A Multilingual Handwritten Character Dataset: T-H-E Dataset

Gaye Ediboğlu Bartos[1], Yaşar Hoşcan[1], András Kauer[2], Éva Hajnal[3]

[1]Eskisehir Technical University, Department of Computer Engineering, 2 Eylül Campus, 26555 Eskisehir, Turkey, gayeediboglu@eskisehir.edu.tr, hoscan@eskisehir.edu.tr

[2] Székesfehérvári SzC Széchenyi István Secondary Technical School, Budai út 45, 8000 Székesfehérvár, Hungary, kauer.andras@gr-szechenyi.hu

[3]Óbuda University Alba Regia Technical Faculty, Budai út 45, 8000 Székesfehérvár, Hungary, hajnal.eva@amk.uni-obuda.hu

Abstract: The absence of handwritten special Latin character datasets prompted the creation of the T-H-E Dataset (Turkish-Hungarian-English handwritten character dataset) contributing to the recognition of multilingual handwritten texts. This paper represents a public-domain dataset including handwritten Turkish, Hungarian and English characters collected from 200 participants. The T-H-E Dataset is formed from 78 different letters represented in 156000 binary characters including both the upper and lower-case versions. The dataset can be downloaded from the web in six different versions enabling users to combine the different alphabets for different recognition purposes. The evaluation of the dataset is carried out by applying the same deep learning architecture on the T-H-E dataset and the EMNIST dataset. The dataset is publicly available at https://github.com/bartosgaye/thedataset.

Keywords: public dataset; handwritten character dataset; offline character recognition; OCR, multilingual

1 Introduction

Handwritten text datasets can be found in several forms, such as, full-page handwritten images, handwritten sentences, handwritten words and handwritten individual characters. However, the majority of the available datasets focus on a single language ignoring the existence of multilingual texts. As a result of globalization, multilingual handwritten texts are increasingly generated. By raising the number of multilingual datasets, the success on the single language handwriting recognition could similarly be achieved for multilingual handwritten texts. It is worth mentioning that offline handwriting recognition, of a single language, remains an unresolved problem, since there is no standard form in handwriting, unlike in print documents. Despite the available datasets for English characters, recognition of offline handwriting remains a challenge for several languages, such as, Turkish and Hungarian. In the case of Turkish, some researchers used datasets of their own which are not publicly available [1]–[5]. In order to be able to develope algorithms which deliver solid performance on handwritings with puncuations, handwritten character datasets on alphabets including a high number of punctuations are needed. In this paper we present a freely available character dataset consisting of 78 classes (Table 1) referring to 52 classes for English characters (26 upper-case+ 26 lower-case), 8 classes for special Turkish characters (4 upper-case+ 4 lower-case), 4 classes for Turkish and Hungarian joint characters (2 upper-case+ 2 lower-case) and 14 classes for special Hungarian characters (7 upper-case+ 7 lower-case) [5]. The two main reasons behind creating such dataset are lack of offline datasets for recognition of languages with special characters such as Turkish and Hungarian and secondly contributing to the existing Latin character datasets with a variety of handwritings collected from Hungarian and Turkish citizens. In addition to those motives, the proposed dataset offers an easier platform for designing multilingual unified recognition systems.

Handwritings collected from merely Turkish citizens mostly contain texts which are written using discrete characters only whereas texts written by Hungarians mainly consist of cursive characters. Gathering handwritings from both nations give the diversity to the dataset and such feature is believed to provide a positive impact on the classification process. In the next section, the earlier offline handwritten English character datasets and multilingual recognition systems are going to be represented.

2 Related Works

In machine learning, having access to right data in right format allows the researchers to develop, advance and assess their learning techniques. Therefore, regardless of the language of the handwriting, the condition and amount of the input data plays a crucial part in the performance of any handwriting recognition system. In this paper, we present a digitized, preprocessed, and labeled image dataset which consists of handwritten letters from three different languages. In the literature, handwritten character datasets can be found for several languages however, when it comes to multilingual handwritten character datasets, not many can be found. In order to be able to establish a multilingual recognition system, researchers either merge single language character/word datasets or adopt existing multilingual word datasets. The examples of multilingual handwriting recognition systems are presented in the following section. The majority of the studies focus on the recognition of English and French, due to the fact that there are existing datasets for those languages. In 2012, Wshah et al. used the IAM dataset [6] for English, the AMA dataset for Arabic [7] and the LAW dataset for Devanagari [8] together with a synthetic dataset in order to evaluate the proposed multilingual word spotting system [9]. Kozielski et al. carried out a study on recognizing real-world handwritten images in English and French in 2014 [10]. IAM, RIMES and Maurdor datasets [11] were used to train and evaluate their multilingual system. Bluche and Messina proposed a multilingual handwriting recognition system which was trained on datasets in English, French, Spanish, Portuguese, German, Italian and Russian in 2017 [12]. They used IAM, RIMES [13] and Maurdor datasets alongside with private collections they collected for those languages without available public or private datasets to evaluate their model. Lately in 2019, Swaileh et al. proposed a unified multilingual handwriting recognition system which was trained and evaluated using IAM and RIMES datasets for English and French respectively [14].

The abovementioned studies are all carried out on a combination of word or document based datasets in different languages. The following section puts forward the most popular offline English handwritten character datasets. One of the earliest handwritten Latin character dataset, the CEDAR dataset, dates back to 1994, it consists of both handwritten words, such as, city names and postal codes and characters containing separated letters and numbers [15]. The separated letters and characters were put into 62 classes (26 upper-case+ 26 lower-case+ 10 digits) consisting of approximately 50000 samples. A year later, in 1995, the NIST Special Dataset 19 Hand printed Forms and character dataset was published containing full page binary image of handwritten forms and also characters (digits and letters) segmented from those forms (128x128). In the NIST dataset there are 62 labelled classes for digits ‘0-9’, characters ‘a-z’ and ‘A-Z’. Later in 1998, MNIST (Modified-NIST) dataset was created containing only digits (70000 samples) and it became a benchmark for digit recognition purposes [16]. In 2016, the 2nd version of the NIST dataset was published with full page binary images of 3699 handwritten sample forms and 814255 sample digits and characters of the same 62 classes [17]. It is possible to say that NIST dataset has become a benchmark for character recognition problem. Finally in 2017, EMNIST dataset, an extension of the MNIST dataset was published [18], [19]. EMNIST dataset is superior to its previous versions by many features such as number of instances, the balanced representation of characters, grayscale representation and the variety of classes provided. It contains 814255 samples of letters and digits (28x28). In addition to NIST and MNIST, EMNIST not only provides two class hierarchies namely By Class (every character into a different class with a different label) and By Merge (similar characters into the same class with the same label) but also provide four more options namely: balanced dataset which is easy to apply due to its balanced subset of all the By Merge classes; letters dataset generated to increase the number of errors occurring from case confusion by merging all of the uppercase and lowercase classes, to form a balanced 26-class classification task; digits dataset being a balanced subset of the digits dataset containing 28000 samples of each digit and a copy of MNIST dataset. Fig. 1 below shows the distribution of the different letters in the EMNIST By Class dataset.

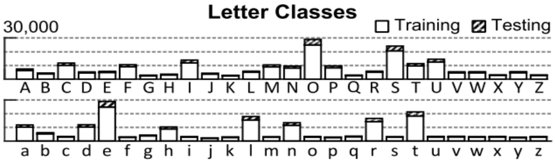

Figure 1

Representation of the letters in the EMNIST By Class dataset [18]

Finally in 2006, distinctly from previous datasets, a cursive character dataset C-Cube (Cursive Character Challenge) came out [20]. The C-Cube dataset includes 57293 characters including 26 upper and 26 lower case versions of each Latin letter. In our previous works, we adopted C-Cube data set after changing the way data was represented in the original dataset [21].

3 T-H-E Dataset

The T-H-E Dataset includes handwritten letters from multiple alphabets, namely from English (ISO Basic Latin Alphabet), Turkish and Hungarian. However, since the dataset includes many Latin characters, it is very easy for other researchers to modify the data set for their needs (add/ remove special characters) and use it as a whole. The characters included in the dataset are presented in Table 1 below.

Table 1

Characters in the T-H-E Dataset

	Lower case	Number of instances	Upper case	Number of instances
English Characters	a	2000	A	2000
	b	2000	B	2000
	c	2000	C	2000
	d	2000	D	2000
	e	2000	E	2000
	f	2000	F	2000
	g	2000	G	2000
	h	2000	H	2000
	i	2000	I	2000

	j	2000	J	2000
	k	2000	K	2000
	l	2000	L	2000
	m	2000	M	2000
	n	2000	N	2000
	o	2000	O	2000
	p	2000	P	2000
	q	2000	Q	2000
	r	2000	R	2000
	s	2000	S	2000
	t	2000	T	2000
	u	2000	U	2000
	v	2000	V	2000
	w	2000	W	2000
	x	2000	X	2000
	y	2000	Y	2000
	z	2000	Z	2000
Turkish Special Characters	ç	2000	Ç	2000
	ğ	2000	Ğ	2000
	ı	2000	İ	2000
	ş	2000	Ş	2000
Turkish and Hungarian Joint Characters	ö	2000	Ö	2000
	ü	2000	Ü	2000
Hungarian Special Characters	á	2000	Á	2000
	é	2000	É	2000
	í	2000	Í	2000
	ó	2000	Ó	2000
	ő	2000	Ő	2000
	ú	2000	Ú	2000
	ű	2000	Ű	2000
Total Number of Characters	**39**	**78000**	**39**	**78000**

In order to generate the dataset, handwriting samples were collected, in an ethical way, from 200 participants who predominantly were at that time, high school and university students (Turkish and Hungarian citizens mixed), in a controlled environment. The participants were given a blank white paper and were asked to write the given text in their native language in their own handwriting. It can be said that there was less noise found in the images, since the paper used was new and blank. Then, the papers were scanned at 300 DPI. Subsequently, the images were thickened using morphological thickening provided by the MATLAB 9.3 environment [22] and line, word and character segmentation was performed [23]. These steps usually include a noise removal step, in order to get rid of the noise occurring in the scanned documents. However, the noise removal step was skipped in order to maintain every accent and punctuation in the images. The character segmentation phase includes several processes, namely, separating each character, getting rid of the white space around each character and binarization of the character, using Otsu's Algorithm [24]. Finally, every character is normalized to a 28x28 pixel shape. A representation of the sample characters, after the normalization step, can be found in Fig. 2.

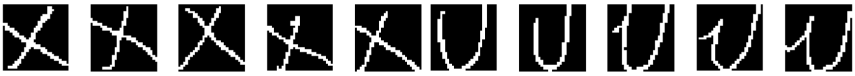

Figure 2
Sample Characters from the T-H-E Dataset

3.1. Structure of the Dataset

Including characters from several alphabets, the T-H-E dataset is established in six versions, to provide for ease of use and flexibility when switching between alphabets, for different researchers with different approaches. The abovementioned six versions are explained below:

entire_augmented: This version represents the entire dataset. It includes all the 28x28 pixel binary characters from the three alphabets together forming a balanced dataset with 156000 characters belonging to 78 classes (Table 1).

tr_augmented: It consists of merely 12 Turkish special characters (6 upper-case and 6 lower-case). 2000 samples of each character can be found in this version forming a 24000-character dataset.

hu_augmented: Similar to the tr_augmented version, this includes 18 Hungarian special characters only (9 lower-case and 9 upper-case) forming a 36000-character dataset.

en_augmented: The fourth version includes 2000 samples of 52 English characters (26 upper-case and 26 lower-case) forming a 104000-character dataset.

This representation enables us to merge English letters with Hungarian special characters and work only on Hungarian characters by just putting two versions together. A fair warning should be provided about the Turkish alphabet; putting tr_augmented and en_augmented together does not result in the Turkish alphabet since there are no letters 'q', 'w' and 'x' in the Turkish alphabet. The users may want to exclude those 3 letters (3 lower-case and 3-upper-case) from the en_augmented in order to work on Turkish alphabet accurately.

merged_augmented: This version is derived from the entire_augmented version which includes all the characters from different alphabets together. The characters having a similar way of representation in their upper-case and lower-case form are put into the same class in this version such as lower case ‘o’ and upper case ‘O’. The characters merged are shown in the Table 2 below. In this group there are 55 classes and 156000 samples. However, only in this version are the number of instances, in each class, not balanced. Some classes have 2000 samples, while merged ones, are represented in 4000 samples.

Table 2
Merged Characters

	Merged Classes	Number of Instances		Merged Classes	Number of Instances
1	c- C	4000	13	s-S	4000
2	i-I	4000	14	ş-Ş	4000
3	í- Í	4000	15	u-U	4000
4	ı-İ	4000	16	ú -Ú	4000
5	j-J	4000	17	Ü-Ü	4000
6	k-K	4000	18	ű- Ű	4000
7	m-M	4000	19	v-V	4000
8	o-O	4000	20	w-W	4000
9	ó- Ó	4000	21	x-X	4000
10	Ö-Ö	4000	22	y-Y	4000
11	ő- Ő	4000	23	z-Z	4000
12	p-P	4000			

entire_raw: The original handwritten characters (1000 instances for every 78 classes) are put forward in the sixth version. Using this version, it is possible to experiment different distortion techniques and their impact on the classification performance can be tested. 78000 characters from 78 different classes, can be found in this version.

One important point to be noted is that there are 4 special characters (ü, ö, Ü and Ö) which are used both in Turkish and Hungarian, therefore, they repeat in tr_augmented and hu_augmented versions. Another crucial point was discovered during the handwriting collection process concerning those 4 joint characters. In the Hungarian alphabet, there are two special characters ‘ő’ and ‘ö’ which are apparently represented in one single character ‘ö’ in Turkish alphabet (Similarly, letter ‘ü’ and ‘ű’ are presented as ‘ü’). The shape of the accent over the letter does not make a difference in the Turkish alphabet (based on the handwritings collected), however, they represent two different characters in the Hungarian alphabet. Therefore, it is crucial to understand the differences before carrying out the recognition. In order to avoid confusion, in this dataset, the Turkish and Hungarian joint characters, ‘ö’ and ‘ü’, were carefully segmented by adding only short slanted versions into these classes, by avoiding some of the Turkish participants’ handwritings. Users might want to merge the classes ‘ö’ and ‘ő’ into one single class, if they are training for a Turkish recognition, instead of just discarding the letter ‘ő’ (the same applies for ‘ü’ and ‘ű’).

3.2. Data Augmentation

Augmenting the input image by applying distortions in order to increase the variance and therefore, performance, is a very common use both in character and text recognition [25]–[27]. Examples of different distortions such as shifting, scaling, skewing, and compression is represented in the popular MNIST dataset.

As represented in the previous section, the T-H-E dataset contains 2000 samples of every character. However, this number includes 1000 original handwritten characters and 1000 generated characters from those 1000 original characters. The number of handwritten characters was increased, by augmenting the existing characters by applying distortions on the original characters.

The augmentations include affine transformations and elastic distortions. Every single handwritten character is distorted randomly once using one of the distortions. If it is desired to have an even larger dataset, the same random distortion algorithm can be run on the original set time after time, generating 78000 randomly distorted character images at every attempt (the source code used for randomly generating images can be downloaded together with the dataset). The distortion methods applied on two different characters can be seen in the Fig. 3 below.

Tilting randomly to the left or right using Piecewise Linear Transformation: tilting right (50% chance) refers to moving the top left corner to the right randomly by 7 to 14 pixels and lowering it randomly by 5 to 12 pixels; tilting left (50% chance) refers to moving the bottom right corner to the left and top randomly by 1.1 to 1.5 times 28. After the tilt operation image is resized to the 28x28 pixel keeping the aspect ratio [28].

Adding Fisheye Effect: This effect was given by applying either barrel (50% chance) or pincushion effect (50% chance) randomly to the original image with a random distortion amount between 0.1 and 0.9 [29].

Rotating: Rotation of the images randomly to the left (50% chance) or right (50% chance) is applied by MATLAB [22]. Rotating to the left and right refers to randomly rotating the input by 5,10,15,20 or 25 degrees then resizing the result to fit the 28x28 matrix.

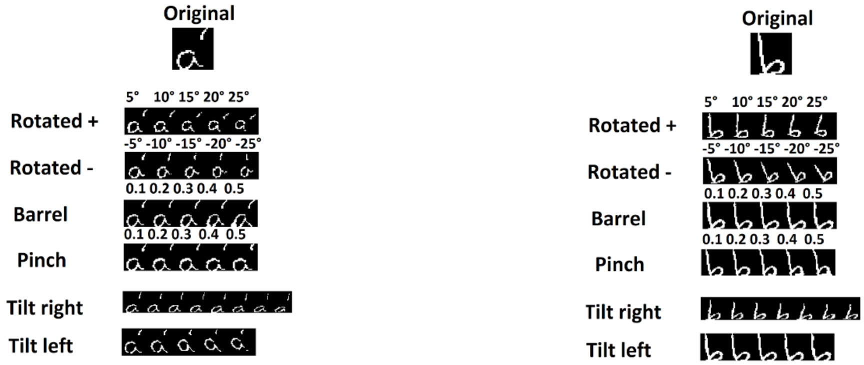

Figure 3

Adopted Distortion Methods Applied to Two Different Characters

3.3. Evaluation of the T-H-E Dataset

Deep learning is subtopic of machine learning that is capable of performing both supervised and unsupervised learning, using a feature, similar to the human brain, which is the ability to grasp patterns and recognize things accordingly [30]. Recent studies propose that deep learning algorithms outperform the traditional machine learning algorithms in the case of image classification since they do not deal with handcrafted features as can be seen in the Fig. 4 [31]–[35].

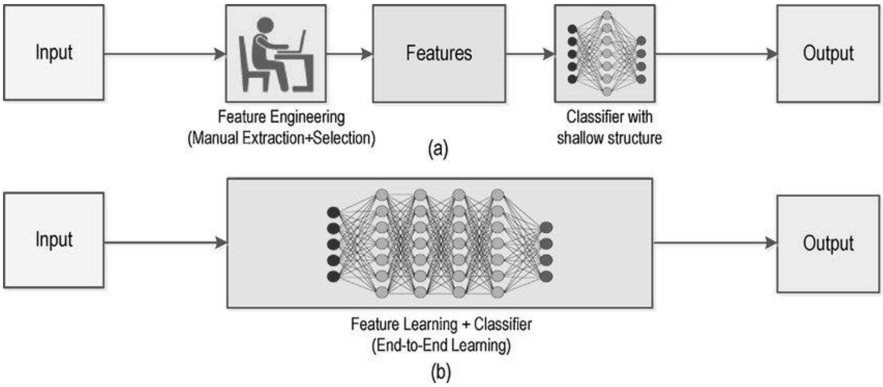

Figure 4

(a) Traditional Machine Learning Workflow vs. (b) Deep Learning Workflow [32]

Deep learning is made of multiple processing layers in order to learn representations of data with multiple levels of abstraction [36]. It is based on a hierarchically layered system, in which, each layer of nodes, is responsible for extracting distinct features using the previous layers' output. The further you go with the layers; the more advanced features can be extracted. In this study, a deep learning algorithm called Convolutional Neural Networks (CNN) is going to be adopted in order to evaluate the T-H-E dataset [37], [38].

A CNN architecture consists of an input layer, an output layer and hidden layers. An input can be a 1D signal, 2D image or 3D video. Thereafter, the input goes through a serious of hierarchical layers including convolutional layers, pooling layers in order to extract distinct features in the input. Finally, extracted features form the input layer of a Fully Connected MLP at the very end of the architecture for recognition. A brief CNN architecture can be seen in Fig. 5 below.

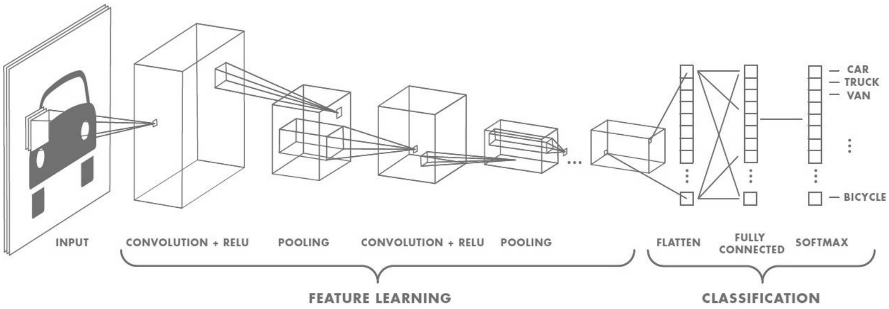

Figure 5

Convolutional Neural Network Architecture [39]

Classic CNN architectures include popular models such as LeNet-5 [16], AlexNet [40], GoogLeNet [41] and VGG [42]. Out of these models, LeNet-5 is the most suitable model for the recognition of images with small input sizes and widely adopted for the field of handwriting recognition [16].

Convolution Layer: In this layer, a convolution filter is applied to the input matrix to generate feature maps as can be seen in Fig. 6. The size of the filter is pre-determined according to the input matrix [43].

Activation Layer (ReLU): ReLU operation replaces all negative pixel values in the feature map by zero [44]. Thus, it allows faster and more effective training.

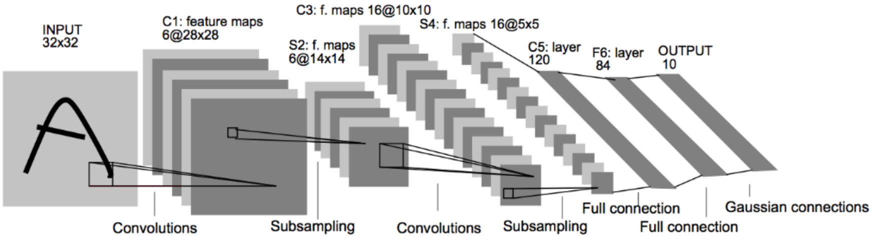

Figure 6

LeNet-5 Model [16]

Pooling Layer: Pooling layer aims at reducing the size of the feature maps for the next convolution layer generally by applying a sum, average or max filter to the feature map. However, the reduction does not necessarily result in data loss, but eliminates the least significant data resulting in easier computation in the upcoming layers [44], [45]. The operation performed by this layer is also called subsampling or downsampling.

3.3.1. Experiments

In this section two small scale experiments are represented, to confirm the validity and applicability of the proposed dataset. Additionally, third experiment compares the entire_augmented and merged_augmented datasets. As mentioned in the related works section, EMNIST dataset [18] has become a standard benchmark for handwriting character recognition purpose. Therefore, in order to evaluate the proposed dataset, the same LeNet-5 architecture was applied on both on the proposed dataset and EMNIST By_Class dataset with 20 epochs in the MATLAB 9.3 environment [22]. One important point to mention is, LeNet-5 architecture requires 32x32 pixel images as the input. For that reason, all 28x28 images were widened to 32x32 images by adding black pixels to the margins of the images (left, right, bottom and top). Another point to mention is the difference in the color of the input images in two datasets. The proposed dataset consists of characters 28x28 pixel binary images for every 72 class. However, the EMNIST By_Class dataset includes 28x28 grayscale images for 52 classes representing the lower and upper case of English Alphabet (see Fig. 1).

In terms of validation parts, a similar validation partition to the evaluation of the EMNIST dataset [18] is applied. Every class was divided into two parts namely train and test parts without using validation. The training part consists of 900 and testing part 100 characters (90% and 10% for the experiment 3).

It should be noted that the first two experiments are carried out in order to evaluate the usability of the T-H-E dataset by comparing its results with a part of the EMNIST dataset which is the benchmark in the field. Having comparable results with EMNIST dataset is the main goal of the experiments. Therefore, the performance of the recognition is not paramount.

Experiment 1

The first experiment represents the comparison of en_augmented set and EMNIST By_Class dataset under equal conditions in terms of the input size and the colors of the input images. In order to have the same sample size for both datasets, 1000 characters out of 2000 characters for each class label in en_augmented set were randomly picked (52x1000=52000). As mentioned above, in EMNIST dataset characters are represented in 28x28 grayscale images in comparison to the binary 28x28 images in the T-H-E dataset. Therefore, for the first experiment, randomly chosen 1000 characters from all 52 letter classes (26 upper case and 26 lower case) from the EMNIST By_Class dataset were binarized using Otsu's algorithm [24].

Experiment 2

The second experiment is carried out very similarly to the first one. The only difference being that the original grayscale input images from the EMNIST dataset are kept as they are.

Experiment 3

In the last experiment entire_augmented and merged_augmented datasets are compared using the LeNet-5 architecture. Although the input sizes are the same in both versions (156000), entire_augmented has 78 class labels whereas merged_augmented only has 55 class labels. The difference in the size of the output is expected to result in the favor of the merged_augmented version with smaller class labels however, it should also be noted that merged_augmented is an unbalanced set referring to the fact that not every class has the same number of instances (some have 2000 characters and others 4000). One of the previous studies conducted by the authors showed that the unbalanced nature of the dataset has a negative impact on the classification performance [21].

3.3.2. Results

This section puts forward the results of abowementioned three experiments. MATLAB 9.3 environment was used for carrying out the experiments using the LeNet-5 architecture for feature extraction and classification. In first two experiments, the input was classified into 52 classes, whereas in the third experiment, two inputs had a different number of output sizes (78 and 55). Although, 20 epochs were set for the network, the experiments stopped after the 17th epoch in all 4 cases. The Fig. 7 below is a screenshot from the results of the en_augmented version of the proposed dataset in the 1st experiment. In the image, every column represents an epoch and it can clearly be seen that the accuracy does not change significantly after the 3rd epoch.

The classification accuracies, 95% confidence intervals and highly misclassified letters, for all five inputs, in all three experiments, are shown in the Table 3 below. By looking at the results of Experiment 1 and 2, it can be seen that the portion randomly picked from the en_augmented dataset performed the best under such conditions compare to the randomly picked 52000 characters from the EMNIST By_Class dataset. Having the same input size and number of classes, the difference in the results could be explained by the fact that characters in the T-H-E dataset mainly consists of the handwritings of high school and university students. This may have brought about a more standardized way in handwriting. Although, the classification accuracy is slightly lower than 80%; as can be seen in the Table 4 below; misclassified letters are predominantly the same letters with their upper- or lower-case versions.

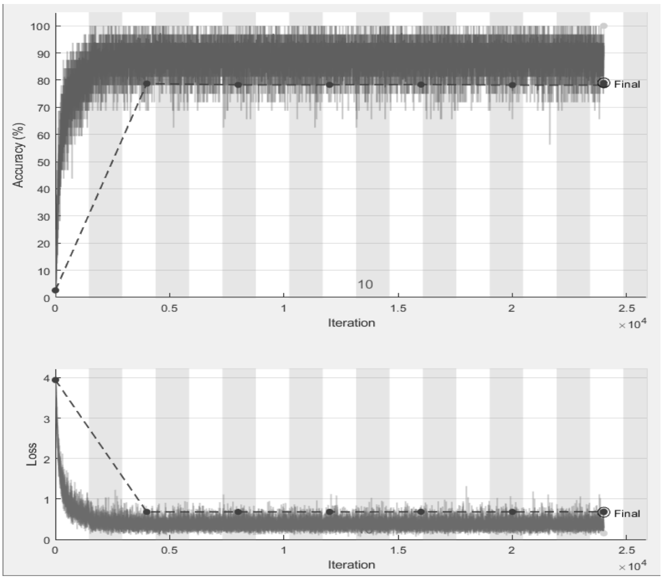

Figure 7

Classification Performance of the T-H-E Dataset

Looking at the different representations of EMNIST by_class dataset in Experiment 1 and Experiment 2, grayscale representation of the input images gave slightly better performance than the binary versions of the same images. As mentioned earlier, in this section the performance of the classifier was not crucial since the comparison was on the input not on the classifier. We believe that, adopting more sophisticated methods for classification and using a larger input set, with the participation of a more diverse group of people, rather than substantially students, could have a positive impact on the classification performance.

Table 3

Classification Performances

	Input	Input Size and #Classes	Classification Accuracy	95% Confidence Interval	Misclassified Letters
Exp.1	en_augmented	52000- 52	79.12%	1.10%	y-Y, z-Z, x-X
	EMNIST binary	52000- 52	74.77%	1.18%	p-P, t-T, J-m
Exp. 2	EMNIST grayscale	52000- 52	75.58%	0.75%	p-P, t-T, J-m
Exp.3	entire_augmented	156000- 78	71.65%	0.60%	x-X, y-Y,p-P, ő-Ő

	merged_augmented	156000- 55	82.49%	0.71%	(i-I)-(í-Í), (ö-Ö)-(ő-Ő), (z-Z)-(x-X), r-(v-V)

The comparison of the 78-class entire_augmented set and 55-class merged_augmented set in the experiment 3 resulted in favor of the merged set. The overall accuracy for the entire_augmented version was recorded 71.65% whereas; the merged_augmented version had 82.49% accuracy. The performance difference in both datasets was mainly caused by the misclassification of the upper and lower-case letters. More specifically in the entire_augmented version, the letters ‘x’, ’y’, ’p’ and ‘ő’ were highly misclassified with their uppercase forms as can be seen from Table 3. However, in the merged_augmented form of the dataset, most of the misclassification was caused by inaccurately classifying similar letters such as ‘ö’ and ‘ő’. A clearer and more detailed representation of the highly misclassified letters are demonstrated in the Table 4 below.

Table 4

Detailed Description of the Highly Misclassified Letters

Input	Letter	Accuracy	Letter	Accuracy
en_augmented	c	72.8%	C	90.1%
	x	53.9%	X	48.2%
	y	25.6%	Y	5.6%
	z	68.4%	Z	62.1%
EMNIST binary	p	52.5.%	P	54.1%
	t	51.9%	T	52.3%
	m	54.7%	J	53.3%
EMNIST grayscale	p	47.8%	P	49.4%
	t	56.0%	T	52.5%
	m	58.7%	J	55.6%
entire_augmented	x	50.3%	X	50.2%
	y	53.2%	Y	67.5%
	p	31.2%	P	79.8%
	ő	41.7%	Ő	26.5%
merged_augmented	i-I	81.8%	ı- İ	73.8%
	ö-Ö	87.7%	ő-Ő	82.2%
	z-Z	89.6%	x-X	75.6%
	r	86.3%	v-V	61.9%

Looking at the results, it can be said that merging upper and lower-case characters, have a positive effect on the class performance. Additionally, application of more sophisticated classifiers might contribute to the elimination of the errors as well as increasing the size of the input images for the merged version. An interesting point is seen by looking at the results as the letter ‘Y’ only has 5.6% accuracy rate for the en_augmented input, whereas, it has 67.5% accuracy rate for entire_augmented input. Considering that both inputs are derived from the same characters, such a difference stands out. By looking deeper into the results, it can be seen that 83.3% of the letter ‘Y’s in en_augmented, are misclassified as the letter ‘y’. The only apparent explanation for such gap can be the variation in the sample size in two inputs. As can be seen in Table 3, entire_augmented has 2000 samples of each character forming a 156000-character set whereas en_augmented used in the experiment 1 has only 1000 samples of each character forming a 52000-character set. The difference in the input size of the classifier and the sample size for each character may explain the difference in the recognition performance of the letter ‘Y’. By carrying out the three experiments, the usability of the proposed dataset was evaluated in this section.

Conclusions

In this paper, a free-to-use, multilingual handwritten character dataset, compatible with different platforms and classifiers, is presented. The handwritings were collected in an ethical way, from 200 participants, representing a diverse mixture of Turkish and Hungarian citizens. The pre-processing and segmentation phases were described and in addition to those steps, the augmentation techniques used for the letters, are described herein. Finally, the evaluation of the T-H-E dataset is carried out in three different experiments. In the first two experiments, the English letters proposed in the T-H-E dataset, are compared to the EMNIST by_class dataset, which is the benchmark for English handwriting recognition. The results of the experiment 1 and 2 demonstrated that the T-H-E dataset outperformed the randomly chosen part of the EMNIST by_class dataset. This could result from the fact that the handwritings in the T-H-E dataset, may be more standardized, since the people contributing to it were mostly high school and university students or alternatively, the T-H-E dataset might include a greater variety in handwritings since it is collected from Turkish and Hungarian Citizens, thus, presenting more distinct examples for the deep learning algorithm, to learn from. Besides outperforming the other dataset, the en_augmented version, presented very few misclassifications between different letters. Having a 79.12% accuracy rate, a majority of the errors were caused by misclassifying the same letters, with their upper- and lower-case versions. This could easily be overlooked by merging the upper- and lower-case classes or at the post processing phase, of the recognition, by using a dictionary. As for the last experiment, the same LeNet-5 architecture was applied to two out of six different versions of the proposed dataset, namely, the entire_augmented and merged_augmented versions. Both versions had the same 156000-character input size, however, the output sizes differed. The version representing letters from three different alphabets separately both in upper- and lower-case classes included 78 letters whereas, the merged version had only 55 letters, merging similarly written upper- and lower-case letters into one class. Naturally, merging two classes into one, resulted in imbalance in the dataset, having 2000 samples for unmerged classes and 4000 samples in merged classes.

Although merged_augmented has an imbalanced nature, it outperformed the entire_augmented version with over a 10% difference in accuracy rates, having only a 0.71% confidence score. As mentioned for the en_augmented version above, lower- and upper-case versions of the same characters form the highest misclassifications in the experiments. Therefore, having both versions put in the same class as in the merged_augmented version eliminates such inaccuracies. Having the six different versions provided in the T-H-E dataset makes it possible to test the performance of a classifier using the entire dataset, as well as, carry out more specific tasks, such as, effects various distortions of characters, Turkish handwriting recognition and Hungarian and English mixed handwriting recognition.

Consequently, it is possible to say that the T-H-E dataset can be an alternative for earlier datasets, in terms of English character recognition, and outperforms those in terms of the variety of letters provided. In addition to being an alternative, it is the only handwritten Turkish and Hungarian handwritten character dataset in the field. The T-H-E dataset could be adopted for single language recognition purposes, namely, Turkish, Hungarian or English character recognition systems, as well as, multilingual recognition systems. We believe, creation of multilingual character datasets will contribute to advancements in recognition systems, thus to the recognition of multilingual texts. Alongside with handwriting recognition, it could be used as an input, to evaluate other supervised and unsupervised learning systems.

In the next versions of the T-H-E dataset, we will aim at increasing the number of handwritten characters, as well as, augmentation with meta-data regarding the participants, namely, the age, gender, occupation, left or right-handed, level of education and nationality of the participants. It should be noted that finding a large number and diversity in participants, for such a purpose, is a major challenge, since the collection of the handwriting should be in person. While overcoming such a challenge, it may be possible to add other special characters from different languages, such as, Portuguese and/or French. We plan to add more alphabets to widen the scope of the dataset and we, in conjunction, plan to generate a handwritten document dataset consisting of handwritten documents in Turkish, Hungarian and English.

Dataset Availability

All the data generated in this study including the paper are publicly available at https://github.com/bartosgaye/thedataset [46]. Additional data related to this paper may be requested from the authors.

References

[1] A. Çapar, K. Taşdemir, Ö. Kılıç, and M. Gökmen, “A Turkish Handprint character recognition system,” in *Computer and Information Sciences*, 2003

[2] E. Vural, H. Erdogan, O. Oflazer, and B. Yanikoglu, “An online

handwriting recognition system for Turkish," *Proc. IEEE 12th Signal Process. Commun. Appl. Conf.*, pp. 607-610, 2004

[3] M. Şekerci and R. Kandemir, "Sözlük Kullanarak Türkçe El yazısı Tanıma," *Elektr. Bilgi. Sempozyum (ELECO 2006)*, pp. 2-6, 2006

[4] M. Şekerci, "Birleşik ve Eğik Türkçe El Yazısı Tanıma Sistemi," Trakya University, 2007

[5] G. Ediboğlu Bartos and É. Hajnal, "Optical Character Recognition for Turkish - a Survey," in *Vasil Levski National Military University Annual Scientific Conference 2017*, 2017

[6] U. V. Marti and H. Bunke, "The IAM-database: An English sentence database for offline handwriting recognition," *Int. J. Doc. Anal. Recognit.*, Vol. 5, No. 1, pp. 39-46, 2003

[7] "Applied Media Analysis," *Arabic-Handwritten-1.0.*, 2007 [Online] Available: http://appliedmediaanalysis.com/Datasets.htm

[8] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, "Database Development and Recognition of Handwritten Devanagari Legal Amount Words," in *2011 International Conference on Document Analysis and Recognition*, 2011, pp. 304-308

[9] S. Wshah, G. Kumar, and V. Govindaraju, "Multilingual word spotting in offline handwritten documents," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, pp. 310-313

[10] M. Kozielski, P. Doetsch, M. Hamdani, and H. Ney, "Multilingual off-line handwriting recognition in real-world images," *Proc. - 11th IAPR Int. Work. Doc. Anal. Syst. DAS 2014*, pp. 121-125, 2014

[11] S. Brunessaux *et al.*, "The Maurdor Project: Improving Automatic Processing of Digital Documents," in *2014 11th IAPR International Workshop on Document Analysis Systems*, 2014, pp. 349-354

[12] T. Bluche and R. Messina, "Gated Convolutional Recurrent Neural Networks for Multilingual Handwriting Recognition," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017, pp. 646-651

[13] E. Augustin, J. Brodin, M. Carré, E. Geoffrois, E. Grosicki, and F. Prêteux, "RIMES Evaluation Campaign for Handwritten Mail Processing," *Work. Front. Handwrit. Recognit.*, No. 1, pp. 1-5, 2006

[14] W. Swaileh, Y. Soullard, and T. Paquet, "A Unified Multilingual Handwriting Recognition System using multigrams sub-lexical units," *Pattern Recognit. Lett.*, Vol. 121, pp. 68-76, 2019

[15] J. J. Hull, "A Database for Handwritten Text Recognition Research," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 16, No. 5, pp. 550-554, 1994

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffiner, "Gradient based learning applied to document recognition," *Proc. IEEE,* Vol. 86, No. 11, pp. 86(11):2278-2324, 1998

[17] P. J. Grother and K. K. Hanaoka, "NIST Special Database 19," pp. 1-30, 2016

[18] G. Cohen, S. Afshar, J. Tapson, and van A. Schalik, "EMNIST : an extension of MNIST to handwritten letters," 2017

[19] M. Thoma, "The HASYv2 dataset," pp. 1-8, 2017

[20] F. Camastra, M. Spinetti, and A. Vinciarelli, "Offline Cursive Character Challenge : a New Benchmark for Machine Learning and Pattern Recognition Algorithms .," *18th Int. Conf. Pattern Recognit.*, Vol. 2, pp. 913-916, 2006

[21] G. Ediboğlu Bartos, É. Hajnal, and Y. Hoşcan, "Performance Analysis of Character case-sensitive and case-insensitive Classification in Handwritten Character Recognition," in *13th International Symposium on Applied Informatics and Related Areas (AIS 2018)*, 2018

[22] The MathWorks Inc, "MATLAB and Statistics Toolbox Release 2017b." The MathWorks, Inc., Natick, Massachusetts, United States, 2017

[23] G. Ediboglu Bartos and É. Hajnal, "Pre-processing Techniques for Hungarian Handwriting Recognition," in *11th International Symposium on Applied Informatics and Related Areas (AIS 2016),* Budapest: Óbudai Egyetem, 2016, pp. 94-99

[24] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Trans. Syst. Man, Cybern.*, Vol. 9, pp. 62-66, 1979

[25] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis," in *Seventh International Conference on Document Analysis and Recognition (ICDAR 2003),* 2003

[26] C. Wigington, S. Stewart, B. L. Davis, W. A. Barrett, B. L. Price, and S. Cohen, "Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017, pp. 639-645

[27] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, "Improving CNN-RNN hybrid networks for handwriting recognition," *Proc. Int. Conf. Front. Handwrit. Recognition, ICFHR*, Vol. 2018-Augus, pp. 80-85, 2018

[28] MathWorks, "Create a Gallery of Transformed Images," 2019 [Online] Available: https://www.mathworks.com/help/images/creating-a-gallery-of-transformed-images.html#GalleryTransformedImagesExample-7 [Accessed: 05-Jul-2019]

[29] de J. Vries, “barrel and pincushion lens distortion correction,” *MathWorks*, 2012 [Online] Available: https://www.mathworks.com/matlabcentral/fileexchange/37980-barrel-and-pincushion-lens-distortion-correction [Accessed: 05-Jul-2019]

[30] H. Wehle, “Machine Learning, Deep Learning, and AI: What’s the Difference?,” in *Data Scientist Innovation Day*, 2017, no. July

[31] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep Learning for Computer Vision: A Brief Review,” *Comput. Intell. Neurosci.*, Vol. 2018, pp. 1-13, 2018

[32] N. O’Mahony *et al.*, “Deep Learning vs. Traditional Computer Vision,” *Adv. Intell. Syst. Comput.*, Vol. 943, pp. 128-144, 2019

[33] L. Deng, “The MNIST Database of Handwritten Digit Images for Machine Learning Research,” *IEEE SIGNAL PROCESSING MAGAZINE*, No. November, pp. 141-142, 2012

[34] Y. Lecun *et al.*, “Comparison of Learning Algorithms for Handwritten Digit Recognition,” *Int. Conf. Artif. Neural Networks*, pp. 53-60, 1995

[35] Y. LeCun *et al.*, “Learning Algorithms For Classification: A Comparison On Handwritten Digit Recognition,” *Neural Networks Stat. Mech. Perspect.*, pp. 261-276, 1995

[36] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nat. Methods*, Vol. 13, No. 1, p. 35, 2015

[37] K. Fukushima, “Neural network model for a mechanism of pattern recognition unaffected by shift in position Neocognitron.,” *Trans. IECE*, Vol. 62, No. 10, pp. 658-665, 1979

[38] Y. LeCun *et al.*, “Backpropagation Applied to Handwritten Zip Code Recognition,” *Neural Comput.*, Vol. 1, pp. 541-551, 1989

[39] MathWorks, “Convolutional Neural Network 3 things you need to know.” [Online] Available: https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html?s_tid=srchtitle [Accessed: 04-Jun-2019]

[40] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Adv. Neural Inf. Process. Syst.*, Vol. 1, No. 4, 2012

[41] C. Szegedy *et al.*, “Going deeper with convolutions,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Vol. 07-12-June, pp. 1-9, 2015

[42] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” in *International Conference on Learning Representations*, 2015, pp. 1-14

[43] L. Lab, “Convolutional Neural Networks (LeNet),” 2018 [Online]

Available: http://deeplearning.net/tutorial/lenet.html

[44] R. Haridas, "Convolutional Neural Networks : A Comprehensive Survey," Vol. 14, No. 3, pp. 780-789, 2019

[45] X. Zhang, C. Xv, M. Shen, X. He, and W. Du, "Survey of Convolutional Neural Network," 2018

[46] G. Ediboglu Bartos, "T-H-E Dataset," 2020 [Online] Available: https://github.com/bartosgaye/thedataset. [Accessed: 05-Jul-2020]