TalkLoRA: Low-Rank Adaptation for Speech-Driven Animation

Jack Saunders[1, 2]
https://jsaunders909.github.io/
Vinay P Namboodiri[1]
https://vinaypn.github.io/

[1] University of Bath
Bath, UK
[2] Deepreel Ltd
124 Goswell Rd,
London, UK

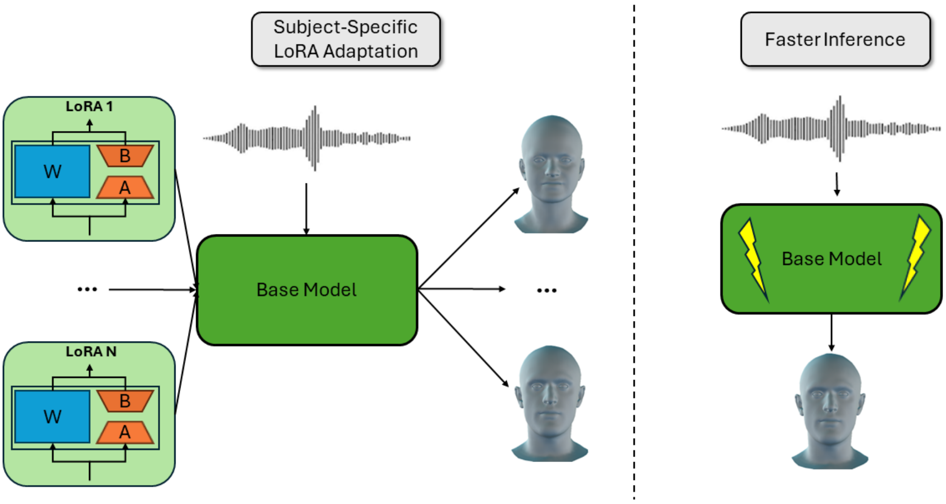

Figure 1: We present TalkLoRA, a method for improving any transformer-based speech-driven animation model. We use Low Rank Adaptation to effectively and efficiently adapt to new identities and chunking to improve inference speed, with no loss of quality.

Abstract

Speech-driven facial animation is important for many applications including TV, film, video games, telecommunication and AR/VR. Recently, transformers have been shown to be extremely effective for this task. However, we identify two issues with the existing transformer-based models. Firstly, they are difficult to adapt to new personalised speaking styles and secondly, they are slow to run for long sentences due to the quadratic complexity of the transformer. We propose TalkLoRA to address both of these issues. TalkLoRA uses Low-Rank Adaptation to effectively and efficiently adapt to new speaking styles, even with limited data. It does this by training an adaptor with a small number of parameters for each subject. We also utilise a chunking strategy to reduce the complexity of the underlying transformer, allowing for long sentences at inference time. TalkLoRA can be applied to any transformer-based speech-driven animation method. We perform extensive experiments to show that TalkLoRA archives state-of-the-art style adaptation and that it allows for an order-of-complexity reduction in inference times without sacrificing quality. We also investigate and provide insights into the hyperparameter selection for LoRA fine-tuning of speech-driven facial animation models.

1 Introduction

3D Digital humans are very pervasive across many forms of media. TV, video games, movies, telepresence and marketing, make extensive use of them. Furthermore, they are a critical component in 2D Talking Head generation [22, 27, 28, 30, 31]. As social creatures, humans pay a lot of attention to each other's faces [10]. This makes us very good at discerning details relating to the face. Of particular importance is the motion of the face. With even small errors in this facial animation, the end result enters into what is known as the 'uncanny valley', an unsettling phenomenon that prevents acceptance of the digital human [23].

A traditional way of obtaining high-quality facial animations is for skilled artists to manually pose the face into keyframes, and to interpolate between these. This process is, however, very slow and expensive, making it feasible only for the most important facial animation. Another popular method is the use of performance capture which involves attempting to match an actor's performance to a 3D face rig (e.g. [8, 13, 25]). This process is, again, costly. For high-volume facial animations, speech-driven facial animation is popular. Furthermore, certain applications require *procedurally generated* facial animations that are synthesised on the fly from audio or text input. For example text-driven avatars, or conversational agents.

A recent surge of methods has excelled at producing high-quality facial animations from speech signals [2, 7, 9, 12, 26, 29]. These methods, however, mostly require a significant amount of data per person. To date, only a small number of works have considered how best to adapt speech-driven animation systems to new identities [29].

Furthermore, a large number of these works are transformer-based. This means to generate a facial expression at time t, they look at all tokens in the range $[0, t-1]$, making them $O(N^2)$ in complexity where N is the length of the animation. This is unsuitable for long animations. It is also a counterintuitive approach, conditioned on audio, why should the facial expression at any given time depend on the expression from, for example, ten seconds ago? The use of full-length context is therefore unnecessary and detrimental.

We propose **TalkLoRA** to address both these problems. TalkLoRA uses Low-Rank Adaptation to adapt existing pre-trained transformer speech-driven models. LoRA allows for more efficient and effective fine-tuning in both limited and plentiful data scenarios. Furthermore, TalkLoRA utilizes a fixed context window in the transformer, reducing the computational complexity to a constant level and allowing much longer sequences to be processed without a drop in quality.

It is important to note that while we base TalkLoRA on two specific models, it could equally be applied to any transformer-based model, including emotional models [9] or even speech-driven implicit models [2, 15].

In summary, our contributions are:

- A methodology for effective and efficient adaptation of any transformer-based speech-driven animation model (Table 1) that runs with an order-of-complexity reduction in inference time (Section 3.3).

- A LoRA-based adaptor for fitting speech-driven animation models to new subjects, together with an analysis on the improvement over state-of-the-art (Table 1), the effect of dataset size (Table 1) and the impact of hyperparameter selection (Sec 5.3).

- The use of a limited context window in the transformer through chunking, that reduces the inference time. We also include analysis of this component, showing that it does not reduce quality, and an investigation into the required context length (Figure 2 & 4).

2 Related Work

2.1 Speech-Driven 3D Facial Animation

Speech-driven 3D facial animation involves producing facial animations, as sequences of either blendshapes or vertices from an input audio sequence. Very early methods use rule-based approaches based on visemes (the visual counterpart to phonemes) [1, 6, 11, 19]. These methods work fairly well for high-volume digital humans, but are very simplistic and cannot faithfully produce high-quality animations. Since the advent of deep learning, many data-driven methods have been proposed with greater capabilities [7, 9, 12, 20, 29]. Karras et. al. [20] train an auto-regressive CNN-based model to predict vertices from audio segments. This method only works on the actor that it is trained for. VOCA [7] improves upon this by adding one network shared across several identities, disambiguation is achieved through a one-hot encoding of identity, and novel identities can be added using a linear interpolation of existing ones. Faceformer [12] works similarly with one-hot encodings and achieves better results with the use of transformers. Several works apply this transformer-based, one-hot identity model to subdomains, for example, emotional speech-driven animation [9] or speech-driven animation of implicit models [2, 14]. There are two significant drawbacks to this. First, the one-hot encoding severely limits the ability of the model to adapt to new identities. Secondly, the transformer decoders use the entire audio segment for generation with the self-attention mechanism. This quickly becomes infeasible for long sentences. Imitator [29] looks to address the first of these problems by fine-tuning certain layers in their model. While this does help, it is both slow and sub-optimal. Our method, by comparison, is able to effectively and efficiently adapt to new identities using LoRA [17] and can handle longer sequences using a fixed-context window.

2.2 Transfer Learning

Transfer Learning is a very general problem in the field of machine learning. The objective is to take a large model that has been trained on one dataset and to adapt it for use on another, often smaller, dataset. Transfer learning has been most prominently investigated in the context of NLP (e.g. [4, 16, 24]) and vision [5, 18, 32]. Recent work has also investigated the topic of speech recognition [33], which is closely related to our work on speech-driven animation.

A naive way of achieving transfer learning to resume the training of a model using the pre-trained weights and optimising all parameters. However, this has several issues. For one, this is a slow process and it is very memory intensive. Furthermore, if the new dataset is small, training a large model in this way makes it very vulnerable to overfitting. This has given rise to the field of parameter-efficient fine-tuning (PEFT), where only a small number of parameters are updated. Some methods focus on tuning a subset of network layers, however, this restricts the level of abstraction at which the tuning operates. More recently, methods have addressed this problem by optimising more layers using lower dimensional decomposition with small neural networks [16]. Unfortunately, this introduces a large overhead at inference time. Therefore Low-Rank Adaptors (LoRA) were proposed [17]. LoRA works by decomposing weight matrices into two lower-rank matrices whose product is the same shape as the weight matrix. This can be added to the pre-trained weights and used with no overhead, yet still allowing for parameter-efficient finetuning.

In the context of speech-driven facial animation, the goal of transfer learning is to adapt pre-trained models to new identities. Often, there is little person-specific data available on which to perform adaptation. This means that it is essential to avoid overfitting. Imitator [29] achieves this by fine-tuning a style code and the final layer. This works well but is highly restrictive, effectively serving as an interpolation of existing styles with an additional linear transformation. We propose using LoRA [17] which allows us to adapt the model in a non-linear and more flexible way, leading to better results.

3 Method

Our objective is to improve any existing transformer-based speech-driven animation system. We therefore propose components that do not make any additional assumptions about the model. We first describe two existing state-of-the-art speech-driven animation models (Section 3.1) to show the differences, our model works on both. We then discuss our use of Low-Rank Adaptors to allow us to adapt existing models to new identities (Sec 3.2). Finally, we improve the inference speed of existing models for long sequences using a chunking strategy (Sec 3.3). We show these steps in Figure 1.

3.1 Architectures

The basic architecture of our model is determined by the base model we use for adaptation. For the case of our experiments this is either FaceFormer [12] or Imitator [29]. In either case, there are some significant similarities. Each model consists of three components, an audio encoder, a transformer decoder and an per-frame decoder.

Audio Encoder: For both Imitator and Faceformer the audio encoder is the same. Wav2Vec2 [3] is used due to its powerful ability as a feature extractor. Wav2Vec2 is trained for speech recognition on a large and diverse dataset. The choice of speech recognition as a task means that the output features of the model are person-agnostic, allowing for good generalisation of the speech-driven models to novel audio. The final layer is discarded and the outputs of the final hidden layer are taken as the audio features. The Wav2Vec2 model outputs features at 50Hz, these are linearly resampled to match the fps of the target animations, and a learnable linear layer converts these to the desired dimensionality.

Tranformer Decoder Both models also make use of a transformer decoder to consider temporal information. They each use cross-attention using the audio features from the audio encoder. The key difference is that Imitator's transformer is person-independent while FaceFormer's is not. FaceFormer encodes the vertices from the previous time step, adds a person-specific style code and then passes the result to a transformer. Whereas Imitator uses a predefined start token, which is the same for all subjects, and produces person-independent viseme tokens from the audio features. It is important to note that TalkLoRA does not require either of these choices over the other, both our adaptor and the chunking procedure can be applied to any transformer-based model.

Motion Decoder The final step is to produce the vertices from the transformer output. As FaceFormer includes style information in the transformer, only a single linear layer is used to project the transformer output onto the FLAME vertices. Imitator, on the other hand, introduces the person-specific style code here by adding it to the transformer output. Imitator then uses an MLP to predict vertices.

3.2 LoRA

In order to adapt a baseline model to a new subject we make use of Low-Rank Adaptors (LoRA) [17]. LoRA is a method for parameter-efficient fine-tuning originally proposed for large language models. Instead of training all the weights in a model, or some subset of the layers, LoRA adds an offset to the weight matrices of layers using rank decomposed matrices. Specifically, consider a weight matrix $W \in \mathbb{R}^{N \times M}$. LoRA models the adaption as:

$$W' = W + \Delta W = W + AB^T \tag{1}$$

Where $A \in \mathbb{R}^{N \times r}$ and $B \in \mathbb{R}^{M \times r}$, with $r << \min(N, M)$. The idea is that as new datasets have low intrinsic dimensionality, the required weight updates will also.

We have several considerations for which components of the network we should apply LoRA to. First, for the audio encoders, we consider it counterintuitive to apply LoRA. The audio encoder is powerful precisely because it is highly generalised. This allows it to encode audio from any person (for example from a TTS system or voice actor) into a common feature space. While adapting the audio encoder may improve the performance for audio coming from the subject, it is likely to overfit and hinder performance from other audio sources. We therefore do not consider applying LoRA to this part of the network.

For the decoders, however, we want the model to adapt to just a single identity. That is, we want the decoded motion to take on the style of the speaker. This gives us two candidates for LoRA application. The transformer decoder and/or the motion decoder. We show which is preferable in Section 5.3.

LoRA introduces a small set of parameters for fine-tuning. This allows for a tradeoff between the representational power of the model and regularization. That is, how likely it is to overfit. Given a dataset with a lower intrinsic dimensionality, we would expect to use a smaller value of r. In Section 5.3, we determine the optimal value of r for our datasets empirically.

3.3 Limiting Context Window for Speedup

Faceformer [12] and Imitator [29], as well as other transformer-based speech-driven animation methods [2, 9], all use transformers with unlimited context lengths. To compute what the lips should look like at time $t = 60s$, they will look at the entire history of the lip motions from time $t = 0s$. This is a holdover from the original use of transformers in NLP, where words may hold influence over long time periods. For facial animation, however, this assumption does not make sense. The audio is used as input, meaning there is no need for the transformer to learn an internal language model, only the audio in a short context window, and the last few frames of the facial expression should matter.

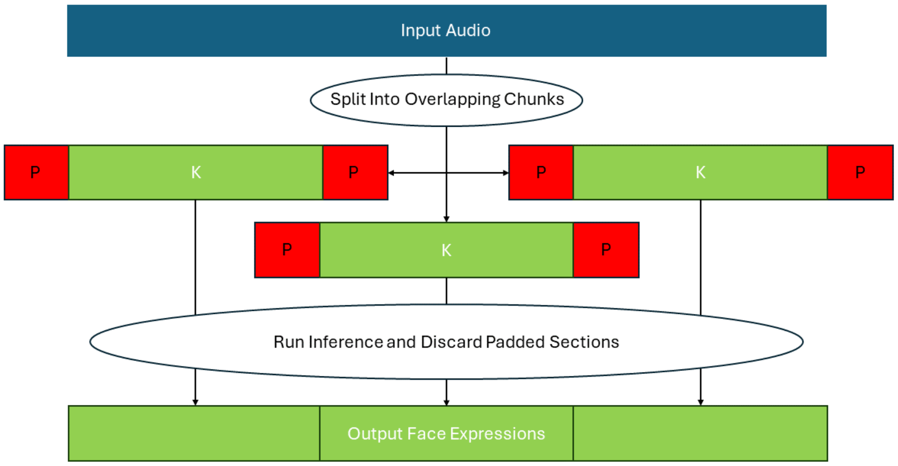
Input Audio
Split Into Overlapping Chunks
P
K
P
P
K
P
P
K
P
Run Inference and Discard Padded Sections
Output Face Expressions

Figure 2: The chunking process is used to limit the context window of the transformers. We split incoming audio into overlapping chunks of size K+2P and process these in parallel. The padding is then removed and the results concatenated.

We therefore look to reduce the context window of the transformers. **Note that we are not retraining the base models**. Therefore, we need to alter the architecture at inference time in such a way that it only sees a small context window without degrading performance.

To do this, we apply chunking. We split the input audio of length T into overlapping padded chunks of fixed size $K+2P$. Here P is the size of the padding. The idea is that the padding “warms up" the transformer, providing good context for inference. The predictions in this padded region are likely to be poor, so these are simply discarded. An example of this is shown in Figure 2.

Transformers notoriously have quadratic complexity in the size of the input sequence ($\mathcal{O}(N^2)$. By using a constant and fairly small value $K < N$ the complexity is simply a linear $\mathcal{O}(NK^2) << \mathcal{O}(N^2)$ ($\frac{N}{K}$ sequences of length K). This is an order of complexity reduction. We determine the values of K and P required for optimal performance in Section 5.4.

4 Implementation Details

For each base model, we train using the procedures set out in the respective papers. We use these base models as a baseline. For person-specific adaptation we use the loss weights outlined in Imitator [29] with $\lambda_{rec} = 1.0$ and $\lambda_{vel} = 10.0$. We use the AdamW optimiser with a learning rate of 0.001. Unless otherwise specified, we use a LoRA rank of 4 and LoRA alpha value of 8. We find empirically that the model converges after just 50 epochs, so we train our adaptor for this many.

Table 1: Comparison of adaptation strategies, we compare our method to Imitator's described adaptation [29], and FaceFormer with adaptation of the style code [12]. We show that our adaptation strategy improves both models. We also include the baseline models for reference. The best results for each dataset size are shown in **bold**, the best adaptation for each base model and dataset combination is underlined.

	Subject A				Subject B			
	$L_2^{Face}\downarrow$	$L_2^{Lip}\downarrow$	Lip-Max ↓	Time (mm:ss)↓	$L_2^{Face}\downarrow$	$L_2^{Lip}\downarrow$	Lip-Max↓	Time (mm:ss)↓
Faceformer (Base)	0.933	0.181	6.097	-	0.710	0.098	3.694	-
Imitator (Base)	0.902	0.157	5.609	-	0.736	0.141	4.781	-
Imitator (1 Sentence)	0.911	0.136	5.021	07:29	1.065	0.091	3.462	10:43
Imitator + Ours (1 Sentence)	0.900	**0.135**	**4.975**	**00:30**	1.059	**0.090**	**3.440**	**00:42**
Faceformer + Style (1 Sentence)	0.881	0.153	5.397	03:46	0.715	0.118	4.160	04:27
Faceformer + Ours (1 Sentence)	**0.876**	0.153	5.395	00:40	**0.698**	0.114	4.074	00:50
Imitator (3 Sentence)	0.904	0.136	4.946	24:23	1.114	**0.094**	3.515	34:57
Imitator + Ours (3 Sentence)	0.903	**0.134**	**4.884**	**01:55**	1.113	**0.094**	**3.157**	**02:17**
Faceformer + Style (3 Sentences)	0.875	0.151	5.366	12:13	0.707	0.114	4.025	13:50
Faceformer + Ours (3 Sentences)	**0.852**	0.144	5.248	02:14	**0.811**	0.97	3.790	02:29
Imitator (5 Sentence)	0.878	0.131	4.780	50:35	1.070	0.093	**3.481**	61:03
Imitator + Ours (5 Sentence)	0.877	**0.128**	**4.657**	**03:19**	1.103	**0.092**	3.490	**04:07**
Faceformer + Style (5 Sentences)	0.859	0.149	5.359	20:45	**0.703**	0.110	3.936	23:40
Faceformer + Ours (5 Sentences)	**0.838**	0.139	5.137	03:49	0.830	0.098	3.863	04:19
Imitator (10 Sentence)	0.867	0.129	4.659	93:39	0.865	0.086	**3.244**	107:22
Imitator + Ours (10 Sentence)	0.888	**0.126**	**4.547**	**06:33**	0.783	**0.086**	3.252	**07:45**
Faceformer + Style (10 Sentences)	0.848	0.147	5.340	39:29	**0.697**	0.098	3.726	43:32
Faceformer + Ours (10 Sentences)	**0.816**	0.133	4.960	07:18	0.705	0.094	3.731	07:53
Imitator (30 Sentence)	0.697	0.125	4.470	259:19	0.635	0.083	**3.058**	316:54
Imitator + Ours (30 Sentence)	**0.695**	**0.124**	**4.367**	**17:07**	**0.595**	**0.082**	3.130	**21:00**
Faceformer + Style (30 Sentences)	0.790	0.139	5.08	117:36	0.612	0.091	3.631	129:37
Faceformer + Ours (30 Sentences)	0.739	0.125	4.677	20:43	0.611	0.90	3.604	23:23

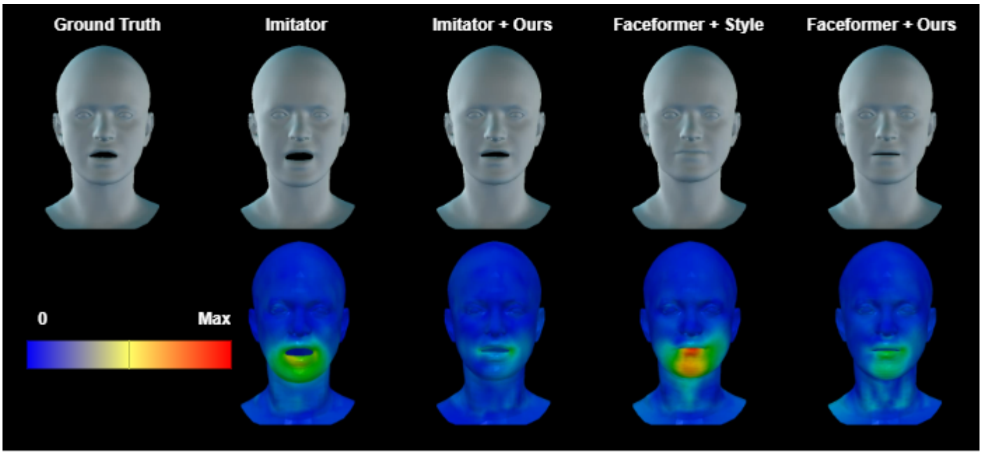

Figure 3: Qualitative results of our method showing a sentence on one of the train subjects. We compare our adaptation method on both Imitator and Faceformer and show improvements over their respective adaptation methods.

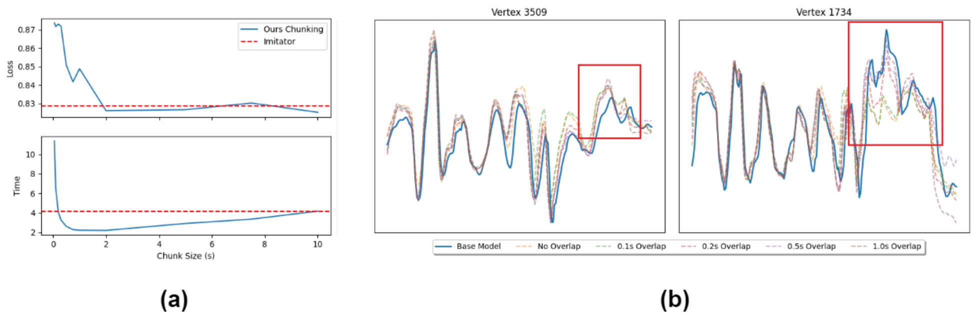

Figure 4: Graphs for determining the values of chunk size (K) and padding size (P) for chunking. **(a)** shows the effect of the size (K) of chunks compared vs the inference time and the validation loss for a validation subject. Too small a chunk takes a long time due to the padding, and also has poor quality. We find a sweet spot for time savings and quality at around 1-3 second chunks. **(b)** shows the effect of overlap size (P) in chunking. We show the y-postion of two lip vertices over time. It can be seen that a 0.2s overlap in chunking allows for outputs that are close to the un-chunked base model.

5 Results

5.1 Data

For all of our experiments we use VOCASET [7]. VOCASET consists of the meshes from 12 subjects each speaking 40 sentences at 60fps. VOCASET is split into 8 train subjects, 2 validation subjects and 2 test subjects. For our experiments, we train the base models on the 8 train subjects and use the 2 test subjects for our person-specific adaptions. We name these test subjects Subject A and Subject B. We further split the data for subjects A and B into train and test sets. We withhold the final 10 sentences as a test set and use various subsets of the remaining 30 for adaptation depending on the experiment.

5.2 Comparison to State-of-the-art

To date, only Imitatior [29] has attempted person-specific adaptation. We therefore compare our results primarily to this model. Specifically, we compare our method of LoRA fine-tuning to Imitator's method of fine-tuning the final layer and style code of the model for 300 epochs each. We show this for several different person-specific dataset sizes, ranging from 1 sentence (about 4 seconds) up to a maximum of 30 sentences (approximately 2 minutes). Faceformer is not designed to be adapted for new identities, however, this can be done to some degree by optimising the style code used in the model. We denote this as Faceformer + Style. Following previous work, we do this for 300 epochs. In addition to this, we also include baseline results from FaceFormer and Imitator without person-specific adaptation. To do this, we run inference using each of the 8 training styles from VOCASET and select the one with the best metrics. We show results for TalkLoRA using both Imitator and Faceformer as a base model, showing that it can be applied to any transformer-based speech-driven model.

To compare models, as is standard practice, we use the L_2 distance between vertices in the last 10 sentences of the test subjects. We separate this into a full-face metric L_2^{Face} using all the vertices and a lip-only metric L_2^{Lip} that uses only the lip vertices. Following MeshTalk [26] we also use the Lip-Max metric, which is defined as the mean of the maximum L_2 distance of any lip vertex across all frames. In addition, we also measure the time taken for each adaptation model. Specifically, we record the time to train each adaptation using an NVIDIA L4 GPU.

The results are shown in Table 1. Our method of adaptation achieves state-of-the-art results in the vast majority of training configurations and metrics, while being significantly faster to train. It consistently improves both Imitator and Faceformer, suggesting it can be applied to any transformer-based speech-driven animation model. The improvement is also seen visually in Figure 3.

5.3 LoRA Parameter Selection

LoRA [17] may be applied to any combination of layers in the network. Each base model is split into three components (see section 3.1). We find that application to the audio encoder is a bad idea, as this is specifically designed to be person-agnostic to allow for audio from any person to be used. We, therefore, only consider applying LoRA to the transformer decoder and the motion decoder. There is a single parameter that has significant influence over LoRA models, this is the rank r of the decomposed matrices.

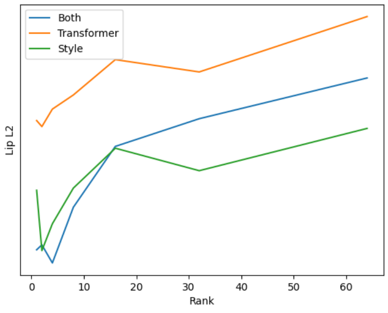

Figure 5: The effect of rank on lip L_2 loss across random training subsets. ≈ 4 yields the best results.

To determine the best value of rank r we design a short experiment. We use the following procedure: For a random test subject, we randomly select an integer value between 1 and 30 representing the number of sequences we will use for fine-tuning. We then randomly select this many sequences from the training set of the given subject. We then use set r to each of the values $\{1,2,4,8,16,32\}$ and compute the lip L_2 loss. We run this random sampling approach 30 times and take the average for each value of r. The results are shown in Figure 5. It can be seen that the optimal value for r is around 4 so this is the value we choose. Lower than this does not properly exploit the available data, while much greater means the model overfits. This suggests that the person-specific data in VOCASET has a low intrinsic dimensionality.

5.4 Effects of Chunking

We also design an experiment to test the efficacy of our chunking method on long audio sequences. To do this, while still using ground truth data for computing metrics, we create artificial long sentences. This is done simply by concatenating the ten test sentences from each VOCASET subject with one second of silence in between. When calculating metrics, we mask out these silent regions. We experiment with various values of chunk size K and padding P. We use the pre-trained base Imitator for this experiment.

We calculate the L_2 loss across the long sequences for various values of K and record the run time. The results of the chunking experiments are shown in Figure 4 (a). It can be seen that small chunk sizes cause much higher losses. This is because the transformer never gets adequate context. This effect diminishes at around 2-second long chunks. It can be seen that the run-time is optimal at 0.5-2 seconds. We therefore use a chunk size of 2 seconds for our model. For P we find that the effect on the loss is much less noticeable. However, errors occur around the cut points when P is too small. This can be seen in Figure 4 (b) in the red boxes. Small amounts of padding cause the first few frames to not have enough context leading to differences between the chunked model and the base model. We find that 0.2s of padding is enough to alleviate this without a significant increase in inference time.

6 Limitations and Future Work

While our work can adapt effectively to new identities and run with faster inference, there are still some drawbacks. First, even with properly calibrated values of K and P, chunking will still reduce the quality of models trained with long context. This means for short sentences it is often not worth chunking. In future, it may be worth investigating training models with this chunking. We hypothesise this may help prevent the model from overfitting to spurious temporal correlations in speech segments in the data. Another interesting line of research would be to include learnable weights that perform the fusing of the chunks.

While we were able to find a good set of parameters to train our LoRA models, we have only considered fixed parameters for all dataset sizes. It is likely that given more data, one may want to increase the LoRA rank as the risk of overfitting is reduced and intrinsic dimensionality increased. Exploring questions like this, as well as considering other methods of adaptation (e.g. controlnet [34] or BOFT [21]) is left as future work.

7 Conclusion

We have presented our work, **TalkLoRA** for adapting transformer-based, speech-driven animation models to new identities using LoRA. Our method is capable of adapting to new speakers better than existing state-of-the-art models, as we are better able to avoid overfitting to the low intrinsic dimensionality of 3D talking head datasets. TalkLoRA also improves inference speed through our chunking strategy. Our method is applicable to any speech-driven model, provided it uses transformers, making it easy for general adoption. Our method has applications in video games and film/TV using digital characters, as well as in photorealistic, 2D, audio-driven talking head animation.

8 Acknowledgments

We want to thank the team at DeepReel for providing computing resources and excellent discussions on this line of research. This work was supported in part by the UKRI grant EP/S023437/1.

References

[1] Facial animation based on context-dependent visemes. *Computers Graphics*, 30(6): 971–980, 2006. ISSN 0097-8493. doi: https://doi.org/10.1016/j.cag.2006.08.017.

[2] Shivangi Aneja, Justus Thies, Angela Dai, and Matthias Nießner. Facetalk: Audio-driven motion diffusion for neural parametric head models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[3] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[5] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022.

[6] D. Cosker, D. Marshall, P.L. Rosin, and Y. Hicks. Speech driven facial animation using a hidden markov coarticulation model. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 128–131 Vol.1, 2004. doi: 10.1109/ICPR.2004.1334024.

[7] Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael Black. Capture, learning, and synthesis of 3D speaking styles. *Computer Vision and Pattern Recognition (CVPR)*, pages 10101–10111, 2019.

[8] Radek Danecek, Michael J. Black, and Timo Bolkart. EMOCA: Emotion driven monocular face capture and animation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20311–20322, 2022.

[9] Radek Daněček, Kiran Chhatre, Shashank Tripathi, Yandong Wen, Michael Black, and Timo Bolkart. Emotional speech-driven animation with content-emotion disentanglement. ACM, December 2023. doi: 10.1145/3610548.3618183. URL https://emote.is.tue.mpg.de/index.html.

[10] Katharina Dobs, Isabelle Bülthoff, and Johannes Schultz. Use and usefulness of dynamic face stimuli for face perception studies-a review of behavioral findings and methodology. *Front. Psychol.*, 9:1355, August 2018.

[11] Pif Edwards, Chris Landreth, Eugene Fiume, and Karan Singh. Jali: an animator-centric viseme model for expressive lip synchronization. *ACM Trans. Graph.*, 35(4), jul 2016. ISSN 0730-0301. doi: 10.1145/2897824.2925984. URL https://doi.org/10.1145/2897824.2925984.

[12] Yingruo Fan, Zhaojiang Lin, Jun Saito, Wenping Wang, and Taku Komura. Faceformer: Speech-driven 3d facial animation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[13] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (ToG), Proc. SIGGRAPH*, 40(4):88:1–88:13, August 2021.

[14] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[15] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[16] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 09–15 Jun 2019.

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[18] Shibo Jie and Zhi-Hong Deng. Convolutional bypasses are better vision transformer adapters. *arXiv preprint arXiv:2207.07039*, 2022.

[19] G.A. Kalberer and L. Van Gool. Face animation based on observed 3d speech dynamics. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No.01TH8596)*, pages 20–251, 2001. doi: 10.1109/CA.2001.982373.

[20] Tero Karras, Timo Aila, Samuli Laine, Antti Herva, and Jaakko Lehtinen. Audio-driven facial animation by joint end-to-end learning of pose and emotion. *ACM Trans. Graph.*, 36(4):94:1–94:12, 2017.

[21] Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and Bernhard Schölkopf. Parameter-efficient orthogonal finetuning via butterfly factorization. In *ICLR*, 2024.

[22] Yifeng Ma, Suzhen Wang, Zhipeng Hu, Changjie Fan, Tangjie Lv, Yu Ding, Zhidong Deng, and XinYu. Styletalk: One-shot talking head generation with controllable speaking styles. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2023.

[23] Masahiro Mori. The uncanny valley. *IEEE Robotics & Automation Magazine*, 19(2): 98–100, 2012.

[24] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[25] George Retsinas, Panagiotis P. Filntisis, Radek Danecek, Victoria F. Abrevaya, Anastasios Roussos, Timo Bolkart, and Petros Maragos. 3d facial expressions through analysis-by-neural-synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[26] Alexander Richard, Michael Zollhoefer, Yandong Wen, Fernando de la Torre, and Yaser Sheikh. Meshtalk: 3d face animation from speech using cross-modality disentanglement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[27] Jack Saunders and Vinay Namboodiri. Dubbing for everyone: Data-efficient visual dubbing using neural rendering priors. *arxiv*, 2024.

[28] Jack Saunders and Vinay P. Namboodiri. Read avatars: Realistic emotion-controllable audio driven avatars. In *British Machine Vision Conference (BMVC)*, 2023.

[29] Balamurugan Thambiraja, Ikhsanul Habibie, Sadegh Aliakbarian, Darren Cosker, Christian Theobalt, and Justus Thies. Imitator: Personalized speech-driven 3d facial animation, 2022.

[30] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4): 1–12, 2019.

[31] Justus Thies, Mohamed Elgharib, Ayush Tewari, Christian Theobalt, and Matthias Nießner. Neural voice puppetry: Audio-driven facial reenactment. *ECCV 2020*, 2020.

[32] Yi Xin, Siqi Luo, Haodi Zhou, Junlong Du, Xiaohong Liu, Yue Fan, Qing Li, and Yuntao Du. Parameter-efficient fine-tuning for pre-trained vision models: A survey. *arXiv preprint arXiv:2402.02242*, 2024.

[33] Qiantong Xu, Alexei Baevski, and Michael Auli. Simple and effective zero-shot cross-lingual phoneme recognition. *ArXiv*, abs/2109.11680, 2021. URL `https://api.semanticscholar.org/CorpusID:237635125`.

[34] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models.