ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230337

SENA: Similarity-Based Error-Checking of Neural Activations

Raul Sena Ferreira [1;*], Joris Guerin[2], Jeremie Guiochet[3] and Helene Waeselynck[4]

Abstract. In this work, we propose SENA, a run-time monitor focused on detecting unreliable predictions from machine learning (ML) classifiers. The main idea is that instead of trying to detect when an image is out-of-distribution (OOD), which *will not always result in a wrong output*, we focus on detecting if the prediction from the ML model is not reliable, which *will most of the time result in a wrong output*, independently of whether it is in-distribution (ID) or OOD. The verification is done by checking the similarity between the neural activations of an incoming input and a set of representative neural activations recorded during training. SENA uses information from true-positive and false-negative examples collected during training to verify if a prediction is reliable or not. Our approach achieves results comparable to state-of-the-art solutions without requiring any prior OOD information and without hyperparameter tuning. Besides, the code is publicly available for easy reproducibility at https://github.com/raulsenaferreira/SENA.

1 Introduction

State-of-the-art OOD detectors were empirically demonstrated to yield a considerable number of false positives (i.e., correct predictions rejected by the monitor) [32, 15, 8] and false negatives (i.e., ML model prediction errors not detected by the monitor). On the one hand, a high number of false positives have a negative impact on the performance of the ML model for ID data. One of the main reasons is that current monitors treat OOD data as data that the ML model should avoid. Hence, such monitors tend to be activated for all OOD data. However, except for new objects, not all OOD data lead to a failure in the ML model [12]. On the other hand, false negatives tend to decrease the safety of an ML-based system [10] as unsafe data instances are not detected. They are mostly due to the fact that ML models still have to deal with possible wrong predictions for ID data as well, and recent works theoretically demonstrated that wrong ML predictions for both ID and OOD data may be linked to the same phenomena of model misestimation problem [37].

In addition, recent works showed that OOD detectors solely based on uncertainty [31], generative models [37] and its densities [22], cannot safely guarantee that the OOD detection is correct, except for specific combinations of datasets, which highlights the importance of approaches that can detect when the ML model can correctly deal with the data, independently if it is ID or OOD.

* 1) Corresponding Author. LAAS-CNRS: raulsenaferreira@gmail.com.
2) LAAS-CNRS, IRD, Université de Montpellier: joris.guerin@ird.fr
3) LAAS-CNRS, Université de Toulouse: jeremie.guiochet@laas.fr
4) LAAS-CNRS, Université de Toulouse: helene.waeselynck@lass.fr

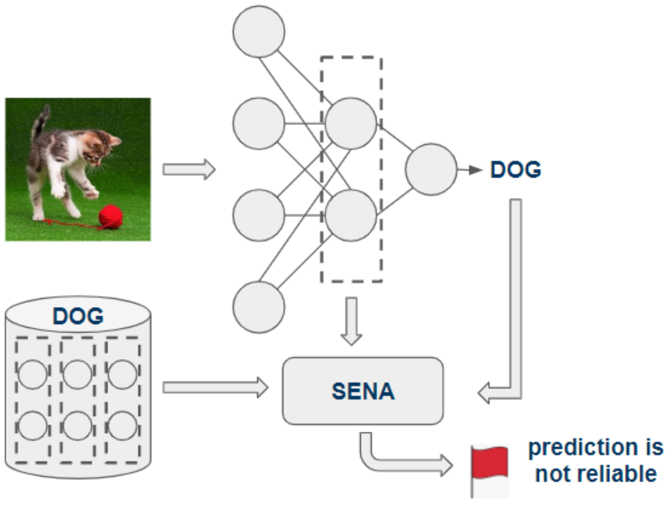


Figure 1: **Similarity-based Error-checking of Neural Activations.** SENA checks neural activations that may lead to erroneous ML model predictions by comparing them to representative neural activations from the training set. The motivation for using correct and incorrect examples from the training dataset is to help the monitor to deal with not only OOD data but also incorrectly classified ID data.

Therefore, we propose SENA, a monitor that uses a similarity-based error-checking of neural activations to monitor incorrect predictions at runtime. As illustrated in Figure 1, SENA uses the activation functions of the ML model combined with its prediction to monitor if an image can lead to a unreliable prediction at runtime.

This work is organized as follows: In Section 2, we point to the related work regarding OOD detectors in the literature. In Section 3, we detail our approach by showing how our monitor is built with training data, how it automatically chooses thresholds, and how it performs the ML monitoring at runtime. In Section 4, we show the experiments, and the analysis of our results using classification metrics and safety ones. Finally, in Section 5, we present our final considerations, limitations, and possible improvements to our method.

2 Related Work

Related works usually monitor an ML model at runtime through one of three types of source of information: 1) the input; 2) the model's internal values; 3) the outputs from the last layer.

An example of input monitoring is to adversarially train an ML model to reconstruct a noisy image to an image similar to a previously known class [30]. It means that a new image arrives at runtime, and this ML model tries to reconstruct this image to a known one from the set of known classes previously seen in the training phase. The actual image is considered OOD if there is a considerable difference between the reconstructed image and the actual image. Another recent approach uses reverse knowledge distillation from a teacher-student (T-S) model's one-class embedding [4].

Regarding the model internals monitoring, we can mention model logits [19, 13] or activation function patterns such as outside-the-box [16]. Such a strategy inspects the output of an activation function during runtime after a new image passes through the ML model. If the pattern is different from what is known during training, it raises the alarm considering this input as a novel one. Another recent approach uses a threshold over the k-nearest distances between the embedding of test input and the embeddings of the training set [33].

Finally, regarding the model outputs monitoring, the most common approach is to monitor confidence values [18] by applying techniques that decrease the DNN confidence values on novel inputs. Other approaches also use thresholds on the model outputs to take a decision such as max softmax probability [13], and energy [26].

SENA differs from related works in two main aspects: 1) It uses information from both true positives (TP) and false negatives (FN) collected during the model training, which is a unique feature and allows SENA to automatically choose thresholds that are better adapted for a specific dataset. 2) It uses a statistical concept known as core support extraction combined with a simple distance metric, which makes our approach less sensitive to ID outliers. Another aspect of SENA is its independence from prior OOD data or hyperparameter tuning, such as rejection thresholds or the number of clusters [16].

3 Similarity-based Error-checking of Neural Activations

SENA is built as a one-class monitor, such as one-class support vector machines [24], class-reconstruction-based methods [30], or methods based on class-activation-function patterns [16]. Since not all classes need to be monitored (e.g., 100% of accuracy), we monitor by class. Therefore, an independent monitor is produced for each class of interest from the monitored model, and the ML prediction given at runtime is used to choose which monitor will be used. We do not use information from OOD data or other ID classes since at runtime one cannot assess which data are exposed to the ML.

3.1 Monitor building

As illustrated in Figure 2, SENA monitor is built in four steps:

1. **Training features extraction**. We feed the trained model with all the training images labeled as c. Then, we extract the neural activation vectors from these images and store them in two distinct sets: S_{TP}, containing all the feature vectors corresponding to correctly classified data (i.e., predicted as c), and S_{FN}, containing the incorrectly classified feature vectors. S_{TP} represents the features corresponding to the TP for class c, while S_{FN} contains the FN.
2. **Core support extraction**. As the monitored neural network is mostly correct on its training data, we usually have $|S_{TP}| >> |S_{FN}|$, and the number of samples in S_{TP} can quickly become very large. Hence, the second step of SENA intends to reduce the size of S_{TP} by selecting a small subset of representative samples. To do so, we apply a core support extraction algorithm [11]. Core support samples are the ones that contain the most informative characteristics of the underlying distribution of the set, i.e., they can represent most of the entire original distribution. Besides reducing the size of S_{TP}, core support extraction helps SENA to filter outliers. A core support extraction algorithm has two steps:
   - *Weighting*: In this step, the probability distribution from which elements in S_{TP} were drawn is estimated using any multivariate density estimator, such as Gaussian mixture models (GMM) [29] or kernel density estimation (KDE) [38]. In this work, we chose KDE for its flexibility since it does not require knowing the parameters of the distribution to be fitted. In this work, the KDE algorithm uses the Gaussian kernel, and the bandwidth value is automatically tuned using a grid search algorithm. To choose the Gaussian Kernel, we looked at the distributions of distances between FN and TP for different datasets (CIFAR-10 on Fig. 3) and they appeared Gaussian. Hence, we train the KDE algorithm with S_{TP} and use the same trained estimator to calculate the probability density function value (PDF) for each sample in S_{TP}. Thus, each sample in S_{TP} is weighted based on its PDF. Worth mentioning that KDE is not applied to FN due to its low occurrence in the training set. Our method is a redundant mechanism to handle potential misclassifications and is not intended to replace a well-trained ML model, which ideally has few FN.
   - *Selection*: Then, we select the k densest samples in S_{TP}, i.e., the feature vectors corresponding to the k highest PDF values attributed for S_{TP}. A feature vector h with a high PDF value means that the sample is more likely to be drawn from the distribution learned by the KDE. Hence, these samples are the ones that best represent the original distribution of S_{TP}. The selected samples are stored in a set R_{TP}, containing the k most representative elements of S_{TP}. The choice of k represents a trade-off between accuracy in the training set and the amount of memory required to store the samples to be used at runtime. Previous works showed that such density estimation selection can discard up to 90% of the original dataset without a drastic performance drop [9]. Therefore, in this work, we chose $k = 10\%$ of the training set. Besides, increasing the value of k should not compromise performance, as it allows SENA to use more information from the distribution. Due to space constraints, we did not conduct a sensitivity analysis for k.
3. **Distances computation**. Next, SENA uses the set of true positive representative activations R_{TP} and the set of false negative activations S_{FN}, to calculate two sets of distances: 1) the set D_{TP-TP} contains all the distances among elements of R_{TP}, and 2) the set D_{TP-FN} contains all the distances between elements of R_{TP} and elements of S_{FN}. In this work, we use Euclidean distances between vectors.
4. **Automatic threshold computation**. Finally, the overlapping region between the distributions of D_{TP-TP} and D_{TP-FN} are analyzed. A threshold α belonging to this region is estimated by the algorithm (more details in Subsection 3.2). This threshold is used at runtime to determine if a model prediction is reliable or not.

After iterating over all training images labeled as c, we produce a SENA monitor for this class, composed of a set of representative true positive feature vectors R_{TP}, and a distance threshold α^c. By repeating this process for every possible output class of the monitored neural network, we can build a complete SENA monitor.

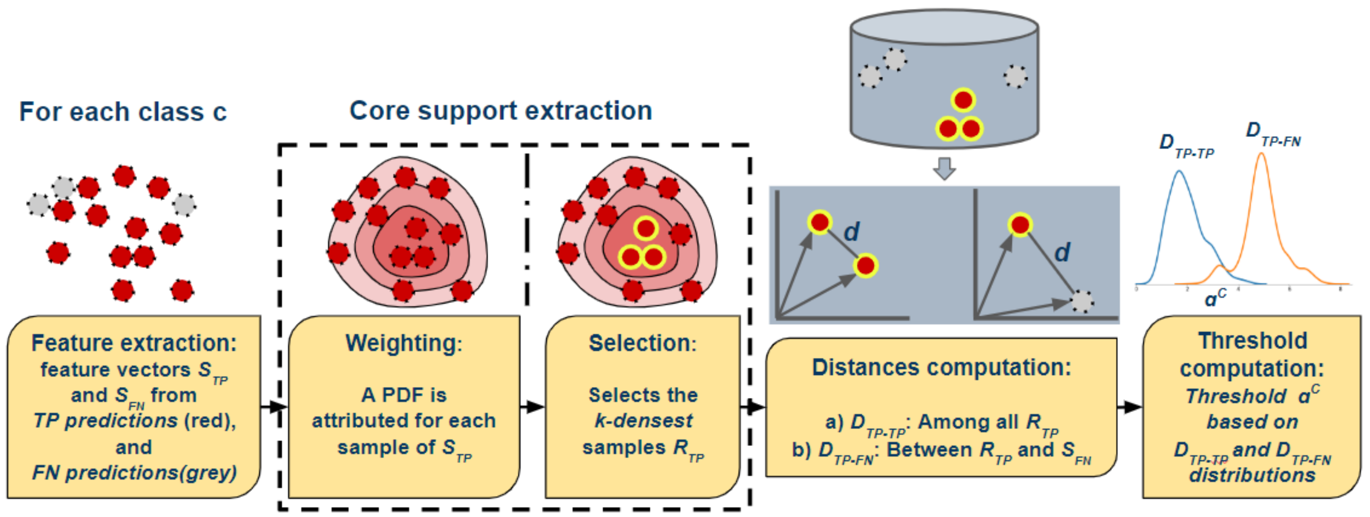

For each class c
Core support extraction
Feature extraction: feature vectors S_{TP} and S_{FN} from TP predictions (red), and FN predictions(grey)
Weighting: A PDF is attributed for each sample of S_{TP}
Selection: Selects the k-densest samples R_{TP}
Distances computation: a) D_{TP-TP}: Among all R_{TP} b) D_{TP-FN}: Between R_{TP} and S_{FN}
Threshold computation: Threshold a^C based on D_{TP-TP} and D_{TP-FN} distributions
d
d
D_{TP-TP}
D_{TP-FN}
a^C

Figure 2: **SENA monitor building.** 1) A core support extraction algorithm is applied for weighting and selection of the most representative true positive neural activations to be stored and used at runtime. 2) It calculates the neural activation similarities among representative true positives, and also between true positives and false negatives.

3.2 Automatic threshold selection

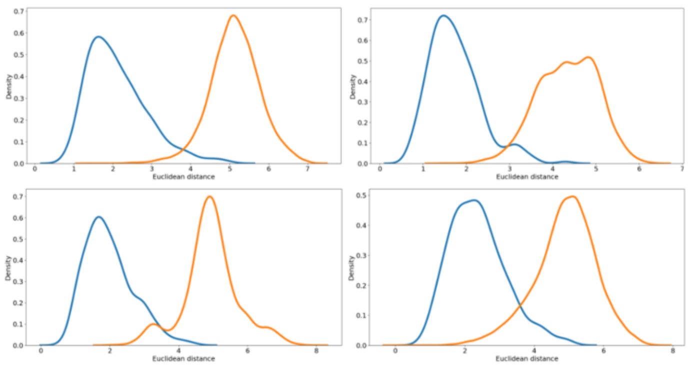


Figure 3: **Activation vectors distances of the first 4 classes of CIFAR-10.** The x-axis represents the Euclidean distance values, and the y-axis represents the distribution density (e.g., amount of points). The blue line represents the distance distribution between true positives, and the orange line represents the distance distribution between true positives against false negatives.

As mentioned earlier, for a specific class, SENA generates the threshold α by analyzing the distributions of D_{TP-TP} and D_{TP-FN}. To understand how D_{TP-TP} and D_{TP-FN} are distributed, Figure 3 illustrates the densities of D_{TP-TP} (in blue) and D_{TP-FN} (in orange) for the first four classes of the CIFAR-10 dataset. Activation features were extracted from a ResNet model.

In the CIFAR-10 dataset, we can observe overlapping areas between D_{TP-TP} and D_{TP-FN}. It indicates that monitors that rely on activation function vectors might experience false positives (i.e., rejecting valid predictions) coming from the regions where there exist outliers, that is, FN with a higher distance density in the original distribution than TP. A threshold will be chosen considering the existence of such outliers to decide when flagging a wrong prediction.

To decrease the human supervision from the daunting task of choosing a threshold, we apply a simple strategy of varying the threshold α based on the overlapping area of the distributions.

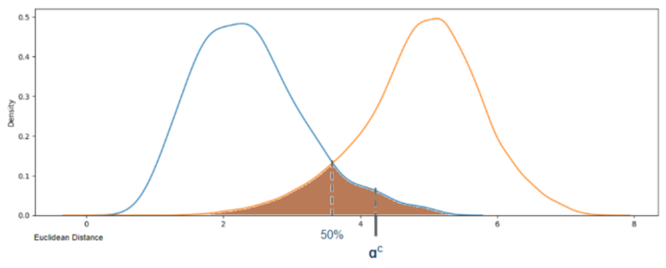


Figure 4: **Flexible thresholds.** SENA threshold for TP and FN.

As illustrated in Figure 4, the threshold α is set to $\mu + \sigma$, where μ and σ are respectively the mean and standard deviation of the intersection set, i.e., all the elements in D_{TP-TP} and D_{TP-FN} that are greater than $\min(D_{TP-FN})$ and smaller than $\max(D_{TP-TP})$. The standard deviation is very useful for measuring the data dispersion in the distributions regardless of whether data are normally distributed. This allows α to assume different values depending on how the data in S_{TP} and S_{FN} are spread. Therefore, smaller overlapping regions α (uncertainty regions) between TP / FN activation functions mean better separation between both distributions, leading to less uncertainty for the monitor to decide if an activation function comes from a reliable prediction (TP) or not (FN). This is important since the literature shows a high amount of false detections from activation function-based approaches due to activation functions that fall in that regions. If no FN exists for a specific class, the value of α is the maximum value of D_{TP-TP}, and if an average distance $\bar{d}$ falls outside the range of D_{TP-TP} it is considered an unreliable prediction.

3.3 Monitor at runtime

For a given class c, SENA starts with two artifacts previously calculated in the monitor building process: a threshold α^c, and a set of TP representatives (R^c_{TP}). As illustrated in Figure 5, SENA verifies if the incoming prediction might be unreliable. It extracts the neural activation vector h from a particular neural-network layer during the ML model prediction on an image X. This work takes it from the penultimate layer since it is the most informative one [16]. Then,

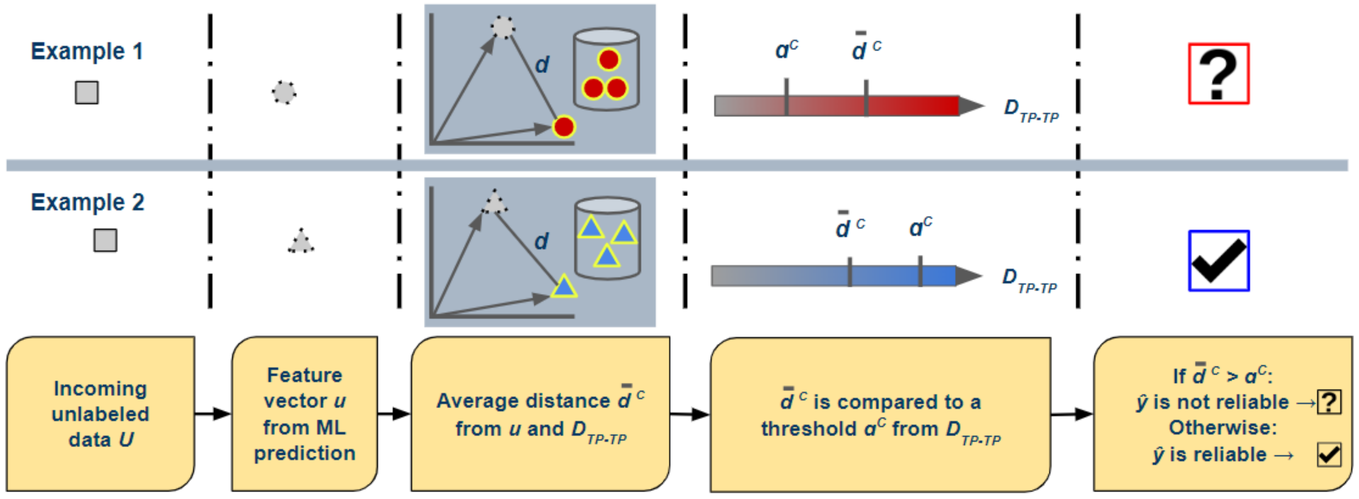

Example 1
Example 2
d
α^c
$\bar{d}^c$
$D_{TP\text{-}TP}$
?
Incoming unlabeled data U
Feature vector u from ML prediction
Average distance $\bar{d}^c$ from u and $D_{TP\text{-}TP}$
$\bar{d}^c$ is compared to a threshold α^c from $D_{TP\text{-}TP}$
If $\bar{d}^c > \alpha^c$: $\hat{y}$ is not reliable →
Otherwise: $\hat{y}$ is reliable →

Figure 5: **Two examples of SENA monitoring a class at runtime.** The average distance $\bar{d}^c$ is calculated by comparing the feature vector from the incoming image to D_{TP-TP}: 1) SENA triggers an alarm since $\bar{d}^c > \alpha^c$ (unreliable prediction); 2) SENA does not interfere in the ML prediction since $\bar{d}^c \leq \alpha^c$ (reliable prediction).

SENA calculates the average distance $\bar{d}^c$ between h and elements of R^c_{TP}. If $\bar{d}^c$ is higher than α^c, then SENA considers the prediction unreliable, otherwise, it considers the prediction reliable.

Finally, regarding the computational costs, SENA only computes the average Euclidean distance between the incoming and representative vectors. The memory cost is limited to a vector $n * 10\%$ of the data, where n is the number of neurons in the monitored layer.

4 Experiments

Our experiments are conducted on several image classification tasks. Besides, since we want our experiments to be as close as possible to real-world scenarios, we set some important constraints, which can lead to different results from what has been reported in the literature previously. We use the concept of out-of-model-scope (OMS) introduced in this section and in [12]. Here are the details:

- **OOD vs OMS**. In the OOD evaluation scenario, the monitor has to trigger an alarm when an OOD image arrives. However, in our experiments, the goal is to detect unreliable predictions instead of OOD data and we consider that a monitor is correct if it rejects wrong predictions, independently of their ID/OOD status. This evaluation is also known as out-of-model-scope detection (OMS).
- **Stream of random images**. We consider a scenario in which the images are randomly given to the ML model in a streaming fashion, that is, the data stream will feed one image at a time to the ML model. Therefore, approaches based on time series, or from batches of data to perform the detection are not considered.
- **No hyperparameter tuning**. Datasets are divided into ID data (training/test) and OOD data (test), and the monitors are not allowed to use information or assumptions from test data to fine-tune rejection thresholds or their detection strategies. That is, monitors must be built using only the training data since in a real situation, it is unrealistic to consider that we know which kind of data will be fed to the ML model at runtime. Hence, related works that use information from test data (e.g., [25]) are not considered.
- **No optimal performance metrics**. Several works in the literature test their solutions on the entire dataset with different rejection thresholds and display the results corresponding to the best results according to some metric (e.g., F1, ROC curves, etc) [12]. Although this approach can be relevant to demonstrate the optimal performance of a detector over all possible threshold choices, in a real-world scenario, one cannot perform such threshold optimization using the test data. Therefore, in this work, all analyses of the results are performed using objective metrics that do not depend on multiple runs using the test data. Thus, we decrease bias and conflicts between parameter fine-tuning and observed results.

4.1 Experiment settings

We perform 38 experiments using three popular image datasets as ID, of which two are RGB: CIFAR-10 [21], and SVHN [27]; and one is grayscale: MNIST [6]. For each ID dataset, we split it into train and test, in which the train is used to fit the monitors, and the test is used to evaluate the monitor under ID data. Except for novelty tasks, the ID test data is used to generate the OOD datasets through image transformations. Below, we present the tested OOD scenarios:

- **16 class novelty experiments**:
  1. For CIFAR-10 we use CIFAR-100 [21], GTSRB [17], SVHN, LSUN [36], Fractal [14], and TinyImageNet (subset of ImageNet [5]) to represent the novel data.
  2. For SVHN we use CIFAR-10, CIFAR-100, Fractal, GTSRB, LSUN, and TinyImageNet.
  3. For MNIST [6], we use Fashion-MNIST [35], E-MNIST (letters) [2], American sign language (ASL) MNIST [34], and Simpsons MNIST [1].
- **16 distributional shift experiments**: for each RGB ID dataset, we apply eight image transformations from the AugLy library [28]: brightness (factor=5), blur (radius=4), pixelization (ratio=0.1), shuffled pixels (factor=0.3), contrast (factor=9), opacity (level=0.2), rotate (degrees=25), and saturation (factor=17).
- **6 adversarial attack experiments**: for each RGB ID dataset, we apply three adversarial attacks from Torchattacks [20], with default parameters: fast gradient sign method (FGSM), DeepFool, and projected gradient descent (PGD).

Once the scenarios are set, we choose two different deep learning architectures to test with the monitors: the ResNet models from [23] for RGB images, and a custom CNN [6] model for grayscale images.

We compare our proposal to four related works: outside-the-box (OTB), max softmax probability (MSP), max logits, and energy.

An important note is that except for SENA and OTB, the monitors tested in our experiments require selecting a rejection threshold on the monitoring scores. The best strategy for choosing thresholds is not addressed in this work. However, in our experiments, we conducted two simple steps: 1) We fit the monitors using the same training data as the ML model. 2) We choose a threshold for the fitted monitors based on the best Matthews Correlation Coefficient (MCC) value for detecting correct/incorrect ML predictions from the training set. Such threshold tuning based on the training set is challenging but realistic. The oracle for the monitor evaluation is as follows:

- TP: the monitor is triggered and the model prediction is wrong.
- TN: the monitor is not triggered and the model prediction is right.
- FP: the monitor is triggered and the model prediction is right.
- FN: the monitor is not triggered and the prediction is wrong.

For the metrics, we apply four classification metrics for imbalanced datasets: Matthews Correlation Coefficient (MCC), false positive rate (FPR), false negative rate (FNR), and macro-F1 scores regarding the monitor's output. For the OOD categories with more than 15 experiments, it is possible to perform Wilcoxon signed-rank tests with statistical guarantees [7] for each pair of tested methods. Thus, we compare if the methods are significantly different from each other across multiple experiments in a specific OOD category [3].

4.2 Results

Below we show the results separated by novelty class, distributional shift, and adversarial attack. The best results are written in **bold**.

4.2.1 Novelty class

Table 1: **MCC results for novelty class**: organized by ID - OOD.

Experiments	OTB	MSP	Logit	Energy	SENA
CIFAR 10 - CIFAR 100	**0.41**	0.27	0.29	0.28	0.38
CIFAR 10 - Fractal	**0.76**	0.54	0.42	0.35	0.71
CIFAR 10 - GTSRB	**0.66**	0.34	0.28	0.25	0.55
CIFAR 10 - LSUN	0.78	0.52	0.44	0.38	**0.80**
CIFAR 10 - SVHN	0.48	0.33	0.23	0.17	**0.60**
CIFAR 10 - T. Imagenet	0.69	0.51	0.43	0.38	**0.75**
SVHN - CIFAR 10	0.81	0.29	0.45	0.51	**0.85**
SVHN - CIFAR 100	0.80	0.29	0.45	0.52	**0.84**
SVHN - Fractal	**0.85**	0.42	0.53	0.58	0.77
SVHN - GTSRB	0.67	0.24	0.33	0.38	**0.72**
SVHN - LSUN	0.78	0.38	0.53	0.59	**0.79**
SVHN - T. ImageNet	0.81	0.40	0.54	0.60	**0.82**
MNIST - ASL MNIST	**0.93**	0.87	0.60	0.90	0.90
MNIST - EMNIST	**0.67**	0.30	0.46	0.37	0.60
MNIST - F. MNIST	**0.74**	0.51	0.50	0.63	0.70
MNIST - S. MNIST	**0.91**	0.82	0.64	0.88	0.83
Average rank	**1.5**	4	3.7	3.2	**1.5**

The MCC results corresponding to novelty OOD scenarios are reported in Table 1. Looking at the MCC results and the average rank, both SENA and OTB obtained the best results to avoid wrong predictions in the novelty class scenario. All the other three methods (MSP, Max logit, and energy) performed similarly between themselves.

These results indicate that deploying methods along with ML models for this scenario is a promising research direction. Besides, the methods based on activation functions (OTB, SENA) provided the best MCC results among the methods. To understand better such behavior, we illustrate a positive and negative analysis in Figure 6.

In Figure 6a, the average false negative rate of activation-function-based methods is way lower than the other methods. This shows a better capacity of such a strategy in identifying data that is OMS. The figure also shows that the softmax probability can be very useful in the verification of possible false positives in the novelty class scenario. As can be seen in the Sub-figure 6b, on average, the macro-F1 values for OTB and SENA show stable and superior results.

Finally, we perform a statistical analysis of the results. Blue boxes mean the MCC values measured through the experiments, from two paired algorithms, were not significantly different from each other. On the contrary, p-values lower than 0.05 indicates that two paired algorithms have a statistical difference between their results for an OOD category. Figure 7 shows that SENA and OTB are significantly different than the other methods. Table 1 shows that OTB and SENA obtained the best average ranks. The combined analysis indicates that OTB and SENA were the best methods in the novelty class scenario.

4.2.2 Distributional shift

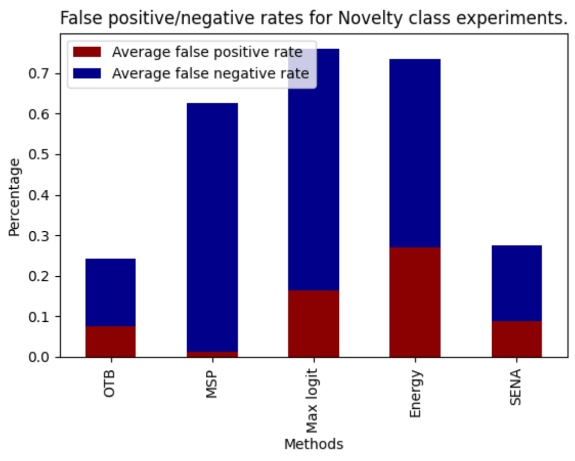
Table 2: **MCC for distributional shift**: ID data - Transformation.

Experiments	OTB	MSP	Logit	Energy	SENA
CIFAR 10 - Blur	**0.65**	0.20	0.12	0.09	0.57
CIFAR 10 - Brightness	0.32	0.33	0.38	0.38	**0.48**
CIFAR 10 - Contrast	0.42	0.39	**0.44**	**0.44**	0.36
CIFAR 10 - Opacity	0.39	**0.46**	**0.46**	0.44	0.43
CIFAR 10 - Pixelization	**0.62**	0.21	0.14	0.10	0.58
CIFAR 10 - Rotate	**0.43**	0.37	0.36	0.34	0.38
CIFAR 10 - Saturation	**0.42**	0.36	**0.42**	0.41	0.35
CIFAR 10 - Shuffled pixels	0.37	0.33	0.35	0.33	**0.56**
SVHN - Blur	0.58	0.24	0.35	0.40	**0.60**
SVHN - Brightness	**0.72**	0.16	0.19	0.21	0.55
SVHN - Contrast	0.50	0.46	0.52	**0.53**	0.28
SVHN - Opacity	**0.55**	0.31	0.42	0.47	0.29
SVHN - Pixelization	0.66	0.20	0.31	0.36	**0.67**
SVHN - Rotate	**0.45**	0.31	0.40	0.43	0.27
SVHN - Saturation	**0.57**	0.32	0.40	0.44	0.34
SVHN - Shuffled pixels	**0.53**	0.29	0.39	0.43	0.39
Average rank	**1.6**	3.8	2.6	2.6	2.5

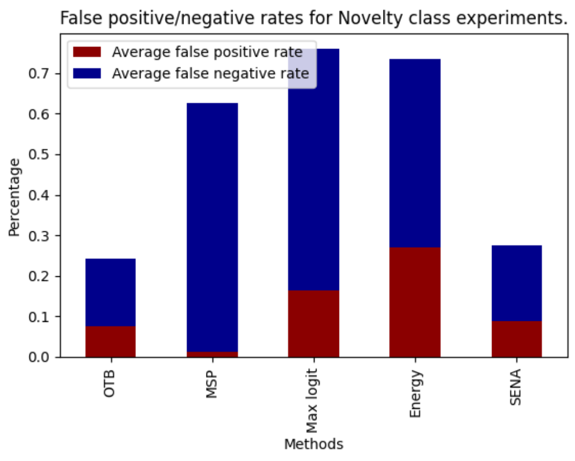
The parameter values for the distributional shift transformations (e.g., level of blur, opacity, etc) were chosen such that the ML alone achieves MCC results between 0.2 and 0.8. It means that the ML model is placed in a challenging scenario that justifies the use of a monitor since it is not capable of having strong results (MCC < 0.8) while having better results than a random classifier (MCC > 0.2).

According to Table 2, all methods, except the one based on softmax, have comparable results, in which SENA achieved the best MCC results four times. The overall performance in the distributional shift scenario was worse than in the novelty class scenario. This decrease in the results indicates that activation function methods are better for novelty and less good for distributional shift.

We can see in Figure 8a the huge amount of false positives yielded by OTB and SENA since the activation function-based methods tend to have difficulties in distributional shift scenarios. On the other hand, MSP, Max logit, and energy have lower FP rates since their thresholds are entirely based on the performance of the ML model over



(a) FPR / FNR for novelty class experiments.



(b) F1 for novelty class experiments.

Figure 6: **False positives and negative analysis for novelty class experiments.**
Figure 6a shows a lower number of false positives and false negatives for OTB and SENA.
Figure 6b illustrates a good balance and stability for OTB and SENA regarding recovery and precision over new classes.

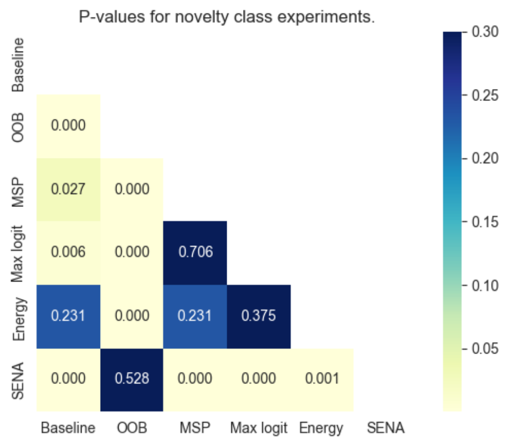


Figure 7: Novelty class experiments.

the ID data. Thus, since ID data support overlaps with OOD data, these methods tend to have lower FP rates. However, the amount of FN is as high as the other methods. Such phenomenon leads to not-so-good overall macro-F1 results (Figure 8b). Finally, Figure 9 combined with the average ranks in Table 2 shows that OTB was the best method, while the other methods had similar performance.

4.2.3 *Adversarial attacks*

The tested methods were not originally developed to detect adversarial images, and to perform such detection requires particular defense strategies for each type of attack which is an open research problem. However, we find it relevant to experiment on these three adversarial scenarios to give possible insights for future work and to complement the analysis of all OOD categories mentioned early on in this work.

Table 3 shows the MCC results in the adversarial attack scenario in which the monitoring strategy applied by the two activation-function-based methods was consistently better than the others.

Table 3: **MCC results for adversarial attack**: ID data - Attack.

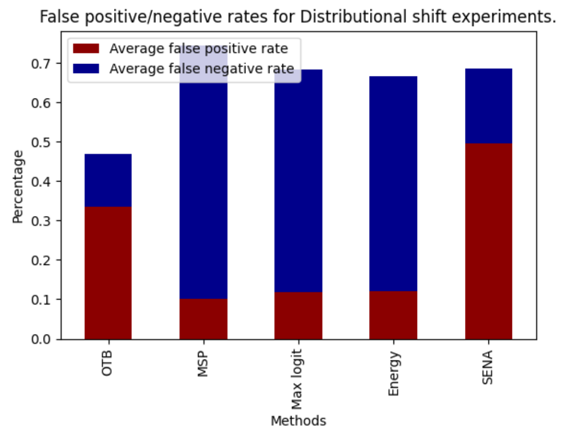
Experiments	**OTB**	**MSP**	**Logit**	**Energy**	**SENA**
CIFAR 10 - Deep fool	0.39	0.16	0.14	0.20	**0.47**
CIFAR 10 - FGSM	0.48	0.44	0.45	0.43	**0.62**
CIFAR 10 - PGD	0.20	0.06	0.20	0.21	**0.68**
SVHN - Deep fool	**0.50**	0.13	0.05	0.12	0.48
SVHN - FGSM	0.63	0.24	0.33	0.37	**0.38**
SVHN - PGD	**0.54**	0.09	0.15	0.38	0.50
Average rank	1.5	4.3	4	3.3	**1.2**

Figure 11 shows good macro-F1 values for OTB and SENA. Such results reflect precision and recall with equal importance. It means that even though their results are not bad when looking for both ID and OOD, it will depend on the amount of OOD data exposed at runtime. Such phenomenon is better understood by seeing Figure 10. SENA obtained the lowest amount of FN among the tested methods.

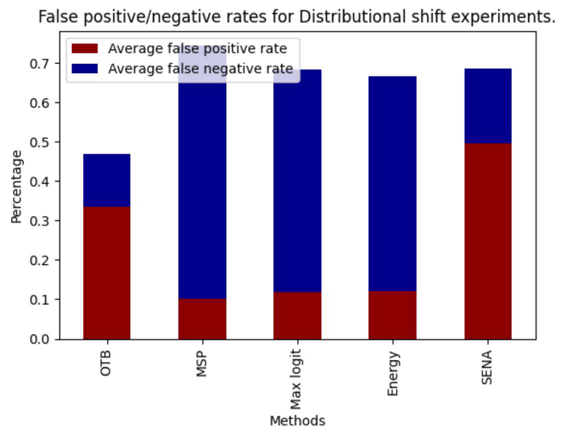
5 Conclusion

In this work, we propose SENA, a similarity-based error-checking of neural activations approach to monitor incorrect predictions at runtime. SENA uses a statistical weighting and filtering method to select the minimal set of representative samples from the training set combined with a simple similarity algorithm such as a Euclidean distance. The SENA threshold is chosen automatically based on the distributions of TP and FN. Our main findings are:

- *Neural activation vectors from TP and FN can be very similar:* most related works rely on the assumption that incorrect classifications can be spotted by inspecting all neural activation vectors that are not similar to the distribution of correct ones. However, this is not entirely true for ID data or other types of OOD data. We showed that feature vectors of TP and FN can be similar to each other inside ID data. The necessity of filtering the right data to better separate TP and FN data led us to the second finding.
- *Not all training data is needed to build a monitor:* just a part of TP samples is needed to extract the necessary information to perform



(a) FPR/FNR for distributional shift.



(b) F1 for distributional shift experiments.

Figure 8: **False positives and negative analysis for distributional shift experiments.**
Figure 8a shows a high rate of false negatives for all methods but a low rate for those not based on activation functions.
Figure 8b shows that OTB still achieves a better balance between precision and recovery despite yielding more false positives.

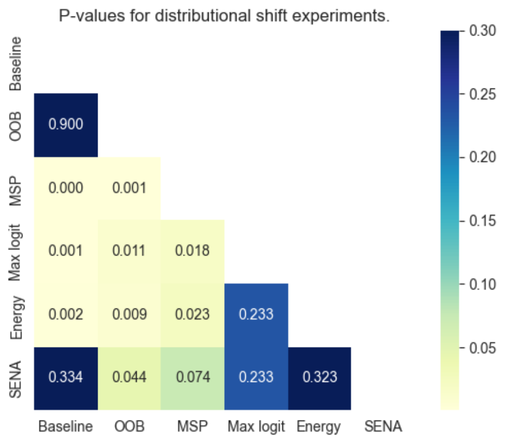


Figure 9: Distributional shift experiments.

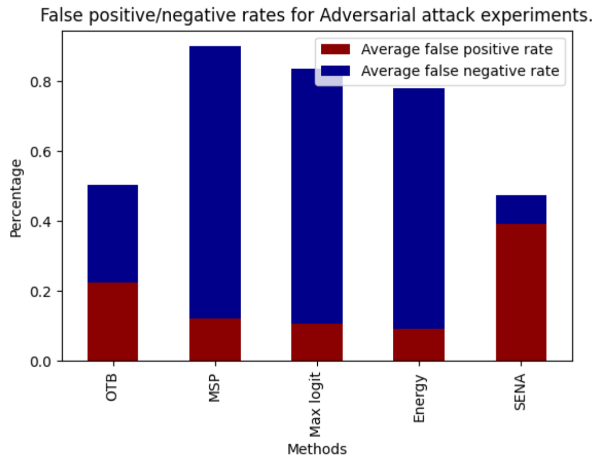


Figure 10: FPR/FNR for adversarial attack.

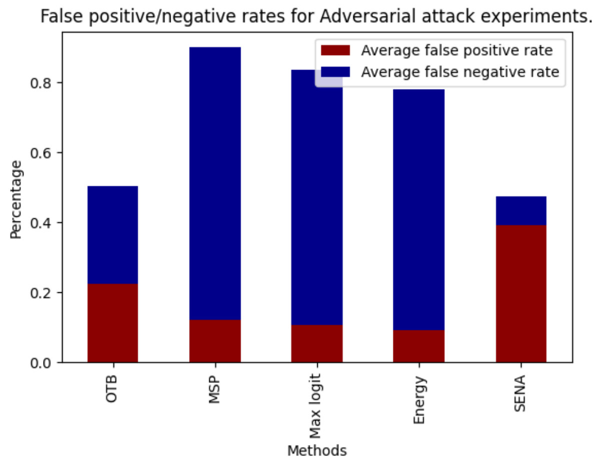


Figure 11: F1 for adversarial attack experiments.

monitoring of a class. As shown, there are several outliers present in the training data.By outliers, we mean activation function values from FN that are similar to TP, sometimes even more similar than other TP compared between themselves. Hence, a filtering mechanism, such as the core support extraction algorithm applied in this work, can help to exclude samples that contribute negatively to make a better separation between FN and TP. However, it can be hard to select different thresholds for different ID datasets. This showed the importance to investigate further sources of information, which led us to the third finding.

- *Extracting information from both TP and FN is advantageous:* the use of both, when possible, allowed us to check different boundaries for the monitor's thresholds by checking the distribution of FN (e.g., error-prune ID data) instead of just checking what is normal (e.g., ID data that the ML model gives correct classification).

Acknowledgements

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 812.788 (MSCA-ETN SAS). This publication reflects only the authors' view, exempting the European Union from any liability: http://etn-sas.eu/. This research also has benefited from the AI Interdisciplinary Institute ANITI, funded by the French "Investing for the Future – PIA3" program under the Grant Agreement No ANR-19-PI3A-0004.

References

[1] Alexandre Attia. Sinpson mnist., 2018.
[2] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik, 'Emnist: Extending mnist to handwritten letters', in *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, (2017).
[3] Janez Demšar, 'Statistical comparisons of classifiers over multiple data sets', *Journal of Machine learning research,*, **7**(Jan), pp. 1–30, (2006).
[4] Hanqiu Deng and Xingyu Li, 'Anomaly detection via reverse distillation from one-class embedding', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2022).
[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, 'Imagenet: A large-scale hierarchical image database', in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, (2009).
[6] Li Deng, 'The MNIST database of handwritten digit images for machine learning research [best of the web]', *IEEE Signal Processing Magazine*, **29**(6), 141–142, (2012).
[7] Alok Kumar Dwivedi, Indika Mallawaarachchi, and Luis A Alvarado, 'Analysis of small sample size studies using nonparametric bootstrap test with pooled resampling method', *Statistics in medicine*, **36**(14), 2187–2205, (2017).
[8] Raul Sena Ferreira, Jean Arlat, Jérémie Guiochet, and Hélène Waeselynck, 'Benchmarking safety monitors for image classifiers with machine learning', in *2021 IEEE 26th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 7–16. IEEE, (2021).
[9] Raul Sena Ferreira, Bruno MA da Silva, Wendell Teixeira, Geraldo Zimbrao, and Leandro Alvim, 'Density-based core support extraction for non-stationary environments with extreme verification latency', in *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 181–187. IEEE, (2018).
[10] Raul Sena Ferreira, Joris Guérin, Jérémie Guiochet, and Helene Waeselynck, 'Simood: Evolutionary testing simulation with out-of-distribution images', in *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 68–77. IEEE, (2022).
[11] Raul Sena Ferreira, Geraldo Zimbrão, and Leandro GM Alvim, 'Amanda: Semi-supervised density-based adaptive model for non-stationary data with extreme verification latency', *Information Sciences*, **488**, 219–237, (2019).
[12] Joris Guérin, Kevin Delmas, Raul Sena Ferreira, and Jérémie Guiochet, 'Out-of-distribution detection is not all you need', in *Proceedings of the AAAI conference on artificial intelligence (2023)*, (2023).
[13] Dan Hendrycks and Kevin Gimpel, 'A baseline for detecting misclassified and out-of-distribution examples in neural networks', *Proceedings of International Conference on Learning Representations*, (2017).
[14] Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song, and Jacob Steinhardt, 'Pixmix: Dreamlike pictures comprehensively improve safety measures', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16783–16792, (2022).
[15] Jens Henriksson, Christian Berger, Markus Borg, Lars Tornberg, Cristofer Englund, Sankar Raman Sathyamoorthy, and Stig Ursing, 'Towards structured evaluation of deep neural network supervisors', in *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*, pp. 27–34. IEEE, (2019).
[16] Thomas A Henzinger, Anna Lukina, and Christian Schilling, 'Outside the box: Abstraction-based monitoring of neural networks', in *24th European Conference on Artificial Intelligence-ECAI 2020*, (2020).
[17] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel, 'Detection of traffic signs in real-world images: The german traffic sign detection benchmark', in *The 2013 international joint conference on neural networks (IJCNN)*, pp. 1–8. Ieee, (2013).
[18] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira, 'Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10951–10960, (2020).
[19] Heinrich Jiang, Been Kim, Melody Y Guan, and Maya Gupta, 'To trust or not to trust a classifier', *arXiv preprint arXiv:1805.11783*, (2018).
[20] Hoki Kim, 'Torchattacks: A pytorch repository for adversarial attacks', *arXiv preprint arXiv:2010.01950*, (2020).
[21] Alex Krizhevsky, Geoffrey Hinton, et al., 'Learning multiple layers of features from tiny images', in *Technical report, University of Toronto*. Toronto, ON, Canada, (2009).
[22] Charline Le Lan and Laurent Dinh, 'Perfect density models cannot guarantee anomaly detection', *Entropy*, **23**(12), 1690, (2021).
[23] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin, 'A simple unified framework for detecting out-of-distribution samples and adversarial attacks', *Advances in neural information processing systems*, (2018).
[24] Kun-Lun Li, Hou-Kuan Huang, Sheng-Feng Tian, and Wei Xu, 'Improving one-class svm for anomaly detection', in *Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE Cat. No. 03EX693)*, volume 5, pp. 3077–3081. IEEE, (2003).
[25] Shiyu Liang, Yixuan Li, and R. Srikant, 'Enhancing the reliability of out-of-distribution image detection in neural networks', in *International Conference on Learning Representations*, (2018).
[26] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li, 'Energy-based out-of-distribution detection', *Advances in Neural Information Processing Systems*, **33**, 21464–21475, (2020).
[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng, 'Reading digits in natural images with unsupervised feature learning', in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, (2011).
[28] Zoe Papakipos and Joanna Bitton. Augly: Data augmentations for robustness, 2022.
[29] Douglas Reynolds, 'Gaussian mixture models', *Encyclopedia of biometrics*, 827–832, (2015).
[30] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli, 'Adversarially learned one-class classifier for novelty detection', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3379–3388, (2018).
[31] Adrian Schwaiger, Poulami Sinhamahapatra, Jens Gansloser, and Karsten Roscher, 'Is uncertainty quantification in deep learning sufficient for out-of-distribution detection?', in *AISafety@ IJCAI*, (2020).
[32] Alireza Shafaei, Mark Schmidt, and James J Little, 'A less biased evaluation of out-of-distribution sample detectors', *30th British Machine Vision Conference, Cardiff, Wales*, (2019).
[33] Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li, 'Out-of-distribution detection with deep nearest neighbors', in *International Conference on Machine Learning*, pp. 20827–20840. PMLR, (2022).
[34] Tecperson. Sign language mnist, version 1., 2017.
[35] Han Xiao, Kashif Rasul, and Roland Vollgraf, 'Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms', *arXiv preprint arXiv:1708.07747*, (2017).
[36] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao, 'Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop', *arXiv preprint arXiv:1506.03365*, (2015).
[37] Lily Zhang, Mark Goldstein, and Rajesh Ranganath, 'Understanding failures in out-of-distribution detection with deep generative models', in *International Conference on Machine Learning*. PMLR, (2021).
[38] Xibin Zhang, Maxwell L King, and Rob J Hyndman, 'A bayesian approach to bandwidth selection for multivariate kernel density estimation', *Computational Statistics & Data Analysis*, **50**(11), 3009–3031, (2006).