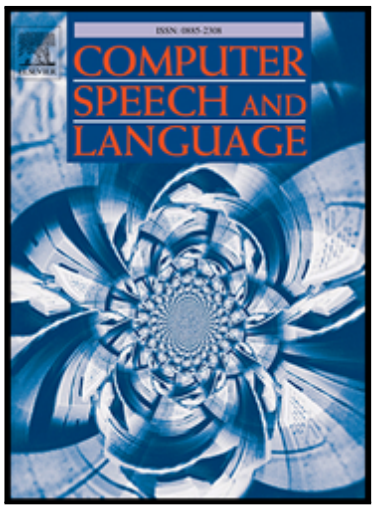

Synthesising visual speech using dynamic visemes and deep learning architectures

Ausdang Thangthai, Ben Milner, Sarah Taylor

PII: S0885-2308(18)30027-5
DOI: https://doi.org/10.1016/j.csl.2018.11.003
Reference: YCSLA 963

To appear in: *Computer Speech & Language*

Received date: 24 January 2018
Revised date: 24 September 2018
Accepted date: 9 November 2018

Please cite this article as: Ausdang Thangthai, Ben Milner, Sarah Taylor, Synthesising visual speech using dynamic visemes and deep learning architectures, *Computer Speech & Language* (2018), doi: https://doi.org/10.1016/j.csl.2018.11.003

Highlights

- We propose a method to synthesise visual speech from linguistic features
- Best performance is found with dynamic visemes and an LSTM many-to-many architecture
- Using subjective tests to compare to other techniques, the proposed method produces more natural animations

Synthesising visual speech using dynamic visemes and deep learning architectures

Ausdang Thangthai, Ben Milner*, Sarah Taylor
School of Computing Sciences, University of East Anglia, UK

Abstract

This paper proposes and compares a range of methods to improve the naturalness of visual speech synthesis. A feedforward deep neural network (DNN) and many-to-one and many-to-many recurrent neural networks (RNNs) using long short-term memory (LSTM) are considered. Rather than using acoustically derived units of speech, such as phonemes, viseme representations are considered and we propose using dynamic visemes together with a deep learning framework. The input feature representation to the models is also investigated and we determine that including wide phoneme and viseme contexts is crucial for predicting realistic lip motions that are sufficiently smooth but not under-articulated. A detailed objective evaluation across a range of system configurations shows that a combined dynamic viseme-phoneme speech unit combined with a many-to-many encoder-decoder architecture models visual co-articulations effectively. Subjective preference tests reveal there to be no significant difference between animations produced using this system and using ground truth facial motion taken from the original video. Further-

*Corresponding author
Email address: b.milner@uea.ac.uk (Ben Milner)

more, the dynamic viseme system also outperforms significantly conventional phoneme-driven speech animation systems.

Keywords: Talking head, visual speech synthesis, deep neural network, dynamic visemes

1. INTRODUCTION

Talking heads that are animated automatically by computers are becoming increasingly commonplace in a range of scenarios. Animation of characters in films has typically been performed either by artists who produce speech animations manually or by using motion capture with a human actor. The quality of the animation is of primary concern which is why such methods are employed, despite their time and expense. Being able to automate this process and still produce suitably realistic animations would save much effort and is therefore the focus of this work. Previously, automated systems have not generally been considered good enough for commercial applications, although hybrid approaches can be used where automatically generated animations are refined by artists. Virtual characters in computer games and as intelligent assistants are another application for talking heads and are characterised by the animations needing to be generated automatically from speech and in real-time. In these contexts more realistic character animation would lead to better user experiences.

Underlying these applications is visual speech synthesis which we consider to be the process of transforming a linguistic representation (e.g. word sequence) or acoustic representation (e.g. audio speech signal) into a sequence of visual speech parameters that can subsequently be used to animate a talking

head. It is the aim of this work to improve visual speech synthesis to produce a more accurate sequence of visual speech parameters and subsequently a more natural animation. Specifically, this paper explores the application of deep learning approaches, that have been successful in audio speech processing, to now predict visual speech features from a linguistic input with the aim of producing natural speech animations. We first propose using a joint speech unit that now combines phonemes with visual speech units made from *dynamic* visemes instead of previously used (static) visemes. Second, we explore a range of low-level (frame-based) features that include phoneme and viseme information and subsequently examine how these effect the resulting sequence of visual parameters. Third, we explore and compare different deep learning models that include feedforward neural networks and recurrent structures using LSTMs. Specifically, we consider one-to-one, many-to-one and many-to-many architectures to find which gives best performance when integrated with different linguistic features and speech units. We also explore the effectiveness of objective measures of visual feature prediction and observe that global variance is a good predictor of the subjective naturalness of animations, when considered alongside the more conventional measures of correlation and mean squared error.

The paper is organised as follows. Section 2 describes related work, while Section 3 explains the phoneme, static viseme and dynamic viseme speech units. Section 4 introduces the hierarchical linguistic input features that are extracted from the input text and the output AAM visual features. The feedforward and recurrent neural network architectures are explained in Section 5. Section 6 presents the results of optimisation and analysis on

the effect of different speech units, linguistic features and model structures. Finally, Section 7 presents results of practical evaluations using objective measurements and subjective preference tests to analyse the naturalness of the resulting animations from a range of system configurations.

2. RELATED WORK

Existing methods of automatic speech animation can be classified loosely as key-frame interpolation (Cohen and Massaro, 1994; Ezzat et al., 2002), concatenative (Bregler et al., 1997; Taylor et al., 2012; Mattheyses et al., 2011) or model-based (Anderson et al., 2013; Fan et al., 2015a; Kim et al., 2015). Key-frame interpolation involves transitioning between pre-defined lip shapes, for example a pose placed at key phoneme onsets. Concatenative or sample-based systems stitch together short facial video clips from a database to create an animation and can be considered similar to the unit selection method of audio speech synthesis (Hunt and Black, 1996). Model-based methods use a form of generative statistical model of the face or lips with the visual speech synthesiser generating a sequence of model parameters that drives the animation. This is similar to audio speech synthesis methods that generate acoustic parameters to drive models of speech production (Zen et al., 2009). Whilst concatenative methods have the potential to produce photo-realistic speech animation, they are limited by the size and coverage of the database. Model-based methods are more flexible as they are able to produce deformations that do not appear in the training data, and are therefore able to generate richer animations. For this reason, we take the model-based approach in this work.

Visual speech synthesisers can be driven from a text input or from an acoustic input. From a text input, a linguistic representation (such as phoneme labels and context) is first created which is fed into a previously learnt model that produces a sequence of visual speech parameters (Kim et al., 2015). If required, the same linguistic features can be used to create an audio speech signal (Zen et al., 2009), which can then be combined with the visual speech parameters to give audio-visual speech synthesis. In situations where an acoustic speech signal is already available, and an accompanying visual synthesis is required, this can be achieved either by decoding the audio speech into a linguistic representation using an automatic speech recogniser, or by learning a mapping from acoustic features (e.g filterbank) to visual features (Ding et al., 2014; Taylor et al., 2016). In this work, we take the approach of using a linguistic representation to form the input to the visual speech synthesiser. A key challenge with this approach is deciding upon the basic unit of speech that is presented to the mapping model. A phonetic representation is the logical approach taken by audio speech synthesis systems since a phoneme describes a distinctive speech sound, but for visual synthesis there is no equivalent unit. A large number of different viseme sets have been proposed to define groups of distinctive lip shapes, but no definitive grouping has yet been established (Lidestam and Beskow, 2006; Parke and Waters, 1996). Furthermore, when visemes are used as the units for visual synthesis, the animation has often been found to lack realism (Taylor et al., 2012).

Mapping a sequence of visual speech parameters from a phoneme or viseme sequence is a non-trivial problem that requires sophisticated models to produce a suitably realistic output. Building on their success in modelling

audio speech for audio speech recognition and synthesis applications, hidden Markov models (HMMs) have been shown to be an effective way to model visual speech and have been used for visual speech synthesis (Luo et al., 2014; Deena, 2012; Schabus et al., 2014; Thangthai and Theobald, 2015). However, recent studies have shown that HMMs are limited by their Gaussian mixture model-based states (with the assumption of diagonal covariances) and by the decision tree clustering of visual features within states (Watts et al., 2016). Mirroring developments in audio speech processing, deep neural network (DNN) methods for visual speech synthesis have been proposed and overcome some of these problems. Feedforward DNNs are able to model high dimensional and correlated feature vectors and learn the complex non-linear mappings between input linguistic features and output visual parameters which makes them well suited to visual speech synthesis. Input phoneme labels have been mapped successfully to active appearance model (AAM) visual speech parameters using DNNs to produce both neutral looking speech (Taylor et al., 2017) and expressive speech (Filntisis et al., 2017; Parker et al., 2017). However, these feedforward networks do not consider fully the temporal nature of the speech signal. This temporal dependency can be modelled by using a window of frames as input to the model, but ultimately the DNN assumes that input features are sampled independently. Smoothing the outputs can improve the appearance of the predicted animation, but a better approach is to model the temporal dynamics explicitly. Instead, recurrent neural networks (RNNs) can be more effective with the output dependent not only on the current input vector but upon a sequence of input vectors. Furthermore, bidirectional RNNs (BRNNs) (Schuster and Paliwal,

1997) have been applied successfully to audio speech synthesis and model both past and future input features. However, when attempting to model long span relationships using RNNs, propagated gradients can become very small and vanish. The long short-term memory (LSTM) model has been shown to avoid this by using a series of gates to control the flow of information and when combined with an RNN is an effective architecture (Hochreiter and Schmidhuber, 1997).

3. SPEECH UNITS

In audio text-to-speech synthesis the underlying unit of speech is normally derived from a phonetic unit, such as a phoneme, diphone, triphone or quinphone (Hunt and Black, 1996; Zen et al., 2009). For visual speech synthesis an underlying speech unit is still required although there is no clearly defined visual equivalent to a phoneme. Previous work in visual speech synthesis has continued to use phonemes as the speech unit while other approaches have employed static visemes (Ezzat et al., 2002). In this work, both approaches are evaluated and we also propose using dynamic visemes which have been shown to generate more realistic speech animation (Taylor et al., 2012).

3.1. Phonetic units

Using phonemes as the basic unit of speech is well established in audio speech synthesis and has also been effective in visual speech synthesis within both HMM and RNN architectures (Schabus et al., 2014; Thangthai and Theobald, 2015; Fan et al., 2015b; Ding et al., 2014). From a text input, it is straightforward to obtain a phoneme sequence using a lexicon along with

associated durational information. If the input is audio then a sub-word speech recogniser, for example based on triphones, can provide a time-aligned phoneme sequence (Brugnara et al., 1993). In this work the set of 39 ARPAbet phonemes is used (Shoup, 1980).

3.2. Static viseme units

Visemes have traditionally been defined as groups of phonemes that are expected to exhibit visually similar lip shapes. Their definition is much more subjective than that of phonemes and a number of viseme sets have been defined and contain from 3 (Lidestam and Beskow, 2006) to 18 (Parke and Waters, 1996) units. Phonemes are typically mapped to visemes through a simple many-to-one mapping which is insufficient to model the complex relationships between visual gestures and the resulting speech sounds (Taylor et al., 2012). A small number of studies have proposed many-to-many mappings by clustering phonemes in context (De Martino et al., 2006; Mattheyses et al., 2013), yet to date there exists no definitive set of visemes or an agreed mapping from phonemes to visemes. Thus, the definition of a viseme is informal and as a unit of speech for computer facial animation it is poorly defined. In this work 24 visemes are used as the static viseme speech units and are taken from Fisher's phoneme-to-viseme mapping (Fisher, 1968).

3.3. Dynamic viseme units

In addition to static visemes this work considers dynamic visemes (DVs) as a speech unit (Taylor et al., 2012). Dynamic viseme units were introduced specifically for visual speech processing and represent groupings of similar lip *motions*. This is opposed to static visemes in which lip *poses*, that are

associated with phonemes, are grouped on their visual similarity. Dynamic visemes are extracted with a data-driven approach using only visual information to determine an atomic set of distinctive and reliable lip motions that occur during natural speech. As such, dynamic viseme boundaries are not tied to the boundaries of the underlying phonemes, and DVs often extend across several phonemes. Dynamic visemes better represent visual speech as each viseme serves a particular function, and so substituting one dynamic viseme for another changes the meaning of the utterance visually, which is an analog to a phoneme. The dynamic nature of DVs means that coarticulation effects are modelled explicitly.

3.3.1. Training a set of dynamic visemes

A set of dynamic visemes is learnt by clustering visual speech parameters, which in this work are active appearance model (AAM) features (see Section 4.2). AAM sequences extracted from training data are first segmented into a set of visual gestures by identifying points where AAM acceleration coefficients change sign which indicates instances where visible articulators (lips, jaw, etc) change direction. This segmentation produces a set of N variable length and non-overlapping visual gestures, $\mathbf{\Psi} = \{\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_N\}$, each comprising a sequence of AAM vectors. Clustering is then applied to produce a set of V dynamic visemes, $\Phi = \{v_1, \ldots, v_V\}$, each representing a specific motion of the lips (Taylor et al., 2012).

3.3.2. Generating dynamic viseme sequences

One challenge with using DV labels as input to a visual speech synthesiser is how to derive them from the linguistic input since they have a many-to-many

mapping to phonemes. We consider two approaches in this work. The first approach is to use *reference* DVs which provide a reliable basis for evaluation when optimising and analysing various input feature representations and models, as reported in Section 6. The second approach is to use DV sequences that are generated *automatically* from the linguistic input which is necessary in real applications, and evaluated objectively and subjectively in Section 7. The procedures for obtaining reference and automated DV sequences are now discussed.

Generating reference dynamic viseme sequences - a reference dynamic viseme sequence is obtained by first segmenting a sequence of AAM vectors that have been extracted from validation or test sentences, with the same procedure as used in training, described in Section 3.3.1. Each of the resulting segments is then compared to the set of the segments, $\mathbf{\Psi}$, extracted from the training data, using a Euclidean distance. The dynamic viseme class, v_i, associated with the closest segment is then assigned as the DV label for that segment (Taylor et al., 2012). To determine the number of dynamic viseme classes, V, a series of animations was generated using reference dynamic visemes extracted from the validation test set using different sizes of viseme sets. Examination of the animations found that using V=160 was a good trade off between animation quality and number of classes and aligns with other studies (Taylor et al., 2012).

Automatically generated dynamic viseme sequences - in the practical scenario of generating a dynamic viseme sequence automatically from a word or phoneme sequence input we propose utilising a phoneme-based visual speech synthesiser. This process is shown as the first two stages of the overall

system architecture in Figure 6. After converting the sequence of words to be synthesised into a phoneme representation, a phoneme-based visual speech synthesiser is used to generate a sequence of AAM vectors. As is shown in later chapters, we have developed a range of deep learning architectures for phoneme-based visual speech synthesis and employ the best performing of these (from analysis in Section 7 we use systems DNN_PH or LSTM_PH) to synthesise AAM vectors from the phoneme representation. The synthesised AAM vector sequence is then segmented and dynamic viseme labels assigned to each segment following the procedure used for generating reference dynamic viseme sequences. An iterative approach was also explored whereby the resulting AAM sequence produced from visual synthesis using a joint phoneme-dynamic viseme system was re-segmented to create a new sequence of dynamic visemes. However, this was found to give no significant improvement in performance.

4. INPUT AND OUTPUT FEATURES

The purpose of the input features is to represent the words to be synthesised in a suitably informative and discriminative linguistic way that enables a sequence of output visual features to be produced by the model. The input word sequence needs to be decomposed into lower level features that can be based on phoneme, viseme or dynamic viseme representations or a combination of these. Initial testing established that including a wide temporal context within the features is important for synthesising realistic sequences of output visual features. This motivated us to experiment with using a set of hierarchical linguistic features that represent input information from the

frame-level through to the utterance level and that consider both phoneme and viseme speech units.

4.1. Hierarchical contextual input features

A set of hierarchical features is extracted from an input word sequence. These hierarchical features exist at varying acoustic, visual and linguistic levels from phoneme/viseme information, through to segment, syllable, word, phrase and utterance levels. For HMM-based visual speech synthesis, this level of contextual labelling is sufficient. However, for training DNN or RNN models it is necessary to provide lower-level, frame-based features to create smooth output feature trajectories (this is demonstrated in Section 6.1). In practice it is likely that many contextual factors affect visual articulation, including, for example, phonetic context and number of syllables in the current word (Tokuda et al., 2002). This work investigates a number of such factors and ascertains the effect of frame, segment, syllable, word, phrase and utterance level features on the synthesised visual features. Table 1 summarises the set of hierarchical input features for phoneme units (PH) and for viseme units (V) (both static and dynamic visemes), which are now explained in more detail.

4.1.1. Frame level features

Frame level features represent the lowest linguistic level and contain both instantaneous and contextual information. Several of these features have been found to be effective in other works and we also introduce further frame level features, in particular to represent viseme contexts. For animation, this work requires an output visual frame rate of 29.97 frames per second and therefore input features are produced at that same frame rate. From phoneme

Table 1: *Hierarchical input features for phoneme (PH) and viseme (V) speech units at varying levels, showing elements for each feature.*

Level	Feature	PH	V	Elements
Frame	Phoneme context	×		$M \times P$ binary
	Position in phoneme	×		{s, m, e}
	Number of frames in phoneme	×		Integer
	Forward phoneme span	×		Integer
	Acoustic class	×		57-D binary
	Viseme context		×	$M \times V$ binary
	Position in viseme		×	{s, m, e}
	Number of frames in viseme		×	Integer
	Forward viseme span		×	Integer
Segment	Quinphone context	×		$5 \times P$ binary
	Quinviseme context		×	$5 \times V$ binary
	Number of phonemes in viseme		×	Integer
Syllable	Position of phoneme in syllable	×		{s, m, e, single}
	Number of phonemes in syllable	×		Integer
	Position of viseme in syllable		×	{s, m, e, single}
	Number of visemes in syllable		×	Integer
Word	Position of syllables in word	×	×	{s, m, e, single}
	Number of syllables in word	×	×	Integer
Phrase	Position of syllables in phrase	×	×	{s, m, e, single}
	Number of syllables in phrase	×	×	Integer
	Position of words in phrase	×	×	{s, m, e, single}
	Number of words in phrase	×	×	Integer
Utterance	Position of syllable in utterance	×	×	{s, m, e, single}
	Position of word in utterance	×	×	{s, m, e, single}
	Position of phrase in utterance	×	×	{s, m, e, single}

or viseme transcriptions, at a given time frame, t, the current phoneme is given as $p_t \in \theta$ and the current viseme is given as $v_t \in \Phi$, where θ and Φ represent the set of P phonemes and V visemes, respectively.

Contextual information is included by considering information across a window of M frames and this captures information about the speech content preceding and following the current frame. The width of this window is an odd number which can be varied to consider different resolutions of temporal information and should be large enough to capture contextual and coarticulation factors but short enough to avoid over-smoothing of output features. An analysis is carried out in Section 6.2 on the effect of varying M with different models and windowing methods. In terms of the **phoneme context** feature in Table 1, this comprises $M \times P$ binary features that indicate the phonetic class of the current frame (i.e. centre of the window) and of the $\frac{M-1}{2}$ frames preceding and following the current frame. In each of the M rows, a single element is set to one which corresponds to the phoneme identity while the remaining elements are all zero.

The **position in phoneme** feature has three binary elements that correspond to whether the centre frame, t, is at the *start*, *middle* or *end* of the current phoneme, while the **number of frames in phoneme** feature is an integer that indicates how many frames are in the current phoneme (Zen et al., 2013). The **forward phoneme span** indicates how many frames remain of the current phoneme before changing to the next phoneme (Kim et al., 2015). **Acoustic class** is represented by a 57-D binary feature where each element is a response to a set of 57 questions taken from the contextual questions in the HMM/DNN-based Speech Synthesis System (HTS), such

as '*Is the current phoneme voiced?*' and '*Is the current phoneme nasalised?*' (Zen et al., 2007). The next four rows of frame-level features in Table 1 represent a similar set of features but now defined for viseme speech units, both static and dynamic. There is no acoustic class feature for visemes as the units are visually-based. To illustrate this frame-level feature extraction the upper part of Figure 1 shows the frame-level phonetic features that are extracted from the sentence '*she looked*'. The example uses a context window width of $M = 5$ with the centre of the window at frame $t = 7$.

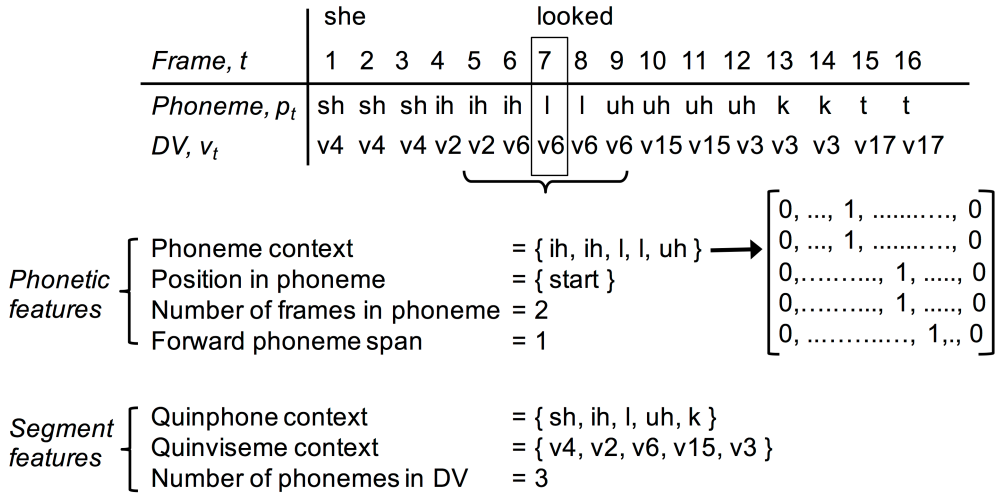

Figure 1: *Example of frame-level phonetic features and segment level features extracted from the sentence* 'she looked'*, with a context width of* $M = 5$ *centred at frame* $t = 7$.

4.1.2. Segment level features

We define a segment as being five phonemes or five visemes in duration and centred about the middle unit, as preliminary tests found a context of five units to give best performance. Phoneme segments were found to have a mean duration of 84ms while dynamic viseme segments are longer with a

mean duration of 183ms. The five phonemes in the segment are represented by the $5 \times P$ **quinphone context** binary feature that indicates the current, the two preceding and the two following phonemes. Similarly, the **quinviseme context** feature is a $5 \times V$ binary feature that indicates the five visemes in the segment. The **number of phonemes in viseme** feature is an integer representing the number of phonemes spanned by the current viseme. For *static* visemes, this is always 1, while for dynamic visemes it will typically be larger. The lower part of Figure 1 shows extraction of the segment level features from the sentence '*she looked*' using dynamic viseme units.

4.1.3. Syllable, word, phrase and utterance features

The syllable level features of **position of phoneme in syllable** and **position of viseme in syllable** are binary features that indicate whether the current unit (phoneme or viseme) is at the *start*, *middle* or *end* of the current syllable. A fourth option, *single*, is used when there is only one frame in the syllable. The **number of phonemes in syllable** and **number of visemes in syllable** features are integers that indicate how many phonemes or visemes are in the current syllable. At the word, phrase and utterance levels the *position* and *number* features encode similar information as for syllable level features, but are no longer unique to phonemes or visemes.

The final linguistic feature vector is represented as $\boldsymbol{l}(M)_t$, where M signifies the number of frames used to represent the phoneme and viseme contexts and t represents the frame number.

4.2. Visual output features

The choice of output feature is governed by the requirement that it should be suitable for visualising a facial animation. Features such as 2-D DCT, whilst successful in several audio-visual speech processing applications (Almajai et al., 2006), are not generative. Consequently, active appearance model (AAM) features are used as they are well suited for visualisation and are an effective and well studied visual feature (Cootes et al., 2001).

Our AAM features are built from a set of images that have been hand-labelled with 34 2-D vertices delineating the contour of the lip and jaw of the speaker. The lower facial region is isolated since broader facial features such as the nose, eyes and eyebrows were found to introduce noise into the model. The hand-labelled sets of vertices are then normalised for translation, scale and rotation and are stacked to create 68-D vectors, $\mathbf{r}$, that encode the 34 pairs of normalised x-y co-ordinates (Gower, 1975). D_s coefficients from a principal components analysis (PCA) are extracted to give parameters, $\mathbf{s}$, that encode the shape of the facial pose. The appearance is modelled with two independent linear models representing the pixel intensities of the inner mouth area and jaw respectively. The regions are modelled separately since the inner mouth area can change somewhat independently to the rest of the jaw due to the presence and positions of the teeth and tongue. The images are warped to the mean shape and the pixels from each region are extracted. PCA is applied to the stacked pixel intensities to extract appearance vectors, $\mathbf{b}_m$ and $\mathbf{b}_j$, that model the mouth and jaw facial regions with D_m and D_j components respectively. The shape and both appearance features are then stacked, and a final PCA is performed to decorrelate the features. The final

AAM vector, $\boldsymbol{a}$, is computed as

$$\boldsymbol{a} = \boldsymbol{R} \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{b}_m \\ \boldsymbol{b}_j \end{bmatrix} \tag{1}$$

where matrix $\mathbf{R}$ is a PCA derived matrix that combines and compresses the shape and appearance components. For the shape and appearance vectors, the dimensionality was selected such that 98% of the total variation is captured which resulted in 30 dimensional AAM vectors. Once constructed, the AAM can be fitted to new images by solving for the model parameters (Cootes et al., 2001). In this way, every video frame in the dataset can be tracked and parameterised into a feature vector, $\boldsymbol{a}_t$, that encodes the visual speech.

5. MAPPING MODELS

Given the success of deep learning techniques across a range of speech processing applications we consider two models for mapping from input linguistic features to output visual features, namely feedforward and recurrent neural networks. Different structures for framing the input and output features are examined as well as one-to-one, many-to-one and many-to-many architectures.

5.1. Feedforward deep neural network

The task of the feedforward deep neural network (DNN) is to map from input linguistic features to output AAM vectors through a set of Q hidden layers (Goodfellow et al., 2016). Given input vector, $\boldsymbol{x}_t$ and output vector, $\boldsymbol{y}_t$, the mapping can be expressed as

$$\boldsymbol{h}_t^1 = g(\boldsymbol{W}_{xh^1}\boldsymbol{x}_t + \boldsymbol{b}^1) \tag{2}$$

$$\boldsymbol{h}_t^q = g(\boldsymbol{W}_{h^{q-1}h^q}\boldsymbol{h}_t^{q-1} + \boldsymbol{b}^q) \tag{3}$$

$$\boldsymbol{y}_t = \boldsymbol{W}_{h^Q y}\boldsymbol{h}_t^Q + \boldsymbol{b}^y \tag{4}$$

where $\boldsymbol{h}_t^q$ represents the qth hidden layer at time instant, t, $\boldsymbol{W}_{xh^1}$ the weight matrix from the input to the first hidden layer, $\boldsymbol{W}_{h^{q-1}h^q}$ the weight matrix from the $q-1$th hidden layer to the qth hidden layer and $\boldsymbol{W}_{h^Q y}$ the weight matrix from the last hidden layer to the output. Bias vectors in each hidden layer are represented as $\boldsymbol{b}^q$ and as $\boldsymbol{b}^y$ for the output layer. Outputs from the hidden layers are passed through a non-linear activation function, g.

We consider two methods of framing the input linguistic feature vectors, $\boldsymbol{l}_t$, from Table 1, to capture context and form the input vectors, $\boldsymbol{x}_t$, to the DNN. In the first method, which we term *wide window*, the input feature comprises a single linguistic feature vector, $\boldsymbol{l}(M)_t$, with the phoneme/viseme context set to M according to the amount of context desired, i.e.

$$\boldsymbol{x}_t = \boldsymbol{l}(M)_t \tag{5}$$

In the second method, which we term *concatenated window*, the input feature, $\boldsymbol{x}_t$, is created by concatenating K 'narrow' linguistic features, as

$$\boldsymbol{x}_t = [\boldsymbol{l}(1)_{t-\frac{K-1}{2}}, \ldots, \boldsymbol{l}(1)_t, \ldots, \boldsymbol{l}(1)_{t+\frac{K-1}{2}}] \tag{6}$$

The term *narrow* is used as the context width of each linguistic feature is set to $M = 1$. Context is now included in the input feature, $\boldsymbol{x}_t$, through the concatenation of the K linguistic features, $\boldsymbol{l}(1)_t$.

The output vector, $\boldsymbol{y}_t$, from the DNN comprises L AAM vectors, $\boldsymbol{y}_t = [\boldsymbol{a}_{t-\frac{L-1}{2}}, \ldots, \boldsymbol{a}_t, \ldots, \boldsymbol{a}_{t+\frac{L-1}{2}}]$. In the situation where $L > 1$, the predicted AAM vectors that overlap, representing the same time instant, are averaged to form the final output sequence used for animation. All input and output features are normalised to zero mean and unit variance.

The network is trained by minimising a mean squared error loss function using the backpropagation of errors algorithm in conjunction with stochastic gradient descent to learn the weight and bias values (Rojas, 1996). Various hyperparameters were adjusted during training with best performance on a validation test set found with a mini-batch size of 128, a momentum of 0.9, a learning rate of 0.3, a dropout probability of 0.5 to avoid overfitting and g as a rectified linear unit activation function (Nair and Hinton, 2010). The maximum number of epochs was set to 150. Evaluations also investigated the number of hidden layers and numbers of units and found a network comprising three hidden layers each with 3,000 units gave best performance. All deep learning was implemented using the Keras framework with the Theano back-end (Chollet et al., 2015; Theano Development Team, 2016). Section 6.2.2 examines the effect of K and L.

5.2. Recurrent neural network

DNNs are able to model the static mapping between input and output features, but, to account for temporal information, a recurrent neural network (RNN) is proposed. A bi-directional RNN is able to use both past and future

inputs to compute the mapping to output features. However, when training an RNN, the vanishing gradient problem is known to be problematic when modelling long-time feature dependencies, and so this work considers the long short-term memory (LSTM) model (Hochreiter and Schmidhuber, 1997). This uses memory cells to selectively remember previous inputs and to forget those not relevant to the learning task. LSTMs have been effective in several areas of speech processing where requirements to model long temporal dynamics are necessary (Graves et al., 2013; Fan et al., 2015a). Two LSTM architectures are considered in this work, many-to-one and many-to-many.

5.2.1. Many-to-one architecture

The many-to-one architecture takes a sequence of feature vectors, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, as input and outputs a sequence of vectors, $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T\}$, and is illustrated in Figure 2a. Assuming a shallow network (for ease of notation) and iterating over time, the RNN computes predicted output features as

$$\boldsymbol{h}_t = g(\boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}) \tag{7}$$

$$\boldsymbol{y}_t = \boldsymbol{W}_{hy}\boldsymbol{h}_t + \boldsymbol{b}_{\boldsymbol{y}} \tag{8}$$

Compared to Eq. (3) the RNN structure has weight matrix $\boldsymbol{W}_{hh}$ which allows activations from the previous time step, $\boldsymbol{h}_{t-1}$, to affect the current time step and so propagates information across time which allows past events to be modelled. This work uses LSTMs so the units in each layer are replaced with LSTM cells (Hochreiter and Schmidhuber, 1997).

The input features to the many-to-one LSTM are a sequence of K

linguistic features, $\boldsymbol{x}_t = \boldsymbol{l}(1)_t$, where $(t - \frac{K-1}{2}) \leq t \leq (t + \frac{K-1}{2})$. Each $\boldsymbol{x}_t$ contains no contextual information as this will be modelled by the internal recurrent structure of the model. The many-to-one architecture produces a single output vector that contains a concatenation of L AAM vectors, i.e. $\boldsymbol{y}_t = [\boldsymbol{a}_{t-\frac{L-1}{2}}, \ldots, \boldsymbol{a}_t, \ldots, \boldsymbol{a}_{t+\frac{L-1}{2}}]$. The LSTM makes a prediction of L frames at each time instant so the output AAM vectors that represent the same time instant are averaged to form the final sequence used for animation. Section 6.2.2 explores the effect of varying the number of input vectors, K, and number of AAM vectors, L, in each output. Learning of the weights and biases was accomplished by minimising a mean squared error loss function using backpropagation through time (BPTT) training (Rojas, 1996). Best performance on a validation set was found with three hidden layers each with 256 units, a mini-batch size of 128, a momentum of 0.9, a learning rate of 3e-5 and a dropout of probability of 0.5. The maximum number of epochs was set to 150.

5.2.2. Many-to-many architecture

The many-to-many architecture is based on an encoder-decoder structure as this allows input and output sequence lengths to be different. This is desirable as it is likely that longer duration linguistic information will map to a narrower sequence of visual features and this is investigated in Section 6.2.2. The many-to-many model is shown in Figure 2b and exploits the LSTM structure used in the many-to-one system. The model first constructs a context vector by passing input features through the encoder LSTM. The context vector is then input into the decoder LSTM which predicts a sequence of visual features. A sequence of K input vectors, $\boldsymbol{x}_t = \boldsymbol{l}(1)_t$ where $(t - \frac{K-1}{2}) \leq$

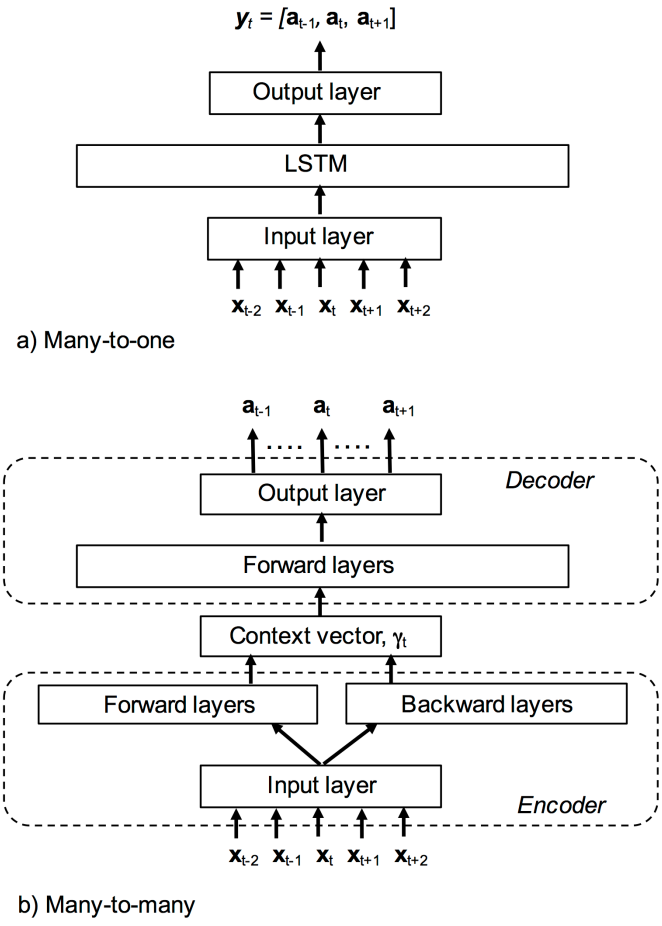

Figure 2: *LSTM architectures showing a) many-to-one and b) many-to-many encoder-decoder, with as an example, $K = 5$ inputs and $L=3$ outputs.*

$t \leq (t + \frac{K-1}{2})$, is passed into the input layer of the encoder, which is a bi-directional LSTM structure comprising forward and backward layers. The outputs are combined in an output layer to create a context vector, $\boldsymbol{\gamma}_t$, which summarises the information from the input sequence in both directions. The context vector is input to the decoder which is formed from a uni-directional LSTM with L output nodes corresponding to L predicted AAM vectors $\{\boldsymbol{a}_{t-\frac{L-1}{2}}, \ldots, \boldsymbol{a}_{t+\frac{L-1}{2}}\}$. A window of K input features is advanced by one frame at each time instant and predicted AAM vectors representing the same time instant are averaged to form the final output sequence used for animation.

Model parameters are learnt by minimising a mean squared error loss function using backpropagation through time (BPTT) training and in a series of tests using validation data, best performance was found with three forward and three backward layers, each comprising 256 units, in the encoder and three forward layers, each comprising 256 units in the decoder. For the context vector, sizes from 256 to 2024 units were tested and best performance found with 768 units. As with the many-to-one LSTM, a mini-batch size of 128, a momentum of 0.9, a learning rate of 3e-5 and a dropout probability of 0.5 were used, and the maximum number of epochs was set to 150.

6. OPTIMISATION AND ANALYSIS TESTS

Objective testing is performed to both optimise and analyse the performance of the various system configurations proposed. The tests examine the effect of different frame-level features, input and output vector sequence lengths, types of speech unit and model architectures. Where dynamic viseme

speech units are employed, these use the reference sequences (as outlined in Section 3.3.2) to avoid errors in DV sequences that could affect the analysis.

Our experiments use the KB-2k audio-visual speech dataset which contains 2,543 phonetically balanced sentences taken from the TIMIT corpus which total around 8 hours of speech (Taylor et al., 2012). The recordings contain the frontal view of a single professional male speaker talking in a reading style with no emotion. The video was captured at 29.97fps and the audio at a sampling rate of 48kHz. Each of our linguistic feature vectors are therefore extracted every $1/29.97 = 33$ms. Of the 2,543 sentences, 2,035 are used for training, 254 for evaluation and 254 for testing to give an 80:10:10 split.

To evaluate the effectiveness of predicting AAM features we use correlation, normalised root mean square error (NRMSE) and global variance (GV) as metrics (Wang et al., 2011), which are defined as

$$\text{Corr} = \frac{1}{N \times D} \sum_{t=1}^{N} \sum_{j=1}^{D} \frac{(a(j)_t - \mu_a(j))(\hat{a}(j)_t - \mu_{\hat{a}}(j))}{\sigma_a(j)\sigma_{\hat{a}}(j)} \tag{9}$$

$$\text{NRMSE} = \frac{1}{D} \sum_{j=1}^{D} \sqrt{\frac{1}{N} \sum_{t=1}^{N} \left[\frac{(a(j)_t - \hat{a}(j)_t)}{a_{max}(j) - a_{min}(j)} \right]^2} \tag{10}$$

$$\text{GV} = \frac{1}{N \times D} \sum_{t=1}^{N} \sum_{j=1}^{D} [\hat{a}(j)_t - \mu_{\hat{a}}(j))]^2 \tag{11}$$

where $a(j)_t$ and $\hat{a}(j)_t$ represent the jth element of the tth reference and predicted AAM vectors, $\mu_a(j)$, $\mu_{\hat{a}}(j)$, $\sigma_a(j)$ and $\sigma_{\hat{a}}(j)$ represent their respective means and standard deviations, and $a_{max}(j)$ and $a_{min}(j)$ represent their

maximum and minimum values. N is the number of vectors under test and D is the number of coefficients in the AAM vectors that are measured. Correlation and NRMSE are commonly used metrics but we also include GV as we find this to correlate better with the subjective test results in Section 7. We chose to base the objective measures on the first five coefficients (i.e. $D = 5$) of the predicted and reference AAM sequences as their effect on the resulting facial animation was clearly perceptible. We found that higher order coefficients had much less effect on the animations and as we wished for the objective performance to, ideally, correlate with subjective performance we limited to measure the first five coefficients, which represent 80% of the variance.

6.1. Analysis of frame-level features

In this experiment we investigate the importance of the various frame-level features defined in Table 1. As a baseline, we focus on the effect of phoneme-based frame-level features using the feedforward DNN described in Section 5.1 with the wide window framing method and the phoneme context set to either M=1 or M=11 (tests in Section 6.2.2 show this to be a suitable value). Seven combinations of features are considered and these are defined in Table 2. All of these frame-level combinations are augmented with segment, syllable, word, phrase and utterance features. The correlation, NRMSE and GV for each combination is shown in Table 3. Note that better performance is indicated by high correlation and GV values and low NRMSE.

The features in System A encode just the current phoneme identity, with no wider phoneme context (i.e. M=1), along with the position and number of frames in the current phoneme. As this contains no contextual

Table 2: *Frame-level feature combinations.*

System	Phoneme context (M)	Position in phoneme	No. frames in phoneme	Acoustic class	Forward phoneme span
A	1	×	×		
B	11				
C	1			×	
D	11			×	×
E	11	×	×		×
F	11	×	×	×	
G	11	×	×	×	×

Table 3: *Correlation, NRMSE and global variance of seven frame-level feature combinations using a phoneme-based feedforward DNN model (brackets show ±1 standard deviation).*

	Correlation	NRMSE	GV
System A	0.73(0.07)	10.54(2.14)	802.89(162.42)
System B	0.80(0.07)	9.53(2.19)	920.22(174.20)
System C	0.73(0.08)	10.61(2.16)	840.36(174.40)
System D	**0.81(0.07)**	9.45(2.12)	922.10(189.92)
System E	0.79(0.07)	9.65(2.24)	955.86(190.35)
System F	**0.81(0.07)**	9.31(2.22)	952.40(183.90)
System G	**0.81(0.07)**	**9.26(2.22)**	**964.55(200.50)**

information, this is one of the worst performing feature combinations with the predicted AAM sequences being step-like with discontinuous trajectories. This is illustrated as the blue dotted line in Figure 3 which shows the first AAM coefficient from the utterance '*If dark came they would lose her*'. The features in System B encode the phoneme identities of the neighbouring 5 frames in both directions from the current phoneme (M=11). With this additional phonetic context the model is able to predict AAM sequences that more closely match the reference features as seen in the improved objective scores. This is illustrated as the red line in Figure 3 which shows the predicted trajectory to be much closer to the original trajectory than the output from System A. System C supplements the current phoneme identity with the acoustic class feature, but again this performs poorly due to the lack of contextual information (i.e. M=1) and leads to a discontinuous output. Given the effectiveness of the phoneme context feature, Systems D-G all include this with various combinations of the other frame-level features. The results show that using all frame-level features gives best performance, but indicate that phoneme context together with acoustic class play a crucial role.

6.2. Windowing and speech unit selection

In Section 5.1 we introduced both *wide* windowing of linguistic features (each input feature vector encodes a context of M frames) and *concatenated* windowing (each input feature vector is a concatenation of K narrow-context feature vectors). In this section we examine which windowing method is better for the different model architectures. Additionally, we measure the effect of varying the size of the input and output context windows and the type of speech unit.

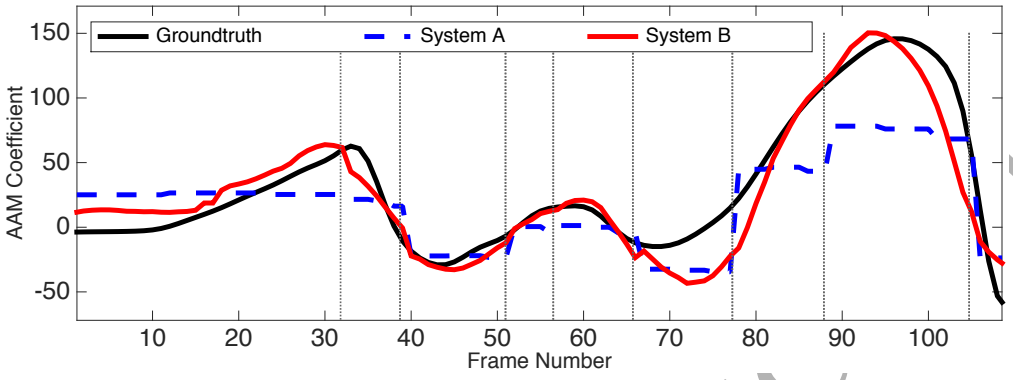

Figure 3: *The first dimension of the reference and predicted AAM features for the sentence 'If dark came they would lose her', both with phonetic context (System B) and without (System A). Vertical lines show the phoneme boundaries.*

6.2.1. Analysis of windowing method

We now compare the wide and concatenated window methods of creating input features using the feedforward DNN proposed in Section 5.1, and the many-to-one and many-to-many (encoder-decoder) LSTM architectures as introduced in Section 5.2. These tests use the combined phoneme and dynamic viseme frame-level feature set. For the wide window features the context width is set to M=11 and for the concatenated window and LSTM experiments the value K=11 is used, thereby giving equivalent context widths for all systems. In all cases we train the model to predict an output of L=3 concatenated AAM feature vectors which represents approximately 100ms of facial motion. These values are shown in Section 6.2.2 to give good performance.

Table 4 shows the performance of the four configurations in terms of correlation, NRMSE and GV. Our results indicate that, for the DNN, features extracted using the wide windowing method clearly outperform those extracted using the concatenated windowing approach. One explanation for

this is that although the same phoneme and DV context is provided in both sets of input features, the concatenated window feature vector is larger and introduces redundancy by the concatenation of consecutive single-context vectors that partially encode the same information. For the LSTM, the many-to-many architecture outputs a more accurate AAM sequence than the many-to-one architecture. This can be attributed to the output layer of the many-to-many model explicitly modelling the visual co-articulation of the output sequence. The correlation and NRMSE of the wide window DNN and the many-to-many LSTM are very similar, but the GV of the LSTM is substantially higher. Observing animations from the two models reveals those from the LSTM to be more realistic.

Model	Correlation	NRMSE	GV
DNN: wide window	**0.88(0.04)**	**6.02(1.17)**	1282(225)
DNN: concatenated window	0.85(0.05)	6.78(1.42)	1033(183)
LSTM: many-to-one	0.86(0.05)	6.43(1.20)	1290(267)
LSTM: many-to-many	**0.88(0.05)**	6.07(1.27)	**1478(268)**

Table 4: Performance of a DNN trained using wide and concatenated windowed input features, and an LSTM using many-to-one and many-to-many architectures. All models use an input context of 11 vectors and an output context of 3 vectors.

6.2.2. Analysis of input and output window widths

The effect of varying the amount of context encoded in the input linguistic features and the output visual features is now investigated by computing the correlation, NRMSE and GV for contexts of increasing duration. We consider the best performing models from the previous experiment, namely the DNN and the many-to-many LSTM trained using the combined phoneme

and dynamic viseme speech features. Figure 4 shows the NRMSE and GV as the input context width, K, is varied between 1 (33ms) and 23 (759ms) with a fixed output context of 3 for the two systems. For compactness, correlation is not shown as this exhibited less variation, although a similar trend. NRMSE reduces as the sequence length is increased and more temporal information is included in the input. GV also improves with increasing sequence length for the LSTM, while for the DNN a peak between K=9 and K=11 is observed. We visualised the resulting animations and established that a value of K=11 for the DNN and K=17 for LSTM gave best performance.

We consider next the effect of varying the size of the output layer of the DNN and many-to-many LSTM, which corresponds to how many AAM feature vectors are output at each time instant. Figure 5 shows NRMSE and GV as the number of time steps, L, in the output layer is varied between 1 (33ms) and 9 (300ms). These do not show such a clear trend as for varying, K, as now NRMSE reduces as L is increased to 5 and then fluctuates. Conversely, GV reduces up to L=5 and then increases for the LSTM and decrease from L=3 for the DNN. Examining the AAM trajectories and observing the resulting animations for different values of L confirm that the realism of the predicted animation is less sensitive to varying L but does suggest that a value of L=3 vectors gives best performance for both systems.

6.3. Analysis of speech units

A comparison is now made between using phonemes, static visemes and dynamic visemes, or a combination, as the speech unit. For phoneme units all the PH features shown in Table 1 are included, while for both (static) viseme and dynamic viseme units all the V features are included. Tests are conducted

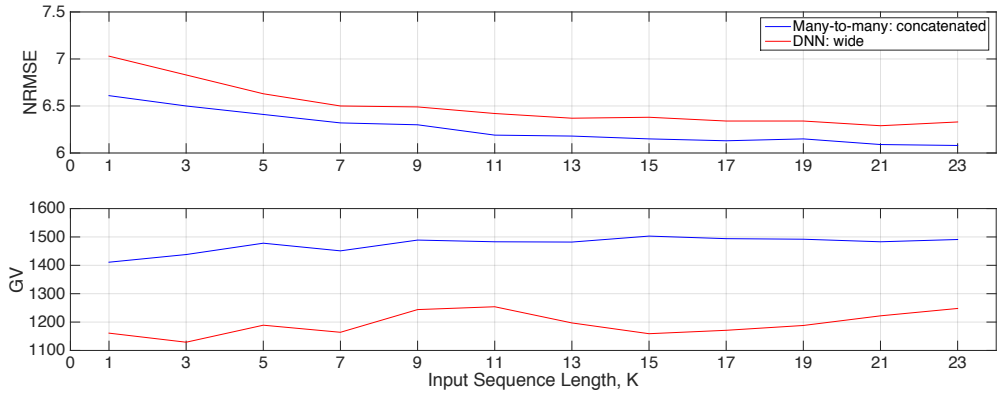

Figure 4: *NRMSE and GV as input sequence length, K, is varied within the DNN and many-to-many encoder-decoder LSTM architectures.*

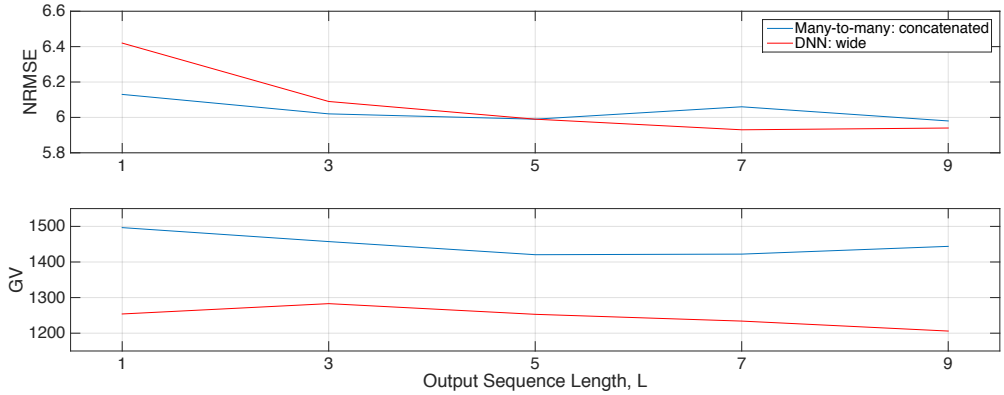

Figure 5: *NRMSE and GV as output sequence length, L, is varied within the DNN and many-to-many encoder-decoder LSTM architectures.*

using the DNN with wide windowing and the many-to-many LSTM, using the input and output contexts determined in Section 6.2.2.

The correlation, NRMSE and GV are shown in Table 5. For both models we determine that using phonemes as the speech unit is better than using static visemes. However, dynamic visemes outperform both static visemes and phonemes as the speech unit for both models. Analysis of sequences of estimated AAM features shows a primary benefit of dynamic viseme units is improved synchrony in relation to the reference AAM features compared to phoneme units and static viseme units (which themselves are derived directly from phoneme units). We expect that this is because DVs describe the facial movements in a sequence rather than the sounds and so they provide the model with more discriminant information regarding the expected facial position. These results suggests that using a well defined visual speech unit is beneficial over an acoustically motivated speech unit. Although dynamic viseme units improve synchrony they were found to sometimes produce an over-smoothed output which gives an under articulated animation. Combining dynamic visemes with phoneme units alleviated this problem as having complementary information relating to acoustic and visual information was beneficial and this gave best performance for both the DNN and LSTM models.

7. PRACTICAL TESTING AND COMPARISON

We present now a formal comparison of a representative set of system configurations on a set of held out test sentences using both objective measures and subjective preference tests. Furthermore, we also consider the practical situation when no reference dynamic viseme sequence is available. In this

Table 5: *Effect of using phoneme, static viseme, dynamic viseme or combined phoneme/dynamic viseme speech units for DNN and LSTM systems.*

Model	Speech unit	Correlation	NRMSE	GV
DNN	Phoneme	0.83(0.06)	7.43(1.84)	992(196)
DNN	Static viseme	0.81(0.06)	7.77(1.82)	903(177)
DNN	Dynamic viseme	0.82(0.05)	7.01(1.21)	1072(231)
DNN	PH + DV	**0.88(0.04)**	**6.02(1.17)**	1282(225)
LSTM	Phoneme	0.82(0.07)	7.53(1.90)	1240(246)
LSTM	Static viseme	0.81(0.07)	7.73(1.89)	1197(245)
LSTM	Dynamic viseme	0.83(0.05)	6.82(1.20)	1362(286)
LSTM	PH + DV	**0.88(0.05)**	6.07(1.27)	**1478(268)**

situation the dynamic viseme sequence is generated automatically using the procedure described in Section 3.3.2, with the full system architecture shown in Figure 6. A phoneme-based visual synthesiser first generates an AAM sequence from the input phonetic representation. A sequence of dynamic visemes is then obtained from this sequence and these, combined with the phoneme sequence, are input into the joint phoneme/DV visual synthesiser.

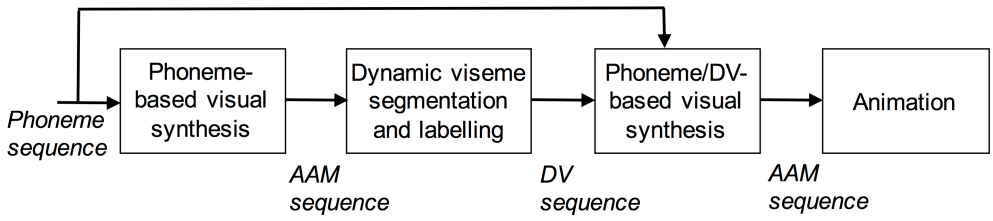

Figure 6: *Full architecture for automated animation from input phoneme sequence.*

The tests compare two model architectures, namely the DNN using a wide window of features with M=11 and L=3, and the many-to-many LSTM with K=17 and L=3. For each model, four variants of speech unit are tested: phoneme-only (PH), dynamic viseme-only (DV), combined

phoneme/dynamic viseme (PH_DV) and combined phoneme/dynamic viseme but using the predicted dynamic viseme sequence rather than the reference sequence (PH_DV_PRED). These tests allow us to determine which system is best if the reference dynamic viseme sequence is not available - i.e. is it better to use phoneme-only speech units or the predicted dynamic viseme units. As a further comparison, a baseline HMM system is included which is similar to that proposed in (Schabus et al., 2014). This uses five state hidden semi-Markov models with each state modelled by a single Gaussian with diagonal covariance. Quinphone HMMs were created using decision tree clustering that considered phoneme, syllable, word, phrase and utterance level questions and resulted in 11,893 models (Schabus et al., 2014). The systems evaluated in the comparative tests are specified in Table 6 which shows the model, speech unit and how, if included, the dynamic viseme sequence is obtained.

Table 6: *System configurations used in comparative tests showing model, speech unit and the source of the dynamic viseme sequence.*

System	Model	Speech unit	DV labels
HMM_PH	HMM	Phoneme	-
HMM_DV	HMM	DV	Reference
DNN_PH	DNN	Phoneme	-
DNN_DV	DNN	DV	Reference
DNN_PH_DV	DNN	Phoneme + DV	Reference
DNN_PH_DV_PRED	DNN	Phoneme + DV	Predicted
LSTM_PH	LSTM	Phoneme	-
LSTM_DV	LSTM	DV	Reference
LSTM_PH_DV	LSTM	Phoneme + DV	Reference
LSTM_PH_DV_PRED	DNN	Phoneme + DV	Predicted

7.1. Objective comparison

Table 7 shows the correlation, NRMSE and GV for the ten systems shown in Table 6. In terms of correlation and NRMSE, the DNN and LSTM systems have similar performance across the respective speech units. However, the global variance of vectors produced by the LSTM systems is higher than that for DNN systems, for the same respective speech units. A higher global variance could be the result of more noisy vectors being predicted, but examining the trajectories and looking at the resulting animations has shown this not to be the case. Instead, the higher global variance of visual vectors produced by the LSTM systems give a more articulated and natural appearance compared to those from the DNN with lower global variance. Considering now the HMM systems, these perform substantially worse than the DNN and LSTM systems across all measures.

Comparing systems using dynamic viseme speech units (HMM_DV, DNN_DV and LSTM_DV) to those using phoneme speech units (HMM_PH, DNN_PH and LSTM_PH) shows that across all models, as a single speech unit, DVs are more effective than phonemes. Both the DNN and LSTM systems perform substantially better with a combined phoneme/dynamic viseme speech unit when the dynamic viseme sequences are derived from reference video. When using the dynamic viseme sequences that are generated automatically, correlation and NRMSE both worsen although global variance is effected less, and actually increases for the DNN system. In the absence of reference dynamic viseme sequences, the phoneme-based systems (*_PH) outperform the phoneme+predicted dynamic viseme systems (*_PH_DV_PRED) in terms of both correlation and NRMSE, while inversely, global variance is

substantially better for the latter. This is a rather mixed result, and one that is resolved in the subsequent subjective testing.

Table 7: *Objective comparisons showing correlation, NRMSE and global variance.*

System	Correlation	RMSE	GV
HMM_PH	0.78(0.08)	8.17(1.88)	849.22(165.91)
HMM_DV	0.81(0.06)	7.37(1.48)	971.26(197.44)
DNN_PH	0.83(0.06)	7.43(1.84)	992.09(196.22)
DNN_DV	0.82(0.05)	7.01(1.21)	1072.79(212.62)
DNN_PH_DV	**0.88(0.04)**	**6.02(1.17)**	1282.10(225.14)
DNN_PH_DV_PRED	0.81(0.07)	7.91(1.90)	1397.48(250.35)
LSTM_PH	0.82(0.07)	7.53(1.90)	1240.97(245.56)
LSTM_DV	0.83(0.05)	6.82(1.20)	1362.38(285.76)
LSTM_PH_DV	**0.88(0.05)**	6.07(1.27)	**1478.21(268.81)**
LSTM_PH_DV_PRED	0.80(0.07)	7.98(1.94)	1425.82(255.84)

7.2. Subjective comparison

We use subjective preference tests to compare the naturalness of pairs of animations that have been obtained from different combinations of four systems defined in Table 6. We also include ground truth facial animations that were reconstructed from AAM features extracted from the original video (ORIG). In total eight different system pairs are compared as shown in the first two columns of Table 8. In the first three system pairs the ground truth is compared against the baseline HMM and the best performing DNN and LSTM systems. A further three tests then compare the HMM_PH, DNN_PH_DV and LSTM_PH_DV systems with one another. The final two tests consider the situation where reference dynamic viseme sequences are not available. In particular, the LSTM based on phoneme and reference DVs (LTSM_PH_DV) is first compared to a phoneme-only LSTM (LTSM_PH)

and then to an LSTM that uses phonemes and automatically generated DVs (LTSM_PH_DV_PRED).

Evaluation is carried out by playing animations from two systems simultaneously and side-by-side to subjects using a web-based interface. For each pair of videos, subjects are presented with the question "Which one looks more natural?", and they then make their choice by selecting the appropriate video. In preliminary tests we found playing the two animations simultaneously to be more effective than playing the two animations one after the other, as seeing the two animations together made it easier for subjects to identify differences and consequently the effect on naturalness. Furthermore, subjects were allowed to play the animations as many times as they wished before scoring. We decided upon this comparative scoring of naturalness as the differences in animations from the various systems was often quite subtle and an absolute score was found to be less effective at highlighting differences in naturalness.

A total of thirty participants took part in the tests and each was played four examples from each pair of eight system comparisons, with sentences chosen randomly and played in a random order. Table 8 shows the preference results for the systems compared and also reports on whether the result is statistically significant at the 99% confidence level using a binomial test with Holm-Bonferroni correction (Gravetter and Wallnau, 2012; Holm, 1979). We used the Holm-Bonferroni method to control the family-wise error rate (FWER) for the eight statistical significance tests at the 99% confidence level in Table 8.

Compared to animations using the ground truth AAM sequences, both

the HMM and DNN systems are significantly less natural. However, the naturalness of animations produced using the LSTM are much closer to those using the original AAM sequences and found not to be statistically different. Comparing the naturalness of the HMM, DNN and LSTM systems, those from the DNN are significantly better than from the HMM, while those from the LSTM are significantly better than those from both the HMM and DNN. These reflect the results found in the objective tests.

Considering now the situation where reference dynamic viseme sequences are not available, the AAM sequences produced from the system using just phoneme-based speech units (LTSM_PH) leads to animations that are significantly less natural than when including the reference dynamic visemes (LTSM_PH_DV). The final test shows that using automatically predicted dynamic viseme sequences (LTSM_PH_DV_PRED) instead of using the reference dynamic viseme sequences (LTSM_PH_DV) gives slightly lower naturalness although the difference is not statistically significant. This suggests that the predicted dynamic viseme sequences are suitable for producing natural animations and that this is better than using just phoneme-based speech units when no reference dynamic viseme sequences are available. This observation aligns with the global variance scores in the objective tests in Table 7, which were somewhat contrary to the correlation and NRMSE results. This suggests that, providing correlation and NRMSE are sufficiently good (i.e. that the predicted vectors are not simply noisy), then global variance is an effective predictor of the naturalness of the animations and can reflect the amount of articulation present.

To demonstrate further the effect of using the predicted dynamic viseme

Table 8: *Subjective preference test results for eight different combinations of systems showing also whether the result is statistically significant.*

System A	System B	Percentage	Significant
ORIG	HMM_PH	**83.47/16.53**	✓
ORIG	DNN_PH_DV	75.63/24.37	✓
ORIG	LSTM_PH_DV	52.99/47.01	×
HMM_PH	DNN_PH_DV	34.71/65.29	✓
HMM_PH	LSTM_PH_DV	20.49/79.51	✓
DNN_PH_DV	LSTM_PH_DV	29.75/70.25	✓
LSTM_PH_DV	LSTM_PH	68.33/31.67	✓
LSTM_PH_DV	LSTM_PH_DV_PRED	56.30/43.70	×

sequences, Figure 7 shows the first AAM feature generated using the reference (LSTM_PH_DV) and predicted (LSTM_PH_DV_PRED) dynamic viseme sequences, along with the corresponding ground truth trajectory for the phrase '*steve collects rare and novel coins*'. The features generated from the reference dynamic viseme sequence follow more closely the ground truth trajectory than those generated using the predicted sequence, although the underlying structure is clearly captured in both cases. In addition, Figure 8 shows facial animations that are reconstructed from the ground truth and predicted visual features for the shortened phrase '*collects rare and novel*'. The resulting animations using the reference and predicted dynamic visemes look more similar than the AAM sequences would suggest, although it is interesting to observe the slight over-articulation of the word '*rare*'.

8. CONCLUSION

This work has proposed a many-to-many encoder-decoder LSTM architecture using dynamic viseme speech units for predicting AAM vector

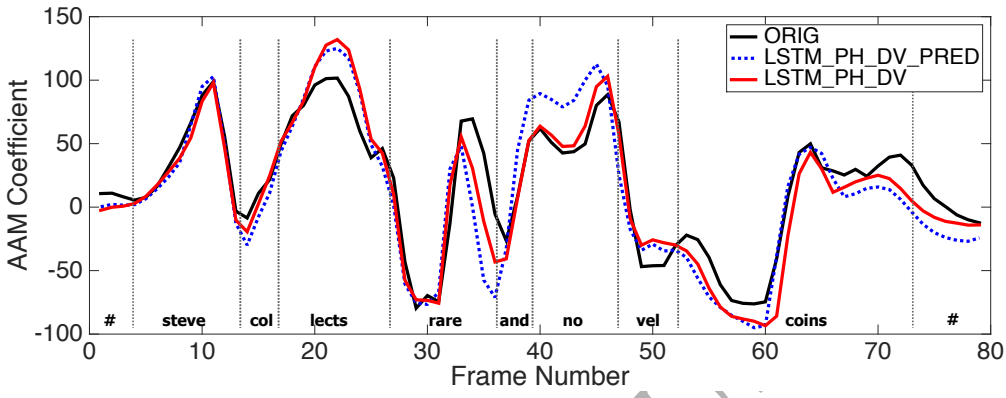

Figure 7: *AAM trajectories generated using reference and predicted dynamic viseme sequences, and for comparison the reference AAM track, for the phrase* 'steve collects rare and novel coins'.

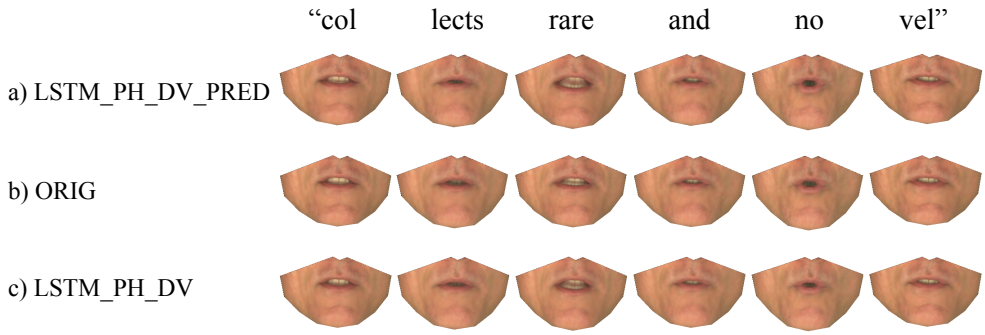

Figure 8: *Sequence of faces taken from animations produced using predicted (a) and reference (c) dynamic viseme sequences, and for comparison using the reference AAM features (b), for the phrase* 'collects rare and novel'.

sequences from a linguistic input. Objective and subjective tests have shown this to outperform a number of other approaches including a many-to-one LSTM, feedforward DNN and an HMM. Dynamic viseme speech units were also found to be superior over phoneme or static viseme units as they are better able to represent visual speech. When combined with phoneme units even better performance was obtained due to the complimentary information they represent. Moving to a more practical scenario, where reference dynamic viseme sequences are unavailable and instead have to be predicted, subjective preference tests found no significant difference between animations when the DV sequences were generated automatically as opposed to using reference sequences. Interestingly, this result was reflected in the global variance objective measure but not in the correlation or NRMSE. This suggests that GV is a useful predictor of the naturalness of animations from AAM vectors, provided that correlation and NRMSE are sufficiently good to indicate that the high global variance is not the result of a more noisy sequence of visual vectors. In terms of the linguistic context in the input to the models, all features were found to make a contribution to performance but in particular a wide phonetic context was important for creating smooth outputs and improving co-articulation.

In terms of further work, the evaluations in Section 7 have shown that performance reduces when moving from reference to automatically generated dynamic viseme sequences. It would therefore be beneficial to explore improved ways to generate automatically the dynamic viseme sequence from the linguistic input. Furthermore, it would also be useful to combine the proposed joint phoneme/dynamic viseme speech unit with more advanced

deep learning architectures, such as have found recent success in acoustic speech synthesis (for example (Wang et al., 2017)), with the aim of further improving the naturalness of animation. The work presented has concentrated on synthesising neutral speech and it would also be interesting to extend the proposed systems to produce more expressive speech.

REFERENCES

Almajai, I., Milner, B., Darch, J., 2006. Analysis of correlation between audio and visual speech features for clean audio feature prediction in noise. In: Interspeech. pp. 2470–2473.

Anderson, R., Stenger, B., Wan, V., Cipolla, R., 2013. Expressive visual text-to-speech using active appearance models. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3382–3389.

Bregler, C., Covell, M., Slaney, M., 1997. Video rewrite: Driving visual speech with audio. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '97. pp. 353–360.

Brugnara, F., Falavigna, D., Omologo, M., Aug. 1993. Automatic segmentation and labeling of speech based on hidden Markov models. Speech Communication 12 (4), 357–370.

Chollet, F., et al., 2015. Keras. `https://keras.io`.

Cohen, M., Massaro, D., 1994. Modeling coarticualtion in synthetic visual speech. In: Thalmann, N., D, T. (Eds.), Models and Techniques in Computer Animation. Springer-Verlag, pp. 141–155.

Cootes, T. F., Edwards, G. J., Taylor, C. J., jun 2001. Active appearance models. IEEE Trans. on Pattern Analysis and Machine Intelligence 23 (6), 681–685.

De Martino, J. M., Magalhães, L. P., Violaro, F., 2006. Facial animation based on context-dependent visemes. Journal of Computers and Graphics 30 (6), 971 – 980.

Deena, S. P., 2012. Visual speech synthesis by learning joint probabilistic models of audio and video. Ph.D. thesis, School of Computing Sciences, The University of Manchester.

Ding, C., Xie, L., Zhu, P., 2014. Head motion synthesis from speech using deep neural networks. Multimedia Tools and Applications 74 (22), 9871–9888.

Ezzat, T., Geiger, G., Poggio, T., 2002. Trainable videorealistic speech animation. In: Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '02. pp. 388–398.

Fan, B., Wang, L., Soong, F. K., Xie, L., 2015a. Photo-real talking head with deep bidirectional LSTM. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP). IEEE, pp. 4884–4888.

Fan, B., Xie, L., Yang, S., Wang, L., Soong, F. K., 2015b. A deep bidirectional LSTM approach for video-realistic talking head. Multimedia Tools and Applications, 1–23.

Filntisis, P., Katsamanis, A., Tsiakoulis, P., Maragos, P., 2017. Video-realistic expressive audio-visual speech synthesis for the Greek language. Speech Communication 95, 137–152.

Fisher, C. G., 1968. Confusions among visually perceived consonants. Journal of Speech and Hearing Research 11, 796–804.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Gower, J., 1975. Generalized procrustes analysis. Psychometrika 40, 33–51.

Graves, A., Jaitly, N., Mohamed, A.-r., 2013. Hybrid speech recognition with deep bidirectional LSTM. In: IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). pp. 273–278.

Gravetter, F. J., Wallnau, L. B., 2012. The binomial test. In: Statistics for the Behavioral Sciences, 9th Edition. Wadsworth Publishing, Belmont, CA, USA, Ch. 19, pp. 644–660.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9 (8), 1735–1780.

Holm, S., 1979. A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics 6, 65–70.

Hunt, A., Black, A., 1996. Unit selection in a concatenative speech synthesis system using a large speech database. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 373–376.

Kim, T., Yue, Y., Taylor, S., Matthews, I., 2015. A decision tree framework for spatiotemporal sequence prediction. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 577–586.

Lidestam, B., Beskow, J., August 2006. Visual phonemic ambiguity and

speechreading. Journal of Speech, Language and Hearing Research (JSLHR) 49, 835–847.

Luo, C., Yu, J., Li, X., Wang, Z., July 2014. Realtime speech-driven facial animation using Gaussian mixture models. In: Proceedings of the Multimedia and Expo Workshop (ICMEW). pp. 1–6.

Mattheyses, W., Latacz, L., Verhelst, W., 2011. Automatic viseme clustering for audiovisual speech synthesis. In: Proceedings of Interspeech. pp. 2173–2176.

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

Nair, V., Hinton, G. E., 2010. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10). pp. 807–814.

Parke, F. I., Waters, K., 1996. Computer Facial Animation. CRC Press.

Parker, J., Maia, R., Stylianou, Y., Cipolla, R., 2017. Expressive visual text to speech and expression adaptation using deep neural networks. In: ICASSP. pp. 4920–4924.

Rojas, R., 1996. The backpropagation algorithm. In: Neural Networks: A Systematic Introduction. Springer-Verlag, Ch. 7, pp. 151–184.

Schabus, D., Pucher, M., Hofer, G., April 2014. Joint audiovisual hidden semi-

Markov model-based speech synthesis. IEEE Journal of Selected Topics in Signal Processing 8 (2), 336–347.

Schuster, M., Paliwal, K. K., 1997. Bidirectional recurrent neural networks. IEEE Transactions Signal Processing 45, 2673–2681.

Shoup, J. E., 1980. Phonological aspects of speech recognition:. Prentice-Hall.

Taylor, S., Kim, T., Yue, Y., Mahler, M., Krahe, J., Rodriguez, A., Hodgins, J., Matthews, I., Jul. 2017. A deep learning approach for generalized speech animation. ACM Trans. on Graphics 36 (4), 270–287.

Taylor, S., Milner, B., Kato, A., 2016. Audio-to-visual speech conversion using deep neural networks. In: Proceedings of Interspeech. pp. 1482–1486.

Taylor, S. L., Mahler, M., Theobald, B.-J., Matthews, I., 2012. Dynamic units of visual speech. In: Eurographics/ACM SIGGRAPH Symposium on Computer Animation. pp. 275–284.

Thangthai, A., Theobald, B., 2015. HMM-based visual speech synthesis using dynamic visemes. In: Proceedings of the Conference on Facial Analysis, Animation and Auditory-Visual Speech Processing (FAAVSP). pp. 88–92.

Theano Development Team, May 2016. Theano: A Python framework for fast computation of mathematical expressions. arXiv e-prints abs/1605.02688. URL `http://arxiv.org/abs/1605.02688`

Tokuda, K., Zen, H., Black, A. W., Sept 2002. An HMM-based speech synthesis system applied to English. In: Proceedings of IEEE Workshop on Speech Synthesis. pp. 227–230.

Wang, L., Wu, Y. J., Zhuang, X., Soong, F. K., May 2011. Synthesizing visual speech trajectory with minimum generation error. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 4580–4583.

Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., J. Weiss, R., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., A. Saurous, R., 2017. Tacotron: A fully end-to-end text-to-speech synthesis model. In: Interspeech. pp. 1–2.

Watts, O., Henter, G., Merritt, T., Wu, Z., King, S., 2016. From HMMs to DNNs: where do the improvements come from? In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP). pp. 5505–5509.

Zen, H., Nose, T., Yamagishi, J., Sako, S., Masuko, T., Black, A. W., Tokuda, K., 2007. The HMM-based speech synthesis system version 2.0. In: Proceedings of ISCA Workshop on Speech Synthesis (SSW6). pp. 294–299.

Zen, H., Senior, A., Schuster, M., 2013. Statistical parametric speech synthesis using deep neural networks. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). pp. 7962–7966.

Zen, H., Tokuda, K., Black, A., Nov. 2009. Statistical parametric speech synthesis. Speech Communication 51 (11), 1039–1064.