2022 3rd Int. Conf. on Innovations in Science, Engineering and Technology (ICISET)
26-27 February 2022, Chittagong, Bangladesh

Outcomes of Deep Neural Network Hyperparameter Tuning on Bengali Speech Token Classification

Md Rakibul Hasan
Department of Electrical and Electronic Engineering
Khulna University of Engineering & Technology, Khulna
and BRAC University, Dhaka
Bangladesh
rakibul.hasan@bracu.ac.bd

Md. Mahbub Hasan
Department of Electrical and Electronic Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh
mahbub01@eee.kuet.ac.bd

Md Zakir Hossain
Biological Data Science Institute
Australian National University
and CSIRO A&F, Black Mountain
Canberra, Australia
zakir.hossain@anu.edu.au

2022 International Conference on Innovations in Science, Engineering and Technology (ICISET) | 978-1-6654-8397-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICISET54810.2022.9775837

***Abstract*—Speech-based human-computer interaction is a vital research area where Bengali speech requires substantial research for being efficient like English. Nowadays, most speech-related research involves Deep Neural Networks (DNNs). We present comprehensive results on DNN hyperparameter tuning on isolated Bengali vowel and word classification. Two separate datasets of isolated Bengali vowels and words are utilized in a four-hidden-layered DNN configuration. Five formant frequencies have been extracted for all fourteen vowels and words, and after applying a necessary feature standardization technique, these acoustic features are utilized in the DNN input layer. Three hyperparameters—optimizer, batch size, and dropout—have been tuned to develop an optimum DNN classification model. We report the outcome achieved after applying different sets of values for these three hyperparameters. Finally, we recommend an Adadelta optimizer, a batch size of 128 or 256, and a dropout between 0.25 to 0.4 after the first hidden layer. The DNN formed by utilizing these recommended hyperparameters will be a potential pre-configured speech classifier for various human-computer interaction-based devices and systems.**

***Index Terms*—Hyperparameter tuning, Deep neural network, Bengali speech recognition, Optimizer, Batch size, Dropout**

I. Introduction

Having 267.7 million total users in total, Bengali is the 6th most spoken language in the world, including 228.7 million native speakers [1]. However, speech recognition research on the Bengali language is not comparable to high-resource languages like English. For this reason, efficient uses of Bengali in real human-machine interaction devices are still facing challenges. Support of Bengali language in popular speech recognition services like Google Assistant, Apple Siri, Amazon Alexa, and Microsoft Cortana is either not efficient or not supported at all. To eradicate these issues, there is no other option than more investigations in these domains.

Most state-of-the-art speech recognition researches involve Deep Neural Networks (DNNs), at least to some extent. Whether it is a basic feedforward DNN or a more advanced architecture like Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN), it consists of fully-connected DNNs. Therefore, the proper development of a DNN model has significant importance.

A basic DNN or a more advanced CNN model can solve most classification problems. In many cases, a basic DNN model is preferred over a more complex CNN model due to its simplicity. For example, recently, authors in [2] have demonstrated a complex classification task employing simpler DNNs (also called multilayer perceptrons) rather than a CNN. Moreover, a more simplistic and efficient model is preferable for deploying the model in low-computation-capable end devices such as smartphones, Arduino, and Raspberry Pi. That is why we have selected a basic DNN for the isolated vowel and word classification.

Speech token (isolated vowels, words, etc.) classification has several crucial applications, such as emotion recognition, speech dictation in vehicles, smartphones, and laptops, home automation, and assisting physically-challenged and old-age people [3]. Yusof et al. classified five Malaysian vowels using a DNN with formant frequency features [4]. Authors in [5] classified read and conversational speech of four Indian languages, including Bengali, by utilizing a single-layer feedforward neural network model. Furthermore, they used a multi-layer DNN model for phoneme recognition. Authors in [6] classified ten English command words employing DNN and CNN. Similarly, reference [7] reported classification of twenty Dari speech tokens using CNN.

There are several works reported on the Bengali language as well. Reference [8] used CNN to classify ten Bengali spoken digits. Furthermore, authors in [9] used CNN and RNN for isolated Bengali speech recognition. The above researches have reported their specific model and performance, with outcomes of one/two hyperparameter variations. In contrast, we report comprehensive results on three specific hyperparameter variations, indicating which hyperparameter to select in similar studies.

Information or features are often retrieved from raw speech sounds through appropriate feature extraction techniques. The resonance frequencies of speakers' vocal tract, known as formant frequency, are used in many speech-related researches, such as classifying Malaysian vowels [4], Turkish vowels [10], and finding the place of articulation [11]. Our work utilized formant frequency as the acoustic cue for the DNN hyperparameter tuning in both Bengali vowel and word classification. To the best of our knowledge, research reports on DNN hyperparameter tuning are rarely found except a few studies [12], [13].

The specific contributions of this article include the outcome of three specific and essential hyperparameter tuning, from where we can conclude which set of hyperparameters are more suitable for such a Bengali speech token classification model. Future research on similar classification may directly employ these optimum hyperparameters without further tuning, or at least they can get a hint on where to tune. Furthermore, this research presents a Bengali speech token recognition pipeline using a DNN-based classification model.

The rest of this article is organized as follows. Section II presents relevant background on what is hyperparameters and three specific hyperparameters we worked on. The datasets, feature extraction, and the DNN architecture are illustrated in Section III. Then, the results on three hyperparameters' tuning are presented and described in Section IV. Finally, the article is summarized in Section V.

II. Deep Neural Network Hyperparameters

A. Hyperparameter vs. Parameter

Hyperparameters are some specific settings, proper values of which need to be set externally to define the behavior of learning algorithms [14]. Generally, these values of hyperparameters are not adapted automatically by the learning algorithm itself. Instead, we need to select the optimum value through a series of experiments with different values. The process of finding out the optimum value of these hyperparameters is called hyperparameter tuning [12].

Another relevant terminology to hyperparameter is the parameter that consists of weights and biases of a DNN—the optimum values of which are adapted by the learning algorithm itself through feedback mechanism [15]. Important DNN hyperparameters include the number of hidden layers, number of hidden neurons on the hidden layers, loss function, activation functions, optimizer, batch size, dropout rate, number of epochs, and so on [13].

B. Optimizer

To find out the best set of parameter values through training, we minimize (or maximize in some cases) the cost (also called loss) function. This task of minimizing or maximizing is called optimization, which is done by an optimizer. The most common example of such an optimizer is gradient descent that works based on the derivative of the cost function [14]. There are several variations built on top of this gradient descent optimization method, such as *Stochastic Gradient Descent (SGD), Adadelta, RMSProp,* and *Adam* [16]. Each of them has pros and cons related to training time, accuracy score, and convergence rate.

C. Batch Size

Batch size refers to the number of training samples passed at once while optimizing (training) via gradient descent-based optimizers. The batch size equal to one refers to the *SGD* optimizer where only one sample is passed at once. All other optimizers mentioned above (*Adadelta, RMSprop,* and *Adam*) have the batch size hyperparameter that we need to specify before training the DNN model.

D. Dropout

Dropout is a powerful regularization technique used in deep learning [14]. Why do we need regularization? A central difficulty in deep learning is the performance difference in using training data versus test data (i.e., new input). Better performance means the model gives a lower loss score or error and a higher classification accuracy or other related evaluation metrics. DNN models often perform well on training data, but performance drops at new input samples in many cases. Such cases (a too large gap between training and testing scores) are called overfitting [14]. In its simplest form, dropout means randomly ignoring (zeroing out) some hidden neurons of a model. The dropout rate refers to the fraction of the weights zeroed out during training. No neurons are dropped during the testing phase; instead, the output is scaled by the dropout rate to compensate for the zeroing out [15].

III. Methods

A. Dataset

We experimented with hyperparameter tuning on two separate classifications: seven Bengali vowel sounds in one classification and seven Bengali word sounds in another. The isolated vowels were /অ/[/ɔ/], /আ/[/a/], /ই/[/i/], /উ/[/u/], /ঋ/[/ri/], /এ/[/e/], and /ঐ/[/oi/]; the isolated words were বোতল, বন, কপি, দোকান, শেষ, সঠিক, and উপরে. These speech tokens were collected from 20 male and female volunteers aged between 20 to 26 years using a typical smartphone. Instructing the volunteers to pronounce the speech tokens in two different accents, 40 isolated utterances were prepared in each of the 14 speech tokens. Thus, there were $40 \times 14 = 560$ speech samples in total. Detailed description and data curation process are available at [17]. The complete datasets (vowels and words) are publicly available for research purposes [18].

B. Feature Extraction & Standardization

We chose formant frequency as an acoustic feature that holds speech information. We extracted five formant frequencies by taking samples at every 6 ms interval in a window length of 25 ms using PRAAT script [19]. Since these features had values at different ranges, we standardized them according to (1).

$$x'_{F_i} = \frac{x_{F_i} - \overline{x_F}}{\sigma_F}; \qquad i = 1, 2, \ldots, N \tag{1}$$

where, x_F represents each of the five formant frequencies (F1, F2, F3, F4, and F5), N is the total number of samples for each formant frequency, $\overline{x_F}$ and σ_F are the mean and standard deviation of corresponding formant frequency, and x'_{F_i} is the standardized features having a mean of zero and variance of one. These features were supplied in the input layer of the DNN-based classification model. The above standardization process essentially helps DNNs and other machine learning algorithms to converge faster [20].

C. Deep Neural Network Architecture

We utilized a four-hidden-layered DNN architecture to experiment with previously mentioned hyperparameters. As depicted in Fig. 1, the number of hidden neurons in these hidden layers were 128, 64, 32, and 16, respectively.

Fig. 1. Four-hidden-layered DNN configuration used in this study. The same model was used for both vowel and word classifications.

Seven neurons in the output layer represent seven vowel classes or seven word classes, depending on whether it is the vowel or word classification. All hidden layers had *tanh* activation function, whereas the output layer had *softmax* activation function since it is a multi-class classification model. We utilized *categorical cross-entropy* as the cost or loss function. As mentioned previously, these values—number of hidden layers, neurons, loss, and activation functions—are also hyperparameters. We also came up with these hyperparameters' values through tuning, but it is out of the scope of this article.

We evaluated classification *accuracy* as the metric to compare among hyperparameters. Along with correct predictions, DNN often mispredicts some sample inputs unless the accuracy is 100%. For illustrations, let us consider any specific class as "class X." Now, any "class X" sample predicted as "class X" is a *true positive* prediction, and any "not class X" sample predicted as "not class X" is a *true negative* prediction. Correct predictions include *true positive* and *true negative* predictions, whereas inaccurate predictions are any "class X" sample predicted as "not class X," or any "not class X" sample predicted as "class X." As shown in (2), the *accuracy* is the ratio of correct predictions to all predictions (both correct and incorrect).

$$\text{Accuracy} = \frac{\text{True positive} + \text{True negative}}{\text{All predictions}} \tag{2}$$

IV. Results & Discussion

A. Changes in Optimizer

We used a batch size of 32 in the following experiments with *Adadelta*, *RMSprop*, and *Adam* optimizers. The learning rate was the Keras default value of 0.001, and other optimizer parameters were also the default value available in the optimizer functions of Keras (Keras [21] is a software library we used to implement the DNN-based classification). Fig. 2 depicts vowel classification training and validation accuracy curves using *Adadelta* optimizer. After training and validating for 50 epochs, the model achieved around 94% training accuracy and 86% validation accuracy.

Fig. 2. Vowel classification accuracy with *Adadelta* optimizer at batch size = 32.

The same classification model optimized with *RMSprop* optimizer achieved 90% training accuracy and 80% validation accuracy with the same number of training and validation epochs, as shown in Fig. 3.

Fig. 3. Vowel classification accuracy with *RMSprop* optimizer at batch size = 32.

Furthermore, we tested the same classification model with *Adam* optimizer with the same number of training and validation epochs. It achieved 75% training accuracy and 65% validation accuracy shown in Fig. 4.

Fig. 4. Vowel classification accuracy with *Adam* optimizer at batch size = 32.

Tho above comparisons reflect that *Adadelta* optimizer performed better with the classification model at batch size = 32.

B. Changes in Batch Size

The following experiments with different batch sizes utilized *Adam* optimizer. In Fig. 4, we have already observed 75% training accuracy and 65% validation accuracy at batch size = 32. Increasing the batch size to 128 yielded a training and validation accuracy of around 97.5% and 80%, respectively. The corresponding accuracy curves with epochs are shown in Fig. 5.

Fig. 5. Vowel classification accuracy at batch size = 128.

Increasing the batch size further to 200 did not improve the classification performance. In the training and validation curves shown in Fig. 6, it achieved 88% training accuracy and 80% validation accuracy. This training accuracy is noticeably lower than what we observed at batch size = 128.

Finally, a batch size of 256 resulted in 97% training and 80% validation accuracy shown in Fig. 7.

Therefore both training and validation accuracy were higher for batch size = 128 and 256. Nevertheless, the same validation accuracy was observed on batch size = 128, 200, and 256.

C. Changes in Dropout

As discussed previously, dropout helps reduce overfitting—it decreases the distance between training and validation scores. Table I presents the effect of different dropout rates after the first, second, and third hidden layers. Loss and accuracy scores at both training and validation phases are reported with corresponding differences.

Fig. 6. Vowel classification accuracy at batch size = 200.

Fig. 7. Vowel classification accuracy at batch size = 256.

Without dropout, the table shows noticeable differences between training and validation scores—a difference of 0.2769 in loss and 0.1004 in case of accuracy. When a dropout of 0.15 was applied after the first hidden layer, the loss difference reduced to 0.2168, and the accuracy difference dropped to 0.0701, which proves a reduction of overfitting. However, at the same time, validation accuracy also got lowered.

Additionally, we tested applying dropout after the second and third hidden layers. Only a 0.15 dropout rate after both second and third hidden layers showed similar outcomes of either 0.40 or 0.45 dropout rate after only the first hidden layer. Lastly, when dropout was applied after the first three hidden layers, even a small dropout of 0.15 showed the least overfitting here, but at the same time, the least accuracy. Therefore, there was no point in adding more dropouts after this stage.

We further demonstrate the effect of dropout change graphically. Fig. 8 presents word classification loss and accuracy curves for both training and validation phases when four different dropout rates were applied after the first hidden layer. The classification model was trained and validated for 300 epochs here to thoroughly observe the dropout effect.

Fig. 8a depicts the curves when a dropout of 0.25 was applied after the first hidden layer. Fig. 8b shows the effect of replacing the above dropout by 0.35. Increasing the dropout rate from 0.25 to 0.35 resulted in a closer distance

TABLE I
WORD CLASSIFICATION LOSS AND ACCURACY AT DIFFERENT DROPOUT RATES. A SINGLE DROPOUT RATE INDICATES THAT IT IS APPLIED AFTER ONLY THE FIRST HIDDEN LAYER.

	Loss			Accuracy		
Dropout Rate(s)	**Training**	**Validation**	**Difference**	**Training**	**Validation**	**Difference**
0	0.7795	1.0564	0.2769	0.7218	0.6214	0.1004
0.15	0.8458	1.0626	0.2168	0.6882	0.6181	0.0701
0.25	0.8793	1.0332	0.1539	0.6778	0.6318	0.046
0.35	0.9423	1.0410	0.0987	0.6474	0.6158	0.0316
0.40	0.9643	1.0579	0.0936	0.6360	0.6107	0.0253
0.45	1.0050	1.0702	0.0652	0.6220	0.6039	0.0181
0.15, 0.15[a]	0.9759	1.0539	0.078	0.6363	0.6054	0.0309
0.25, 0.15[a]	1.0095	1.0826	0.0731	0.6252	0.6028	0.0224
0.15, 0.15[a], 0.15[b]	1.1515	1.1535	0.002	0.5637	0.5663	0.0026

[a] After second hidden layer
[b] After third hidden layer

(a) Dropout = 0.25

(b) Dropout = 0.35

(c) Dropout = 0.40

(d) Dropout = 0.45

Fig. 8. Word classification loss and accuracy curves when four different dropout rates were applied after the first hidden layer.

between training and validation curves. Replacing the dropout by 0.40 further reduced the overfitting (Fig. 8c). Finally, Fig. 8d shows that further increase of dropout to 0.45 almost diminished the overfitting, but at the same time, the accuracy score got lowered.

Therefore, dropout does reduce overfitting, but in the same instance, it reduces validation accuracy that we genuinely care about. Thus, we should balance between overfitting and validation accuracy. Comparing all these dropout values in Table I and Fig. 8, we recommend a

dropout rate between 0.25 to 0.4 after only the first hidden layer to keep the balance.

D. Comparative Analysis

We achieved the highest accuracy of 86% using *Adadelta* optimizer on vowel classification. The closest related work [4] classified five Malaysian vowels using DNN with formant frequency features. After different hyperparameter tuning with adaptive learning rates, their best accuracy was 87.04% using six formant frequency-related features (three raw formants and their differences). Although the primary aim of our study is to demonstrate outcomes on DNN hyperparameter tuning rather than presenting the best accuracy, our result—with only five raw formant features and without any adaptive learning rate—is quite competitive. Our future work will include other DNN architectures, such as CNN and RNN, to improve the performances and to report comparable accuracy with other works in the domain.

V. Conclusion

Although the Bengali language is quite widespread, research on it to use as real communication devices is not that widespread. Speech token recognition—for example, classification of isolated vowels and words—has several applications, such as emotion recognition, speech dictation gadgets, home automation, etc. DNN is a well-known algorithm used in many studies to classify these tokens. One of the critical steps in implementing DNN is to select optimum values of its hyperparameters, such as optimizer, batch size, and dropout rate. We report hyperparameter searching outcomes on DNN-based isolated Bengali vowel and word classification. Experimental results reveal several key findings. Firstly, *Adadelta* optimizer performs better at batch size = 32. Secondly, training and validation accuracy are higher at batch size = 128 and 256. Thirdly, dropout reduces overfitting, but at the same time, it reduces validation accuracy, and to keep a balance between these two, we recommend a dropout rate between 0.25 to 0.4 after only the first hidden layer. The outcomes of this article would be useful in several cases. Future research on similar speech-token classification may utilize these hyperparameter values to develop an optimized and robust DNN classification model. The DNN configuration presented in this article can also be used as a base classification model in similar studies. Thus, it will help boost research on the Bengali language, and with research, we hope to compete with the rapidly changing world.

References

[1] D. M. Eberhard, G. F. Simons, and C. D. Fennig, Eds., *Ethnologue: Languages of the World*, 24th ed. Dallas, Texas: SIL International, 2021. [Online]. Available: https://www.ethnologue.com

[2] H. Touvron, P. Bojanowski, M. Caron, M. Cord, A. El-Nouby, E. Grave, G. Izacard, A. Joulin, G. Synnaeve, J. Verbeek, and H. Jégou, "Resmlp: Feedforward networks for image classification with data-efficient training," *arXiv*, 2021. [Online]. Available: https://arxiv.org/abs/2105.03404

[3] M. R. Hasan, M. M. Hasan, and M. Z. Hossain, "How many mel-frequency cepstral coefficients to be utilized in speech recognition? a study with the bengali language," *The Journal of Engineering*, vol. 2021, no. 12, pp. 817–827, 2021.

[4] S. A. M. Yusof, P. M, and S. Yaacob, "Classification of malaysian vowels using formant based features," *Journal of Information and Communication Technology*, vol. 7, pp. 27–40, 2008.

[5] K. Tripathi, M. K. Reddy, and K. S. Rao, "Multilingual and multimode phone recognition system for indian languages," *Speech Communication*, vol. 119, pp. 12–23, 2020.

[6] X. Yang, H. Yu, and L. Jia, "Speech recognition of command words based on convolutional neural network," in *2020 International Conference on Computer Information and Big Data Applications (CIBDA)*. IEEE, 2020, pp. 465–469.

[7] M. Dawodi, J. A. Baktash, T. Wada, N. Alam, and M. Z. Joya, "Dari speech classification using deep convolutional neural network," in *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. IEEE, 2020, pp. 1–4.

[8] R. Sharmin, S. K. Rahut, and M. R. Huq, "Bengali spoken digit classification: A deep learning approach using convolutional neural network," *Procedia Computer Science*, vol. 171, pp. 1381–1388, 2020.

[9] J. Islam, M. Mubassira, M. R. Islam, and A. K. Das, "A speech recognition system for bengali language using recurrent neural network," in *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2019, pp. 73–76.

[10] Y. Korkmaz and A. Boyacı, "Classification of turkish vowels based on formant frequencies," in *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*. IEEE, 2018, pp. 1–4.

[11] B. H. Story and K. Bunton, "Relation of vocal tract shape, formant transitions, and stop consonant identification," *Journal of Speech, Language, and Hearing Research*, vol. 53, no. 6, pp. 1514–1528, Dec. 2010.

[12] T. K. Dugar, S. Gowtham, and U. K. Chakraborty, "Hyperparameter tuning for enhanced authorship identification using deep neural networks," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2019, pp. 206–211.

[13] F. F. Firdaus, H. A. Nugroho, and I. Soesanti, "Deep neural network with hyperparameter tuning for detection of heart disease," in *2021 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. IEEE, 2021, pp. 59–65.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. USA: MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org

[15] F. Chollet, *Deep learning with Python*, 1st ed. USA: Manning Publications, 2018.

[16] E. Okewu, P. Adewole, and O. Sennaike, "Experimental comparison of stochastic optimizers in deep learning," in *Computational Science and Its Applications – ICCSA 2019*, S. Misra, O. Gervasi, B. Murgante, E. Stankova, V. Korkhov, C. Torre, A. M. A. Rocha, D. Taniar, B. O. Apduhan, and E. Tarantino, Eds. Cham: Springer International Publishing, 2019, pp. 704–715.

[17] M. R. Hasan and M. M. Hasan, "Investigation of the effect of mfcc variation on the convolutional neural network-based speech classification," in *2020 IEEE Region 10 Symposium (TENSYMP)*. IEEE, Jun. 2020, pp. 1408–1411.

[18] M. R. Hasan and M. M. Hasan, "Isolated bengali vowel and word speech sounds," *Mendeley Data,* V1, 2021, [Dataset].

[19] P. Boersma and D. Weenink, "Praat, a system for doing phonetics by computer," *Glot international*, vol. 5, no. 9, pp. 341–345, Jan. 2001.

[20] J. Grus, *Data science from scratch: first principles with python*, 1st ed. USA: O'Reilly Media, 2019.

[21] F. Chollet *et al.*, "Keras," 2015. [Online]. Available: https://keras.io