computers

MDPI

Article

Learn2Write: Augmented Reality and Machine Learning-Based Mobile App to Learn Writing

Md. Nahidul Islam Opu [1,†], Md. Rakibul Islam [1,†], Muhammad Ashad Kabir [2,*], Md. Sabir Hossain[1] and Mohammad Mainul Islam [3]

[1] Department of Computer Science and Engineering, Chittagong University of Engineering and Technology, Chottogram 4349, Bangladesh; u1604073@student.cuet.ac.bd (M.N.I.O.); u1604060@student.cuet.ac.bd (M.R.I.); sabir.cse@cuet.ac.bd (M.S.H.)

[2] Data Science Research Unit, School of Computing, Mathematics and Engineering, Charles Sturt University, Bathurst, NSW 2795, Australia

[3] Verizon Media Australia, Sydney, NSW 2015, Australia; sujan.cse.cuet@gmail.com

* Correspondence: akabir@csu.edu.au; Tel.: +61-2-633-86259

† These authors contributed equally to this work.

Abstract: Augmented reality (AR) has been widely used in education, particularly for child education. This paper presents the design and implementation of a novel mobile app, *Learn2Write*, using machine learning techniques and augmented reality to teach alphabet writing. The app has two main features: (i) guided learning to teach users how to write the alphabet and (ii) on-screen and AR-based handwriting testing using machine learning. A learner needs to write on the mobile screen in on-screen testing, whereas AR-based testing allows one to evaluate writing on paper or a board in a real world environment. We implement a novel approach to use machine learning for AR-based testing to detect an alphabet written on a board or paper. It detects the handwritten alphabet using our developed machine learning model. After that, a 3D model of that alphabet appears on the screen with its pronunciation/sound. The key benefit of our approach is that it allows the learner to use a handwritten alphabet. As we have used marker-less augmented reality, it does not require a static image as a marker. The app was built with ARCore SDK for Unity. We further evaluated and quantified the performance of our app on multiple devices.

Keywords: mobile app; augmented reality; machine learning; alphabet learning; handwriting recognition

check for updates

Citation: Opu, M.N.I.; Islam, M.R.; Kabir, M.A.; Hossain, M.S.; Islam, M.M. *Learn2Write*: Augmented Reality and Machine Learning-Based Mobile App to Learn Writing. *Computers* **2022**, *11*, 4. https://doi.org/10.3390/computers11010004

Academic Editor: Paolo Bellavista

Received: 4 December 2021
Accepted: 22 December 2021
Published: 27 December 2021

1. Introduction

Writing is an important part of the learning process [1]. The traditional way of teaching writing involves a pencil and paper, or chalk and a chalkboard [2]. Digital content and the mobile app-based writing process replace paper or a chalk board with an electronic display screen (e.g., mobile, tablet, etc.) and a pencil or chalk with a stylus or bare hand. This process also replaces a human instructor to guide the learner and is mainly based on tracing [3]. In tracing methods, learners move their finger or stylus following points or an arrow or the actual letter on the screen to write a letter [4,5].

Augmented reality is a technology combining the real and virtual worlds. It is used in numerous areas, including education, to perceive real experiences [6]. One can learn effectively through the seamless interaction between the real and virtual environments. The ability to change the position and orientation of the virtual object means this technology can create enhanced contemporary educational environments and enriched learning opportunities for students [7,8]. The visualization and realization of any learning topic become more enjoyable using this technology [9]. There are various ways that augmented reality can be used in education, such as a head-mounted display, handheld displays, and pinch gloves [10]. For this research, we have chosen a smart phone-based augmented reality, which lies in the handheld device category.

Many research studies have reported the uses and benefits of AR technology in education, including early childhood education and preschool settings [11–13]. Several studies [5,14–17] have been conducted on teaching and testing handwriting through digital devices. Some of them (e.g., [15,16]) used mobile apps with the tracing method to teach handwriting, whereas others (e.g., [17]) adopted AR to make the learning enjoyable. However, none of them include a handwriting evaluation feature.

Some studies have focused on automated handwriting testing using machine learning and image processing [18,19]. Although all these works have focused on handwriting teaching and evaluation, none used AR or provided the feature to evaluate handwriting on paper or a board.

Our work includes augmented reality for evaluating handwriting besides on-screen testing. For learning, we developed a step-by-step line tracing guide to show how to write each letter of the alphabet. We support both the Bangla and English languages. The evaluation system is based on a deep learning model. In our app, when a user places the camera on a handwritten alphabet, the writing is evaluated by a machine learning model trained by us and a corresponding 3D model of the identified alphabet is displayed using marker-less augmented reality. In addition, the sound of the alphabet is played in the background. In other words, our app will display a virtual object with a virtual letter while simultaneously playing a sound in the background. As a result, it provides the learner with an immersive learning environment.

We used marker-less augmented reality in this research. Marker is a simple image that is used to trigger the AR engine to show the 3D virtual model on a mobile screen [20]. For marker-based augmented reality, learners should have the printed marker to work. To mitigate the need for a static marker, we have used surface tracking to instantiate a 3D model of the detected alphabet during the AR session. We have trained and used a machine learning model to detect and evaluate handwritten letters on paper and on-screen.

The machine learning model we used here is a convolution neural network (CNN). We have conducted an empirical evaluation by training several prominent CNN models on the same dataset and comparing them based on several criteria to select the best model for our app. Our empirical evaluation considers several matrices: model accuracy, loading time, testing time, and model size. The models used in our experiment are BornoNet [21], EkushNet [22], DenseNet [23], Xception [24], and MobileNetV2 [25]. Finally, we have selected EkushNet for our mobile app due to its faster loading and testing time and smaller size, which are important factors for the performance of a mobile app.

The rest of the paper is organized as follows. Section 2 discusses the related apps and research on handwriting teaching and evaluation. Our app architecture is presented in Section 3. In Section 4, we describe our app's features and uses. After presenting the experimental evaluation results in Section 5, we conclude the paper in Section 6.

2. Related Work

2.1. Mobile Apps to Learn to Write the Alphabet

Many apps are available in the Google Play store and Apple app store to teach handwriting. We selected apps for review based on several criteria, such as the app store rating, number of downloads, and user reviews. Finally, we have selected five leading apps—iTrace [26], ABC kids [15], Writing Wizard [27], Kids Learn Bangla Alphabet [28], and Hatekhori [16]. All of them use the tracing method to teach writing. However, they do not have any testing features to check the learners' handwriting.

Marker-based narrator AR [17] provides custom markers through their website that require printer support. Kids can write on the custom marker and AR shows the writing using 3D animation. Specifically, this method does not teach the learners; rather it makes writing enjoyable through visualization.

2.2. Research Studies

Recently, some studies have been conducted on teaching and testing learners' writing. Tablet-based handwriting for Arabic words is proposed in [29]. An Android application on a Samsung Galaxy Note smartphone was used to test the potential of tablet-based handwriting evaluation and to compare it with paper-based learning. Alvaro et al. [4] developed software to use with a stylus on a touch screen monitor. All the characters were divided into six modules according to the number and type of strokes (vertical, horizontal, cursive, circular). A GIF image is used to teach and a neural network is used to test the learners' written characters.

Maxim et al. [30] developed a game in which handwriting is taught and tested in each level of the game to go to the next level. El-Sawy et al. [31] developed a system for the Arabic alphabet. The testing was based on the written character's stroke count and stroke order, using fuzzy logic. Al-Barhamtoshy et al. [19] provided a guided system to teach. The learner needs to write over the given character to learn. The first level involves writing a single character. The highest level involves writing complete words, and in this case, the word is segmented into characters using a machine learning model. They also developed a method to test the learners using a machine learning technique. Sybenga and Rybarczyk [18] used the MNIST dataset, image processing, and several machine learning algorithms including Naive Bayes to test handwriting. They built a mobile app for this purpose.

Yanikoglu et al. [5] developed a system to teach and test the Turkish language with guided teaching using tracing. It also included modules for handwriting and arithmetic, which were thought to benefit the most from handwriting recognition technologies. Iwayama et al. [32] developed a similar system but with Kanji letters. They created handwriting-based learning materials and built an online handwriting recognition module to check handwriting. They surveyed students at a school and found that automatic checking saved time and increased the motivation of the students to learn.

Jordan et al. [33] developed a mobile application called *LetterSchool* that teaches handwriting in both visual and guided ways. However, the app does not have a feature for testing. Hu et al. [34] focused on the Chinese language. The authors only developed a system to test handwriting. As an intelligent tutor, the system can detect and correct handwriting errors such as incorrect stroke production, sequence, and relationship. They used attributed relational graph matching to locate the handwriting errors, and a pruning strategy was used to reduce the computational time. Hape et al. [14] conducted a study on the effects of the mobile application called "Handwriting Without Tears" on first graders. However, they did not experiment to test the students. The study provided preliminary evidence supporting the effectiveness of implementing their app's curriculum within first-grade classrooms.

Table 1 summarizes and contrasts all the above discussed apps and research and how they are different from our *Learn2Write* app. Only [5] has guided teaching functionality with on-screen testing, but it only supports the Turkish language and does not have AR-based testing. To the best of our knowledge, there are no other apps or research studies that have used AR for testing handwriting as we have done.

Table 1. Literature review summary and comparison.

Literature		Language	Alphabet	Digits	Guided Teaching	Testing		
						On Screen	AR	Approach
App-store Apps	[26]	English	✓	✓	✓	✗	✗	–
	[15]	English	✓	✗	✓	✗	✗	-
	[27]	English	✓	✓	✓	✗	✗	-
	[28]	Bangla	✓	✓	✓	✗	✗	-
	[16]	Bangla, English	✓	✗	✓	✗	✗	-
	[35]	Bangla	✓	✗	✓	✗	✗	-
	[36]	Bangla, English	✓	✓	✓	✗	✗	-
Research	[4]	English	✓	✗	✗	✓	✗	Neural Network
	[30]	English	✓	✗	✗	✓	✗	Rule Based
	[31]	Arabic	✓	✗	✗	✓	✗	Fuzzy Logic
	[19]	Arabic	✓	✗	✓	✓	✗	HMM and SVM
	[18]	English	✓	✓	✗	✓	✗	Image Processing & ML
	[5]	Turkey	✓	✓	✓	✓	✗	-
	[32]	Kanji, English	✓	✓	✗	✓	✗	-
	[33]	English	✓	✓	✓	✗	✗	-
	[34]	Chinese	✓	✓	✗	✓	✗	Template matching
	[14]	English	✓	✓	✓	✗	✗	-
	[29]	Arabic	✓	✗	✓	✗	✗	-
	Our	English, Bangla	✓	✓	✓	✓	✓	Deep Learning

3. Learn2Write App Architecture

Figure 1 shows the architecture of our Learn2Write app. The app is developed with the Unity3D game engine, and C-sharp is the primary programming language used. The app does not require an internet connection to function because it includes all the 3D models of alphabets and the trained machine learning model. On the left side of Figure 1 the building of this deep learning model is shown. It is then converted to .nn model from .h5 format and integrated into the app. We used the Unity "Barracuda" library [37] to run our machine learning model.

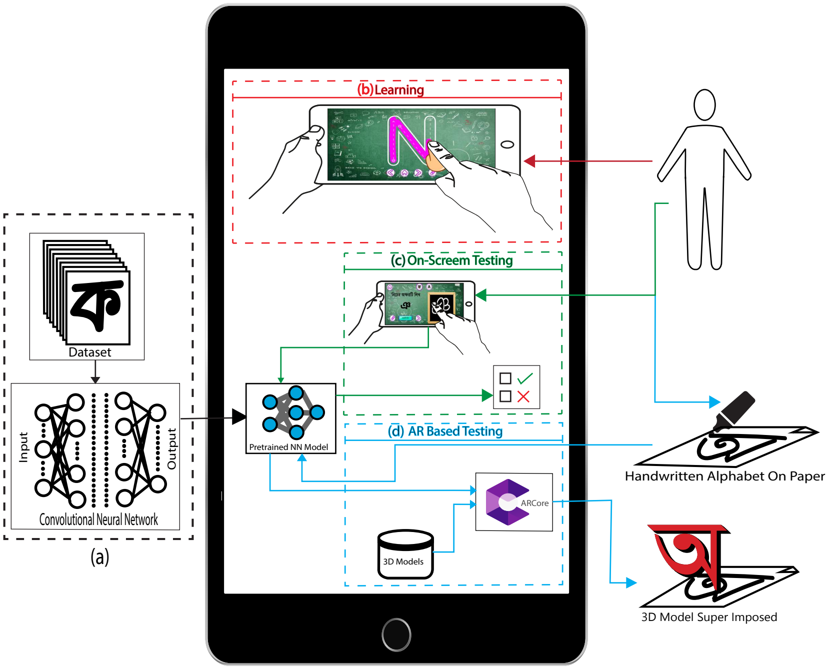

Figure 1. Working procedure of our developed app. (**a**) Deep learning model creation with CNN and saving the model. (**b**) On screen learning with the tracing method. (**c**) On screen testing based on the deep learning model. (**d**) AR-based testing with ARCore to handle 3D models and interaction with deep learning based testing.

The app has three important features: alphabet learning, on-screen testing, and augmented reality testing. In the learning module (Figure 1b), the learner can choose an alphabet, and step-by-step instructions are presented to teach the user how to write that alphabet. In the on-screen test module (Figure 1c), the learner is asked to write a specific alphabet using a finger on the mobile screen and the trained machine learning model is used to test whether the written alphabet is correct or not. In augmented reality testing (Figure 1d), the learner writes a given alphabet on paper and the app captures a photo of the paper to apply the machine learning model to test. The app places a 3D model of the detected alphabet in AR to make the testing session more enjoyable.

3.1. ARCore and 3D Models

ARCore SDK [38] is one of the modern solutions of marker-less augmented reality. To use marker-less technology, it is required to detect a surface first. The surface can be horizontal or vertical. ARCore can easily detect both surfaces. It can also refer to a flat table or chair surface.

ARCore aims build augmented reality applications to enrich the real world with additional 2D or 3D digital information using a smartphone or tablet [39]. To use the ARCore [38] enabled application, it is essential to install an additional library, which can be downloaded from the Google Play store. There are three main features of ARCore [38,39]:

- Tracking: allows the phone to understand and track its position relative to the world;
- Environment Understanding: allows the phone to detect the size and location of all type of surfaces, such as horizontal, vertical, and angled surfaces like the ground, a coffee table, or walls;
- Light Estimation: allows the phone to estimate the environment's current lighting conditions

We have created a 3D model for each letter in the Bangla and English alphabet. Our models are created by Maya [40]. Figure 2 shows our developed 3D models samples of the English and Bangla alphabet and digits.

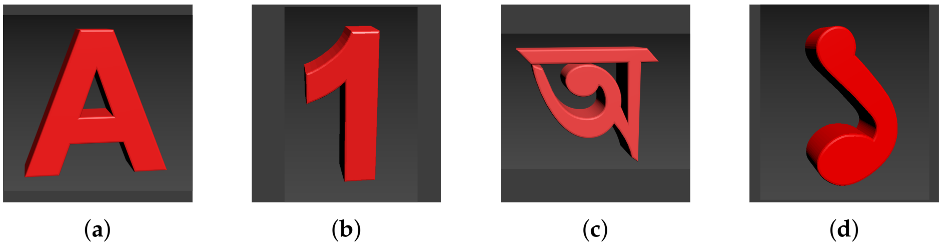

Figure 2. Screenshots of sample 3D models used in our app. (**a**) English alphabet. (**b**) English digit. (**c**) Bangla alphabet. (**d**) Bangla digit.

3.2. Machine Learning Models

The heart of this application is a machine learning model. We have considered five state-of-the-art deep learning architectures and conducted an empirical study (see Section 5) to select the best architecture for training our model. In this section, we briefly introduce those architectures. BornoNet [21] and EkushNet [22] are state-of-the art models for Bangla handwritten character classification. DenseNet [23], Xception [24], and MobileNetV2 [25] are prominent deep learning architectures.

BornoNet [21] consists of 13 layers with 2 sub-layers: Convolutional, pooling, and fully connected layers, as well as dropout as a regularization method, are used to construct this network. It has three dropout layers to prevent overfitting. It uses ReLU activation. The final output layer has softmax activation.

EkushNet [22] has a total of 23 layers including 10 sub-layers. This model has a convolutional layer, a max pooling layer, and a fully connected layer. Different regularization methods such as batch normalization and dropout are used. After the fifth layer, the network is divided into two sections—one has four layers and the other has six. These 2 sections are merged on the 16th layer. The final layer requires some output nodes with softmax activation.

DenseNet [23] is a densely connected CNN where each layer is connected to all the previous layers. Thus, it creates a very dense network. It requires fewer parameters than equivalent traditional CNNs.

Xception [24] architecture is made up of 36 convolutional layers. Except for the first and last modules, the 36 convolutional layers are divided into 14 modules with linear residual connections surrounding them. In a nutshell, the Xception architecture is a depth-wise separable convolution layer stack with residual connections. The pointwise convolution is followed by a depth-wise convolution in the Xception.

MobileNetV2 [25] architecture is specially designed for mobile and embedded vision applications with reduced computation. It is a lightweight architecture. It uses depth-wise convolution followed by pointwise convolution, called depth-wise separable convolution instead of normal convolution. This change reduces the number of parameters, thereby reducing the total number of floating-point multiplication operations. However, there are some sacrifices of accuracy. MobileNetV2 is made up of two types of blocks. One is a one-stride residual block. Another option for downsizing is a block with a stride of two. Both blocks have three layers—1 × 1 convolution with ReLU is the first layer, the depth-wise convolution is the second layer, and another 1 × 1 convolution is used in the third layer.

4. App Description

The AR app consists of two main features—one is learning and the other is testing. The testing module determines whether the user can write a given character independently. There are two testing features in the app—on-screen testing and AR-based testing. The on-screen testing and learning is done by the guided learning section. It provides instructions to draw an alphabet on the screen. This part does not require augmented reality. A normal android phone can run this part of our app very smoothly. However, in AR-based testing, we have used a marker-less AR technology called ARCore SDK. AR-based testing allows students to check their written alphabet in the real-world environment. For this part, the app needs white paper or a board. The learner has to write the alphabet on the paper or board. They have to detect the surface first; after detecting the surface, they need to put the camera close to the written alphabet. They then need to tap on the mobile screen to check whether it is right or wrong. The app collects the texture from the AR camera. Our machine learning model will evaluate the textures and provide an index of the written alphabet. Then by using the ARCore engine, a 3D model will appear on the screen. This part of our app is only supported by devices that can run ARCore SDK [41].

The instructions in this app, both voice and text, are in the Bangla language by default. Users can choose between Bangla and English from the homepage by tapping on the language button (Figure 3).

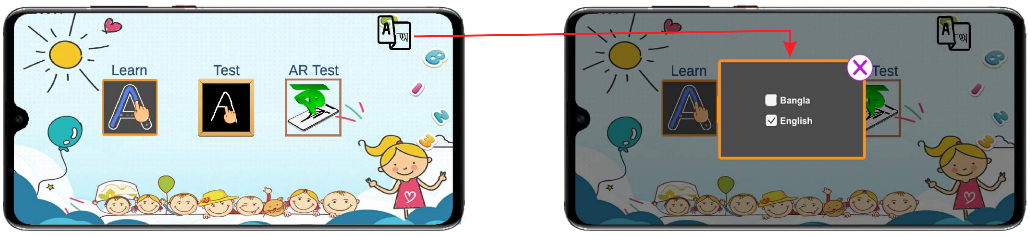

Figure 3. First screen of the app—option to change app language.

4.1. Learn Writing

The learner can switch between Bangla vowels, consonants, digits, English alphabets, and digits. By clicking the learn button on the landing page, the user enters the learning module. The first page of this module contains categories as described above and lists of characters to select. The user can select any of them to switch categories. By clicking on a character, another page opens where the user can trace over the given gray lines to learn to write the character. Figure 4 shows the complete workflow of the learning module. After completing the tracing, a pop-up message with sound effects congratulates the user for success. The user can erase and rewrite or skip.

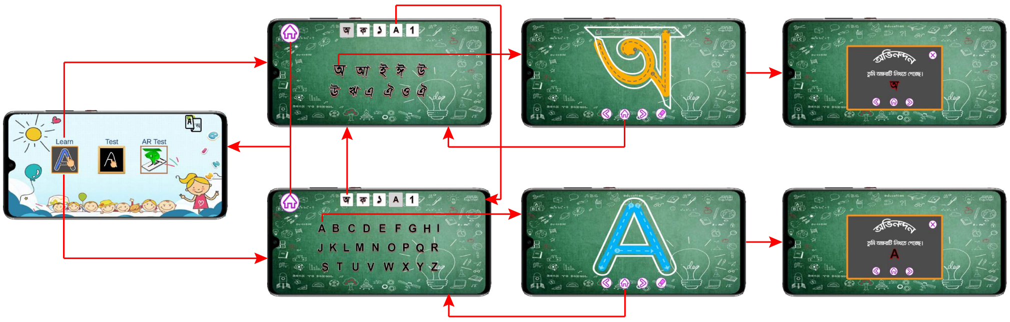

Figure 4. Screenshots of learning module workflow.

4.2. *Testing*

Both test systems in this app are based on the deep neural network. In both cases the image of the written letter is sent to the model, which evaluates it and predicts a result.

4.2.1. On-Screen Testing

By clicking the middle button of the landing page, the user enters the on-screen test module. Here, the user can switch between Bangla and English languages. As shown in Figure 5, the user is presented with a random character to write. When the user presses the check button after completing the writing, the neural network takes the image and predicts a result. The result is matched with the given character and a pop-up is shown accordingly. The user can continue for as long as they want and skip characters.

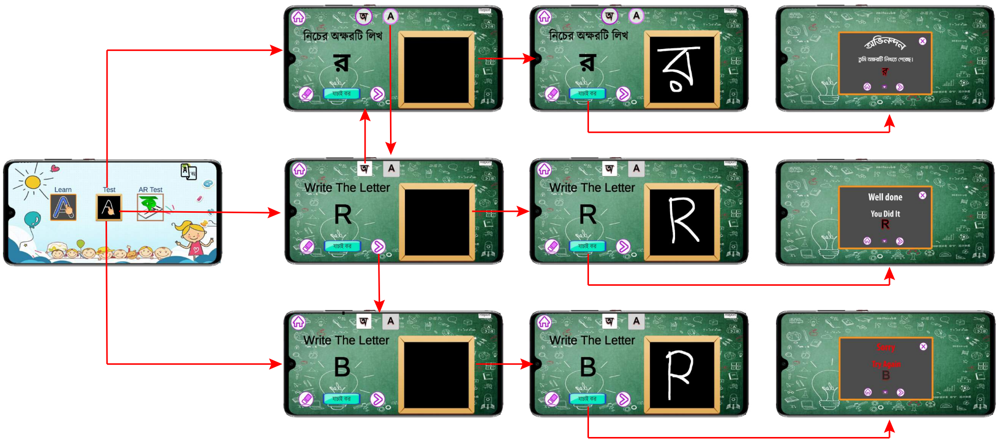

Figure 5. Screenshots of on-screen testing module.

4.2.2. AR-Based Testing

By clicking the right-side button on the landing page, the user enters the AR-based testing scene where they can test their writing and see the result in the AR environment (see Figure 6). At first, when the scene is loaded, the AR camera will be turned on and a pop-up appears on the screen to assist the user. The user must locate a surface. Any surface will work in our case, but a white surface with no texture variants will not work.

The surface is detected by the ARCore on its texture variants. After detecting a surface, the user will see some white dots on the plane, indicating how much area is detected. Then the user will see a random alphabet on the screen and, needs to write it on white paper with a black marker. After writing this alphabet, the user needs to touch the screen to evaluate the letters.

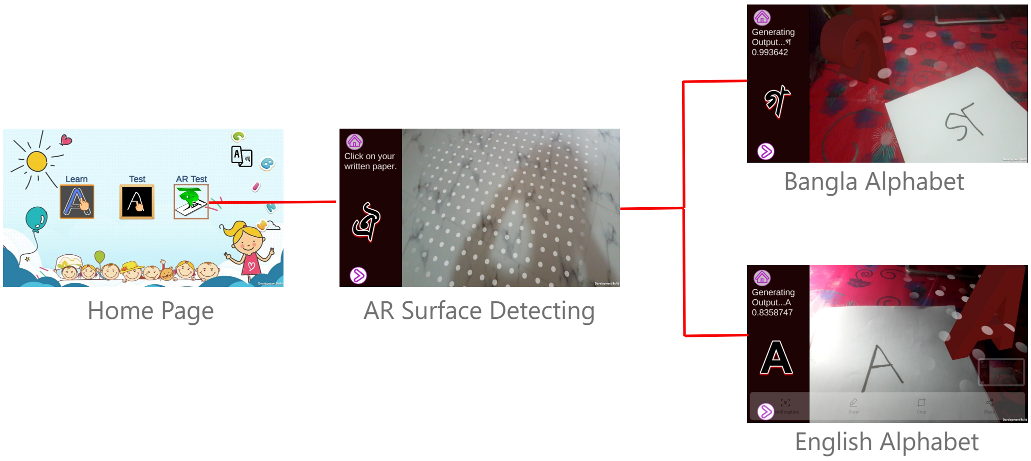

Figure 6. Screenshots of AR-based testing module.

The app takes a screenshot of a certain area from our app screen. Then the system resizes the image and sends it to the ML model, which returns the best probability of letter index. This index identifies which 3D model and letter to show on the screen. If the index of our random alphabet and the user's writing match, the app displays a 3D model of our alphabet on the screen. We instantiate the 3D model where the user touched. If our machine learning model's index and current display image index do not match, we display a pop-up dialogue informing the user that the writing is incorrect.

5. Experimental Evaluation

5.1. Model Evaluation

We experimented with and evaluated the selected deep learning models to identify the best one for our app. The whole process has three steps: dataset collection, training, and evaluation. The models are trained and tested on Bangla handwritten character datasets for model selection.

The four popular Bangla handwritten character datasets are—CMATERdb [42], BanglaLekha Isolated [43], ISI [44], and Ekush [45]. We used all four datasets to build the deep learning model and combined them into a single dataset. Table 2 describes the characteristics of the datasets.

Table 2. Bangla handwritten character datasets.

Dataset	Type	Number of Classes	Samples	Total Samples
CMATERdb	Digit Alphabet	10 50	6000 15,000	21,000
BanglaLekha—Isolated	Digit Alphabet	10 50	19,748 98,950	118,698
ISI	Digit Alphabet	10 50	23,368 37,858	61,226
Ekush	Digit Alphabet	10 50	30,785 155,570	186,355
Mixed Dataset	Digit Alphabet	10 50	79,901 307,378	387,279

We have a dataset with 387,279 images after combining all of datasets. Eighty percent of these images were used as training data, with the remaining 20% divided equally between validation and test sets.

The models for English were built using the EMNIST [46] dataset, which has six versions. We used the "ByMerge" version, which has 47 classes. ByMerge excludes the characters with the same or similar upper case and lower case symbols. This reduces the class number from 52 to 47. The total number of images in EMNIST ByMerge is 814,255.

To select a model for mobile devices, it is important to consider the model size, prediction time, and accuracy as the performance depends on all these matrices. In brief, it must have a tolerable trade-off among these factors. Table 3 shows us the comparison among the evaluated models to find the most suitable model in the case of Bangla. The evaluation was performed on a PC with a 7th generation core i5 processor with 8GB of RAM. After comparing based on accuracy, time, and model size, we chose EkushNet for our mobile app.

Table 3. Comparison between state-of-the-art deep learning models.

Model	Train Acc.	Test Acc.	Loading TIME (s)	Testing Time (s)	Model Size (MB)
BornoNet	94.70%	96.28%	1.974 ± 0.034	8.813 ± 0.179	51.6
EkushNet	96.40%	96.71%	2.801 ± 0.0307	9.235 ± 0.084	18.6
DenseNet	98.51%	96.90%	36.35 ± 0.75	98.879 ± 0.829	144.3
Xception	97.46%	96.63%	48.42 ± 1.10	58.956 ± 1.898	403.2
MobileNetV2	96.80%	96.22%	20.956 ± 0.588	26.05 ± 1.68	28.4

5.2. *App Efficiency Evaluation*

We measured app efficiency based on app performance and resource consumption. App performance shows how fast an app responds. Resource consumption shows the CPU and memory usage. The purpose of this measurement is not to generalize the performance of our app through an extensive experiment, but rather to provide an indication whether the performance is acceptable.

The app performance of the writing testing module is tested based on two matrices—loading time and execution time. The time required for loading the machine learning model in the memory is called the loading time. It is done only once after launching the app. The time required for each test is called the execution time. This is the time between the input of data in the model and the output being retrieved. Twenty samples of each of the two categories were taken. Table 4 shows the loading and execution times.

Table 4. App performance results.

Type	Min Time (ms)	Max Time (ms)	Avg. Time (ms)
Loading Time	457.56	779.32	678.67 ± 70.27
Execution Time	5.69	12.53	7.09 ± 1.68

The application is also tested on real Android devices to analyze its performance, including CPU and memory (RAM) usage. The app is used for 10 minutes and data is taken at 10 s intervals. Table 5 shows the CPU usage in percentage and memory usage in megabytes. The major portions of the resources are consumed by AR components for rendering and visualizing 3D models.

Table 5. App resource consumption results.

Device	Avg. CPU Usage (%)	Avg. Memory Usage (MB)
Xiaomi Redmi Note 7 Pro	9.965	349.64
Samsung A71	13.343	398.45

6. Conclusions and Future Work

Using AR in educational apps motivates learners to learn very quickly. It creates an interest in learning among the learners. It is strongly suggested that more augmented reality educational apps be developed to motivate and engage learners in learning alphabet and digits. To advance the technology used in the current apps, in this paper, we have integrated a machine learning model to AR so that the learner's handwriting alphabet on paper or a board can be evaluated and at the same time it can be used as a marker to show the corresponding 3D model. We have used four popular handwritten character datasets to train and evaluate state-of-the-art deep learning models to identify the best one for our app. We have further measured the efficiency of our app in terms of performance and resource consumption.

In the future, we plan to extend our app by incorporating more features, including detecting and correcting handwriting errors such as incorrect stroke production, sequence, and relationship. It is important to note that the success of an educational AR application depends not only on the provided features and technological advancement but also on the pedagogical characteristics of the context in which the app is used. Moreover, studies also reveal that establishing an appropriate learning environment is crucial when AR is used. Therefore, it is not enough to have the educational apps or devices, but an adaptation of these apps to the environment in which it will be applied is needed [47]. A users study is required to understand learners experience and expectation from these apps. Thus, we have plan to conduct a users study to evaluate the usability of our app, and, in particular, the handwriting evaluation feature.

Author Contributions: Conceptualization, M.N.I.O., M.R.I., M.A.K., M.S.H. and M.M.I.; methodology, M.N.I.O., M.R.I. and M.A.K.; software, M.N.I.O. and M.R.I; validation, M.N.I.O., M.R.I. and M.A.K.; formal analysis, M.N.I.O., M.R.I. and M.A.K.; resources, M.M.I.; writing—original draft preparation, M.N.I.O., M.R.I. and M.A.K.; writing—review and editing, M.A.K., M.S.H. and M.M.I.; visualization, M.N.I.O., M.R.I. and M.A.K.; supervision, M.A.K.; project administration, M.S.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AR	Augmented Reality
CNN	Convolution Neural Network
ML	Machine Learning

References

1. Odell, L. The process of writing and the process of learning. *Coll. Compos. Commun.* **1980**, *31*, 42–50. [CrossRef]
2. Muttappallymyalil, J.; Mendis, S.; John, L.J.; Shanthakumari, N.; Sreedharan, J.; Shaikh, R.B. Evolution of technology in teaching: Blackboard and beyond in Medical Education. *Nepal J. Epidemiol.* **2016**, *6*, 588–592. [CrossRef] [PubMed]
3. Wellner, P. Interacting with Paper on the DigitalDesk. *Commun. ACM* **1993**, *36*, 87–96. [CrossRef]
4. Alvaro, A.K.S.; Cruz, R.L.D.D.; Fonseca, D.M.T.; Samonte, M.J.C. Basic handwriting instructor for kids using OCR as an Evaluator. In Proceedings of the 2010 International Conference on Networking and Information Technology, Manila, Philippines, 11–12 June 2010; pp. 265–268.
5. Yanikoglu, B.; Gogus, A.; Inal, E. Use of handwriting recognition technologies in tablet-based learning modules for first grade education. *Educ. Technol. Res. Dev.* **2017**, *65*, 1369–1388. [CrossRef]
6. Wu, H.K.; Lee, S.W.Y.; Chang, H.Y.; Liang, J.C. Current status, opportunities and challenges of augmented reality in education. *Comput. Educ.* **2013**, *62*, 41–49. [CrossRef]

7. Elmqaddem, N. Augmented Reality and Virtual Reality in education. Myth or reality? *Int. J. Emerg. Technol. Learn.* **2019**, *14*, 234–242. [CrossRef]
8. Abrar, M.F.; Islam, M.R.; Hossain, M.S.; Islam, M.M.; Kabir, M.A. Augmented reality in education: A study on preschool children, parents, and teachers in Bangladesh. In *International Conference on Human-Computer Interaction*; Springer: Cham, Switzerland, 2019; pp. 217–229.
9. Sumadio, D.D.; Rambli, D.R.A. Preliminary evaluation on user acceptance of the augmented reality use for education. In Proceedings of the 2010 2nd International Conference on Computer Engineering and Applications, ICCEA 2010, Bali Island, Indonesia, 19–21 March 2010; Volume 2, pp. 461–465. [CrossRef]
10. Kesim, M.; Ozarslan, Y. Augmented Reality in Education: Current Technologies and the Potential for Education. *Procedia—Soc. Behav. Sci.* **2012**, *47*, 297–302. [CrossRef]
11. Garzón, J. An Overview of Twenty-Five Years of Augmented Reality in Education. *Multimodal Technol. Interact.* **2021**, *5*, 37. [CrossRef]
12. Aydoğdu, F. Augmented reality for preschool children: An experience with educational contents. *Br. J. Educ. Technol.* **2021**, 23. [CrossRef]
13. Madanipour, P.; Cohrssen, C. Augmented reality as a form of digital technology in early childhood education. *Australas. J. Early Child.* **2020**, *45*, 5–13. [CrossRef]
14. Hape, K.; Flood, N.; McArthur, K.; Sidara, C.; Stephens, C.; Welsh, K. A pilot study of the effectiveness of the Handwriting Without Tears® curriculum in first grade. *J. Occup. Ther. Sch. Early Interv.* **2014**, *7*, 284–293. [CrossRef]
15. RV AppStudios ABC Kids. 2016. Available online: http://www.rvappstudios.com (accessed on 25 November 2021).
16. ShurjoMukhi Ltd Hatekhori. 2020 Available online: https://shurjomukhi.com.bd (accessed on 25 November 2021).
17. Plympton Labs Narrator AR. 2019. Available online: https://www.narratorar.com.au (accessed on 25 November 2021).
18. Sybenga, S.; Rybarczyk, Y. Using machine learning and image processing for character recognition: An application for teaching handwriting. In Proceedings of the 28th International Conference on Computer Applications in Industry and Engineering, San Diego, CA, USA, 12–14 October 2015.
19. Al-Barhamtoshy, H.; Abdou, S.; Al-Wajih, F.A. A Toolkit for Teaching Arabic Handwriting. *Int. J. Comput. Appl.* **2012**, *975*, 8887. [CrossRef]
20. Park, H.; Park, J.I. Invisible Marker–Based Augmented Reality. *Int. J. Hum.—Comput. Interact.* **2010**, *26*, 829–848. [CrossRef]
21. Rabby, A.S.A.; Haque, S.; Islam, S.; Abujar, S.; Hossain, S.A. Bornonet: Bangla handwritten characters recognition using convolutional neural network. *Procedia Comput. Sci.* **2018**, *143*, 528–535. [CrossRef]
22. Rabby, A.S.A.; Haque, S.; Abujar, S.; Hossain, S.A. Ekushnet: Using convolutional neural network for bangla handwritten recognition. *Procedia Comput. Sci.* **2018**, *143*, 603–610. [CrossRef]
23. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
24. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
25. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
26. DevPocket. iTrace. 2013. Available online: https://itraceapp.com (accessed on 25 November 2021).
27. L'Escapadou. Writing Wizard. 2013. Available online: https://lescapadou.com (accessed on 25 November 2021).
28. TopOfStack Software Ltd Kids Learn Bangla Alphabet. 2020. Available online: https://topofstacksoftware.com (accessed on 25 November 2021).
29. Aizan, N.L.K.; Mansor, E.I.; Mahmod, R. Preschool children handwriting evaluation on paper-based and tablet-based settings. *Children* **2014**, *17*, 18.
30. Maxim, B.R.; Patel, N.V.; Martineau, N.D.; Schwartz, M. Work in progress-learning via gaming: An immersive environment for teaching kids handwriting. In Proceedings of the 2007 37th Annual Frontiers In Education Conference-Global Engineering: Knowledge without Borders, Opportunities Without Passports, Milwaukee, WI, USA, 10–13 October 2007; p. T1B-3.
31. El-Sawy, A.; Loey, M.; Hazem, E.B. Arab Kids Tutor (AKT) System For Handwriting Stroke Errors Detection. *Int. J. Technol. Enhanc. Emerg. Eng. Res.* **2016**, *4*, 42–49.
32. Iwayama, N.; Akiyama, K.; Tanaka, H.; Tamura, H.; Ishigaki, K. Handwriting-based learning materials on a tablet pc: A prototype and its practical studies in an elementary school. In Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition, Kokubunji, Japan, 26–29 October 2004; pp. 533–538.
33. Jordan, G.; Michaud, F.; Kaiser, M.L. Effectiveness of an intensive handwriting program for first grade students using the application LetterSchool: A pilot study. *J. Occup. Ther. Sch. Early Interv.* **2016**, *9*, 176–184. [CrossRef]
34. Hu, Z.; Xu, Y.; Huang, L.; Leung, H. A Chinese handwriting education system with automatic error detection. *JSW* **2009**, *4*, 101–107. [CrossRef]
35. Business Empower Australia Pty Ltd Bangla Alphabet. 2018. Available online: https://www.businessempoweraustralia.com (accessed on 25 October 2021).

36. Creative Apps BD Kids Alphabet Learning Made Easy. 2019. Available online: https://play.google.com/store/apps/details?id=com.creativeapps.kids_learning_bangla (accessed on 12 August 2021).
37. Barracuda Library for Unity. Available online: https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/manual/index.html (accessed on 21 November 2021).
38. AR Core SDK Developed by Google. Available online: https://github.com/google-ar/arcore-unity-sdk (accessed on 25 October 2021).
39. Oufqir, Z.; El Abderrahmani, A.; Satori, K. ARKit and ARCore in serve to augmented reality. In Proceedings of the 2020 International Conference on Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 9–11 June 2020; pp. 1–7.
40. Maya 3D Modeling Software Developed by Autodesk. Available online: https://www.autodesk.com/products/maya/overview (accessed on 25 October 2021).
41. AR Core Supported Device List. Available online: https://developers.google.com/ar/devices (accessed on 25 October 2021).
42. Sarkar, R.; Das, N.; Basu, S.; Kundu, M.; Nasipuri, M.; Basu, D.K. CMATERdb1: A database of unconstrained handwritten Bangla and Bangla–English mixed script document image. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2012**, *15*, 71–83. [CrossRef]
43. Biswas, M.; Islam, R.; Shom, G.K.; Shopon, M.; Mohammed, N.; Momen, S.; Abedin, A. Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters. *Data Brief* **2017**, *12*, 103–107. [CrossRef] [PubMed]
44. Bhattacharya, U.; Chaudhuri, B.B. Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 444–457. [CrossRef] [PubMed]
45. Rabby, A.S.A.; Haque, S.; Islam, M.S.; Abujar, S.; Hossain, S.A. Ekush: A Multipurpose and Multitype Comprehensive Database for Online Off-Line Bangla Handwritten Characters. In *International Conference on Recent Trends in Image Processing and Pattern Recognition*; Springer: Cham, Switzerland, 2018; pp. 149–158.
46. Cohen, G.; Afshar, S.; Tapson, J.; Van Schaik, A. EMNIST: Extending MNIST to handwritten letters. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 2921–2926.
47. López-Belmonte, J.; Moreno-Guerrero, A.J.; López-Núñez, J.A.; Hinojo-Lucena, F.J. Augmented reality in education. A scientific mapping in Web of Science. *Interact. Learn. Environ.* **2020**, 1–15. [CrossRef]