Implicit Neural Representation as vectorizer for classification task applied to diverse data structures

Thibault Malherbe
Inetum

Abstract

Implicit neural representations have recently emerged as a promising tool in data science research for their ability to learn complex, high-dimensional functions without requiring explicit equations or hand-crafted features. Here we aim to use these implicit neural representations weights to represent batch of data and use it to classify these batch based only on these weights, without any feature engineering on the raw data. In this study, we demonstrate that this method yields very promising results in data classification of several type of data, such as sound, images, videos or human activities, without any prior knowledge in the relative field.

1 Introduction

Implicit neural representations (INRs) have shown great promise in a variety of tasks, including image and shape synthesis, rendering, and inversion. INRs are neural networks that learn to represent a high-dimensional space implicitly without requiring an explicit parametric form for the function.

When it comes to classifying subsets of data using INRs, there are a few approaches that can be taken. One possible approach is to use INRs to learn a representation of the data that is optimized for classification. This can be done by training an INR on a dataset and using the learned weights as extracted features that are then fed into a classifier such as XGBoost or a neural network.

Extracting relevant information from sets of data is a fundamental task in many fields such as machine learning, data science, and engineering. The process of comparing and classifying sets of data often requires a minimum knowledge of the data itself, such as its statistical characteristics (min, max, average, etc.) of each of their variables. However, this approach can be limiting as it requires the data to be preprocessed in a specific way, and the choice of statistical characteristics may not always capture the most important features of the data.

In this paper, we demonstrate the effectiveness of using INRs for data classification. We show that the weights of a neural network can be used as a vector representation of a structured data set such as sound, images, videos or accelerometer data, and that this representation allows a model such as XGBoost to accurately classify data. Our experiments demonstrate the potential of INRs for several kinds of data and provide insights into their use for other similar applications.

Furthermore, INR are mainly used for data reconstruction, but this is not our goal here. That's why in this paper we will show if there is a direct correlation between the capacity to reconstruct data and the capacity to represent the data and have a good classification score.

2 Background and related work

Implicit neural representations (INRs) appear to be a good way to represent signals by continuous functions parameterized by neural networks. It has been used to represent diverses kinds of data such as shape parts (Genova, Cole, Vlasic, et al. 2019; Genova, Cole, Sud, et al. 2019), objects (Park et al. 2019; Michalkiewicz et al. 2019; Atzmon and Lipman 2020; Gropp et al. 2020), or scenes (Sitzmann, Zollhöfer, and Wetzstein 2019; Jiang et al. 2020; Peng et al. 2020; Chabra et al. 2020) but also images (Strümpler et al. 2022; Feng, Jabbireddy, and Varshney 2022), videos (H. Chen et al. 2021; Z. Chen et al. 2022) and audio (Szatkowski et al. 2022; Szatkowski et al. 2023; Lanzendörfer and Wattenhofer 2023).
All papers using implicit neural representations do not use them the same way. There is several applications such as super-resolution, compression or interpolation (Sitzmann, Martel, et al. 2020).
In particular SIREN (Sitzmann, Martel, et al. 2020) appears to be the most used architecture in all cited applications. SIREN architecture demonstrate that it's possible to cor-

Accepted pre-registered proposal at the 1st ContinualAI Unconference, 2023, Virtual. Full report to follow. Copyright 2023 by the author(s).

rectly perform classification task using INRs in the process but not directly on the weights of the INR (Xu et al. 2022).

3 Hypotheses and proposed method

3.1 Implicit neural representation

An implicit neural representation (INR) is a type of neural network architecture that learn to represent objects or scenes in a way that is independent of their explicit geometry or parametric representation. In other words, an INR is a neural network that can learn to represent complex shapes and patterns without explicitly encoding their geometry, topology, or other explicit mathematical descriptions.

An INR can learn to represent any type of data, whatever the dimension. By processing the raw signal data through a NN INR, the resulting vector representation can capture the salient features and patterns in the signal data without requiring explicit knowledge of the underlying signal processing or physics.

The generation of INRs is a consequence of machine learning procedures conducted on a specific data collection. Thus, it is imperative to treat the data intended for categorization as a well-structured dataset. Consequently, we adapt our data manipulation techniques to accommodate one dimension less. This is exemplified in the instance of a video, which can be dissected and interpreted as an image dataset. The neural network architecture typically employed for INR execution is likely a 2DConv neural network. Dimensionality reduction can be leveraged as a strategic advantage, allowing us to approach our data as a more conventional problem. This, in turn, simplifies the architecture of the neural network required for the task.

3.2 INR vectorization generalization

Let's consider a classification problem with a dataset E, with n classes and P features such as:
$n \in \mathbb{N}^*$,
$P \in \mathbb{N}^*$,
$E = \{A_1, A_2, ..., A_K\}$, where K is the number of subset that compose E,
$\forall i \in [1, K]$, $A_i \subset E$, with $\#(A_i) > 0$ and $\#(A_i) \leq \#(E)$,
$\forall i, j \in [1, K]$ such as $i \neq j$, $A_i \cap A_j = \emptyset$,
Every subset A_i could be classified in class C_i, with $C_i \in \{C_1, C_2, ..., C_n\}$.
Now let's take a neural network model M, with any achitecture and k weights. $\forall i \in [1, K]$, M is trained from scratch on A_i with a constant initialization. $\forall i \in [1, K]$, we obtain a vector W_i composed of k values, that are the weights of the model trained on A_i.

Now each A_i subset is vectorized into a W_i vector, that can be classified with any standard classifier.

3.3 Functions comparison

If a function represents a set of data, then it is possible to compare these functions directly, and in particular to classify these functions in the context of a classification.

In the context of implicit neural representation, the set of weights of a model trained on a subset of data represents this subset, and thus allows these weight vectors to be classified via a more conventional machine learning algorithm. This set of weight could be named *functa* (Dupont et al. 2022), "*a concise term for INRs that are to be thought of as data*".

According to XGBoost research (T. Chen and Guestrin 2016), XGBoost is a highly effective machine learning model for classification tasks, demonstrating superior performance through its regularized model formalization that effectively controls overfitting. Its parallelizable nature allows it to leverage the capabilities of multi-core computers, enhancing its speed and efficiency. Furthermore, XGBoost is versatile, capable of handling a variety of data types, missing values, and outlier values, and can be applied to both regression and classification tasks, including those involving categorical features. In the case of *functa* classification, the main advantage of XGBoost is that even in scenarios with a high number of features, XGBoost performs well due to its tree-based nature, which is renowned for handling high-dimensional data effectively, because depending to the architecture used to create *functas*, the number of features in the Machine learning processus could explodes.

However, it is important to note that XGBoost, like any machine learning model, can be affected by the "curse of dimensionality" when the feature space becomes exceedingly large, making it challenging for the model to identify patterns.

3.4 Reconstruction and classification correlation

The quality of an Implicit Neural Representation is often measured by how well it can reconstruct or generate the data it was trained on. This can be quantified using various metrics depending on the specific task. For instance, in image generation, one might use metrics like Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), or perceptual loss based on pre-trained networks. For 3D shapes, one might use Chamfer distance or Earth Mover's distance. In the case of graph data, one might use graph-based metrics like Graph Edit Distance or subgraph matching scores. But here we don't really want a good INR but a good *functa* so whatever the type of data processed, the only metric that interest us is the accuracy of our XGBoost classifier.

During our experiment we will demonstrate that the INR quality (capacity of data reconstruction) and the *functa* quality (capacity of being well classified) are not always correlated, and depending on our objectif, reconstruction or classification, we will not choose the same architecture of neural network.

4 Experimental protocol

The use of different types of data in our experiments is of paramount importance. The reason for this is twofold. Firstly, it allows us to evaluate the generalizability of functa across different data modalities. An approach that performs well across diverse datasets is likely to be more robust and versatile. Secondly, different types of data come with their unique challenges and characteristics. For instance, image data might involve dealing with high-dimensional inputs and complex spatial dependencies, while audio data might involve handling temporal dependencies. By testing on different types of data, we can gain insights into how well functa can handle these different challenges, which can guide future research and development.

4.1 Datasets

To test the approach and demonstrate its generalizability, we will work on four types of datasets: image, sound, video, and accelerometers.

4.1.1 CIFAR-10

The CIFAR-10 dataset is a widely used collection of images for research in image classification. It is commonly employed in the fields of computer vision and machine learning to evaluate and compare the performance of image classification algorithms. It consists of a total of 60,000 color images divided into 10 different classes, with 6,000 images per class. The classes include common objects such as cars, airplanes, birds, cats, dogs, and more. Each image is 32x32 pixels in size and encoded with three color channels (red, green, blue).

This dataset is interesting for classification research due to several challenges it presents:

- Class complexity: CIFAR-10 classes can be challenging to distinguish due to their visual similarity. For example, images of cats and dogs can be quite similar in terms of shape and color, making classification more difficult.
- Instance variability: Images within each class exhibit significant variability in terms of pose, orientation, scale, brightness, etc. This variability requires robust classification models capable of generalizing well to new instances.
- Dataset size: With 60,000 images, CIFAR-10 provides a sufficient amount of data to train machine learning models and evaluate their performance meaningfully.
- Image size: The 32x32 pixel images are relatively small compared to other datasets like ImageNet. This makes machine learning models lighter and faster to train, facilitating experimentation and iteration in research.

4.1.2 ESC-50

The ESC-50 (Environmental Sound Classification) dataset is a widely used dataset in research on environmental sound classification. It is specifically designed for the analysis and classification of sounds from everyday environments. It consists of a total of 2,000 audio clips, each lasting 5 seconds. The clips are divided into 50 different categories, representing various sounds such as dog barking, car horns, ocean waves, bird chirping, phone ringing, and more. Each class contains 40 audio examples.

This dataset is interesting for sound classification research for several reasons:

- Class variety: ESC-50 offers a wide variety of environmental sounds from different sources and contexts. This allows researchers to study and develop classification models capable of recognizing and distinguishing a wide range of real-world sounds.
- Classification challenges: Classifying environmental sounds can be complex due to acoustic variations, background noise, overlaps, and other factors. ESC-50 provides a realistic testing environment to evaluate the models' ability to tackle these challenges.
- Dataset size: With 2,000 audio clips, ESC-50 provides a sufficient amount of data to train classification models and evaluate their performance meaningfully.

4.1.3 HMDB-51

The HMDB-51 (Human Motion DataBase) dataset is a widely used dataset in research on human motion classification. It is specifically designed for the analysis and recognition of actions and movements performed by humans. It consists of a total of 6,766 video clips from 51 different action classes. The classes include movements such as walking, running, jumping, smiling, waving, lying down, dancing, and more. Each class contains a variable number of videos, with an average of about 70 videos per class. The videos are captured in diverse contexts, with different individuals, camera angles, lighting conditions, and more.

This dataset is interesting for research in human motion classification for several reasons:

- Action variety: HMDB-51 offers a wide range of human movements and actions, spanning from basic actions like walking and running to more complex actions like dancing and sports. This allows researchers to study and develop classification models capable of recognizing an extensive range of human actions.

- Recognition challenges: Recognizing human actions from videos can be challenging due to variations in poses, clothing, backgrounds, camera angles, and more. HMDB-51 provides a realistic challenge to evaluate the models' ability to identify and classify human movements under diverse conditions.

- Dataset size: With over 6,000 video clips, HMDB-51 provides a significant amount of data to train and evaluate classification models. This enables conducting experiments and statistically robust studies.

4.1.4 UCI-HAR

UCI-HAR dataset was developed to help in human activity recognition field (Jorge et al. 2012), which involved 30 volunteers engaging in various daily activities such as sitting, lying down, walking, standing, walking upstairs, and walking downstairs. The authors used a smartphone equipped with an accelerometer and gyroscope to capture tri-axial linear acceleration and angular velocities. The data was sampled at a rate of 50Hz and consisted of nine features. The data was then segmented into fixed-width sliding windows with 50% overlap, resulting in a total of 10,299 samples that have been segmented by user id.

4.2 Data Computation

Each dataset have to be processed differently. For image data (CIFAR-10), this involves training an implicit neural representation (INR) for each image. For audio data (ESC-50), this might involve training an INR for each audio clip. For video data (HMDB-51), this might involve training an INR for each video frame or sequence. For sensor data (UCI-HAR), this might involve training an INR for each sensor reading sequence.

There are several ways to create INR on each type of data. In all cited papers, INR to reconstruct image is a fonction that map the coordinate of of pixel to its value. But an image could be seen as a time series, and a LSTM could be trained on it, and, given the N first pixel values, predict the N+1 pixel value.

For each dataset we will test different approaches and discuss the relevance of the INR and the *functa* created this way. It will permit us to see if their is any direct correlation between the INR quality and *functa* quality. Moreover, we will be able to see if certain architectures are more suitable to create good *functa*.

4.3 Evaluation metrics

This paper evaluates the effectiveness of *functas* with XGBoost using various performance metrics, as described below:

Accuracy is defined as the ratio of correctly predicted samples to the total number of samples, where TP denotes true positives, FN denotes false negatives, TN denotes true negatives, and FP denotes false positives.
Accuracy = (TP + TN) / (TP + TN + FP + FN)

A confusion matrix (CM) is a square matrix that provides a complete performance analysis of a classification model. The rows of the CM represent instances of true class labels, while the columns represent predicted class labels. The diagonal elements of the matrix indicate the percentage of points for which the predicted label is equal to the true label.

We will also evaluate the quality of the INR used, with a PSNR wich is most commonly defined via the mean squared error (MSE) between two images.

4.4 Baseline comparison

During these experiments we'll need two types of baseline:

- An INR baseline to compare what is a good INR for data reconstruction. That baseline will be classified as *functa* too. To achieve that we will use SIREN as baseline if possible for each type of data.

- A classifier baseline to compare with a more common classifier for the current type of data. We will use 2dConv model as baseline image classifier, a 3Dconv model for videos, 1Dconv model for sound classifier, logistic regression for human activity recognition.

4.5 Ablation Studies

To understand the contribution of different components of the functa models, we perform ablation studies, which involve removing or modifying certain components and observing the effect on performance.

4.6 Analysis

Finally, we analyze the results, compare the performance of the *functa* models with the baseline models, discuss the results of the ablation studies, and provide insights into why the functa models performed as they did.

5 Experimental Results

This section details the evaluation approach for different data types using three distinct sections. Initially, the process of data learning and reconstruction by the Implicit Neural Representation (INR) models is discussed. The subsequent section focuses on baseline results, presenting the Peak Signal-to-Noise Ratio (PSNR) for the baseline INR models and the accuracy metrics for the baseline classifiers. The final section highlights the superior results achieved using functas.

5.1 Common approach

5.1.1 INRs structure

Network Architecture: The architecture of an INR is usually a fully connected neural network. For the best results the model has 3 dense layers.

Activation Functions: Special activation functions such as sinusoidal functions (e.g., sine and cosine) are often used instead of more common activations like ReLU. This choice helps in learning the smooth and continuous nature of the underlying function.

5.1.2 Weight Initialization

In tensorflow, by default the weights of Dense layers are drawn from a uniform distribution in order to break symmetry between neurons. If all weights are initialized the same, neurons will learn the same features during training, which is inefficient and ineffective.

Here neurons need to have the same ”role” across all the INRs creation, because funtas values have to represent the same feature, and that is why it’s important to iniatilize all the INRs the same way.

Also, it could be more efficient if the initial weights could be in the center of the funtas space.

So all INR models used to create functas begin with a standard initialization process. To accomplish this, two samples from each class are utilized to construct INRs with random initial weights. Each INR is trained using the same number of epochs and batch size as utilized in the broader experimental framework.

After this initial training phase, the model designated for the experimental runs is initialized using the average weights derived from this preliminary training.

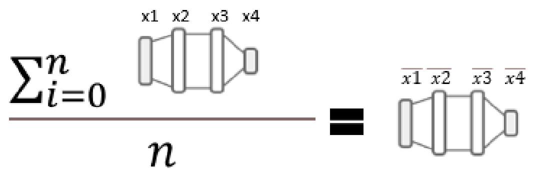
$\sum_{i=0}^{n}$
x1 x2 x3 x4
n
$\overline{x1}$ $\overline{x2}$ $\overline{x3}$ $\overline{x4}$

Figure 1: Weight initialization method

5.1.3 Filtering and Normalization

Before launching the training of the classifier, once the functas had been extracted, we implemented certain optimization measures for efficiency reasons.

During the training of INRs, some weights could stay unchanged for every training. These weights values are useless in the functaset to differenciate two objects. So a filtering step is implemented to check for invariant values across the functas. If such invariant values are identified, the corresponding columns are removed to streamline the data structure.

Additionally, to enhance the learning performance of the XGBoost algorithm, the remaining columns are normalized.

5.2 CIFAR-10

5.2.1 Implicit neural representation

Network Input: The inputs to the network are the coordinates of the pixels. For a 2D image, these coordinates are typically the (x,y) positions of each pixel.

Training Data: During training, the network learns from a set of coordinates and their corresponding pixel values. This training can be seen as fitting a high-dimensional curve to the image data.

Output: The output of the network for any given input coordinate is the predicted pixel value at that coordinate. This allows the model to generate the image continuously across resolutions and coordinates.

5.2.2 Baseline results

INR Baseline: For the INR baseline, we use a Sinusoidal Representation Networks (SIREN) model. For images, the SIREN model achieved a mean PSNR of **52.63 dB**.

Classifier Baseline: The baseline model in this context is a 2D Convolutional Neural Network (2D Conv), which is widely used for image classification tasks due to its efficiency in handling spatial hierarchies in images. The accuracy of this model reached **0.795**, demonstrating its competence in classifying images accurately.

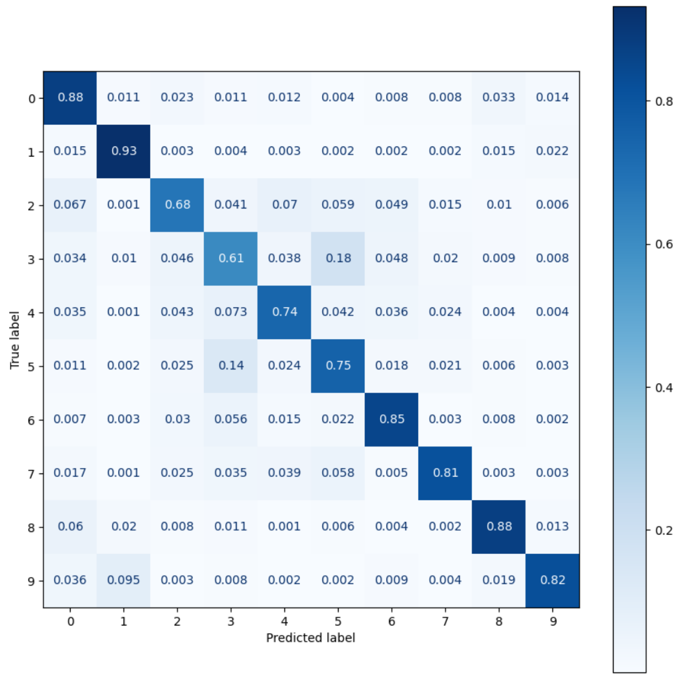

Figure 2: Confusion matrix resulting from the baseline classifier on CIFAR-10 dataset

5.2.3 Functa results

Table 1: Performance comparison of baseline model and INR with the proposed model on the CIFAR-10 dataset

Model	**Accuracy**	**PSNR**
SIREN baseline with XGBoost	0.752	52.63
CONV2D	0.795	-
Best functa with XGBoost	0.779	60.24

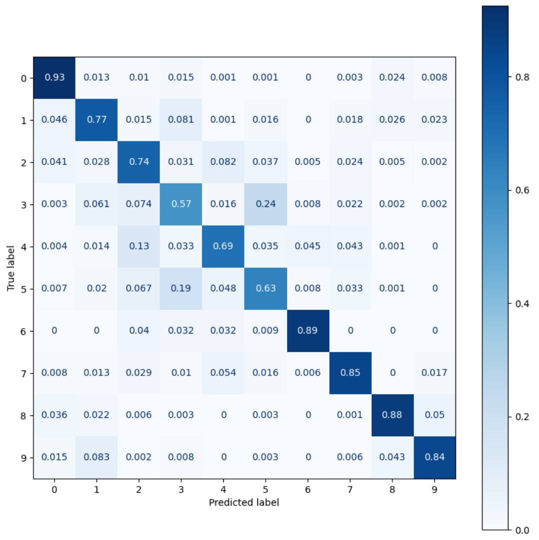

Figure 3: Confusion matrix resulting from the examination of the proposed model on CIFAR-10 dataset

5.3 HMDB-51

5.3.1 Implicit neural representation

Network Input: The inputs to the network are the coordinates of the pixels in addition to time. For a video, these coordinates are typically the (x, y, t) positions, where x and y represent spatial pixel positions and t represents the time or frame index.

Training Data: During training, the network learns from a set of spatial coordinates, time points, and their corresponding pixel values. This training can be seen as fitting a high-dimensional curve to the video data, capturing not only the appearance but also the changes over time.

Output: The output of the network for any given input coordinate and time point is the predicted pixel value at that spatial location and time. This allows the model to generate video frames continuously across resolutions, coordinates, and time, offering potential for generating high-resolution videos or slow-motion effects from limited data.

5.3.2 Baseline results

INR Baseline: For the INR baseline, we use a Sinusoidal Representation Networks (SIREN) model. For videos, the SIREN model achieved a mean PSNR of **61.4 dB**.

Classifier Baseline: For video classification, a 3D Convolutional Neural Network (3D Conv) serves as the baseline. This model is adept at processing both spatial and temporal dimensions, making it a standard choice for video analysis tasks. It achieved an accuracy of **0.386**, confirming its robustness in recognizing and classifying dynamic patterns across video frames.

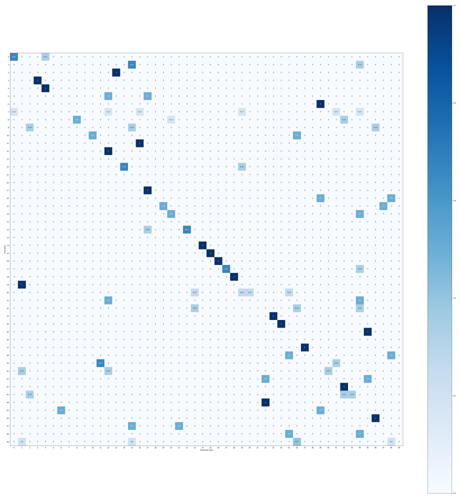

Figure 4: Confusion matrix resulting from the baseline classifier on HMDB-51 dataset

5.3.3 Functa results

Table 2: Performance comparison of baseline model and INR with the proposed model on the HMDB-51 dataset

Model	Accuracy	PSNR
SIREN baseline with XGBoost	0.09	61.4
CONV3D	0.386	-
Best functa with XGBoost	0.324	62.77

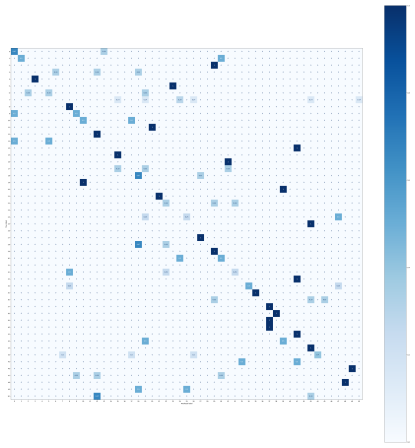

Figure 5: Confusion matrix resulting from the examination of the proposed model on HMDB-51 dataset

5.4 UCI-HAR

5.4.1 Implicit neural representation

Network Input: The input to the network for both magnetometer and accelerometer data is solely the time coordinate, represented as t. This reflects the time at which each sensor reading is taken.

Training Data: During training, the network learns from a dataset consisting of time coordinates and their corresponding sensor readings across the x, y, and z axes. This training approach fits a high-dimensional curve to the time series data, modeling the complex patterns and variations in acceleration and magnetic fields over time.

Output: The output of the network for any given input time coordinate t is the predicted values of sensor readings at that moment along the x, y, and z axes. This model thus generates continuous, high-fidelity reproductions of the sensor data, which is particularly beneficial for tasks requiring fine-grained analysis, such as motion tracking, orientation detection, and condition monitoring.

5.4.2 Baseline results

INR Baseline: For the INR baseline, we use a Sinusoidal Representation Networks (SIREN) model. For images, the SIREN model achieved a mean PSNR of **22.5 dB**.

Classifier Baseline: The baseline classifier for this experiment was a logistic regression model, chosen for its efficiency in multiclass classification tasks involving time-series data like those from accelerometers and magnetometers. With an accuracy of **0.527**, the logistic regression model demonstrates a low level of precision in classifying different types of human activities, this model does'nt seems to be relevant for HAR tasks.

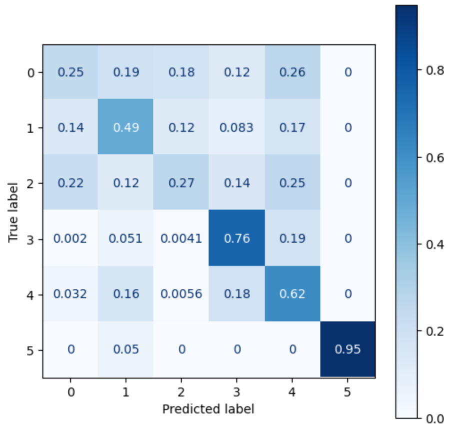

Figure 6: Confusion matrix resulting from the baseline classifier on UCI-HAR dataset

5.4.3 Functa results

Table 3: Performance comparison of baseline model and INR with the proposed model on the UCI-HAR dataset

Model	**Accuracy**	**PSNR**
SIREN baseline with XGBoost	0.839	22.5
Logistic regression	0.527	-
Best functa with XGBoost	0.865	20.81

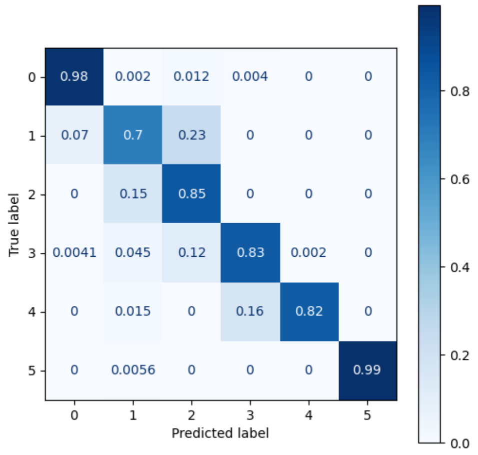

Figure 7: Confusion matrix resulting from the examination of the proposed model on UCI-HAR dataset

5.5 ESC-50

5.5.1 Implicit neural representation

Network Input: The inputs to the network are the time coordinates of the sound sample. For sound, these coordinates are typically the t values, representing different time points in the audio clip.

Training Data: During training, the network learns from a set of time coordinates and their corresponding sound amplitudes. This training can be seen as fitting a high-dimensional curve to the sound data, effectively modeling the waveform over time.

Output: The output of the network for any given input time coordinate is the predicted sound amplitude at that time. This allows the model to generate the sound continuously across time points, which can be particularly useful for creating high-resolution audio or extending audio durations.

5.5.2 Baseline results

INR Baseline: For the INR baseline, we use a Sinusoidal Representation Networks (SIREN) model. For sound, the SIREN model achieved a mean PSNR of **55 dB**.

Classifier Baseline: Alongside the INR, we also benchmark against a conventional sound classification model. For this purpose, a 1D Convolutional Neural Network (1D Conv) is employed, given its proficiency in analyzing temporal sequence data such as audio. The accuracy of this model reached **0.57**, showcasing its effectiveness in accurately classifying sound patterns and frequencies.

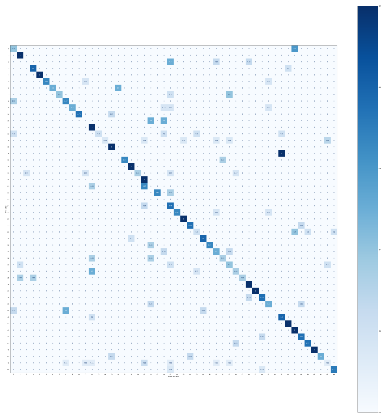
Figure 8: Confusion matrix resulting from the baseline classifier on ESC-50 dataset

5.5.3 Specific preprocess

Throughout the experimentation phase, we encountered significant performance challenges. The functas did not achieve the anticipated classification accuracy when the INRs were trained directly on raw data. To address this issue, we opted to employ Mel Frequency Cepstral Coefficients (MFCC) for training our INRs. MFCC is a widely recognized technique in sound processing, known for its effectiveness in capturing the key temporal and spectral characteristics of audio signals.

5.5.4 Functa results

Table 4: Performance comparison of baseline model and INR with the proposed model on the ESC-50 dataset

Model	Accuracy	PSNR
SIREN baseline with XGBoost	0.073	74.11
SIREN baseline with XGBoost/mfcc	0.383	15.76
CONV1D	0.57	-
Best functa with XGBoost	0.118	13.17
Best functa with XGBoost and mfcc	0.465	15.75

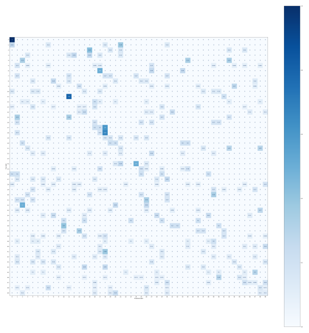
Figure 9: Confusion matrix resulting from the examination of the proposed model on ESC-50 dataset

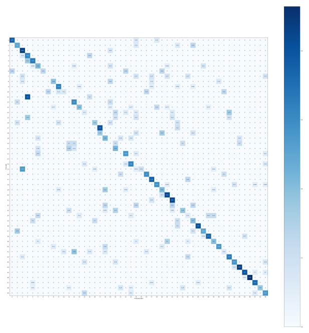
Figure 10: Confusion matrix resulting from the examination of the proposed model with mfcc treatment on ESC-50 dataset

6 Analysis

6.1 Images and videos

The experimental outcomes for both image and video data, while slightly below the established baselines, are nonetheless encouraging. Our results confirm the viability of using INRs with an XGBoost classifier for these types of data, which are well-known to be suitable for INR applications.

Performance Comparison : Our findings demonstrate that, although our results did not surpass the baseline metrics, the performance of our INR models closely approaches these benchmarks. This slight underperformance is not unexpected in the initial stages of implementing a new methodology and underscores the fundamental soundness of the INR approach for handling complex visual data.

Potential for Optimization : There is optimism that with further optimization and continued refinement of our models, the performance can not only meet but potentially exceed the established baselines. Adjustments in the network architecture, training procedures, and parameter tuning are anticipated to enhance the efficacy of our models.

Suitability and Expectations : The fact that both videos and images are effective candidates for INR-based models is well-documented. Our results reinforce this understanding and provide a solid foundation for future investigations into this area. The successful application of INRs for these media types aligns with our expectations and contributes to the growing body of evidence supporting the robustness of INR techniques.

Future Work : The current study lays the groundwork for additional research aimed at refining these techniques. Further experiments and development efforts are required to fully realize the potential of integrating INRs with machine learning classifiers like XGBoost. Exploring various configurations and settings within this framework will be crucial for advancing our understanding and application of this technology.

6.2 Sensor data

Our experiments with sensor data, specifically from accelerometers, magnetometers, and gyroscopes, using INRs combined with XGBoost classification, have yielded results that are notably impressive compared to our established baseline. However, this comparison requires careful interpretation due to the inherent limitations of the baseline model.

Comparison to Baseline: While our results significantly surpass the baseline performance, it's important to acknowledge that the baseline itself is not an optimal classifier for this type of data. The baseline model demonstrates considerably poor performance, which somewhat diminishes the impact of our comparative success. Thus, our achievements, though notable, are benchmarked against a relatively weak standard.

State-of-the-Art Comparison: When contrasting our results with state-of-the-art models, specifically some CNN models that achieve an accuracy of 92.7%, our model's accuracy of 86.5% appears competitive but still not up to the leading edge in this field. This comparison underscores the potential yet untapped in optimizing our approach to better match or even surpass these top-performing models.

Potential for Optimization: The current performance of our INR models suggests that with further refinement and optimization, particularly in how we tune and train our INRs, there could be significant improvements. Moreover there are other classifiers that could improve significantly the accuray.

6.3 Sound

The examination of sound data using INRs integrated with an XGBoost classifier produced mixed outcomes. Notably, the results on raw sound data were disappointing, which can be attributed primarily to the inherent complexities and the substantial size of the data involved.

Challenges with Large Data Sets: For a typical 5-second sound clip, we are dealing with a dataset exceeding 200,000 values. Such extensive data sets pose significant challenges for INRs, which may struggle to effectively represent this level of complexity within a manageable computational framework. The initial hypothesis was that larger neural networks, trained over many epochs, might be better suited to handle this scale of data.

Parameter Adjustments and Sampling Rate Changes: Despite multiple attempts to optimize the model by adjusting parameters and reducing the sampling rate to decrease the data size, the outcomes were not satisfactory. These modifications did not yield the improvements in performance that were anticipated, underscoring the difficulty of working with high-dimensional sound data in an INR framework.

Utilizing Basic Data Processing: In light of these challenges, we resorted to applying basic preprocessing techniques to the raw data, similar to those used for the baseline models. Specifically, we implemented Mel Frequency Cepstral Coefficients (MFCC) processing, which is a standard treatment for simplifying sound data before classification tasks. This approach led to results that, while still slightly below the baseline, were much closer to what was expected, indicating that this method can be effective with minimal alterations to the raw data.

7 Ablation Studies

In our ablation studies, we sought to understand the impact of reducing the computational resources on the performance of our INR models. Specifically, we halved the size of each layer in the network, aiming to analyze the trade-offs between training speed and model accuracy.

The following table summarizes the accuracy results of the INRs trained with full-sized and half-sized layers for different types of data:

Table 5: Accuracy Results with Full and Half Layer Sizes

Data Type	Full Layer Size (%)	Half Layer Size (%)
Images	0.779	0.652
Videos	0.324	0.288
HAR	0.865	0.811
Sound	0.465	0.36

The results indicate a consistent pattern across all data types: reducing the size of each layer by half resulted in a significant increase in processing speed, approximately doubling the speed of training. However, this reduction in layer size also led to a decrease in model accuracy, although the extent of the decline varied across different data types.

Conclusively, this ablation study highlights the feasibility of using smaller networks for faster processing, which may be beneficial in scenarios where computational resources are limited or rapid model deployment is necessary. However, for applications where maximum accuracy is critical, utilizing full-sized layers remains advisable. This balance between speed and accuracy must be carefully considered in the design of practical Functas-based systems.

8 Conclusion

The results presented in this paper provide a promising foundation for future research in utilizing INRs coupled with simple classifiers such as XGBoost. Our findings demonstrate the functionality and potential of this approach across various data types, including images, videos, sound, and sensor data. Each of these applications presents unique challenges, but also opportunities for significant advancements in the field of machine learning. The classification performance of our functa-based approach closely aligns with the selected baseline models. It is important to note that we did not compare our results against state-of-the-art models, as these are often highly optimized for specific tasks. In contrast, our approach is designed to be broadly applicable across various data types, which inherently limits the extent of optimization for any one domain but enhances its versatility. This versatility positions our method as a particularly promising candidate for multimodal classification tasks, where it has the potential to integrate and process diverse data types seamlessly.

The application of INR models to complex data sets, such as large-scale sound and high-dimensional sensor data, has shown that while the initial results are promising, they also require substantial enhancements to match the performance of state-of-the-art models. The success of basic preprocessing in improving model performance underscores the importance of effective data simplification prior to employing these complex models. However, the continual need to refine model architectures and adjust training strategies parallels the challenges faced in continual learning.

Optimizations in the domain of continual learning could significantly enhance the efficiency and effectiveness of using INRs. Techniques such as dynamic architectural adjustments, effective regularization strategies, and novel training paradigms could be adapted to improve the training processes and utility of INR models in handling the complexity and vastness of the data types explored.

Future research will therefore not only continue to refine and optimize INR models and their integration with classifiers but also explore how principles of continual learning can be applied to improve the adaptability and efficiency of these models. By addressing these challenges, we can enhance the scalability and applicability of INRs, making them more robust and versatile tools in the machine learning toolkit. Finally we can explore other existing extraction approach of INRs, like the utilization of modulation, without forgetting that there are multitude of classifiers that could increase our results, and some other way to treat these INR such as graphs.

References

Atzmon, M. and Y. Lipman (2020). “Sal: Sign agnostic learning of shapes from raw data”. In: *Proc. CVPR*.

Chabra, R. et al. (2020). “Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction”. In: *arXiv preprint arXiv:2003.10983*.

Chen, H. et al. (2021). *NeRV: Neural Representations for Videos*. arXiv: 2110.13903 [cs.CV].

Chen, T. and C. Guestrin (Aug. 2016). “XGBoost”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145%2F2939672.2939785.

Chen, Z. et al. (2022). *VideoINR: Learning Video Implicit Neural Representation for Continuous Space-Time Super-Resolution*. arXiv: 2206.04647 [eess.IV].

De Luigi, L. et al. (2023). “Deep Learning on Implicit Neural Representations of Shapes”. In: *International Conference on Learning Representations (ICLR)*.

Dosovitskiy, A. et al. (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=YicbFdNTTy.

Dupont, E. et al. (2022). *From data to functa: Your data point is a function and you can treat it like one*. arXiv: 2201.12204 [cs.LG].

Feng, B. Y., S. Jabbireddy, and A. Varshney (Nov. 2022). “VIINTER: View Interpolation with Implicit Neural Representations of Images”. In: *SIGGRAPH Asia 2022 Conference Papers*. ACM. DOI: 10.1145/3550469.3555417. URL: https://doi.org/10.1145%2F3550469.3555417.

Genova, K., F. Cole, A. Sud, et al. (2019). “Deep Structured Implicit Functions”. In: *arXiv preprint arXiv:1912.06126*.

Genova, K., F. Cole, D. Vlasic, et al. (2019). “Learning shape templates with structured implicit functions”. In: *Proc. ICCV*, pp. 7154–7164.

Gropp, A. et al. (2020). “Implicit geometric regularization for learning shapes”. In: *arXiv preprint arXiv:2002.10099*.

Jiang, C. et al. (2020). “Local implicit grid representations for 3d scenes”. In: *Proc. CVPR*, pp. 6001–6010.

Jorge, R.-O. et al. (2012). *Human Activity Recognition Using Smartphones*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54S4K.

Kabir, H. M. D. (2023). *Reduction of Class Activation Uncertainty with Background Information*. arXiv: 2305.03238 [cs.CV].

Lanzendörfer, L. A. and R. Wattenhofer (2023). *Siamese SIREN: Audio Compression with Implicit Neural Representations*. arXiv: 2306.12957 [cs.SD].

Michalkiewicz, M. et al. (2019). “Implicit surface representations as layers in neural networks”. In: *Proc. ICCV*, pp. 4743–4752.

Oquab, M. et al. (2023). *DINOv2: Learning Robust Visual Features without Supervision*. arXiv: 2304.07193 [cs.CV].

Park, J. J. et al. (2019). “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”. In: *Proc. CVPR*.

Peng, S. et al. (2020). “Convolutional occupancy networks”. In: *arXiv preprint arXiv:2003.04618*.

Sitzmann, V., J. N. P. Martel, et al. (2020). *Implicit Neural Representations with Periodic Activation Functions*. arXiv: 2006.09661 [cs.CV].

Sitzmann, V., M. Zollhöfer, and G. Wetzstein (2019). “Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations”. In: *Proc. NeurIPS*.

Strümpler, Y. et al. (2022). *Implicit Neural Representations for Image Compression*. arXiv: 2112.04267 [eess.IV].

Szatkowski, F. et al. (2022). *HyperSound: Generating Implicit Neural Representations of Audio Signals with Hypernetworks*. arXiv: 2211.01839 [cs.SD].

— (2023). *Hypernetworks build Implicit Neural Representations of Sounds*. arXiv: 2302.04959 [cs.LG].

Xu, D. et al. (2022). *Signal Processing for Implicit Neural Representations*. arXiv: 2210.08772 [cs.CV].

9 PSNR and accuracy

During our experiments, we initially hypothesized a strong correlation between the Peak Signal-to-Noise Ratio (PSNR) and accuracy. The rationale behind this expectation is straightforward: a higher PSNR indicates that the Implicit Neural Representation (INR) more accurately reproduces the object it represents. Consequently, we anticipated that functas derived from INRs with higher PSNR values would be classified more accurately, suggesting a direct correlation between these two metrics.

However, our experimental observations did not conform to these expectations. As illustrated in the graphs below, which plot accuracy against PSNR for different types of data, the relationship varies significantly depending on the data type. For instance, in some cases, even moderate PSNR improvements led to substantial increases in accuracy, whereas in others, substantial enhancements in PSNR did not translate into corresponding improvements in classification accuracy.

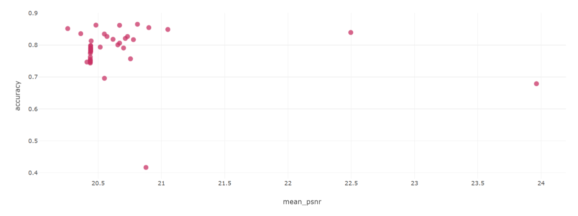

Figure 11: Graphs depicting the relationship between PSNR and accuracy for HAR experiments

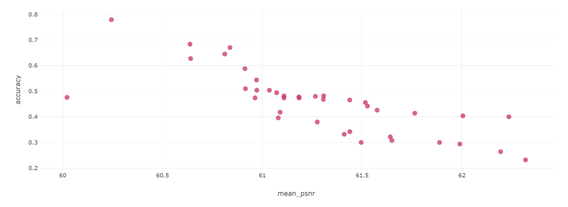

Figure 12: Graphs depicting the relationship between PSNR and accuracy for images experiments

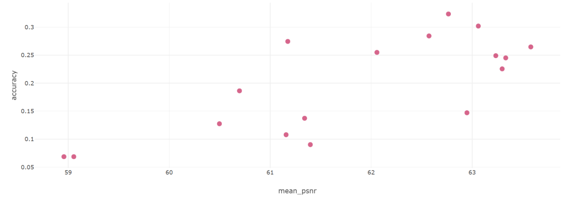

Figure 13: Graphs depicting the relationship between PSNR and accuracy for videos experiments

Consequently, while PSNR remains a valuable metric for assessing the quality of data reconstruction by INRs, we cannot conclusively assert a direct correlation between PSNR and classification accuracy across all types of data

10 Models description

In this section main models used will be explicitly descibed.

10.1 Best functa

For each data type, the same model was used to create functas. This model was created with tensorflow and has the following structure :

- An **Input layer** with shape $(input - size,)$, designed to take a single value representing a time-encoded feature.
- A **Dense layer** with 128 units using a Sine activation function with a frequency of 30.
- Another **Dense layer** with 64 units, also using a Sine activation.
- An **Output layer** with $ouput - size$ unit, employing a sigmoid activation.

Input and output sizes depend of data type processed:

Table 6: Input and output sizes of models used for functa creation

Sizes	Sound	Image	Video	HAR
Input	1	2	3	1
Output	1	3	3	9

10.2 Siren baseline

For all baseline INR, the model used is a SIREN model that come from the library tf-siren. Input and output are the same than previously in best functas. These SIREN are created this way:

Listing 1: Python code for creating a the SIREN baseline model

```
from tf_siren import
   SinusodialRepresentationDense
model = models.Sequential([
    layers.Input(shape=(3,)),  # Input
       layer with shape 2
    SinusodialRepresentationDense(64,
       w0=30.0),  # Dense layer with
       Sine activation
    SinusodialRepresentationDense(64,
       w0=1.0),  # Dense layer with
       Sine activation
    SinusodialRepresentationDense(64,
       w0=1.0),  # Dense layer with
       Sine activation
    SinusodialRepresentationDense(64,
       w0=1.0),  # Dense layer with
       Sine activation
    SinusodialRepresentationDense(64,
       w0=1.0),  # Output layer with
       sigmoid activation
    SinusodialRepresentationDense(3,
       activation="linear")    #
       Output layer with linear
       activation
])
```

10.3 XGBoost

The model used to classify functas is an XGBoost with the following parameters:

- An **max-depth** : 6
- An **eta** : 0.3
- An **objective** : multi:softmax
- An **eval-metric** : mlogloss
- An **seed** : 42

11 Multimodality

Our experiments have delved into the fascinating field of multimodality, particularly in the context of transforming diverse types of data into vectors of the same dimension. This approach simplifies the process of combining multiple vectors to create a singular vector that carries information from various data types.

In our case, we focused on video and sensor data types, linking each video with a corresponding human activity. This approach allowed us to create a multi-dimensional representation of both the visual and sensor-based aspects of the activity.

The initial results have been promising. Without any additional optimization on the data, our model achieved an accuracy of 77.5%. This suggests that our approach to multimodal data representation can effectively integrate and interpret information from different types of data.

However, it's important to note that while our initial results are encouraging, there is still room for improvement. Further optimization of the data and refinement of our model could potentially lead to higher accuracy rates. This could involve fine-tuning the process of transforming data into vectors or improving the way we link videos with human activities.

In conclusion, our exploration into multimodality has demonstrated its potential in handling and interpreting diverse types of data. With further research and optimization, this approach could significantly improve the way we analyze and understand complex data sets.

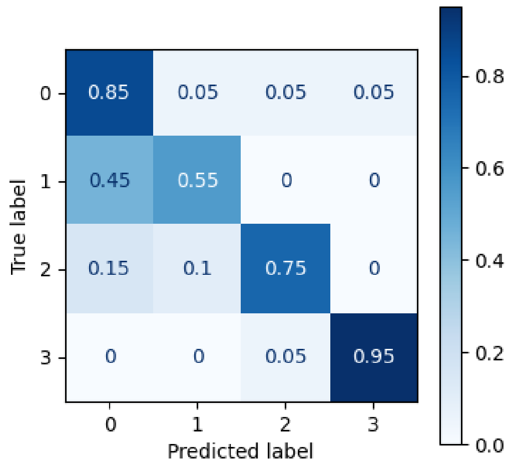

Figure 14: Multimodal experiment confusion matrix

Acknowledgements

We extend our heartfelt gratitude to Anis Takka, whose invaluable assistance was pivotal in setting up the experiments, contributing to the writing of this paper, and providing numerous insightful ideas that enriched our research. We are also deeply appreciative of the support and guidance provided by Etienne Gay and Gabriel Prevot. Their roles as managers were instrumental in fostering a productive research environment, and their meticulous review and methodological advice significantly enhanced the quality of our work.