Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 46 (2015) 60 – 67

Procedia Computer Science

International Conference on Information and Communication Technologies (ICICT 2014)

Duration Modelling Using Neural Networks for Hindi TTS System Considering Position of Syllable in a Word

Shreekanth.T[a,*], Udayashankara.V[b], Chandrika.M[c]

[a] *JSSRF, SJCE Campus, Department of ECE, SJCE, Mysore,570006, India*
[b] *Department of IT, SJCE, Mysore,570006, India*
[c] *Department of ECE, SJCE, Mysore,570006, India*

Abstract

The main criterion in duration modeling is to model the duration pattern of the natural speech, considering various features that affect the pattern. Proper estimation of segmental durations plays a vital role in natural sounding text-to-speech (TTS) synthesis. The primary reason for choosing the syllable as a basic unit is that the Indian languages are syllable centered. This paper presents a novel text processing and a syllable based data driven modelling of segmental duration for Hindi, using feed forward neural networks. The effectiveness of the system is demonstrated by synthesizing natural sounding speech for Hindi, national language of India.

Peer-review under responsibility of organizing committee of the International Conference on Information and Communication Technologies (ICICT 2014)

Keywords: Duration; Neural Networks; Hindi; Mean Opinion Score (MOS); PRAAT; TTS; UNICODE.

1. Introduction

The TTS system is intended to convert an arbitrary input text to corresponding natural sounding speech in a more intelligible way. The two main components of a TTS system are text processing and speech generation. The function of the text processing component is to generate appropriate sequence of phonemic units.

* Corresponding author. Tel.: +91-998-648-3968.
E-mail address: speak2shree@gmail.com

Peer-review under responsibility of organizing committee of the International Conference on Information and Communication Technologies (ICICT 2014)
doi:10.1016/j.procs.2015.01.056

These phonemic units are realized by the speech generation component either by synthesis from parameters or by selection of units from a large speech corpus. For natural sounding speech synthesis, it is essential that the text processing component produces an appropriate sequence of phonemic units corresponding to an arbitrary input text[1].

Computation of correct segmental durations is vital for natural sounding text-to-speech (TTS) synthesis. Variation in segmental duration is a key to the identity of a speech sound and helps to segment a continuous flow of sounds into words and phrases thereby increasing the naturalness and intelligibility[2].

The duration models are generally grouped into rule-based models and corpus based. The main difference between rule-based and statistical models is that a rule-based model can be built on relatively less speech data. Rule based methods involve manual analysis of segment durations. The derived rules get better in terms of accuracy, as the amount of speech data for analysis is increased. But with large amount of data the process of manually deriving the rules becomes tedious and time consuming. Hence, rule-based methods are limited to small amount of data. Also, this method depends on linguistic and phonetic literature about the factors that affect duration of the units (segments, syllables or phones). The complex interaction among the linguistic features at various levels makes the rule based methods more difficult to analyse and implement. Statistical data-driven methods are attractive when compared to rule based methods. This method works when large phonetically rich sentences are present in the corpora and is based on either parametric or non-parametric model that uses probability or likelihood functions[2].

One of the early attempts for developing rule-based duration models was in 1970s[4]. The model was based on information present in linguistic and phonetic literature about different factors affecting segmental durations. The rules were derived by analysing a set of phonetically balanced sentences. Following this model, similar models were developed for other languages like French[5] and Mandarin[6]. Of late, a corpus based duration model has been developed for Hindi (the Indian national language) TTS system[2, 3].

This paper intends to design a duration modelled Hindi TTS system considering position of syllable in a word using data-driven method. The syllable is used as a unit in this work as it captures the co-articulation effects and it is also a convenient unit for speech in Indian languages. The syllable used is of the form V and CV. This paper is organized as follows: The section 2 provides the information about the Hindi language. Section 3 describes the developed duration and speech database. Section 4 presents the Methodology. Section 5 depicts the results and discussion, and as a final point section 6 concludes and remarks about some of the future aspects.

2. Overview of Hindi Language

Hindi, the national language of India, spoken by 33 percent of the population has 33 consonants and 13 vowels. Hindi language is having one to one correspondence with the spoken language and the written form. The phonemes are divided into two types: vowels (swaras) and consonants (vyanjanas). Vowels (Swaras): Vowels are the independently existing letters which are called swaras and the sound of Vowels cannot be modified. Consonants (Vyanjanas): Consonants are those which depend on vowels to form their independent letter. Consonants sound can be modified by combining vowels with consonants[7]. For this reason Hindi language is phonemic in nature. Amalgamation of vowels with consonants will form a syllable and it is also called as "Baraha Khadi".

3. Duration Database and Speech Database Building

During the process of speech synthesis the required speech units are fetched from the database, concatenated and further processed using a suitable algorithm. Hence creating an error free database considering syllable as a basic unit is of greatest importance.

In order to carry out this task, a set of about 1540 words were collected from standard Hindi to English dictionary[11]. Later speech recording was done using the utility software for windows operating system called Praat[10]. The syllables were recorded with a sampling frequency of 16 kHz and represented using 16-bits. The pitch and formant frequency of the syllable fluctuate with position of the syllable in uttered speech. Consequently the syllable level speech database is generated for all the possible position of occurrences. The syllable can befall at three possible positions.

- Beginning of the word (Start)

- Between the two syllables (Middle)
- At the end of the word (End)

To get natural sounding synthesized speech, each syllable was extracted from a word containing syllables. Three words were selected such that, they contained the required syllable in the said three positions. Then the necessary syllables were extracted from the particular positions using Praat tool and stored in directories named start, middle and end with a unique numerical value for each based on the corresponding Unicode[7]. For the development of duration database based on the position of the word, seven samples of speech utterances of a particular word were considered. This provides the information of duration of each syllable in seconds. The seven duration samples of each syllable with respect to its position in a word is shown in Fig. 1. (a) to Fig. 1. (c). A similar procedure was adopted for the remaining syllables.

a

Syllable	Duration samples for each syllable						
क	0.0925	0.0899	0.0885	0.0901	0.0896	0.0910	0.0892
का	0.3004	0.3006	0.275	0.2963	0.3018	0.2952	0.3050
कि	0.1045	0.0996	0.1099	0.1012	0.1034	0.1100	0.0985
की	0.2845	0.2795	0.2789	0.283	0.281	0.2820	0.2799
कु	0.0979	0.0912	0.0956	0.096	0.0932	0.0942	0.0923
कू	0.2214	0.2196	0.220	0.2216	0.218	0.2156	0.2225
कृ	0.1281	0.127	0.1269	0.1258	0.127	0.1264	0.1235
के	0.3093	0.3064	0.3101	0.3054	0.307	0.3083	0.3110
कै	0.354	0.3516	0.3532	0.3512	0.3537	0.3522	0.3543
को	0.3116	0.3102	0.3096	0.308	0.310	0.3122	0.3075
कौ	0.2683	0.2674	0.2662	0.2651	0.264	0.2645	0.2690
कं	0.120	0.121	0.1196	0.119	0.118	0.122	0.126
कः	0.443	0.440	0.399	0.444	0.398	0.422	0.410

b

Syllable	Duration samples for each syllable						
क	0.1071	0.1052	0.1019	0.0996	0.1034	0.1069	0.1082
का	0.3084	0.3052	0.3031	0.3004	0.2945	0.3022	0.3010
कि	0.098	0.095	0.092	0.089	0.086	0.096	0.093
की	0.2642	0.2616	0.2594	0.2561	0.2554	0.2525	0.2625
कु	0.2158	0.2134	0.2108	0.2072	0.2051	0.2120	0.2041
कू	0.2703	0.2674	0.2632	0.2612	0.2654	0.2710	0.2722
कृ	0.1431	0.1409	0.1368	0.1332	0.1304	0.1421	0.1399
के	0.3196	0.3172	0.3153	0.3126	0.3109	0.3181	0.3155
कै	0.2951	0.2932	0.2904	0.2864	0.2832	0.2946	0.2947
को	0.3003	0.2964	0.2932	0.2901	0.2952	0.2954	0.3100
कौ	0.2378	0.2354	0.2321	0.2304	0.2274	0.2359	0.310
कं	01261	0.1232	0.1201	0.1196	0.1172	0.1254	0.1297
कः	0.443	0.440	0.399	0.444	0.398	0.460	0.380

c

Syllable	Duration samples for each syllable						
क	0.0297	0.0265	0.0243	0.0221	0.0214	0.0278	0.0301
का	0.345	0.342	0.339	0.336	0.334	0.355	0.369
कि	0.1751	0.1734	0.1701	0.1696	0.1654	0.1745	0.1711
की	0.4342	0.4326	0.4304	0.4289	0.4264	0.4356	0.4296
कु	0.3426	0.3417	0.3396	0.3354	0.3371	0.344	0.3375
कू	0.3455	0.3432	0.3421	0.3408	0.3386	0.3399	0.3446
कृ	0.1842	0.1824	0.1796	0.1776	0.1743	0.1855	0.1726
के	0.2987	0.2964	0.2926	0.2891	0.2864	0.288	0.2975
कै	0.2941	0.2922	0.2901	0.2867	0.2432	0.2566	0.2788
को	0.3441	0.3421	0.3401	0.3386	0.3354	0.3399	0.343
कौ	0.2811	0.2802	0.2786	0.2765	0.2745	0.2798	0.282
कं	0.120	0.121	0.1196	0.119	0.118	0.129	0.113
कः	0.443	0.440	0.399	0.444	0.398	0.449	0.395

Fig. 1. (a) Start duration database; (b) Middle duration database; (c) End duration database.

4. Methodology

This section covers in brief the steps involved in building the proposed TTS system and rules applied during pre-processing of the input text. The functional flow of duration modelling in TTS system using neural networks is depicted in Fig. 2.

Text processing unit acts as the front end of the TTS system. Orthographic text provided as input, is converted to the respective Unicode and then to its equivalent decimal value and is stored in a text file. This decimal output is used as input to neural network. Speech corpus consists of syllable level speech database that is segmented and labelled based on text processing output. Neural network block consists of set of duration data and the target data that are trained for accurate duration output. The duration database is prepared based on the position of the syllable in a word i.e., starting syllable, middle syllable and end syllable.

The algorithm fetches the decimal output one after the other and matches each time with the duration database. Then the data processing takes place in the neural network block and the corresponding speech files fetched from speech corpus are concatenated.

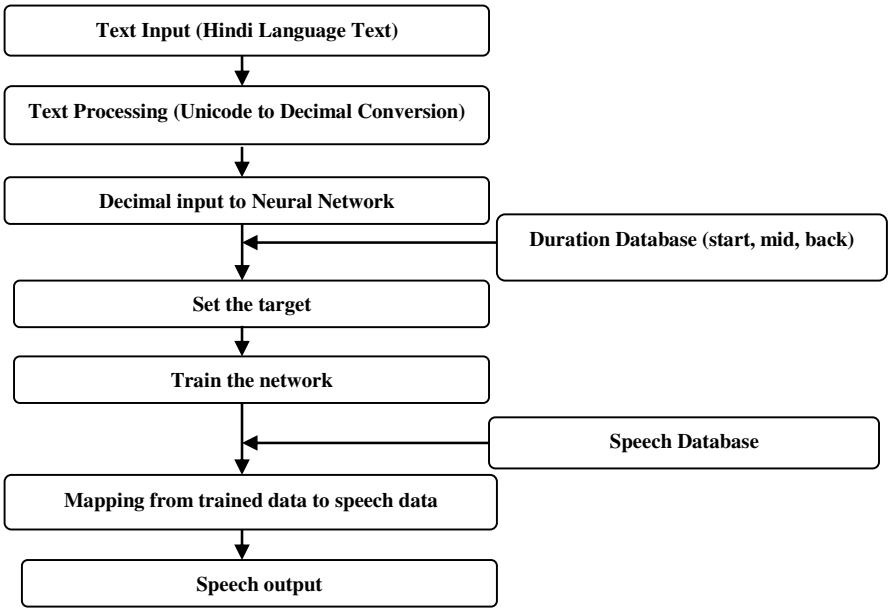

Fig. 2. Overview of Proposed TTS System

4.1 Text Processing

The steps involved in pre-processing the orthographic input text is shown in Fig. 3. Text processing is the first and foremost step involved in developing TTS system. The main intention of this stage is to convert the input text into its equivalent Unicode and later it is mapped to a unique decimal code. The Hindi symbols and alphabets are congregated into diverse classes as revealed in Fig. 4. (a) to Fig. 4. (f). Based on the above assumptions an algorithm is developed using .NET programming language.

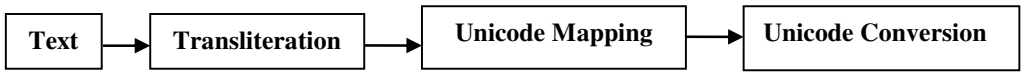

Fig. 3. Text pre-processing steps

With the entry of Hindi text as input, pre-processor maps the Hindi text to its corresponding English transliteration code, character by character and stores the transliteration output as a string in specified memory

location. In the next block, transliteration output is mapped with already stored Unicode, character by character and the corresponding hexadecimal values of the characters are obtained. The mapped Unicode is then converted to its corresponding decimal equivalent. Far ahead decimal values are padded with digits to differentiate amid dependent and independent consonants. Finally, the padded outputs are stored in a text file.

To resolve the ambiguities present in understanding Hindi alphabets, consonants and vowels are grouped into different classes and programmed as illustrated in Fig. 4. (a) and Fig. 4. (b).

a

Alphabets	Unicode	Decimal Equivalent
अ	0905	2309
आ	0906	2310
इ	0907	2311
ई	0908	2312
उ	0909	2313
ऊ	090A	2314
ऋ	090B	2315
ए	090F	2319
ऐ	0910	2320
ओ	0913	2323
औ	0914	2324

b

Alphabets	Unicode	Decimal Equivalent
क	0915	2325
ख	0916	2326
ग	0917	2327
घ	0918	2328
ङ	0919	2329

c

Alphabets	Unicode	Decimal Equivalent
◌	093E	2366
◌ा	093F	2367
◌ि	0940	2368
◌ी	0941	2369
◌ु	0942	2370
◌ू	0943	2371
◌े	0947	2375
◌ै	0948	2376
◌ो	094B	2379
◌ौ	094C	2380
◌्	094D	2381

d

Alphabets	Unicode	Decimal Equivalent
◌	093E	---
◌ा	093F	01
◌ि	0940	02
◌ी	0941	03
◌ु	0942	04
◌ू	0943	05
◌े	0947	06
◌ै	0948	07
◌ो	094B	08
◌ौ	094C	09
◌्	094D	10

e

Numbers	Unicode	Decimal Equivalent
0	0966	2406
1	0967	2407
2	0968	2408
3	0969	2409
4	0970	2410
5	0971	2411
6	0972	2412
7	0973	2413
8	0974	2414
9	0975	2415

f

Symbols	Unicode	Decimal Equivalent
◌ँ	0901	2305
◌ं	0902	2306
◌ः	0903	2307
ॐ	0950	2384
॰	0950	2416
।	0964	2404
॥	0965	2405

Fig. 4. Developed Unicode's for different symbols (a) Independent Vowel; (b) Consonants; (c) Dependent Vowel Sign; (d) Padding; (e) Numbers; (f) Special Symbols.

The program reads the entered text, character by character and generates the modified Unicode as the output. The algorithm for this as follows.

4.1.1 Algorithm for text processing

Step1. Check whether the user has entered the valid input text or not, by computing the length of the entered text.
Step2. If the entered text length is zero, it means no text has been entered. Otherwise calculate the total length of the entered text.
Step3. Split the entered text into sentences and sentences into words by spotting spaces present between every word in a sentence.

Step4. Store segmented words in memory for further analysis. Map each stored word with an array element which encompasses "English transliteration codes" of Hindi language.
Step5. Store the transliterated output in a separate file.
Step6. Read the warehoused result character by character, and check whether the character belongs to the vowel (V) group or the consonant (C) group.
Step7. If the read character is a vowel then its Unicode is directly mapped from the vowel group depicted in Fig. 4. (a) e.g., if the read character is अ, its decimal code is 2309 and this is padded with two zeroes directly. The modified decimal code will be 230900.
Step8. If the read character is a consonant then check the subsequent character and if that also is a consonant then map the previous consonant directly as in Fig. 4. (b) e.g., if the character read is ण, its decimal code is 2339, and if the next character read is also a consonant then the decimal code remains as 2339 itself.
Step9. If the read character is a consonant then check the subsequent character and if that is a dependent vowel sign as in Fig. 4. (c) then the decimal code of the consonant is padded with corresponding two digit values as in Fig. 4. (d) e.g., if the read character is रु, then it is divided into र whose decimal code is 2352 and ु whose code is 2370. The padding value for this combination as obtained from Fig. 4. (d) is 04. Consequently the Unicode becomes 235204.
Step10. If the read character is a special character or a number then the decimal code is directly mapped as in Fig. 4. (e) and Fig. 4. (f) respectively.
Step11. Repeat the above steps until the end of the text is reached. The resulting modified decimal codes are used for further processing.

The output of the proposed algorithm for the entered text अरुण and अरुणकुमार are as follows: If the entered word is अरुण then its modified decimal code output will be 230900 235204 2339, the presence of spaces between each Unicode helps us to differentiate the individual characters in the entered word. If the entered sentence is अरुणकुमार then its modified decimal code output will be 230900 235204 2339 101010 232501 2350 2352. Decimal code 101010 acts as a space between two words and is used to differentiate two words during sentence formation.

4.2 Neural Network Architecture

In order to capture the relationship between input and output vectors Feed forward neural networks (FFNN) are used[8, 9]. For modelling syllable durations, a four layer feed forward neural network is employed.

The first layer is the input layer consisting of linear elements, followed by, second and third layers which are hidden layers. The second layer (first hidden layer) of the network has more units than the input layer, and is meant to capture some local features in the input space. The third layer (second hidden layer) has fewer units than the first layer and is meant to capture some global features[8, 9]. The fourth layer is the output layer having one unit representing the syllable duration. The learning algorithm used for adjusting the weights of the network is standard back propagation algorithm to minimize the mean squared error for each syllable duration[9].

4.3 Neural Networks Algorithm:

The algorithm developed for duration modelling using neural networks is the furtherance of text processing algorithm. It consists of an algorithm which accepts text file generated by text processing algorithm as input and gives concatenated duration modelled speech as output. Based on text processing result algorithm detects the speech file present in speech database and concatenates the individual speech files and as a result outputs the concatenated speech. Fig. 5. gives the step by step flow of the duration modelling using neural networks.

4.3.1 Algorithm for duration modelled speech synthesis

The text file generated during text processing process is directly imported to MATLAB path. Once the text file is available on MATLAB path the code reads the text file and computes the length of that text file. After calculating

the length of the text file the algorithm first checks whether user entered a word or sentence by detecting number of spaces present in entered text. The space is identified by a decimal value 101010, and if a space is present in the entered text, then it will be considered as a sentence. The positions of 101010 are stored in a text file. Here each syllable is trained for the accurate duration by setting the target and speech data is fetched according to the trained output. At the output end, each syllable is concatenated according to the text entered by checking the space and results with concatenated speech as output. Later the text file is read to detect the length of each word present in a sentence.

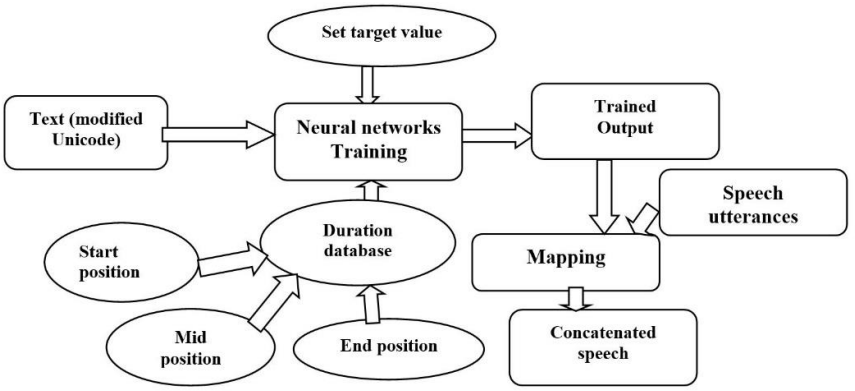

Fig. 5. Flow chart of neural network design

Step1. If the identified word length is 1, then the algorithm compares the text file output with front duration database and trains with these target data and maps with the corresponding speech file with respect to the trained output.

Step2. If word length is 2, then the algorithm compares the text file output with front duration database and back duration database respectively and trains with the set target data, in turn mapping with the corresponding speech file with respect to the trained output.

Step3. If word length is 3, then the algorithm compares the text file output with front, mid and back duration database and trains with the set target data and maps with the corresponding speech file with respect to the trained output.

Step4. If word length is greater than 3 then algorithm displays a message 'Limit exceeded'.

The above mentioned steps are repeated for the whole text. Finally the concatenated speech signal is passed through a silence removal algorithm to eliminate the delay present between each concatenation point. Here 6630 samples of data are selected for training and remaining 2652 samples are selected for testing purpose.

5. Results and Discussion

The results of this work is evaluated to check the naturalness of the synthesized speech on the two sets of developed databases, one considering syllable position (DB1) with duration variation and the other without considering the duration variation of the syllable (DB2). To verify the naturalness of the synthesized speech obtained from these two databases Mean Opinion Score (MOS) is taken from various listeners.

Mean opinion score is the arithmetic mean of all the individual scores and it gives the numerical indication of the perceived audio quality. To check the naturalness of the speech a listeners test have been conducted to evaluate the results of the proposed technique. The algorithm is tested on both the databases DB1 and DB2 in order to compare the results. Ten listeners were asked to rate the quality of the speech synthesized using the proposed technique for both the databases. Each listener were asked to rate on a scale from 1 to 5, where 1 represents the lowest perceived audio quality, while 5 represents the highest perceived audio quality. Table. 1. depicts the comparison of the proposed technique with the already existing ones. It is apparent from the Table. 1. that the performance of the

proposed algorithm is better for DB1 as compared to DB2 and also the proposed method is more efficient than the existing ones. This indicates that the speech database developed with duration modelling using neural networks accounts to the natural speech.

Table. 1. Comparison of Mean Opinion Score obtained from listeners.

Algorithm	MOS of DB1 Database	MOS of DB2 Database
Klatt [4]	2.4	1.5
Proposed	4.5	3.5

6. Conclusion

The data driven syllable based duration modelling for Hindi language is presented. A novel text processing algorithm has been implemented using .NET tool and the duration modelling using neural networks is developed using MATLAB programming. A four layered feed forward neural networks trained with a back propagation algorithm is followed. The model is subjectively evaluated by computing the Mean Opinion score. The naturalness of speech output between DB1 built considering position of syllable with duration variation and DB2 built without considering syllable duration variation is validated. Mean opinion score (MOS) is taken from 10 listeners for each word synthesized. The average MOS obtained for DB1 is 4.5 and for DB2 it is 3.5. Hence from the average MOS we can conclude that, TTS system developed using DB1 with duration modeling will give better quality speech output without any spectral discontinuity compared to TTS system developed using database DB2 output. The performance of the proposed system can be improved through increasing the size of the database.

Acknowledgement

We would like to thank Council of Scientific and Industrial Research (CSIR), New Delhi, India for providing the financial backing for this work under the research scheme No. **22(0613)/12/EMR-II** and also the work is supported by JSS Research Foundation, JSS Mahavidyapeetha, Mysore, Karnataka, India.

References

1. Ravi DJ, Sudarshan PK. A Novel Approach to Develop Speech Database for Kannada Text-to Speech System. *International Journal on Recent Trends in Engineering & Technology* 2011. p. 119-122.
2. Sridhar Krishna N, Hema AM. Duration modeling of Indian Languages Hindi and Telugu. *Indian Institute of Technology Madras Chennai* 2009.
3. Sreenivasa Rao K, Yegnanarayana B. Modelling Syllable Duration in Indian Languages Using Neural networks. *Indian Institute of Technology Madras Chennai* 2009.
4. Klatt DH. Linguistic uses of segmental duration in English: Acoustic and perceptual evidence, *Journal of Acoustic Society of America* 1976. p. 1209-1221.
5. Bartkova K, Sorin C. A model of segmental duration for speech synthesis in French. *Speech Communication* (6) 1987. p. 245–260.
6. Chen SH, Lai WH, Wang YR. A new duration modeling approach for Mandarin speech. *IEEE Transactions on Speech and Audio Processing* 2003. p. 308–320.
7. Arun Kumar C, Shreekanth T, Udayashankara V. Development of Speech Database for Hindi Text to Speech system Considering Syllable as a Basic Unit. *International Journal of Advanced Research in Computer Science and Software Engineering* 2014. p. 531-549.
8. Yegnanarayana B. *Artificial Neural Networks*. New Delhi: Prentice-Hall; 2012.
9. Haykin S. *Neural Networks: A comprehensive foundation*. 2nd ed. New Delhi: Prentice-Hall; 2006.
10. Boersma, Weenik. *PRAAT: A tool for phonetic analysis and sound manipulations*. www.praat.org.
11. Badrinath K. *Practical Hindi-English Dictionary*. India: Prabhat Prakashan; 2012.