symmetry

MDPI

Article

Introducing Urdu Digits Dataset with Demonstration of an Efficient and Robust Noisy Decoder-Based Pseudo Example Generator

Wisal Khan [1,†], Kislay Raj [2,†], Teerath Kumar [3,†], Arunabha M. Roy [4,*] and Bin Luo [1,*]

[1] School of Computer and Technology, Anhui University, Hefei 230039, China
[2] School of Computing, Dublin City University, SFI for Research Training in Artificial Intelligence, Dublin 9, Ireland
[3] Department of Software Engineering, School of Computing, National University of Computer and Emerging Sciences, Islamabad 44000, Pakistan
[4] Aerospace Engineering Department, University of Michigan, Ann Arbor, MI 48109, USA
* Correspondence: arunabhr.umich@gmail.com (A.M.R.); luobin@ahu.edu.cn (B.L.)
† These authors contributed equally to this work.

Abstract: In the present work, we propose a novel method utilizing only a decoder for generation of pseudo-examples, which has shown great success in image classification tasks. The proposed method is particularly constructive when the data are in a limited quantity used for semi-supervised learning (SSL) or few-shot learning (FSL). While most of the previous works have used an autoencoder to improve the classification performance for SSL, using a single autoencoder may generate confusing pseudo-examples that could degrade the classifier's performance. On the other hand, various models that utilize encoder–decoder architecture for sample generation can significantly increase computational overhead. To address the issues mentioned above, we propose an efficient means of generating pseudo-examples by using only the generator (decoder) network separately for each class that has shown to be effective for both SSL and FSL. In our approach, the decoder is trained for each class sample using random noise, and multiple samples are generated using the trained decoder. Our generator-based approach outperforms previous state-of-the-art SSL and FSL approaches. In addition, we released the Urdu digits dataset consisting of 10,000 images, including 8000 training and 2000 test images collected through three different methods for purposes of diversity. Furthermore, we explored the effectiveness of our proposed method on the Urdu digits dataset by using both SSL and FSL, which demonstrated improvement of 3.04% and 1.50% in terms of average accuracy, respectively, illustrating the superiority of the proposed method compared to the current state-of-the-art models.

Keywords: semi-supervised learning (SSL); few-shot learning (FSL); encoder–decoder; Urdu digits dataset; deep learning

Citation: Khan, W.; Raj, K.; Kumar, T.; Roy, A.M.; Luo, B. Introducing Urdu Digits Dataset with Demonstration of an Efficient and Robust Noisy Decoder-Based Pseudo Example Generator. *Symmetry* **2022**, *14*, 1976. https://doi.org/10.3390/sym14101976

Academic Editor: Mihai Postolache

Received: 10 August 2022
Accepted: 16 September 2022
Published: 21 September 2022

1. Introduction

Deep learning (DL) has shown significant performance gain in image classification [1–7], computer vision and object detection [8–10], text classification [11–14] audio classification [15–18], brain–computer interface [19–21], biomedical applications [22–28], and various future computational aspects [29,30]. The state-of-the-art DL methods heavily depend on correctly labeled data. However, the acquisition of labeled data from particularly large datasets is a tedious task to perform [31,32].

With the emergence of big data technology, unlabeled data are sufficiently available on a large scale [32,33], whereas there is only a handful of labeled samples available [34]. The labeling of the large dataset can be expensive, time-consuming, and often unreliable [31,32,34–38]. In this regard, semi-supervised learning (SSL) helps to auto-label the unlabeled datasets using a few labeled data samples. There are several ways to label the unlabeled data, but in the

conventional method, the model is first trained on labeled data, then the trained model is employed to assign the pseudo-labels to unlabeled data. Finally, both the initial labeled data and the pseudo-labeled data can be merged. Thus, SSL can significantly reduce errors and human annotation efforts. However, SSL can result in erroneous results if a significant distribution gap exists between labeled and unlabeled data.

To resolve the issue, various data augmentation methods [2] have been applied to a few available labeled data to match the diversity between labeled and unlabeled data. In recent years, several studies have been geared toward various semi-supervised approaches such as the manifold embedding technique using the pre-constructed graph of unlabeled data [2,35], whereas, in a separate study, the latent representation was exploited by dividing the variational autoencoder (VAE) into two parts, then regularizing the autoencoder by imposing a prior distribution on both parts by making them independent, which led to latent representation [37]. More recently, a new approach was proposed to exploit VAE by adding a classification layer on the topmost layer of the encoder and then merging it with the re-sampled latent layer of the decoder [38].

As mentioned earlier, in some scenarios, only a handful of labeled data are available in the absence of unlabeled data. This can pose challenges concerning obtaining good performance while utilizing a limited handful of only labeled data. Few-shot learning (FSL) is an emerging technique that could be applicable in such cases. In recent years, several approaches considering FSL have been employed. Notably, in [34], firstly, the large network was trained using a few samples; then, knowledge distillation that transfers knowledge from the large model to the small model was optimized to generate pseudo-examples. Along similar lines, a large network was trained for each class separately and then distilled to a small network using a linear function for both small and large networks [39]. In [40], both labeled and unlabeled data were trained simultaneously in a supervised manner where, at first, pseudo-labels were assigned to unlabeled data; subsequently, a denoising autoencoder and dropout were utilized.

However, the aforementioned methods suffer from mediocre performance in terms of accuracy and robustness. To overcome such an issue, in the present work, we proposed an efficient and robust model combining FSL and semi-supervised learning in a unique and efficient way that can significantly improve the accuracy of the model.

The key contributions of the present work can be summarized as follows:

- We propose an efficient way of generating pseudo-examples by using only the decoder network separately for each class that has shown to be effective for both SSL and FSL.
- In the proposed approach, the decoder is trained for each class sample using random noise, and multiple samples are generated using the trained decoder.
- Furthermore, we are the first to release a manually labeled Urdu digits dataset consisting of 10,000 images in total collected through various methods for diversity (https://www.kaggle.com/teerathkumar142/Urdudigits, accessed on (11 April 2022).
- A varied range of experiments were performed, specifically on the Urdu digits dataset, which elucidate the competitiveness and superiority of the proposed network in terms of performance over existing state-of-the-art models.
- Our generator-based approach outperforms previous state-of-the-art SSL and FSL approaches, obtaining an absolute average improvement of 3.04 and 1.50 in terms of accuracy, respectively.

2. Related Work

2.1. Semi-Supervised Learning

Semi-supervised learning (SSL) can be helpful when significantly fewer labeled data are available than large-scale unlabeled data. In recent years, there has been tremendous progress in SSL. Considering that, relevant work on SSL has been briefly reviewed. Recently, an SSL-based encoder–decoder network was extended to VAE that combines the classification layer, mean layer, and standard deviation layer with the topmost encoder layer, combined with the resampled latent layer for the decoder structure [38]. For this architecture, new samples were generated from Gaussian noise fed to the classifier using mean and standard

deviation, which has shown impressive performance. In [41], a joint framework considering representation learning and supervised learning was proposed and then applied to SSL. During training, encoder and supervised classifier loss were significantly reduced. In [37], the latent representation of the autoencoder was divided into two parts, one for content and the other for style. It was concluded that the latent representation associated with the content can be beneficial for classification data. The work demonstrated better performance compared to the vanilla autoencoder. Along similar lines, in [4], firstly, encoder–decoder architecture was trained for each class. In the next phase, the encoder was removed, and noise was passed to the decoder several times to generate diverse samples. However, our experimental results suggest that only training a decoder can be an effective strategy for generating samples for each class. Additionally, the proposed approach by replacing the encoder–decoder [4] with a decoder network can significantly minimize training time and saves computational overhead.

2.2. Few-Shot Learning

Few-shot learning (FSL) can be effective when the availability of labeled data is limited, and the model has to learn utilizing the shallow data. Although numerous methods have been proposed for FSL, we will cover only the relevant works for a fair comparison. In [34], a relatively large network was considered a reference model trained on a few label samples, and knowledge distillation from significant to small models was employed. In addition, pseudo-examples were generated and optimized by employing a high-fidelity optimization procedure. This method illustrated that a relatively small network trained on fewer labels can outperform an initially trained larger network model. In [39], a linear predictor was trained for each class separately and simultaneously distilled to the target model. Subsequently, the bidirectional distillation method was employed, passing the sample to the target and the reference model. During training, the specific class predictor was activated, trained, and distilled to the target model employing MSE. Such a linear distillation technique achieved significant performance improvement. Additionally, various SSL and FSL methods exist which thrive in terms of improving performance. In the current study, the proposed approach can be used for FSL by generating pseudo-examples. To this end, we designed a novel FSL technique to improve performance and achieve state-of-the-art results.

3. Proposed Approach

We propose a novel pseudo-examples generation technique to improve semi-supervised and few-shot learning performance in the present work. The schematic of the basic architecture of our approach is shown in Figure 1, where we train the decoder for a single class. Once trained, we pass normal distribution noise to generate samples of a specific class. We repeat this for all classes. As shown in Figure 2, the overall process of our approach is to employ a separate decoder for each class.

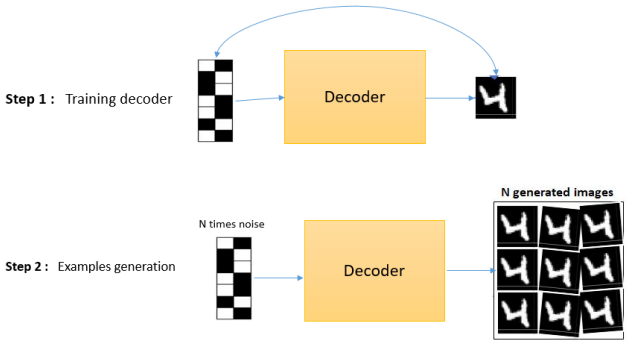

Figure 1. Schematics of pseudo-example generation model consisting of two architecture steps—Step1: train ecoder on specific class samples; Step 2: generate multiple uniform samples that have been passed to a decoder to obtain new samples for the specific class.

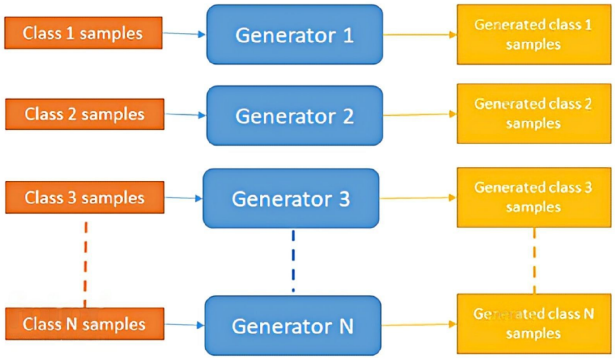

Figure 2. The schematic of the overall architecture: the decoder (generator) was trained for each class in a cascade manner with random noise passed to each decoder.

3.1. Decoder Architecture

As previously mentioned, we used only a decoder to generate the examples of each class. While varieties of decoder architecture exist, in the present work, we chose standard dense layer decoder architecture, where input is the noise of dimension d and is passed and mapped to examples of the specific class C_i. For the decoder, five layers with dimensions of 10, 2000, 500, 500, and 784 are used as shown in Figure 3. We trained the decoder using stochastic gradient descent (SGD) [42]. Each layer uses ReLu activation and kernel initialization [43] with a scale parameter value of $1/3$ with normal distribution.

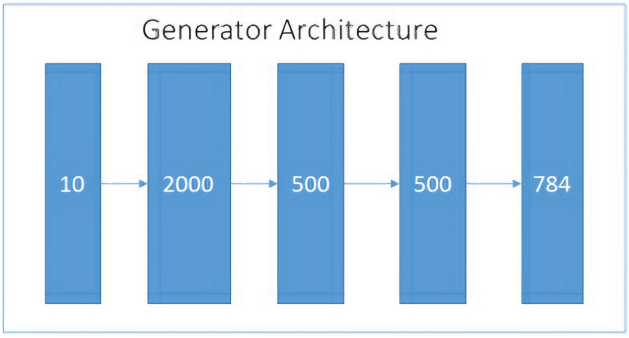

Figure 3. The proposed standard dense layer decoder (generator) architecture that consists of five layers with respective dimensions of 10, 2000, 500, 500, and 784.

Training

During training, we set the batch size to 5. Two different learning rates: 0.1 (for the MNIST dataset) and 0.04 (for FashionMNIST) are prescribed with a momentum value of 0.9. For all cases, standard MSE is evaluated following Equation (1).

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{1}$$

3.2. Work Flow

In this section, we describe the overall workflow and corresponding algorithm of our proposed approach. Let us consider that X_n and Y_n are the limited original samples and their corresponding labels, respectively. Let X_c be the number of examples belonging to the specific class c. In the proposed workflow as described in Algorithm 1, at first, we train a decoder on X_c supplemented with normal noise [44–46] as input that produces $\bar{X}_c^{(i)}$ as the output as shown in line **5** of Algorithm 1. Once the training is completed, we pass normal noise to the trained decoder N times to obtain N examples of that particular class c as reflected in line **5** of Algorithm 1. In order to obtain corresponding Y_c as the

labels of class c for these generated examples, we add an N-dimensional vector having c in the vector with respect to Y as described in line **6** of Algorithm 1. In such a way, we generate N examples with their corresponding labels for a specific class c. Therefore, we initially have no information on X_n and Y_n, which represent the examples of class and their corresponding labels, respectively. In the next step, the whole process was repeated for all classes, as shown in Figure 2 and described in the loop of Algorithm 1. Finally, we have a large number of labeled data in the form of X and Y, respectively, as examples and corresponding labels as described in line **7** of Algorithm 1. In the end, FSL and SSL take the benefits of generated data X and Y from the proposed pseudo-sample generation model.

Algorithm 1 Decoder-Training-And-Generating(X_n,Y_n,N)

Input: X_n: Samples only
Y_n : Labels of the Samples
N: Number of samples per class to be generated
Output: dataGenerated, Labels

1 X=[ ]
Y = []
for *Choose* $c \in UniqueClasses(Y_n)$ **do**
2 model = Decoder() // C
3 reating Decoder model Train(model, Gaussian Noise,X_c) // Training Decoder for class i samples
4 output = SynthesisData(Model,Gaussian Noise, N) // Generating labels
5 Y = $Y \cup [N] * i$ // Generate N samples of class i
6 X = $X \cup output$
7 X, Y // Data and corresponding labels

4. Newly Introduced Urdu Digits Dataset

4.1. Dataset Motivation

The Urdu language is widely used in Asian countries, in particular, Pakistan, India, Bangladesh, and Afghanistan [47]. It is also regarded as the national language of Pakistan. In addition, Urdu, Arabic, Pashto, and Persian languages share various similarities. Due to different applications of Urdu numbers [48–50] that mainly include the automated reading of postal numbers, cheque numbers, digitization, and preserving manuscripts from old ages the acquisition and labeling of Urdu number datasets are of utmost importance and the driving motivation of the current study. However, there is no research work present in the literature that is geared toward the collection and labeling of the largest Urdu language digits dataset. In the present work, to the best of our knowledge, we are the first to release a manually labeled extensive challenging Urdu digits dataset consisting of a total of 10,000 images with 8000 training and 2000 test images. In Urdu digits dataset, some digits, in particular, 3 and 4 are almost symmetric in terms of shape. Additionally, digits 7 and 8 are in reflection symmetry. Such kinds of partial/ non-partial symmetric cases induce additional challenges in the dataset for the neural network to learn.

4.2. Dataset Collection

We used three different methods to collect data: the Microsoft (MS) paint tool, online search, and paper-based collection from different participants to increase the variability in the dataset.

4.2.1. Microsoft (MS) Paint-Based Collection

In MS paint-based collection, we set up an MS paint tool, in which we fixed the window at 28×28 pixels, and then filled it with the black background color. The numbers are written in a fixed window size by five different people using various brush sizes. Following such an approach, a different set of Urdu digits of a total of 37,000 images was

generated and collected. Some representative dataset samples collected with the MS paint tool are shown in Figure 4.

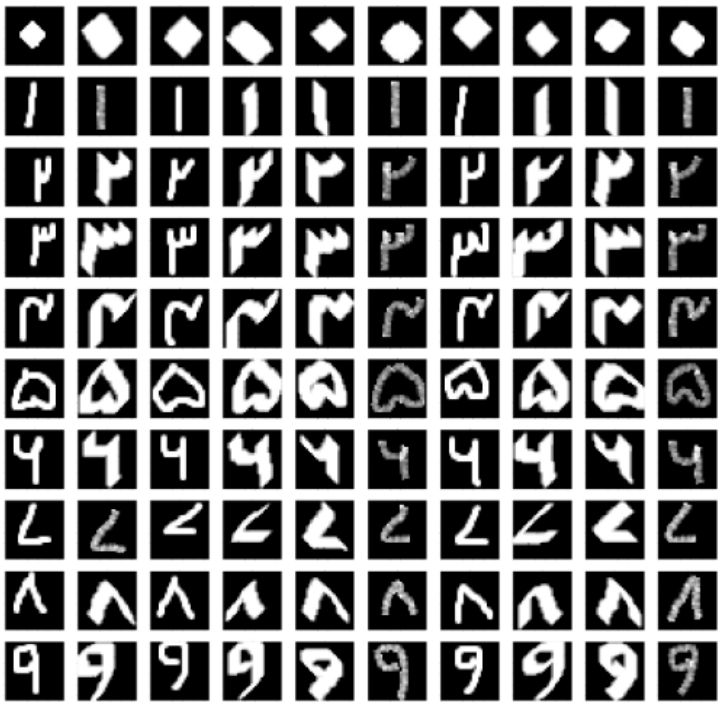

Figure 4. Representative samples in Urdu digits dataset collected with MS paint tool where different variants of digits are arranged row-wise.

4.2.2. Online Data Collection

To include diversity in the dataset, we used Python code scarper to obtain images from the internet using the keyword Urdu Digit, then asked ten different users to crop the Urdu numbers. In this way, we collected a total of around 3000 additional images.

4.2.3. Paper-Based Data Collection

To further increase the diversity of the data, we asked ten different participants to write multiple numbers on an A4 page, take the picture through the mobile camera, and then crop those numbers. Using this setup, we collected around 60,000 images. Some representative samples from the paper-based data collection procedure are depicted in Figure 5. Overall, we have 10,000 images in the Urdu digits dataset consisting of 8000 images for the training set and 2000 images for the testing set. After collecting data through the aforementioned procedures, we performed pre-processing on the image data. First, we converted the digit into white and the background into black color for all the collected images. Then we resized the image by 28×28 pixels while maintaining the same aspect ratio. After resizing, we normalize images in the range of 0 to 1, dividing by 255. Following the preprocessing steps similar to MNIST and Fashion-MNIST dataset, we keep the Urdu digits dataset in grayscale while not applying any mean centering to the collected images.

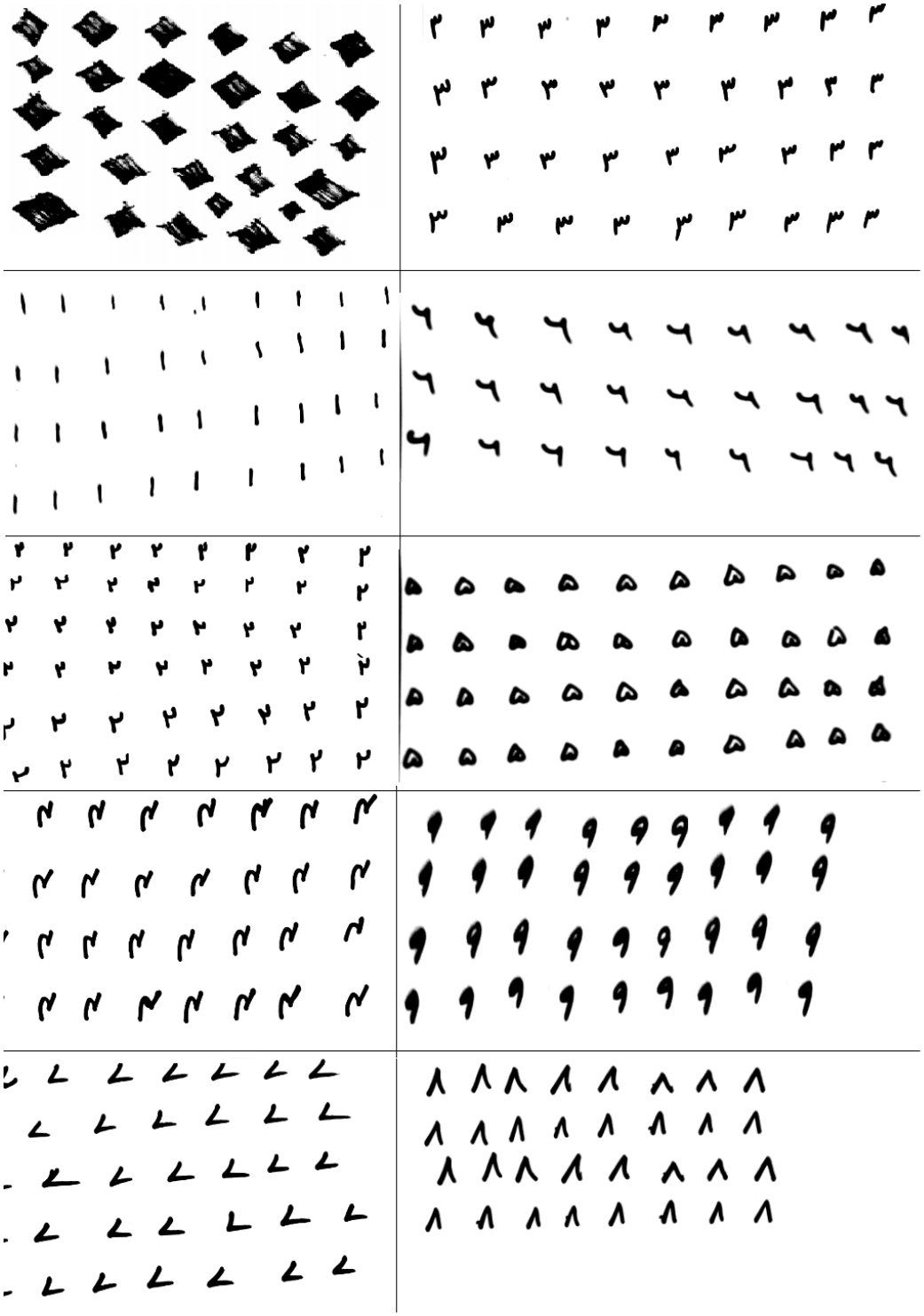

Figure 5. Representative samples in Urdu digits dataset collected with paper-based data collection where different variants of digits are grouped.

5. Experiment and Results

In this section, we report our experimental findings in order to demonstrate the performance of the proposed model. We followed various settings in terms of datasets, CNN architectures, and the model parameters, which are detailed in the subsequent sections. We use Equation (2) to calculate the entropy.

$$L_{\text{cross-entropy}}\left(\hat{\mathbf{y}}, \mathbf{y}\right) = -\sum_{i} y_i \log(\hat{y}_i) \tag{2}$$

5.1. Datasets

To check the effectiveness of the proposed approach, we used MNIST [51] and Fashion-MNIST [52] datasets using semi-supervised and few-shot learning. The MNIST dataset has 60,000 training and 10,000 testing samples of 10-digit classes (range from 0 to 9) with 28 × 28 grayscale images. The fashion-MNIST dataset which is used for clothes and accessories has 60,000 training and 10,000 testing samples of ten different classes with sizes of 28 × 28 grayscale images. Some representative samples of MNIST and Fashion-MNIST datasets are shown in Figures 6 and 7, respectively. Additionally, we performed an extensive analysis on the newly introduced Urdu digit dataset.

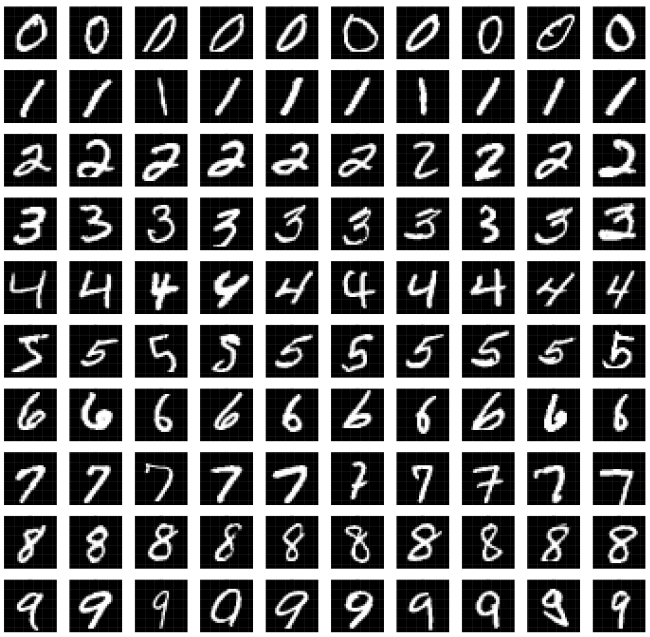

Figure 6. Representative generated pseudo-samples from the proposed model for MNIST dataset consisting of 10-digit classes (range from 0 to 9) with 28 × 28 grayscale images.

Figure 7. Representative generated pseudo-samples from the proposed model for the Fashion-MNIST dataset consisting of ten different classes with sizes of 28 × 28 grayscale images.

5.2. Result from Semi-Supervised Learning

In this section, we report the results obtained from the SSL. We implemented and used a CNN network [4,38] for SSL results based on autoencoder using standard deviation and mean. For the experiments, we used a total of 100 and 1000 labels from the MNIST and Fashion-MNIST datasets, respectively. For our Urdu digits dataset, we used various numbers of labels, i.e., 100, 200, 500, 1000, and 2000. Utilizing our proposed method, we then generated the data and subsequently the SSL model was applied.

For MNIST and Fashion-MNIST datasets, various state-of-the-art models including CCNs [38], (MS) [38], and CNNs (AE) [4] were considered and directly compared with the proposed model. Note, in these tables, CNNs correspond to a supervised model, CNNs (MS) refers to semi-supervised based on the mean standard deviation layers of the autoencoder, whereas CNNs (Our) is based on a semi-supervised learning method using pseudo-examples. The accuracy values obtained from these models are presented in Tables 1 and 2 for MNIST and Fashion-MNIST datasets, respectively. For the MNIST dataset, CCNs (MS) provides the best results with accuracy values of $81.10 \pm 6.16\%$ for 100 labels, as shown in Table 1. However, with increasing labels 1000, our proposed model achieved the best accuracy of $95.11 \pm 2.30\%$ which is a 1.40% accuracy improvement over CCNs (MS). However, for the Fashion-MNIST dataset, our model provides the best results, achieving an accuracy of $74.52 \pm 1.42\%$, whereas with an increasing number of sample size 1000, CCNs (MS) provides the best result with an accuracy of $83.67 \pm 1.09\%$. In almost all cases, our approach improves the accuracy by over 2% except for the case of 1000 label MNIST. Overall, for both datasets, the proposed model illustrates its superiority by providing state-of-the-art results. Finally, for the Urdu digits dataset, the accuracy values are presented for various numbers of labels as shown in Table 3. It is noteworthy that with an increasing number of labels, the accuracy improves. For example, with a relatively small number of labels, 20, the accuracy reaches up to 84.90%, whereas it attains an impressive accuracy value of 96.70% for a large number of labels, 200. In short, the proposed model demonstrated superior performance with a reasonable amount of labeled data for the Urdu digits dataset.

Table 1. Comparison of accuracy values (in %) between various state-of-the-art models and the proposed model evaluated in MNIST dataset.

Model	100 Labels	1000 Labels
CCNs [38]	76.51 ± 3.21	89.11 ± 2.10
CCNs (MS) [38]	$\mathbf{81.10 \pm 6.16}$	94.51 ± 1.13
CNNss (AE) [4]	78.89 ± 1.92	89.33 ± 2.17
CNNss (Ours)	78.16 ± 1.10	$\mathbf{95.11 \pm 2.30}$

Table 2. Comparison of accuracy values (in %) between various state-of-the-art models and the proposed model evaluated in Fashion-MNIST dataset.

Model	100 Labels	1000 Labels
CCNs [38]	66.22 ± 1.02	80.30 ± 1.98
CCNs (MS) [38]	72.41 ± 0.87	$\mathbf{83.67 \pm 1.09}$
CNNs (AE) [4]	72.51 ± 2.57	79.93 ± 1.46
CNNs (Ours)	$\mathbf{74.52 \pm 1.42}$	82.71 ± 1.47

Table 3. Accuracy values (in %) obtained from the proposed SSL model for different numbers of labels evaluated in Urdu digits dataset.

Labels	10	20	50	100	200
SSL (Ours)	80.15	84.90	89.05	93.45	96.70

5.3. Results from Few-Shot Learning

In this section, we reported the results obtained from the FSL. For the comparison, a large knowledge distillation model [34] was considered and trained on a few label samples. In addition, pseudo-examples were generated and then optimized and selected using high fidelity techniques. This method was shown to outperform the original large model using the relatively small model on a few label datasets. In our approach, we utilize a relatively small CNN model [34], conduct experiments on various datasets, and compare the results between these two models. At first, we generate a various number of examples for each experiment using a few selected examples. Selected examples are then combined to train the model. Thus, a different number of examples is generated using our approach. Several examples that consider various hyperparameters were discussed. Each experiment was repeated three times and average accuracy was reported.

For MNIST, Fashion-MNIST, and Urdu digits datasets, we used a different number of samples i.e., 10, 20, 50, 100, and 200 per class. Using these few examples, we generated multiple samples and then trained the model. The model's performance is presented in Tables 4–6 with respect to MNIST, Fashion-MNIST, and Urdu digits datasets, respectively. As shown in Table 4, our proposed model outperformed the current state-of-the-art model by achieving accuracy of 50.33 % and 54.59% in 10 and 20 examples per class, respectively. For a relatively higher number of examples per class, our model performs comparatively with the performance of the state-of-the-art models.

Table 4. Accuracy comparison between FSL and other models evaluated on MNIST dataset, where Imt = Imitation, opt = optimize, fd = fidelity.

Labels	10	20	50	100	200
NN [34]	37.90	46.00	66.00	78.30	86.70
GP [34]	39.90	51.60	64.60	73.20	80.00
Imt [34]	43.50	51.20	67.70	78.10	86.10
Imt, opt [34]	44.10	53.70	70.00	79.50	**86.70**
Imt, opt, fd [34]	44.10	53.90	**70.40**	**80.00**	86.60
CNN (AE) [4]	46.30	54.30	59.40	67.40	76.40
FSL (Ours)	**50.33**	**54.59**	68.14	76.34	86.41

Table 5. Accuracy comparison between FSL and other models evaluated on Fashion-MNIST Dataset, where Imt = Imitation, opt = optimize, fd = fidelity.

Labels	10	20	50	100	200
NN [34]	39.30	47.90	58.30	64.90	71.30
GP [34]	44.60	52.40	59.90	65.70	71.40
Imt [34]	43.60	50.90	60.00	67.30	72.50
Imt, opt [34]	41.20	49.70	60.10	67.30	72.20
Imt, opt, fd [34]	44.80	52.70	62.10	67.30	72.50
CNN (AE) [4]	48.20	56.10	58.80	65.80	69.49
FSL (Ours)	**54.42**	**59.06**	**67.51**	**70.82**	**74.37**

Table 6. Accuracy values from FSL evaluated on Urdu digit dataset

Labels	10	20	50	100	200
Our	78.80	83.65	87.95	92.56	96.0

For the Fashion-MNIST dataset, as shown in Table 5, the proposed method with the FSL model outperformed all other current state-of-the-art models in terms of accuracy for the same network configuration and optimization scheme. Thus, our extensive experiments

elucidate the superior performance in terms of the accuracy of the proposed FSL method for various numbers of levels.

6. Parametric Study

In both SSL and FSL experiments, the generated number of examples is different for the different number of selected levels. Thus, the number of generated examples can be treated as one of the hyperparameters. Therefore, we conduct extensive experiments to find the influence of a number of generated examples on the accuracy of the model. For the calculation of average accuracy, each experiment was repeated three times.

6.1. Performance of SSL

In the SSL model, we selected a few labeled samples ranging from 1000 to 5000 with an increment of 1000 for each individual class. Each sample size was trained in the SSL network at each interval. The experimental results suggest that generating a different number of examples can significantly influence the performance of the SSL model concerning the different numbers of selected examples for various datasets. As we can see from Figure 8, where the x- and y-axes represent the number of generated examples and the accuracy value, respectively. The increasing number of examples boosts the performance of both the MNIST and the FMINST datasets. For example, SSL provides the best performance in the case of 10 samples per class with 2000 generated samples for the MNIST dataset, as shown in Figure 8. Additionally, 100 samples per class with 5000 generated samples show superior performance. We treated its superiority in terms of accuracy. For the Fashion-MNIST dataset, as shown in Figure 8, generating 2000 samples gives the best performance in the case of 10 samples per class. With increasing 100 samples per class, generating 5000 samples provides the best performance.

Similarly, we extended our experiments on the Urdu digits dataset using a different number of examples as shown in Figure 9. Our extensive study reveals that various example sizes together with generated examples can significantly influence the accuracy of the model.

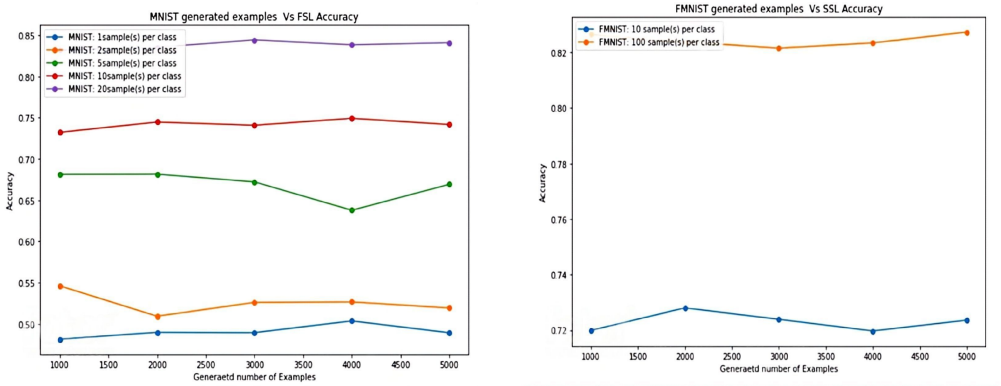

Figure 8. (**Left**) Accuracy vs. number of generated examples from semi-supervised learning (SSL) for MNIST; (**Right**) Accuracy vs. number of generated examples from semi-supervised learning (SSL) for Fashion-MNIST.

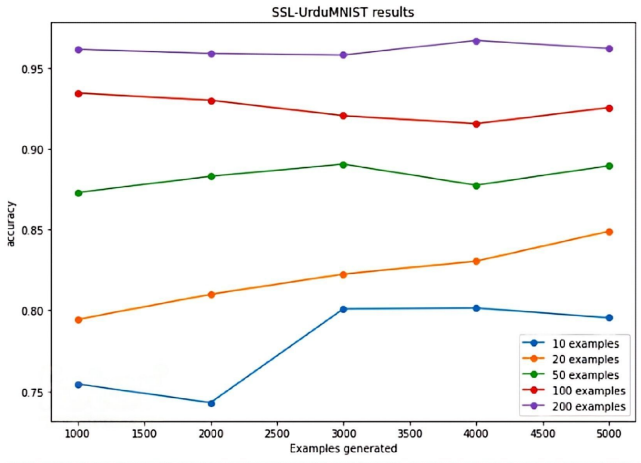

Figure 9. Accuracy vs. number of generated examples from semi-supervised learning for Urdu digits dataset.

6.2. Performance of FSL

For few-shot learning, we first selected several examples to train the generator for those selected examples. Then we generated a different number of examples. In our experiment, the generated number of examples can be treated as a hyper-parameter. We generated 1000 to 5000 examples with an interval of 1000. At each interval, we trained the CNN model. In each case, generating a different number of examples gives different performances for different datasets, as we can see from Figure 10 for the MNIST dataset and Fashion-MNIST dataset. The x-axis represents the number of generated examples, while the y-axis shows the performance. For the MNIST dataset, as shown in Figure 10, for 200 examples per class, the proposed model provides the best performance. Similarly, for the Fashion-MNIST dataset, as shown in Figure 11, in the case of 1 sample per class, generating 3000 samples gives the best performance; similarly, in the case of 2 samples per class, 5 samples per class, 10 samples per class, and 20 samples per class, generating 5000 samples, 3000 samples, 5000 samples, and 4000 samples, respectively, gives the best performance. Similarly, we conducted experiments on the Urdu digits dataset using a different number of examples, as shown in Figure 10. From the comparison, we can see that higher numbers of examples per class significantly improves the accuracy. However, a moderate amount of examples can provide state-of-the-art results for relatively higher numbers of example generation.

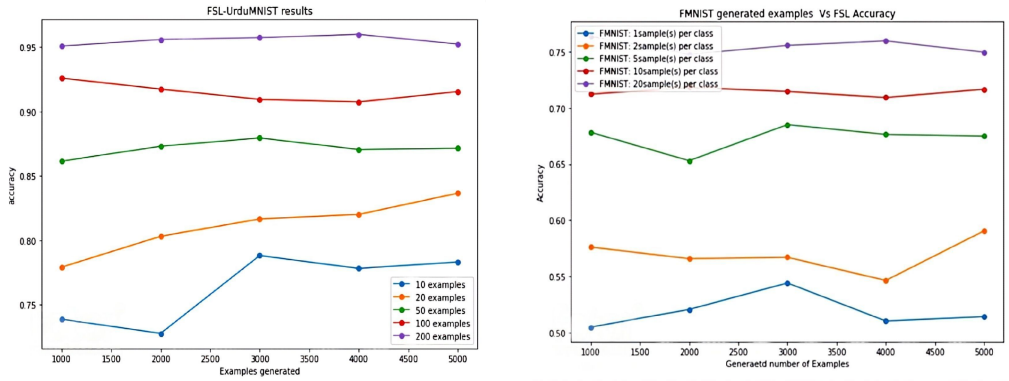

Figure 10. (**Left**) Accuracy vs. the number of generated examples from few-shot learning (FSL) for the Urdu digit dataset; (**Right**) Fashion-MNIST generated examples vs. FSL Accuracy.

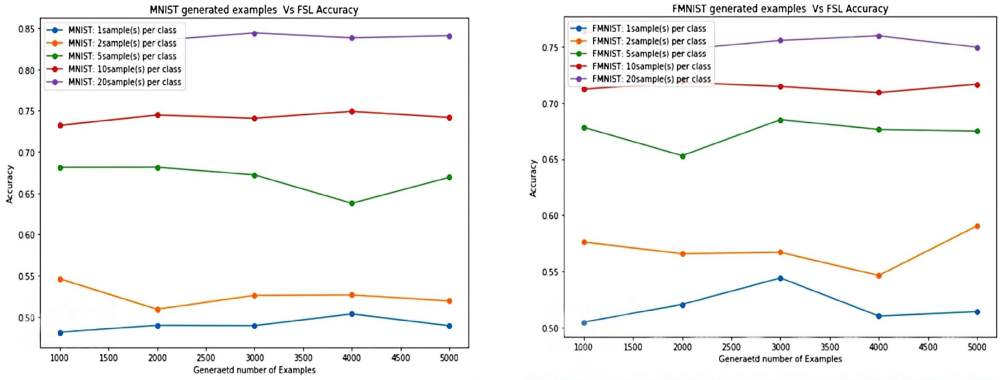

Figure 11. (**Left**) Accuracy vs. number of generated examples from few-shot learning (FSL) for MNIST; (**Right**) Accuracy vs. number of generated examples from few-shot learning (FSL) for Fashion-MNIST.

7. Conclusions

In summary, in the current study, we proposed a novel approach to improve performance concerning generating pseudo-examples by addressing the current drawbacks in the existing state-of-the-art approaches. In the proposed model, we only used a decoder network, which is easier and faster to train compared to both encoder–decoder architectures. Another advantage of such a strategy is that training a decoder using random values and images can generate different images of the same class, which is impossible in an encoder–decoder that only generates images corresponding to the same training class. Furthermore, we are the first to release a manually labeled Urdu digits dataset collected through various methods. In order to show the efficacy of the proposed approach, we extensively tested the model in different datasets with various samples using both SSL and FSL. The performance comparison in terms of average classification accuracy demonstrates the superiority of the proposed model in that it outperforms current state-of-the-art models for both SSL and FSL. Future works could be geared toward designing an efficient encoder–decoder model, replacing the decoder-only model and building various other valuable datasets.

Author Contributions: Conceptualization, W.K., K.R., T.K., B.L. and A.M.R.; methodology, W.K., K.R. and T.K.; validation, W.K. and T.K.; formal analysis, W.K. and T.K.; investigation, W.K. and K.R.; resources, B.L. and A.M.R.; data curation, W.K.; writing—original draft preparation, W.K., R.R. and K.R.; writing—review and editing, B.L. and A.M.R.; visualization, T.K.; supervision, B.L. and A.M.R.; project administration, B.L. and A.M.R.; funding acquisition, B.L. and A.M.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by Aeronautical Research and Development Board (Grant No. DARO/08/1051450/M/I). The APC was fully funded by MDPI AG.

Informed Consent Statement: We published and used the images with the consent of participants.

Data Availability Statement: Not applicable.

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

References

1. Vailaya, A.; Jain, A.; Zhang, H. On image classification: City images vs. landscapes. *Pattern Recognit.* **1998**, *31*, 1921–1935. [CrossRef]
2. Shorten, C.; Khoshgoftaar, T. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
3. Kumar, T.; Park, J.; Ali, M.; Uddin, A.; Ko, J.; Bae, S. Binary-classifiers-enabled filters for semi-supervised learning. *IEEE Access* **2021**, *9*, 167663–167673. [CrossRef]
4. Kumar, T.; Park, J.; Ali, M.; Uddin, A.; Bae, S. Class Specific Autoencoders Enhance Sample Diversity. *J. Broadcast Eng.* **2021**, *26*, 844–854.

5. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 84–90. [CrossRef]
6. Jamil, S.; Abbas, M.S.; Roy, A.M. Distinguishing Malicious Drones Using Vision Transformer. *AI* **2022**, *3*, 260–273. [CrossRef]
7. Alam, A.; Ullah, I.; Lee, Y. Video Big Data Analytics in the Cloud: A Reference Architecture, Survey, Opportunities, and Open Research Issues. *IEEE Access* **2020**, *8*, 152377–152422. [CrossRef]
8. Roy, A.M.; Bhaduri, J. A Deep Learning Enabled Multi-Class Plant Disease Detection Model Based on Computer Vision *AI* **2022**, *2*, 413–428. [CrossRef]
9. Roy, A.M.; Bose, R.; Bhaduri, J. A fast accurate fine-grain object detection model based on YOLOv4 deep neural network *Neural Comput. Appl.* **2022**, *34*, 3895–3921.
10. Roy, A.M.; Bose, R.; Bhaduri, J. Real-time growth stage detection model for high degree of occultation using DenseNet-fused YOLOv4. *Comput. Electron. Agric.* **2022**, *193*, 106694. [CrossRef]
11. Ullah, I.; Khan, S.; Imran, M.; Lee, Y. RweetMiner: Automatic identification and categorization of help requests on twitter during disasters. *Expert Syst. Appl.* **2021**, *176*, 114787.
12. Kowsari, K.; Jafari Meimandi, K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text classification algorithms: A survey. *Information* **2019**, *10*, 150.
13. Aggarwal, C.; Zhai, C. A survey of text classification algorithms. In *Mining Text Data*; Springer: Boston, MA, USA, 2012; pp. 163–222.
14. Ikonomakis, M.; Kotsiantis, S.; Tampakas, V. Text classification using machine learning techniques. *WSEAS Trans. Comput.* **2005**, *4*, 966–974.
15. Kumar, T.; Park, J.; Bae, S. Intra-Class Random Erasing (ICRE) augmentation for audio classification. In *Proceedings of the Korean Society of Broadcast Engineers Conference*; The Korean Institute of Broadcast and Media Engineers: Anseong, Korea, 2020; pp. 244–247.
16. Park, J.; Kumar, T.; Bae, S. Search for optimal data augmentation policy for environmental sound classification with deep neural networks. *J. Broadcast Eng.* **2020**, *25*, 854–860.
17. Chandio, A.; Shen, Y.; Bendechache, M.; Inayat, I.; Kumar, T. AUDD: Audio Urdu digits dataset for automatic audio Urdu digit recognition. *Appl. Sci.* **2021**, *11*, 8842.
18. Turab, M.; Kumar, T.; Bendechache, M.; Saber, T. Investigating Multi-Feature Selection and Ensembling for Audio Classification. *arXiv* **2022**, arXiv:2206.07511.
19. Roy, A.M. An efficient multi-scale CNN model with intrinsic feature integration for motor imagery EEG subject classification in brain-machine interfaces *Biomed. Signal Process. Control* **2022**, *74*, 103496.
20. Roy, A.M. A multi-scale fusion CNN model based on adaptive transfer learning for multi-class MI-classification in BCI system. *bioRxiv* **2022**. [CrossRef]
21. Roy, A.M. Adaptive transfer learning-based multiscale feature fused deep convolutional neural network for EEG MI multiclassification in brain–computer interface *Eng. Appl. Artif. Intell.* **2022**, *116*, 105347. [CrossRef]
22. Ranjbarzadeh, R.; Tataei Sarshar, N.; Jafarzadeh Ghoushchi, S.; Saleh Esfahani, M.; Parhizkar, M.; Pourasad, Y.; Anari, S.; Bendechache, M. MRFE-CNN: Multi-route feature extraction model for breast tumor segmentation in Mammograms using a convolutional neural network. *Ann. Oper. Res.* **2022**,*11*. [CrossRef]
23. Baseri Saadi, S.; Tataei Sarshar, N.; Sadeghi, S.; Ranjbarzadeh, R.; Kooshki Forooshani, M.; Bendechache, M. Investigation of Effectiveness of Shuffled Frog-Leaping Optimizer in Training a Convolution Neural Network. *J. Healthc. Eng.* **2022**, *2022*, 4703682. [CrossRef] [PubMed]
24. Saadi, S.; Ranjbarzadeh, R.; Amirabadi, A.; Ghoushchi, S.; Kazemi, O.; Azadikhah, S.; Bendechache, M.; Others Osteolysis: A literature review of basic science and potential computer-based image processing detection methods. *Comput. Intell. Neurosci.* **2021**, *2021*, 4196241. [CrossRef] [PubMed]
25. Valizadeh, A.; Jafarzadeh Ghoushchi, S.; Ranjbarzadeh, R.; Pourasad, Y. Presentation of a segmentation method for a diabetic retinopathy patient's fundus region detection using a convolutional neural network. *Comput. Intell. Neurosci.* **2021**, *2021*, 7714351.
26. Jafarzadeh Ghoushchi, S.; Memarpour Ghiaci, A.; Rahnamay Bonab, S.; Ranjbarzadeh, R. Barriers to circular economy implementation in designing of sustainable medical waste management systems using a new extended decision-making and FMEA models. *Environ. Sci. Pollut. Res.* **2022**, *32*. [CrossRef]
27. Ranjbarzadeh, R.; Dorosti, S.; Jafarzadeh Ghoushchi, S.; Safavi, S.; Razmjooy, N.; Tataei Sarshar, N.; Anari, S.; Bendechache, M. Nerve optic segmentation in CT images using a deep learning model and a texture descriptor. *Complex Intell. Syst.* **2022**, *8*, 3543–3557.
28. Ghoushchi, S.; Ranjbarzadeh, R.; Dadkhah, A.; Pourasad, Y.; Bendechache, M. An extended approach to predict retinopathy in diabetic patients using the genetic algorithm and fuzzy C-means. *BioMed Res. Int.* **2021**, *2021*, 5597222. [CrossRef] [PubMed]
29. Roy, A.M. Evolution of martensitic nanostructure in NiAl alloys: Tip splitting and bending. *Mater. Sci. Res. India*. **2020**, *17*, 3–6. [CrossRef]
30. Roy, A.M. Finite element framework for efficient design of three dimensional multicomponent composite helicopter rotor blade system. *Eng* **2021**, *2*, 69–79. [CrossRef]

31. Li, W.; Wang, Z.; Li, J.; Polson, J.; Speier, W.; Arnold, C. Semi-supervised learning based on generative adversarial network: A comparison between good GAN and bad GAN approach. In Proceedings of the CVPR Workshops; Long Beach, CA, USA, 16–20 June 2019; pp. 55–65.
32. Kingma, D.; Mohamed, S.; Jimenez Rezende, D.; Welling, M. Semi-supervised learning with deep generative models. In Proceedings of the Advances In Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; Volume 27.
33. Khan, W.; Kumar, T.; Cheng, Z.; Raj, K.; Roy, A. M.; Luo, B. SQL and NoSQL Databases Software architectures performance analysis and assessments - A Systematic Literature review. *arXiv* **2022**, arXiv:2209.06977.
34. Kimura, A.; Ghahramani, Z.; Takeuchi, K.; Iwata, T.; Ueda, N. Few-shot learning of neural networks from scratch by pseudo-example optimization. *arXiv* **2018**, arXiv:1802.03039.
35. Weston, J.; Ratle, F.; Mobahi, H.; Collobert, R. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*; Springer: Cham, Switzerland, 2012; pp. 639–655.
36. Li, Y.; Pan, Q.; Wang, S.; Peng, H.; Yang, T.; Cambria, E. Disentangled variational auto-encoder for semi-supervised learning. *Inf. Sci.* **2019**, *482*, 73–85. [CrossRef]
37. Tachibana, R.; Matsubara, T.; Uehara, K. Semi-supervised learning using adversarial networks. In Proceedings of the 2016 IEEE/ACIS 15th International Conference On Computer And Information Science (ICIS), Okayama, Japan, 26–29 June 2016; pp. 1–6.
38. Berkhahn, F.; Keys, R.; Ouertani, W.; Shetty, N.; Geißler, D. Augmenting variational autoencoders with sparse labels: A unified framework for unsupervised, semi-(un) supervised, and supervised learning. *arXiv* **2019**, arXiv:1908.03015.
39. Asadulaev, A.; Kuznetsov, I.; Filchenkov, A. Interpretable few-shot learning via linear distillation. *arXiv* **2019**, arXiv:1906.05431.
40. Lee, D. Others Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *Workshop Chall. Represent. Learn. ICML* **2013**, *3*, 896.
41. Haiyan, W.; Haomin, Y.; Xueming, L.; Haijun, R. Semi-supervised autoencoder: A joint approach of representation and classification. In Proceedings of the 2015 International Conference On Computational Intelligence And Communication Networks (CICN), Jabalpur, India, 12–14 December 2015; pp. 1424–1430.
42. Robbins, H.; Monro, S. A stochastic approximation method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference On Computer Vision, Santiago, Chile, 7–13 December2015; pp. 1026–1034.
44. Mohammed Abd-Alsalam Selami, A.; Freidoon Fadhil, A. A study of the effects of gaussian noise on image features. *Kirkuk Univ. J.-Sci. Stud.* **2016**, *11*, 152–169. [CrossRef]
45. Russo, F. A method for estimation and filtering of Gaussian noise in images. *IEEE Trans. Instrum. Meas.* **2003**, *52*, 1148–1154. [CrossRef]
46. Kaur, P.; Singh, J. A study on the effect of Gaussian noise on PSNR value for digital images. *Int. J. Comput. Electr. Eng.* **2011**, *3*, 319. [CrossRef]
47. Hussain, S. Resources for Urdu language processing. In Proceedings of the 6th Workshop On Asian Language Resources, Hyderabad, India, 11–12 January 2008.
48. Plötz, T.; Fink, G. Markov models for offline handwriting recognition: A survey. *Int. J. Doc. Anal. Recognit. (IJDAR)*. **2009**, *12*, 269–298. [CrossRef]
49. Lee, C.; Leedham, C. A new hybrid approach to handwritten address verification. *Int. J. Comput. Vis.* **2004**, *57*, 107–120. [CrossRef]
50. Ul-Hasan, A.; Ahmed, S.; Rashid, F.; Shafait, F.; Breuel, T. Offline printed Urdu Nastaleeq script recognition with bidirectional LSTM networks. In Proceedings of the 2013 12th International Conference On Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; pp. 1061–1065.
51. LeCun, Y. The MNIST Database of Handwritten Digits. 1998. Available online: http://yann.Lecun.Com/exdb/mnist/ (accessed on 11 December 2021).
52. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.