A Hybrid Temporal Modeling Phoneme Recognized Network for Real-time Speech Animation

Zixiao Yu[1], Haohong Wang[2] and Jian Ren[1]
[1]Department of ECE, Michigan State University, East Lansing, MI 48824-1226.
Email: {yuzixiao,renjian}@msu.edu
[2]TCL Research American, 2025 Gateway Place, Suite 460, San Jose, CA 95110
Email: haohong.wang@tcl.com

***Abstract*—Real-time speech animation is a compelling but challenging topic, where multimodal inputs such as audio, video or depth information, have been considered to leverage performance. However, in conditions that audio is the only available input, the quality of the outcome relies heavily on the real-time phoneme recognition. In this paper, we introduce a novel deep-learning based Realtime Phoneme Recognition Network architecture to leverage spatial, separate temporal and unified temporal modeling in the input audio data. Our network can achieve super performance in phoneme recognition. Our empirical results show that compared to the state-of-the-art algorithms, RealPRNet can achieve up to 20% PER improvement and 4% animation quality improvement based on the subject tested.**

***Index Terms*—Real-time speech animation, phoneme recognition, deep-learning**

I. Introduction

The virtual characters have been very popular nowadays in streaming video, gaming, and virtual reality related applications. These characters communicate with each other in a virtual world, and sometimes they interact with the viewers or players in the physical world. Human beings are very sensitive to any facial artifacts, uncoordinated or unsynchronized performance of virtual characters, which make facial animation, in particular speech animation production, very challenge since animation simultaneously involves voice and mouth movement.

The realistic speech animation [1], [2] has been always very compelling, where artists seek the most immersive experiences with high-fidelity human-like virtual characters. However, the high cost involved in the production process as well as the huge data requirements significantly undermine its scaling potential. For applications that accept lower realism effects but requires good scaling possibility, audio data could become the only media that is available during the process. In such a scenario, the virtual characters are required to behave properly with mimic mouth movement matching with the voice input seamlessly, this is what we call it believable speech animation. The "believable" speech animation requires that the algorithm works, under practical resource constraints, for all possible virtual faces with various 3D models and produces synthesized video with sufficient realism for users on the remote end to feel comfortable.

Recent research [3] has demonstrated that real-time believable speech animation is achievable. In [3], the audio input is segmented into small pieces called audio frames, and fundamental frequency features are extracted from each frame. Phonemes are predicted by recognizing vowels and basic fricative consonants from the features, and then mapped into the corresponding static animation called viseme (its counterpart in the visual domain [4]). In this frame-based processing mechanism, the latency is negligible as it almost equals to the processing time of a single frame. However, lack of considering on neighborhood context information during the process significantly limited the recognition accuracy of the phoneme as the system can only recognize basic phonemes, which contributed directly to the low quality of the generated animation. In [5], a word-based processing mechanism is adopted to achieve much higher quality animation because a group of richer phonemes (around 39 different phonemes) can be considered in this scenario and compared to the fundamental phonemes (below 10 different phonemes). However, on the downside, the latency of word-level duration becomes unacceptable for real-time applications. Hence, if a midway between frame-based and word-based approaches can be found, the problem of real-time believable speech animation might be resolved.

With the latest advances in deep learning, the phoneme recognition topic has been revisited using completely new methodologies [6], [7]. Compared with the traditional models, such as Hidden Markov models (HMM) (error rate 26.61%) [8], these deep-learning neural network (DNN) based approaches generally decreases the predicted phoneme error rate by 10%. In [7], it was reported that a feed-forward deep neural network architecture achieved a lower error rate of 16.49%, and a 4 layers standard Long Short-Term Memory (LSTM) network achieved even better result, 15.02%. In [6], a network architecture called CLDNN that combines Convolutional Neural Network (CNN), LSTM and fully connected DNN was proposed. It can further improve the performance of 4-6% compared to the LSTM.

For real-time applications, to find the balance between latency and accuracy is very critical as indicated in the design philosophy of RNN and LSTM. The temporal correlation between neighbor audio frames can play very significant roles

978-1-7281-4272-2/20/$31.00 ©2020 IEEE
DOI 10.1109/MIPR49039.2020.00019

in recognition accuracy improvement and. In this work, we propose a novel deep network architecture, called RealPRNet, to address this problem. In RealPRNet, a novel concept called LSTM Stack Block (LSB), which follows the CNN layer, is introduced to maximize the learning efficiency by performing separate temporal modeling on every different temporal-spatial patterns. After CNN filter and LSB learn the spatial and separate temporal correlation between these frequency signals input, the unified temporal modeling is performed by LSTM. LSB separately processes the intermediate output features from different CNN filters, each output feature will pass to an individual LSTM in the LSB.

To build an end-to-end audio-driven real-time believable speech animation system, the RealPRNet is inserted into a typical procedural speech animation process that converts the audio input into a recognized phoneme sequence and drives a facial animation module. Inspired by the JALI model [5] and properties of the blend shape facial model, the animation is achieved by mapping the recognized phoneme label to a set of parameters to control four basic blend shapes that have hidden physical correlation.

The remaining part of the paper is organized as follows: In Section II, we describe the details of our animation production system framework and introduce the architecture of RealPRNet with our insight in Section III. We present the experimental setup and the evaluation results in Section IV. In Section V, we conclude our work.

II. System Design

Our system is designed as a real-time version of the procedural system that recognizes the phoneme stream first from the audio input, then maps it to the corresponding parameter stream and drives the 3D facial models.

As shown in Fig. 1, When an audio signal is received by the system, it is transformed into the corresponding MFCCs features, which is the input of RealPRNet. The RealPRNet predicts the phoneme stream and then uses it to produce the corresponding animation curves (Jaw and Lip parameters) that drive the 3D facial model.

To support real-time interactivities, the latency between receiving audio input and outputting animation is required to be controlled below certain limits. A buffer mechanism thus is adopted in the system to dynamically determine the size of the sliding window to ensure that the latency limit is never exceeded as shown in Fig. 1.

A. Input Features

When the raw audio signals are received in the system, they are transformed into frequency-domain signals called MFCC, which is a data format that has been widely applied in automatic speech recognition (ASR). For each audio frame, the first and the second derivative components of the MFCCs are aggregated to a vector that represents the single audio frame, f. The input feature x_t at time t to the network feature f_t at time t surrounded by its forward and backward contextual vectors, denoted as $[f_{t-n}, \ldots, f_t, \ldots, f_{t+m}]$. The value of n represents the number of audio frames before time t, and the value of m represents the number of future audio frames. The integrated x_t is used to predict the phoneme label at time t.

B. Phoneme Recognition System

The phoneme recognition is conducted by RealPRNet and HMM tri-phone decoder. The RealPRNet leverages different neural networks such as CNN, LSTM and so on. The details of the network architecture of RealPRNet is described in Section III. The output of the RealPRNet is an HMM tied-state H, which is used as input of the decoder to calculate the output phoneme P. For the system to produce a smooth animation, the phoneme recognition system needs to be able to output the predicted phoneme with a constant time interval of A ms in reality.

In the first step, input raw audio signal is transformed into the f_t every A ms time interval (equal to the sampling interval) and the f_t is stored in the input feature buffer B_1. The calculation of raw audio signal to audio features transformation process causes the first delay $d_1 + e_1$ where d_1 is median calculation time and e_1 is the corresponding fluctuation time. In our experiment, time consumption by this process is very stable and 100% less than A ms.

All the audio frame features in B_1 is then used to construct x_t. The size of B_1 depends on two things: (1) the selected values m and n. (2) RealPRNet time consumption $d_2 + e_2$ for each prediction. To guarantee a smooth output, the following inequality needs to be satisfied:

$$d_2 + e_2 \leq br \cdot A, br = 1, 2, 3, \cdots \tag{1}$$

where br is the batch size used in prediction progress. The neural network can parallelly predict br outputs in each run by a minimal increase in computational overhead if br is a small value (i.e. $br = 10$). The major time consumption is in data parsing and transmission in this scenario. The RealPRNet takes the input features from B_1 every $br \cdot A$ ms and predicts br outputs $[h_t, h_{t-1}, \cdots]$. These predicted outputs are stored in HMM tied-state buffer B_2. In our experiment, the value of br is 4 which can ensure that equation (1) is satisfied in 99% of the cases. There are br sub-buffers in B_1 with size $m + n + 1$ each and contents $f_{t-n-i}, \cdots, f_{t-i}, \cdots, f_{t+m-i}$. Because m forward audio frames are used to construct the x, the system latency is increased to $m \cdot A + br \cdot A$ ms.

The predicted HMM tied-states in B_2 are used as input of the HMM tri-phone decoder. For each time interval $br \cdot A$, the decoder takes all the values in B_2, calculates the corresponding phonemes $[P_t, \cdots, P_{t-br+1}]$ and stores it in B_3. The calculation time is $d_3 + e_3$ and the size of B_2 depends on the number of previous states used to calculate the $[P_t, \cdots, P_{t-br+1}]$. In our experiment, the calculation time consumed in decoding is stable and 100% less than $br \cdot A$.

Thus, the overall latency from raw input audio signal to the corresponding output phoneme is $(m+br) \cdot A + D + e_t$, where $D = d_1 + d_2 + d_3$ and $e_t = e_1 + e_2 + e_3$.

B_3 is the buffer that is used to control the final output of P of the phoneme recognition system. The system continuously

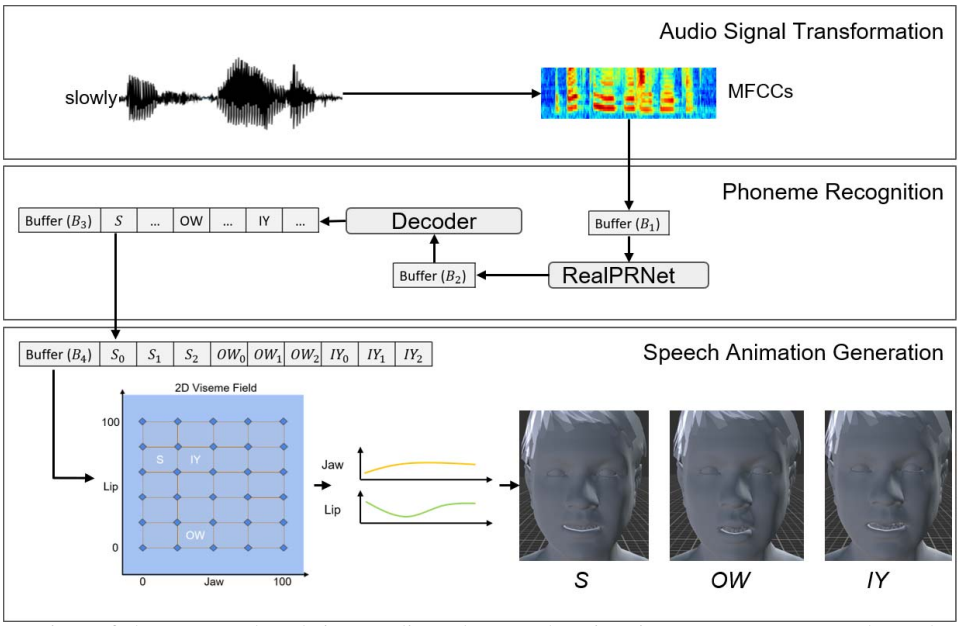

Fig. 1: System overview of the proposed real-time audio only speech animation system corresponds to the phoneme stream {S, OW, IY} of vocabulary /slowly/.

takes the first phoneme from B_3 or the latest phoneme (if B_3 is empty) and uses it as the input to the animation system every A ms time interval. With this mechanism, the phoneme recognition system can have a stable output stream with time interval A ms and the overall latency from raw input audio signal to the corresponding output phoneme stream is $(m+br)\cdot A + D$, where $D = d_1 + d_2 + d_3$. This stable output phoneme stream is stored in B_4 and uses to produce the corresponding speech animation.

C. Speech Animation

For real-time speech animation, the system cannot directly use the output phoneme sequence from the phoneme recognition system because it predicts the output phoneme for every audio frame with time interval A. Thus B_4 is used to select the appropriate next phoneme frame for the animation curve generation. The size of B_4 is the average single phoneme pronunciation time in the audio frame. The output phoneme of the recognition system is stored in this buffer first. The phoneme pronunciation is dynamic progress which means that the viseme phoneme at the beginning of the pronunciation is not exactly the same as the corresponding phoneme viseme as shown in Fig. 2. Thus, an appropriate phoneme frame (for example, the phoneme frame can represent the rightmost one in Fig. 2 in a sequence of phoneme /o/ frames) should be selected from certain phoneme pronunciation frames to calculate the complete viseme's transformation time which uses for animation curve generation. Thus, the upcoming phoneme is selected based on the following rules:

(1) The same phonemes are continuously appended to the buffer for at least pr_{min} units.

(2) If the number of continuous phonemes is more than pr_{min} units and less than the size of B_4. The appropriate frame that represents the phoneme will be selected from that part of the buffer.

(3) If the number of the continuous phonemes is more than the size of B_4, all phonemes in the buffer will be used to select the appropriate frame and no new frame will be selected until the next different phoneme is appended to the buffer.

The speech animation system has been developed following [5] with slight difference. Most of the procedural systems, such as [5], [9], that use the key frame viseme to produce the final animation have a similar problem, that is the phoneme is mapped to one fixed static viseme without any variation. Through our observation of human speech behaviors, visemes corresponding to a phoneme may be slightly different in different situations. Thus, the 2D viseme field has been divided into different blocks (20 blocks in our implementation) as shown in Fig. 3. Each phoneme corresponds to a block region rather than a static point in the 2D field.

III. Network Architecture

The overall network architecture is shown in Fig. 4. The CNN layer takes x_t as input and applies frequency modeling on the input features. The n-dimensional vector output of CNN is passed into an n-layer stacks of LSTMs for parallel processing and temporal modeling. The output is combined with the f_t to form an input to additional LSTM layers for temporal modeling, and then goes through a fully connected layer. In the end, the HMM tri-phone decoder is adopted to predict the phoneme label.

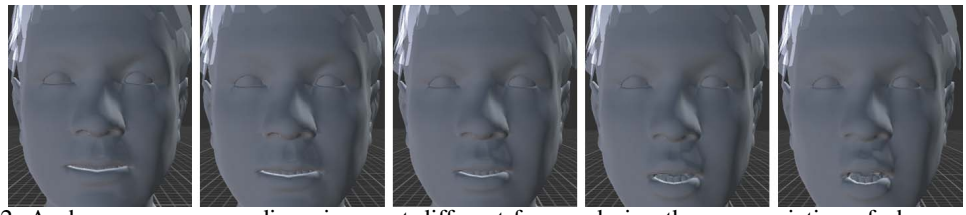

Fig. 2: A phoneme corresponding viseme at different frames during the pronunciation of phoneme /o/.

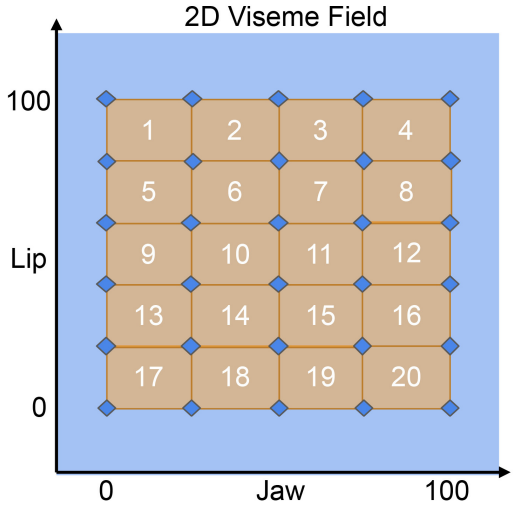

Fig. 3: 2D viseme field with block index.

A. CNN Layer

The CNN layer is a frequency modeling layer. Based on the observation that the voice of different people contains different ranges of frequencies even when they are speaking the same utterance. An important contribution of the CNN layers is to reduce frequency variation [6].

CNN layers also play an important role in the temporal-spatial domain, here spatial refers to frequency-domain audio feature pattern detection and learning. The input feature contains frequency information since each coefficient in Mel-frequency cepstral is generated by passing-through different frequency filter banks. Each CNN filter can learn the different frequency patterns from the input features during the training. Our network architecture emphasizes these frequency patterns in the input features by separately connecting the CNN output features (CNN_{fout}) of different CNN filters to different LSTM in the LSTM stack Block module.

B. LSTM Stack Block

After frequency modeling is applied to CNN layers and the CNN filters have learned the acoustic features from the input feature, each CNN output channel produces the intermediate features and these features are applied to a parallel process of the temporal modeling in LSTM Stack Block (LSB).

The CNN_{fout}'s from different CNN channels pass to different LSTM modules, called LSTM Tube (LT) which is inside the LSB as shown in Fig. 4. The number of LT inside the LSB depends on the number of the last CNN layer output channel, one LT for each CNN output channel. An LT is a relatively small and independent LSTM network. The LSTM has been proved to be advantageous on temporal modeling because it has memory cells to remember the information on previous features [10], [11]. Each output from different CNN channels can be viewed as a derivative feature of the original audio feature with the context within a sliding window. Thus, separate temporal modeling is applied to each different CNN_{fout} and the corresponding outputs features are aggregated together as the input feature for the next LSTM layer.

C. LSTM Layer

In [7], it is demonstrated that the LSTM layer has its advantage in extracting temporal patterns in input feature space as we mentioned in the LSB. The gate units in LSTM are used to control the information flows inside the LSTM. The input gate decides what information can be added to the cell state. The forget gate decides what information needs to be removed from the cell state and the output gate decides what information can be used in the output. Thus the LSTM can selectively "remember" the temporal features. After the LSB performs the separate temporal modeling, we pass its output to the LSTM layers for the unified temporal modeling with the same logic.

D. Fully Connected Layer

Following the state-of-the-art design of typical deep learning network, we use the fully connected layer with softmax activation as the last layer so that a fully connected layer can provide the model with the ability to mix output signals from all neurons in the previous layer, and the softmax can reshape the output probabilities of each class so that the target class has a higher probability.

E. Multi-Scale Features Addition

The idea of a multi-scale feature addition was originally explored in computer vision and also used in ASR. The input features of the different layers are complementarity and the network's performance improvement has been observed by using these techniques [6]. In our implementation, we have explored two feature addition strategies, which are illustrated in Fig. 4 through line (1) and line (2), where (1) represents the original frame feature, f_t, in x_t combines with the LSB output and (2) represents the LSB output combines with the LSTM output.

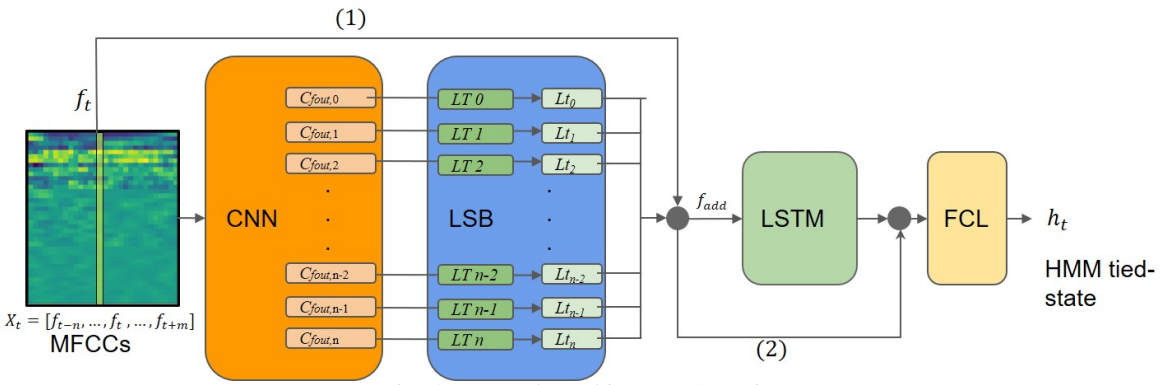

Fig. 4: Network Architecture Overview

The first features addition explores the complementary information in short term feature f_t and long term feature output feature from LSB (high order representation of x_t). x_t is aggregated by using f_t and it's context features $[f_{t-n}, \cdots, f_{t-1}]$ and $[f_{t+1}, \cdots, f_m]$. However the original LSTM does not consider their different values in prediction but takes all f's in x_t as consecutive features [12] and equally considers all the f's in x_t. This features addition emphasizes the importance of f_t in x_t when predicts p_t.

The second feature addition checks the LSB and the LSTM outputs, the separated and unified complementary temporal feature information. These features are the high order feature representations of x_t with different pattern complementarity since network layers focus on different information in x_t with these patterns.

In the evaluation part, the performance improvement by multi-scale feature addition has been explored and the results show a positive effect on the network performance.

IV. Experimental Results

In this section, we evaluate the performance of the proposed RealPRNet in two areas: (1) the phoneme recognition accuracy, (2) the subjective and objective speech animation quality, and

A. Experiment Setup

Our experimental results are conducted using the TIMIT data set which is widely used for phoneme recognition evaluation.

We use Kaldi's TIMIT s5 recipe [13] to calculate phoneme duration in each utterance through force alignment technique and generate the corresponding HMM tied-state tri-phone model. We also use a tri-phone decoder with a bigram phone model in our phoneme recognition system to improve the overall performance. For the network training, 10 epochs have been set as the minimum training epoch and enabled early stop (if the validate loss change in epochs is less than 0.001) during the training.

The performance of the network under different latency scenarios (different value of m) in the input feature was evaluated in this section. In our experiment, value n in x_t has been set to $m + 1$.

B. Error Metrics

1) Phoneme error rate: The standard TIMIT phoneme recognition evaluation error metric was used to evaluate the phoneme error rate (PER) of our proposed RealPRNet. The original TIMIT data set uses 60 phonemes to create the dictionary. During the evaluation, we first map the 60 phonemes to 39 phonemes, and then calculate the "Levenshtein distance" between the recognized phoneme sequence and the ground truth phoneme sequence. Levenshtein distance is a method to measure the difference between the two sequences. It calculates the minimum number of single-character edits (including insertions, deletions and substitutions) required to change one sequence into another. The number of the edits required to change the recognized phoneme sequence into the ground truth phoneme sequence is first calculated followed by the ratio between this minimum number of edits and the whole phoneme sequence length. This ratio is the PER. Under the real-time situation, the phoneme is predicted for each audio frame. Thus, the PER was calculated based on the audio frame level phoneme sequence in our evaluation.

2) Block distance error: The block distance error measures the average distance between the trajectory curve in the viseme field produced by the recognized phoneme sequence and the curve produced by the ground truth phoneme sequence. The basic unit used here is the short edge of the 2D viseme field block. For each audio frame, the corresponding points on the two curves and the absolute distance between these two points are calculated, also known as Euclidean distance. Then the average distance between these two curves on each time t was calculated.

C. Phoneme Recognition Performance

In the experiment, RealPRNet was compared with two other phoneme recognition systems: (1) the 4 layers LSTM architecture presented in [7]. (2) the CLDNN discussed in [6].

In Fig. 5, the real-time performance of these networks were compared. The x-axis in the figure represents the number of frames used in a single x_t and the y-axis represents the

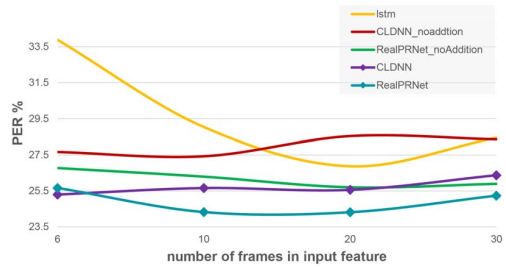

Fig. 5: Various Networks Performance in Phoneme Error Rate

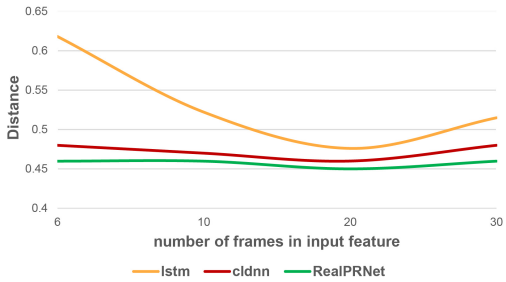

Fig. 6: Various Networks Performance in Block Distance Error

phoneme error rate. It is interesting to observe that LSTM has low performance when the temporal context information is insufficient (i.e., when x_t is aggregated by using less than 10 fs), but its performance improvement continuously with the increasing of the x_t value until a sweat spot (20 audio frame features in x_t) is achieved. In the figure, RealPRNet outperformed LSTM by up to 20% and CLDNN by up to 10%. The combination of frequency and temporal modeling enabled the RealPRNet a smooth performance across a various selection of the number of frames for x_t.

As illustrated in the previous section, the performance of the network can be further improved by multi-scale feature addition techniques. We explore this technique with our RealPRNet and compare its performance with the strongest baseline model CLDNN with feature addition structure and also the previous networks without feature addition. As shown in Fig 5, the performance of both RealPRNet and CLDNN with multi-scale feature addition has been improved. The additional short term feature gave complementary information to the intermediate input features which forced the networks to focus on the current frame f_t. Thus, RealPRNet outperforms other networks in most of the latency scenarios. In particular, it can achieve an additional 4% relative improvement in real-time PER evaluation.

D. Animation Quality Assessment

In this section, we evaluate the quality of the output speech animation of our system. We compare the trajectory curve in the viseme field produced by the recognized phoneme streams predicted by various reference networks and RealPRNet with the trajectory curve produced by the ground truth model. The block short edge length is used as a distance unit to calculate the difference between the recognized phoneme produced curve and the ground truth curve at each time instance.

The result is shown in Fig. 6, the x-axis in the figure represents the number of frames used in input features and the y-axis represents the block distance error in the basic unit. The RealPRNet also achieves the minimum error under different latency scenarios, which is up to 27% less than other networks.

V. Conclusions

In this paper, we proposed an audio-driven believable speech animation production framework with a new phoneme recognition neural network called RealPRNet. The RealPRNet has performed much better than the strong baseline model in phoneme recognition. Our framework can be easily implemented in most of the existing virtual avatars without creating a new avatar.

References

[1] Y. Zhou, Z. Xu, C. Landreth, E. Kalogerakis, S. Maji, and K. Singh, "Visemenet: Audio-driven animator-centric speech animation," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 161, 2018.

[2] S. Taylor, T. Kim, Y. Yue, M. Mahler, J. Krahe, A. G. Rodriguez, J. Hodgins, and I. Matthews, "A deep learning approach for generalized speech animation," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 93, 2017.

[3] G. Llorach, A. Evans, J. Blat, G. Grimm, and V. Hohmann, "Web-based live speech-driven lip-sync," in *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pp. 1–4, IEEE, 2016.

[4] C. G. Fisher, "Confusions among visually perceived consonants," *Journal of speech and hearing research*, vol. 11, no. 4, pp. 796–804, 1968.

[5] P. Edwards, C. Landreth, E. Fiume, and K. Singh, "Jali: an animator-centric viseme model for expressive lip synchronization," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 127, 2016.

[6] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584, IEEE, 2015.

[7] J. Michalek and J. Vaněk, "A survey of recent dnn architectures on the timit phone recognition task," in *International Conference on Text, Speech, and Dialogue*, pp. 436–444, Springer, 2018.

[8] I. Bromberg, Q. Qian, J. Hou, J. Li, C. Ma, B. Matthews, A. Moreno-Daniel, J. Morris, S. M. Siniscalchi, Y. Tsao, *et al.*, "Detection-based asr in the automatic speech attribute transcription project," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[9] A. Pearce, B. Wyvill, G. Wyvill, and D. Hill, "Speech and expression: A computer solution to face animation," in *Graphics Interface*, vol. 86, pp. 136–140, 1986.

[10] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, IEEE, 2013.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth annual conference of the international speech communication association*, 2014.

[13] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF, IEEE Signal Processing Society, 2011.