal with is the compute and hardware requirements. Training networks to generate videos at such high-resolutions runs into hardware issues. Also, such networks are extremely slow to train and work with small batch sizes, leading to poor performance.

As an alternative, using SR methods to upsample the Wav2Lip outputs is also not an ideal solution. The major reasons being: (i) Although Wav2Lip generates accurate lip and jaw regions, the resultant videos lack fine-grained facial features like teeth, lip color, and face texture (in the generated lower-half of the face). These artifacts magnify when we apply the SR methods to obtain high-quality results; (ii) Wav2Lip generates videos at a resolution of $96 \times 96$ pixels. Upsampling these outputs to ultra-high resolutions like 4K would need video SR methods that can work at high scale-factors (like $8\times$ and $16\times$). However, the existing video SR methods [20, 9] are known to work effectively and generate high-quality results only at low scale-factors like $4\times$.

### 3.1.4 Our Contributions

To address the problem of obtaining ultra-high resolution videos, we modify the existing approaches in the following way: We obtain a quantized generative pipeline that decodes *ultra-high resolution images*. The intermediate quantized representations in the generative pipeline are used to learn lip-synchronization using appropriate discriminators in the quantized latent space. Overall our generated faces contain $64$-times more pixels than the current $96 \times 96$ output from Wav2Lip [39]. Our model works for any in-the-wild unseen identities, languages, and voices (including synthetic text-to-speech voice). Since the existing talking-face video datasets are limited in resolution, we collected a new 4K dataset from publicly available videos on YouTube. Our dataset spans a total of $\approx 30$ hours, covering a diverse set of identities and an extensive vocabulary (see Figure 3.2). We train our model to synthesize high-quality talking-face videos with this dataset in hand.

| YouTube link | Category |
|---|---|
| https://youtu.be/ieXLNVwshRU | Interview |
| https://youtu.be/8LshTX3tILA | Finance |
| https://youtu.be/YVhIIxOeN2E | Makeup/beauty |
| https://youtu.be/QjEYyLFtJcc | Podcast |
| https://youtu.be/dhAmMXCBIcg | Tech Review |

**Table 3.2** Some links from our dataset

## 3.2 4K Talking Face Dataset

Previous datasets like MEAD [58], AVspeech [16] and HDTF [66] have done an incredible job collecting high fidelity data but were limited in terms of resolution. We introduce the 4K Talking Face Dataset (4KTF), a new audio-visual dataset in 4K resolution. Our dataset consists of 140 YouTube high-quality (resolution: 4K) videos, amounting to $\approx$ 30 hours. The left pie chart in Figure 3.3 shows the distribution of the categories of the videos in our dataset. Table 3.2 shows some sample links from our dataset. Collecting such large datasets causes significant challenges in storage and processing. The total size of the dataset containing 30 hours (140 videos) of data is around 200 GB. A dataset like LRS2 [11] with a similar amount of data uses less than 30 GB of storage. We used the YouTube playlist feature extensively for storing videos initially. Candidate videos were screened manually, and videos with issues like loud background music, violent content, etc., were removed. Around 50 hours of video data were finally downloaded and processed for active speaker detection. We use the publicly available codebase of SyncNet [12] for this purpose. We create 30 second chunks of videos and calculate the offset between audio and video streams. We remove chunks that contain an offset of more than $+/-3s$. We also remove chunks containing a face smaller than $256 \times 256$ and finally use the selected chunks to train our algorithms. The videos are of varying lengths, ranging from 40 seconds to 40 minutes, with over 2.5 million frames containing a taking face. The dataset predominantly contains English language videos and has a vocabulary of $\sim$ 10,000 words. The videos are selected from different channels, including technical reviews, interviews, podcasts, educational content and movie scenes. This results in a wide range of topics, a large vocabulary and different speaking styles. Although most of the videos comprise a single speaker, we use active speaker detection [12] for the multi-speaker case to discard the segments in which the visible face and audio are out of sync. In addition, we use the YouTube transcripts to remove the segments containing inappropriate or violent language. We perform face detection using S3FD [65] to obtain the facial crops. At 4K resolution, face detection is not only slower but also surprisingly inaccurate. Therefore we resize the videos by a factor of 4 to perform face detection and then scale the coordinates back to the original resolution. We use the pre-processed videos with the face crops for the pipelines described in the next section. As well, please note that the collected videos are in 4K resolution whereas the face crops are $768 \times 768$ pixels. Different statistics from the

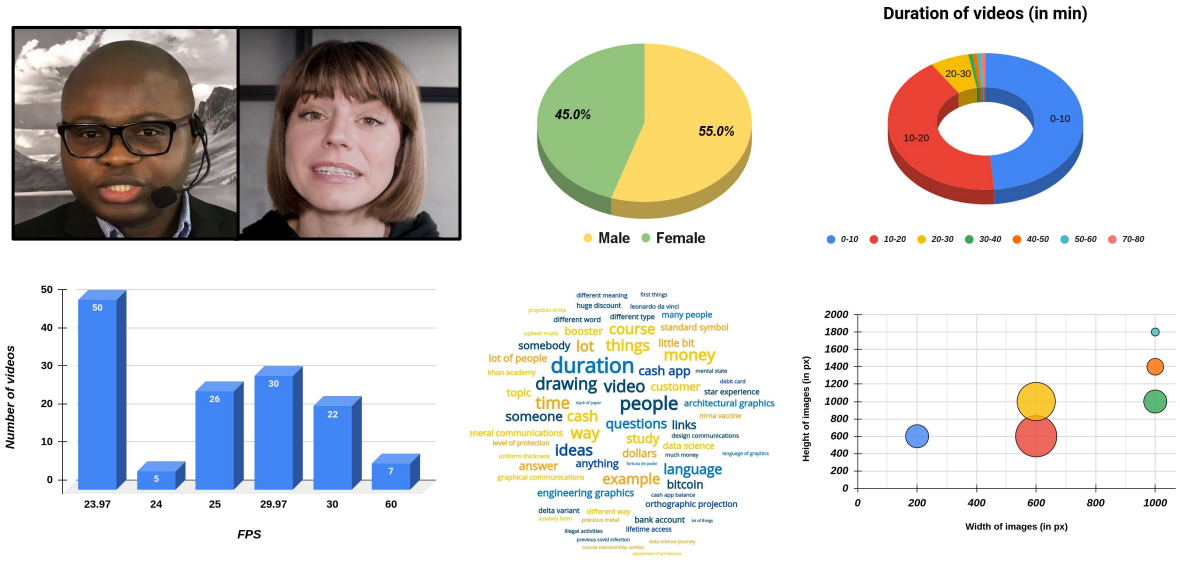


**Figure 3.2** Samples and statistics of our newly collected 4K dataset (videos gathered from YouTube). Our dataset has a nearly equal male-female ratio, contains varying video lengths and FPS, spans a large vocabulary and contains high-resolution frames.

dataset, along with some sample frames, are depicted in Figure 3.2 and 3.3. We use this newly collected data to train all the networks. Since our dataset contains talking-face videos, speech, and automatically generated text transcripts (not used in this work), it will also be useful for several other related problems in the space involving the face, lip movements, speech, and text! We will release the dataset, and the train and test splits to aid future research in the audio-visual field.

## 3.3 Generating Ultra-High Resolution Talking-Faces

Recent advances in high-resolution image synthesis have shown that learning compact vector spaces [17] helps in high-resolution synthesis. Methods like [17] first learn a VQGAN and then use it to generate intermediate quantized embeddings to represent the HD images. Downstream tasks like image-to-image translation, super-resolution, or random image generation are done using the quantized embeddings, i.e., in the quantized space. The final output from such downstream tasks is generated using the VQGAN decoder to convert resultant embeddings into RGB images.

### 3.3.1 Stage-1: Lip-sync Generator

**Representing a Face and Head Pose in Quantized Space:** In our work, we take a leaf out of this strategy and first learn a compact quantized space to represent higher resolution faces. We start

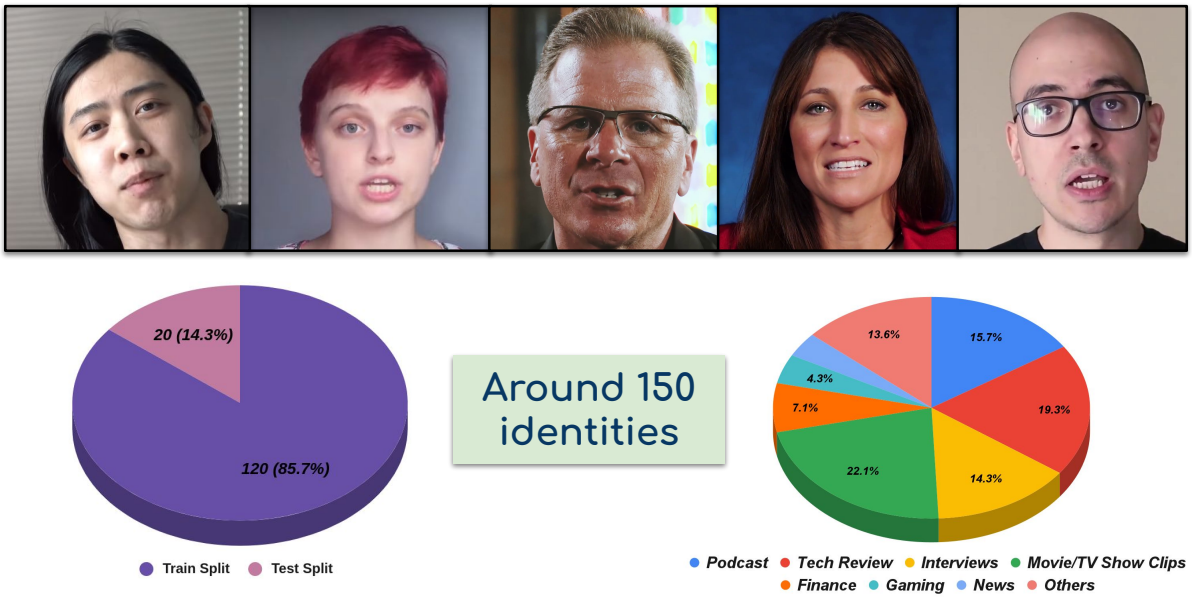


**Figure 3.3** Some additional statistics and samples from our 4KTF Dataset. Our dataset has videos from various channels ranging from podcast and news to gaming, movie clips etc.

by training a VQGAN [17], $V_f$, using the publicly available implementation[3]. The VQGAN encoder converts an input face image, $F_{in} \in \mathbb{R}^{H \times W \times 3}$ to an intermediate embedding $E_q \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times 256}$ through a set of convolution layers. A learnable codebook of $N_c \times 256$ is used to perform vector quantization on $E_{in}$. In our setup, we choose $H = W = 256$ and $N_c = 1024$ as the number of codebook entries. We obtain the vector quantized output $E_q$, which is then passed to a standard VQGAN decoder that reconstructs $F_{in}$ (identical to [17]). Details regarding the losses and hyperparameters can be found in the same.

Similar to Wav2Lip [39] and LipGAN [40], we aim to morph the lip movements of the speaker and not change the target head pose. During training the lip-sync generator, it is paramount not to leak information regarding the mouth shape in the ground-truth face while providing the network with an accurate target head pose. Both WavLip and LipGAN achieve this by masking the lower half of the ground-truth face and conditioning the generator on the speech signal to generate it back. Unfortunately, we cannot directly use this trick in the quantized space. Masking the lower half of $E_q$ does not stop the leakage of mouth information encoded in the top half of the embedding. Thus, we train a separate Pose-VQGAN, $V_p$, with only the top half of the face to avoid any unnecessary leakage. The encoder of $V_p$ ingests a face image with the lower half masked, $F_p \in \mathbb{R}^{H \times W \times 3}$ and outputs a quantized embedding $E_p \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times 256}$. The decoder then learns to generate the input $F_p$ back from the quantized embedding. The network is trained with the losses mentioned in VQGAN [17].

[3] https://github.com/CompVis/taming-transformers

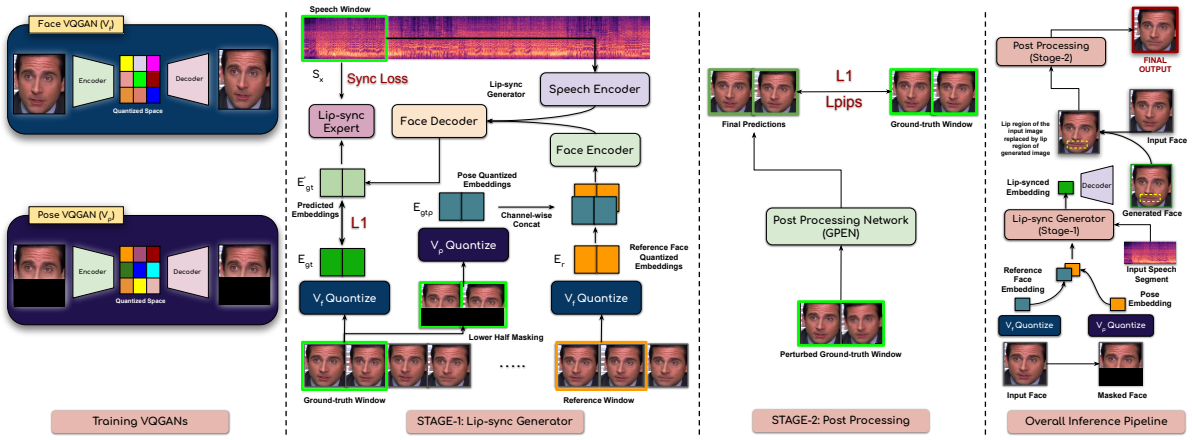


**Figure 3.4** We present our pipeline for generating ultra-high resolution lip-synced videos. We first train Face VQGAN and Pose VQGAN networks (col-1) to encode the faces and head poses in a compact $16 \times 16$ dimensional space. We then train a lip-sync generator in the quantized space and get back the image using the Face VQGAN decoder. (stage-1, col-2). An optional post-processing network is used to improve the quality of the generated outputs (stage-2, col-3). We also show the overall inference pipeline (col-4) to understand our framework better.

Once we have trained both $V_f$ and $V_p$ to encode full faces and head poses, the next step is to train the lip-sync generator in the quantized space. We follow a similar training strategy as that of Wav2Lip [39]. We first train a lip-sync expert which acts as a critic in training the lip-sync generator.

**Training a lip-sync expert in the quantized space:** Our lip-sync expert uses a similar architecture proposed in Wav2Lip [39], but we train our expert in the quantized space $V_f$, rather than the RGB space used in Wav2Lip. The network majorly contains video and speech encoders. The video encoder ingests the quantized embeddings of $T_f$ consecutive frames and outputs a $D-$dimensional vector denoted by $w_f$. The speech encoder takes in a $T_s$ length melspectrogram obtained from the input speech segment and generates a $D-$dimensional vector $w_s$. The final layer of both the encoders are ReLU activated, to ensure the vectors only have positive elements. For training the lip-sync expert, we sample video-speech pairs from the same time step (in-sync, i.e., positive pairs) and random pairs from different time-steps (out-of-sync, i.e., negative pairs). The network is trained using contrastive learning. We calculate the cosine similarity between $w_s$ and $w_f$ and back-propagate a binary cross-entropy loss to train the network. The lip-sync expert is trained on only 25 Frames-per-Second (FPS) videos with $T_f = 25$ frames and $T_s = 1$ second (100 melspectrogram time-steps). We can handle longer sequence length than the original Wav2Lip's lip-sync expert (trained with 5 frames and 200ms of speech) due to the smaller memory footprints of the quantized embeddings compared to images (see Section 3.5.2 for a comparative analysis). Once the lip-sync expert discriminator is trained in the quantized space $V_f$, the next step is to train the generator network, which we discussed below.

**Architecture of the Generator:** Our generator network comprises three components: (i) face encoder, (ii) speech encoder and (iii) face decoder. Both the face and the speech encoders output $256-$dimensional embeddings. These are concatenated to form a $512-$dimensional encoding, which is given as input to the decoder. The encoders and decoder contain a stack of 2D convolution layers with residual blocks, batch normalization layers and ReLU activation. In addition, we also include the skip connections between the face encoder and the face decoder for better gradient flow and to preserve the crucial facial features. The decoder finally generates a quantized embedding in the latent space $V_f$.

**Training details:** The generator network is trained to generate accurate lip shapes conditioned on a given speech segment. To prepare the input to the speech encoder, we take a short window of $T_x = 20$ melspectrogram time steps (200 ms of speech), denoted by $S_x$. We then take the middle frame, $F_{gt}$, of this speech window and consider it as the ground truth frame. We pass $F_{gt}$ through the encoder of $V_f$ to get the ground truth embedding $E_{gt}$, and mask the lower half of $F_{gt}$, pass it through $V_p$, and generate the pose embedding $E_{gtp}$. A reference frame, $F_r$, from a different time-step is selected and given to $V_f$, which generates the reference quantized embedding $E_r$. We channel-wise concatenate $E_{gtp}$ and $E_r$, which acts as input to the face encoder. The speech encoder ingests the input speech melspectrogram $S_x$. We then concatenate the output of both the encoders. The decoder uses this concatenated embedding to predict the output embedding $E'_{gt}$. The network is trained using the $L1$ loss between $E'_{gt}$ and $E_{gt}$. We also compute the sync loss using our pre-trained lip-sync expert discriminator, which takes the audio-video pair ($S_x$, $T_f$) and detects if they are in-sync or out-of-sync.

**Additional Training details:** The model is trained for about four days on 4 NVIDIA RTX 2080 Ti GPUs. We use Adam optimizer with a learning rate of 1e-4. We used Python 3.8, torch 1.7.0, and CUDA 10.2 for training the model. The total number of parameters in the model is 98.9M. At inference time, the model is able to process 22 frames per second. Please note that this time does not account for the post-processing time, which may be required in the case of movies etc. Table 3.3 shows the computation cost, training, and inference time of Wav2lip [39] and our method.

| **Method** | **# params** (in Million) | **Inference Time** (in fps) | **Training Time** (in days) |
|---|---|---|---|
| Wav2Lip | 36.3 | 60 | 2 |
| **Ours** | **98.9** | **22** | **4** |

**Table 3.3** Comparison of Computation cost, training, and inference time.

**Inference details:** During inference, we consider a sliding window of 200ms (20 mel time-steps) across the full speech segment. Each speech window is inferred separately through our lip-sync generator. Assuming we have a video during inference, we take the corresponding video frame and pass it to $V_f$ which generates the reference embedding. We also input a masked version of the frame to $V_p$ which encodes the pose. Both the reference and the pose embeddings are channel-wise concatenated and given

to the lip-sync generator along with the melspectogram input. The decoder finally outputs a lip-synced quantized embedding.

### 3.3.2 Stage-2 (Optional): Post-Processing stage 1 output

This is an optional stage to further improve the visual quality of the generated output from the stage 1. We use GPEN [63] as the post processing network and found that we get slightly improved and sharper results. We train GPEN [63] following the original training procedure and losses on our newly collected 4KTF dataset. During inference, we feed a modified face crops to the network: we replace only the lip region of the original face crops of the video with the output generated from stage 1 using the lip landmarks obtained from Mediapipe [30]. The synthesized outputs are then pasted back into the original video. This stage is totally optional and can be replaced with any post-processing network.

### 3.3.3 Watermarking the Final Outputs

Talking-face generation models [39, 46, 59, 68, 23] enable a plethora of positive applications. However, there are potential negative impacts due to the possibilities for harmful "deepfakes". We add an invisible watermark to our dataset using the invisible watermarking technique [31] [4]. There is no change in the perceptual visual quality of the image. A randomly generated fixed string is embedded (watermarked) into the image in the frequency space using the DWT + DCT + SVD transformations. We can decode the image to get back the fixed string using the inverse of each transform. We first watermark the whole dataset and then train the network. It ensures the watermark is inherently learned by the model and outputs it in each of the generated face crops, which are finally pasted back into the full frame. While testing, we first detect each face region present in a video. We then try to decode the watermark in the detected facial areas in each frame. If $50\%$ of the total frames contain the watermark, we assume it to be a match. Further, we will release our codes and models only for research purposes after having specific written agreements with the users. This will help us keep records of the users of our codebase and avoid misuse.

## 3.4 Experiments

In this section, we evaluate various aspects of the generated outputs from our method on different datasets. We also include several visual results from our technique and compare them with the current state-of-the-art methods.

[4] https://github.com/ShieldMnt/invisible-watermark

### 3.4.1 Quantitative Evaluations

**Metrics:** To evaluate the quality of lip-synchronization, we use "Lip Sync Error - Confidence" (LSE-C) and "Lip Sync Error - Distance" (LSE-D) metrics introduced in Wav2Lip [39]. The publicly available pre-trained model SyncNet [12] is used to calculate the lip-sync errors. More details about the two metrics can be found in Wav2Lip [39]. In addition to these metrics, we also use the popular Fréchet Inception Distance (FID) to evaluate the perceptual quality of the generations at a frame level. Similarly, we use the Fréchet Video Distance (FVD) [55] proposed to measure the perceptual quality at the video level. FVD measures both temporal coherence and sharpness at the frame level. These metrics are calculated using only the face crops, ensuring the high-resolution background does not play any role in the calculations.

**Baselines:** We compare our work with multiple baselines. We modify the publicly available codebases of "You said that?" [10], "LipGAN" [40] and "Wav2Lip" [39] and train them using the same settings and datasets as our model ($768 \times 768$ pixel resolution). We make suitable changes in the architectures to handle the higher resolution input. As another baseline, we use the publicly available Wav2Lip model at original resolution ($96 \times 96$) and use the pre-trained state-of-the-art video super-resolution model "TecoGAN" [9] to obtain the super-resolved videos at the target resolution. We evaluate all the models on 5000 selected videos from the AVSpeech test-set and the test-set from the proposed 4K dataset. Please note that the AVSpeech test set is evaluated at 1080p resolution.

**Results:** As seen in Table 3.4, we outperform the competing methods by a significant margin. Our method produces lip-synced videos at very high-resolutions (indicated by LSE metrics). The generated outputs are sharper and highly temporally coherent compared to the previous works (indicated by FID and FVD metrics). Our method surpasses the existing baselines in generating high-quality frames with very less artefacts (also validated in Figure 3.5, and demo video[5]).

**Table 3.4** Quantitative scores of different methods on AVSpeech [16] datasets.

| **Method** | **LSE-C** ↑ | **LSE-D** ↓ | **FID** ↓ | **FVD** ↓ | **LSQ** ↑ | **Shrp.** ↑ | **OE** ↑ |
|---|---|---|---|---|---|---|---|
| You-said-that-4K [10] | 0.98 | 10.01 | 9.12 | 9.81 | 2.50 | 1.32 | 1.98 |
| LipGAN-4K [40] | 1.09 | 9.52 | 7.63 | 8.52 | 2.63 | 1.71 | 2.31 |
| Wav2Lip-4K [39] | 2.66 | 9.13 | 8.01 | 8.41 | 3.17 | 1.65 | 2.18 |
| Wav2Lip-orig [39] + TecoGAN [9] | 4.17 | 6.33 | 7.47 | 7.16 | 3.26 | 1.94 | 2.27 |
| **Ours** | **7.26** | **6.21** | **5.18** | **6.41** | **3.72** | **4.51** | **4.32** |

[5] https://youtu.be/l3Lbxz3J_WU

**Table 3.5** Quantitative scores of different methods on our new 4KTF datasets. Our model outperforms all baselines by a large margin. Using our approach, we can obtain high-quality outputs (indicated by FID and FVD) and accurate lip synchronisation (indicated by LSE-C and LSE-D). Note that FVD is scaled by a factor of 100 for better readability. We also report the human evaluation scores based on: (i) Lip-sync Quality (LSQ), (ii) Sharpness (Shrp.) and (iii) Overall Experience (OE).

| **Method** | **LSE-C** ↑ | **LSE-D** ↓ | **FID** ↓ | **FVD** ↓ | **LSQ** ↑ | **Shrp.** ↑ | **OE** ↑ |
|---|---|---|---|---|---|---|---|
| You-said-that-4K [10] | 1.07 | 10.47 | 18.34 | 9.83 | 1.32 | 1.44 | 1.41 |
| LipGAN-4K [40] | 1.43 | 8.18 | 14.21 | 9.16 | 1.47 | 1.42 | 1.31 |
| Wav2Lip-4K [39] | 3.12 | 8.74 | 7.54 | 7.91 | 3.52 | 1.37 | 2.63 |
| Wav2Lip-orig [39] + TecoGAN [9] | 4.03 | 7.24 | 7.18 | 8.86 | 3.43 | 1.72 | 2.14 |
| **Ours** | **7.10** | **6.32** | **3.11** | **6.66** | **3.61** | **4.43** | **4.62** |

### 3.4.2 Human Evaluations

Since the quality of lip-sync is highly subjective, we perform human evaluations on the generated videos. We show the outputs from different algorithms to 50 users and ask them to rate the videos on a scale of $1-5$, with 1 being the lowest rating and 5 being the highest. The users are asked to rate on the following three attributes: (i) Lip sync Quality, (ii) Sharpness and other details of the face and (iii) Overall Experience of the video. We report the mean opinion scores in Table 3.4 and 3.5. In-line with the quantitative evaluation, our method achieves the highest scores in all these attributes, indicating the robustness of our approach.

Figure 3.5 depicts the samples generated from different models. We can observe that our model generates a highly detailed lip region compared to the current methods. It effectively reconstructs fine-grained facial features like teeth, lip color, lip, and jaw texture and has minimal to no artifacts. We find the visual results to corroborate the findings in our quantitative and human evaluations.

**Table 3.6** Our method works well on silent regions of the video.

| **Method** | **LSE-C** ↑ | **LSE-D** ↓ | **FID** ↓ | **FVD** ↓ |
|---|---|---|---|---|
| Wav2Lip-orig [39] + TecoGAN [9] | 1.08 | 12.73 | 7.124 | 10.88 |
| **Ours** | **4.18** | **8.21** | **3.101** | **9.03** |

### 3.4.3 Performance on Silent Regions

While Wav2Lip [39] generates accurate lip-sync in most cases, it struggles with long silent regions. The original lip movements in the video interferes with the generated ones and result in significant

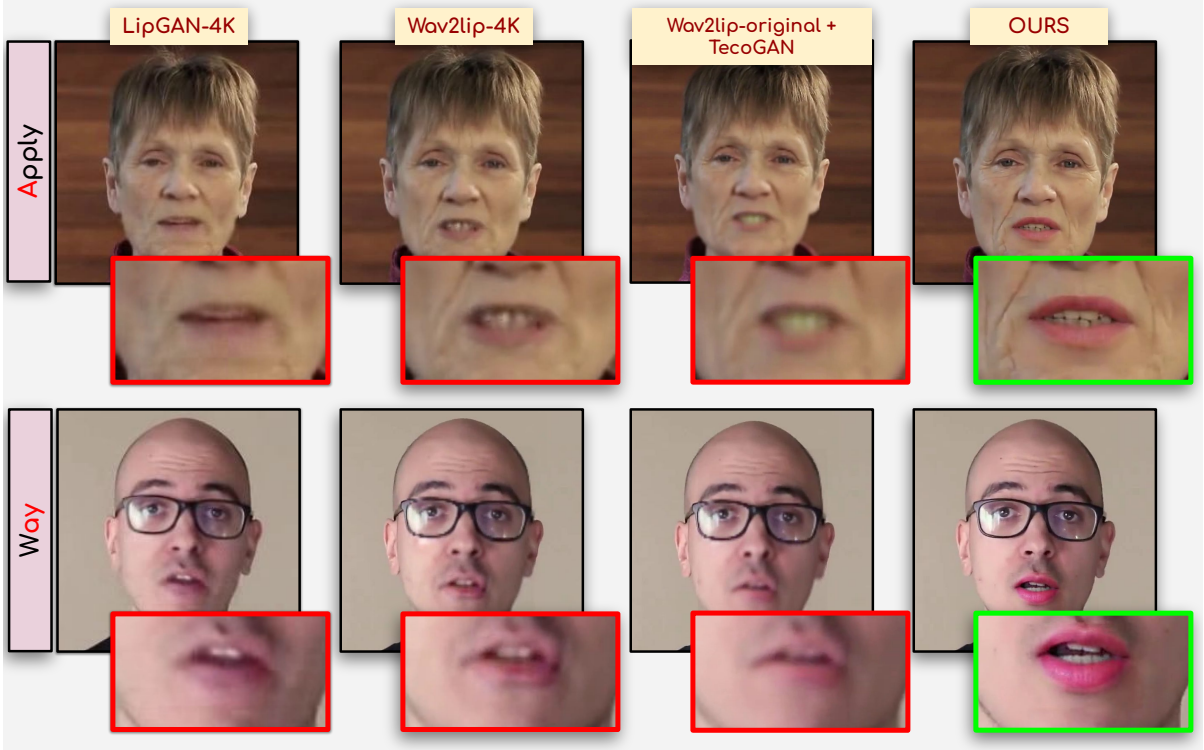


**Figure 3.5** Sample results from different algorithms. Clearly, our model generates far better, sharper and higher-quality outputs. Our model captures intricate details like teeth, wrinkles of skin and lip color, which the previous models fail to generate.

quivering of lips. We provide silent audio as input to all the videos in the test set and compare our results to that of Wav2Lip in Table 3.6. A visual demonstration of samples is also provided in Figure 3.6. As seen from both the table and the figure, our model handles silences far better than Wav2Lip. We hypothesize the reason for this to be learning lip-sync in the quantized space, which is richer than the image space that Wav2Lip was trained on.

## 3.5 Ablation Studies

We perform several ablations to verify the effect of our different components. The scores are reported on the test set of 4KTF dataset.

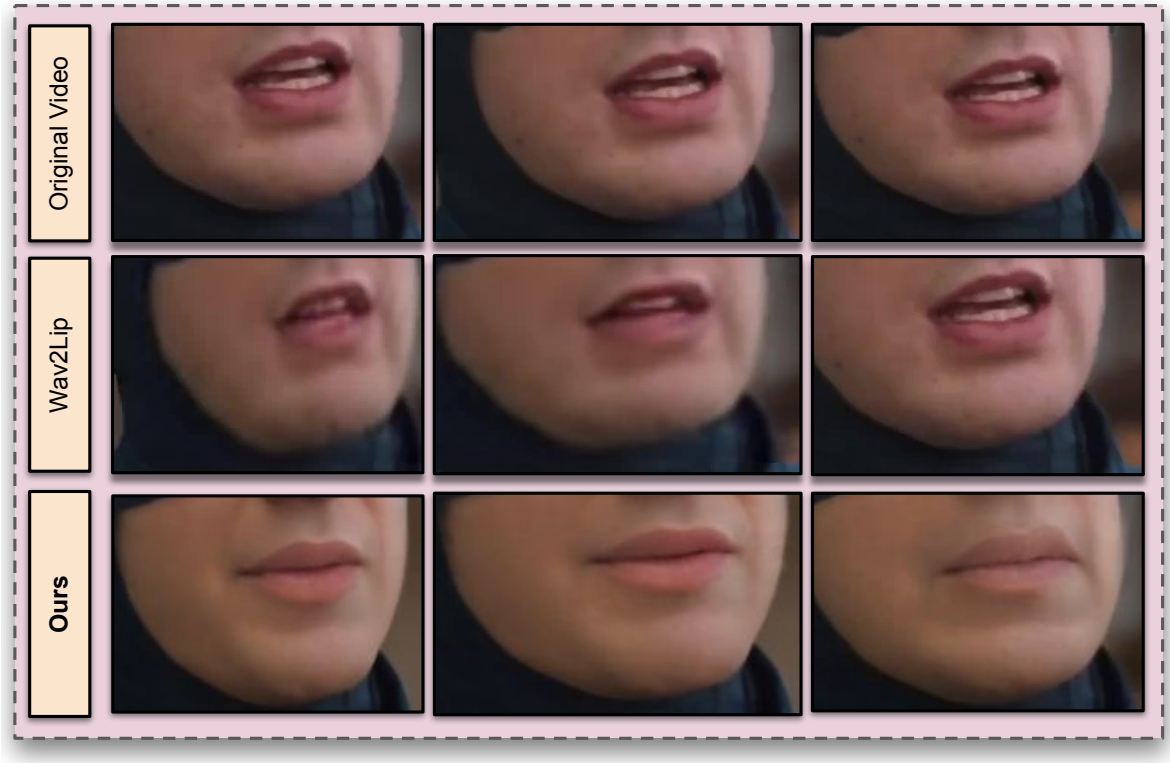


**Figure 3.6** Performance evaluation on silent speech segments. While the output from Wav2Lip follows the original lip movements, our model can generate closed lip shapes in sync with the silent speech.

### 3.5.1 Importance of Post Processing Network

To assess the importance of the Stage 2 network, we compare the results of stage 1 and 2 of our pipeline. While the results have decent lip-sync, the stage 2 results are slightly sharper, as also can be seen in Table 3.7.

**Table 3.7** Comparison of stage 1 and 2 results.

| **Method** | **LSE-C** ↑ | **LSE-D** ↓ | **FID** ↓ | **FVD** ↓ |
|---|---|---|---|---|
| Ours w/o Stage 2 | 7.01 | 6.31 | 4.43 | 7.48 |
| **Ours** | **7.10** | **6.32** | **3.11** | **6.66** |

### 3.5.2 Importance of the lip-sync expert

We train a lip-sync generator without using the sync loss and report results in Table 3.8. We also vary the context window size $T$ - we test with $T = 5$ and $T = 25$. We find that the lip-sync expert trained on

longer audio-visual sequences perform better and is selected for the final version. We also calculate the accuracy of the lip-sync expert by creating random audio-visual pairs that are in-sync and out-of-sync with 50% probability. Table 3.8 indicates that the best accuracy is achieved for the model trained with sync loss using a context window of 25 frames.

**Table 3.8** We evaluate the importance of lip-sync expert and also show the effect of using different context windows.

| **Method** | **LSE-C** ↑ | **LSE-D** ↓ | **Acc.** ↑ |
|---|---|---|---|
| Ours w/o Sync Loss | 1.13 | 11.01 | - |
| Ours with Sync Loss, T=5 | 3.12 | 10.38 | 65.1% |
| **Ours with Sync Loss, T=25** | **7.10** | **6.32** | **91.2%** |

## 3.6 Applications

We believe our model is a perfect fit for several applications at a time when the amount of multimedia content around the globe is growing exponentially. Few of the potential applications enabled by our model are as follows.

### 3.6.1 Movie and television industries

Dubbing Movie and TV shows make the content accessible to a larger audience. Studies [51] have shown that people prefer dubbed content over the subtitled counterpart. Movie and TV Show dubbing has a market size of 3 Billion US Dollars! [6] which will only grow in the future. We believe our work is an ideal fit for such a scenario. Since our model does not depend on speaker-specific data, we can handle movie scenes without re-training. Our model can lip-sync such dubbed movies with ease and improve the viewing experience. Similarly, other forms of dubbed content like interviews, documentaries, and lectures can also be precisely lip-synced

### 3.6.2 Marketing Videos, Conversational AI and News Reading

Video content is an irreplaceable part of a business's marketing strategy. Creating such marketing videos costs anywhere between 1000$ to 50000$. [7]. Not only is the cost higher, but the creation of such videos also take days to produce. One can accurately lip-sync a person talking in front of a green screen with a marketing script to create the marketing video in just a few minutes. As a result, such videos can be created more cost-effectively and at scale.

[6] https://www.marketwatch.com/press-release/film-dubbing-market-size-top-manufacturers-growth-fac
[7] https://hingemarketing.com/blog/story/what-is-the-cost-of-video-production-for-the-web

Conversational AI is another area that is getting a lot of attention currently. It is an integral part of the business. Chatbots and virtual assistants help increase user engagement and satisfaction. To take this one step further, we can create digital talking face avatars which can interact with the user to make the experience more immersive.

### 3.6.3 Online meetings

Because of the COVID-19 pandemic, video conferencing platforms have gained tremendous popularity. The number of virtual meetings has skyrocketed, and many people are dealing with video call exhaustion "Zoom fatigue".[8]. One of the reasons for this fatigue is constantly looking into the camera. We can first record a short clip of a person looking at the camera. The same clip can then be lip-synced with the spoken content in a loop using our method and generate an infinitely long video feed. Such video feeds can be redirected to platforms like Zoom using virtual cameras. We can also generate a virtual stream in cases of loss of visual signal.

### 3.6.4 Animations

Even though our model is never trained on CGI faces, it still performs well on animated characters. This allows our model to be used in gaming and animated movies

### 3.6.5 Lipreading Tutors

Lip reading is a technique for understanding speech contents by visually interpreting the lips, face, and tongue movements when typical sound is unavailable. Over $5\%$ of the world's population have hearing loss or deafness [9] and this number is only expected to increase in the future. Such people regularly use this technique to communicate better. However, Lip reading is a challenging skill to acquire. Organizations like `lipreading.org` use recorded videos of an instructor mouthing different words to teach lip-reading to users. However, such content is only recorded on a small scale and covers a small part of the vocabulary. Moreover, the languages and accents covered are minimal. Using our method, an instructor can generate realistic lip/mouth movements on any avatar/person. We can first use different text-to-speech systems to create speech segments corresponding to words and phrases. We can then use our algorithm to lip-sync talking face videos with the generated speech. The final outputs of our algorithm contain accurate lip movements corresponding to the words. It could potentially replace the need to record videos of tutors and produce the same synthetically, increasing lip-reading tutor systems' reach. We can seamlessly cover multiple languages and styles with little to no manual effort.

[8] https://news.stanford.edu/2021/02/23/four-causes-zoom-fatigue-solutions/
[9] https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss

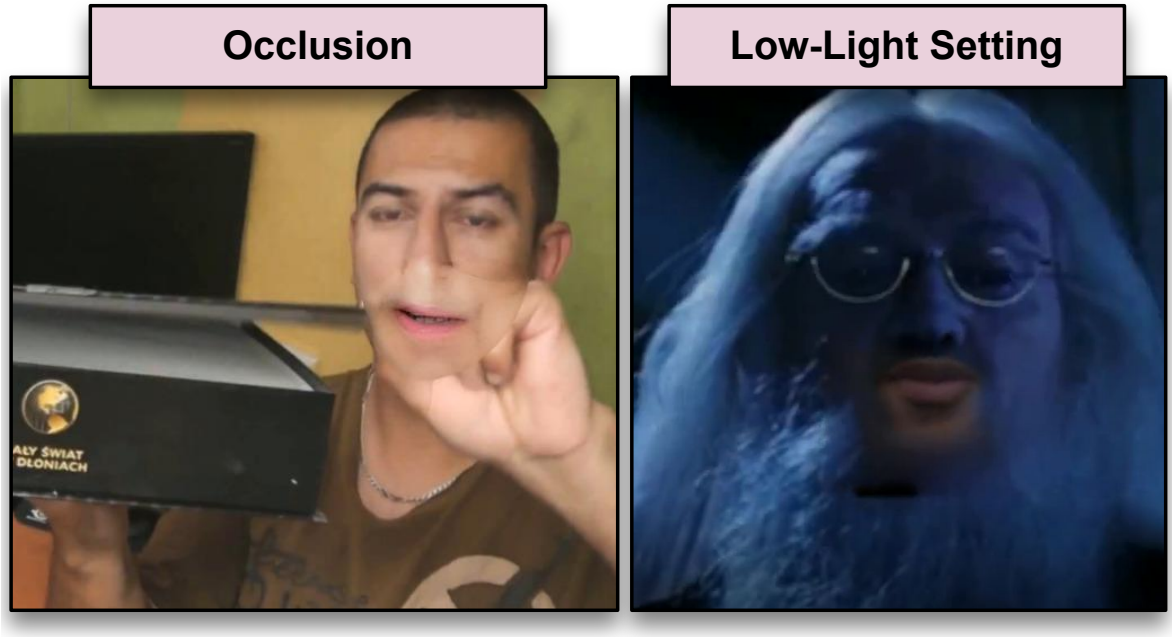


**Figure 3.7** Examples of failure cases of our model: (a) Fails to handle occlusions in the lip region (left) and (b) Fails in low light settings (right).

### 3.6.6 Making GIFs and Video Memes

GIPHY [10] is a platform to search, share and discover GIFs on the internet, which has over 700 Million daily users, and over 10 Billion gifs are shared everyday! [11]This shows the popularity of this media, which is a perfect application for our algorithm. Our algorithm works for any face and language, making the creation of such content easy.

## 3.7 Limitations

Although our model generates high-quality results in most scenarios, there are a few cases where our model fails to generate the desired outputs. Figure 3.7 shows two such cases. Our model expects the lips to be fully visible and thus fails to handle occlusions. Occlusion by hands or any other object causes the model to generate fake lips in an approximate location (see 3.7 - left part). Another case where our model generates sub-optimal results is low lighting conditions. Low light often changes the skin's overall characteristics, which are not captured in the output from our model. However, our model works for a wide range of inputs and we anticipate that the core idea of utilizing the compact VQ space would open up new avenues and directions for future research.

[10] https://giphy.com/
[11] https://expandedramblings.com/index.php/giphy-facts-statistics/

## 3.8 Conclusion

This work presents the first approach in generating ultra-high resolution talking-face videos. With our approach, it is now possible to synthesize talking-face videos with accurate lip shapes at very high-resolutions (4K). Our work revolves around a two-stage framework where we first learn to lip-sync in a compact vectorized space and then render the high-resolution face outputs. We generate state-of-the-art, realistic, high-quality results at such high resolutions for the first time and mark significant improvements over the competitive methods. To avoid the potential misuse of our model, we also learn an invisible watermarking scheme. We believe our work will positively impact several industries, open up new applications and make movie-making much easier!

*Chapter 4*

# Conclusions

In this thesis, we explore the idea of producing and translating video content at scale. With machine learning being extensively used in video editing/production, we modified an existing open source video editor - OpenShot and integrated various SOTA machine learning algorithms in it, all the video editing features and ML techniques in a single video editor! With the intuitive UI of the video editor now, it is possible to use ML algorithms with a button click. This makes it possible for a person who is not an ML expert in using ML algorithms and doing video editing. We also conducted a user study to show the effectiveness of the video editor both in the quality of the outputs and the reduction in the time taken to edit/produce video. The fact that anyone can use this video editor and it takes less time to edit a video using our video editor than any other editor makes it possible to produce video content at scale. Also, we believe our editor opens up a new paradigm in video editing that allows interactive use of the latest published research and improves the overall user experience.

We further explore current lipsync works [40, 39] and their shortcomings, the current models perform lipsync at 96x96 resolution, which is quite low. These models were trained on datasets like AVspeech [16], LRS [11, 3] which are all low resolution datasets. To overcome this issue, we developed a model that can perform lipsync for up to 4k video resolutions, but no suitable dataset existed for training such a model. Therefore, we curated a new 4k talking face dataset. We trained our proposed model on this dataset and demonstrated the qualitative and quantitative results in Chapter 2. To avoid the potential misuse of our model, we also learn an invisible watermarking scheme. We believe that our work will open up new applications that could be done on modern-day high-resolution videos and make dubbing movies, creating marketing videos, etc., easier! Additionally, we believe our 4KTF dataset will assist research in video synthesis at higher resolution.

# Related Publications

- **Anchit Gupta**, Faizan Farooq Khan, Rudrabha Mukhopadhyay, Vinay P. Namboodiri, and C. V. Jawahar. 2021. Intelligent video editing: incorporating modern talking face generation algorithms in a video editor. Proceedings of the Twelfth Indian Conference on Computer Vision, Graphics and Image Processing. Association for Computing Machinery, New York, NY, USA, Article 25, 1–9. DOI: https://doi.org/10.1145/3490035.3490284

- **Anchit Gupta**, Rudrabha Mukhopadhyay, Sindhu B Hegde, Faizan Farooq Khan, Vinay P Namboodiri, C.V. Jawahar. Towards Generating Ultra-High Resolution Talking-Face Videos with Lip synchronization. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023.

- Darshan Singh S, **Anchit Gupta**, C.V. Jawahar, Makarand Tapaswi. Unsupervised Audio-Visual Lecture Segmentation. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023. *(Not a part of thesis)*

- Madhav Agarwal, **Anchit Gupta**, Rudrabha Mukhopadhyay, Vinay P. Namboodiri, C V Jawahar. Compressing Video Calls using Synthetic Talking Heads. British Machine Vision Conference (BMVC), 2022. *(Not a part of thesis)*

# Bibliography


[1] Amazon transcribe. https://aws.amazon.com/transcribe/?nc=sn&loc=1, 2021. 13

[2] Speech-to-text: Automatic speech recognition google cloud. https://cloud.google.com/speech-to-text, 2021. 1, 11

[3] T. Afouras, J. S. Chung, and A. Zisserman. Lrs3-ted: a large-scale dataset for visual speech recognition. *arXiv preprint arXiv:1809.00496*, 2018. 37

[4] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. Engel, L. Fan, C. Fougner, T. Han, A. Hannun, B. Jun, P. LeGresley, L. Lin, S. Narang, A. Ng, S. Ozair, R. Prenger, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, Y. Wang, Z. Wang, C. Wang, B. Xiao, D. Yogatama, J. Zhan, and Z. Zhu. Deep speech 2: End-to-end speech recognition in english and mandarin, 2015. 1, 11

[5] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis, 2019. 14

[6] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate, 2016. 12

[7] G. Balakrishnan, A. Zhao, A. V. Dalca, F. Durand, and J. Guttag. Synthesizing images of humans in unseen poses. 2018. 9

[8] A. Bansal, S. Ma, D. Ramanan, and Y. Sheikh. Recycle-gan: Unsupervised video retargeting, 2018. 9

[9] M. Chu, Y. Xie, J. Mayer, L. Leal-Taixé, and N. Thuerey. Learning temporal coherence via self-supervision for gan-based video generation. *ACM Trans. Graph.*, 39, July 2020. 22, 29, 30

[10] J. S. Chung, A. Jamaludin, and A. Zisserman. You said that? In *British Machine Vision Conference*, 2017. 8, 21, 29, 30

[11] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Lip reading sentences in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3444–3453. IEEE, 2017. 21, 23, 37

[12] J. S. Chung and A. Zisserman. Out of time: automated lip sync in the wild. In *Workshop on Multi-view Lip-reading, ACCV*, 2016. 23, 29

[13] R. Collobert, C. Puhrsch, and G. Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system, 2016. 1, 11

[14] deepfakes. Faceswap. https://github.com/deepfakes/faceswap, 2021. 7

[15] dunnousername. Front-end tool for first-order-motion. https://github.com/dunnousername/yanderifier, 2021. 7

[16] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. 37(4), July 2018. xii, 22, 23, 29, 37

[17] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, June 2021. 24, 25

[18] O. Fried, A. Tewari, M. Zollhöfer, A. Finkelstein, E. Shechtman, D. B. Goldman, K. Genova, Z. Jin, C. Theobalt, and M. Agrawala. Text-based editing of talking-head video. *ACM Trans. Graph.*, 38(4):68:1–68:14, July 2019. 20

[19] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generativeadversarialnets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press. 9

[20] M. Haris, G. Shakhnarovich, and N. Ukita. Recurrent back-projection network for video super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 22

[21] K. Ito and L. Johnson. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017. 13

[22] JaidedAI. Jaidedai/easyocr: Ready-to-use ocr with 80 supported languages and all popular writing scripts including latin, chinese, arabic, devanagari, cyrillic and etc. https://github.com/JaidedAI/EasyOCR, 2021. 14

[23] X. Ji, H. Zhou, K. Wang, W. Wu, C. C. Loy, X. Cao, and F. Xu. Audio-driven emotional video portraits. *arXiv preprint arXiv:2104.07452*, 2021. 28

[24] H. Kim, M. Elgharib, M. Zollhöfer, H.-P. Seidel, T. Beeler, C. Richardt, and C. Theobalt. Neural style-preserving visual dubbing. 38(6), 2019. 7

[25] J. Kim, S. Kim, J. Kong, and S. Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8067–8077. Curran Associates, Inc., 2020. 2, 6, 13

[26] G. H. Kranz. Transcribe. https://aws.amazon.com/transcribe/, 1986. 1, 11

[27] R. Kumar, J. Sotelo, K. Kumar, A. D. Brébisson, and Y. Bengio. Obamanet: Photo-realistic lip-sync from text. *ArXiv*, abs/1801.01442, 2018. 20, 21

[28] A. Lahiri, V. Kwatra, C. Früh, J. Lewis, and C. Bregler. Lipsync3d: Data-efficient learning of personalized 3d talking faces from video using pose and lighting normalization. *CoRR*, abs/2106.04185, 2021. 20

[29] M. A. Lee. Python tesseract. https://github.com/madmaze/pytesseract, 2021. 14

[30] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann. Mediapipe: A framework for building perception pipelines. *ArXiv*, abs/1906.08172, 2019. 28

[31] K. Navas, M. C. Ajay, M. Lekshmi, T. S. Archana, and M. Sasikumar. Dwt-dct-svd based watermarking. In *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*, pages 271–274. IEEE, 2008. 28

[32] L. OpenShot Studios. https://www.openshot.org/, 2021. 6, 16

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 8

[34] I. Perov, D. Gao, N. Chervoniy, K. Liu, S. Marangonda, C. Umé, M. Dpfks, C. S. Facenheim, L. RP, J. Jiang, S. Zhang, P. Wu, B. Zhou, and W. Zhang. Deepfacelab: Integrated, flexible and extensible face-swapping framework, 2021. 7

[35] J. Philip, V. P. Namboodiri, and C. Jawahar. CVIT-MT systems for WAT-2018. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation: 5th Workshop on Asian Translation: 5th Workshop on Asian Translation*, Hong Kong, 1–3 Dec. 2018. Association for Computational Linguistics. 1, 12, 13

[36] J. Philip, V. P. Namboodiri, and C. V. Jawahar. A baseline neural machine translation system for indian languages, 2019. 1, 6, 12

[37] J. Philip, S. Siripragada, U. Kumar, V. Namboodiri, and C. V. Jawahar. CVIT's submissions to WAT-2019. In *Proceedings of the 6th Workshop on Asian Translation*, pages 131–136, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. 1, 12

[38] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller. Deep voice 3: 2000-speaker neural text-to-speech. In *International Conference on Learning Representations*, 2018. 2, 13

[39] K. R. Prajwal, R. Mukhopadhyay, V. Namboodiri, and C. V. Jawahar. A lip sync expert is all you need for speech to lip generation in the wild. *CoRR*, abs/2008.10010, 2020. x, 3, 4, 8, 9, 10, 18, 19, 21, 22, 25, 26, 27, 28, 29, 30, 37

[40] K. R. Prajwal, R. Mukhopadhyay, J. Philip, A. Jha, V. Namboodiri, and C. V. Jawahar. Towards automatic face-to-face translation. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, New York, NY, USA, 2019. ACM. 3, 6, 8, 21, 25, 29, 30, 37

[41] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. *ArXiv*, abs/2006.04558, 2021. 2, 13

[42] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu. Fastspeech: Fast, robust and controllable text to speech. In *NeurIPS*, 2019. 2, 13

[43] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783, 2018. 2, 13

[44] Z. Shen, R. Zhang, M. Dell, B. C. G. Lee, J. Carlson, and W. Li. Layoutparser: A unified toolkit for deep learning based document image analysis. *arXiv preprint arXiv:2103.15348*, 2021. 14

[45] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe. Animating arbitrary objects via deep motion transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 9

[46] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe. First order motion model for image animation. In *Conference on Neural Information Processing Systems (NeurIPS)*, December 2019. 3, 7, 9, 21, 28

[47] L. Song, W. Wu, C. Qian, R. He, and C. C. Loy. Everybody's talkin': Let me talk as you want. *arXiv preprint*, arXiv:, 2020. 20

[48] Y. Song, J. Zhu, D. Li, X. Wang, and H. Qi. Talking face generation by conditional recurrent adversarial network, 2019. 7

[49] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. 1, 12

[50] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman. Synthesizing obama: Learning lip sync from audio. *ACM Trans. Graph.*, 36(4), July 2017. 7, 20, 21

[51] S. Sánchez-Mompeán. Netflix likes it dubbed: Taking on the challenge of dubbing into english. *Language & Communication*, 80:180–190, 2021. 33

[52] S. Team. Silero models: pre-trained enterprise-grade stt / tts models and benchmarks. https://github.com/snakers4/silero-models, 2021. 1, 6, 11, 12

[53] Tesseract-Ocr. tesseract-ocr/tesseract: Tesseract open source ocr engine (main repository). https://github.com/tesseract-ocr/tesseract, 2021. 14

[54] J. Thies, M. Elgharib, A. Tewari, C. Theobalt, and M. Nießner. Neural voice puppetry: Audio-driven facial reenactment. *arXiv preprint arXiv:1912.05566*, 2019. 20, 21

[55] T. Unterthiner, S. van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. 29

[56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017. 12

[57] vocali.se. https://vocali.se/en, 2021. 10

[58] K. Wang, Q. Wu, L. Song, Z. Yang, W. Wu, C. Qian, R. He, Y. Qiao, and C. C. Loy. Mead: A large-scale audio-visual dataset for emotional talking-face generation. In *ECCV*, August 2020. 23

[59] T.-C. Wang, A. Mallya, and M.-Y. Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 21, 28

[60] Z. Wojna, A. Gorban, D.-S. Lee, K. Murphy, Q. Yu, Y. Li, and J. Ibarz. Attention-based extraction of structured information from street view imagery, 2017. 14

[61] H. Wu, J. Jia, H. Wang, Y. Dou, C. Duan, and Q. Deng. *Imitating Arbitrary Talking Style for Realistic Audio-Driven Talking Face Synthesis*, page 1478–1486. Association for Computing Machinery, New York, NY, USA, 2021. 21

[62] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. 13

[63] T. Yang, P. Ren, X. Xie, and L. Zhang. Gan prior embedded network for blind face restoration in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 672–681, 2021. 28

[64] X. Yao, O. Fried, K. Fatahalian, and M. Agrawala. Iterative text-based editing of talking-heads using neural retargeting. *ACM Trans. Graph.*, 40(3), Aug. 2021. 7

[65] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Li. $S^3$fd: Single shot scale-invariant face detector. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 192–201, 2017. 23

[66] Z. Zhang, L. Li, Y. Ding, and C. Fan. Flow-guided one-shot talking face generation with a high-resolution audio-visual dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3661–3670, 2021. 21, 23

[67] H. Zhou, Y. Liu, Z. Liu, P. Luo, and X. Wang. Talking face generation by adversarially disentangled audio-visual representation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019. 7

[68] H. Zhou, Y. Sun, W. Wu, C. C. Loy, X. Wang, and Z. Liu. Pose-controllable talking face generation by implicitly modularized audio-visual representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 21, 28

[69] Y. Zhou, X. Han, E. Shechtman, J. Echevarria, E. Kalogerakis, and D. Li. Makeittalk: Speaker-aware talking-head animation. *ACM Trans. Graph.*, 39(6), Nov. 2020. 3, 8, 21