Optimization-Based Sensing for Compressed Learning Based on Structural Information

Mehdi Abdollahpour, Carsten Bockelmann, Armin Dekorsy
Department of Communications Engineering, University of Bremen, Bremen 28359, Germany
Email: abdollahpour@ant.uni-bremen.de, bockelmann@ant.uni-bremen.de, dekorsy@ant.uni-bremen.de

2024 12th European Workshop on Visual Information Processing (EUVIP) | 979-8-3503-9114-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/EUVIP61797.2024.10772901

***Abstract*—This work presents a novel compressed learning framework designed for optimized image sampling for classification tasks. In this study, we replace the conventional random sampling method in classical compressed sensing with an optimization-based approach to derive a specific sensing mask that maximizes classification accuracy within a dataset. The proposed approach recognizes and uses structural information from input images for sampling, specifically relevant to the downstream task. Leveraging a genetic algorithm as an optimizer within the framework, we aim to improve the classification performance of a pre-trained convolutional neural network by enhancing the sensing mask. Two benchmark datasets, MNIST and Fashion MNIST, are used for performance evaluation. In addition to the masking, the framework is tested in a traditional sensing setup with sensing matrix optimization. The results suggest that optimization-based sampling could be a good alternative to random sampling in compressed sensing due to its superior performance.**

***Index Terms*—compressed sensing, compressed learning, optimization, classification.**

I. Introduction

Compressed sensing (CS) uses a mathematical framework for data acquisition to reconstruct the original data from a small number of sampling points. It gained attention because it was able to guarantee perfect signal reconstruction with high probability even when sampling below the Nyquist rate. CS has the most potential when the signal has a sparse representation in another domain [1]. Most of the classical signal reconstruction methods use optimization algorithms and rely on the sparsity of the data to reconstruct data. The high complexity of the optimization process to find the sparsifying basis simultaneously with signal reconstruction makes CS cumbersome in practical implementations. In addition, finding a sparsity level prior to the reconstruction is necessary, which adds to the difficulty [2], [3].

To overcome these problems, deep learning-based methods have been developed that rely on neural networks to reconstruct the original data from compressively sensed random measurements in a non-iterative way. The reported results show significant improvements in reconstruction error and time complexity in comparison to state-of-the-art iterative CS reconstruction methods [4]–[6].

By means of deep learning methods, compressed learning (CL) also gained attention. CL integrates compressed sensing and machine learning into a single framework. In this framework direct inference can be carried out with compressively sensed measurements. Direct inference means the reconstruction of the data is not necessary, and downstream machine learning tasks including detection and recognition can be carried out directly from compressed data [7], [8]. For the first time, Calderbank et al. [9] demonstrated that under specific conditions, the support vector machine (SVM) classifier's performance in a compressed domain is nearly equivalent to that of the linear threshold classifier in an uncompressed domain.

In CL, the previous requirement of random sampling, which was a prerequisite for traditional CS-based reconstruction, is no longer necessary. As a result, designing a specific sensing matrix through optimization instead of using random sampling matrices has gained attention lately. The majority of the research in this field focuses on integrating downstream machine learning techniques and optimized sampling in a deep learning-based end-to-end framework [10], [11]. However, in an end-to-end setup where the inference model training and the sensing matrix optimization take place jointly in a neural network, it is not possible to evaluate the effectiveness of the optimized sensing method.

In a different approach, Fakruddin Ali et al. [12] have tried to use the genetic algorithm and wavelet domain deep CNN to find the optimized sensing mask for classification. However, their method is responsive at higher sampling rates, and the employed dataset in that study does not contain any structural information and could not illustrate the advantages of the masking.

Motivated by these problems, we introduce a CS framework that finds an optimized sensing mask to achieve the highest accuracy in a downstream classification task. In contrast to end-to-end models, we separate the processes of sensing mask optimization and classifier training. This separation provides an opportunity to evaluate how the choice of a sensing mask influences the outcomes of classification. In addition, we propose a novel approach to train a CNN that is sensitive to changes in the sensing point of the input data. This approach makes sensing mask optimization faster and more efficient. The proposed framework is flexible and not designed to be task-specific. It could be used in different sensing setups with different classifiers and optimizers. Furthermore, the framework maintains its consistency for different sensing rates, as low as 1%.

In the current paper, two different sensing setups are studied. The first case is the classical formulation of CS, where $\mathbf{x}$ is a

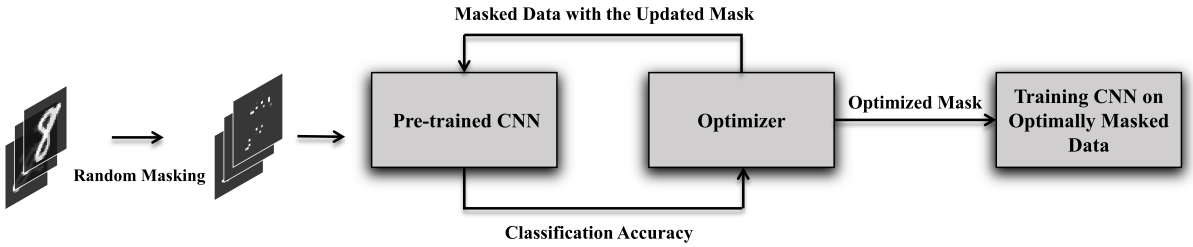

Fig. 1. Workflow of the proposed framework.

$N \times 1$ signal to be compressed, $\mathbf{y}$ is the $M \times 1$ compressed signals, and Φ is an $M \times N$ sensing matrix:

$$\mathbf{y} = \Phi \mathbf{x}. \tag{1}$$

In this equation, the sensing matrix performs a transformation from a higher dimension to a lower dimension, which can be interpreted as compression or linear projection. In the second case, sensing can be formulated as the following where $\odot$ represents Hadamard or element-wise production, and the sensing mask Φ' has the same size as $\mathbf{x}$:

$$\mathbf{y}' = \Phi' \odot \mathbf{x}. \tag{2}$$

Masking could also be implemented by a single-pixel camera, a practical implementation of the CS problem without additional complications. The main concentration of this study will be masking. In addition, an experiment is conducted to test the proposed method in a classical setup and compare it with existing studies in the CL literature.

II. Methodology

The workflow of the proposed framework is illustrated in Fig. 1. From left to right, we first have random masking of the dataset, followed by a pre-trained CNN and optimizer to find the optimized sensing mask, and lastly, training CNN from scratch using compressed data with the optimized sensing mask. We will continue by describing the framework in more detail.

A. Framework of Optimization for Masking

The whole framework has two major parts: the classification model, and the optimizer. Making a connection between these two parts in an effective way is the key point in our problem. The goal here is to find a sensing mask with a specific number of sampling points to apply to the data and get the highest classification accuracy.

The optimizer works in an iterative way until it converges to a specific mask. However, training and testing of the classification model at each iteration could be time-consuming. Therefore, in our scenario, before optimization, the classifier will be trained and just be used for testing different masks during the optimization. In addition, the classifier should be trained in a way that can respond to different combinations of sampling points in the input data. In other words, it should be sensitive to changes in the mask.

This concept is implemented through the manipulation of the training data. Every image in the dataset is masked with a different random mask so that pixels are chosen randomly from different images. The masking is carried out by Hadamard multiplication of the mask and image. The masked data is then used to train a convolutional neural network (CNN). By doing so, CNN learns how to interpret incomplete or partially occluded data. This exposure to diverse masked data during training helps the CNN to be resilient in dealing with different missing patterns during inference.

Additionally, this approach significantly reduces the time required to find the optimized mask because instead of training the model at each iteration of the optimization we train it once and just test the model at each iteration.

The next step is the optimization of the sensing mask. In our problem, a large search space exists, including all of the possible combinations of the different sensing points, and finding the most suitable sensing mask could be considered an NP-hard problem. On the other hand, using fast optimization algorithms like greedy search could give non-optimal solutions [13]. To overcome these issues, the genetic algorithm (GA) [14] is used to find the best possible sensing mask in the current search space.

B. Genetic Algorithm

GA is a population-based metaheuristic algorithm based on the biological evolution process. The basic element in the optimization process of GA is genes, which normally take binary values of 0 or 1 and form a chromosome. A chromosome can be interpreted as a solution to the optimization problem. A set of these chromosomes forms a population that gets updated generation by generation. GA can find the best population according to the fitness values of the chromosomes in the population, inspired by the survival of the fittest. The fitness values can be calculated from a predefined fitness function specific to the problem. Key operators in the GA optimization process include selection, crossover, and mutation. Further details on these operators can be found in [15].

In our setup, classification accuracy served as the fitness value, and CNN served as the fitness function. At each iteration, a sensing mask was generated by the GA and applied to the training set. Then the CNN was evaluated based on the masked data and the accuracy of the model fed to the GA. This process was repeated until the GA converged. In this setup, the optimization variables to be found were the

indices of the non-zero elements in the mask. For instance, for a mask with a size of 28 by 28, optimization variables can take values between 0 and 784 as indices of non-zero elements. The number of optimization variables depends on the rate of sampling. For a 2% sampling rate, we would have 16 optimization variables. Throughout the optimization process, these variables were updated within chromosomes as binary strings.

C. Classification model

CNN is chosen as the classification model for our work. In addition to the great success of the CNNs in machine learning tasks, especially for image data, they have one advantage specifically for our method regarding the random masking of the images in the dataset. A convolutional layer in a CNN can capture local patterns and structural information in the input images [16]. Random masking is used in CNN's pre-training stage. Considering the sampling rate, a predetermined number of randomly selected image pixels are replaced with zeros during the masking process. As a result, the remaining pixels in the image retain their original positions and structural details. This makes it possible for the convolutional layers to capture the required information from the existing pixel value to predict the class of the image. The reason for this could be that the averaging property of the convolutional filters in the first layer provides an abstract presentation of the area under the filter in the image, and it is more likely that this area contains at least a few non-zero elements. This makes it possible to have a consistent representation of the input images in the receptive field of the CNN layers despite the changes in the masking of input images.

The architectures of CNN models are designed through trial and error in such a way that the models can give satisfactory classification accuracies on the datasets. We were inspired by [17] when designing CNN for the MNIST dataset. A brief overview of the model architectures is shown in Table I.

TABLE I
DETAILED ARCHITECTURES OF THE CNNS FOR THE CLASSIFICATION OF THE MNIST AND FMNIST DATASETS

Dataset	Type	Size	Numbers	Dropout
MNIST	Conv1	(3×3)	32	-
	Max1	(2×2)	-	-
	Conv2	(3×3)	64	-
	Conv3	(3×3)	64	-
	Max3	(2×2)	-	-
	Dense1	-	100	-
	Dense2	-	10	-
Fashion MNIST	Conv1	(3×3)	32	-
	Conv2	(3×3)	32	0.2
	Max1	(2×2)	-	-
	Conv3	(2×2)	64	-
	Dense1	-	200	0.2
	Dense2	-	100	0.2
	Dense3	-	10	-

D. Training CNN from scratch

Two different scenarios are considered for the inference based on the sensed data.

In the first scenario, after finding the optimum sensing mask, the whole dataset is masked and then used for the training of the CNN model. The advantage of this is that there is no need to design a new model from scratch, and the model architecture is the same as the first-step architectures in Table I.

In the second scenario, the sensed points from the image are concatenated into a 1D vector. The advantage of this scenario is the smaller size of the input data, which can accelerate the training process of the model. In this case, a 1D CNN is designed to carry out the classification. The 1D CNN comprises two consecutive convolution layers with 64 filters and a kernel size of 3, followed by a fully connected dense layer with 100 neurons [17].

III. EXPERIMENTAL RESULTS

In this section, we discuss the optimization process of the GA and its convergence. Additionally, as mentioned earlier two scenarios are considered, and an experiment is conducted on each to evaluate the performance of the proposed method. In the first experiment, the objective is to determine how much a sensing mask that is tailored to the structural information of the input data is advantageous in comparison to a random sensing mask. In the second experiment, the focus is on evaluating the effectiveness of subsampled data by feeding the non-zero pixel values to a 1D CNN for classification and assessing the impact of losing spatial relationships in the masked data. Finally, a third experiment is designed with a framework modification to enable a fair comparison of the proposed method's performance with the baseline studies in CL. The Python implementation of the experiments is available at https://github.com/m95abdollahpour/Optimization-Based-Sensing.

A. Dataset

Two benchmark datasets, MNIST [18] and Fashion MNIST [19] are used for the performance evaluation of the proposed method. MNIST contains grayscale images of handwritten digits from 0 to 9, and Fashion MNIST contains images of different clothing. The image size in both datasets is 28 by 28, and each of these datasets has 60000 images for training and 10000 images for testing.

B. Performance of the GA in the optimization framework

In this section, we will illustrate the process of searching for the optimized sensing mask using GA. The parameters of GA are chosen by trial and error as follows: Generations = 1000, chromosomes in population = 10. Before GA begins to iterate, the CNN is trained, and the optimizer will use the pre-trained CNN and utilize only the test accuracy of this model. To have a fair assessment, we do not use the test data to find the mask. Instead, 5000 images from the train set of the datasets are used as a new validation set to test the CNN model and calculate the accuracy or fitness value for the GA. The reason

for not using the entire training set is the large size of the set. In addition, as we tried the entire set, it did not add any useful information to the problem, and the GA was able to capture the structural information of the images with a small subset of the training set. Fig. 2 shows the changes in fitness value as the GA progresses generation by generation. Two cases are considered, with 5% and 10% sampling rates, and the results show the ability of the GA to converge to the sensing mask, which gives the highest classification accuracy. Intuitively, these plots indicate that the GA is able to find a sensing mask that could capture the structural information needed to discriminate data in different classes.

The accuracies for the test set are shown solely to illustrate the suitability of the mask for both test and training sets, and the testing data is not used in the optimization process.

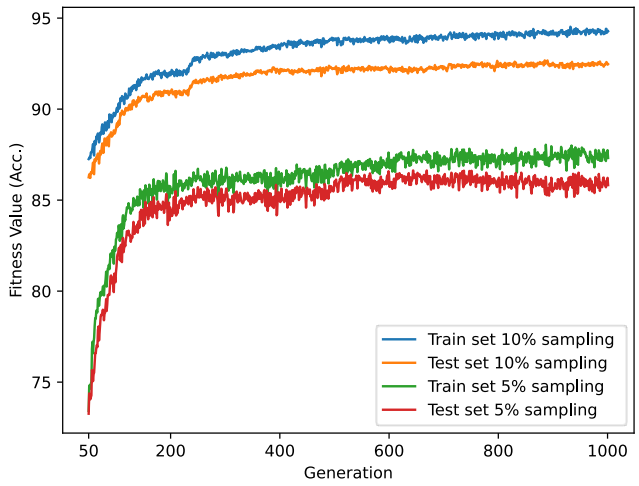

Fig. 2. Testing and training accuracies of the CNN model are shown during the optimization of the sensing mask using GA for 5 and 10 percent sampling rates of the MNIST dataset.

C. Random Sensing Versus Optimized Sensing

Random sensing is the baseline for most of the classical CS methods, and it is important to have it as a baseline for comparison with our method.

a) Experiment 1, Masking ($\mathbf{y}' = \Phi' \odot \mathbf{x}$)*:* The data here are masked images of size 28 by 28 from the MNIST and Fashion MNIST datasets. These masked images are used for training and testing of the models described in Table I. The results are shown in Table II. Each column shows the classification accuracy for a different sampling rate, and each row indicates random masking and optimized masking for the two datasets. For a fair comparison, for random masking, the experiment was repeated 10 times, and the reported results are the average over all repetitions. The 100% sampling rate shows the data without compression and provides a baseline to see how the classification models are working. 99% and 92.5% are the achieved test accuracies for the MNIST and Fashion MNIST datasets, respectively. For the lowest sampling rate of 2% or 16 pixel values, 53.3% accuracy is achieved with random masking, while the proposed method could achieve 84.9% for the MNIST dataset, which shows a considerable improvement.

As illustrated in Table II the classification accuracy is better for the MNIST dataset in comparison to the FMNIST. The reason for this is the more complex structure of the FMNIST dataset, which contains real-world fashion and clothing items. In this dataset, the images of one class might be more diverse in terms of textures, styles, and patterns, which makes it difficult to distinguish them. In spite of all of these challenging conditions, the proposed method was able to find a sensing mask suitable for this dataset to improve classification accuracy in comparison to random masking.

TABLE II
CLASSIFICATION ACCURACIES FOR DIFFERENT SAMPLING RATES IN EXPERIMENT 1

Dataset	Sampling Rate			
	100%	**10%**	**5%**	**2%**
MNIST (Random Masking)	99.0%	91.9	80.5%	53.5%
MNIST (Optimized Masking)		97.5	95.4%	84.9%
FMNIST (Random Masking)	92.5%	85.8	81.2%	71.5%
FMNIST (Optimized Masking)		87.5	84.5%	78.4%

b) Experiment 2, subsampling: This experiment is a more realistic case of CS because just the sensed pixel without redundant values are concatenated in a 1D vector. In this scenario, by taking just the non-zero values, the data loses its structure, and the spatial relations between the pixel values are not available to be used in the CNN to distinguish different classes. This data was then fed to a 1D CNN for classification.

The results are illustrated in Table III. It can be seen that the results are similar to those of experiment 1, with some tolerance. This shows the sampled points independent of their position can provide useful information to distinguish different classes. However, it should be pointed out that a simple 1D CNN architecture is used for the classification, and the structure is the same for data with different sizes corresponding to different sampling rates. This could be a reason not to expect the best possible classification result for different sampling rates. This approach could also be considered a downsampling technique based on the structural information in the dataset.

c) Experiment 3, classical sensing ($\mathbf{y} = \Phi\mathbf{x}$)*:* This experiment is conducted to evaluate the performance of the proposed framework in comparison to the baseline literature in the CL subject area and to demonstrate the resilience of the method for a different application and setup. In this experiment, the masking or Hadamard product that was used in experiment 1 is replaced by matrix multiplication. The GA is used as an optimizer to find a specific Φ that maximizes classification accuracy. For the MNIST images of size 28 by 28 and measurement rates of 5% and 1%, sensing matrices with sizes 784 by 39 and 784 by 8 are optimized via GA, respectively. The elements of Φ are chosen to take values of

TABLE III
CLASSIFICATION ACCURACIES FOR DIFFERENT SENSING RATES IN EXPERIMENT 2

Dataset	Sampling Rate			
	100%	**10%**	**5%**	**2%**
MNIST (Random Sensing)	97.5%	92.1	81.6%	53.8%
MNIST (Optimized Sensing)		97.4	95.7%	81%
FMNIST (Random Sensing)	91.4%	85.4	82.1%	74.1%
FMNIST (Optimized Sensing)		87.2	84.7%	79.7%

± 1 to limit the range for each optimization parameter and to ease the practical implementation with the digital micromirror device and single-pixel camera [20], [21].

Different classifiers, including support vector machines (SVM), K-nearest neighbors, and CNN, are used. In each iteration, a classifier was trained and tested with subsets of the training set, and test accuracy was used as a measure to determine how well the sensing matrix performs. After convergence of the GA, the train and test sets of the MNIST dataset were compressed with the optimized Φ, and the classifier was trained and tested with the compressed data one last time. The resulting test accuracies are shown in Table. IV.

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT CL METHODS ON THE CLASSIFICATION OF THE MNIST DATASET FOR DIFFERENT MEASUREMENT RATES BASED ON OVERALL ACCURACY

Method	Classifier	Sampling Rate	
		5%	**1%**
Optimization Based sensing (Proposed Approach)	SVM	95.55	**88.97**
	KNN	94.95	86.04
	CNN	95.02	85.06
Dynamic-Rate NN [22]		**95.74**	69.29
CNN based [8]		94.82	58.94
Smashed Filters [23]		46.79	36.97

In Table. IV we compared the MNIST classification accuracies of our method with baseline methods in CL. These methods use random or prefixed matrices for sampling. It can be seen that for a measurement rate of 5% the results are similar for different methods except smashed filters, which shows poor performance. However, for a lower measurement rate of 1%, a significant drop can be seen in classification accuracy in comparison to the 5% case. By decreasing the number of sampling points, random sampling is shown to be ineffective, which is one of the major disadvantages of classical CS [8]. On the other hand, optimized sampling illustrates consistent performance even for a 1% sampling rate.

In our framework, SVM was able to achieve better results in comparison to the other classifiers, especially CNN. The reason for this lies in the randomness of the output of the neural network, which is generally introduced to the model in the random initialization and dropout layers. This randomness causes inconsistency in the input of the GA, which slows down the convergence and affects the final result. Furthermore, SVM performs better for different sampling rates because it is less sensitive to input size in comparison to CNN.

IV. CONCLUSION

Motivated by the need to make CS task-specific and more efficient in the sensing part, this paper proposes a framework that finds an optimized sensing mask employing GA. This sensing mask is optimized in a way to sense the useful structural information for data discrimination in the image data to carry out classification. The results showed good improvement in comparison to the classical sampling methods in CS. In the current work, a specific task of classification is considered as the objective, and two benchmark datasets are used for performance evaluation. For future work, a deeper evaluation could be performed on the proposed method using real-world datasets. In addition, the setup could be adapted to solve a more generalized task like data reconstruction, and further investigation is needed to evaluate the performance of the framework in terms of reconstruction error.

V. ACKNOWLEDGMENT

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 497286574

REFERENCES

[1] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
[2] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, "Introduction to compressed sensing." 2012.
[3] Y. Wu, M. Rosca, and T. Lillicrap, "Deep compressed sensing," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6850–6860.
[4] A. L. Machidon and V. Pejović, "Deep learning for compressive sensing: a ubiquitous systems perspective," *Artificial Intelligence Review*, vol. 56, no. 4, pp. 3619–3658, 2023.
[5] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir, "Robust compressed sensing mri with deep generative priors," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 938–14 954, 2021.
[6] H. Yao, F. Dai, S. Zhang, Y. Zhang, Q. Tian, and C. Xu, "Dr2-net: Deep residual reconstruction network for image compressive sensing," *Neurocomputing*, vol. 359, pp. 483–493, 2019.
[7] Y. Mo, J. Huang, and G. Qian, "Deep learning approach to uav detection and classification by using compressively sensed rf signal," *Sensors*, vol. 22, no. 8, p. 3072, 2022.
[8] S. Lohit, K. Kulkarni, and P. Turaga, "Direct inference on compressive measurements using convolutional neural networks," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 1913–1917.
[9] R. Calderbank, S. Jafarpour, and R. Schapire, "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain," *preprint*, 2009.
[10] C. Mou and J. Zhang, "Transcl: Transformer makes strong and flexible compressive learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 5236–5251, 2022.
[11] J. Zhang, C. Zhao, and W. Gao, "Optimization-inspired compact deep compressive sensing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 765–774, 2020.

[12] B. F. A. B. H and P. Ramachandran, "Compressive wavelet domain deep cnn for image classification using genetic algorithm based sensing mask learning," *IEEE Access*, vol. 11, pp. 35 567–35 578, 2023.
[13] Y. Wu, Y. Liu, F. Chen, M. Zhang, and S. Ma, "Beyond greedy search: pruned exhaustive search for diversified result ranking," in *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, 2018, pp. 99–106.
[14] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.
[15] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, pp. 8091–8126, 2021.
[16] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *Mechanical systems and signal processing*, vol. 151, p. 107398, 2021.
[17] J. Brownlee, "How to develop a cnn for mnist handwritten digit classification," *Machine Learning Mastery*, 2019.
[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[19] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.
[20] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 83–91, 2008.
[21] J. Bacca, L. Galvis, and H. Arguello, "Coupled deep learning coded aperture design for compressive image classification," *Optics express*, vol. 28, no. 6, pp. 8528–8540, 2020.
[22] Y. Xu, W. Liu, and K. F. Kelly, "Compressed domain image classification using a dynamic-rate neural network," *IEEE Access*, vol. 8, pp. 217 711–217 722, 2020.
[23] S. Lohit, K. Kulkarni, P. Turaga, J. Wang, and A. C. Sankaranarayanan, "Reconstruction-free inference on compressive measurements," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 16–24.