A Deep Learning Approach for Generalized Speech Animation

SARAH TAYLOR, University of East Anglia
TAEHWAN KIM, YISONG YUE, California Institute of Technology
MOSHE MAHLER, JAMES KRAHE, ANASTASIO GARCIA RODRIGUEZ, Disney Research
JESSICA HODGINS, Carnegie Mellon University
IAIN MATTHEWS, Disney Research

Fig. 1. A machine learning approach is used to learn a regression function mapping phoneme labels to speech animation. Our approach generates continuous, natural-looking speech animation for a reference face parameterization that can be retargeted to the face of any computer generated character.

We introduce a simple and effective deep learning approach to automatically generate natural looking speech animation that synchronizes to input speech. Our approach uses a sliding window predictor that learns arbitrary non-linear mappings from phoneme label input sequences to mouth movements in a way that accurately captures natural motion and visual coarticulation effects. Our deep learning approach enjoys several attractive properties: it runs in real-time, requires minimal parameter tuning, generalizes well to novel input speech sequences, is easily edited to create stylized and emotional speech, and is compatible with existing animation retargeting approaches. One important focus of our work is to develop an effective approach for speech animation that can be easily integrated into existing production pipelines. We provide a detailed description of our end-to-end approach, including machine learning design decisions. Generalized speech animation results are demonstrated over a wide range of animation clips on a variety of characters and voices, including singing and foreign language input. Our approach can also generate on-demand speech animation in real-time from user speech input.

CCS Concepts: • **Computing methodologies** → **Neural networks**; **Procedural animation**; **Motion processing**; *Real-time simulation*; *Visual analytics*;

Additional Key Words and Phrases: Speech Animation, Machine Learning.

ACM Reference format:
Sarah Taylor, Taehwan Kim, Yisong Yue, Moshe Mahler, James Krahe, Anastasio Garcia Rodriguez, Jessica Hodgins, and Iain Matthews. 2017. A Deep Learning Approach for Generalized Speech Animation. *ACM Trans. Graph.* 36, 4, Article 93 (July 2017), 11 pages.
DOI: 10.1145/3072959.3073699

0730-0301/2017/7-ART93 $15.00
DOI: 10.1145/3072959.3073699

1 INTRODUCTION

Speech animation is an important and time-consuming aspect of generating realistic character animation. Broadly speaking, speech animation is the task of moving the facial features of a graphics (or robotic) model to synchronize lip motion with the spoken audio and give the impression of speech production. As humans, we are all experts on faces, and poor speech animation can be distracting, unpleasant, and confusing. For example, mismatch between visual and audio speech can sometimes change what the viewer believes they heard [McGurk and MacDonald 1976]. High-fidelity speech animation is crucial for effective character animation.

Conventional speech animation approaches currently used in movie and video game production typically tend toward one of two extremes. At one end, large budget productions often employ either performance capture or a large team of professional animators, which is costly and difficult to reproduce at scale. For example, there is no production level approach that can cost-effectively generate high quality speech animation across multiple languages. At the other extreme, low-budget, high-volume productions may use simplified libraries of viseme lip shapes to quickly generate lower-quality speech animation.

More recently, there has been increasing interest in developing data-driven methods for automated speech animation to bridge these two extremes, for example [De Martino et al. 2006; Edwards et al. 2016; Taylor et al. 2012]. However, previous work requires predefining a limited set of viseme shapes that must then be blended together. Simple blending functions limit the complexity of the dynamics of visual speech that can be modeled. Instead, we aim to leverage modern machine learning methods that can directly learn the complex dynamics of visual speech from data.

We propose a deep learning approach for automated speech animation that provides a cost-effective means to generate high-fidelity speech animation at scale. For example, we generate realistic speech

animation on a visual effects production level face models with over 100 degrees of freedom. A central focus of our work is to develop an effective speech animation approach that may be seamlessly integrated into existing production pipelines.

Our approach is a continuous deep learning sliding window predictor, inspired by [Kim et al. 2015]. The sliding window approach means our predictor is able to represent a complex non-linear regression between the input phonetic description and output video representation of continuous speech that naturally includes context and coarticulation effects. Our results demonstrate the improvement of using a neural network deep learning approach over the decision tree approach in [Kim et al. 2015]. The use of overlapping sliding windows more directly focuses the learning on capturing localized context and coarticulation effects and is better suited to predicting speech animation than conventional sequence learning approaches, such as recurrent neural networks and LSTMs [Hochreiter and Schmidhuber 1997].

One of the main challenges using machine learning is properly defining the learning task (i.e., what are the inputs/outputs and training set) in a way that is useful for the desired end goal. Our goal is an approach that makes it easy for animators to incorporate high-fidelity speech animation onto any rig, for any speaker, and in a way that is easy to edit and stylize. We define our machine learning task as learning to generate high-fidelity animations of neutral speech from a single reference speaker. By focusing on a reference face and neutral speech, we can cost-effectively collect a comprehensive dataset that fully captures the complexity of speech animation. The large training data set allows us to reliably learn the fine-grained dynamics of speech motion using modern machine learning approaches. In contrast to previous work on procedural speech animation [De Martino et al. 2006; Edwards et al. 2016; Taylor et al. 2012], our approach directly learns natural coarticulation effects from data. Defining our input as text (as phoneme labels) means we learn a speaker independent mapping of phonetic context to speech animation. We require only off-the-shelf speech recognition software to automatically convert any spoken audio, from any speaker, into the corresponding phonetic description. Our automatic speech animation therefore generalizes to any input speaker, for any style of speech, and can even approximate other languages. In summary, our contributions include:

- A definition of a machine learning task for automatically generating speech animation that may be integrated into existing pipelines. In particular, we define the task to be speaker independent and generate animation that can be retargeted to any animation rig.
- A deep learning approach that directly learns a non-linear mapping from the phonetic representation to visual speech in a way that naturally includes localized context and coarticulation effects, and can generate high-fidelity speech animation.
- An empirical evaluation comparing against strong baselines. We include both quantitative and qualitative evaluations demonstrating the improved performance of our approach.
- A demonstration of the ease with which our approach can be deployed. We provide a wide range of animation clips on a variety of characters and voices, including examples of singing and foreign languages, as well as a demonstration of on-demand speech animation from user input audio.

2 RELATED WORK

Production quality speech animation is often created manually by a skilled animator, or by retargeting motion capture of an actor. The advantage of hand animation is that the artist can precisely style and time the animation, but it is extremely costly and time consuming to produce. The main alternative to hand animation is performance-driven animation using facial motion capture of an actor's face [Beeler et al. 2011; Cao et al. 2015, 2013; Fyffe et al. 2014; Huang et al. 2011; Li et al. 2013; Weise et al. 2011; Weng et al. 2014; Zhang et al. 2004]. Performance-driven animation requires an actor to perform all shots, and may generate animation parameters that are complex and time consuming for an animator to edit (e.g. all parameters are keyed on every frame). In contrast, our goal is to automatically generate production quality animated speech for any style of character given only audio speech as input.

Prior work on automated speech animation can be categorized into three broad classes: interpolating single-frame visual units, concatenating segments of existing visual data, and sampling generative statistical models.

Single-frame visual unit interpolation involves key-framing static target poses in a sequence and interpolating between them to generate intermediate animation frames [Cohen et al. 1994; Ezzat et al. 2002]. One benefit of this approach is that only a small number of shapes (e.g. one per phoneme) need to be defined. However, the realism of the animation is highly dependent on how well the interpolation captures both visual coarticulation and dynamics. One can either hand-craft such interpolation functions [Cohen et al. 1994] which are time consuming to refine and ad-hoc, or employ a data-driven approach based on statistics of visual speech parameters [Ezzat et al. 2002]. These approaches make strong assumptions regarding the static nature of the interpolant and do not address context-dependent coarticulation. This issue is partially considered in [Ezzat et al. 2002], which uses covariance matrices to define how much a particular lip shape is allowed to deform, but the covariance matrices themselves are fixed which can lead to unnatural deformations. In contrast, our method generates smooth animation without making strong assumptions about the distribution of visual speech.

Sample-based synthesis stitches together short sequences of existing speech data that correspond either to fixed-length (e.g. words or phonemes) [Bregler et al. 1997; Cao et al. 2005; Liu and Ostermann 2012; Mattheyses et al. 2013; Theobald and Matthews 2012; Xu et al. 2013] or variable length [Cosatto and Graf 2000; Edwards et al. 2016; Ma et al. 2006; Taylor et al. 2012] units. Unit selection typically involves minimizing a cost function based on the phonetic context and the smoothness. One limitation is that the context typically considers only the phoneme *identity*, and so a large amount of data is required to ensure sufficient coverage over all contexts. Sample-based animation is also limited in that it can only output units seen in the training data. In contrast, our approach is significantly more data efficient, and is able to learn complex mappings from phonetic context to speech animation directly from training data.

A more flexible approach is to use a generative statistical model, such as GMMs [Luo et al. 2014], switching linear dynamical systems

[Englebienne et al. 2007], switching shared Gaussian process dynamical models [Deena et al. 2010], recurrent neural networks [Fan et al. 2015], or hidden Markov models (HMMs) and their variants [Anderson et al. 2013; Brand 1999; Fu et al. 2005; Govokhina et al. 2006; Schabus et al. 2011; Wang et al. 2012; Xie and Liu 2007]. During training of a HMM-based synthesiser, context-dependent decision trees cluster motion data and combine states with similar distributions to account for sparsity of the phonetic contexts in the training set. Synthesis involves first traversing the decision trees to select appropriate models and then generating the maximum likelihood parameters from the models. Models are typically trained using static features augmented with derivatives to constrain the smoothness of the HMM output by ensuring that the velocity and acceleration of the generated static features match the maximum likelihood velocity and acceleration. However, HMM-based synthesis may appear under articulated because of the limited number of states and the smoothness constraints on the parameters [Merritt and King 2013].

Within the context of previous work, our sliding window deep learning approach addresses all the above limitations. We employ a complex non-linear predictor to automatically learn the important phonetic properties for co-articulation and context. Our approach directly learns to predict a sequence of outputs (i.e., an animation sequence), and so we can directly model local dynamics of visual speech while making minimal assumptions. As such, our approach avoids the need for ad-hoc interpolation by directly learning a mapping of arbitrary phonetic (sub-)sequences to animation (sub-)sequences.

Recently, deep learning has been successfully applied to problems in the domains of computer vision [Krizhevsky et al. 2012], natural language processing [Collobert et al. 2011], and speech recognition [Graves and Jaitly 2014]. It has also been very effective in sequence generation problems, including: image-caption generation [Xu et al. 2015], machine translation [Bahdanau et al. 2014], and speech synthesis [van den Oord et al. 2016].

From a machine learning perspective, our setting is an instance sequence-to-sequence prediction [Fan et al. 2015; Kim et al. 2015; Sutskever et al. 2014]. There are two high level approaches to making sequence-to-sequence predictions, sliding window models [Kim et al. 2015] versus recurrently defined models [Fan et al. 2015; Sutskever et al. 2014]. The former emphasizes correctly modeling the local context and ignores long-range dependences, whereas the latter emphasizes capturing long-range dependences using a low-dimensional state that gets dynamically updated as the model processes the input sequence. We employ a sliding window architecture, inspired by [Kim et al. 2015], which better fits the requirements of speech animation. We discuss this further in Section 5.1.

3 APPROACH OVERVIEW

We make the following requirements for our speech animation approach in order for it to be easily integrated into existing production pipelines:

(1) *High Fidelity.* The generated animations should accurately reflect complex speaking patterns present in visible speech motion, such as co-articulation effects.

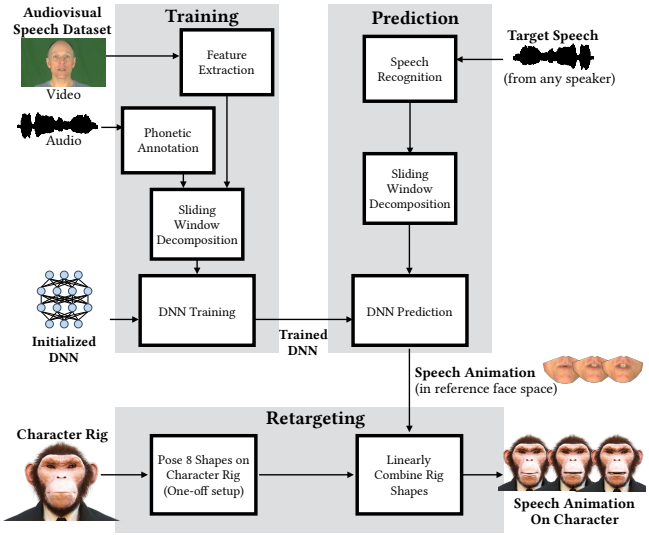

Fig. 2. An overview of our system. See Section 4 for details for dataset, Section 5 for details of training and prediction, and Section 6 for details of the retargeting.

(2) *Speaker Independent.* The system should not depend on the specific speaker, speaking style, or even the language being spoken. Rather, it should be able to generate speech animation synchronized to any input speech.
(3) *Retargetable and Editable.* The system should be able to retarget the generated animations to any facial rig. Furthermore, the retargeted animations should be easy to edit and stylize by animators.
(4) *Fast.* The system should be able to generate animations quickly, ideally in real-time.

Figure 2 depicts an overview of our approach. To satisfy high fidelity (Requirement 1), we take a data-driven approach to accurately capture the complex structure of natural speech animation. To keep the learning problem compact, we train a predictor to generate high-fidelity speech animation for a single reference face model. By learning for a single face, we can control for speaker-specific effects, and focus the learning on capturing the nuances of speech animation. One practical benefit of this approach is that we can cost-effectively collect an appropriate training set (i.e., for just a single speaker) that comprehensively captures a broad range of speech patterns. This approach also satisfies being retargetable and editable (Requirement 3), since it is straightforward to retarget high-quality speech animation from a single reference face to any production rig, as well as import the animation into editing software such as Autodesk Maya. We discuss in Section 5 specific design decisions of our machine learning approach in order to learn to generate high fidelity animations in real-time (Requirement 4).

To satisfy being speaker independent (Requirement 2), we train our predictor to map input text (as a phoneme transcript) to speech animation, rather than mapping directly from audio features. After training, we can use any off-the-shelf speech recognition software

to convert spoken audio into a phonetic transcript. We describe in Section 5.2 our extended input phoneme representation.

More formally, let $\mathbf{x}$ denote an input phoneme sequence that we wish to animate. Our goal is to construct a predictor $h(\mathbf{x}) := \mathbf{y}$ that can predict a realistic animation sequence $\mathbf{y}$ for any input $\mathbf{x}$. Note that $\mathbf{y}$ corresponds to the specific reference face model. A training set of $(\mathbf{x}, \mathbf{y})$ pairs collected from the reference speaker is used for training (see Section 4). In general, h can be complex and learn complex non-linear mappings from $\mathbf{x}$ to $\mathbf{y}$ (see Section 5).

After h is learned, one can perform a *one-time pre-computation* of any retargeting function from the reference face model to any character CG model of any rig parameterization. Afterwards, we can automatically and quickly make predictions to the retargeted face for any input phoneme sequence. In summary, our pipeline is described as follows:

Training:

(1) Record audio and video of a reference speaker reciting a collection of phonetically-balanced sentences.
(2) Track and parameterize the face of the speaker to create the reference face animation model $\mathbf{y}$.
(3) Transcribe the audio into phoneme label sequences $\mathbf{x}$.
(4) Train a predictor $h(\mathbf{x})$ to map from $\mathbf{x}$ to the corresponding animation parameters $\mathbf{y}$.
(5) Pre-compute a retargeting function to a character CG model (e.g., using existing retargeting techniques).

Animation:

(1) Transcribe input audio into a phoneme sequence $\mathbf{x}$ (e.g., via off-the-shelf speech recognition software). The input can be from any language and any speaker.
(2) Use $h(\mathbf{x})$ to predict the animation parameters $\mathbf{y}$ of the reference face model corresponding to $\mathbf{x}$.
(3) Retarget $\mathbf{y}$ from the reference face model to a target CG model (can be repeated for multiple target rigs).

Note that Steps 1-4 during Training are performed only once for all use cases. Step 5 needs to be pre-computed once for each new target face model. Given a transcribed audio sequence (Step 1 during Animation), our approach can then automatically generate the accompanying visual speech animation in real-time.

Section 4 describes the training data. Section 5 describes our deep learning sliding window approach. Section 6 describes retargeting approaches. For speech-to-text transcription, we used either off-the-shelf software such as the Penn Phonetics Lab Forced Aligner [Yuan and Liberman 2008] that is based on the HTK toolbox [Young et al. 2006], or manual transcription in special cases.

4 AUDIO-VISUAL SPEECH TRAINING DATA

For our training set, we use the existing KB-2k dataset from [Taylor et al. 2012]. KB-2k is a large audio-visual dataset containing a single actor reciting 2543 phonetically diverse TIMIT [Garofolo et al. 1993] sentences in neutral tone. The face in the video is front facing and captured at 1080p29.97. All sentences in the dataset have been manually annotated in the Arpabet phonetic code.

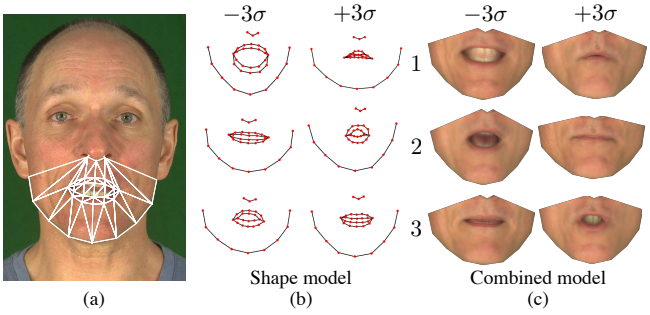

Fig. 3. (a) The 34 vertices of the AAM shape component. (b) The first three modes of variation (highest energy) in the AAM shape component shown at ±3 standard deviations about the mean. (c) The first three modes of variation of the combined AAM model shown at ±3 standard deviations about the mean.

The TIMIT corpus was designed as a phonetically diverse speech training dataset and achieves high coverage of the relevant coarticulation effects while minimizing the amount of speech recording required.

4.1 Reference Face Parameterization

The video data of KB-2k is compactly parameterized using the coefficients of linear models of lower facial shape and appearance that an Active Appearance Model (AAM) optimizes to track the video frames [Cootes et al. 2001; Matthews and Baker 2004]. The shape component represents $N = 34$ vertices of the lower face and jaw, $\mathbf{s} = \{u_1, v_1, u_2, v_2, ..., u_N, v_N\}^T$, as the linear model, $\mathbf{s} = \mathbf{s}_0 + \sum_{i=1}^{m} \mathbf{s}_i p_i$, using $m = 16$ modes to capture 99% of shape variation, see Figure 3(b). The mean shape is $\mathbf{s}_0$, each $\mathbf{s}_i$ is a shape basis vector, and the shape parameters are p_i.

The appearance model is separated into $k = 2$ non-overlapping regions $A^k(\mathbf{u})$, where $\mathbf{u}$ represents the set of 40 thousand (u, v) pixel coordinates sampled at $\mathbf{s}_0$. Using two regions allows the pixels within the inner mouth area (when visible) to vary independently of the remaining face pixels of the lips and jaw, $A^k(\mathbf{u}) = A_0^k(\mathbf{u}) + \sum_{i=1}^{n} \lambda_i^k A_i^k(\mathbf{u})$. The mean appearance of each region is A_0^k, the basis vectors A_i^k, and appearance parameters λ_i^k.

The reference face representation, $\mathbf{y}$, is a $q = 104$ dimensional description of both deformation and intensity changes of a human face during speech described as a linear projection of concatenated shape and appearance parameters. An appropriate weight, w, balances the energy difference of intensity and shape parameters [Cootes et al. 2001],

$$\begin{pmatrix} w\mathbf{p} \\ \boldsymbol{\lambda}^1 \\ \boldsymbol{\lambda}^2 \end{pmatrix} = U\mathbf{y}V^T = \sum_{i=1}^{q} \mathbf{j}_i y_i. \quad (1)$$

The first three modes of joint variation, $\mathbf{j}_i$, are shown in Figure 3(c). Complete details are included in [Taylor et al. 2012].

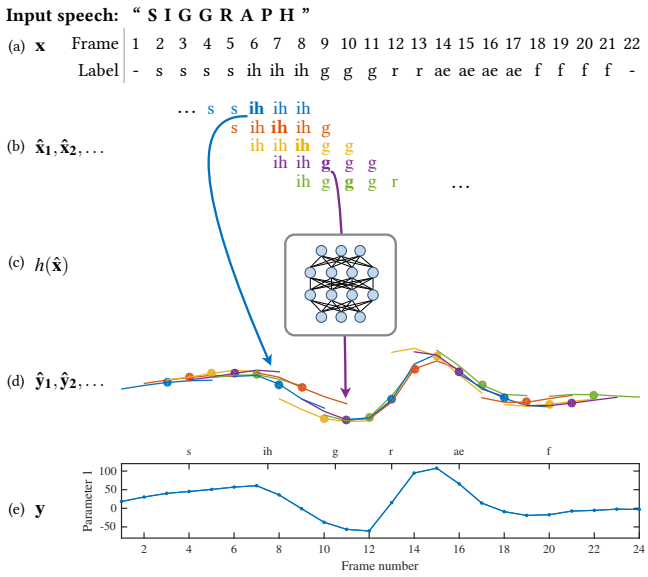

Fig. 4. Depicting our deep learning sliding window regression pipeline. We start with a frame-by-frame sequence of phonemes $\mathbf{x}$ as input (a). We convert $\mathbf{x}$ into a sequence of overlapping fixed-length inputs $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots)$ (b). We apply our learned predictor to predict on each $\hat{\mathbf{x}}_i$ (c), which results in a sequence of overlapping fixed-length outputs $(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \ldots)$ (d). We blend $(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \ldots)$ by averaging frame-wise to arrive at our final output $\mathbf{y}$ (e). Note the center frame of $\hat{\mathbf{y}}_i$ is highlighted, but all predicted values contribute to $\mathbf{y}$. Only the first predicted parameter value is shown for clarity.

5 DEEP LEARNING SLIDING WINDOW REGRESSION

Our sliding window neural network deep learning approach is inspired by [Kim et al. 2015], and is motivated by the following assumptions.

Assumption 1. *Coarticulation effects can exhibit a wide range of context-dependent curvature along the temporal domain. For example, the curvature of the first AAM parameter, Figure 4(e), can vary smoothly or sharply depending on the local phonetic context, Figure 4(a).*

Assumption 2. *Coarticulation effects are localized, and do not exhibit very long range dependences. For example, how one articulates the end of "prediction" is effectively the same as how one articulates the end of "construction", and does not depend (too much) on the beginning of either word.*

These assumptions motivate the main inductive bias in our learning approach, which is to train a *sliding window regressor* that learns to predict arbitrary fixed-length subsequences of animation. Figure 4 depicts our prediction pipeline, which can be summarized as:

(1) Decompose the input phonetic sequence $\mathbf{x}$ into a sequence of overlapping fixed-length inputs $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_T)$ of window size K_x (Figure 4(b)).
(2) For each $\hat{\mathbf{x}}_j$, predict using h, resulting in a sequence of overlapping fixed-length outputs $(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \ldots, \hat{\mathbf{y}}_T)$, each of window size K_y (Figure 4(c) and Figure 4(d)).
(3) Construct the final animation sequence $\mathbf{y}$ by blending together $(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \ldots, \hat{\mathbf{y}}_T)$ using the frame-wise mean (Figure 4(e)).

Since the mapping from phonetic subsequences to animation subsequences can be very complex, we instantiate h using a deep neural network. Our learning objective is minimizing square loss between the ground truth fixed-length subsequence and its corresponding prediction outputs among training data.

5.1 Deep Learning Details & Discussion

Deep learning approaches have become popular due to their ability to learn expressive representations over raw input features, which can lead to dramatic improvements in accuracy over using handcrafted features [Krizhevsky et al. 2012].

For our experiments, we use a fully connected feed forward neural network with a (sliding window) input layer connected to three fully connected hidden layers and a final output layer. There are 3000 hidden units per hidden layer, each using a hyperbolic tangent transfer function. We employ standard mini-batch stochastic gradient descent for training, with mini-batch size of 100. To counteract overfitting, we use dropout [Srivastava et al. 2014] with 50% probability. The final output layer is standard multi-linear regression trained to minimize the squared loss. One can train this model using any off-the-shelf deep learning platform.[1]

As mentioned earlier, the key property of our deep learning sliding window approach is that it can jointly predict for multiple frames simultaneously, which is directly motivated by the assumption that we should focus on capturing local temporal curvature in visual speech. One can equivalently view our sliding window predictor as a variant of a convolutional deep learning architecture.

In contrast, many recent deep learning approaches to sequence-to-sequence prediction use recurrent neural networks (and their memory-based extensions) [Fan et al. 2015; Sutskever et al. 2014], and model such dependencies indirectly by propagating information from frame to frame via hidden unit activations and, in the case of LSTMs, a state vector. While RNNs and LSTMs have the capacity to capture complex temporal curvature, their inductive bias is not necessarily aligned with our modeling assumptions, thus potentially requiring a large amount of training data before being able to reliably learn a good predictor. Instead, we focus the learning on capturing neighborhoods of context and coarticulation effects. We show in our experiments that the sliding window architecture dramatically outperforms LSTMs for visual speech animation.

Our approach has two tuning parameters, K_x and K_y. The input window length K_x must be large enough to capture the salient coarticulation effects, and the output window length K_y must be large enough to capture the salient local curvature of $\mathbf{y}$. For example, making K_x too small will not allow the model to disambiguate between two plausible coarticulations (due to the disambiguating phoneme lying outside the input window), and having K_y be too small can lead to noisy predictions. However, the larger that K_x and K_y are, the more training data is required to learn an accurate model since the intrinsic complexity of the model class (and thus risk of overfitting to a finite training set) increases with K_x and K_y.

[1] We used Keras (http://keras.io/) with Theano [Bastien et al. 2012]

- Does phone /s/ span L input frames of the subsequence starting from the k-th frame? (position, identification and length of span)
- Is the phone at k-th input frame a nasal consonant? (attribute)
- Are the phones at k-th and k+1-th input frames in a specific cluster of consonant-vowel pairs? (transition category)

Fig. 5. Example linguistically motivated indicator features used to augment the phoneme label input features.

We find that K_x and K_y are straightforward to tune, in part due to how quickly our model trains. From our experiments, we find $K_x = 11$ and $K_y = 5$ give the best results on our training and test sets.

5.2 Feature Representation

The final major design decision is the choice of feature representation. The most basic representation is simply a concatenated feature vector of phoneme identity indicator variables per input frame. Because our dataset contains 41 phonemes, this would result in a $41 \times K_x$ dimensional input feature vector to represent each input subsequence $\hat{\mathbf{x}}$. We call this the *raw feature representation.*

We also incorporated a linguistically motivated feature representation. These are all indicator features that correspond to whether a certain condition is satified by the input subsequence $\hat{\mathbf{x}}$. We procedurally generate three groups of features:

- **Phoneme identification spanning specific locations**. Every feature in this group corresponds to an indicator function of whether a specific phone spans a specific set of frames. E.g., "Does the phone /s/ span frames j through k of the input subsequence?"
- **Phoneme attribute category at a specific location**. Every feature in this group corresponds to an indicator function of whether a phone belonging to a specific category at a specific frame location. E.g., "Is the phone at frame j of the input a nasal consonant?"
- **Phoneme transitions at specific locations**. Every feature in this group corresponds to an indicator function of whether two adjacent frames correspond to a specific type of phoneme transition. E.g., "Are the phones at k-th and k + 1-th input frames in a specific cluster of consonant-vowel pairs?"

Figure 5 shows some example queries. In our experiments, we found that using linguistically-motivated features offered a small improvement over using just the raw features. The supplementary material contains a full expansion of all the linguistic features.

6 RIG-SPACE RETARGETING

To generalize to a new output face model the predicted animation must be retargeted. The AAM reference face representation described in Section 4.1 captures both shape and appearance changes (e.g. teeth and tongue visibility) during speech and any potentially complex and content-dependent retargeting function may be used

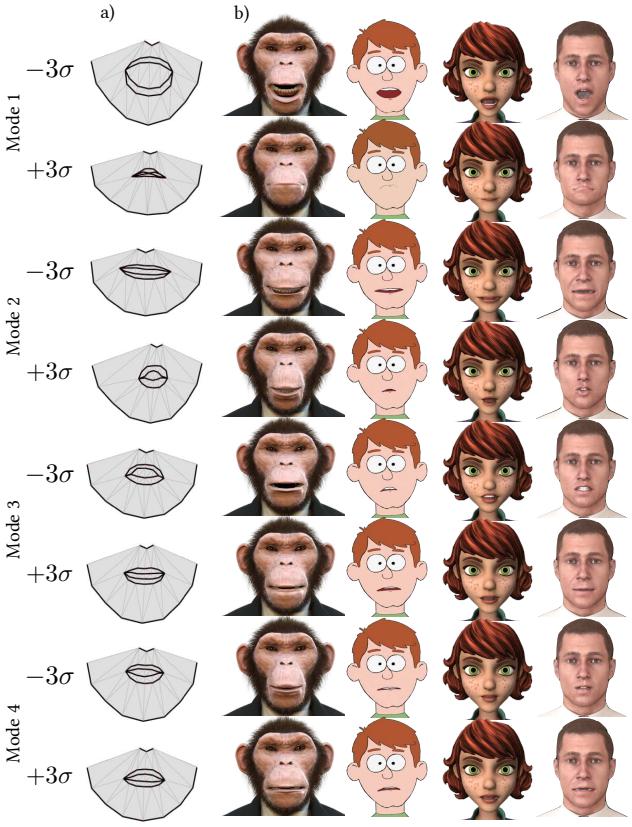

Fig. 6. a) Four modes of the reference shape model at $\pm 3\sigma$ from the mean create eight speech retargeting shapes. b) Corresponding poses transferred to a variety of face rigs by an artist.

to compute animation parameters for any rig implementation and character style.

Retargeting approaches that are of particular interest are those that can be pre-computed once by exploiting the known subspace of facial motion captured by the AAM representation. To accomplish this, the retargeting function must be well-defined over the entire range of poses that the reference face model can take. One effective approach is to use piece-wise linear retargeting where a small set of poses is manually mapped from the reference face model to the target face model. However, we note that any other retargeting approach may be used.

Our implementation pre-computes a retargeting function that spans the animation space of the neural network by manually posing a subset of the shape bases, $\mathbf{s}_i$, of the reference AAM representation and the mean shape, $\mathbf{s}_0$, on a target character. We use the first four shape modes for retargeting as these modes describe the most significant motion (91% energy) of the lower face and are interpretable by an animator.

To better represent non-linear behavior on the target rig we pose the output character at both +3 and −3 standard deviations from the mean, resulting in a set of eight poses, $\mathbf{s}_1^{-3}, \mathbf{s}_1^{+3}, \ldots, \mathbf{s}_4^{-3}, \mathbf{s}_4^{+3}$, where $\mathbf{s}_k^u = \mathbf{s}_0 + \mathbf{s}_k * u\sqrt{p_k}$ is relative to the mean pose, $\mathbf{s}_0$.

Figure 6 depicts an example retargeting process. For each of eight retargeting poses of the reference face, we create a one-time corresponding pose on each of the target rigs. We find that it is straightforward to pose these shapes manually, largely due to the fact that the basis shapes in the reference face are easy to interpret. For example, the first mode corresponds to how open the mouth is.

The rig parameters corresponding to the eight poses (effectively rig eigenvectors) are stored, giving $\mathbf{R} = \{\mathbf{r}_1^{-3}, \mathbf{r}_1^{+3}, \ldots, \mathbf{r}_4^{-3}, \mathbf{r}_4^{+3}\}$, relative to the mean pose $\mathbf{r}_0$. Subsequently predicted speech animation from the neural network can be directly transferred to the target rig by forming linear combinations of columns of $\mathbf{R}$ (i.e. rig-space interpolation). The 8-dimensional weight vector, $\mathbf{w}$, that determines the contribution of each pose is calculated by:

$$w_k^u = \max(\frac{\hat{p}_k}{u\sqrt{p_k}}, 0) \tag{2}$$

where $\hat{p}_k$ is the shape component of the neural network prediction and $u \in \{-3, +3\}$ dependent on whether the pose is associated with a negative or positive deviation from the mean. To retarget the predicted pose to a character, the rig parameters are combined as follows:

$$\mathbf{R}_t = (\mathbf{R} - \mathbf{r}_0)\mathbf{w} + \mathbf{r}_0 \tag{3}$$

The initial character setup is only performed once for each new character and is independent of how the rig is implemented (for example, blend-shapes, deformer based, etc.). Afterwards the animation pipeline is fully automatic. Examples of animation created using this rig-space retargeting approach are shown in the supplementary video. Rig-space retargeting is a simple pre-computable approach that captures the energy of speech articulation and yields consistently high quality animation. For well rigged characters it is easy for an animator to edit the resulting neutral speech animation, for example to overlay an emotional expression.

Other retargeting approaches are possible, and by design, independent of our speech animation prediction approach. Mesh deformation transfer [Sumner and Popović 2004] may be used to automate retargeting of reference shapes for rig-space deformation for example. Deformation transfer could also be used per-frame to transfer prediction animation to an un-rigged character mesh.

7 RESULTS

For visual inspection we include frames of example predicted speech animations. Please refer to the supplementary video for animation results.

Figure 7 shows how well our neural network model performs in predicting the speech animation of the original reference speaker. The input is one of the held-out sentences of the reference speaker. The resulting predicted speech animation can be directly compared to the (unseen) original video. We see that our approach is able to accurately capture the salient lip and jaw movements. In general, our approach tends to slightly under articulate compared to the original video[2] – however this may be compensated for by scaling up the motion during retargeting if required (we do not).

Figure 8 shows the full sequence of intermediate animations within the prediction pipeline. The first row shows the input speaker (who is not the reference speaker used for training). The second row shows the generated speech animation on the reference face model, and the final rows show the animation retargeted to the example face rigs.

Figure 9 shows neutral speech animation to a target rig with expression stylization added as a post-process by an animator. It is straightforward to import our speech animations into standard animation editing software such as Maya to create edited and stylized final animations.

[2]This is common to all machine learning approaches due to the need for regularization to prevent overfitting and enable generalizing to new inputs.

8 EVALUATION

We present an empirical analysis evaluating our approach using both quantitative and subjective measures against several strong baselines. We test on not only the held-out test sentences from the KB-2k training dataset, but also on completely novel speech from different speakers. Traditionally, machine learning approaches are evaluated on test examples drawn from the same distribution as the training set. However, testing on novel speakers is a much stronger test of generalizability, and is required for production quality speech animation. Because we do not have ground truth, we evaluate that setting solely via subjective evaluation (i.e., a user preference study).

8.1 Baselines

We compare against a variety of state-of-the-art baselines selected based on their performance and availability, or ease of implementation.

HMM-based Synthesis. The current state-of-the-art appoach is the (HTS) HMM-based synthesizer [Zen et al. 2007]. We trained this model using the same reference face parameters $\mathbf{y}$ as our approach. The HMM synthesizer uses context-dependent decision tree clustering [Odell 1995] to account for the sparseness of (quinphone) contexts in the training data by tying states with similar properties. The query set used in clustering is a subset of the indicator features used by our approach (Section 5.2). There are 749 queries which relate to the identity of the phonemes forming the context, and their place and manner of articulation (e.g., vowels, consonants, voiced, voiceless, nasal, etc.) The clustering criterion is the minimum description length (MDL) and each cluster must contain no fewer than 50 observations, which produces 11893 leaf nodes. We use typical left-to-right phone models with five emitting states and a single mixture component per state [Zen et al. 2007].

Dynamic Viseme Animation. Dynamic visemes were proposed as a data-derived visual speech unit in contrast to traditional visemes. Dynamic visemes are defined as speech-related *movements* of the face, rather than static poses. They are identified by segmenting the reference face parameters $\mathbf{y}$ into sequences of non-overlapping, visually salient short gestures which are then clustered. Each cluster represents visually similar lip motions that map to many strings of acoustic phonemes, each of variable length. In [Taylor et al. 2012] animation is predicted using dynamic programming to find the best match. The best dynamic viseme sequence is evaluated by minimizing a cost function which accounts for the probability of producing the phoneme sequence, the smoothness of the resulting animation, and for variable speaking rate. We use the implementation described in [Taylor et al. 2012].

Fig. 7. Comparison of held-out video of the reference speaker compared with AAM reference model rendered predictions. Predicted mouth regions are rendered onto the original face for visual comparison.

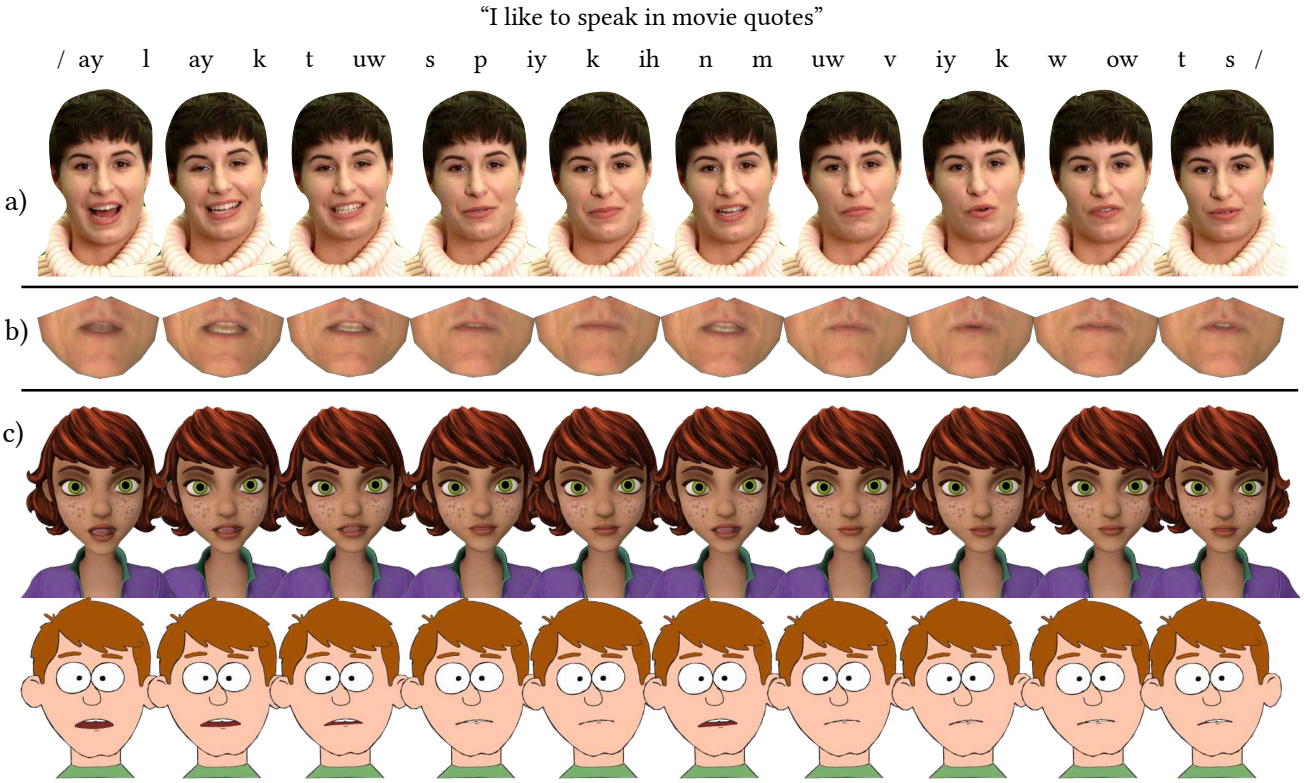

Fig. 8. Animation is transferred from the shape component of the AAM to CG characters using rig-space retargeting. (a) Reference video of the input speech (unseen speaker). (b) Visualization of the predicted animation as AAM. (c) The corresponding rig-space retargeted animation on a selection of face rigs.

Long Short-Term Memory Networks. LSTMs are a memory-based extension of recurrent neural networks, and were recently applied to learning photorealistic speech animation [Fan et al. 2015], which demonstrated some modest improvements over basic HMMs using a small dataset. We follow the basic setup of [Fan et al. 2015], and trained an LSTM network [Bastien et al. 2012] on the KB-2k dataset. We use three hidden layers, a fully-connected layer, and two LSTM layers. We experimented with 100 to 3000 hidden units for each layer, finding 500 achieves the best performance. Mini-batch size was 10, and to prevent overfitting we use dropout with 50% probability [Srivastava et al. 2014].

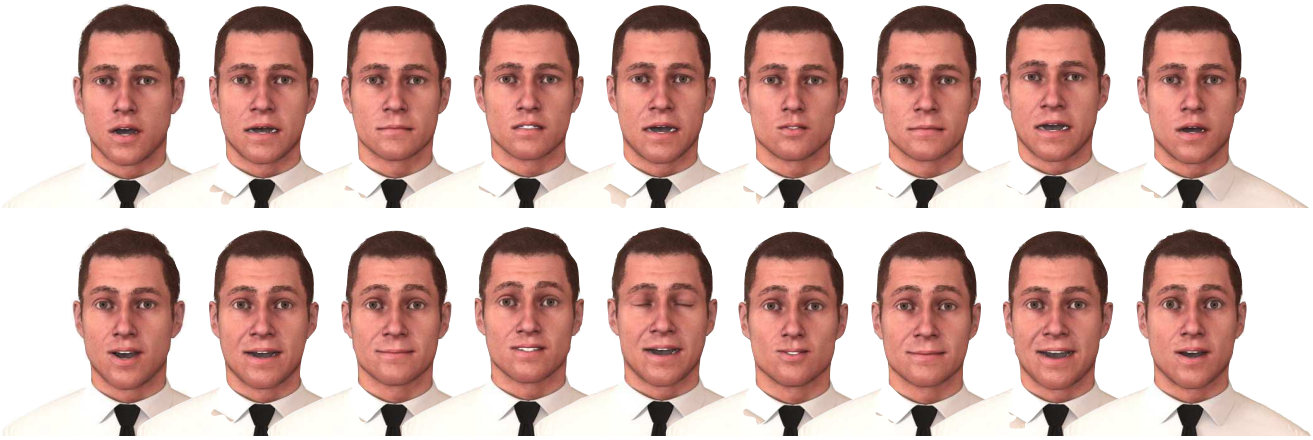

Fig. 9. Expression and stylization can be added to the predicted speech animation using standard animation techniques. (Top row) Frames of neutral speech animation generated using our approach for the sentence "I'll finally be the hero I've always dreamed of being". (Bottom row) The same neutral speech animation with expression and upper facial motion added by an artist.

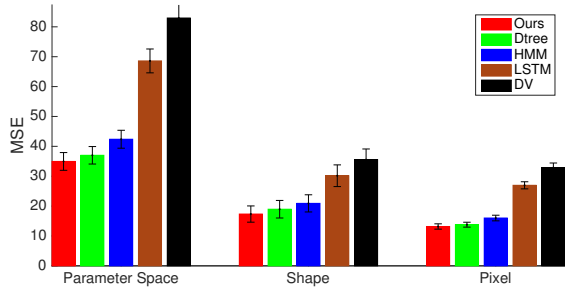

Fig. 10. Showing the mean square error of the KB-2k held out test sentences in the AAM parameter space, the predicted mesh vertex locations (shape), and appearance pixel intensities. We see that our approach consistently achieves the lowest mean squared error.

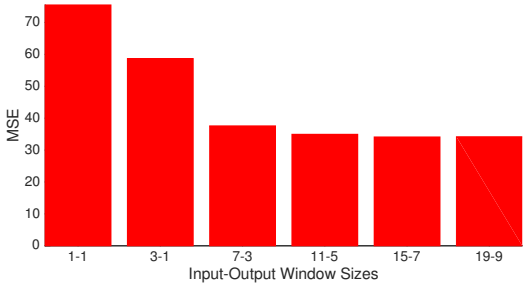

Fig. 11. Showing the mean square error of our approach as we vary the sliding window input-output sizes (K_x and K_y). We see that performance flattens as we increase the window sizes, indicating that there is little to be gained from modeling very long-range coarticulation effects.

Decision Tree Regression. Decision trees remain amongst the best performing learning approaches [Caruana and Niculescu-Mizil 2006] and make minimal distributional assumptions on the training data (e.g., no smoothness assumption). We use the sliding window decision tree implementation described in [Kim et al. 2015] with $K_x = 11$ and $K_y = 5$ and set the minimum leaf size to 10.

8.2 Benchmark Evaluation

In our benchmark evaluation, we evaluate all approaches on the fifty KB-2k held out test sentences. Because we have the ground truth for this data, we evaluate using squared loss of the various approaches. Figure 10 shows the results when measuring squared error in the reference AAM model parameter space, in the predicted shape vertex positions, and in predicted appearance pixel intensities. Decision tree regression is denoted "Dtree", and dynamic visemes is denoted "DV". We see that our approach consistently achieves the lowest squared error. We also see that LSTMs perform significantly worse on our data, which agrees with our intuition as discussed in Section 5.1. The most competitive baselines are the decision tree and HMM-based approaches, which still perform noticeably poorer.[3] These results suggest that our sliding window neural network approach achieves state-of-the-art performance in visual speech animation. Of course, squared error is not perfectly correlated with perceived quality, and modest differences in squared error may not be indicative of which approach produces the best speech animation. To address perceptual issues, Section 8.3 shows user study results.

Figure 11 shows the comparison of our approach as we vary the sliding window input/output sizes (K_x and K_y). We see that the performance converges as we increase the window sizes, indicating that there is little to be gained from modeling very long-range coarticulation effects.

In terms of computational cost, our approach evaluates predictions at ~1000 video frames per second. Training the model takes just a couple of hours on an Nvidia Tesla K80 GPU.

[3] Additional results and detailed analysis are included in the supplemental material.

Table 1. Showing user study results for the fifty KB-2k held out test sentences. For each test sentence, we ran a side-by-side comparison between two methods, and collected 25 pairwise judgments per comparison. A method wins the comparison if it receives the majority of the pairwise judgments for that test sentence. All results except comparison with ground truth AAM are statistically significant with 95% confidence.

Ours vs	AAM	HMM	DV	LSTM	Dtree
W / L	27 / 23	39 / 11	50 / 0	50 / 0	38 / 12

Table 2. Showing user study results for the 24 novel speaker test sentences. The setup is the same as Table 1. All results are statistically significant with 95% confidence.

Ours vs	HMM	DV	LSTM	Dtree
W / L	19 / 5	24 / 0	24 / 0	15 / 9

8.3 User Preference Study

We conducted a user preference study to complement our quantitative experiments. We compared our approach to the baseline implementations using two sets of test sentences. The first are the fifty KB-2k test sentences, which is the same speaker as the training set. The second is a set of 24 sentences each spoken by a different speaker not contained in the training set and represents a challenging generalization test. Note that for the second set of sentences we do not have ground truth parameterized reference video and so there is no analogous AAM benchmark evaluation for them.

We conducted the user preference study on Amazon Mechanical Turk. For each sentence we showed two animations side-by-side and asked the subject to make a forced choice of which animation seems more natural. We collected 25 judgments per sentence and comparison case. A method "wins" the comparison if it receives a majority of the preference judgments (i.e., at least 13). The raw user study results are available in the supplementary material.

Table 1 shows the aggregate results for the fifty KB-2k test sentences. We see that our approach is preferred to the baselines, and is comparable to the ground truth AAM reference representation. Table 2 shows analogous results for the 24 novel speaker test sentences. We again see the same pattern of preferences. These results suggest that our approach enjoys robust perceptual performance gains over previous baselines.

9 SUMMARY

We introduce a deep learning approach using sliding window regression for generating realistic speech animation. Our framework has several advantages compared to previous work on visual speech animation:

- Our approach requires minimal hand-tuning, and is easy to deploy.
- Compared to other deep learning approaches, our approach exploits a key inductive bias that the primary focus should be on jointly predicting the local temporal curvature of visual speech. This allows our approach to generalize well to any speech content using a relatively modest training set.
- The compact reference parameterization means our approach is easy to retarget to new characters.
- It is straightforward to edit and stylize the retargeted animation in standard production editing software.

We demonstrate using both quantitative and subjective evaluations that our approach significantly outperforms strong baselines from previous work. We show that these performance gains are robust by evaluating on input from novel speakers and in novel speaking styles not contained in the training set.

9.1 Limitations & Future Work

The main practical limitation is that our animation predictions are made in terms of the reference face AAM parameterization. This enables the generalization of our approach to any content, but retargeting to a character introduces a potential source of errors. Care must be taken when posing the initial character setup for the retargeting shapes to preserve the fidelity of the predicted animation. Fortunately, this is a precomputation step that only needs to be performed once per character. Moving forward, one interesting direction for future work is to use real animation data to develop a data-driven retargeting technique tailored for automated speech animation.

By learning from only neutral speech we are able to learn a robust model of speech animation that generalizes to any speech content. It is currently the role of the artist to add expression and emotion. An interesting future direction would be to train a much larger neural network on training data from multiple emotional contexts (e.g., angry, sad, etc.) to make the predicted facial motion closer to the emotional intent. One major challenge is how to cost-effectively collect a comprehensive dataset for training. Without a sufficiently comprehensive training set, it can be challenging to employ modern machine learning techniques, because methods such as deep learning are typically highly underconstrained. Possible directions including collecting "messy" data at scale (e.g., from public video repositories), or developing active learning approaches that adaptively selects which video data to collect in order to minimize total collection costs.

A further generalization could train a speech animation model from multiple speakers possessing a variety of facial characteristics (male, female, round, square, fleshy, gaunt etc.) and select the characteristics most closely matching the character model at prediction time. This approach could generalize different facial dynamics for different face shapes according to the talking style of the character. Again, there is a major challenge of how to effectively collect a comprehensive training set.

ACKNOWLEDGEMENTS

We owe great thanks to our always accommodating and professional actor, Ken Bolden. Barry-John Theobald and Ausdang Thangthai contributed their HMM synthesis implementation. Scott Jones at Lucasfilm and Hao Li at USC generously provided facial rigs. Thanks to the diverse members of Disney Research Pittsburgh who recorded foreign language speech examples. The work was supported by EPSRC grant EP/M014053/1.

REFERENCES

Robert Anderson, Bjorn Stenger, Vincent Wan, and Roberto Cipolla. 2013. Expressive Visual Text-To-Speech Using Active Appearance Models. In *Proccedings of the International Conference on Computer Vision and Pattern Recognition*. 3382–3389.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian Goodfellow, Arnaud Bergeron, Nicolas Bouchard, David Warde-Farley, and Yoshua Bengio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop. (2012).

Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W Sumner, and Markus Gross. 2011. High-quality passive facial performance capture using anchor frames. *ACM Transactions on Graphics* 30 (Aug. 2011), 75:1–75:10. Issue 4.

Matthew Brand. 1999. Voice Puppetry. In *Proceedings of SIGGRAPH*. ACM, 21–28.

Christoph Bregler, Michele Covell, and Malcolm Slaney. 1997. Video Rewrite: Driving Visual Speech with Audio. In *Proceedings of SIGGRAPH*. 353–360.

Chen Cao, Derek Bradley, Kun Zhou, and Thabo Beeler. 2015. Real-time high-fidelity facial performance capture. *ACM Transactions on Graphics* 34, 4 (2015), 46.

Chen Cao, Yanlin Weng, Stephen Lin, and Kun Zhou. 2013. 3D Shape Regression for Real-time Facial Animation. *ACM Transactions on Graphics* 32, 4 (2013), 41:1–41:10.

Yong Cao, Wen C Tien, Petros Faloutsos, and Frédéric Pighin. 2005. Expressive Speech-Driven Facial Animation. *ACM Transactions on Graphics* 24, 4 (2005), 1283 – 1302.

Rich Caruana and Alexandru Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In *International Conference on Machine Learning (ICML)*. 161–168.

Michael M Cohen, Dominic W Massaro, and others. 1994. Modeling Coarticualtion in Synthetic Visual Speech. In *Models and Techniques in Computer Animation*, N.M. Thalmann and Thalmann D (Eds.). Springer-Verlag, 141–155.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. 2001. Active Appearance Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23, 6 (2001), 681–685.

Eric Cosatto and Hans Peter Graf. 2000. Photo-realistic Talking-heads from Image Samples. *IEEE Transactions on Multimedia* 2, 3 (2000), 152–163.

José Mario De Martino, Léo Pini Magalhães, and Fábio Violaro. 2006. Facial animation based on context-dependent visemes. *Journal of Computers and Graphics* 30, 6 (2006), 971 – 980.

Salil Deena, Shaobo Hou, and Aphrodite Galata. 2010. Visual speech synthesis by modelling coarticulation dynamics using a non-parametric switching state-space model. In *Proceedings of the International Conference on Multimodal Interfaces*. 1–8.

Pif Edwards, Chris Landreth, Eugene Fiume, and Karan Singh. 2016. JALI: an animator-centric viseme model for expressive lip synchronization. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 127.

Gwenn Englebienne, Timothy F Cootes, and Magnus Rattray. 2007. A Probabilistic Model for Generating Realistic Speech Movements from Speech. In *Proceedings of Advances in Natural Information Processing Systems*. 401–408.

Tony Ezzat, Gadi Geiger, and Tomaso Poggio. 2002. Trainable Videorealistic Speech Animation. In *ACM Transactions on Graphics*. 388–398.

Bo Fan, Lijuan Wang, Frank K Soong, and Lei Xie. 2015. Photo-real Talking Head with Deep Bidirectional LSTM. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 4884–4888.

Shengli Fu, Ricardo Gutierrez-Osuna, Anna Esposito, Praveen K Kakumanu, and Oscar N Garcia. 2005. Audio/visual mapping with cross-modal hidden Markov models. *IEEE Transactions on Multimedia* 7, 2 (2005), 243–252.

Graham Fyffe, Andrew Jones, Oleg Alexander, Ryosuke Ichikari, and Paul Debevec. 2014. Driving High-Resolution Facial Scans with Video Performance Capture. *ACM Transactions on Graphics* 34, 1 (2014), 8.

John S Garofolo, Lori F Lamel, William M Fisher, Jonathon G Fiscus, and David S Pallett. 1993. *Darpa Timit Acoustic-Phonetic Continuous Speech Corpus CD-ROM TIMIT*. Technical Report 4930. NIST.

Oxana Govokhina, Gérard Bailly, Gaspard Breton, and Paul Bagshaw. 2006. TDA: A new trainable trajectory formation system for facial animation. In *Proceedings of Interspeech*. 2474–2477.

Alex Graves and Navdeep Jaitly. 2014. Towards End-To-End Speech Recognition with Recurrent Neural Networks. In *ICML*, Vol. 14. 1764–1772.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

Haoda Huang, Jinxiang Chai, Xin Tong, and Hsiang-Tao Wu. 2011. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. In *ACM Transactions on Graphics*, Vol. 30. ACM, 74.

Taehwan Kim, Yisong Yue, Sarah Taylor, and Iain Matthews. 2015. A Decision Tree Framework for Spatiotemporal Sequence Prediction. In *ACM Conference on Knowledge Discovery and Data Mining*. 577–586.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems*. 1097–1105.

Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. 2013. Realtime Facial Animation with On-the-fly Correctives. *ACM Transactions on Graphics* 32, 4 (2013), 42–1.

Kang Liu and Joern Ostermann. 2012. Evaluation of an image-based talking head with realistic facial expression and head motion. *Multimodal User Interfaces* 5 (2012), 37–44.

Changwei Luo, Jun Yu, Xian Li, and Zengfu Wang. 2014. Realtime speech-driven facial animation using Gaussian Mixture Models. In *IEEE Conference on Multimedia and Expo Workshops*. 1–6.

Jiyong Ma, Ron Cole, Bryan Pellom, Wayne Ward, and Barbara Wise. 2006. Accurate Visible Speech Synthesis Based on Concatenating Variable Length Motion Capture Data. *IEEE Transactions on Visualization and Computer Graphics* 12, 2 (2006), 266–276.

Iain Matthews and Simon Baker. 2004. Active Appearance Models Revisited. *International Journal of Computer Vision* 60, 2 (2004), 135–164.

Wesley Mattheyses, Lukas Latacz, and Werner Verhelst. 2013. Comprehensive many-to-many phoneme-to-viseme mapping and its application for concatenative visual speech synthesis. *Speech Communication* 55, 7–8 (2013), 857–876.

Harry McGurk and John MacDonald. 1976. Hearing lips and seeing voices. *Nature* 264 (Dec. 1976), 746–748.

Thomas Merritt and Simon King. 2013. Investigating the shortcomings of HMM synthesis. In *ISCA Workshop on Speech Synthesis*. 185–190.

Julian James Odell. 1995. *The Use of Context in Large Vocabulary Speech Recognition*. Ph.D. Dissertation. Cambridge University.

Dietmar Schabus, Michael Pucher, and Gregor Hofer. 2011. Simultaneous Speech and Animation Synthesis. In *ACM SIGGRAPH Posters*. 8:1–8:1.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)* 15, 1 (2014), 1929–1958.

Robert W Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. *ACM Transactions on Graphics* 23, 3 (2004), 399–405.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Neural Information Processing Systemsw*. 3104–3112.

Sarah L Taylor, Moshe Mahler, Barry-John Theobald, and Iain Matthews. 2012. Dynamic Units of Visual Speech. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 275–284.

Barry-John Theobald and Iain Matthews. 2012. Relating Objective and Subjective Performance Measures for AAM-based Visual Speech Synthesizers. *IEEE Transactions on Audio, Speech and Language Processing* 20, 8 (2012), 2378.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).

Lijuan Wang, Wei Han, and Frank K Soong. 2012. High quality lip-sync animation for 3D photo-realistic talking head. In *IEEE Conference on Acoustics, Speech and Signal Processing*. IEEE, 4529–4532.

Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. 2011. Realtime Performance-based Facial Animation. In *ACM Transactions on Graphics (TOG)*, Vol. 30. 77:1–77:10.

Yanlin Weng, Chen Cao, Qiming Hou, and Kun Zhou. 2014. Real-time facial animation on mobile devices. *Graphical Models* 76, 3 (2014), 172–179.

Lei Xie and Zhi-Qiang Liu. 2007. A coupled HMM approach to video-realistic speech animation. *Pattern Recognition* 40, 8 (2007), 2325–2340.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044* 2, 3 (2015), 5.

Yuyu Xu, Andrew W Feng, Stacy Marsella, and Ari Shapiro. 2013. A Practical and Configurable Lip Sync Method for Games. In *Proc. ACM SIGGRAPH Motion in Games*. 131–140.

Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, and others. 2006. *The HTK Book*. Cambridge University.

Jiahong Yuan and Mark Liberman. 2008. Speaker Identification on the SCOTUS Corpus. *Journal of the Acoustical Society of America* 123, 5 (2008).

Heiga Zen, Takashi Nose, Junichi Yamagishi, Shinji Sako, Takashi Masuko, Alan Black, and Keiichi Tokuda. 2007. The HMM-based speech synthesis system version 2.0. In *Proceedings of the Speech Synthesis Workshop*. 294–299.

Li Zhang, Noah Snavely, Brian Curless, and Steven M Seitz. 2004. Spacetime Faces: High Resolution Capture for Modeling and Animation. In *ACM Transactions on Graphics*. 548–558.

A Deep Learning Approach for Generalized Speech Animation — Supplementary Material

Online Submission ID: 0509

1 Indicator Feature Sets

A total of 2379 binary indicator features are used to train the models, where the indicator feature $\delta(\cdot)$ is 1 if the condition is met and 0 otherwise. The indicator features are built around a set of 40 American English phonemes plus *silence*: /sil/, /dh/, /ih/, /s/, /z/, /uh/, /p/, /r/, /aa/, /b/, /l/, /m/, /th/, /ae/, /d/, /g/, /ow/, /k/, /n/, /er/, /iy/, /y/, /w/, /eh/, /ch/, /v/, /t/, /sh/, /f/, /ah/, /sp/, /hh/, /aw/, /oy/, /uw/, /ey/, /ao/, /zh/, /ay/, /jh/, /ng/. This section details the indicator features used in training the models. We use $K_x = 11$.

1.1 Phoneme Duration and Location Indicator Features

Indicator features of this category indicate whether a specific phone span specific consecutive frames. For example, "Does the phone /s/ span frames j through k of the input subsequence?"

- $\delta(x_{i:i+k} == p_j)$ where $i = 1, \ldots, K_x, k = 0, 1, 2, 3$, and phoneme $p_j, j = 1..41 \Rightarrow 41 * (K_x + K_x - 1 + K_x - 2 + K_x - 3) = 1558$ indicator features.

1.2 Articulation and Phoneme Attribute Indicator Features

Indicator features of this category indicate whether a specific phone in a specific location belongs to one of sixty categories describing place and manner of articulation and other phonetic attributes. Table 1 details the attributes and corresponding phoneme sets. This set is largely taken from the 51 phonetic questions from [Odell 1995] (Appendix. B). An example indicator feature in this category is: "Is the phone at the i-th frame a nasal consonant?". An additional indicator feature in this category indicates whether there is a consonant in the first or second half of the input subsequence.

- $\delta(x_i \in PC_j)$ where $i = 1, \ldots, K_x$, PC_j is a phonetic category, and $j = 1, \ldots, 60 \Rightarrow K_x * 60 = 660$ indicator features.
- $\delta(x_{c-5:c-2} \in Consonant \vee x_{c+2:c+5} \in Consonant)$ where x_c is a center frame $\Rightarrow$ 1 indicator feature.

1.3 Phoneme Pair Transitions Indicator Features

These indicator features correspond to an indicator function defining whether a pair of frames correspond to a particular phoneme attribute (vowel or consonant) or data-driven transition categories at a specific location. For the data-driven transition categories, we first collect all AAM parameters corresponding to phone pairs presented in training data and then cluster them to two or three clusters. All details of phoneme pair memberships in each cluster are in the supplementary text file (supplementary.txt). E.g., "Are the phones at k-th and $k+1$-th input frames in a specific cluster of consonant-vowel pairs?"

- $\delta(x_{i,i+1} \in C_j^1)$ where $i = 1, \ldots, K_x - 1$ and $C_j^1, j = 1, 2$ is a cluster of Consonant+Vowel pairs $\Rightarrow K_x - 1 * 2 = 20$ indicator features.
- $\delta(x_{i,i+1} \in C_j^2)$ where $i = 1, \ldots, K_x - 1$ and $C_j^2, j = 1, 2, 3$ is a cluster of Vowel+Consonant pairs $\Rightarrow K_x - 1 * 3 = 30$ indicator features.
- $\delta(x_{i,i+1} \in C_j^3)$ where $i = 1, \ldots, K_x - 1$ and $C_j^3, j = 1, 2, 3$ is a cluster of Consonant+Consonant pairs $\Rightarrow K_x - 1 * 3 = 30$ indicator features.
- $\delta(x_i \in Consonant)\delta(x_{i+1} \in Vowel_p)$ where $i = 1, \ldots, K_x - 1$, and $Vowel_p$ is a vowel starting with phoneme $p \Rightarrow K_x - 1 * 5 = 50$ indicator features.
- $\delta(x_i \in Consonant)\delta(x_{i+1} \in Vowel^p)$ where $i = 1, \ldots, K_x - 1$, and $Vowel^p$ is a vowel ending with phoneme $p \Rightarrow K_x - 1 * 3 = 30$ indicator features.

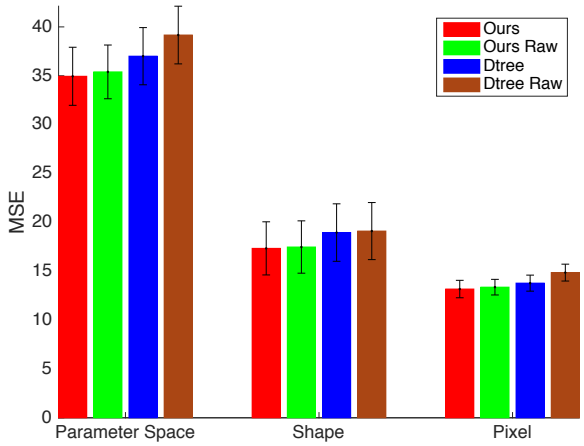

Figure 1: Showing the mean square error of the KB-2k held out test sentences over the face model parameter space, the full AAM shape space, and pixel space.

To examine significance of linguistically-motivated features, we evaluate our approach and decision tree regression on the KB-2k 50 held out test sentences, which share the same feature representation. We compare the results between with full features and only the raw phoneme identity features by measuring squared error in the reference model parameter space, in the raw Active Appearance Model shape space, and in pixel space. Decision tree regression is denoted "Dtree" and Figure 1 shows the results. Note that using only the raw features achieves almost the same performance.

Attribute	Phoneme members
Vowel	/ih/,/uh/,/aa/,/ae/,/ow/,/er/,/iy/,/eh/,/ah/,/aw/,/oy/,/uw/,/ey/,/ao/,/ay/
Vowel starting with /a/	/aa/,/ae/,/ah/,/aw/,/ao/,/ay/
Vowel starting with /e/	/eh/,/ey/
Vowel starting with /i/	/ih/,/iy/
Vowel starting with /o/	/ow/,/oy/
Vowel starting with /u/	/uh/,/uw/
Vowel ending with /h/	/ih/,/uh/,/eh/,/ah/
Vowel ending with /w/	/ow/,/aw/,/uw/
Vowel ending with /y/	/iy/,/oy/,/ey/,/ay/
Plosive	/b/,/d/,/g/,/k/,/p/,/t/
Affricative	/ch/,/jh/
Nasal	/m/,/n/,/ng/
Fricative	/f/,/v/,/th/,/dh/,/s/,/z/,/sh/,/zh/,/hh/
Approximant	/w/,/r/,/y/
Bilabial	/p/,/b/,/m/
Labiodental	/f/,/v/
Dental	/th/,/dh/
Alveolar	/t/,/d/,/n/,/s/,/z/,/l/
PostAlveolar	/ch/,/jh/,/sh/,/zh/,/r/
Velar	/k/,/g/,/ng/,/w/
Unvoiced-Consonant	/p/,/f/,/th/,/t/,/s/,/ch/,/sh/,/k/,/hh/
Voiced-Consonant	/b/,/m/,/v/,/dh/,/d/,/n/,/z/,/l/,/jh/,/zh/,/r/,/y/,/g/,/ng/,/w/
Voiced-Plosive	/b/,/d/,/g/
Unvoiced-Plosive	/p/,/t/,/k/
Voiced-Fricative	/v/,/dh/,/z/,/zh/
Unvoiced-Fricative	/f/,/th/,/s/,/sh/,/hh/
Semi-Consonant	/y/,/w/
Sibilant-Consonant	/ch/,/jh/,/s/,/z/,/sh/,/zh/
Sibilant-Affricate	/ch/,/jh/
Sibilant-Fricative	/s/,/z/,/sh/,/zh/
Front-Vowel	/iy/,/ih/,/en/,/ae/
Central-Vowel	/er/,/ax/,/ah/
Back-Vowel	/uw/,/uh/,/ao/,/aa/,/oh/
Front-Consonant	/b/,/f/,/m/,/p/,/v/,/w/
Central-Consonant	/d/,/dh/,/dx/,/l/,/n/,/r/,/s/,/t/,/th/,/z/,/zh/
Back-Consonant	/ch/,/g/,/hh/,/jh/,/k/,/ng/,/sh/,/y/
Front-Stop	/b/,/p/
Central-Stop	/d/,/t/
Back-Stop	/g/,/k/
Front-Fricative	/f/,/v/
Central-Fricative	/dh/,/s/,/th/,/z/
Back-Fricative	/ch/,/jh/,/sh/,/zh/
Front	/b/,/f/,/m/,/p/,/v/,/w/,/iy/,/ih/,/en/,/ae/
Central	/d/,/dh/,/dx/,/l/,/n/,/r/,/s/,/t/,/th/,/z/,/zh/,/er/,/ax/,/ah/
Back	/ch/,/g/,/hh/,/jh/,/k/,/ng/,/sh/,/y/,/uw/,/uh/,/ao/,/aa/,/oh/
Long-Vowel	/iy/,/er/,/uw/,/ao/,/aa/
Short-Vowel	/ih/,/eh/,/ae/,/ax/,/ah/,/uh/,/oh/
Vowel-Close	/iy/,/ih/,/uw/,/uh/
Vowel-Mid	/eh/,/er/,/ax/,/ao/
Vowel-Open	/ae/,/ah/,/aa/,/oh/
Vowel-Front	/iy/,/ih/,/eh/,/ae/
Vowel-Central	/er/,/ax/,/ah/
Vowel-Back	/uw/,/uh/,/ao/,/aa/,/oh/
Dipthong-Vowel	/ey/,/ay/,/oy/,/ow/,/aw/,/ia/,/ua/,/ea/
Dipthong-Closing	/ey/,/ay/,/oy/,/ow/,/aw/
Dipthong-centring	/ia/,/ua/,/ea/
AVowel	/ay/,/ae/,/aa/,/aw/,/ao/
OVowel	/ao/,/ow/,/oy/,/oh/
UVowel	/ah/,/ax/,/ua/,/uh/,/uw/
silences	/pau/,/h
Total number	**60**

Table 1: Phoneme attributes we exploited for our indicator feature sets in Sec. 1.2.

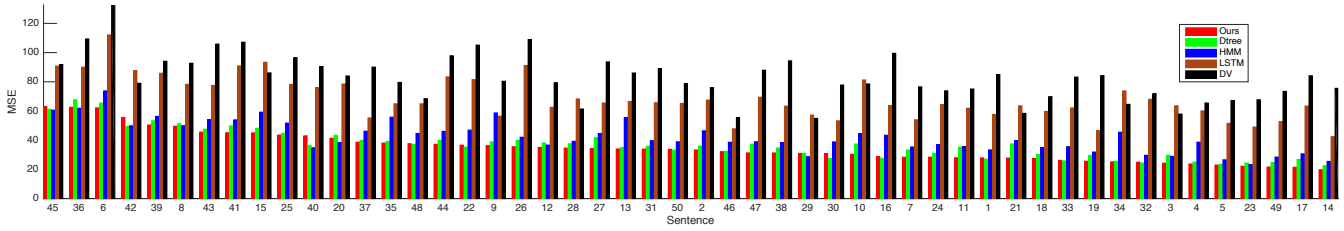

Figure 2: Showing the squared error (in face model space) for each held out KB-2k held out test sentence. The sentences are sorted in descending order of squared error for our approach. We see that our approach consisently outperforms all baselines.

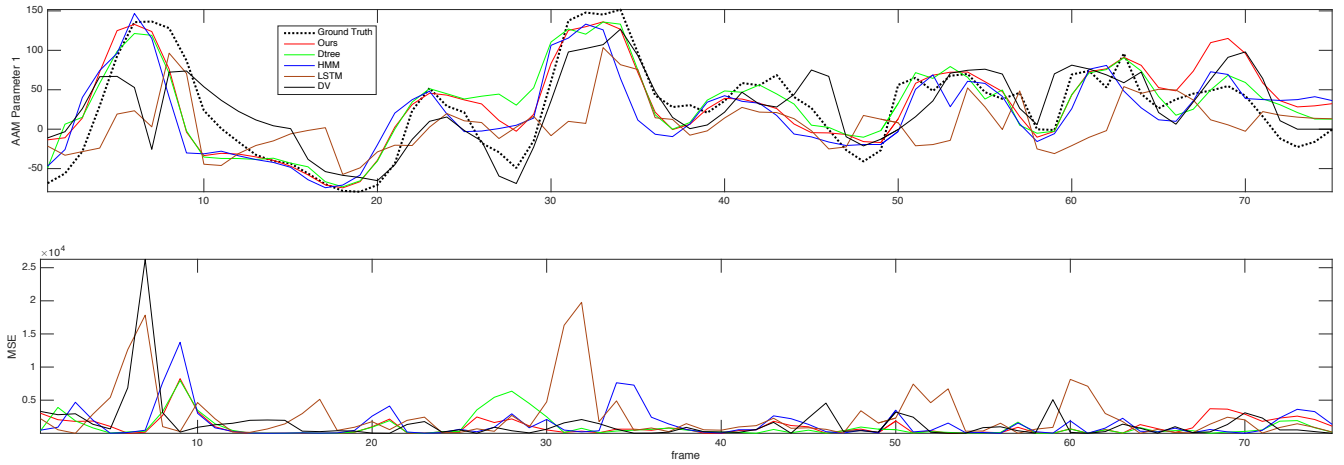

Figure 3: Comparing the predictions of various predictions for the first face model parameter on held out test sentence 48. The first parameter corresponds to how wide open the moth is. The bottom plot shows the per-frame squared error.

2 Prediction and Error Plots

To better investigate the performance difference of the various approaches, we evaluate all approaches per each sentence on the KB-2k 50 held out test sentences. We compute squared error in the reference AAM model parameter space, in the predicted shape vertex positions, and in predicted appearance pixel intensities. Decision tree regression is denoted "Dtree", and dynamic visemes is denoted "DV". In Figure 2, we see that our approach consistently outperforms the baseline approaches on the majority of the held out test sentences. These results suggest that our approach achieves state-of-the-art performance in visual speech animation.

Figure 3 shows the frame-by-frame predictions of the various approaches for the first face parameter on held out sentence 48. The first face parameter corresponds to how wide open the mouth is. We see that LSTMs and Dynamic Visemes suffer extremely large errors, with the LSTM predictions being particularly jittery. The decision tree regression and HMM-based synthesizer are much more competitive, however occasionally suffers a relatively large error. While the average errors from the decision tree and HMM might be small due to it usually matching the ground truth relatively well. However, the spikes in error can dramatically reduce the perceived quality of the resulting animation. We observe much smaller error spikes from our approach.

3 Raw User Study Results

KB-2k 50 held out sentences. This is the raw results that were aggregated into Table 1 in the main paper. Each entry corresponds to how many users (out of 25) preferred our approach versus a baseline on a specific test sentence.

Ours vs DV [23, 21, 18, 24, 20, 23, 22, 24, 23, 22, 22, 23, 22, 24, 21, 23, 24, 22, 24, 24, 21, 24, 22, 22, 22, 24, 23, 24, 22, 22, 23, 23, 24, 23, 21, 22, 23, 23, 23, 18, 24, 21, 22, 23, 24, 23, 24, 24, 21, 22]

Ours vs HMM [12, 17, 16, 17, 18, 20, 16, 14, 19, 17, 10, 21, 20, 16, 13, 16, 16, 14, 14, 17, 13, 9, 13, 19, 9, 14, 15, 12, 17, 21, 11, 17, 16, 15, 15, 22, 13, 15, 12, 6, 12, 15, 11, 13, 21, 15, 12, 15, 10, 19]

Ours vs LSTM [20, 25, 23, 23, 23, 22, 23, 24, 23, 23, 22, 24, 24, 25, 22, 24, 25, 21, 23, 24, 22, 23, 21, 24, 23, 23, 23, 24, 25, 25, 24, 24, 23, 23, 24, 24, 23, 24, 24, 24, 23, 22, 23, 23, 23, 23, 24, 24, 22, 23]

Ours vs DTree [14, 12, 9, 12, 9, 11, 15, 12, 17, 14, 18, 16, 9, 13, 11, 18, 13, 12, 18, 14, 7, 14, 14, 13, 13, 16, 14, 16, 16, 13, 10, 16, 5, 17, 14, 19, 14, 17, 14, 14, 18, 12, 10, 14, 19, 14, 10, 14, 16, 15]

Novel Speakers The 24 novel speaker sentences. This is the raw results that were aggregated into Table 2 in the main paper. Each entry corresponds to how many users (out of 25) preferred our approach versus a baseline on a specific test sentence.

Ours vs DV: [25, 22, 24, 21, 22, 24, 24, 23, 24, 22, 23, 24, 22, 18, 22, 24, 20, 23, 22, 20, 19, 23, 23, 23]

Ours vs HMM [19, 17, 17, 19, 20, 17, 22, 13, 13, 20, 18, 9, 20, 7, 18, 11, 13, 9, 16, 15, 13, 14, 15, 8]

Ours vs LSTM [24, 18, 23, 24, 18, 20, 23, 24, 22, 21, 22, 18, 23, 20, 23, 25, 24, 23, 23, 23, 23, 24, 23, 25]

Ours vs DTree [13, 19, 14, 9, 12, 16, 16, 14, 15, 13, 16, 11, 14, 16, 17, 11, 17, 10, 13, 12, 13, 7, 9, 12]