Improved Representation Spaces for Videos

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

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

by

Bipasha Sen
2021701003
`bipasha.sen@research.iiit.ac.in`

International Institute of Information Technology
Hyderabad - 500 032, INDIA
August 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled **"Improved Representation Spaces for Videos"** by Bipasha Sen, 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 of four

Acknowledgments

My experience pursuing a Master's degree after spending time in the industry has been incredibly fulfilling. One aspect that has stood out to me is the unique bond we develop with our supervisors. Where the relationship between different hierarchies (manager $\rightarrow$ lead $\rightarrow$ developers) is predefined in the industry, our relationship with our supervisors is what we make of it. As my advisor, **Prof. C V Jawahar**, says it's about "a shared goal". But what does this mean exactly? For me, discovering what a shared goal looks like has been one of the most interesting aspects of my Master's program. Unlike in the industry where managers drive the vision, leads drive task items, and developers execute tasks, the role of a supervisor is less defined. As students, it's up to us to figure out how we can mutually drive our shared goal. It takes many days and hours of abstract discussions, multiple projects, and many months to understand what your version of a shared goal is – it's a unique and fun experience. The freedom to define the relationship with our supervisors and pursue a shared goal has made my Master's program truly different and rewarding.

Prof. Jawahar and **Prof. Vinay Namboodiri**, my advisors, have been instrumental in shaping me as a researcher. I have learned to consider the broader context of my work and be more critical of my work. Submitting research work and getting critical reviews is part and parcel of the process. Everyone loves good reviews but analyzing and really looking through a scathing review, introspecting where we went wrong, and improving our work so much that it becomes a sure-shot accept is something I learned through them. **Prof. Vinay** has provided me with a valuable reviewer's perspective, helping me understand how to respond effectively to criticism and when the best time to do so is. He has emphasized the importance of thoroughness in both writing and experimentation. When we publish a paper, we are essentially trying to convince someone else that our ideas are valid, which requires clear and concise writing and strong supporting evidence. **Prof. Jawahar** has played a pivotal role in helping me choose the right direction for my research. He has challenged me to think critically about why certain ideas are important or not worth pursuing, giving me greater clarity in my thinking. It is an understatement to say that I have been blessed with two wonderful supervisors. They brought the best out of me and made me a better researcher and person.

Rudrabha. Thank you for being the funniest person you are and cracking me up with your mind-blowing stories! I always found it impressive how you simultaneously managed to be a part of ten different projects! **Zeeshan. Rupak. Anchit. Siddhant.** Thank you for just existing and showing your faces in the lab - I always loved seeing you guys around. You guys made it feel like home! **Sai**

Niranjan, although you worked remotely, I had a great time discussing different ideas and working on different submissions with you.

I am genuinely 100% grateful that I took Mobile Robotics (MR) in the first semester and got to meet **Prof. Madhava Krishna** (course instructor and my mentor) - who is just such an amazing, humble, one-of-a-kind human being. I am amazed by your unique perspective on life and academia. Thank you for trusting me, letting me be the TA for MR the following semester, and making me such an incredible part of your group!

I want to thank **Vishal, Shankara, and Gaurav**, who are my co-authors and friends and are really fun people to be around. Discussing anything (including Harry Potter, future dreams, and aspirations) with them is always interesting!

Aditya. This journey has been much more efficient, fun, driving, and every single positive adjective I could think of - because of him. He has been my co-author, project partner, course partner, and my friend.

Most importantly, **my family. My dad, mom, and my brother**. I love you all so much. I wish I could spend more time with you; I am guiltily glad I joined CVIT during covid. I spend the first few months (the ramp-up period) with them. Seeing their faces every day I woke up made me so happy – kept my emotional coefficient high, and in turn, I could be more efficient at work! I miss you all so much, and I wish I could spend every single minute of my life with you three. Thank you for being so understanding while I worked non-stop when I visited you. You said to me “your presence is enough,” – and I could work without guilt at home every time I visited you. I cannot say this enough - I love you all so much.

Abstract

Videos form an integral part of human lives and act as one of the most natural forms of perception spanning both the spatial and the temporal dimensions: the spatial dimension emphasizes the content, whereas the temporal dimension emphasizes the change. Naturally, studying this modality is an important area of computer vision. Notably, one must efficiently capture this high-dimensional modality to perform different downstream tasks robustly. In this thesis, we study representation learning for videos to perform two key aspects of video-based tasks: classification and generation. In a classification task, a video is compressed to a latent space that captures the key discriminative properties of a video relevant to the task. On the other hand, generation involves starting with a latent space (often a known space, such as standard normal) and learning a valid mapping between the latent and the video manifold. This thesis explores complementary representation techniques to develop robust representation spaces useful for diverse downstream tasks. In this vein, this thesis starts by tackling video classification, where we concentrate on a specific task of "lipreading" (transliterating videos to text) or in technical terms - classifying videos of mouth movements. Through this work, we propose a compressed generative space that self-augments the dataset improving the discriminative capabilities of the classifier. Motivated by the findings of this work, we move on to finding an improved generative space in which we touch upon several key elements of video generation, including unconditional video generation, video inversion, and video superresolution.

In the classification task, we aim to study lipreading (or visually recognizing speech from the mouth movements of a speaker), a challenging and mentally taxing task for humans to perform. Unfortunately, multiple medical conditions force people to depend on this skill in their day-to-day lives for essential communication. Patients suffering from 'Amyotrophic Lateral Sclerosis' (ALS) often lose muscle control, consequently, their ability to generate speech and communicate via lip movements. Existing large datasets do not focus on medical patients or curate personalized vocabulary relevant to an individual. Collecting large-scale datasets of a patient needed to train modern data-hungry deep learning models is, however, extremely challenging. We propose a personalized network designed to lipread for an ALS patient using only one-shot examples. We depend on synthetically generated lip movements to augment the one-shot scenario. A Variational Encoder-based domain adaptation technique is used to bridge the real-synthetic domain gap. Our approach significantly improves and achieves high top-5 accuracy with 83.2% accuracy compared to 62.6% achieved by comparable methods for the patient. Apart from evaluating our approach on the ALS patient, we also extend it to people with hearing impairment, relying extensively on lip movements to communicate.

In the next part of the thesis, we focus on representation spaces for video-based generative tasks. Generating videos is a complex task that is accomplished by generating a set of temporally coherent images frame-by-frame. This approach confines the expressivity of videos to image-based operations on individual frames, necessitating network designs that can achieve temporally coherent trajectories in the underlying image space. We propose INR-V, a video representation network that learns a continuous space for video-based generative tasks. INR-V parameterizes videos using implicit neural representations (INRs), a multi-layered perceptron that predicts an RGB value for each input pixel location of the video. The INR is predicted using a meta-network which is a hypernetwork trained on neural representations of multiple video instances. Later, the meta-network can be sampled to generate diverse novel videos enabling many downstream video-based generative tasks. Interestingly, we find that conditional regularization and progressive weight initialization play a crucial role in obtaining INR-V. The representation space learned by INR-V is more expressive than an image space showcasing many interesting properties not possible with the existing works. For instance, INR-V can smoothly interpolate intermediate videos between known video instances (such as intermediate identities, expressions, and poses in face videos). It can also in-paint missing portions in videos to recover temporally coherent full videos. We evaluate the space learned by INR-V on diverse generative tasks such as video interpolation, novel video generation, video inversion, and video inpainting against the existing baselines. INR-V significantly outperforms the baselines on several of these demonstrated tasks, clearly showcasing the potential of the proposed representation space.

In summary, this thesis makes a significant contribution to the field of computer vision by exploring representation learning for videos. The proposed methods are thoroughly evaluated through extensive experimentation and analysis, which clearly demonstrate their advantages over existing works. These findings have the potential to advance a range of video-based applications, including personalized healthcare, entertainment, and communication. By developing robust representation spaces that improve video classification and generation, this work opens up new possibilities for more natural and effective ways of perceiving, understanding, and interacting with videos.

***Keywords* –** video classification, lipreading, implicit neural representations, hypernetworks, video generation, video inversion, video interpolation, video superresolution, video prediction

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Studying the visual modality (images, videos, etc.) is an interesting area of research as it most closely resembles how humans see the world. However, the primary challenge in tackling this modality stems from the high-dimensional nature of the datapoints. For example, a single low-resolution image contains a whopping 196-thousand values ($256 \times 256 \times 3$). At the same time, a short 16-seconds long video has 3-million values. It is evident that storing, processing, and generating such high-dimensional datapoints is exceptionally complex, thus highlighting the need for techniques that can transform these explicit high-dimensional datapoints into compact representations.

Ideally, a compact representation for a given datapoint X must retain the essential properties of X to accurately perform a given task. For example, given an in-the-wild image of a dog, one can divide the image into the foreground (relevant pixels) and background (irrelevant or does not affect the class, dog) for the task of animal <u>classification</u>. On the other hand, for a machine learning (ML) algorithm to <u>generate</u> a high-resolution image of a dog, a compressed representation (input to the ML algorithm) must encode both - the background, the foreground, and the relationship between the background and the foreground for the algorithm to de-compress it to the high-dimensional output space of a high-fidelity image. In this aspect, it is imminent to study what is a good representation space for the visual modality!

Background: The community has seen massive progress in developing techniques and networks that can capture good representations of the visual modality [73, 24, 42, 43, 41, 53, 5, 6, 89, 79, 80, 4]. Most of the works have used supervised classification to find good representations of the datapoints. For example, pioneering datasets like ImageNet [77] have been used to train architectures like Resnet [36] and VGG [83] in a supervised setting. In such cases, the last layer of the neural network (before the final discriminative layer) acted as the compressed representation. Such representations could either be finetuned for a different task or as input for a downstream network. Recent techniques like Variational AutoEncoders (VAEs) [46] and Generative Adversarial Networks (GANs) [31] do not require supervised datasets for learning data representations and, thus, can capture task-agnostic representations.

A variational autoencoder (VAE) [73, 102, 24] uses self-supervision to capture relevant information about a given datapoint. In this technique, X is first reduced to a compressed space using an encoder, $\mathcal{E}$; this compressed space is regularized using a KL-divergence loss to resemble a standard distribution (such as Gaussian distribution); finally, a decoder, $\mathcal{D}$, is used to transform the compressed space to the output image, X' such that the $\{||X - X'||_p = 0\}$, where p is the norm. On the other hand, GANs aim to start with a standard distribution and learn a mapping to the output space using adversarial training. [42, 43, 53, 21, 89] have shown tremendous editing capabilities on images outside of the training data through inversion [42, 43, 68], showcasing the semantic capabilities of the underlying latent space. Most recently, diffusion models [37, 72, 75, 61] have shown improved capabilities over VAEs and GANs by generating high-quality and detailed datapoints while showcasing similar or improved editing capabilities.

1.1 Representation Learning for Videos

Despite the tremendous advancements in representation learning within the computer vision community [42, 43, 53, 89, 24, 41, 73], progress specifically in video representation learning has been somewhat limited. Unlike images, videos contain an additional temporal dimension that captures changes in content over time. While some approaches treat videos as an extension of images, this oversimplifies the unique properties of the medium. For instance, existing architectures [97, 95, 90, 111] attempt to disentangle motion and content by focusing on a single frame to model the content and the optical flow across frames to model motion. However, this approach can limit the expressiveness of the video to frame-based operations, as demonstrated in this thesis. For example, consider the task of editing a video. With frame-by-frame architectures, each frame must be edited in isolation, requiring the development of complex, temporally consistent editing networks such as [97, 95, 90].

Video-based tasks can be broadly categorized into two fundamental categories: classification and generation. Classification involves reducing high-dimensional datapoints into a compressed space that can be used to differentiate between different classes. Conversely, generation begins with a compressed space and learns a transformation to the high-dimensional output space. This thesis explores representation learning for videos in both these complementary tasks. By exploring both aspects of video analysis, we aim to develop a more comprehensive understanding of video representation learning, which can ultimately lead to more effective and powerful video-based applications.

1.1.1 Video Classification: Lipreading

The thesis starts by tackling the classification task of lipreading, which involves detecting words from mouth movements in the form of speech or text. While lipreading has been extensively studied in computer vision [16, 55, 17, 107], it has not been addressed in a medical setting. Individuals with speech-related disabilities such as Ataxia or Amyotrophic lateral sclerosis (ALS) rely on lipreading for day-to-day communication. Still, their mouth movements differ significantly from those without disabilities, making it challenging for models trained on large-scale datasets [16] of individuals without disabilities to perform well in this context.

To address this challenge, we create 200 classes of words and classify mouth movements (video) into corresponding words (classes). However, a primary challenge in this setting is the need for more data to train a model. To overcome this challenge, we augment the one-shot dataset using synthetic lip-synced videos [70]. Then, we use a novel domain-adaptation strategy called Variational Encoders to augment the real datapoints directly in the compressed space.

Variational Encoders, illustrated in Fig. 2.3, are based on VAEs, but differ in that the compressed space is trained using a classification objective instead of a reconstruction (e.g., $||X - X'||_p$) objective. Specifically, given a video V, we first reduce it to a latent vector z_v (sampled from a distribution $\mathcal{N}(\mu_v, \sigma_v)$ predicted by an encoder $\mathcal{E}_v$). Rather than reconstructing V using a decoder, we classify z into the corresponding class. Unlike VAEs, which regularize the compressed space against a known distribution (like Gaussian) by minimizing KL divergence, Variational Encoders regularize the space through the classification objective. This results in a multimodal compressed space, where each mode corresponds to a particular class. This formulation induces noise in the latent space during training, allowing the downstream classifier to see variations of an example in the latent space.

1.1.1.1 Motivation for Video Generation

Generative modeling fundamentally aims to generate novel datapoints that fit a certain distribution. While many existing works focus on generating data directly in the output (image) space [90, 97, 95], this work takes a different approach by generating novel datapoints in the compressed space. Specifically, our work predicts a distribution for each datapoint, which is then regularized using a downstream classification loss. However, the resulting learned representation is specific to the task of lipreading the set of words it is trained on.

While an auto-encoding setup trained in a self-supervised manner could learn task-agnostic representations, generating videos (the decoding step) by itself is a complex task requiring modeling and upsampling in both the spatial and temporal dimensions. This complexity makes designing an architecture for video generation extremely challenging. To tackle this challenge, the next part of the thesis focuses on designing a representation space for video generation that enables diverse downstream video-based generative tasks. By doing so, we hope to enable more sophisticated generative models for a variety of applications.

1.1.2 A Novel Representation Space for Video Generation

Generating videos is a complex task that involves learning a transformation from a low-dimensional space ($1 \times K$, where K is the dimension of the latent space) to a high-dimensional video datapoint. One of the primary sources of complexity arises from the spatiotemporal nature of this modality, where one has to model both - the content and the motion. To tackle this, existing works model videos frame-by-frame, often extending an image-based generative network. In such a setting, video generation relies on an image space (such that each point in the latent space corresponds to an image in the output space), and a temporally coherent trajectory is sampled to generate the video frame-by-frame. Such a complex architecture and modeling technique makes executing downstream tasks (such as video inversion, video prediction, and superresolution) complex.

We instead propose a video representation space in which each latent point corresponds to a full video instead of an image. To achieve this, we rely on a recent parameterization scheme of "implicit neural representations" (INR) that generates a video one pixel at a time by taking a pixel location as input ($f(x, y, t) \rightarrow RGB_{xyt}$, where f is an MLP-based neural network, x, y, and t are the spatial and temporal pixel location in the given video). Next, we use a meta-network (a hypernetwork [35]) that is trained to generate a function f_v corresponding to a video v conditioned by a latent vector l_v. In this manner, v is encoded to a single latent vector l_v sampled from an underlying distribution, Φ called the video space. This approach reduces the complexity of downstream tasks by eliminating the need for modeling and upsampling at both the spatial and temporal dimensions. Our proposed parameterization scheme enables many downstream tasks, including (but not limited to) unconditional video generation, video inversion, video interpolation, video prediction, video superresolution, and frame interpolation.

1.1.3 Related Publications: Applications of Representation Learning for Videos (do not form contribution for the thesis)

Learning robust representations for videos has diverse applications, for example, syncing lip movements to audio [15], generating lip movements from audio [70, 40, 82], action recognition [103, 105, 32], pedestrian path prediction for autonomous driving [47], etc.

One exciting application is in the creative industry, specifically face-swapping for double and the actor in movies. Doubles play an indispensable role in the movie industry. They replace the actors in dangerous stunt scenes or scenes where the same actor plays multiple characters. The double's face is later replaced with the actor's face and expressions manually using expensive CGI technology, costing millions of dollars and taking months to complete. An automated, inexpensive, and fast way can be to use face-swapping techniques to swap an identity from a source face video (or an image) to a target face video. However, such methods cannot preserve the source expressions of the actor important for the scene's context. To tackle this challenge, [5] introduces video-to-video (V2V) face-swapping, a novel task of face-swapping that can preserve (1) the identity and expressions of the source (actor) face video and (2) the background and pose of the target (double) video. This work proposes FaceOff, a V2V face-swapping system that operates by learning a robust blending operation to merge two face videos following the constraints above. It reduces the videos to a quantized latent space and then blends them in the reduced space. FaceOff is trained in a self-supervised manner and robustly tackles the non-trivial challenges of V2V face-swapping. FaceOff significantly outperforms alternate approaches qualitatively and quantitatively.

A different application involves the healthcare industry. Many people with some form of hearing loss consider lipreading as their primary mode of day-to-day communication. However, finding resources to learn or improve one's lipreading skills can be challenging. This is further exacerbated in the COVID19 pandemic due to restrictions on direct interactions with peers and speech therapists. Today, online MOOCs platforms like Coursera and Udemy have become the most effective form of training for many types of skill development. However, online lipreading resources are scarce as creating such resources requires months of manual effort to record hired actors. Because of the manual pipeline, such platforms are also limited in vocabulary, supported languages, accents, and speakers and have a high usage cost. [6] investigates the possibility of replacing real human talking videos with synthetically generated videos. Synthetic data can easily incorporate larger vocabularies, variations in accent, and even local languages and many speakers. The authors propose an end-to-end automated pipeline to develop such a platform using state-of-the-art talking head video generator networks, text-to-speech models, and computer vision techniques. The authors then perform an extensive human evaluation using carefully thought-out lipreading exercises to validate the quality of the designed platform against the existing lipreading platforms. The studies concretely point toward the potential of the proposed approach in developing a large-scale lipreading MOOC platform that can impact millions of people with hearing loss.

1.2 Contributions

The thesis investigates representation learning in the context of videos, which is a complex modality that encompasses both spatial and temporal dimensions, mimicking human perception. To achieve this, the thesis delves into two fundamental aspects of video-based machine learning tasks: classification and generation. The first contribution of the thesis is in the area of classification, where it addresses the challenging task of lipreading. The second contribution is in the area of generation, where the thesis proposes a novel parameterization scheme of a video space that facilitates diverse downstream tasks, such as video inversion, interpolation, prediction, and superresolution. In summary -

1. The thesis tackles lipreading in a one-shot setting to lipread speakers with speaking disabilities. A novel domain-adaptation approach called Variational Encoders is proposed that augments the one-shot datapoints directly in a compressed space and improves the model's performance.

2. A novel video representation space is introduced based on implicit neural representations (INRs) and a hypernetwork that learns a prior (video space) over the INRs. This approach enables many downstream video-based generative tasks, including unconditional video generation, video inversion, video interpolation, video prediction, video superresolution, and frame interpolation.

3. The thesis also comprehensively evaluates the proposed methods on various benchmark datasets, demonstrating their effectiveness and efficiency compared to state-of-the-art methods. A thorough evaluation of the proposed video space that showcases the exact nature of the representation space is also presented.

4. Qualitative results clearly showcasing the improvements over the existing baselines are added.

1.3 Organization of Thesis

1. In Chapter 2, we introduce the task of one-shot lipreading specifically for an ALS patient and propose a novel representation technique to self-augment the dataset directly in the compressed space. We test our proposed method on five speakers suffering from different disabilities (such as ALS, deafness, and no disability).

2. In Chapter 3, we tackle the video-based generative tasks and propose a novel representation space - video space - that represents videos directly in the compressed space. In this, we parameterize videos as INRs and use hypernetworks to learn a prior over the INRs. We also provide several applications of video generation, including video interpolation, video inversion, and video super-resolution, and evaluate the representation space through extensive experimentation.

3. We present our concluding thoughts in Chapter 4.

Chapter 2

Video Classification: One-shot Lipreading for an ALS Patient

Lipreading is the skill of recognizing speech visually from a person's lip movements. Humans naturally rely on lipreading to discern speech, especially in crowded and noisy environments [76]. It is the fundamental mode of communication for many people, such as (1) those suffering from medical conditions such as Amyotrophic Lateral Sclerosis (ALS) - leading them to lose their voice [112, 56], or (2) those with hearing impairment - making it difficult for them to produce proper voice. In such cases, talking to a person without voice may need you to lipread them to understand the spoken words.

Lipreading is mentally taxing and can affect the communication quality. For instance, speakers with hearing impairment may lack holistic audio feedback [98] and ALS patients may have lesser control over their mouth muscles [7, 54]. This may cause them to have irregular and unreliable mouth movements making it difficult for people to lipread them. Applications and automated algorithms capable of lipreading a person can thus significantly improve the day-to-day communication of people dependent on lipreading. Motivated by this need, we tackle the real-world challenge of lipreading a patient suffering from ALS and people with hearing impairment. ALS is a progressive nervous system disease that affects nerve cells in the brain and spinal cord, causing loss of muscle control [112]. An ALS patient may lose their voice and rely solely on mouth movements for communication [38, 62].

2.1 Current Works and Limitations

Current deep learning techniques are inherently data-hungry. Collecting large amounts of data specifically from a patient is, however, not an option. Mouthing words is a tiring maneuver for people suffering from ALS, and thus a patient undergoes physical and mental stress during such data collection exercises. Manually labeling words mouthed by a person is time-consuming. It is thus crucial to use the minimum amount of manually labeled data to build lipreading models that can work well on a person.

Recent years have seen much progress in word-level [55, 25, 96] and sentence-level [2, 107] lipreading. Oxford's Visual Geometry Group released large-scale in-the-wild datasets such as Lipreading in the Wild (LRW) [17] and Lipreading sentences (LRS) [1, 3] consisting of 1000+ speakers. LRW, the most relevant dataset to our task, is a word-level lipreading dataset made of 1000 examples for 500 English

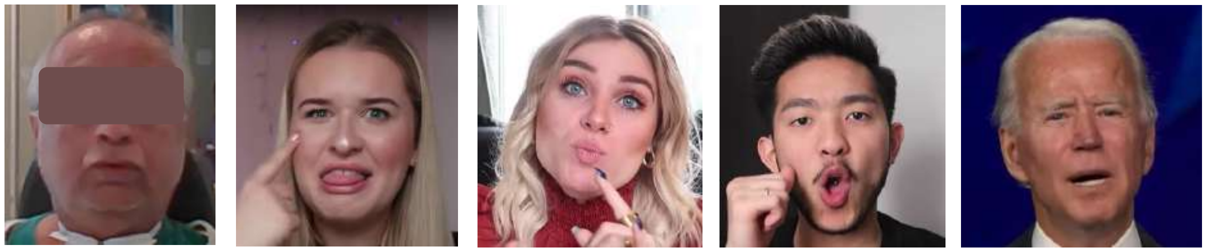


Figure 2.1: Speakers in our study. In left to right order, the first speaker suffers from ALS and relies solely on lip movements. The next two speakers primarily use sign language while making imperfect lip movements. Next speaker uses deaf speech along with sign language for daily communication. The following speaker is the 46th president of the USA, Joe Biden.

words but turns out to be somewhat limiting: (1) The speakers in the dataset do not have any speaking disability thus making perfect mouth movements. (2) It is curated by cropping words from long speech segments resulting in fast-paced speech with co-articulation in the videos. (3) It contains a large amount of head motion and variations like the different characteristics of the mouth region, both of which are unnecessary for lipreading specific medical patients. Thus, SOTA models like LipReading without Pains (LRwP) [25] and Lipreading using Temporal Convolutional Networks (LTCN) [55] trained on LRW do not directly adapt to speakers with speaking disabilities.

LRW only supports a limited pre-curated vocabulary missing out on medically essential words like 'nauseous' or 'backache'. It also lacks a personalized vocabulary relevant to a person's daily communication. Deploying systems to enable a persons' communication would need highly accurate models on their specific lip movements for their particular vocabulary. The problem of personalized lip reading has also been explored in [69]. According to [69], lip movements vary across speakers. Observing a single speaker for an extended period could lead to better speaker-specific lipreading models. They collect $\sim$20 hours of data per speaker to train a personalized lipreading model generating speech purely from an individual speaker's lip movements. Collecting such a large dataset is, however, not always an option.

Lack of medical data has been studied [93, 27, 34] widely in the past. Taking inspiration from these, we formally tackle one-shot lipreading in a personalized setting. We first synthetically generate data using a SOTA lipsync network [70]. We then use a SOTA lipreading network [25] as our backbone and use synthetically generated data along with very limited real examples to train a word-level lipreading network. Our approach includes an important domain adaptation step using a novel network – Variational Encoders – a modified VAE for bringing a vast number of synthetic examples closer to the real domain using only one real instance per class. We specifically tackle the use case of an ALS patient and also explore the same for four other speakers (refer to Fig. 2.1) including the 46th President of the United States, Joe Biden, as an additional example to show that our approach can easily be extended to speakers with no disability. Our contributions in this chapter are threefold:

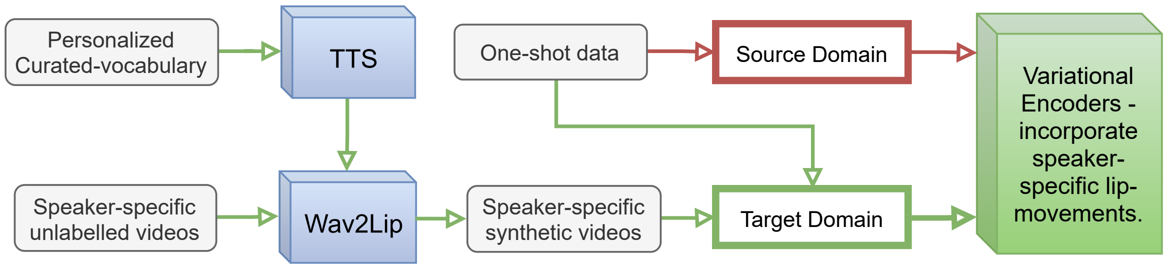

Personalized Curated-vocabulary
TTS
One-shot data
Source Domain
Variational Encoders - incorporate speaker-specific lip-movements.
Speaker-specific unlabelled videos
Wav2Lip
Speaker-specific synthetic videos
Target Domain

Figure 2.2: Personalized lipreading - We start by curating personalized vocabulary and collect unlabeled videos for each speaker. TTS models are used to generate speech utterances for the curated words. The speaker's unlabeled videos and the generated speech utterances are given as input to Wav2Lip that generates synthetic data to augment one-shot data. Variational Encoders then use the synthetic and the one-shot data to train the model for lipreading.

1. We tackle a real-world medical challenge of lipreading speakers with ALS and hearing impairment by developing highly accurate personalized models for each speaker.
2. We propose Variational Encoders, a novel network-based on VAE. Instead of autoencoding, they exploit the loss of the downstream task for generating task-relevant latent distributions. The learned distributions are then used for domain adaptation.
3. To the best of our knowledge, we are the first to propose lipreading in one-shot setting. In this vein, we curate a medical dataset involving speakers with medical conditions.

2.2 One-shot personalized lipreading framework

As mentioned previously we aim to build a personalized lip-reading model for each speaker using only single real examples. Fig. 2.2 presents the pipeline for personalized lipreading.

The use of synthetic data to augment low data in the medical domain [93, 27, 34] has gained traction with improving generative models. Similarly, we augment the one-shot examples collected for each speaker by generating synthetic data for each of them. To achieve this, we use a SOTA talking face generation model, Wav2Lip [70] pretrained on the large-scale LRS2 [16] dataset. Given a speaker's video, Wav2Lip preserves the speaker's pose, facial expressions, and mouth characteristics like beard and skin color while modifying the speaker's lip movements according to a guiding speech. We generate word-level speech utterances using SOTA TTS models FastSpeech2 [74] and GlowTTS [44] as a replacement for the speaker's voice. Using these TTS models allows us to generate variations in the speech in terms of the speed of the spoken word, pitch, and energy. Additionally, we collect unlabeled face videos for each speaker which, along with the generated speech utterances, is used to create 1 hour of speaker-specific synthetic data on an average. The augmented dataset is then used to train an LRwP and LTCN based architecture for the classes curated per speaker.

A combination of the speaker-specific synthetic videos and a single real video per class are used to train our model. The synthetic data helps the model learn the general underlying word-level characteristics for the new classes. However, the properties of personal style of lip-movements for a word – could be because of the medical condition – is not captured in the synthetic dataset. We utilize the one-shot examples for introducing the properties of personal speaking style in the model. Single examples per class are however, not enough to capture the underlying style variations of a speaker. A person may not utter the same word, exactly, each time. To tackle this, we use our novel approach – Variational Encoders.

2.2.1 Variational Encoders: Mapping words into distributions

Deep learning suffers from the fundamental challenge of source-target domain shift - a model trained on a given dataset (source domain) performs poorly on the test examples (target domain). The target domain may lack the amount of labeled data needed for training or fine-tuning a model. Recent techniques include adversarial networks to generate domain invariant features using adversarial losses [28, 59, 100] and Variational Auto Encoders (VAE) [39, 94] to generate a joint latent distribution across domains with KL divergence and reconstruction loss. Unlike adversarial loss [28] that may quickly become lopsided [22], VAEs use a distance-based metric to incorporate domain invariance. The reconstruction loss in VAE ensures that latent representations preserve important domain characteristics. However, using a decoder to reconstruct the input accurately is a non-trivial task, especially for videos needing spatial and continuous temporal reconstruction. We propose Variational Encoders, a modified VAE that uses the loss of the downstream task to generate task-relevant latent distributions.

How do Variational Encoders differ from a standard VAE? Similar to VAEs, Variational Encoders (refer to Fig. 2.3) generate latent distributions given a sample. A key difference with a standard VAE is - instead of autoencoding, it uses the final multi-class classification loss of the downstream task to generate a task-relevant latent distribution that represents the “class” of input instead of a generic input feature. The learned distribution is used to sample a variation of the input example. We hypothesize that this sample introduces the missing speaker-specific style variation for the input class. The sampled variation encourages the encoder to see and align to the potential interpretation of the input class and learn robust representations.

In summary, Variational Encoder retains all the benefits of a VAE, like generating domain invariant features while removing the complicated video reconstruction loss.

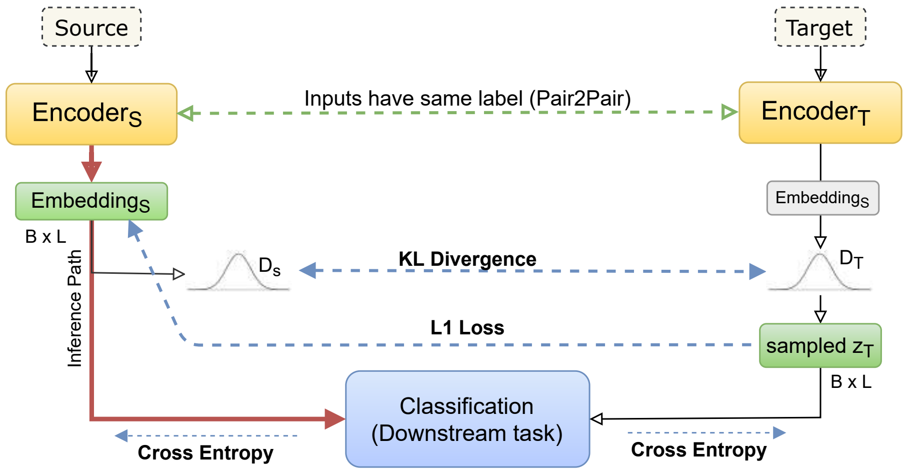


Figure 2.3: Variational Encoders – Encoder$_S$ and Encoder$_T$ denote the source and target encoders respectively. Encoder$_T$ generates a latent distribution using the target examples. The latent distribution is trained against the downstream classification task. Encoder$_S$ aims to learn robust domain invariant features by minimizing the distance between its learned embedding and points randomly sampled from the generated target distribution.

2.2.1.1 Network Architecture

Given a source domain S and a target domain T, domain adaptation aims to bridge the gap between the two domains by generating domain invariant features. We denote S and T as,

$$S = \{(x_1, y_1), (x_2, y_2), ...(x_{N_S}, y_{N_S})\} \text{ and } T = \{(\bar{x}_1, \bar{y}_1), (\bar{x}_2, \bar{y}_2), ...(\bar{x}_{N_T}, \bar{y}_{N_T})\} \tag{2.1}$$

where N_S is the number of source samples and N_T is the number of target samples.

Variational Encoder is a paired domain adaptation network that assumes the target domain is labeled. However, the number of examples in the target domain is expected to be at max k where $k \approx 1$. Thus, we assume $N_S >> N_T$. The adaptation network is made of two encoders - source and target encoders - and a single classifier. For a given input $\hat{x}_i$ belonging to either domain, the encoder learns to generates an embedding $e_{\hat{x}_i}$ for the input. A distribution $p(\hat{x}_i)$ is then generated for the input denoted by $\mu_{\hat{x}_i}$ and $\sigma_{\hat{x}_i}$ from the learnt embedding $e_{\hat{x}_i}$. A random point is sampled from the generated distribution denoted as $z_{\hat{x}_i} = \mu_{\hat{x}_i} + \sigma_{\hat{x}_i} \bigodot \epsilon$.

The encoder network for both domains are identical. Each encoder generates the embedding e, and the latent distribution p. To introduce domain invariance, we use KL divergence and L1 loss. The KL divergence is computed between the two generated distributions $p(x_i) = (\mu_{x_i}, \sigma_{x_i})$ and $p(\bar{x}_i) = (\mu_{\bar{x}_i}, \sigma_{\bar{x}_i})$. In addition, a point $z_{\bar{x}_i}$ is randomly sampled from the target distribution $p(\bar{x}_i)$. L1 loss is then applied between $z_{\bar{x}_i}$ and the source embedding e_{x_i}. This minimizes the distance between the source embedding against several randomly sampled points from the target distribution thus acting as a pseudo for multiple target examples.

Gradient stopping: The L1 Loss between the source embedding and the sampled target embedding will force the target distribution down to a single point, losing the essence of a distribution. To prevent that, we stop the gradient from flowing back through the target encoder. This way, only the source encoder is regularized against the variations sampled from the learnt target distribution, while leaving the target encoder unaffected. We denote the sampled target embedding as $z_{\bar{x}_i detach}$, where $_{detach}$ denotes that the sample $z_{\bar{x}_i}$ does not have any gradient. The combined loss to bridge the two domains is given as,

$$\Delta_{dist} = |e_{x_i} - z_{\bar{x}_i detach}| - \beta \cdot D_{KL}(p(x_i)||p(\bar{x}_i)). \tag{2.2}$$

where β is a hyper-parameter and i denotes both the samples belong to the same class.

For the downstream task of classification, we employ a common classifier that is trained on both domains at the same time. A single classifier allows both encoders to have a common base-loss helping them generate features relevant to the common downstream task. The downstream classifier also trains on large amounts of variations and learns robust representations. The classifier receives a randomly sampled point $z_{\bar{x}_i}$ from the learnt distribution $p_{\bar{x}_i}$ as input for the target domain. For the source encoder, however, the classifier receives the learnt embedding e_{x_i} instead of a randomly sampled latent point z_{x_i}. The motivation behind a target distribution is to encourage the target encoder to hallucinate variations of the single real examples available for each class. However, during inference, we want to obtain a definite point for the input to avoid any uncertainties. The classification loss is given as,

$$\Delta_{entropy} = g(e_i, y) + g(\bar{z}_i, y) \tag{2.3}$$

where g can be any classification loss such as cross entropy or negative log likelihood. y denote the same label for both the source x_i and target $\bar{x}_i$ input.

The combined loss for the entire network is then given as (see Fig. 2.3),

$$\Delta_{ve} = \alpha \cdot (g(e_i, y) + g(\bar{z}_i, y)) + \gamma \cdot (|e_{x_i} - z_{\bar{x}_i detach}| - \beta \cdot D_{KL}(p(x_i)||p(\bar{x}_i))). \tag{2.4}$$

where α and γ are hyper-parameters. All the network components are trained end-to-end.

2.3 Experiments

Dataset: We first collect a set of unlabeled videos for each speaker that are used later to generate the speaker-specific synthetic data using Wav2Lip. Unlabeled videos of the ALS[1] patient are recorded without any manual intervention. For other s peakers, the videos are randomly selected from their respective YouTube channels. As the next step, we curate the personalized vocabulary for each of the speakers. For the ALS patient, we curate a list of 200 words with the help of his family. We modify the existing list for the remaining speakers by removing irrelevant words and adding the most occurring keywords in the transcription of the collected videos. Splits of the curated dataset for each speaker is presented in Table 2.1.

[1]I would like to thank Anuraag Mullick and Vibha Mullick for their continual support in curating the dataset needed for this work. I also thank Harini Bhatt, who is the founder of ASRM Systems, for connecting me with several participants.

		Patient	**Spk-1**	**Spk-2**	**Spk-3**	**Joe Biden**	**Total**
	Classes	200	75	70	80	75	-
Train	Real	200	75	70	80	75	-
	Synth	17×Real	22×Real	22×Real	22×Real	25×Real	-
	Aug.	2×Real	2×Real	2×Real	2×Real	2×Real	-
Test	Real	320	120	80	90	90	710

Table 2.1: Split up for the datasets curated for each of the 5 speakers.

We obtain two sets of manually curated real data from the ALS patient's family. We use the first set for model training and the second set as the intermediate test set. We collect additional data by deploying a website that records the patient's word-level mouthings and displays the inference on our best model. The website needs external help to start and stop recording the patient. The helper can then either select one of the displayed words as the correct label or manually assign the correct label. Through this exercise, we collect an additional 320 examples. Out of these, we use 200 data points, in addition to the original 200 train data points, to train a model in a two-shot setting. We report our test results of one-shot and two-shot models on the combined set of the intermediate test set and the remaining additional 120 examples (see Table 2.1). To simulate the same setting, we maintain one real example for training the rest of the speakers. To generate the train and test examples, we use the transcriptions with timestamps available on YouTube for the selected videos.

Preprocessing: Our preprocessing steps are similar to LRwP. The lip landmarks are first detected using dlib [45]. The lip is then cropped out such that it is horizontally and vertically centered in the cropped image. The image is converted to gray-scale and resized to a fixed dimension of 88×88. A maximum sequence length of 64 frames is used. A batch size of 16 on a multi-GPU NVIDIA GeForce RTX 2080 setup is used. The models are trained up to 200 epochs using a cosine scheduler and Adam optimizer, with a $3e-5$ learning rate and a weight decay of $1e-4$. The encoders in the experiments are adopted from LRwP that uses Resnet18 and BiGRU, and LTCN that uses Resnet18 and Temporal Convolutional Networks.

2.3.1 Training strategy

Table 2.2 presents a comprehensive overview of all the experiments conducted on the 5 speakers. Spk-1 uses deaf-speech and sign language, Spk-2 and Spk-3 use sign language as their primary mode of communication while mouthing words with imperfect lip movements. Table 2.3 presents additional experiments performed on the dataset of the ALS patient. We initialize our models with the weights of LRwP or LTCN, both of them pretrained on LRW. We observed that using the pretrained weights leads to faster convergence.

	Patient		**Spk-1**		**Spk-2**		**Spk-3**		**Joe Biden**		**Avg.**	
Experiment	top1	top5	top1	top5	top1	top5	top1	top5	top1	top5	top1	top5
LRwP												
cl−r	49.3	62.6	32.4	52.1	23.1	42.5	26.7	49.3	33.5	51.5	33.0	51.6
cl−r+s	53.6	68.1	32.3	49.8	**37.2**	45.1	32.4	58.8	57.6	69.3	42.6	58.7
ve−r+s	66.4	81.4	**48.6**	**68.5**	34.4	**59.6**	41.6	71.4	54.1	72.4	49.0	**70.4**
cl−r+s+aug	61.3	75.2	34.6	51.2	33.4	40.4	51.2	63.6	**64.5**	75.3	49.4	61.3
ve−r+s+aug	**68.1**	**83.2**	44.3	62.6	31.2	50.8	**53.4**	**73.5**	61.4	**77.1**	**52.2**	69.4
LTCN												
cl−r+s	55.2	67.1	36.5	52.0	34.8	44.8	34.4	59.6	55.4	67.3	43.3	58.2
ve−r+s	64.5	80.2	**49.3**	**71.1**	**35.7**	**57.9**	46.4	73.2	56.3	71.1	50.4	**70.7**
cl−r+s+aug	61.9	74.2	33.2	49.8	32.4	39.9	52.5	66.1	**62.1**	**76.2**	48.4	61.2
ve−r+s+aug	**66.8**	**81.7**	41.7	60.4	32.4	48.8	**54.9**	**75.2**	61.9	75.3	**51.5**	68.3

Table 2.2: Evaluation of our models against each speaker reported in %. All metrics are evaluated on the curated test set made of only real-data. Spk-1 uses a combination of deaf-speech and sign language, Spk-2 and Spk-3 use sign language for communication. Patient denote the ALS patient in our study. r, s, and aug indicate real, synthetic, and augmented-real datapoints. cl indicates standard classification while VE is the proposed technique.

Baseline (Exp-*cl−r*): We begin by training our model directly on the one-shot examples. Since we are the first to perform one-shot lipreading on a personalized vocabulary, we treat this model as our baseline. Table 2.2 Exp-*cl−r* presents the performance of the baseline model on each of the 5 speakers. The average accuracy of the speakers at top-1 and top-5 is only 33.0% and 51.6% respectively. The accuracy of the current SOTA lipreading model on the LRW dataset is 88.5% at top-1. This presents us with a huge scope for improvement.

Data Augmentation using Synthetic Data (Exp-*cl−r+s*): We augment the one-shot examples with a potentially unlimited number of synthetic examples. We train our models with varying amounts of synthetic data. We plot a graph of accuracy against the combined synthetic and one-shot examples to determine the optimal amount of synthetic data needed for each speaker against the one-shot examples as shown in Fig. 2.5. The optimal number of synthetic examples per speaker is reported in Table 2.1.

As shown in Table 2.2 Exp-*cl−r+s*, we observe a significant jump in the accuracy consistently for every speaker with an overall improvement of 9% and 7% at top-1 and 5, respectively on LRwP, and 7% and 12% at top-1 and 5, respectively on LTCN. Although the accuracy improved over the baseline model, we observe that the model overfits on the synthetic data after a few epochs. Since we use only one example per class for the real domain, the model cannot foresee the target (real) variations that it may encounter during testing. We, therefore, try to introduce variations using Variational Encoders.

Experiment		top-1	top-3	top-5
One-shot (classification)	only fs tts	56.83	68.46	71.92
	only glow tts	57.56	67.73	71.55
	combined	**61.38**	**73.63**	**75.26**
Two-shot	classification	64.33	74.69	82.16
	variational encoders	**71.64**	**76.54**	**89.36**

Table 2.3: Accuracy of additional experiments performed on the ALS patient in %. The test dataset used for both, One-shot and Two-shot experiments is the same. FastSpeech2 and GlowTTS are represented by fs tts and glow tts, respectively. All experiments are conducted using LRwP as the backbone on the combined real, synthetic, and augmented real datasets.

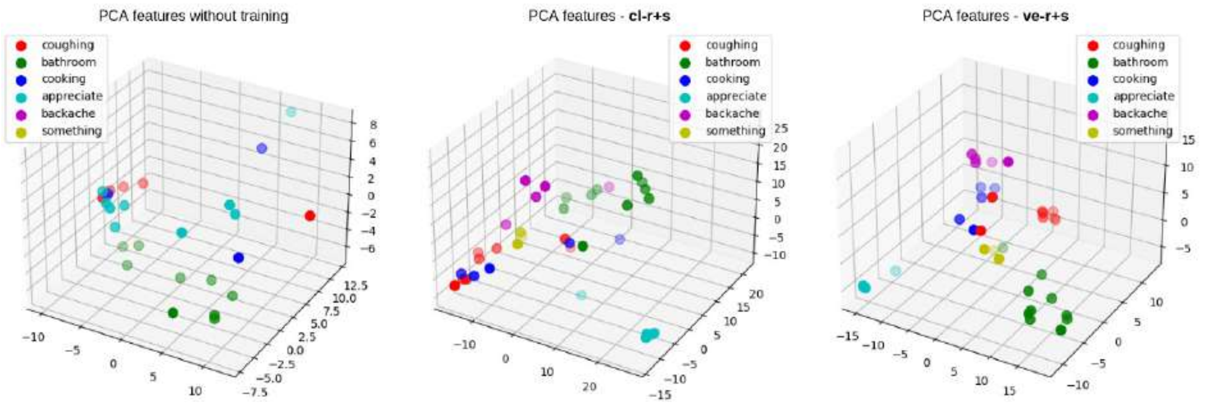


Figure 2.4: PCA visualization for the embeddings generated by the feature extraction layer of LRwP based encoder. (left) before training, (middle) trained on $cl{-}r{+}s$, and (right) trained on$ve{-}r{+}s$. Samples from the ALS patient's test set are used for visualization

Variational Encoders (Exp-$ve{-}r{+}s$): To train the network, we use the one-shot examples as the source domain. For the target domain, we combine the real and synthetic dataset. This lets the source encoder see the existing real examples and also become robust against the additional pseudo examples sampled from the target distribution. We observed that using only the synthetic data in the source encoder makes the training highly unstable and the network does not converge. Instead, we allow the source encoder to first fit and then improve.

As seen from Table 2.2 Exp-$ve{-}r{+}s$, Variational Encoders consistently achieves the highest accuracy at top-5 across LRwP and LTCN. There is an overall improvement of $\sim 19\%$ over the baseline and $\sim 12\%$ over Exp-$cl{-}r{+}s$. The improvement at top-1 accuracy is comparatively much lower, $\sim 15\%$ over the baseline and only $\sim 6\%$ over Exp-$cl{-}r{+}s$. We use PCA visualizations to analyze the latent representations learnt by the LRwP based source encoder, as shown in Fig. 2.4 (right). We observe better separation for each class compared to the separation of Exp-$cl{-}r{+}s$ (middle). We also observe better disentanglement between non-homophenes such as ‘bathroom’, ‘appreciate’, and ‘coughing’ compared to Exp-$cl{-}r{+}s$. For Varitional Encoders we observe that homophenes ‘coughing’, ‘cooking’, ‘something’ are closer together. This suggests that the model trained on Variational Encoders can get confused for homophenes bringing the accuracy at top-1 down and at top-5 higher. For Exp-$cl{-}r{+}s$ however, all the classes seem equally apart.

Ad-hoc Data Augmentation (Exp-$cl{-}r{+}s{+}aug$ and Exp-$ve{-}r{+}s{+}aug$): In addition to the implicit variations introduced by Variational Encoders during the model training, we introduce explicit real-domain variations by augmenting the one-shot dataset. First, we use moviepy [60] library to speed up and speed down the one-shot videos by a factor of $1.2\times$ and $0.8\times$. The videos in our work are unconstrained, that is, the actual mouthing could be spread across several frames placed temporally anywhere in the video. Thus, we increase the video frames sequence length to 85 and add temporal variations during training by padding the videos with random number (between 0 to 20) of empty frames at the start and end.

We observe improvement in the performance across speakers for both, classification (Exp-$cl{-}r{+}s{+}aug$) and Varitional Encoders (Exp-$ve{-}r{+}s{+}aug$). We observe that for Spk-1 and Spk-2, the performance degrades. Upon further analysis, we find that the videos maintain constant pace and have fewer overall variations. Thus, the added variations during training behave as noise driving the performance down. The overall performance improves by 7% and 3% at top-1 on Exp-$cl{-}r{+}s{+}aug$ and Exp-$ve{-}r{+}s{+}aug$ over Exp-$cl{-}r{+}s$ and Exp-$ve{-}r{+}s$ respectively. We observe the best performance on one-shot setting with these ad-hoc additions for classification. For Variational Encoders, the improvements are less significant, especially at top-5, indicating the technique itself makes up for these ad-hoc additions.

Additional Experiments on the ALS Patient’s Dataset: To observe the affect of using different TTS models for generating synthetic data, we perform an ablation by eliminating one TTS model at a time for data generation and compare its performance against the data generated by combining both the TTS models. As shown in Table 2.3, the performance with both the TTS combined gives us the best performance. This indicates that the variations introduced by different TTS models are important for generalization.

Lastly, we train an additional model with the additional data obtained from the patient’s family to evaluate the performance difference between one-shot and two-shot setting. As seen from Table 2.3, the performance improves by $\sim 7\%$ at top-5 against the best performing one-shot model on both training methodologies. Specifically for Variational Encoders, the performance improves to 71.64% and 89.36% at top-1 and top-5. This suggests that the network learns a better real-domain latent distribution using the extra real examples.

Evaluation on Joe Biden: Although Joe Biden represents speakers without disabilities, he speaks in an American accent, while LRW is composed of British speakers. As can be observed from Table 2.2 Exp-$cl{-}r$, the performance for Joe Biden on the baseline model is similar to the performance for the deaf speakers. A different accent can also be thought of as a different style of speaking. Thus the pretrained model does not directly adapt to Joe Biden. We observe in his case, adding synthetic data leads to significant improvements in the accuracy over the baseline with an average gain of 21% at top-1 and top-5 (Exp-*cl*−*r*+*s*). Variational Encoders fail to bring expected improvements compared to the other speakers. With LRwP, at top-1, the accuracy drops by 3%, and we see a marginal improvement at top-5. We note that the TTS used to generate synthetic data is of an American accent. This suggests that the synthetic data captures the variations of the real domain exceptionally well in his case. Variational Encoders, on the other hand, adds more noise than valuable variations.

2.4 Ablation Experiments

In this section, we report ablation experiments to inspect and gain further insights on the different components of our network. Our experimental setup is identical to the setup mentioned in Section 3 of the main paper. For better readability during comparisons, we have copied the results of Table 2: Exp-*cl*−*r* and Exp-*ve*−*r*+*s* of the main paper to Table 2.4.

	cl−r		ve−r		ve−r+s		ve−r+s−mod	
Speaker	top1	top5	top1	top5	top1	top5	top1	top5
ALS patient	49.3	62.6	51.2	65.4	66.4	81.4	64.6	74.9
Spk-1	32.4	52.1	36.9	48.9	48.6	68.5	48.3	60.3
Spk-2	23.1	42.5	30.4	44.7	34.4	59.6	31.5	57.7
Spk-3	26.7	49.3	35.7	59.6	41.6	71.4	39.7	65.3
Joe Biden	33.5	51.5	41.5	57.8	54.1	72.4	54.4	70.8
Avg.	33.0	51.6	**39.1**	**55.2**	**49.0**	**70.6**	47.7	65.8

Table 2.4: Evaluation of our models against each speaker in %. All the metrics are evaluated on the curated test set made of only real-data. Exp-cl−r is the baseline trained on one-shot examples. Exp-ve−r denotes Variational Encoders trained on only one-shot examples. Exp-ve−r+s denotes Variational Encoders trained on synthetic and one-shot examples. Exp-ve−r+s−mod is Variational Encoders trained with no distance loss. All experiments are performed using LRwP as the backbone on real + synthetic dataset.

2.4.1 Variational Encoders on Only Single Real Examples

As reported in Section 3.1 of the main paper, our network is trained on the combined data of synthetic and one-shot examples. In this section, we remove the synthetic data to see its benefit. Instead of passing

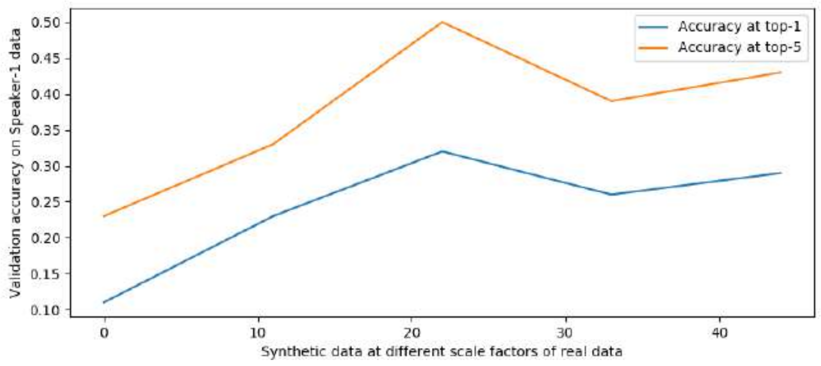


Figure 2.5: Plot to determine the optimal amount of synthetic data against the real examples for Spk-1 at different scale factors of real data. Reference added in the main paper.

synthetic and one-shot examples, we only use one-shot examples as input to both encoders. Since the Variational Encoder itself acts as a data augmentation network, we assume the network to improve over the baseline (Table 2.4 Exp-*cl*$-$*r*).

As shown in Table 2.4 Exp-*ve*$-$*r*, the performance of the reduced network improves by an average of 6% and 4% at top-1 and top-5 respectively over the baseline. However, the performance is still lower than the rest of the experiments mentioned in the main paper (Table 2 Exp-*cl*$-$*r*+*s* and Exp-*ve*$-$*r*+*s*). This suggests that Variational Encoders with just one example cannot surpass methods with additional data augmentation techniques. But, even with only one example, Variational Encoders is capable of bringing significant improvements to the model. Thus, our approach can be a potential data augmentation technique impacting other tasks requiring a low-data setting.

2.4.2 Effect of Δ_{dist}

As reported in Equation (4) of the main paper, Δ_{ve} is a combination of Δ_{dist} that introduces domain invariance by reducing the distance between the feature space of source and target encoder, and $\Delta_{entropy}$ that allows the encoders to generate task-relevant latent embeddings and distributions using the loss of the downstream task. In this section, we remove Δ_{dist} and observe the behavior of the network using only $\Delta_{entropy}$. This allows us to determine the effect of introducing domain invariance. Loss of the reduced network is thus given by,

$$\Delta_{ve} = \Delta_{entropy} = \alpha \cdot (g(e_i, y) + g(\bar{z}_i, y)) \; \{\alpha = 1\} \tag{2.5}$$

As can be seen from Table 2.4 Exp-*ve*$-$*r*+*s*$-$*mod*, the accuracy of the reduced network drops by 1% and 5% across the speakers at top-1 and top-5 respectively over Exp-*ve*$-$*r*+*s*. We observe, $\Delta_{entropy}$ plays a substantial role in our network. Generating variations using Variational Encoders lets the downstream classifier see various examples of the real domain making it robust. We observe the accuracy

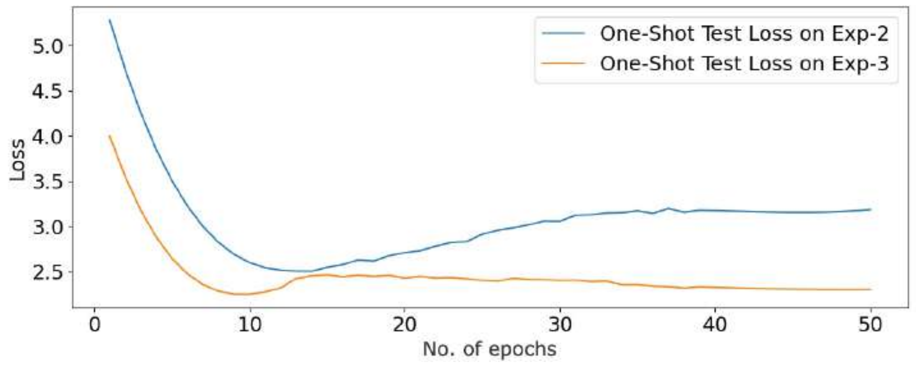


Figure 2.6: Comparison between the test-loss of Exp-cl−r+s in the main paper with Exp-ve−r+s (our proposed approach) for the ALS patient.

drop at top-5 is higher than top-1. The drop suggests that Δ_{dist} may introduce domain invariance across related mouth movements (such as homophenes).

2.5 Need for Variational Encoders

2.5.1 Other Data Augmentation Techniques

In this section we report additional approaches that we tried to augment the one-shot examples. An expert human lipreader can only lipread 40% of a given sentence [9]. Thus, manually labeling unlabeled videos is challenging. We therefore try to automatically label the unlabeled videos.

Best Match using Syncnet - Syncnet [18] is an audio-video synchronization model to synchronize mouth movements and a given speech utterance. It generates a confidence score for a given audio and video segment. We exploit this method to find segments of unlabeled videos closely matching the generated TTS audios. We use different confidence thresholds to label the videos and vary the length of video segments based on the audio length.

Pseudo Labeling - Pseudo Labeling [20, 50] is a technique that iteratively labels the unlabeled data in a semi-supervised setting. A model is trained on the labeled data and is used to label the unlabeled data known as pseudo labels. The model is trained further on the pseudo labels. These steps are repeated until the pseudo labels converge. It assumes the first model trained on the labeled data to be accurate so the pseudo labels are close to the actual labels. Pseudo labels can be noisy, but as the training progresses, labels converge to stable and accurate predictions. We first segment the continuous unlabeled videos into smaller word-level video segments. We use lip-landmarks detected by dlib to measure the rate of change of lip-movements and determine the pauses between words. We then segment the videos on these pauses. We then iteratively label these videos using the pseudo labeling technique.

We observed that these methods led to noisy labels. The performance for each model mentioned Table 2 of the main paper dropped by an average of 15% and 7% on adding pseudo labels and syncnet labels respectively.

2.5.2 Test Loss on One-Shot Examples

Training a classifier on one-shot examples restricts the model from seeing many variations of the real domain and thus tends to underfit the real domain. We show this phenomenon in Fig. 2.6 that plots the test loss of Exp-$cl{-}r{+}s$ (of the main paper) and Exp-$ve{-}r{+}s$. In Exp-$cl{-}r{+}s$, we see loss on the test set flattens and increases after Epoch-12. In Exp-$ve{-}r{+}s$ we see the loss keeps decreasing slowly. This suggests that Variational Encoders introduce meaningful variations of the real domain.

2.6 Chapter Summary

In this chapter, we leap from previous lipreading approaches and propose a one-shot personalized lipreading framework to aid patients suffering from ALS or hearing disabilities. Due to the extreme scarcity of personalized data available for a medical patient, we generate synthetic data to augment our training process. We train our network with Variational Encoders, a domain adaptation technique, to bridge the gap between the synthetically generated examples and the available one-shot real examples for each class. Our method proves to be highly effective, and we achieve over 83% top-5 accuracy for the ALS patient. We also report the performance of speaker-specific models trained for multiple speakers with hearing impairment and a speaker with no disability.

In the next chapter, we extend our current formulation of Variational Encoders that augment one-shot real video examples in the latent space by learning a distribution over classes of videos; We hereby focus on generative modeling for videos, in which, we model a representation space that can be used to generate videos effectively. Essentially, the chapter aims to propose a novel parameterization scheme for videos that allow us to decode videos without explicitly modeling them at an image level, thereby learning a prior over videos enabling us to do many novel video-based generative tasks such as video completion, inversion, and interpolation.

Chapter 3

Video Generation: Implicit Neural Representation for Video-based Generative Space

Learning to generate complex spatio-temporal videos from simple distributions is a challenging problem in computer vision that has been recently addressed in various ways [95, 97, 19, 90, 21, 111, 108]. State-of-the-art (SOTA) works [90, 95, 111] treat video generation as a task of generating a sequence of temporally coherent frames. Although such networks have advanced the SOTA to generate high-quality frames (such as carefully crafted eyes, nose, and mouth for talking-head videos), they come with a major limitation: They rely on an image space. This limits the application of the learned space to image-based operations such as animating images and editing on frames. Direct operations on videos, such as interpolating intermediate videos between two videos and generating future segment of a video, become difficult. This is because such operations require learning the set of frame and motion constraints and ensuring that they are coherently learned.

We propose that videos can be represented as a single unit instead of being broken into a sequence of images. One can learn a latent space where each latent point represents a complete video. However, with existing video generator architectures, such representations are difficult. Firstly, such a video generator would be made of several 3D convolution operations. As the dimension and length of the video increase, such an architecture would become drastically computationally expensive (a GPU with limited memory can only fit a video of limited dimension). Secondly, videos are high-dimensional signals spanning both spatial and temporal directions. Representing such a highly expressive signal by a single latent point would require complicated generator architectures and a very high-dimensional latent space. Instead, videos can be parameterized as a function of space and time using implicit neural representations (INRs). Any point in a video V_{hwt} can be represented by a function $f_\theta(h, w, t) \to RGB_{hwt}$ where t denotes the t^{th} frame in the video and h, w denote the spatial location in the frame and RGB denotes the color at the pixel position $\{h, w, t\}$. Here, the dynamic dimension of videos (a few million pixels) is reduced to a constant number of weights θ (a few thousand) required for the parameterization. A network can then be used to learn a prior over videos in this parameterized space. This can be obtained through a meta-network that learns a function to map from a latent space to a reduced parameter space that maps to a video. A complete video is thus represented as a single latent point.

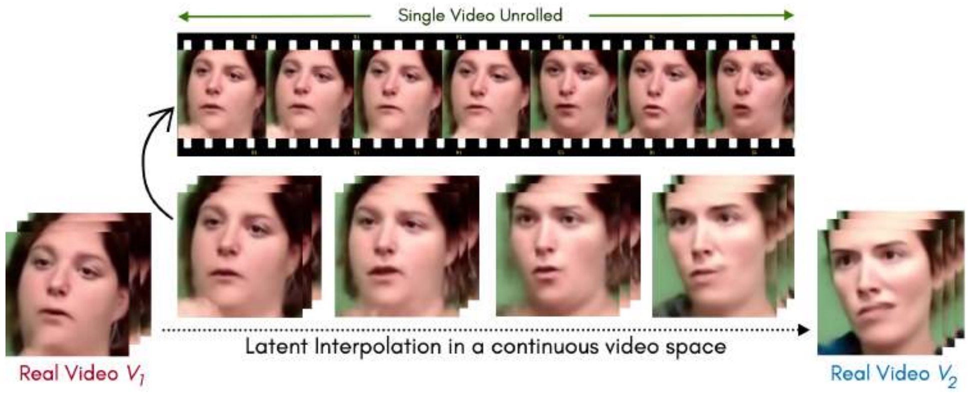


Figure 3.1: Demonstrating the continuity of the video space learned by INR-V by interpolating novel videos between two real videos V_1 and V_2. Note that content (identity, hair) and motion (pose, expressions) gradually transition as we traverse the trajectory in the latent space between V_1 and V_2's latents.

We propose INR-V, a video generator network with a continuous video representation space based on learning an implicit neural representation for videos. It is illustrated in Fig. 3.2. INR-V is made of key elements that, when combined, makes it ideal for video representation: (1) Its INR is free of expensive convolutional layers (millions of parameters) such as in the existing architectures [95, 90] and relies on a few layers of traditional multi-layered perceptrons (MLPs), leading to a very few parameters (a few thousand). (2) Having very few parameters, INR's weights can be populated using a secondary meta-network called hypernetwork [35] that learns a continuous function over the INRs by getting trained on multiple video instances. (3) It is trained on a deterministic distance loss, such as Euclidean or Manhattan distance. This allows INR-V to learn the exact requirements of a coherent video directly from the ground truth video instances.

Hypernetworks have seen wide applications in graphics [85, 86, 13, 88]; however, they have been seldom used for videos. Hypernetworks are notoriously unstable [] to train, especially on the parameterizations of highly expressive signals like videos. Thus, we propose a key prior regularization and a progressive weight initialization scheme to stabilize the hypernetwork training allowing it to scale quickly to more than 30,000 videos. As we show in the experimental section, INR-V demonstrates an expressive and continuous video space by getting trained on these datapoints. The learned prior enables several downstream tasks such as novel video generation, video inversion, future segment prediction, and video inpainting directly at the video level. As shown in Fig. 3.1, INR-V also showcases smooth interpolation of novel videos between two videos by traversing the path between their latent points. Interpolation morphs different identities and motions and generates coherent videos. Interestingly, the properties demonstrated in this chapter are not enforced at training but are natural outcomes of the continuous video space. To summarize, our contributions in this chapter are as follows:

1. We propose considering videos as a single unit and learning a continuous latent space for videos where each latent point represents a complete video.

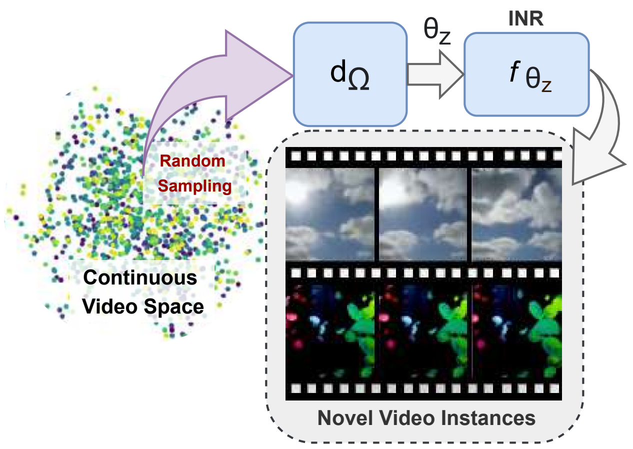


Figure 3.2: **Overview of INR-V:** INR-V learns a continuous video space by first parameterizing videos as implicit neural representations denoted by f_{θ_z}, where z denotes a unique video instance V_z. Next, a meta-network based on hypernetworks denoted by d_Ω is used to learn a continuous representation over the neural representations. d_Ω is conditioned by an underlying continuous video space where each point denotes the condition for a complete video.

2. We propose INR-V, a video representation technique that parameterizes videos using INRs, bringing down the dimension of a video from a dynamic few million to a constant few thousand. INR-V uses a hypernetwork as a meta-network to learn a continuous space over these parameterizations.

3. We demonstrate the benefit of a key regularization and progressive weight initialization scheme to stabilize the hypernetwork training. We scale the hypernetworks to more than 30,000 video points enabling it to learn a continuous meaningful latent space over the INRs.

4. Lastly, we demonstrate key properties of the learned video space, such as video interpolation, video inversion, and so on, by conducting several experiments and evaluations.

3.1 Related Work

Video Generation. Video generation aims to produce novel videos from scratch. It falls under the paradigm of 'video synthesis' that encompasses several categories, including (1) Video prediction [52, 49, 102]: that predicts the next set of frames given the current frames, (2) Frame interpolation [66, 64, 65, 113]: that interpolates frames between given frames of a video. These tasks generate the unseen portion of the video based on the context of the seen portion. On the other hand, video generation produces videos without any expressive prior context, making the task more challenging. The complexity of the problem has led to a plethora of works in this area [95, 97, 90, 19, 21, 111]. VideoGPT [108] tackled this challenge by first reducing the raw videos of up to 128×128 dimension to a quantized space. It then trained a transformer architecture to model a prior over the quantized space.

Our architecture is conceptually similar to VideoGPT, which used a likelihood-based generative model to learn a video prior. However, VideoGPT operates on a quantized space that is discontinuous, making the prior less expressive. INR-V, on the other hand, models a continuous video space. VideoGPT also consists of 3D convolution layers making the model computationally expensive for larger videos. INR-V is a simple MLP, based on a continuous parameterization scheme of INRs, making it agnostic to the video dimension. This allows scaling to multiple resolutions (64×64 or 256×256) at inference without any architectural changes or finetuning. More recent works StyleGAN-V [90], DIGAN [111], and MoCoGAN-HD [95] are a GAN-based setup that model videos as a temporally coherent trajectory over an image space. Use of a continuous representation space for videos has been considered before in [26, 8] for the task of action classification. However, in this chapter, we focus on learning a representation space for generative tasks.

Hypernetworks. Hypernetworks [35] were introduced as a metafunction that initializes the weights for a different network called the primary network. Hypernetworks have been widely used for several purposes, starting from representation learning for continuous signals [67, 86, 84, 85, 57, 88], compression [63, 29], few-shot learning [81, 48], continual learning [101], language modeling [92]. We use hypernetworks to populate our primary video generation network, an MLP parameterizing different video instances.

Implicit Neural Representations. In this paradigm, a continuous signal is represented by a neural network. INRs have had wide adaptations in 3D Computer Vision [67, 30, 87, 57, 86, 58] and Computer Graphics [33, 110]. Recently, INR was adopted for images [89] and videos [11, 85, 111, 12]. INR-GAN [89] first showed the application of INRs in generating high-quality images by replacing the generator component of StyleGAN2 [43] with an MLP-based INR. It then used a hypernetwork to populate the INR. Unlike INR-GAN, which is trained using a stochastic discriminator, INR-V relies on a deterministic distance-based loss to train the hypernetwork. SIREN [85] proposed periodic activation functions for INRs as a replacement for ReLU activation to parameterize many different data types like images, videos, sounds, and 3D shapes, with fine details. NeRV [11] designed an implicit function as a continuous function of time and used convolution blocks at each time step to parameterize discrete frames showcasing an improved frame quality over SIREN. Recently, VideoINR [12] was proposed that used INRs for video superresolution. DIGAN [111] incorporated INRs made of MLP layers for video generation. It consisted of two separate networks that generated spatial and temporal codes for generating videos in a frame-wise fashion. StyleGAN-V [90] also incorporated INRs and relied on continuous non-periodic positional encodings for each timestep of a video. Like NeRV, StyleGAN-V used traditional convolution operations for frame-by-frame video generation. Both DIGAN and StyleGAN-V used a GAN setup to train the video generators. INR-V is based on MLPs with ReLU activation trained in a fashion similar to Light Field Networks (LFNs) [86]. LFNs proposed a novel neural scene representation for novel view synthesis and trained a hypernetwork over multiple object instances using distance-based losses like Euclidean or Manhattan distance. Like LFNs and INR-GAN, INR-V parameterizes the entire signal (a video) using INRs and relies on a single hypernetwork to generate the INRs.

However, unlike LFNs and INR-GAN, INR-V encodes a denser representation of a volumetric 3D signal $\in \mathbb{R}^3$ data making hypernetwork training more challenging.

3.2 INR-V: Implicit Neural Representation for Video Synthesis

Each video instance V_n consists of pixels at locations (h, w) at t^{th} frame. We have a particular parameter vector θ_n that is used by a network f to generate the value of the color RGB_{hwt} for that pixel location (h, w, t). We need to learn a network d with parameters Ω that predicts the parameters θ_n for a particular video V_n. Here, d is a hyper-network. The overall approach to train the network is illustrated in Fig. 3.3.

3.2.1 Hypernetwork for Modeling Multiple Video Instances

As f_θ implicitly stores a single video signal, any new video would need its own implicit function. Let f_{θ_n} denote the implicit function for a given video $\{V_n\}_{n=1}^{N}$ where N is the total number of available videos in the training dataset $\mathcal{D}$. Each of these implicit functions, f_{θ_n} can be modeled using a neural network trained on each pixel value of the video V_n. Thus, implicit functions minimize the following objective:

$$L(\theta_n) = \frac{1}{T}\frac{1}{W}\frac{1}{H}\sum_{t=1}^{T}\sum_{w=1}^{W}\sum_{h=1}^{H}(f_{\theta_n}(h, w, t) - RGB_{hwt})^2 \tag{3.1}$$

Generating a novel video V_z translates to generating a novel implicit function f_{θ_z} that represents the video meaningfully. Let us consider f_{θ_z}, an unseen sample from an underlying distribution Φ. Each point in the distribution Φ denotes an implicit function of a meaningful video. To randomly sample f_{θ_z}, we make use of a meta-network to learn the distribution Φ.

We use a hypernetwork d_Ω as a meta-network to parameterize f_θ, such that $d_\Omega(m_n) = \theta_n$ for video instance V_n. Here m_n is a a d-dimensional point in the latent space, say τ, and serves as an instance code for V_n. Given enough number of samples N, d_Ω learns to map the latent codes sampled from τ to their corresponding parameterized space Φ, as shown in Fig. 3.3. The parameters θ_n are then used to initialize f to generate V_n.

Let us consider τ as a meta-distribution such that $\{m\}_{\mathcal{D}} \in \tau$. At the time of inference, m_z can be sampled from τ. As d_Ω has learned a valid representation over Φ, m_z enables d_Ω to generate a meaningful implicit function $f_{\theta_z} \in \Phi$. Sampling from τ can be made straight forward by making sure τ is regularized during training. At the time of training, Ω and $\{m_n\}_{n=1}^{N}$ are optimized together. θ is a non-learnable parameter and f is initialized as the output of d_Ω. The following objective is optimized:

$$L(\Omega, m) = \frac{1}{N}\frac{1}{T}\frac{1}{W}\frac{1}{H}\sum_{n=1}^{N}\left(\sum_{t=1}^{T}\sum_{w=1}^{W}\sum_{h=1}^{H}(f_{\theta_n}(h, w, t) - RGB_{ijk})^2\right) \quad \text{and} \quad \theta_n = h_\Omega(m_n) \tag{3.2}$$

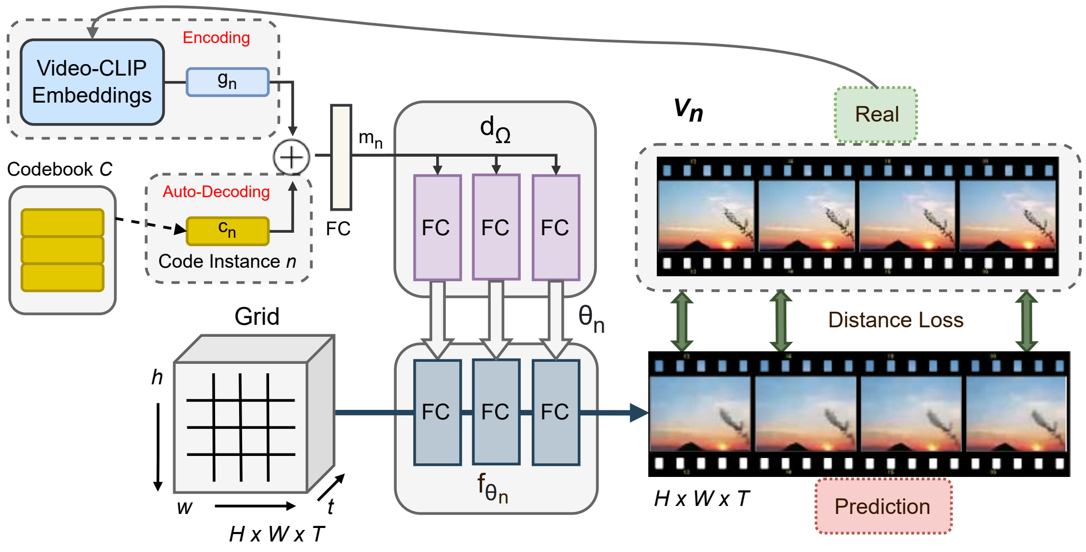


Figure 3.3: **Architecture of INR-V:** Any video instance V_n is represented by its corresponding implicit neural representation, an MLP, f_{θ_n}. f_{θ_n} takes a grid as input denoting the pixel positions of the video encoded using periodic positional encodings. It then generates the pixel values for all the positions. f_{θ_n} is initialized by a meta-network called hypernetworks denoted as d_Ω composed of a set of MLPs. d_Ω is conditioned by an instance code m_n unique to every video instance V_n. m_n is trained by combining (1) auto-decoding framework to regress to a code c_n and (2) encoding-framework to regularize the space using CLIP embedding that generates V_n's semantic code g_n. At the time of inference, m_n is randomly sampled from an underlying learned distribution τ.

3.2.2 Regularizing τ for Hypernetwork Conditioning

To generate a novel video, a random latent m_z is sampled from the latent space τ. d_Ω is then conditioned on m_z generating an implicit function $f_{\theta_z} \in \Phi$. In a standard hypernetwork training [88, 86, 67, 85], m_n is optimized in an auto-decoding framework as given in Eqn. 3.2. However, given the complexity of the signal V (a 3D volumetric representation) that d_Ω has to model, $\{m\}_\mathcal{D}$ can collapse to a single point if τ is not regularized at the time of training, bringing the expressiveness of d_Ω down to a single implicit function. We regularize τ by leveraging pretrained CLIP [71] designed for generating semantically meaningful embeddings for images. We design Video-CLIP that encodes an entire video V_n to a vector g_n. As shown in Fig. 3.4, Video-CLIP first generates the image-level CLIP embeddings. These embeddings are then passed through a bi-directional GRU. The mean of the hidden state outputs of the final layer produces g_n. As shown in Fig. 3.3, the regularized instance code m_n is now:

$$m_n = \phi(c_n, g_n) \tag{3.3}$$

where c_n is the instance code of V_n optimized in an auto-decoding fashion at the time of training, and ϕ is a neural network. The pretrained CLIP embeddings are kept frozen during training and the learnable parameters are the instance codes c_n that are regularized by g_n. CLIP regularization encourages the latent codes to be spaced sufficiently apart by leveraging predefined semantic encoding. This helps

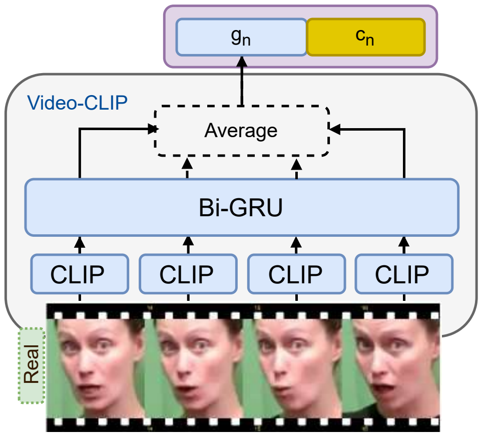


Figure 3.4: **Video-CLIP**: Encoding a video V_n to a latent vector g_n by using image-level CLIP encodings.

avoid mode collapse during the initial stages of training. Please find the ablation on CLIP regularization in Sec.3.5.1.

3.2.3 Progressive Training

A video is a dense 3D volume mandating its neural representation to model every single point in the volume. Although implicit representations have a constant number of parameters made of only a few layers of MLPs in our case, learning a meta-function using a hypernetwork over such dense representations is challenging. As a result, if not appropriately initialized, the hypernetworks can easily collapse to a single representation despite CLIP regularization. Moreover, a sub-optimal hypernetwork initialization could result in a significantly longer convergence period. To tackle this challenge, we adopt a progressive initialization scheme. Firstly, the training is divided into multiple stages. Each stage, denoted by $\{l\}_{l=1}^{\mathcal{K}}$ where $\mathcal{K}$ is the total number of stages, is made of a subset of the training dataset $\mathcal{D}$. The number of samples N_l in each stage l is given as:

$$N_l = \begin{cases} N_{l-1} + \epsilon_l & l > 1 \\ \mathcal{C} & l = 1 \end{cases} \tag{3.4}$$

where $\mathcal{C}$ is a constant and ϵ_l denotes the number of additional samples for l^{th} stage. Each step l consists of $\{V_n\}_{n=1}^{N_l}$ datapoints that is computed as:

$$\{V_n\}_{n=1}^{N_l} = \{V_i\}_{i=1}^{N_{l-1}} + \{V_j\}_{j=N_{l-1}}^{N_{l-1}+\epsilon_l} \tag{3.5}$$

where the order of set $\{V\}$ is maintained across the training stages. At the start of the training, the model is trained on $\mathcal{C} < 10$ examples. This allows the hypernetwork to quickly adapt to the handful of examples and initialize the weights. However, jumping from $\mathcal{C}$ to $\sim$ 30,000 samples causes the network to collapse again. Thus, we adapt the network progressively to the given examples. Each stage of the progressive training is a full training of the model, with the weights in the current stage initialized with the weights learned from the previous stage. This includes reusing the instance codes c_n learned at a previous stage $l-1$ in the current stage l. This step is crucial, as without this, the hypernetwork is pushed to re-learn all the instance codes. The new instance codes added in the current stage are initialized from a Gaussian distribution.

3.3 Experiments

Experimental Setup: We perform our experiments on (1) How2Sign-Faces [23], (2) SkyTimelapse [106], (3) Moving-MNIST [91], and (4) RainbowJelly [90]. Real video samples of each dataset are visualized in Fig. 3.19. How2Sign [23] is a full-body sign-language dataset consisting of 11 signers. The signers have elaborate facial expressions, mouth, and head movements. We modify How2Sign to How2Sign-Faces by cropping the face region out of all the videos and randomly sample 10,000 talking head videos, each of at least 25 frames, of dimension 128×128. SkyTimelapse [106] consists of scenic videos of sky changes. It is made of 1803 videos, each at least 25 frames. The videos are first center-cropped to 360×360 from an original dimension of 360×620 and then resized to 128×128 for training. Moving-MNIST [91] is a video dataset of moving MNIST digits containing a total of 10,000 datapoints. Each video is 20 frames long. RainbowJelly is a single underwater video capturing colorful jellyfishes. The video is first extracted into frames which are then divided into videos of 25 frames each, making a total of 34,526 videos. Similar to SkyTimelapse, the videos are first center cropped to 360×360 and then resized to 128×128.

All experiments are performed on 2 NVIDIA-GTX 2080-ti GPUs with 12 GB memory each. All models, except INR-V, are trained at a resolution of 128×128. To make training computationally efficient, INR-V is trained on a lower resolution of 100×100 videos. Based on INRs, INR-V can infer directly at multiple resolutions (please refer Sec.3.4.2). For evaluations and comparisons, INR-V is inferred at 128×128 like the other models.

Implementation Details The implicit neural representation f_θ is an MLP with three 256-dimensional hidden layers. Each hidden layer is passed through ReLU activations. The hypernetwork d_Ω is a set of MLPs. Each MLP predicts the weights for a single hidden layer and the output layer of f_θ. Each MLP has three 256-dimensional hidden layers. CLIP embeddings are 512-dimensional vectors, Video-CLIP encodes the CLIP embeddings of each frame through three 512 dimensional, GRU layers. As shown in Fig. 3.4, Video-CLIP produces 512-dimensional video-level embedding g_n. c_n is a 512-dimensional context vector that is regressed in an auto-decoding fashion during training. ϕ is made of 3-hidden layers that takes a 1024-dimensional vector as input (concatenation of g_n and c_n) and produces m_n, a 128-dimensional instance code of V_n, as the input for d_Ω. The input to f_θ is a periodic positional encoding of $(\{h\}_{h=1}^{H}, \{w\}_{w=1}^{W}, \{t\}_{t=1}^{T})$ as implemented in [86]. Adam optimizer is used with a learning rate of $1e-4$ during training and $1e-2$ during inversion tasks. No scheduler is used. Progressive training is done at a power of 10 where i^{th} stage is made of min$(10^i, N)$ examples. $i = 0 \ldots \mathcal{K}$ such

	Single-INR			INR-V				
Dataset	$\mathcal{E}\downarrow$	PSNR$_{50}\uparrow$	SSIM$_{50}\uparrow$	$\mathcal{E}_{50}\downarrow$	PSNR$_{50}\uparrow$	SSIM$_{50}\uparrow$	PSNR$_{\text{FULL}}\uparrow$	SSIM$_{\text{FULL}}\uparrow$
How2Sign-Faces	4.83	29.72	0.925	8.29	25.69	0.850	25.84	0.869
SkyTimelapse	4.69	36.19	0.943	5.87	33.69	0.931	33.94	0.924
Moving-MNIST	3.57	37.26	0.978	6.06	29.81	0.949	29.54	0.975
RainbowJelly	4.17	35.93	0.918	5.02	33.34	0.937	33.57	0.938

Table 3.1: Quantitative metrics on reconstruction quality. Comparison set is made of 50 videos per training dataset. INRs trained individually for each video is denoted as Single-INR. INR-V trains a single hypernetwork d_Ω to populate the INRs of all the videos in the training dataset. PSNR$_{50}$ and SSIM$_{50}$ are computed on the comparison set, PSNR$_{\text{FULL}}$ and SSIM$_{\text{FULL}}$ are computed on the entire training set. $\mathcal{E}$ is computed on videos with pixel range $[0, 255]$. INR-V performs comparably with Single-INR despite getting trained on huge datasets of more than 30,000 videos.

Method	How2Sign-Faces	SkyTimelapse	Moving-MNIST	RainbowJelly
MoCoGAN-HD	396.53	321.44	296.95	1856.21
DIGAN	165.89	135.60	144.97	408.19
StyleGAN-V	94.64	**85.05**	109.85	1227.70
INR-V	161.68	153.42	103.24	**260.72**
+ Denoising	**87.22**	-	**47.28**	-

Table 3.2: FVD$_{16}$ metrics computed on random videos generated by the respective models.

that $10^{\mathcal{K}+1} < N + 1$, where N is the total number of training samples. Each stage except the last stage is trained until the reconstruction error reaches a threshold of $1e - 3$.

3.3.1 Comparing INR-V with Single-INR

INR-V uses hypernetworks to learn a distribution over the INRs of videos. A single hypernetwork d_Ω can initialize the INRs for multiple videos $\{V_n\}$ based on their respective instance codes m_n. Thus, measuring if d_Ω generates the INR functions f_θ accurately is crucial. We evaluate this using a set of 50 randomly sampled videos from the training dataset. Each video is first trained to fit a single INR function f_{θ_n} using Eqn. 3.1 denoted as Single-INR. Next, the INRs of these 50 videos are populated using a pretrained hypernetwork d_Ω trained on the entire dataset. We measure the reconstruction quality with PSNR (Peak Signal to Noise Ratio), SSIM (Structural SIMilarity), and the error as:

$$\mathcal{E} = \left(\frac{1}{50} \sum_{n=1}^{50} \frac{1}{HWT} (V_n' - V_n)^2 \right)^{\frac{1}{2}} \tag{3.6}$$

where V_n' denotes the video generated using the implicit function f_θ. Single-INR was optimized for 750 steps using Eqn. 3.1 taking ~ 5.56 minutes for each video (~ 4.63 hours for 50 videos). Table. 3.1 presents quantitative metrics on the videos reconstructed using Single-INR and INR-V. PSNR$_{\text{FULL}}$ computes the PSNR on the entire training dataset, PSNR$_{50}$ computes the metric on the selected 50 videos for comparison. As can be seen, although hypernetwork d_{Ω} is trained on huge datasets, it performs comparably with Single-INR. For RainbowJelly, it even outperforms Single-INR in SSIM metric and performs at par on SkyTimelapse. This indicates that d_{Ω} has learned to accurately generate INRs for complex spatio-temporal signals. Thus, INR-V can be used as a compression technique to compress 1000s of videos with minimal loss in perceptual quality.

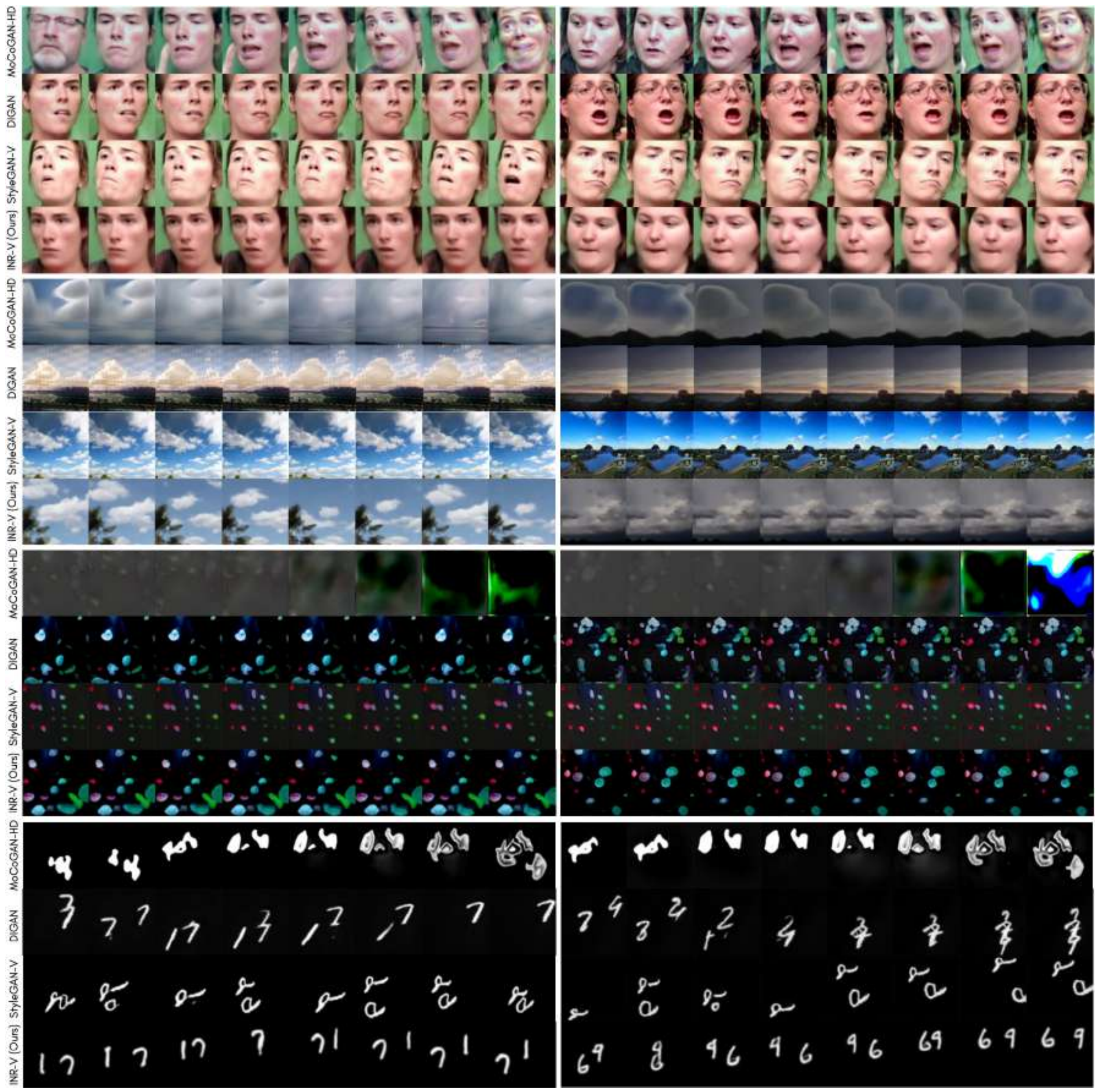


Figure 3.5: Examples of random videos generated on, from top to bottom, How2Sign-Faces [23], Sky-Timelapse [106], RainbowJelly, and Moving-MNIST [91]. For Moving-MNIST, every 2$^{\text{nd}}$ frame of 20 frames long videos and for other datasets, every 3$^{\text{rd}}$ frame of 25 frames long generated are shown. Moving-MNIST and How2Sign-Faces are passed through VQVAE2 denoising network as described in Sec.3.3.2

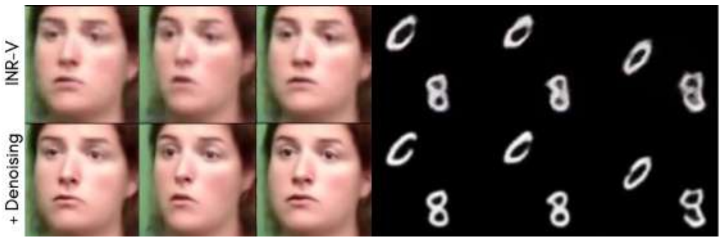


Figure 3.6: Denoising VQVAE2 reconstructions to enhance the visual quality of relatively blurry videos generated by INR-V.

3.3.2 Comparing INR-V with SOTA video generation networks

Overview: Fig. 3.5 and Table 3.2 present qualitative and quantitative comparisons respectively between MocoGAN-HD [95], DIGAN [111], StyleGAN-V [90], and INR-V. All models were trained from scratch. As we train the models on smaller datasets of $\sim$ 10,000 datapoints, MoCoGAN-HD is trained on StyleGAN2-ADA [41] image-generator backend. For each model, the best-performing checkpoint is selected for comparison.

Evaluation: As can be seen in Fig. 3.5, INR-V generates novel videos with coherent content and motion. MoCoGAN-HD fails to maintain the identity in a single video instance. For quantitative evaluation, we use the Frechet Video Distance (FVD) metric as implemented by StyleGAN-V. FVD_{16} is computed on 2048 videos of 16 frames sampled at a resolution of 128×128. As can be seen in Table 3.2, INR-V outperforms the existing networks on Moving-MNIST and RainbowJelly and performs comparably on the remaining datasets.

Enhancing INR-V's Visual Quality Enhancing image and video quality has been an area of extensive research [109, 14, 51, 10] with many breakthroughs. We propose that video generation can be partitioned into two stages (1) generating coherent content and motion (2) enhancing the visual quality. Note that, in the current work, our effort has been (1) to propose a novel continuous representation space for videos. We demonstrate (2) by developing a simple denoising network using a standard VQ-VAE2 [73]. We train VQVAE2 as a frame-by-frame denoising autoencoder making one minor change: Instead of reconstructing the given low-quality input, we use the high-quality frame for computing the error. The low-quality inputs are the intermediate video instances reconstructed by INR-V during training. We train denoising VQVAE2 on How2Sign-Faces and Moving-MNIST. Fig. 3.6 demonstrates the results of the denoising network on blurry instances generated by INR-V. As can be seen from the quantitative metrics in Table. 3.2, using an additional denoising network improves the network's performance by $\sim 2\times$.

3.4 Applications of the continuous video space learned by INR-V

INR-V learns a continuous latent representation for videos allowing complex spatio-temporal video signals to be represented using a single latent point. In this section, we showcase the advantage of such

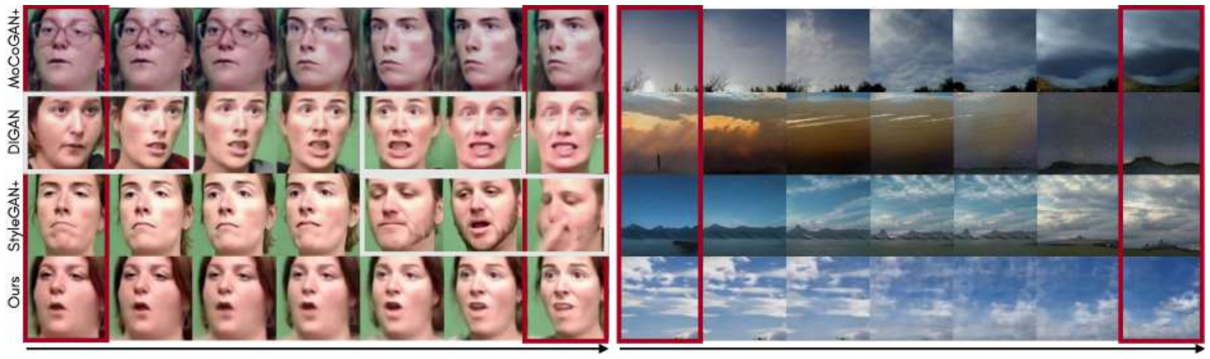


Figure 3.7: **Each cell displays 12th frame of 25 frames long generated videos**. The videos demonstrate interpolation between two given videos (in red boxes) by traversing along a trajectory in the latent space connecting the latent points of the given videos. Here, MoCoGAN+ and StyleGAN+ denote MoCoGAN-HD and StyleGAN-V. White boxes indicate a sudden transition in content (e.g. identity) or motion (e.g. pose).

MoCoGAN-HD	DIGAN	StyleGAN-V
100.00	89.43	95.24

Table 3.3: Video interpolation user study: % of times INR-V interpolation was preferred over existing models.

a latent space through several demonstrated properties and comparisons. We also benchmark several tasks based on the inversion of 256 videos on How2Sign-Faces using full and incomplete video context.

3.4.1 Video Interpolation

Given two videos V_1 and V_2, a continuous video space should be able to make a gradual transition between the two videos such that every point along the trajectory between the two (1) produces a meaningful video and (2) shares content and motion properties from V_1 and V_2. We demonstrate this property in Fig. 3.1 and Fig. 3.7 with Spherical Linear Interpolation (Slerp)[2]. Each cell in Fig. 3.7 demonstrate the 12th frame of the 25 frames long videos. As can be seen, INR-V observed a gradual change in motion (pose, mouth movements, expressions, cloud shift) and content (identity, visibility of sun). The interpolated videos are spatio-temporally coherent. Fig. 3.23 and Fig. 3.24 demonstrate the spatio-temporal transition on How2Sign-Faces and SkyTimelapse. As we represent an entire video in a single point in the continuous video space, interpolation is a natural operation that can be performed with INR-V.

Comparisons: Existing models have different motion and content codes; thus, to interpolate videos, intermediate content codes were interpolated between two videos by Slerp interpolation. INR-V does not have separate motion and content vectors; thus, videos can be interpolated directly using given

[2] https://splines.readthedocs.io/en/latest/rotation/slerp.html

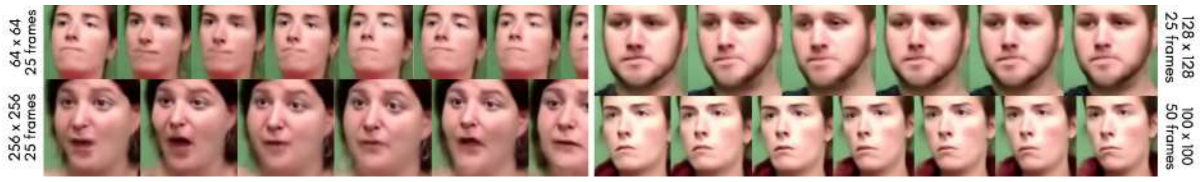


Figure 3.8: **INR-V direct inference on multiple resolutions and frame length**. INR-V trained on only 25 frames long 100×100 videos. Novel videos of multiple resolutions (64×64, 128×128, 256×256) and video length (50) are directly generated on the trained model without any architectural change or finetuning. The images are not upto scale, please refer Appendix Fig. 3.25 for scaled representation.

video's latent points. As shown in Fig. 3.7, INR-V has a gradual transition in motion and content. For How2Sign-Faces, StyleGAN-V abruptly changes motion (cell 5-7), and DIGAN abruptly switches identity (cell 1-2, cell 5-6). This effect is highlighted in white boxes. This is expected as both of these architectures operate in the image space, and thus a gradual spatio-temporal transition is harder to achieve. We performed a user study on 30 users to qualitatively evaluate the interpolation quality of INR-V against the SOTA models and report the metrics in Table. 3.3. INR-V interpolation was randomly shown against either of the other three models. The users provided their preference on which interpolation looked smoother in terms of transition in content and motion. INR-V was preferred at least 85% more than all the SOTA networks. This demonstrates the continuous nature of the video space learned by INR-V.

3.4.2 Multi-Resolution and Multi-Length Inference

In Fig. 3.8 we show INR-V trained on videos of only 100×100 resolution with 25 frames per video, generating novel videos of multiple resolutions and lengths, maintaining the content and motion quality of the output. An underlying property of INRs is a continuous representation of the signal given as $f_\theta(h, w, t) \rightarrow$ RGB. This enables the model to understand a continuous property of the signal making it agnostic to the dimension.

Quantitative Metrics An underlying property of INRs is a continuous representation of the signal. This allows inferring INR-V on multiple spatial and temporal resolutions directly without changing the model's architecture or additional finetuning. To generate a video of an arbitrary dimension $\hat{H} \times \hat{W} \times \hat{T}$, those many number of equally spaced points are sampled between $[-1, 1]$. In Table 3.4, we report FVD$_{16}$ scores on random videos generated on INR-V with varying spatial dimensions on 25 frames. To infer at multiple resolutions, INR-V pretrained on 100×100 dimensional videos of 25 frames was used. Note that the FVD scores do not degrade even at higher spatial resolutions. Additionally, we compare INR-V with existing SOTA superresolution techniques [12] in Table 3.5 on 2048 videos randomly sampled from the RainbowJelly dataset. As can be seen, INR-V performs comparably with methods on the task of superresolution. Moreover, INR-V can be superresolved to any arbitrary resolution ($3.6\times$) and aspect ratio. Please note that we do not solve the task of superresolution but rather show superresolution as a potential application of our work.

	128×128	256×256	360×360
INR-V (Ours)	260.72	**232.43**	251.14

Table 3.4: FVD$_{16}$ ($\downarrow$) metrics on random video generation at multi-resolution on INR-V. Training was done on only 100×100 dimensional videos of 25 frames. Inference was taken directly on multiple resolutions without any finetuning or architectural changes.

	$2\times$		$3\times$		$3.6\times$	
	200×200		300×300		360×360	
	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$
Bicubic	31.53	0.884	32.13	**0.915**	**32.31**	**0.920**
VideoINR	**31.59**	0.883	**33.01**	0.913	-	-
INR-V	28.62	**0.892**	29.17	0.894	29.05	0.896

Table 3.5: Quantitative metrics on video superresolution using INR-V and SOTA superresolution networks on video instance seen at the time of training. INR-V was trained at 100×100 video resolution. INR-V performs comparably with the SOTA superresolution networks.

3.4.3 Video Inversion and its applications

Inversion has been widely adopted in many applications prominently for images. StyleGAN2 [43] is extensively used for image inversion enabling many downstream image editing tasks such as changing the emotion, age, or gender of a given face. In video inversion, we aim to invert a given video back into the latent space of a pretrained video generation network. Existing methods perform frame-by-frame inversion to individually invert the context code for each frame and the motion code for the video. In INR-V, we only need to invert to a single latent code that can be achieved through a simple optimization objective:

$$\underset{m_z}{\text{argmin}} \frac{1}{T}\frac{1}{W}\frac{1}{H}\sum_{T}^{t=1}\sum_{W}^{w=1}\sum_{H}^{h=1}(f_{\theta_z}(h,w,t) - RGB_s)^2 \quad \text{where} \quad \theta_z = h_\Omega(m_z) \tag{3.7}$$

where m_z is the latent point for a video instance V_z. Fig 3.9 shows the qualitative demonstration of INR-V inversion trained on How2Sign-Faces for a video outside of the training dataset $\mathcal{D}$.

3.4.3.1 Video Completion:

Key categories of 'video synthesis' include future frames prediction (future prediction), completing the video between frames (frame interpolation), and predicting the missing part of the video (video inpainting). In INR-V, a video V_z, represented by a single latent code m_z can be generated without any additional knowledge. Thus, all the above operations can be performed using a modified optimization

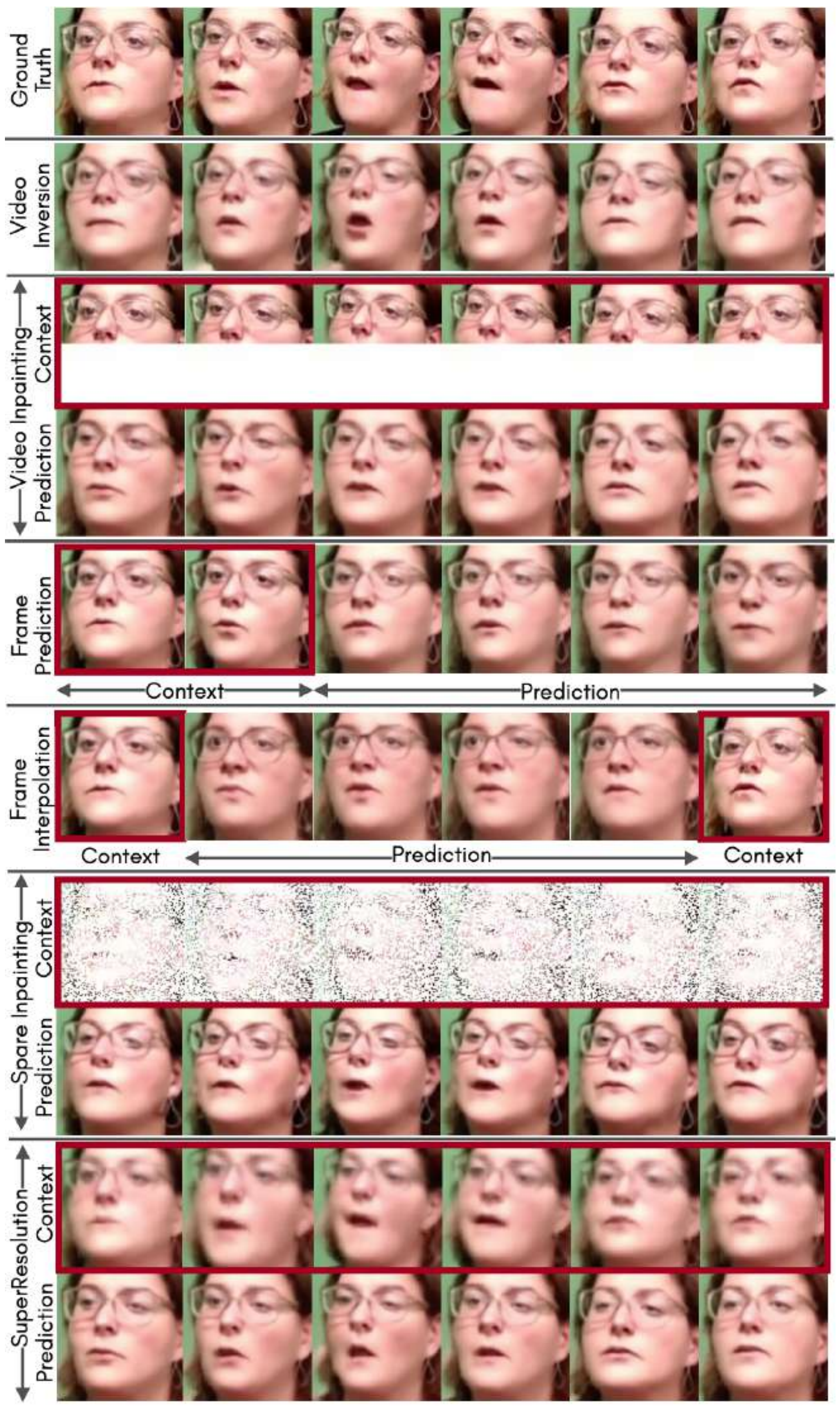


Figure 3.9: **Video Inversion and it's applications.** INR-V can be directly used for several tasks by simply inverting a video to its latent point based on the given context. We demonstrate some qualitative results.

operation based on Eqn 3.7 on the seen part of the video given as:

$$\begin{aligned} &\underset{m_z}{\operatorname{argmin}} \frac{1}{S}(f_{\theta_z}(h_s, w_s, t_s) - RGB_s)^2 \\ &\text{and} \quad \theta_z = h_\Omega(m_z) \end{aligned} \tag{3.8}$$

where S is the number of context points, h_s, w_s, and t_s are the context points of V_z seen at the time of optimization. With the optimized m_z, the full video can simply be generated back with INR-V. Fig. 3.9 demonstrates the results for the various operation on a video outside of $\mathcal{D}$ with ~ 2.5 minutes of optimization on a single 12 GB NVIDIA GTX 2080ti GPU. As can be seen, the network is able to regress to a latent corresponding to the given identity while preserving finer details like spectacles, mouth shape, pose, etc. In the case of 'Video Inpainting', the network understands the person's pose. For 'Frame Prediction', although the pose does not match the ground truth, the overall video is coherent. In 'Frame Interpolation', the model is able to generate a coherent context between two frames, including the pose, expressions, identity, and mouth movements. In 'Sparse Inpainting', we randomly set 25% of all the video pixels as the context points for optimization. Even with very sparse context, INR-V is able to regress to the correct specifications including the finer content details.

3.4.3.2 Video Superresolution through inversion:

Video Superresolution is the task of enhancing the resolution of a given video. Recent works such as [14, 51, 78, 10, 104, 12] have made significant progress in video superresolution, showcasing $4\times$ enhancement. INR-V can directly superresolve seen video instances as showcased in Table. 3.5. For unseen instances, combining the capability of video inversion and multi-resolution video generation, INR-V can superresolve a given video V_z of a lower resolution (say 32×32) simply as following: (1) Invert V_z at the smaller resolution to obtain m_z. (2) Render V_z from m_z directly at a higher resolution (say 256×256). In Fig. 3.9, we demonstrate the qualitative results on a video outside the training dataset. The video was optimized at 32×32 for ~ 2.5 minutes. The inverted video was then superresolved at a scale factor of $8\times$ to 256×256. Additional details are present in Sec.3.4.2.

3.4.3.3 Quantitative Evaluation:

To quantify the performance of INR-V, we prepare a comparison set by randomly sampling 256 videos outside of the training set. We compare against DIGAN on the tasks of Video Inversion, Video Inpainting, Frame Prediction, Frame Interpolation, and Sparse Interpolation and against StyleGAN-V on the task of Video Inversion. Since DIGAN is based on INRs, it can invert incomplete frames, however, StyleGAN-V expects a full frame for backpropagation. Thus we do not compare with StyleGAN-V on the other tasks. For the task of Superresolution, we compare against Bicubic Upsampling and VideoINR at a scale factor of $4\times$ from 32×32 to 128×128.

We evaluate on the following metrics: (1) PSNR, (2) SSIM, (3) Temporally Locally (TL-ID) and Temporally Globally (TG-ID) Identity Preservation, (4) Context-L1, and (5) Ground Truth Identity

Task	Method	GT-ID ↑	TL-ID ↑	TG-ID ↑	Context-L1 ↓	PSNR ↑	SSIM ↑	Cost ↓
Inv.	DIGAN	0.652	0.953	0.9599	45.08	19.59	0.653	$\sim$ 4.25
	Style-V	**0.804**	**0.985**	**0.998**	42.16	19.65	0.665	$\sim$ 3.25
	INR-V	0.770	0.950	0.950	**5.25**	**21.21**	**0.773**	$\sim$ **2.75**
Inp.	DIGAN	0.628	**0.960**	**0.969**	45.80	-	-	$\sim$ 4.25
	INR-V	**0.758**	0.948	0.939	**4.83**	-	-	$\sim$ **2.75**
Pre.	DIGAN	0.603	0.940	0.928	40.26	-	-	$\sim$ 4.25
	INR-V	**0.703**	**0.946**	**0.932**	**4.72**	-	-	$\sim$ **2.75**
Int.	DIGAN	0.653	0.925	**0.921**	48.66	-	-	$\sim$ 4.25
	INR-V	**0.702**	**0.928**	0.905	**7.46**	-	-	$\sim$ **2.75**
Spr.	DIGAN	0.718	0.961	0.967	46.24	19.74	0.671	$\sim$ 4.25
	INR-V	**0.768**	**0.968**	**0.974**	**5.29**	**22.35**	**0.774**	$\sim$ **2.75**
Sup. $4\times$	Bicubic	0.808	0.923	0.903	-	28.36	0.906	-
	VideoINR	**0.939**	**0.982**	**0.974**	-	**32.86**	**0.957**	-
	INR-V	0.734	0.911	0.903	4.92	21.94	0.742	$\sim$ **2.75**

Table 3.6: Comparison of INR-V on various video inversion tasks: Video Inversion (Inv.), Video Inpainting (Inp.), Frame Prediction (Pre.), Frame Interpolation (Int.), Sparse Interpolation (Spr.), and Superresolution (Sup.). Comparison set is made of 256 videos outside of the training dataset. Metrics used for evaluation is explained in Sec. 3.4.3.3. Cost denotes the time to optimize a single video instance in minutes.

(GT-ID) Match. TL-ID and TG-ID were proposed in [99]. They evaluate a video's identity consistency at a local and global level. For both metrics, a score of 1 would indicate that the method successfully maintains the identity consistency of the original video. Context-L1 computes the L1 error on the inverted videos at the given context points. An error of 0 would indicate that the inversion is perfect. GT-ID measures the match in identity between the ground truth and the inverted video. DeepFace[3] face features are extracted for both the videos, and the cosine similarity is computed between the extracted features. Since there is no single correct prediction for tasks like 'Future Frame Prediction', 'Frame Interpolation', and 'Video Inpainting', we do not evaluate these tasks on PSNR and SSIM.

As can be seen, INR-V outperforms all the existing networks in most of the metrics on video inversion and the proposed inversion tasks, except 'Superresolution', indicating the advantage and robustness of the proposed space. For the task of Superresolution, INR-V performs comparably with Bicubic and VideoINR. However, unlike these works that directly superresolve a video, INR-V first inverts the low resolution video to generate a high resolution video. Such a mechanism opens several possibilities, such as inverting a low resolution incomplete video (missing frames due to corruption) to a high resolution video with full context.

[3] https://github.com/serengil/deepface

3.5 Ablation

3.5.1 Effect of Regularization

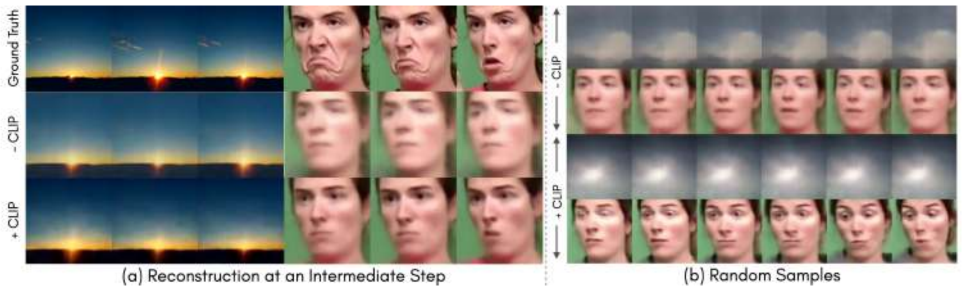


Figure 3.10: Qualitative results of CLIP regularization. Intermediate results are shown after 30 hours of training on 2 NVIDIA GTX 2080ti GPUs. (a) Video reconstruction quality. CLIP regularization enables the meta-network to model the INRs with finer details. (b) Videos generated by random sampling. CLIP regularization improves the quality of the sampled videos and encourages variation in the implicit representations.

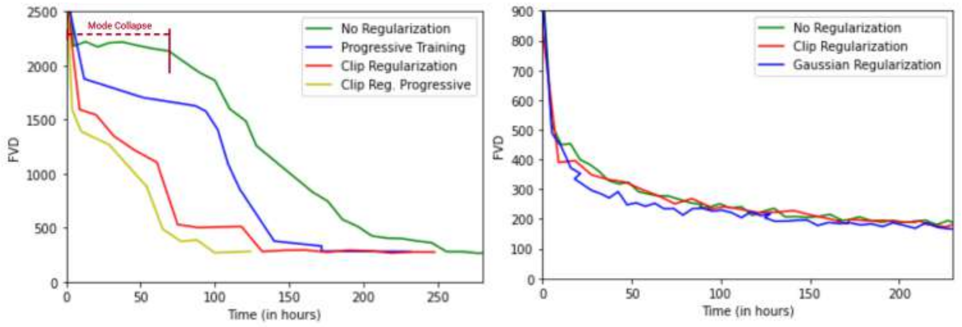


Figure 3.11: Convergence rate for different regularization schemes on RainbowJelly (left) and SkyTimelapse (right) datasets. RainbowJelly consists of $\sim$ 34K datapoints and SkyTimelapse consists of $\sim$ 2K datapoints. As can be observed, SkyTimelapse being a relatively smaller dataset performs equally well on all the regularization schemes. RainbowJelly performs worst without any regularization facing mode-collapse for about the first $\sim$ 75 hours. Progressive training and CLIP regularization help INR-V converge the fastest.

In this section, we compare the training time and the performance of INR-V (1) with/without CLIP regularization and (2) with/without progressive training. Fig. 3.10 presents the qualitative results on INR-V after 30 hours of training on 2 NVIDIA GTX 2080ti GPUs. Fig. 3.11 plots the rate of convergence on RainbowJelly (left) and SkyTimelapse (right) datasets on the same training setup. We also show Gaussian regularization with INR-V by adding the following additional loss term to the overall loss term in Eqn. 3.2:

$$\delta D_{KL}(\ \mathcal{N}(\mu,\sigma)\ ||\ \mathcal{N}(0,1)\ ) \tag{3.9}$$

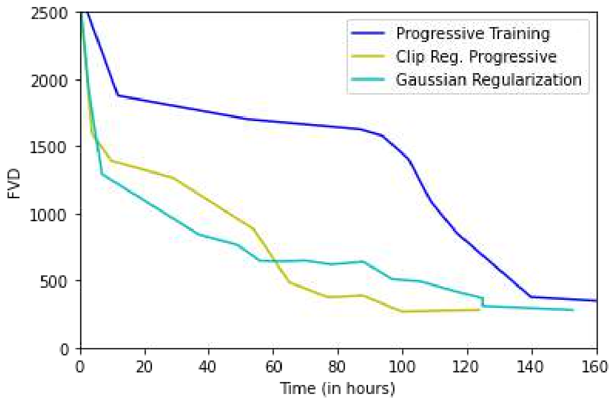


Figure 3.12: Convergence rate for different regularization schemes on RainbowJelly dataset made of $\sim$ 34K instances. All the models are trained progressively. CLIP Regularization leads to the fastest convergence.

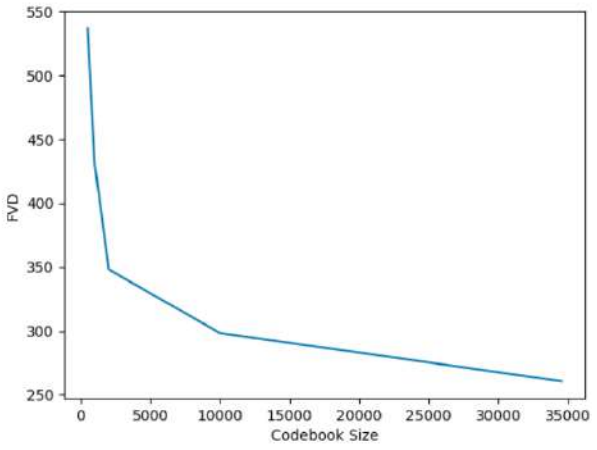


Figure 3.13: Performance of INR-V when trained on varying number of video instances (codebook size). FVD is computed against the full dataset.

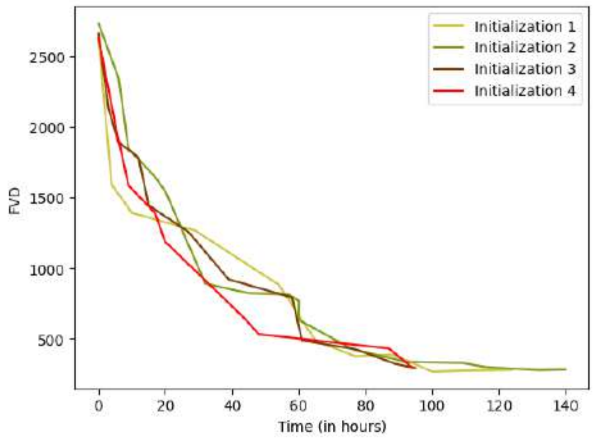


Figure 3.14: INR-V on progressive training with different initializations: the order of video selection for each stage differ across the initialization.

where μ and σ denote the mean and standard deviation over the latent codes $\{m_n\}_{n=1}^{N}$ and δ is a hyperparameter. In our experiments, $\delta = 1.0$.

As can be observed from Fig. 3.10, reconstruction quality is much worse without CLIP. This is expected as Video-CLIP (see Fig. 3.4) assigns semantic meaning to the initialized codes for each video instance. As we observe the novel video instances generated using this model (Fig. 3.10, right), we already see a motion emerging with expressive faces. As can be observed from Fig. 3.11, on RainbowJelly dataset (made of 34526 instances), INR-V takes more than 250 hours ($\sim$ 11 days) to converge without any regularization scheme or progressive training. With progressive training, the convergence time is drastically reduced to less than 180 hours ($\sim$ 7.5 days). The best performance is achieved when progressive training is done along with CLIP regularization where the convergence occurs in less than 120 hours ($\sim$ 5 days) on 2 NVIDIA GTX 2080ti GPUs. On SkyTimelapse dataset (made of 1803 instances), INR-V converges equally on either of the regularization schemes. With a Gaussian prior, we observed a slight advantage in terms of convergence time. Fig. 3.12 plots an additional comparison on RainbowJelly with progressive training on three different regularization methods: Gaussian, CLIP, and no regularization.

The graphs indicate that CLIP regularization is more suitable for a larger dataset like RainbowJelly, however for a smaller dataset like SkyTimelapse (Fig 3.11, right), Gaussian regularization is more effective. INR-V performs equally well on either of the training schemes, given enough time to train. This indicates that the generation capabilities is inherent to the proposed architecture, whereas the different training schemes help in stabilizing the training and thus, lead to a faster convergence. Additional insights are provided in Sec.3.6.

3.5.2 Effect of the size of the codebook

Fig 3.13 presents a comparison between the FVDs of INR-V when trained on varying number of video instances on the RainbowJelly dataset. FVD is computed against the entire dataset made of $34,526$ samples As can be seen from the graph, the performance of INR-V deteriorates as the number of video instances reduce. However, the effect of the added instances is marginal as the codebook size increases; the FVD improving by only 12% when going from 10K to 34K (24K additional instances) video instances. However, the FVD improves by 20% when the codebook size increases from 500 to 1000 (500 additional instances) video instances.

3.5.3 Progressive Training with different Initializations

To stabilize the hypernetwork training, we train our network progressively as explained in Sec. 3.2.3. In this section, we compare the performance of INR-V on training with different video intializations. We randomly assign videos for each stage of the training and the random assignments differ across the different initializations. As shown in Fig. 3.2.3, INR-V takes about the same time to converge for all of them.

	128×128	256×256	512×512	1024×1024
INR-V	1.68	6.71	26.84	107.36
3DConv	252.71	691.76	3860.77	OOM
	25 frames	50 frames	75 frames	100 frames
INR-V	6.71	13.42	20.13	26.84
3DConv	691.76	2036.05	3719.98	6673.27

Table 3.7: Comparison of computational complexity of INR-V against 3D convolutional-based video generative models for different spatial and temporal dimensions. The computational complexity is the total number of multiply-add Giga operations denoted by GMAC (Multiply-Add Cumulation).

3.5.4 Comparison of Computational Complexity of INR-V against 3D Convolutional models

We compare the computational complexity of INR-V against standard implementations of 3D convolution-based video generation models of varying spatial and temporal dimensions. We call these models as "3DConv". 3DConvs do not generate any meaningful output and are used solely to compare the computation costs against INR-V. They comprise of several transpose 3D convolution-based upsampling layers and take a fixed 128-dimensional latent vector as input. For comparison against INR-V, we gradually vary their spatial dimension from 128×128 to 1024×1024 by keeping the temporal dimension fixed to 25 frames. To generate a video of resolution 128×128, three 3D convolution-based upsampling layers are used. An additional upsampling layer is added for every subsequent jump in the spatial dimension. Next, we vary the temporal dimension for 25, 50, 75, and 100 frames by keeping the spatial extent and the number of layers fixed to 256×256 and four, respectively, and adjusting the kernel size corresponding to the temporal dimension. The stride, kernel size, and padding are appropriately adjusted for all the models. The batch size for comparison is fixed to 1 for both 3DConv and INR-V. As can be seen in Table. 3.7, the number of operations (MAC) increases drastically as the spatial dimension increases for the 3DConvs. It becomes prohibitively expensive to generate videos of higher spatial dimensions. For example, generating a single video of 25 frames of dimension 1024×1024 results in out of memory (OOM) on a single NVIDIA GTX 2080 ti GPU with a memory of 12 GB. In summary, MAC remains hundreds of orders of magnitude lower for INR-V compared to its 3D convolution-based counterparts as it mainly comprises inexpensive MLPs.

3.6 Insights on the learned latent space

Unlike the existing video generation networks that are conformed to a predefined latent space (Gaussian or Uniform), INR-V learns a space that best fits a given distribution. The experiments (Sec. 3.4.1 and Sec. 3.4.3.3) and our observations indicate that the learned space is continuous, supports inversion, and smooth video interpolations. Thus, such a space learns a structure in the dataset. For instance,

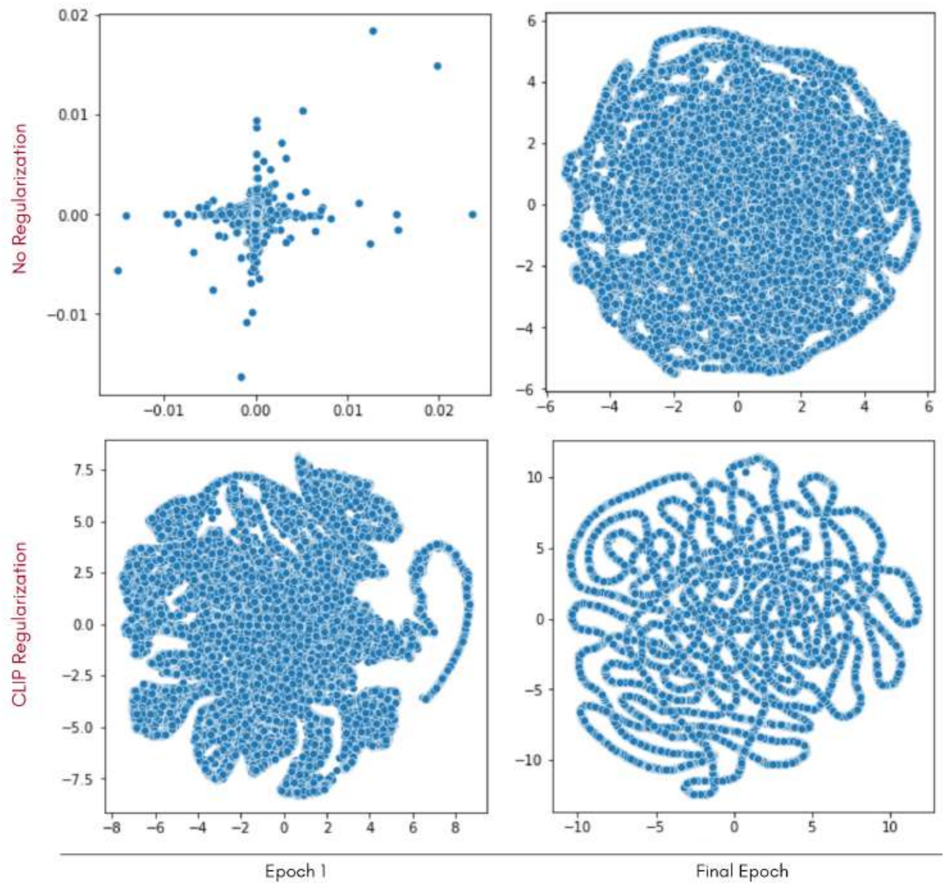


Figure 3.15: t-SNE visualization of the latent codes (m_n) learned by INR-V with and without CLIP regularization on RainbowJelly. In Epoch 1, the latent codes are bounded within a very small region when INR-V is trained without any regularization (top-left). With CLIP regularization, the latents are more spread out (bottom-left). In the final epochs (top & bottom right), both latent representations have spread out farther. INR-V without regularization forms a denser space; and with regularization converges faster. Both spaces are continuous, form meaningful interpolations, and support inversion.

we observe a smooth transition across different poses, expressions, mouth movements, and identity on How2Sign-Faces. Therefore, novel instances are observed as we traverse the path between seen latent points A and B. We use this property to generate novel videos by sampling latent points through Slerp interpolation. In this section, we aim to validate the space learned by INR-V and answer the following question: what does INR-V learn? We analyse this in two ways: (1) by visualizing the latent codes learned by INR-V through t-SNE visualization and (2) by training INR-V on a structured toy dataset to see if it learns the underlying structure.

t-SNE visualization Fig. 3.15 plots t-SNE[4] on the latent codes learned by INR-V on the Rainbow-Jelly dataset. We see a clear pattern of ”interpolation” occurring in the learned latent space in both cases (with and without CLIP regularization). No progressive training is used in this visualization. At the end

[4] https://lvdmaaten.github.io/tsne/

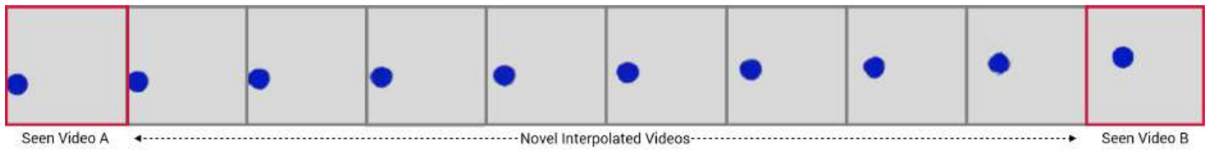


Figure 3.16: We demonstrate the ability of INR-V to learn the underlying structure of a dataset. To do so, we create a toy video dataset called BouncingBall consisting of a blue ball bouncing horizontally at different heights. Given 50 instances of such videos, INR-V learns the structure and we demonstrate the learned structure through interpolation. In this example, we show the 16th frame of each video with the novel interpolated videos in gray boxes. The videos in red boxes are seen during training. We can observe a correct interpolation with smooth transition in motion: the height and the horizontal position of the ball gradually shifts from bottom to top and left to right respectively.

of the first epoch, the latent codes are tightly bounded when trained without CLIP regularization. As the model is trained with CLIP, the latent vectors are more spread out possibly in a semantically meaningful manner. In both the cases, by the final epoch, we observe patterns of interpolation evolve.

3.6.1 Learning Bouncing Ball

In this section, we want to analyze if INR-V can learn the structure of a dataset. To do so, we generate a toy dataset, BouncingBall, with an artificially infused structure. The dataset is made of 50 video instances where each video instance consists of a blue ball bouncing horizontally at different heights as shown in Fig. 3.17. The videos are of dimension 100×100 and are 25 seconds long each.

As shown in Fig. 3.16, INR-V is able to learn the infused structure when trained on BouncingBall. We demonstrate interpolated videos by sampling intermediate points through Slerp interpolation. The intermediate videos (grey boxes) demonstrate a smooth interpolation, and have heights and horizontal displacements gradually varying from Video A to B (red boxes). An example of an intermediate video shown in Fig. 3.18.

3.7 Discussion

3.7.1 Limitations.

Although INR-V has learned a powerful video space demonstrating several intriguing properties, the videos generated by the model are sometimes blurry. This is prominent when moving away to unseen points in the video space far from the seen instances. Fig. 3.6 demonstrates the enhancement on one such blurry sample. This is done by training a standard VQVAE2 network in a denoising fashion (please refer to Sec. 3.3.2). However, the entire process is broken into generating a relatively lower quality output and relying on a second network to improve its quality. A single end-to-end network capable of retaining the demonstrated powerful properties while generating high-quality videos is a potential future work.

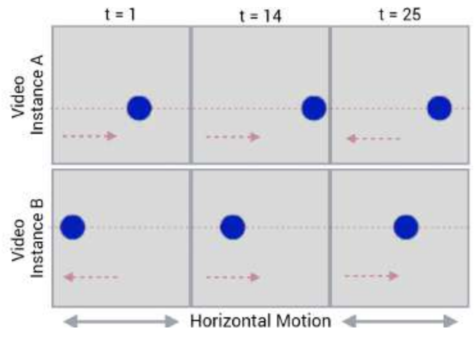


Figure 3.17: BouncingBall dataset with an infused structure. Each video instance is 100×100 and has a ball bouncing horizontally at a specific height. Red lines are added to show the height of the ball and is not a part of the videos.

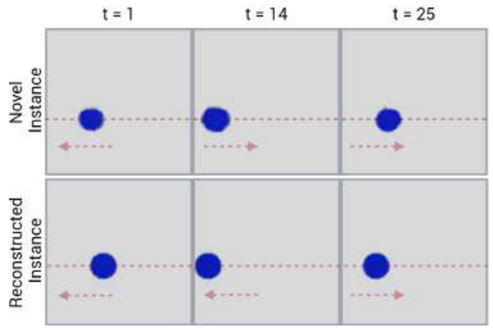


Figure 3.18: Videos generated by INR-V on BouncingBall. Novel video is generated at an unseen height. Red lines are for demonstration and not generated.

Another limitation of INR-V is generating infinitely long videos. Although coupling the content and time into a single latent has clear advantages, it removes the network's ability to leverage the temporal dimension separately and find infinitely long temporally coherent paths in the image space. This can be tackled by training INR-V to encode video segments of multiple lengths in a single space (1 to 50 or more frames long video segments). A temporally and semantically coherent trajectory between these video segments can then be learned. Such a generation technique would directly leverage video segments and potentially remove repetitions in the long videos. We believe that leveraging a video space for generating infinitely long videos at multiple resolutions presents an interesting and exciting direction for future research.

Lastly, we observed that INR-V does not learn a meaningful representation space when trained on datasets like UCF-101 that have extreme diversity and limited structure in motion. A similar issue was observed when training the baseline models on such datasets. However, INR-V can be trained on a single action class of UCF-101 (such as JumpRope) to learn a meaningful representation space even with significant camera motion and in-video subject movement. A single action class limits the visual and motion diversity in the dataset.

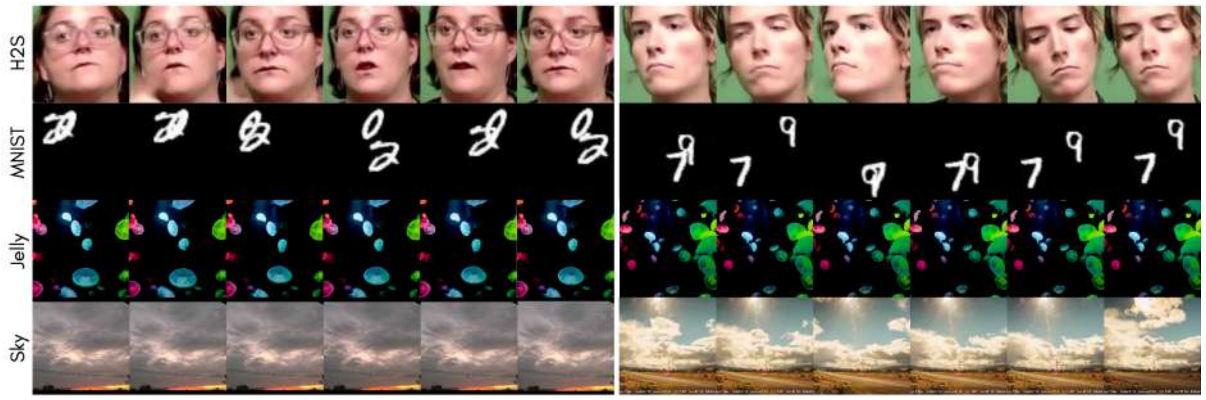


Figure 3.19: Examples of real videos instances of How2Sign-Faces (H2S) [23], Moving-MNIST (MNIST) [91], RainbowJelly (Jelly), and SkyTimelapse (Sky) [106] datasets.

3.7.2 Broader Impact.

The potential negative impact of our work is similar to existing image-based and video-based GANs: creating "photorealistic-deepfakes" and using them for malicious purposes. Our simple training strategy makes it easier to train a model which produces realistic-looking videos. However, this is partly addressed for the following reasons: (1) Even though our network produces diverse novel videos, the perceptual quality of our generated videos falls short of the existing state-of-the-art image-based generators that produce high-resolution images. (2) The availability of high-quality video datasets limits the intended malicious use of this codebase. Despite these limitations, we believe that the potential of our work far outweighs its limitations. A continuous video representation space offers tremendous applications in areas requiring video prediction, interpolation, and conditional video generation. E.g. pedestrian trajectory prediction is an important area of research for self-driving cars. Pedestrian trajectory prediction through future frame generation can serve to reduce accidents in fully-autonomous vehicles in the future. Similarly, conditional video generation can be used for synthesizing novel sign language videos that can be integrated into schools and universities to encourage and enable hard-of-hearing students.

3.8 Additional Qualitative Results

Fig. 3.19 presents the real video instances in the training set. Fig. 3.20 and Fig. 3.21 presents qualitative results on the reconstruction of video instances from different training datasets. Fig 3.22 presents random videos generated by INR-V on different datasets. Fig 3.23 and Fig. 3.24 present spatio-temporal view of video interpolations on How2Sign-Faces and SkyTimelapse respectively. Fig. 3.25 presents the random generation of INR-V on multiple resolutions starting from 32×32 to 256×256 jumping a scale factor of $8\times$. The visualization is up to scale, and one can see the scale jump. INR-V can also be inferred at multiple frame rates. Fig. 3.26 - Fig. 3.32 present the qualitative results and comparisons on the proposed inversion tasks. Fig. 3.30 presents an example of multi-modal future segment prediction.

3.9 Chapter Summary

We present INR-V, a continuous video representation network. Unlike existing architectures that extend superior image generation networks for generating videos one frame at a time, we use implicit neural representations to parameterize videos as complete signals allowing a meta-network to encode it to a single latent point. Given enough examples, the meta-network learns a continuous video space as demonstrated through video interpolation and inversion tasks. INR-V generates diverse coherent videos outperforming many existing video generation networks. INR-V opens the door to a multitude of video-based tasks and removes the dependency on an image generator. To showcase this, we propose several downstream tasks and observe that INR-V outperforms the existing works on a majority of these tasks. This demonstrates the advantages and potential of a continuous video space and we hope to encourage research in this direction.

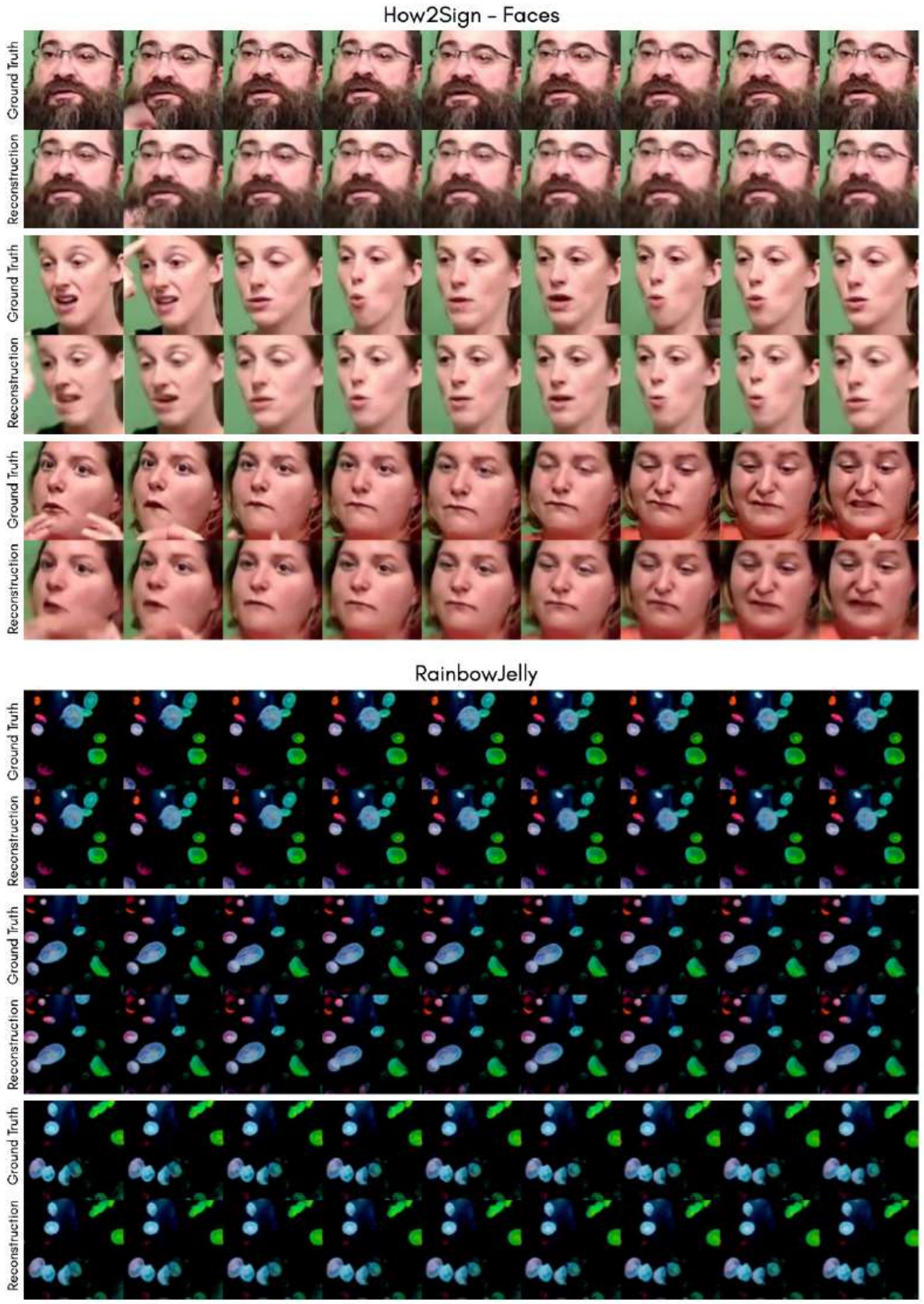
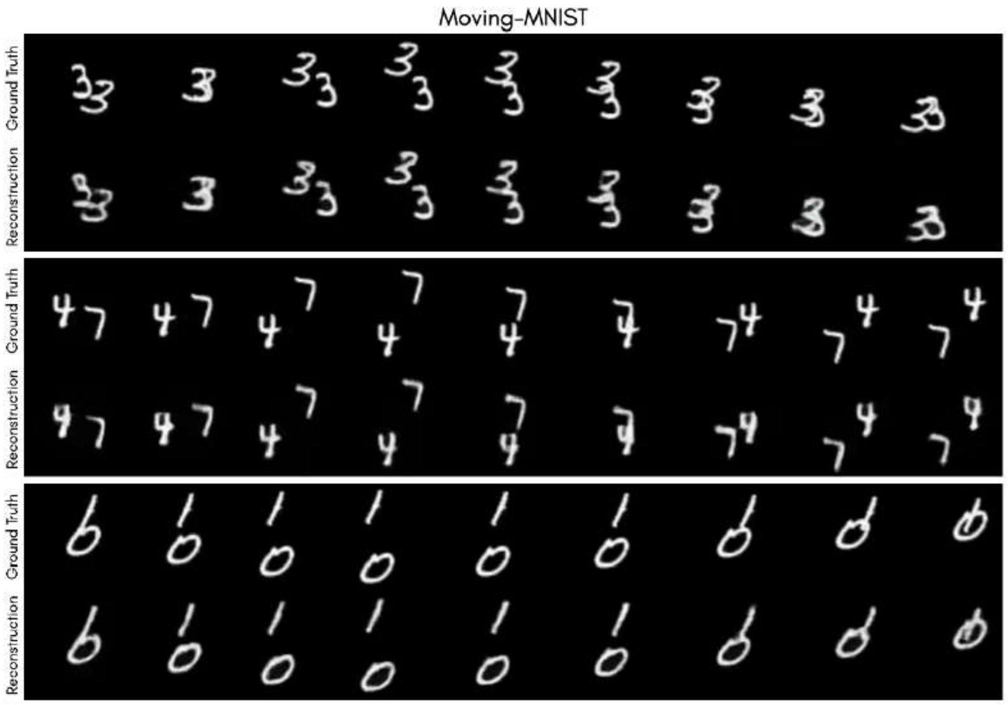


Figure 3.20: Examples of video instances in the training set reconstructed by INR-V.



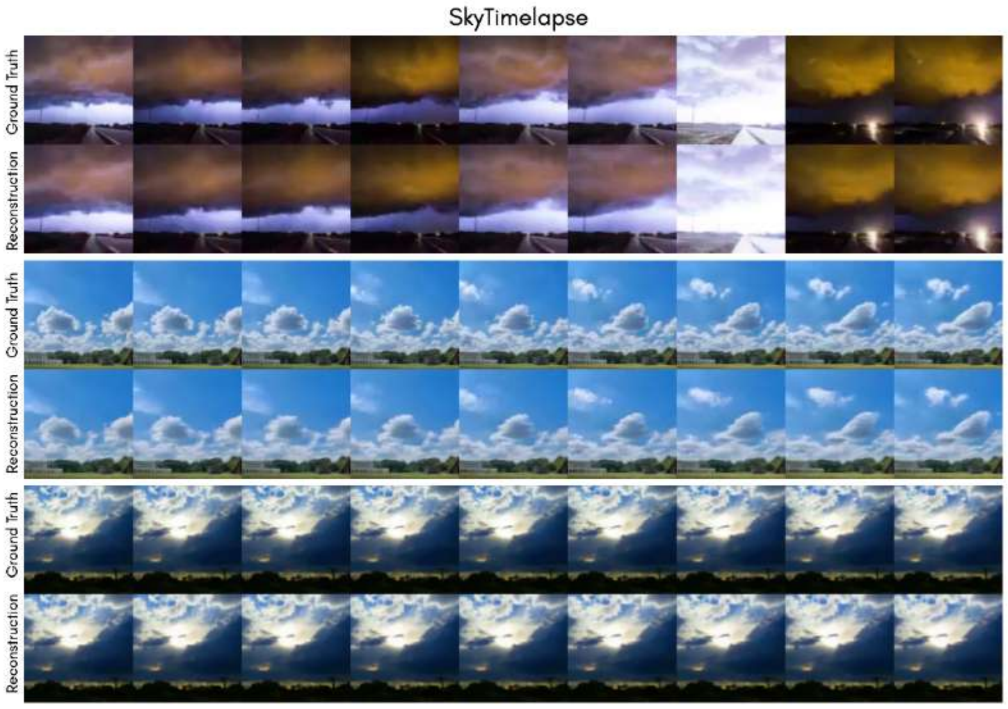


Figure 3.21: Examples of video instances in the training set reconstructed by INR-V.

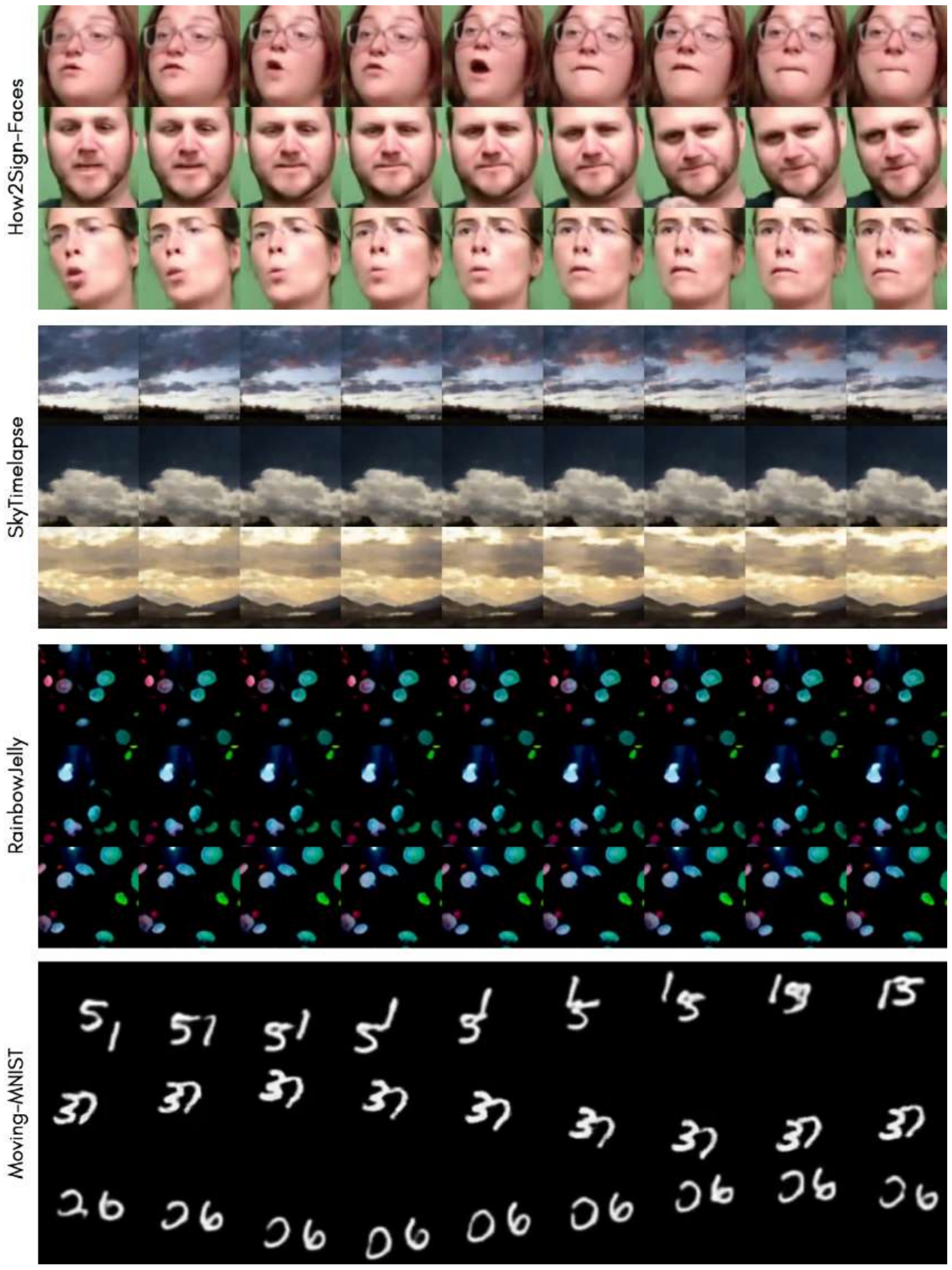


Figure 3.22: Examples of random videos generated by INR-V.

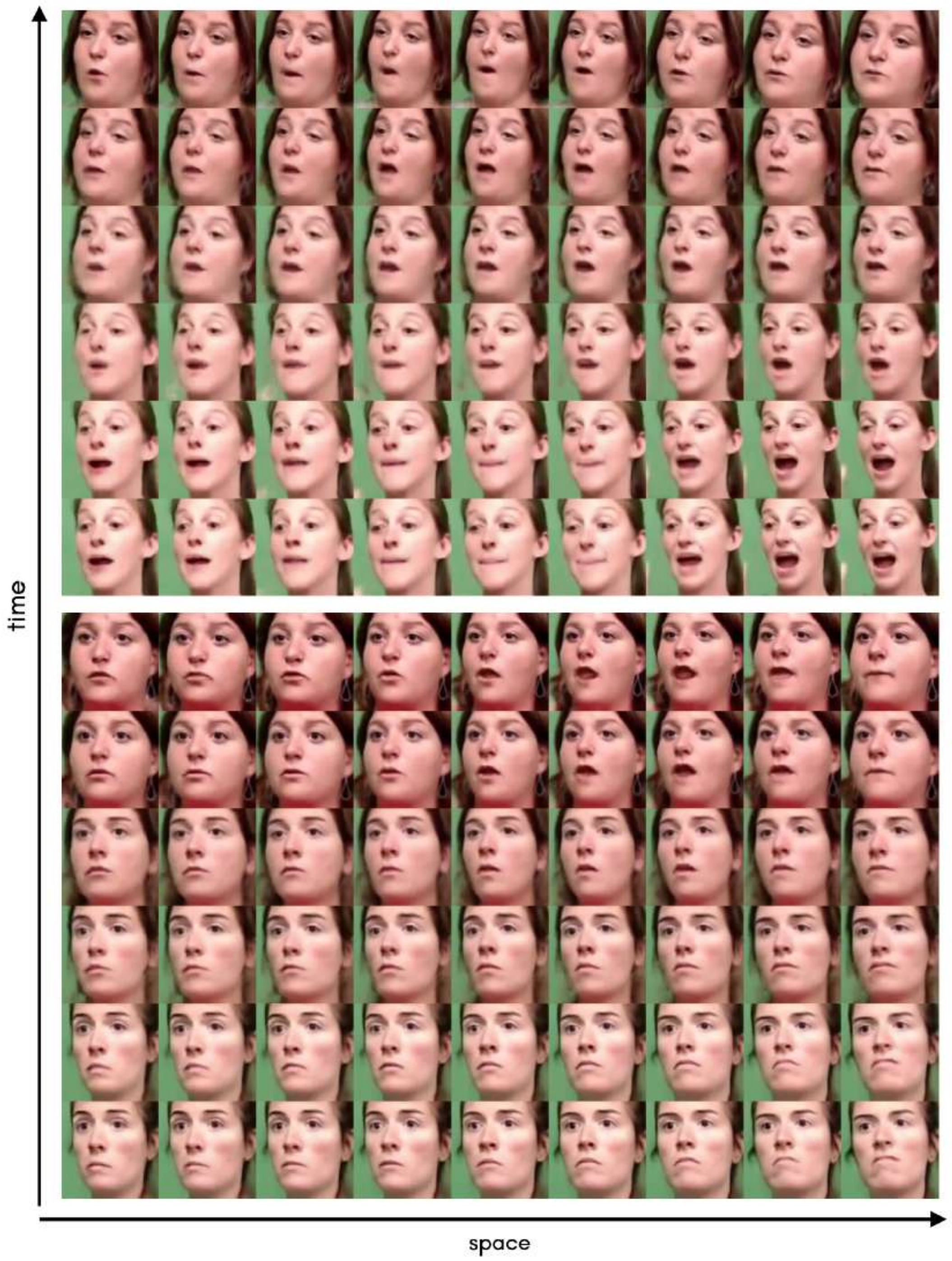


Figure 3.23: Examples of video interpolation in INR-V on How2Sign-Faces. Two latent points are sampled from the training dataset. Intermediate videos are then generated by sampling intermediate latent points using Slerp interpolation technique.

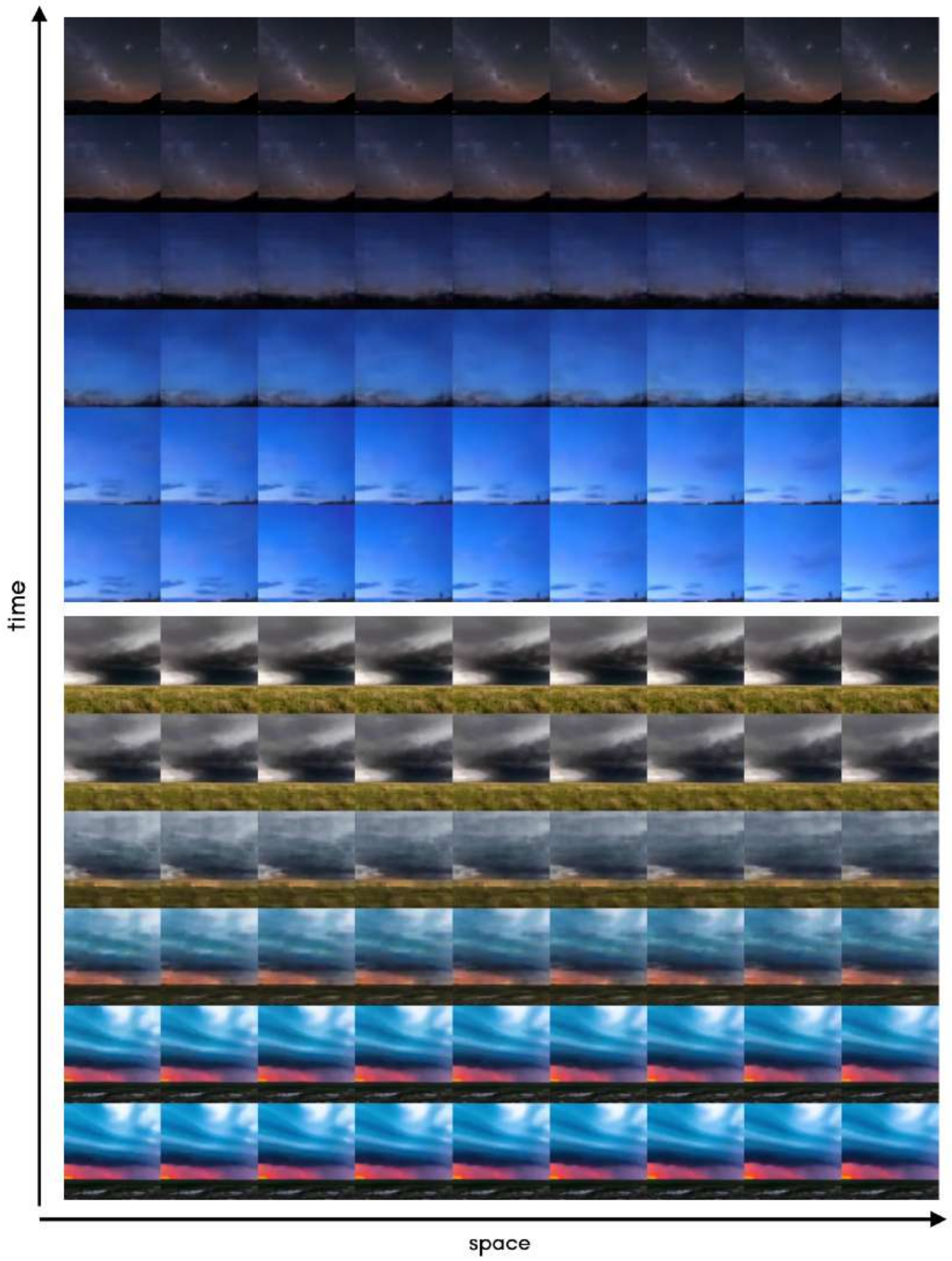


Figure 3.24: Examples of video interpolation in INR-V on SkyTimelapse. Two latent points are sampled from the training dataset. Intermediate videos are then generated by sampling intermediate latent points using Slerp interpolation technique.

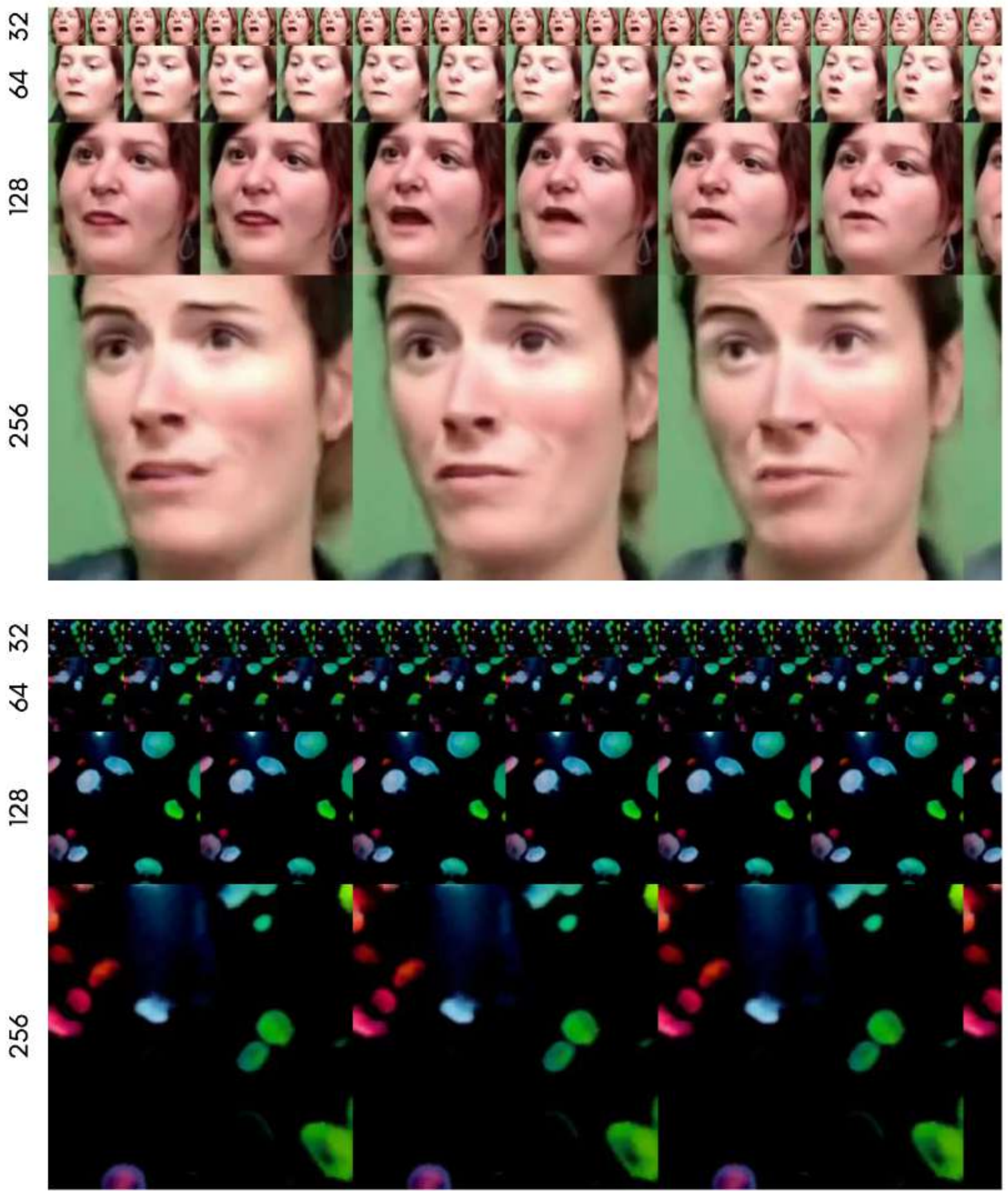


Figure 3.25: Examples of random videos generated by INR-V at multiple resolutions of 32×32, 64×64, 128×128, and 256×256 on How2Sign-Faces (top) and RainbowJelly (bottom). The videos are 25 frames long each. The videos are upto scale. INR-V was trained on videos of only 100×100 resolution.

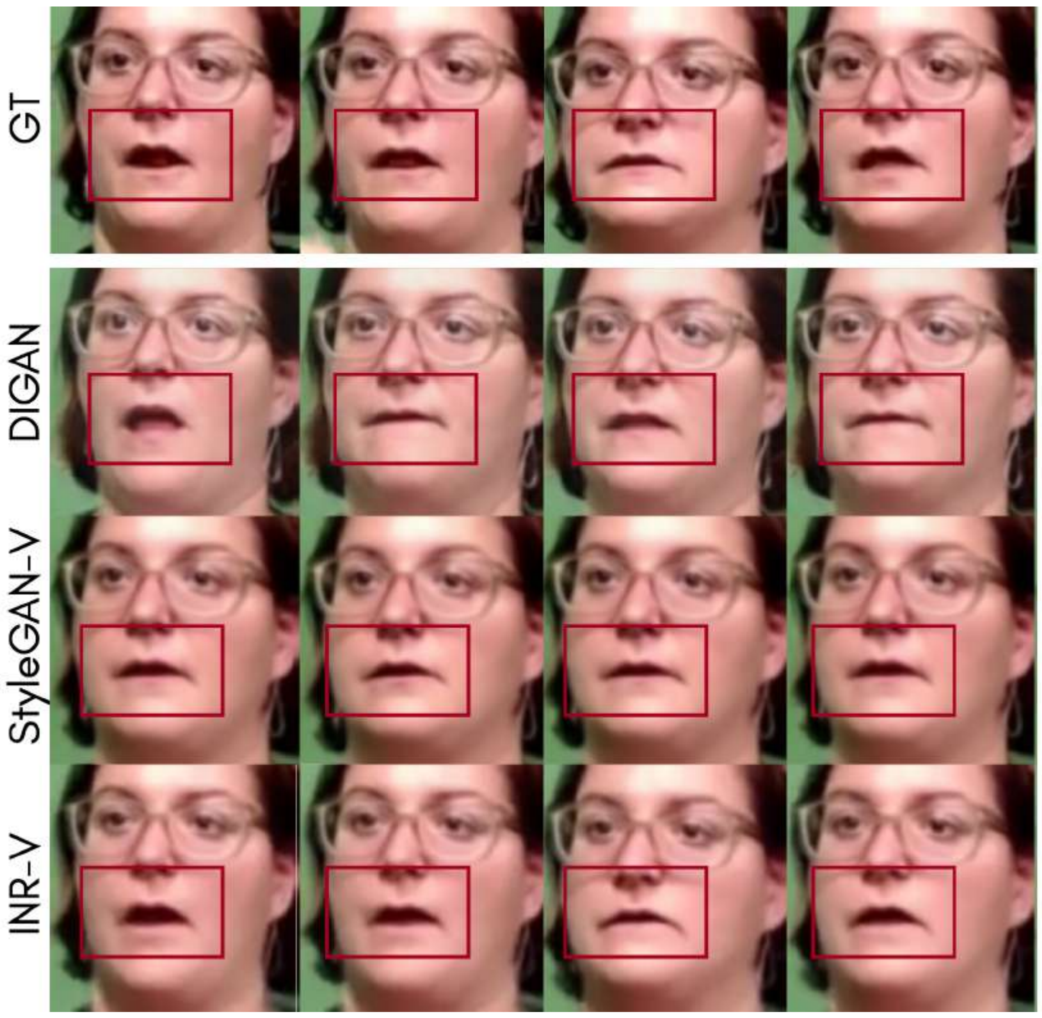


Figure 3.26: Comparison of video inversion. Red boxes highlight the differences and matches between the ground truth (GT) and the various methods. To note, INR-V is able to preserve the finer mouth movements.

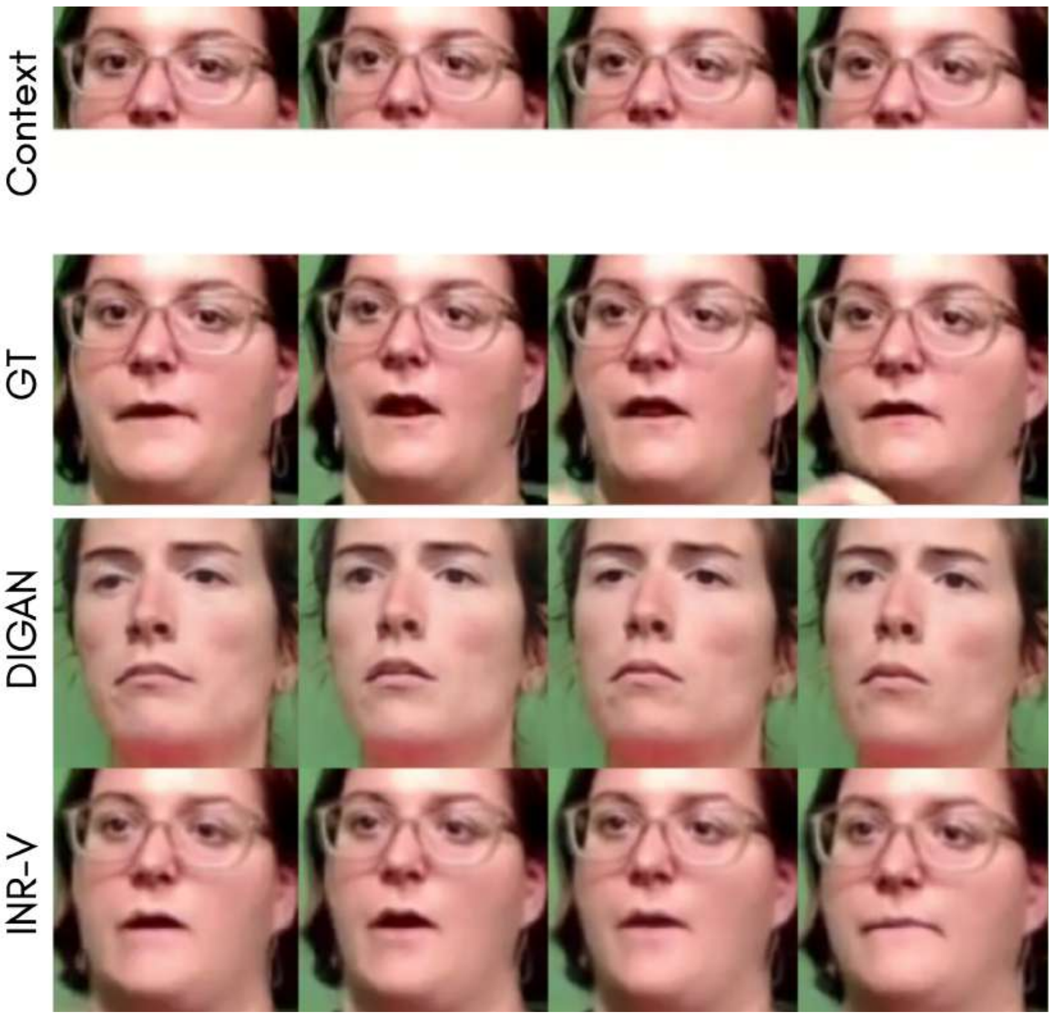


Figure 3.27: Comparison of half-context inversion in an inpainting setting. At the time of optimization, the model only sees the top half of the video. It then generates the full video back. There can be multiple correct predictions, we showcase one such prediction.

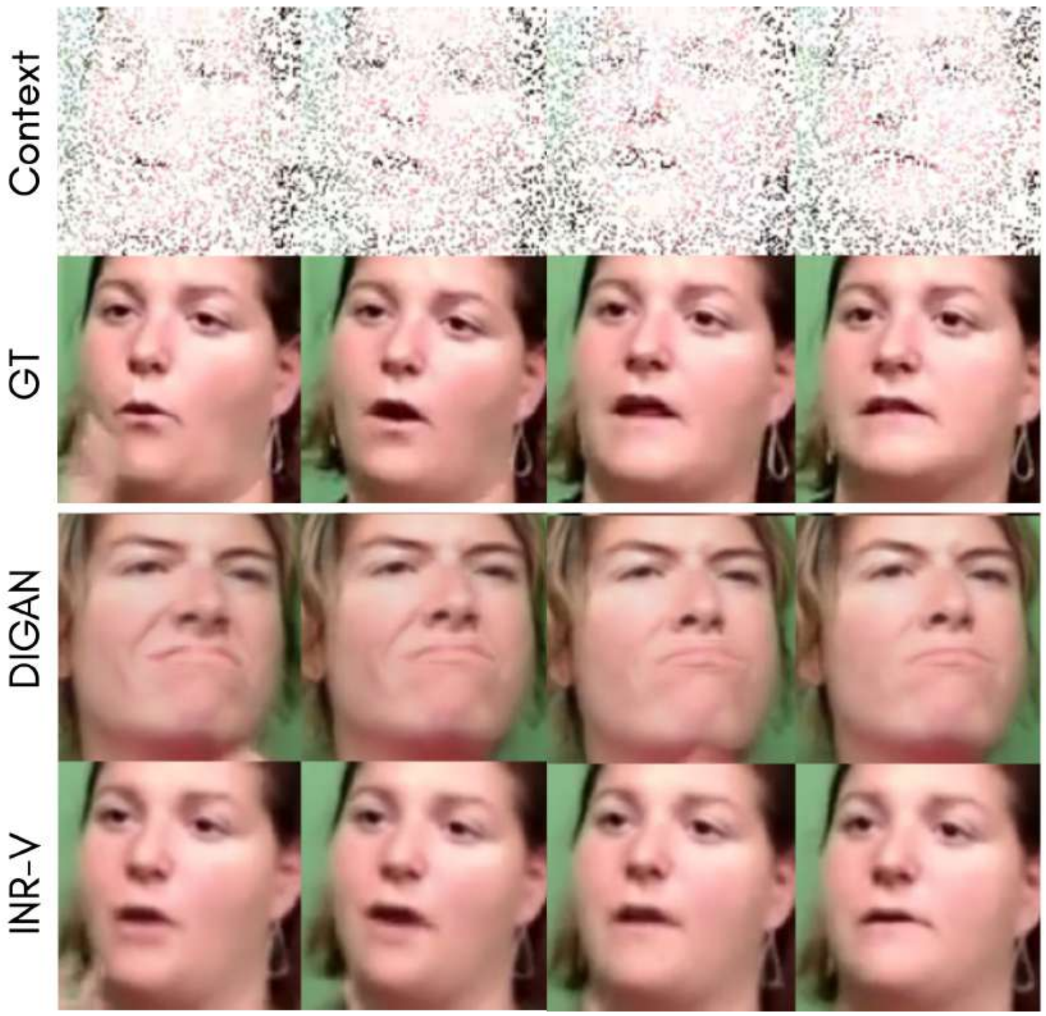


Figure 3.28: Comparison of half-context inversion in a sparse context setting. At the time of optimization, the model only sees 25% of the full video. INR-V preserve the identity including finer content details like earrings. It also preserves motion like pose and mouth movements.

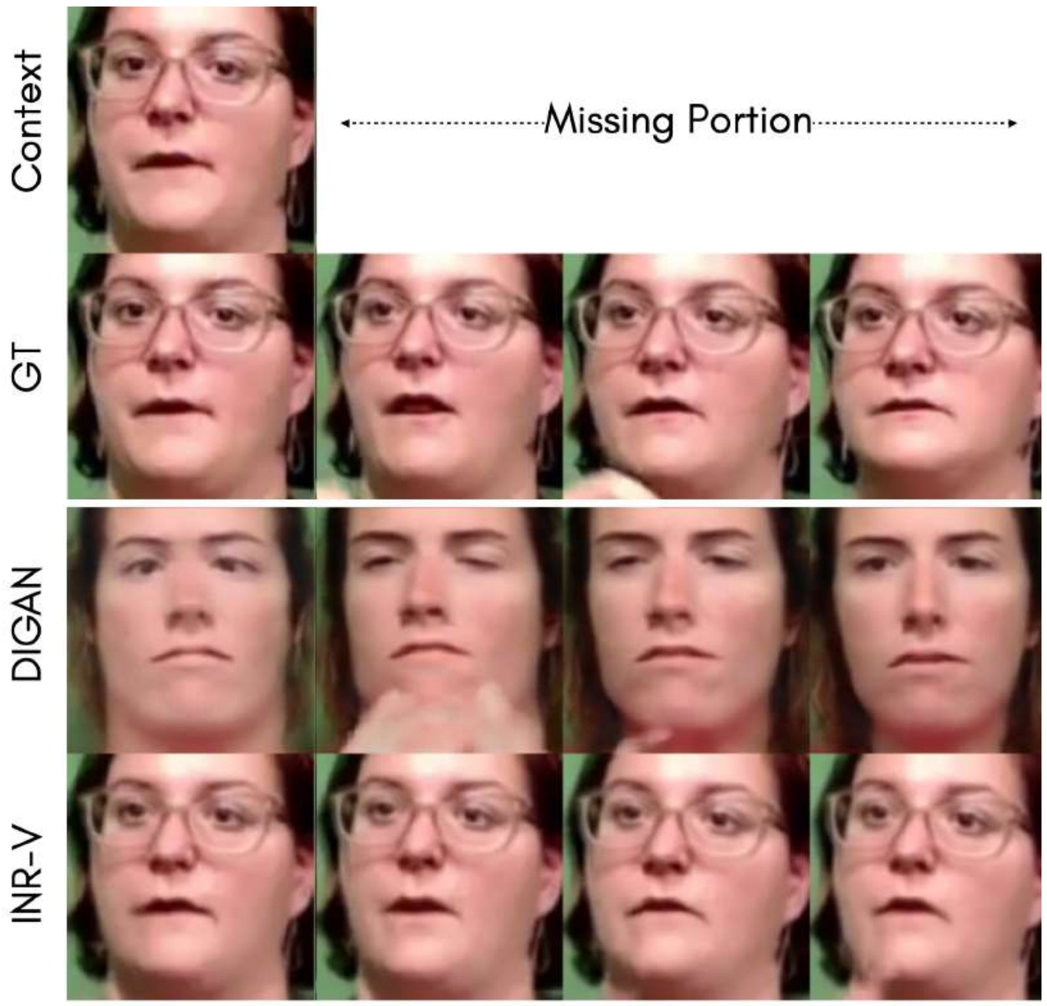


Figure 3.29: Comparison of half-context inversion in a future frame prediction setting. At the time of optimization, the model only sees the first 4 frames of the video. There can be multiple correct predictions given the identity is preserved across the video. We show one such example.

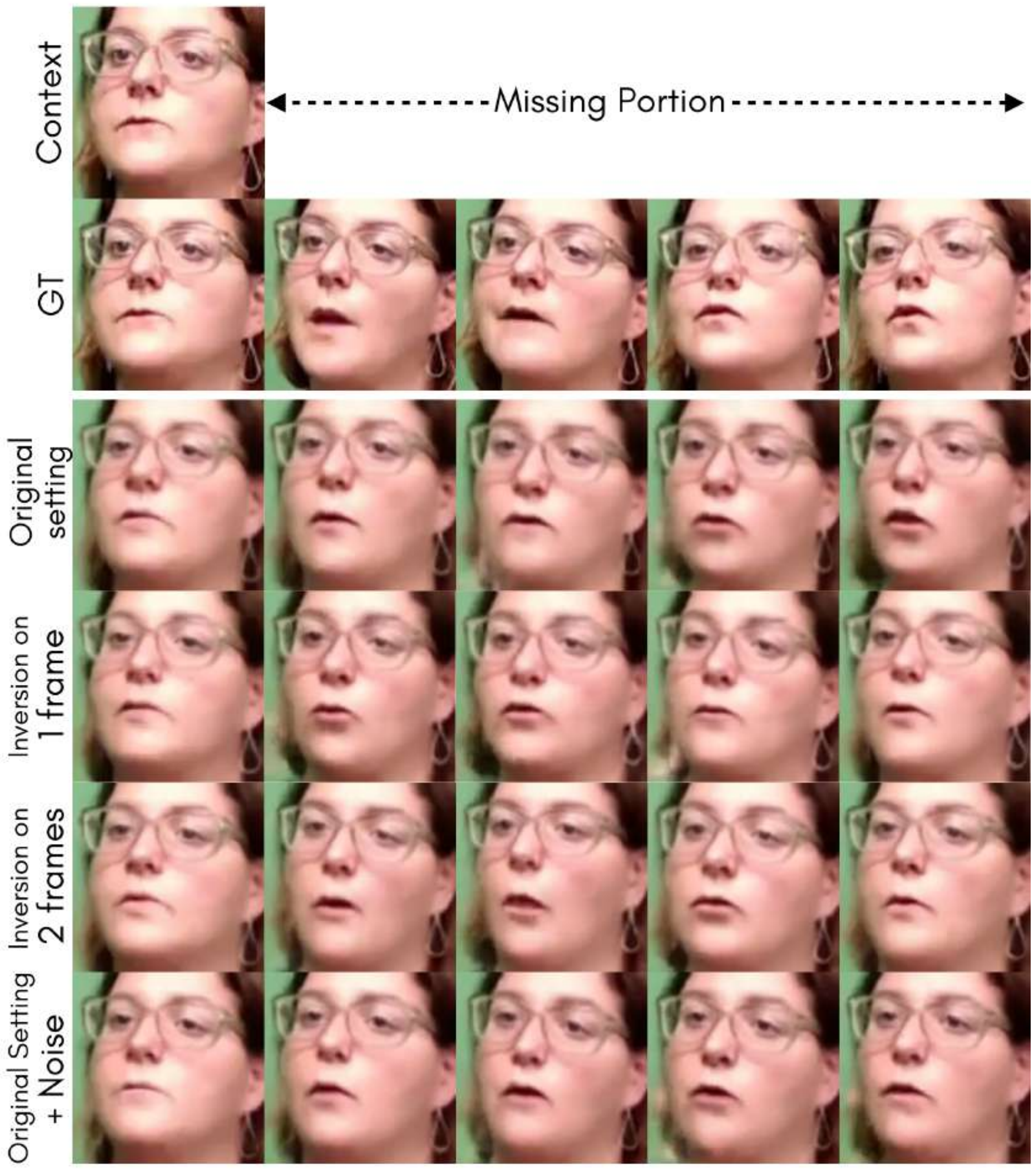


Figure 3.30: Multimodal Future Frame Prediction: Given the first few frames of a video, we want to predict multiple different outcomes. This can be achieved using INR-V by conditionally optimizing the video's latent vector on half-context. In this example, we condition the latent by varying the number of frames used for inversion or by adding Gaussian noise to the optimized latent. As can be seen, the inversion preserves the seen context; and the future predictions have varying mouth movements, pose, and eye gazes.

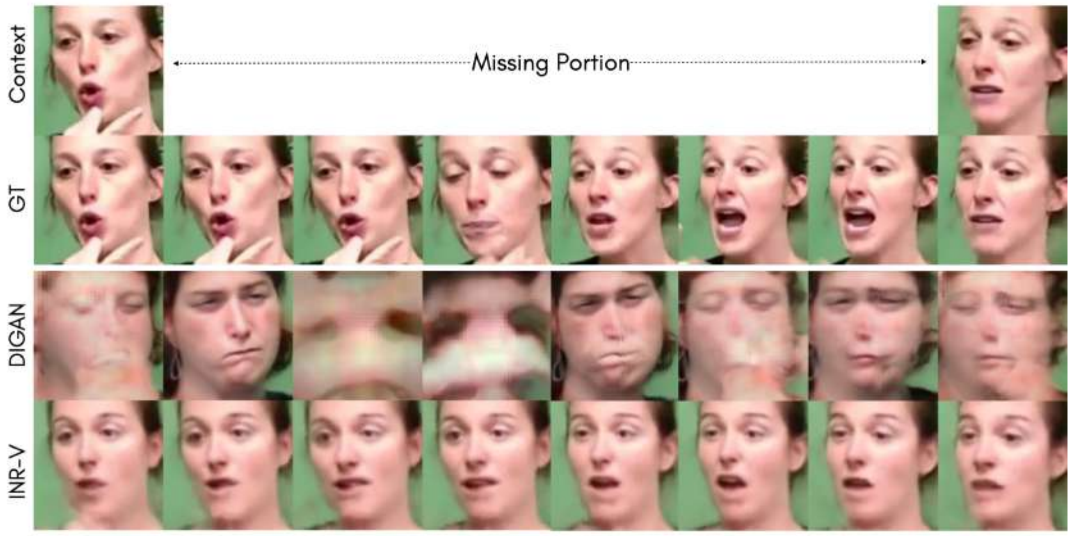


Figure 3.31: Comparison of half-context inversion in a frame interpolation setting. At the time of optimization, the model only sees the first and last frames of the video. As can be seen, the first and the last frame generated by INR-V match the context (pose, identity, mouth movements), whereas the intermediate frames are very different from the ground truth.

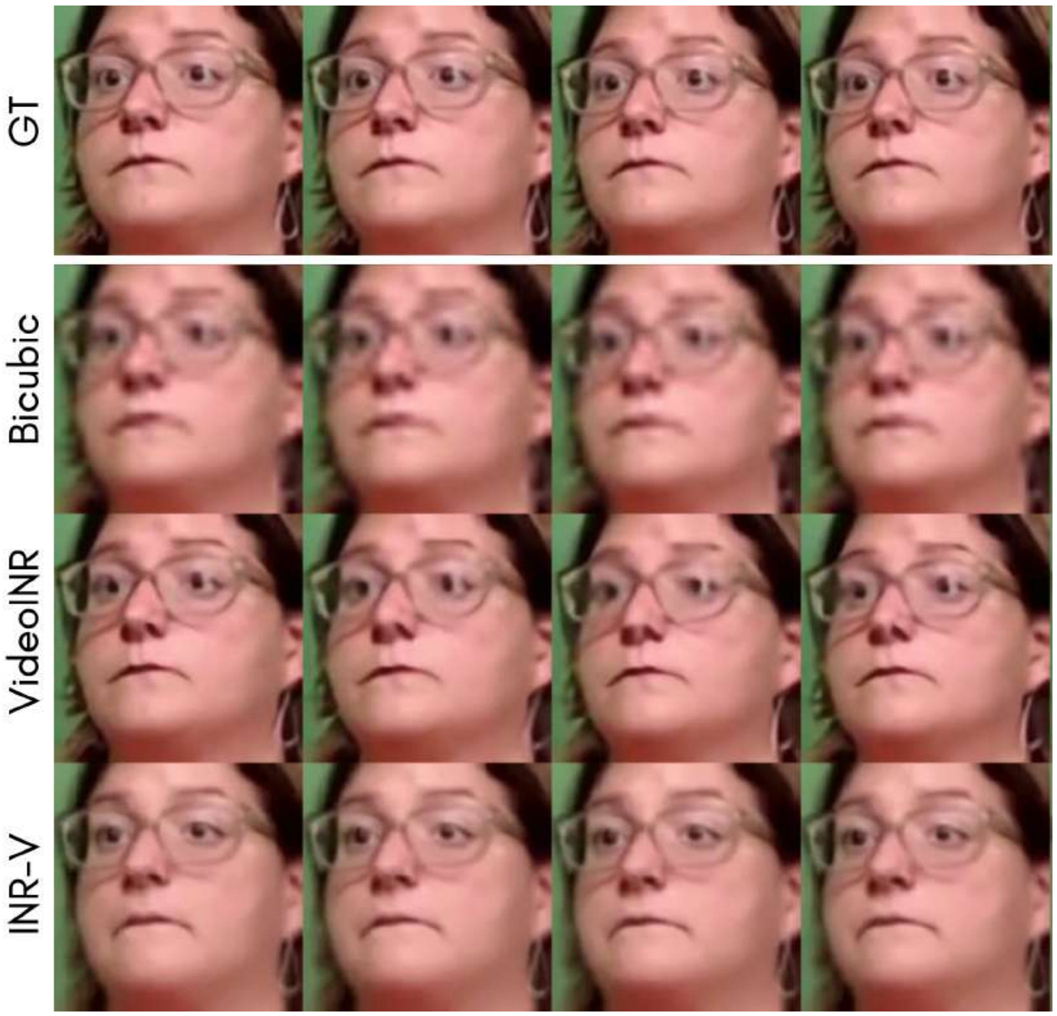


Figure 3.32: Comparison of video superresolution. A video of 32×32 is given as input to INR-V for optimization. Once the video is optimized, INR-V regenerates the video at a higher resolution of 128×128. VideoINR and Bicubic directly see the 32×32 video and superresolves it to 128×128. Here, INR-V is not influenced by the glaze on the spectacles and superresolves to a higher dimension closer to the ground truth.

Chapter 4

Conclusions

Video is an important modality, perhaps most closely resembling human perception encompassing both - the spatial and the temporal dimensions. Such a modality opens up enormous scope for downstream applications that are otherwise impossible to do with static images - for example, pedestrian trajectory prediction. Without an understanding of a pedestrian's past temporal change in trajectory, predicting the future is ill-posed and impossible for even a human. On this front, this thesis aims to answer the question: What is a good representation space for videos?

The thesis starts by exploring the critical problem of lipreading people suffering from speech disabilities. The primary challenge in this setting is the need for more data at scale, often taken for granted in most computer vision applications. Suffering from conditions like Amyotrophic lateral sclerosis (ALS) results in limited body movement, making it extremely taxing to collect any data. To address this, we consider setting a one-shot classification of mouth movements into 200 classes of words. We perform data augmentation to tackle this by adding synthetic videos to the training set using advanced lipsync models. However, an overflow of synthetic data causes a big domain gap between the real and the synthetic datapoints. We propose Variational Encoders (based on Variational AutoEncoders (VAEs)), a novel domain adaptation technique that augments the one-shot real datapoints directly in the latent space. Unlike VAEs, which first reduce a datapoint into a compressed space and reconstructs the data, Variational Encoders use a classifier to classify the latent space. This way, Variational Encoders learn a multimodal distribution, where each mode represents a class. Unlike VAEs that use KL-divergence to regularize the compressed space against a known distribution like Gaussian, Variational Encoders regularize each mode of the real distribution to the corresponding mode of the synthetic distribution, thus performing domain adaptation and encouraging self-replication for the real-examples directly in the compressed space.

Finding a good representation space for videos has been extensively studied through classification tasks - for example, action classification. However, such representation techniques require labeled datapoints for each video example. Architectures like VAEs and generative adversarial networks (GANs) have been widely used to learn representations in a self-supervised manner for images. However, adopting such video architectures is challenging primarily because of the complexity of generating a full video for representation learning. In the next part, the thesis aims to develop a representation space for video generation that can enable diverse downstream video-based generative tasks. The thesis proposes a novel space called a "video space" to do so. Existing architectures generate videos one frame at a time - they extend an image generator that samples temporally coherent trajectory from an image space (where each point corresponds to a frame/image) to generate the video frame-by-frame. On the other hand, this thesis proposes a video space in which a single point corresponds to a full video. To achieve this, the videos are first parameterized as implicit neural representations (INRs) that generate videos one pixel at a time. Such parameterization reduces the high dimensionality of videos to only a fixed number of weights in the neural network. Then a meta-network (hypernetwork) is trained to learn a prior over the INRs. In this manner, an INR function is represented as a single point in the latent space. This formulation not only simplifies complex tasks but also opens up possibilities for diverse applications, such as video interpolation, video inversion, etc.

As videos play a crucial role in modern society, there is a pressing need to develop effective approaches for understanding and analyzing them. The thesis touches upon two aspects of video-based tasks: classification and generation and proposes novel formulations to enable and improve several downstream tasks. The experimental sections show that the proposed methods outperform the existing baselines. The thesis also provides extensive analysis and ablation studies over the proposed representation spaces, clearly showcasing the proposed methods' advantages over the existing ones. The thesis aims to contribute to the advancements of video understanding that can create huge impacts in every field of human lives, including space exploration, general-purpose artificial autonomy, and augmented/virtual reality applications.

4.1 Discussion and Future Work

Video Classification and Generation are two opposite spectrums of a common modality. On the one hand, video classification aims to <u>reduce</u> a the given video to a compressed representation containing important task-relevant details; on the other hand, video generation starts from the compressed representation to <u>upsample</u> a video output belonging to a valid video distribution.

There are similarities in both spectrums – most importantly, both tasks aim to find a higher dimensional space that preserves important details about the video. The primary dissimilarity, however, is the manner in which both of the tasks are modeled. For instance, in a classification task, the representation space plays a major role - the underlying space is semantically meaningful, demonstrating key properties useful for the downstream classification task. In standard generative tasks, however, the representation space is usually a standard distribution (like Gaussian distribution) that does not, by itself, showcase interesting properties. Instead, the decoder plays the key role by mapping a standard distribution to the output video distribution.

AutoEncoders present an interesting alternative to this decoupling that accomplishes encoding and decoding at the same time. For example, a recent work, VQVAE2 [73], encoded a given image into a quantized space. The decoder then efficiently decoded the quantized representation to the input image almost losslessly. This quantized space could then be used, firstly, to retain task-specific information, hence to perform classification, while at the same time, it is presented as an efficient space that can be used for sampling (and thereby generating) high-quality images. This thesis aimed to extend this line of work to videos. Firstly, we used a variant of Variational AutoEncoders (Variational Encoders) for learning robust video representations. In this variant, the model was trained using a classification loss instead of an autoencoding loss. Secondly, unlike the existing generative works that start with a standard Gaussian-like prior, our proposed "video space" (or the video prior) is learned along with the decoder that allows the prior to becoming semantically meaningful by itself. Here, the decoder is constituted by only a few layers of multi-layered perceptrons, leaving the heavy lifting to the underlying semantic space.

On this front, an interesting future work could be to combine both spectrums by building an auto-encoding framework for videos. Such an approach could also be useful for improving the underlying semantic space, which is now learned in an auto-decoding fashion in this thesis. This thesis takes a step in that direction by incorporating CLIP [71] based encodings to improve the underlying semantic space for video generation. However, a more robust approach that is trained for both the tasks (classification and generation) could enable multitudes of applications while improving the representation space.

Another interesting future direction is to investigate the interpretability of video representations, as understanding the learned features can aid in designing better models and developing applications. Furthermore, research can also be done on developing more efficient and scalable video representation learning techniques, as current methods often require significant computational resources and are not practical for real-world applications. Finally, extending the proposed INR-based video parameterization scheme to handle multimodal distributions (such as videos, images, and speech) can lead to improved representation learning and a better understanding of the underlying dynamics in the data.

Related Publications

1. **INR-V: A Continuous Representation Space for Video-based Generative Tasks**, Bipasha Sen*, Aditya Agarwal*, Vinay Namboodiri, C V Jawahar. *Transactions on Machine Learning Research (TMLR), 2022*.

2. **Personalized One-Shot Lipreading for an ALS Patient**, Bipasha Sen*, Aditya Agarwal*, Rudrabha Mukhopadhyay, Vinay Namboodiri, C V Jawahar. *British Machine Vision Conference (BMVC), 2021*.

Publications Not Included In The Thesis:

3. **Towards MOOCs for Lipreading: Using Synthetic Talking Heads to Train Humans in Lipreading at Scale**, Aditya Agarwal*, Bipasha Sen*, Rudrabha Mukhopadhyay, Vinay Namboodiri, C V Jawahar. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023*.

4. **FaceOff: A Video-to-Video Face Swapping System**, Aditya Agarwal*, Bipasha Sen*, Rudrabha Mukhopadhyay, Vinay Namboodiri, C V Jawahar. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023*.

5. **Fréchet Semantic Distance: A new metric to evaluate the underlying semantics of generated images**, Sai Niranjan Ramachandran, Rudrabha Mukhopadhyay, Bipasha Sen, Aditya Agarwal, Vinay Namboodiri, C V Jawahar. *Under Review*.

(*equal contribution)

Bibliography

[1] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Deep audio-visual speech recognition. In *arXiv:1809.02108*, 2018.

[2] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Deep audio-visual speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2019.

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

[4] A. Agarwal and B. Sen. An approach towards action recognition using part based hierarchical fusion. In *Advances in Visual Computing: 15th International Symposium, ISVC 2020, San Diego, CA, USA, October 5–7, 2020, Proceedings, Part I 15*, pages 306–318. Springer, 2020.

[5] A. Agarwal, B. Sen, R. Mukhopadhyay, V. P. Namboodiri, and C. Jawahar. Faceoff: A video-to-video face swapping system. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3495–3504, 2023.

[6] A. Agarwal, B. Sen, R. Mukhopadhyay, V. P. Namboodiri, and C. Jawahar. Towards moocs for lipreading: Using synthetic talking heads to train humans in lipreading at scale. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2217–2226, 2023.

[7] D. Beukelman, S. Fager, and A. Nordness. Communication support for people with ALS. *Neurology Research International*, 2011:1–6, 2011.

[8] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3034–3042, 2016.

[9] cdc.gov. Speech reading. In *Centers for Disease Control and Prevention*. Centers for Disease Control and Prevention, Jun 2020. `https://www.cdc.gov/ncbddd/hearingloss/parentsguide/building/speech-reading.html`.

[10] A. Chadha, J. Britto, and M. M. Roja. iSeeBetter: Spatio-temporal video super-resolution using recurrent generative back-projection networks. *Computational Visual Media*, 6(3):307–317, jul 2020.

[11] H. Chen, B. He, H. Wang, Y. Ren, S. N. Lim, and A. Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021.

[12] Z. Chen, Y. Chen, J. Liu, X. Xu, V. Goel, Z. Wang, H. Shi, and X. Wang. Videoinr: Learning video implicit neural representation for continuous space-time super-resolution. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2037–2047, 2022.

[13] P.-Z. Chiang, M.-S. Tsai, H.-Y. Tseng, W.-S. Lai, and W.-C. Chiu. Stylizing 3d scene via implicit representation and hypernetwork. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1475–1484, 2022.

[14] M. Chu, Y. Xie, J. Mayer, L. Leal-Taixé , and N. Thuerey. Learning temporal coherence via self-supervision for GAN-based video generation. *ACM Transactions on Graphics*, 39(4), aug 2020.

[15] J. S. Chung, A. Jamaludin, and A. Zisserman. You said that? *arXiv preprint arXiv:1705.02966*, 2017.

[16] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Lip reading sentences in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[17] J. S. Chung and A. Zisserman. Lip reading in the wild. In *Asian Conference on Computer Vision*, 2016.

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

[19] A. Clark, J. Donahue, and K. Simonyan. Efficient video generation on complex datasets. *ArXiv*, abs/1907.06571, 2019.

[20] D. Das and C. V. Jawahar. Adapting ocr with limited supervision. In X. Bai, D. Karatzas, and D. Lopresti, editors, *Document Analysis Systems*, pages 30–44, Cham, 2020. Springer International Publishing.

[21] Z. Ding, X.-Y. Liu, M. Yin, and L. Kong. Tgan: Deep tensor generative adversarial nets for large image generation. *arXiv preprint arXiv:1901.09953*, 2019.

[22] N. Dong and E. Xing. Domain adaption in one-shot learning. In *European Conference on Machine Learning and and Principles and Practice of Knowledge Discovery*, pages 573–588, 01 2019.

[23] A. Duarte, S. Palaskar, L. Ventura, D. Ghadiyaram, K. DeHaan, F. Metze, J. Torres, and X. Giro-i Nieto. How2sign: a large-scale multimodal dataset for continuous american sign language. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2735–2744, 2021.

[24] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12868–12878, 2020.

[25] D. Feng, S. Yang, S. Shan, and X. Chen. Learn an effective lip reading model without pains. *arXiv preprint arXiv:2011.07557*, 2020.

[26] B. Fernando, E. Gavves, M. José Oramas, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5378–5387, 2015.

[27] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan. Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. *Neurocomputing*, 321:321–331, Dec 2018.

[28] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks, 2016.

[29] S. Gao, F. Huang, and H. Huang. Model compression via hyper-structure network. *OpenReview preprint*, 2021. `https://openreview.net/forum?id=Oc-Aedbjq0`.

[30] K. Genova, F. Cole, D. Vlasic, A. Sarna, W. T. Freeman, and T. Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7154–7164, 2019.

[31] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[32] S. N. Gowda, M. Rohrbach, and L. Sevilla-Lara. Smart frame selection for action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1451–1459, 2021.

[33] Y. Guo, K. Chen, S. Liang, Y.-J. Liu, H. Bao, and J. Zhang. Ad-nerf: Audio driven neural radiance fields for talking head synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5784–5794, 2021.

[34] P. K. Gyawali, S. Ghimire, P. Bajracharya, Z. Li, and L. Wang. Semi-supervised medical image classification with global latent mixing. In A. L. Martel, P. Abolmaesumi, D. Stoyanov, D. Mateus, M. A. Zuluaga, S. K. Zhou, D. Racoceanu, and L. Joskowicz, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pages 604–613, Cham, 2020. Springer International Publishing.

[35] D. Ha, A. Dai, and Q. V. Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[36] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[37] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022.

[38] C. Hopwood. Concentration fatigue, Feb 2021. `https://deafblind.org.uk/concentration-fatigue/`.

[39] W.-N. Hsu, Y. Zhang, and J. Glass. Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation, 2017.

[40] P. K R, 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, page 1428–1436, New York, NY, USA, 2019. Association for Computing Machinery.

[41] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. *ArXiv*, abs/2006.06676, 2020.

[42] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[43] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[44] J. Kim, S. Kim, J. Kong, and S. Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Advances in Neural Information Processing Systems*, 33:8067–8077, 2020.

[45] D. E. King. Dlib-ml: A machine learning toolkit. *J. Mach. Learn. Res.*, 10:1755–1758, Dec. 2009.

[46] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[47] R. Korbmacher and A. Tordeux. Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[48] A. Lamb, E. S. Saveliev, Y. Li, S. Tschiatschek, C. Longden, S. Woodhead, J. M. Hernández-Lobato, R. E. Turner, P. Cameron, and C. Zhang. Contextual hypernetworks for novel feature adaptation. *ArXiv*, abs/2104.05860, 2021.

[49] G. Le Moing, J. Ponce, and C. Schmid. Ccvs: context-aware controllable video synthesis. *Advances in Neural Information Processing Systems*, 34:14042–14055, 2021.

[50] D.-H. Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[51] J. Liang, J. Cao, Y. Fan, K. Zhang, R. Ranjan, Y. Li, R. Timofte, and L. Van Gool. Vrt: A video restoration transformer. *arXiv preprint arXiv:2201.12288*, 2022.

[52] P. Luc, A. Clark, S. Dieleman, D. d. L. Casas, Y. Doron, A. Cassirer, and K. Simonyan. Transformation-based adversarial video prediction on large-scale data. *arXiv preprint arXiv:2003.04035*, 2020.

[53] A. Makhzani and B. J. Frey. Pixelgan autoencoders. *Advances in Neural Information Processing Systems*, 30, 2017.

[54] T. Makkonen, H. Ruottinen, R. Puhto, M. Helminen, and J. Palmio. Speech deterioration in amyotrophic lateral sclerosis (ALS) after manifestation of bulbar symptoms. *International Journal of Language & Communication Disorders*, 53(2):385–392, Nov. 2017.

[55] B. Martinez, P. Ma, S. Petridis, and M. Pantic. Lipreading using temporal convolutional networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6319–6323, 2020.

[56] P. Masrori and P. V. Damme. Amyotrophic lateral sclerosis: a clinical review. *European Journal of Neurology*, 27(10):1918–1929, July 2020.

[57] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.

[58] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[59] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto. Few-shot adversarial domain adaptation. *Advances in neural information processing systems*, 30, 2017.

[60] moviepy. Creating and exporting video clips, 2020. `https://zulko.github.io/moviepy/getting_started/videoclips.html`.

[61] N. Müller, Y. Siddiqui, L. Porzi, S. R. Bulò, P. Kontschieder, and M. Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. *arXiv preprint arXiv:2212.01206*, 2022.

[62] ndcs.org.uk. Reasons for tiredness in deaf children: Reasons for fatigue. *National Deaf Children's Society*, 2020. `https://www.ndcs.org.uk/information-and-support/parenting-and-family-life/parenting-a-deaf-child/tiredness-in-deaf-children`.

[63] P. Nguyen, T. Tran, K. Le, S. Gupta, S. Rana, D. Nguyen, T. Nguyen, S. Ryan, and S. Venkatesh. Fast conditional network compression using bayesian hypernetworks. In *ECML/PKDD*, 2022.

[64] S. Niklaus and F. Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446, 2020.

[65] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE international conference on computer vision*, pages 261–270, 2017.

[66] J. Park, C. Lee, and C.-S. Kim. Asymmetric bilateral motion estimation for video frame interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14539–14548, 2021.

[67] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.

[68] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021.

[69] K. R. Prajwal, R. Mukhopadhyay, V. Namboodiri, and C. V. Jawahar. Learning individual speaking styles for accurate lip to speech synthesis, 2020.

[70] K. R. Prajwal, R. Mukhopadhyay, V. P. Namboodiri, and C. Jawahar. A lip sync expert is all you need for speech to lip generation in the wild. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 484–492, 2020.

[71] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[72] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[73] A. Razavi, A. Van den Oord, and O. Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[74] 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 preprint arXiv:2006.04558*, 2020.

[75] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[76] L. D. Rosenblum, R. M. Miller, and K. Sanchez. Lip-read me now, hear me better later. *Psychological Science*, 18(5):392–396, May 2007.

[77] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[78] M. S. Sajjadi, R. Vemulapalli, and M. Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6626–6634, 2018.

[79] B. Sen, A. Agarwal, V. P. Namboodiri, and C. Jawahar. Inr-v: A continuous representation space for video-based generative tasks. *Transactions on Machine Learning Research*, 2022.

[80] B. Sen, A. Agarwal, G. Singh, S. Sridhar, M. Krishna, et al. Scarp: 3d shape completion in arbitrary poses for improved grasping. *IEEE International Conference on Robotics and Automation*, 2023.

[81] M. Sendera, M. Przewiezlikowski, K. Karanowski, M. Zieba, J. Tabor, and P. Spurek. Hypershot: Few-shot learning by kernel hypernetworks. *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2468–2477, 2022.

[82] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe. First order motion model for image animation. *Advances in Neural Information Processing Systems*, 32, 2019.

[83] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[84] V. Sitzmann, E. Chan, R. Tucker, N. Snavely, and G. Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020.

[85] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

[86] V. Sitzmann, S. Rezchikov, B. Freeman, J. Tenenbaum, and F. Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34:19313–19325, 2021.

[87] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019.

[88] V. Sitzmann, M. Zollhoefer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[89] I. Skorokhodov, S. Ignatyev, and M. Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10753–10764, 2021.

[90] I. Skorokhodov, S. Tulyakov, and M. Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3626–3636, 2022.

[91] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, 2015.

[92] J. Suarez. Language modeling with recurrent highway hypernetworks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[93] N. Tajbakhsh, Y. Hu, J. Cao, X. Yan, Y. Xiao, Y. Lu, J. Liang, D. Terzopoulos, and X. Ding. Surrogate supervision for medical image analysis: Effective deep learning from limited quantities of labeled data, 2019.

[94] R. Takahashi, A. Hashimoto, M. Sonogashira, and M. Iiyama. Partially-shared variational auto-encoders for unsupervised domain adaptation with target shift. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 1–17. Springer, 2020.

[95] Y. Tian, J. Ren, M. Chai, K. Olszewski, X. Peng, D. N. Metaxas, and S. Tulyakov. A good image generator is what you need for high-resolution video synthesis. In *International Conference on Learning Representations*, 2021.

[96] A. Torfi, S. M. Iranmanesh, N. Nasrabadi, and J. Dawson. 3d convolutional neural networks for cross audio-visual matching recognition. *IEEE Access*, 5:22081–22091, 2017.

[97] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1526–1535, 2017.

[98] Z. Turan. Deaf children with additional disabilities: Description and research, 01 2015.

[99] R. Tzaban, R. Mokady, R. Gal, A. H. Bermano, and D. Cohen-Or. Stitch it in time: Gan-based facial editing of real videos. *SIGGRAPH Asia 2022 Conference Papers*, 2022.

[100] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[101] J. Von Oswald, C. Henning, J. Sacramento, and B. F. Grewe. Continual learning with hypernetworks. *arXiv preprint arXiv:1906.00695*, 2019.

[102] J. Walker, A. Razavi, and A. v. d. Oord. Predicting video with vqvae. *arXiv preprint arXiv:2103.01950*, 2021.

[103] L. Wang, B. Huang, Z. Zhao, Z. Tong, Y. He, Y. Wang, Y. Wang, and Y. Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. *arXiv preprint arXiv:2303.16727*, 2023.

[104] X. Wang, K. C. K. Chan, K. Yu, C. Dong, and C. C. Loy. Edvr: Video restoration with enhanced deformable convolutional networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1954–1963, 2019.

[105] W. Wu, X. Wang, H. Luo, J. Wang, Y. Yang, and W. Ouyang. Bidirectional cross-modal knowledge exploration for video recognition with pre-trained vision-language models. *arXiv preprint arXiv:2301.00182*, 2022.

[106] W. Xiong, W. Luo, L. Ma, W. Liu, and J. Luo. Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2364–2373, 2018.

[107] K. Xu, D. Li, N. Cassimatis, and X. Wang. Lcanet: End-to-end lipreading with cascaded attention-ctc, 2018.

[108] W. Yan, Y. Zhang, P. Abbeel, and A. Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

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

[110] S. Yao, R. Zhong, Y. Yan, G. Zhai, and X. Yang. Dfa-nerf: personalized talking head generation via disentangled face attributes neural rendering. *arXiv preprint arXiv:2201.00791*, 2022.

[111] S. Yu, J. Tack, S. Mo, H. Kim, J. Kim, J.-W. Ha, and J. Shin. Generating videos with dynamics-aware implicit generative adversarial networks. *ArXiv*, abs/2202.10571, 2022.

[112] S. Zarei, K. Carr, L. Reiley, K. Diaz, O. Guerra, P. Altamirano, W. Pagani, D. Lodin, G. Orozco, and A. Chinea. A comprehensive review of amyotrophic lateral sclerosis. *Surgical Neurology International*, 6(1):171, 2015.

[113] Y. Zhang, C. Wang, and D. Tao. Video frame interpolation without temporal priors. *Advances in Neural Information Processing Systems*, 33:13308–13318, 2020.