Neural Networks 153 (2022) 215–223

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

SepNet: A neural network for directionally correlated data

Fuchang Gao [a,*], Yiqing Ma [b], Boyu Zhang [c], Min Xian [d]

[a] Department of Mathematics and Statistical Science, University of Idaho, 875 Perimeter Drive MS 1403 Moscow, ID 83844-1403, United States of America
[b] Department of Computer Science, University of Idaho, 875 Perimeter Drive MS 1010 Moscow, ID 83844-1010, United States of America
[c] Institute for Modeling Collaboration and Innovation, University of Idaho, 875 Perimeter Dr MS 1122, Moscow, ID 83844-1122, United States of America
[d] Department of Computer Science, University of Idaho at Idaho Falls, 1776 Science Center Drive Idaho Falls, ID 83402, United States of America

ARTICLE INFO

Article history:
Received 1 February 2022
Received in revised form 18 May 2022
Accepted 2 June 2022
Available online 9 June 2022

Keywords:
Neural network
Correlation matrix
Directional linear operator
Directional convolutional operator
Spectrogram
Multichannel signal

ABSTRACT

Multi-dimensional tensor data appear in diverse settings, including multichannel signals, spectrograms, and hyperspectral data from remote sensing. In many cases, these data are directionally correlated, *i.e.* the correlation between variables from different dimensions is significantly weaker than the correlation between variables from the same dimension. Convolutional neural networks are readily applicable to directionally correlated data but are often inefficient, as they impose many unnecessary connections between neurons. Here we propose a novel architecture, SepNet, specifically for directionally correlated datasets. SepNet uses directional operators to extract directional features from each dimension separately, followed by a linear operator along the depth to generate higher-level features from the directional features. Experiments on two representative directionally correlated datasets showed that SepNet improved network efficiency up to 100-fold while maintaining high accuracy comparable with state-of-the-art convolutional neural network models. Furthermore, SepNet can be flexibly constructed with minimal restriction on the output shape of each layer. These results reveal the potential of data-specific architecting of neural networks.

1. Introduction

Many successful neural network architectures have been developed in recent years, such as convolutional neural networks (CNNs) (Fukushima, 2004), recurrent neural networks (RNNs) (Rumelhart, Hinton, & Williams, 1986) and graph neural networks (GNNs) (Wu et al., 2021). Generally speaking, CNNs perform well on natural images and videos, RNNs are commonly used in speech recognition and natural language processing, while GNNs fit better for crystallographic information files (Sanyal et al., 2018; Xie & Grossman, 2018; Zhang, Wang and Gao, 2021) and large molecular structures (Cho & Choi, 2018; Schmidt, Pettersson, Verdozzi, Botti, & Marques, 2021; Zhang, Zhou, Wu, & Gao, 2022). The performance of neural network models depends heavily on the data and their representations to which the models are applied. Data can have many different representations. While Fourier (Minami, Nakajima, & Toyoshima, 1999) and wavelet (Zhang & Benveniste, 1992) representations have traditionally been used for capturing global spectral features, recently, methods for local feature extraction have received a great deal of attention. For example, discriminant locality preserving projections (DLPP) have been proposed to find the subspace that best discriminates between different classes and reduces noise (Yu, Teng, & Liu, 2006), and by using fuzzy set theory, Wan et al. (2021) (see also Wan et al., 2016; Wan, Yao, Zhan, & Yang, 2022) solved the problem of sensitivity to outliers in DLPP. Another popular approach is to use attention and transformer (Bahdanau, Cho, & Bengio, 2015; Mnih, Heess, Graves, & Kavukcuoglu, 2014; Vaswani et al., 2017; Weston, Chopra, & Bordes, 2015; Xu et al., 2015) to extract local features. These local features can be used in either a stand-alone manner, or more often, together with RNNs, CNNs or GNNs.

Here we consider the case when local features have been extracted and the data are represented as a multidimensional tensor. These data could be original data such as images or videos, or data where local features or local attentions have been extracted and stored in their relative spatial locations. This is of course not the most general case, but nonetheless covers a large class of real-world data. Currently, a common practice is to feed such data directly into neural networks such as CNNs.

Many well-known CNN models have been developed, such as VGGNet (Szegedy, Ioffe, Vanhoucke, & Alemi, 2017), GoogleNet (Simonyan & Zisserman, 2014), Inception (Xia, Xu, & Nan, 2017; Xie, Girshick, Dollár, Tu, & He, 2017), ResNet (He, Zhang, Ren, & Sun, 2016b), DenseNet (Huang, Liu, Van Der Maaten, & Weinberger, 2017), MobileNet V1/V2 (Chollet, 2017; Sandler, Howard,

* Corresponding author.
E-mail address: fuchang@uidaho.edu (F. Gao).

https://doi.org/10.1016/j.neunet.2022.06.005
0893-6080/

Zhu, Zhmoginov, & Chen, 2018), and EfficientNet (Lee, Bang, & Yang, 2017). VGGNet designed the architecture with many layers and very small kernels based on the developers' experience. GoogleNet and Inception designed their architectures based on the Hebbian principle and intuition of multi-scale processing. ResNet introduced skip connections in the architecture based on optimal information propagation within the network (He, Zhang, Ren, & Sun, 2016a), essentially solving the problem of vanishing gradient. DenseNet strengthened the feature propagation and reduced parameters in ResNet by generalizing the skip connections. MobileNet introduced depth-wise separable convolutions to build light-weight deep neural networks for better efficiency. EfficientNet introduced model scaling to balance network depth, width, and resolution for better performance. These state-of-the-art models were all built without using specific structural information of the multidimensional tensor data.

In this paper, we propose a neural network architecture based on the specific correlation structure of multidimensional tensor data. Our idea is motivated by the observation that many multi-dimensional tensor data have some distinct correlation structure that allows for the construction of much more efficient neural network models. Consider multichannel signals that are generated by different types of sensors. Such data are often expressed as multi-dimensional tensors in which each dimension corresponds to one type of sensor. When there are only two types of sensors, the data may appear as a table in which the information in entry (i, j) is collaboratively collected by ith Type-A sensor and jth Type-B sensor. At first glance, these data may look like simple gray-scale images. However, analysis of the correlation structure can often reveal characteristic patterns showing that the variables within the same row or column are more correlated, while variables from different dimensions are much less correlated. This is understandable on a practical level, as the information collected by the same type of sensor may bear similar marks, such as precision. One typical pattern is that the Pearson correlation coefficients (to be detailed in the next section) are high on a narrow horizontal or vertical band, or on a cross. This is very different from natural images, in which for each pixel, the highly correlated pixels typically lie in a small circular neighborhood surrounding that pixel (Fig. 1a). We call the data *directionally correlated* if the correlation between variables from different dimensions is significantly weaker than the correlation between variables from the same dimension.

Directionally correlated data are not limited to multichannel signals and are in fact very common. For example, the spending history of m people on n types of merchandise can be represented as a sequence of $m \times n$ matrices, resulting in two naturally-occurring dimensions (rows and columns). Data points in the same row or column are likely to be highly correlated, wheres those from different rows or columns are likely to be less correlated, as the result of habitual or stereotyped spending or retail behavior. Hyperspectral data cubes from remote sensing are another example of directionally correlated data.

In addition to data describing natural processes, distinct dimensions can also be introduced during data representation. For example, audio waves are time series data of one dimension, but in applications such as music, linguistics, speech recognition, and seismology, they are often expressed as spectrograms, which have two dimensions: time and frequency. The time dimension appears naturally, whereas the frequency dimension is introduced during data representation. We will show in later Experiment 1 that spectrograms tend to be directionally correlated. Finally, we note that discrete Fourier transforms or wavelet transforms are commonly applied to signals of any dimension. Such representations are likely to create high-dimensional tensor data which are directionally correlated.

Most existing neural networks are readily applicable to directionally correlated data and may perform well. However, without taking advantage of the specific correlation structure of the data, these models will impose many unnecessary connections between neurons, leading to inefficiencies that not only impose extra computational cost but may also affect the model's performance. Here we introduce a novel neural network architecture, called SepNet, which leverages the special correlation structure within the data to design neuron connectivity, significantly increasing network efficiency. In our experiments on two representative directionally correlated datasets, SepNet was up to 100 times more efficient than current CNNs models while maintaining comparable or better performance.

2. Related work

This work is motivated by a recent study of Gao (2021), which describes a quantitative relation between the Pearson correlation coefficients of variables and their connectivity in state-of-the-art neural network architectures. Indeed, qualitative relations between correlation and connectivity can be seen from the early development of CNNs (Fukushima, 2004), and were probably the main reason behind the introduction of sliding windows in convolutional kernels (Lecun, Bottou, Bengio, & Haffner, 1998). For digital images, the sliding window centered at any given pixel is a good approximation to the (circular) region with highly correlated pixel signal intensities. In CNNs, these pixels are directly connected through a convolutional kernel. The remaining pixels are connected to the pixel of interest through later layers, roughly in the order of their correlation strengths. Comparing with fully connected neural networks in which all neurons are connected, CNNs are more efficient because they do not directly connect neurons that are not correlated.

Increasing network efficiency is of key importance. It is also the main achievement of MobileNet (Sandler et al., 2018), in which the usual convolutional operators are decomposed into depthwise convolutional operators and pointwise convolutional operators. Depthwise convolution utilizes different filters for each input channel. A single filter corresponds to one input channel, so depthwise convolution is a depth-level operation. The pointwise convolution is a usual $1 \times 1 \times \cdots \times 1$ convolution filter and creates a linear combination from the output of depthwise convolution.

In the two-dimensional case, the usual convolutional kernel has a receptive field of size $c \times h \times w$. In other words, the signals at nearby locations from all channels are connected. In the depthwise convolutional kernel of MobileNet, the receptive field is $1 \times h \times w$, meaning that only the signals at nearby locations from the same channel are connected. For pointwise convolutional operators, the receptive field is $c \times 1 \times 1$, meaning that only signals from different channels at each location are connected. Thus, MobileNet avoids a direct connection between signals at different locations from different channels.

In the d-dimensional case, MobileNet decomposes a receptive field of size $c \times m_1 \times m_2 \times \cdots \times m_d$ into two fields of sizes $1 \times m_1 \times \cdots \times m_d$ and $c \times 1 \times \cdots \times 1$, reducing the number of neuron connections from $c \times m_1 \times m_2 \times \cdots \times m_d$ to $c + m_1 \times m_2 \times \cdots \times m_d$.

The SepNet we build in this paper uses a different factorization strategy. Instead of performing depth-level operations for all dimensions, we perform operations on each dimension separately. In the d-dimensional case, we decompose the usual convolutional kernel of the receptive field $c \times m_1 \times m_2 \times \cdots \times m_d$ to $d+1$ kernels of receptive fields $c \times m_1 \times 1 \times \cdots \times 1, \ldots, c \times 1 \times \cdots \times 1 \times m_d$, and $c \times 1 \times \cdots \times 1$, reducing the total number of neuron connections from $c \times m_1 \times m_2 \times \cdots \times m_d$ to $c + m_1 + m_2 + \cdots + m_d$. Thus, SepNet is more efficient than MobileNet. The assumption of directional correlation ensures the proposed factorizations can well approximate the original operators.

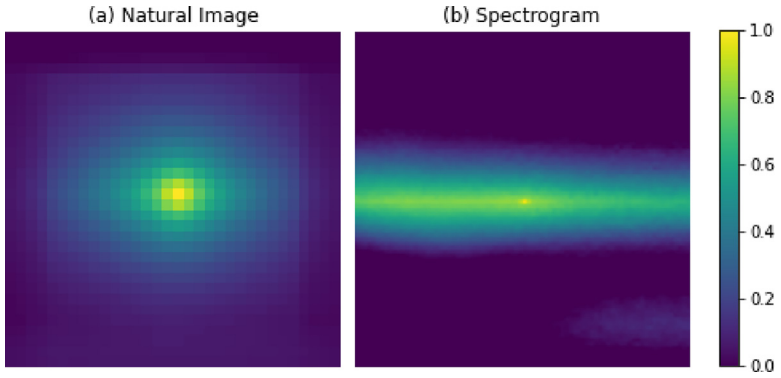

Fig. 1. Correlation matrices at an arbitrary location near the center of the images: (a) Natural images (CIFAR-10 images); (b) spectrograms of human voices saying simple words.

3. Visualization of directional correlation

In this section, we present a visual comparison of non-directionally correlated data and directionally correlated data. We use natural images (CIFAR10) and spectrograms as examples.

First, let us be more precise about the correlation of pixels in images. For two pixels at locations $P_i = (x_i, y_i)$ and $P_j = (x_j, y_j)$ in a stack of N gray-scale images of the same size $m \times n$, let v_i and v_j be the signal intensity of the N images at these two pixels. Thus, v_i and v_j are two N-dimensional vectors. We can compute the Pearson correlation coefficient ρ_{ij} of the vectors v_i and v_j as

$$\rho_{ij} = \frac{\sum_{k=1}^{N}[v_i(k) - \overline{v_i}][v_j(k) - \overline{v_j}]}{\sqrt{\sum_{k=1}^{N}[v_i(k) - \overline{v_i}]^2}\sqrt{\sum_{k=1}^{N}[v_j(k) - \overline{v_j}]^2}},$$

where $\overline{v_i} = \frac{1}{N}\sum_{k=1}^{N} v_i(k)$. For convenience, we call ρ_{ij} the correlation between the points P_i and P_j. The coefficients ρ_{ij} form an $(m\times n)\times(m\times n)$ matrix, which we call the Pearson correlation matrix. For most images, negative coefficients are rare and insignificant. Nevertheless, we take the absolute value $c_{ij} := |\rho_{ij}|$, and call the resulting matrix (c_{ij}) the correlation matrix.

For RGB images, we can either calculate the correlation matrix for each color, and take the average, or treat an RGB image as 3 separate gray-scale images and calculate the correlation matrix using $3N$ gray-scale images.

We could visualize the correlation matrix by showing it as an $(m\times n)\times(m\times n)$ image. Doing so, however, would make it difficult to interpret. Thus, we choose to visualize each row separately. Pick an arbitrary row, say the kth row, where k can be expressed as $x_k n + y_k$ with $1 \le x_k < m$ and $1 \le y_k \le n$. We reshape the kth row of the correlation matrix into an $m \times n$ matrix. In this $m \times n$ matrix, the entry at (x_i, y_i) is the correlation between the pixels at (x_k, y_k) and (x_i, y_i). Fig. 1a shows the correlations between the pixel at the center of the bright circle and other pixels in the image window of the CIFAR dataset.

If we chose a different row, the center (x_k, y_k) will move, and the bright circle will move. However, the circular pattern is largely conserved.

In comparison, Fig. 1b shows the local correlation of the spectrograms of 4000+ human voices saying 20 simple words (see the second experiment for details). Directional correlation is clearly visualized in the bright horizontal band.

Horizontal bands, vertical bands and crosses are typical features of directionally correlated data. Generally speaking, the bright region could be a product set of the form $\{(x_1, x_2, \ldots, x_d) : x_1 \in A_1, x_2 \in A_2, \ldots, x_d \in A_d\}$. However, multiple bands in a given direction would imply some remote correlation (called teleconnection in geoscience). In this paper, we will focus on the following three cases: (1) a narrow band or a narrow cross across the entire dimension; (2) a long narrow strip; (3) a thin cross. We will design a neural network model, called SepNet, specifically for such data. Our experiment 2 showed that SepNet may also applicable to multiple bands. However, we believe data with more genera remote connections are better handled using graph neural networks.

4. Design and method

The guiding principle of our design is that highly correlated variables should be more directly connected in neural networks. To make this point clear, we consider the popular benchmark image classification dataset CIFAR10. Each image has a total of 1024 pixels, which we view as 1024 variables. Considering the signal intensities for all N RGB images at each pixel, we have 1024 sequences of length $3N$. The Pearson correlation coefficients of these 1024 sequences form a 1024×1024 matrix which is visualized in Fig. 2-left. For comparison, we look at the pairwise graph distance D of the 1024 pixels in VGGNet (Szegedy et al., 2017). D is a matrix of size 1024×1024. The graph of the normalized matrix $e^{-D/8}$ is visualized in Fig. 2-right.

As we can see from Fig. 2, highly correlated variables are more directly connected in VGGNet. In fact, the same connections can be observed in other well-performing neural networks.

Now, we use this guiding principle to design our neural network architecture for directionally correlated multidimensional tensor data.

The distinct feature of directional correlation is that the highly correlated region is a union of some narrow bands across an entire dimension, or narrow strips widely spread out along a specific dimension. To better connect pixels from such regions, we introduce directional linear operators and directional group convolutional operators. These operators extract directional features from each dimension. Then, we introduce two types of basic operators to generate high-level features from these directional features. Finally, we present an overall architecture of putting these basic operators together.

Given a d-dimensional data tensor X of the shape $m_1 \times m_2 \times \cdots \times m_d$, we express X as a tensor of shape $m_0 \times m_1 \times m_2 \times \cdots \times m_d$ with $m_0 = 1$. The added channel dimension m_0 is reserved for storing extracted directional features.

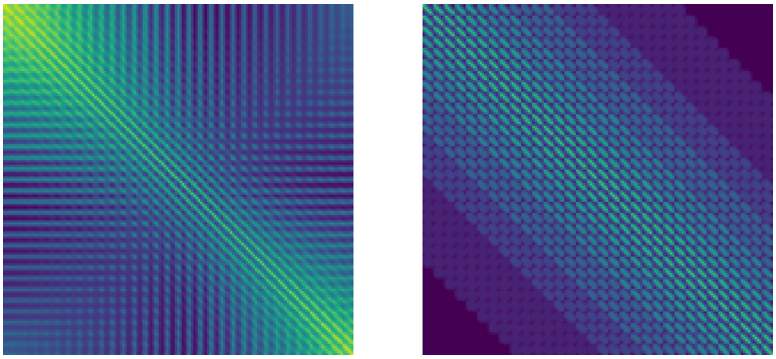

Fig. 2. Relationship between the correlation of variables and their connectivity in neural networks. Left: correlation of variables. Right: connectivity of nodes in VGGNet.

4.1. Directional linear operator

Given an $m_0 \times m_1 \times m_2 \times \cdots \times m_d$ tensor X, for $k = 1, \ldots, d$, we define a directional linear operator in kth dimension of X as a linear (affine) operator that maps each m_k-dimensional column vector to an m'_k-dimensional new vector. If we denote by $L_k(X)$ the image of X under this operator, then $L_k(X)$ is of shape $m_0 \times m_1 \times \cdots \times m_{k-1} \times m'_k \times m_{k+1} \times \cdots \times m_d$. Moreover, if we denote by S_k the linear operator that swaps the kth index of X with the last index, then we can express $L_k(X) = S_k(S_k(X)W + b)$ using matrix products, where W is a matrix so that the product $S_k(X)W$ makes sense, and b is the bias of the affine operator.

If L_k is a directional linear operator that maps a tensor X to Y, an r-fold directional linear operator L^r_k maps X to r independent copies of Y and concatenates them in the reserved channel dimension. Since each copy of Y is of the shape $m_0 \times m_1 \times \cdots \times m_{k-1} \times m'_k \times m_{k+1} \times \cdots \times m_d$, the image of X under L^r_k will have shape $rm_0 \times m_1 \times \cdots \times m_{k-1} \times m'_k \times m_{k+1} \times \cdots \times m_d$. When we need to specify the size of the operator, we will denote L^r_k by $L^r_k[m_k, m'_k]$.

We also introduce a directional linear operator in the channel dimension, and denote it by L_0. This is just the usual $1 \times 1 \times \cdots \times 1$ d-dimensional convolutional operator. We reserve L_0 for generating high-level features.

4.2. Directional group convolutional operator

In 2D CNNs, a convolutional operator is defined as

$$T : (x_{cij})_{m_0 \times m \times n} \mapsto (y_{cij})_{m'_0 \times p \times q},$$

where

$$y_{cij} = \sum_{s,t \in R(i,j)} \sum_{r=1}^{m_0} (w_{rst} x_{rst} + b_c)$$

for a rectangular region $R(i, j)$ which is centered at (i, j) with fixed size; and $\sum_{s,t \in R(i,j)}$ denotes summing over the set

$$\{(s, t) : s, t \in R(i, j), s \in \{1, 2, \ldots, m\}, t \in \{1, 2, \ldots, n\}\}.$$

For the $m_0 \times m_1 \times m_2 \times \cdots \times m_d$ tensor X, and for $k = 1, 2, \ldots, d$, we introduce a directional group convolutional operator C_k as a two-dimensional convolutional operator applied to each two-dimensional slice of X along the channel dimension and the dimension k. Each slice is of the shape $m_0 \times m_k$. If g is an integer which divides m_0, and w, s are positive integers such that s divides $m_k - w$, then the $m_0 \times m_k$ rectangle can be covered by sliding windows of size $g \times w$ with stride (g, s). After applying C_k to each slice along the channel dimension and the dimension k, $C_k(X)$ will have the shape $(m_0/g) \times m_1 \times \cdots \times m_{k-1} \times [(m_k - w)/s + 1] \times m_{k+1} \times \cdots \times m_d$.

Two extreme cases are when $g = 1$ or $g = m_0$. Clearly, if $g = m_0$, then this directional group convolutional operator is just the usual convolutional operator applied to the channel dimension and dimension k. If $g = 1$, then C_k is the depthwise convolutional operator of kernel size $(1, \ldots, 1, w, 1 \ldots, 1)$. When m_0 is small, we suggest using $g = m_0$; when m_0 is large, we may use $g \approx \sqrt{m_0}$ for efficiency.

An r-fold directional group convolutional operator C^r_k is an operator that concatenates the images of r independent copies of C_k in the channel dimension. If the output of C_k has m channels, then the output of C^r_k will have $r \times m$ channels, and total number of output channels from g groups of operators C^r_k is gr. When we need to specify w, s, and g, we will denote C^r_k by $C^r_k[w, s, g]$.

4.3. Basic operators

Both the directional linear operators and directional group convolutional operators act on each dimension separately. In order to mix the information from different dimensions, these operators need to be composed with a directional linear operator in the channel dimension. Thus, after the operator L^r_k or C^r_k, we apply an activation function, followed by the operator L_0. We denote the composition operator by $L_0 \circ C^r_k$ and $L_0 \circ L^r_k$ respectively, and call them the basic operators. These compositions also provide flexibility for changing the shape of the output tensor. Fig. 3 illustrates the basic operator $L_0 \circ L^r_k$ when $d = 2$.

4.4. Overall architecture

Starting from an input tensor, SepNet transforms the shape of the tensor through a sequence of basic operators $L_0 \circ L^r_k$ or $L_0 \circ C^r_k$, which are determined by the correlation structure of the dataset. Fig. 4 illustrates the overall architecture of the neural network.

5. Selecting operators based on correlation

SepNet can flexibly change the shape of the tensors in the middle layers. The main decision is choosing L^r_k or C^r_k for each k. The general guideline is that if the high correlation region is spread cross the entire kth dimension, then L^r_k or C^r_k with a large kernel should be used. If the high correlation region is localized in the kth dimension, then C^r_k with small kernels are more appropriate.

If C^r_k is used, we also need to choose the size of the convolutional kernel. Here we introduce a general approach. For clarity, we present the method for the case of two-dimensional tensors of shape $m \times n$, but the method naturally extends to higher-dimensional cases.

Using the method described in §3, for each location (x, y) in $m \times n$ images, we can compute the $m \times n$ correlation matrix

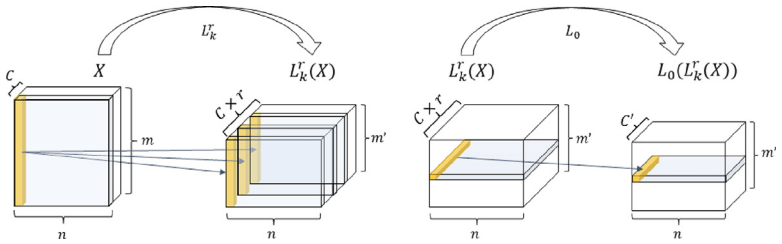

Fig. 3. The basic operators $L_0 \circ L_k^r$ in SepNet. The operator L_k^r acts on the directional dimension k; the operator L_0 acts on the channel dimension.

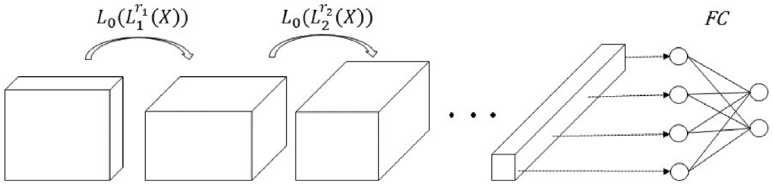

Fig. 4. The overall architecture of SepNet. In each step, though a basic operator, the size in one of the dimensions decreases, while the number of channels increases. Finally, the tensor is flatted to a vector and fed to a fully-connected layer.

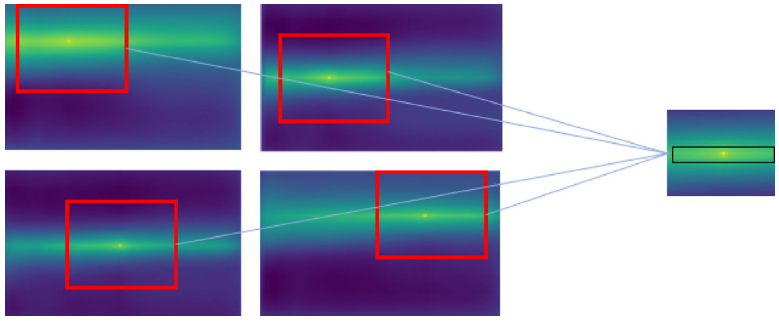

Fig. 5. The process of determining the rectangular region.

at (x, y). The high correlation region of the matrix is a narrow band/strip/cross (cf. Fig. 1b) centered at (x, y). As (x, y) moves over the $m \times n$ image, the narrow band/strip/cross moves with it. The bands/strips/crosses for different (x, y) are not identical but nonetheless have a high level of similarity. In order to compute the average of these bands across different (x, y), we take a $(m/2) \times (n/2)$ window centered at (x, y), where only (x, y) values with corresponding windows that are contained entirely within the image are considered. The average of these $(m/2) \times (n/2)$ windows approximates the overall correlation structure of the data. Next, we select a rectangular region that best represents the high-correlation region within this average $(m/2) \times (n/2)$ window (Fig. 5). The size of this rectangular region determines what operators we will use.

Suppose the rectangular region is of size $a \times b$. If $a < m/2$, we use a directional convolutional operator in dimension 1 with kernel size $a/2$. If $a \approx m/2$, then we use either a directional convolutional operator with kernel size $m/4$ in dimension 1, or a directional linear operator in dimension 1. A similar approach is used for dimension 2.

Finally, while Pearson correlation coefficients are commonly used and easy to compute, other correlations could also be used. For example, one may replace Pearson correlation with distance correlation (Székely & Rizzo, 2009; Székely, Rizzo, & Bakirov, 2007), which is applicable to vectors of any dimension, as well as to categorical variables.

6. Experiments

In this section, we use two representative datasets to demonstrate the effectiveness of SepNet, and we compare SepNet with other popular baseline models. We also use these experiments to explain how to choose directional linear operators or directional convolutional kernels based on correlation structures.

6.1. Voice recognition data

Data preparation. Spectrograms are commonly used in speech recognition. The dataset we used contains spectrograms generated from the wave files of the Kaggle TensorFlow Speech Recognition Challenge Dataset (Warden, 2018). The original dataset contains wave files of human voices speaking 30 simple words, and some background noises. To minimize the influence caused by data imbalance and noise, we selected the wave files of 20 representative words: 'yes', 'no', 'up', 'down', 'left', 'right', 'on', 'off', 'stop', 'go', 'zero', 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', and 'nine'. This gave a total of 47,348 wave files, each of length 1 s with sample rate of 16,000 per second. These files were roughly evenly distributed among the 20 classes. After converting each wave file into logarithmic spectrograms, using segment lengths of 320 with an overlap 160 between adjacent segments, we transformed each wave into a 99×161 gray-scale image, with 99 dimensions in time and 161 dimensions in frequency. After this preparation, the input tensor is of the shape $1 \times 99 \times 161$.

Determining kernel size. To construct the SepNet model for this dataset, we first examined the Pearson correlation coefficients among the pixels. As is typical for spectrograms, the correlation matrix at any given point in the image reveals the presence of directional correlation. Along the frequency dimension, the pixels

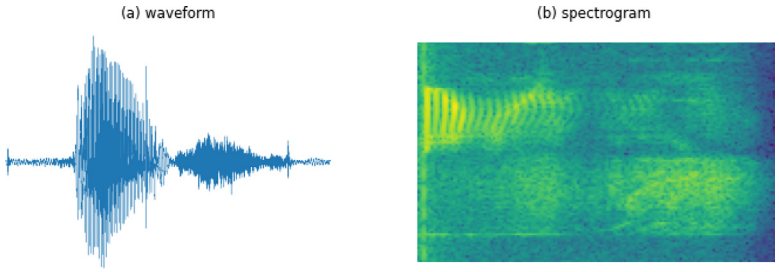

Fig. 6. The waveform and spectrogram of a voice of the word "yes".

Table 1
SepNet-L1 architecture.

Layer	Basic operator	Additional operator	Output shape
1	$L_0[84, 81] \circ L_2^4[161, 21]$		$81 \times 99 \times 21$
2	$L_0[81, 100] \circ C_1^9[9, 3, 9]$		$100 \times 31 \times 21$
3	$L_0[100, 200] \circ C_1^{10}[7, 2, 10]$		$200 \times 13 \times 21$
4	$L_0[300, 300] \circ L_2^{300}[21, 1]$		$300 \times 13 \times 1$
5	$L_0[300, 400] \circ C_1^{15}[4, 1, 15]$	AvgPool1d(2)	$400 \times 5 \times 1$
6	$L_0[400, 512] \circ C_1^{20}[4, 1, 20]$	AvgPool1d(2)	$512 \times 1 \times 1$

Table 2
SepNet-C1 architecture.

Layer	Basic operator	Additional operator	Output shape
1	$L_0[81, 81] \circ C_2^{81}[41, 4, 1]$	Dropout(0.4)	$81 \times 99 \times 31$
2	$L_0[81, 100] \circ C_1^9[4, 1, 9]$	MaxPool1d(2)	$100 \times 48 \times 31$
3	$L_0[100, 200] \circ C_1^{10}[4, 2, 10]$		$200 \times 23 \times 31$
4	$L_0[200, 300] \circ C_2^{10}[4, 1, 10]$	AvgPool1d(2)	$300 \times 23 \times 14$
5	$L_0[300, 400] \circ C_1^{10}[4, 1, 10]$	AvgPool1d(2)	$400 \times 10 \times 14$
6	$L_0[400, 500] \circ C_2^{20}[5, 1, 20]$	MaxPool1d(2)	$500 \times 10 \times 5$
7	$L_0[500, 500] \circ C_1^{20}[3, 1, 20]$	AvgPool1d(2)	$500 \times 4 \times 5$
8	$L_0[500, 512] \circ C_2^{20}[3, 1, 20]$		$512 \times 4 \times 3$
9	$L_0[512, 512] \circ C_1^{32}[3, 1, 32]$	MaxPool1d(2)	$512 \times 1 \times 3$
10	$L_0[512, 512] \circ C_2^{32}[3, 1, 32]$	Dropout(0.5)	$512 \times 1 \times 1$

at the same segment but different frequencies were highly correlated across almost all 160 dimensions, whereas along the time dimension, pixel intensities were highly correlated only locally. In fact, at almost any location, the highly correlated pixels were distributed in a narrow band parallel to the frequency dimension. Fig. 6a illustrates the waveform of a human voice speaking the word "yes", which only has a time dimension. Fig. 6b illustrates its spectrogram, which adds the frequency dimension (horizontal dimension). Fig. 1b in Section 3 indicates it is indeed directionally correlated.

With the method discussed in Section 5, we found the high-correlation rectangular region is of size about 6×80. This suggested that in dimension 1, we should use directional convolutional operators with smaller kernels of size 3 or 4. It also suggested that in dimension 2, we use either directional convolutional operators with larger kernels of size close to 40, or directional linear operators.

Architecture and hyperparameters. For comparison, we designed two SepNet models. The first model (SepNet-L1) uses directional linear operators in frequency dimension with the architecture in Table 1:

The second model (SepNet-C1) uses directional group convolutional operators with larger kernel in the frequency dimension. Table 2 gives the architecture of SepNet-C1.

In both models, the activation function used in the basic operator was ReLU. Batch normalization and Softplus activation were used between the basic operator and the additional operator (if applicable). Finally, a fully-connected layer was applied at the end to output a 20-dimensional vector.

Table 3
Comparison of SepNet with state-of-the-art models on spectrograms.

Model	Accuracy	Parameters	FLOP
MobileNet-v2	93.15%	3.5 M	0.31 G
VGG16	94.86%	138.4 M	15.52 G
ResNet50	**96.22%**	25.6 M	4.14 G
EfficientNetB0	92.4%	5.3 M	0.41
SepNet-L1	94.81	**0.77 M**	**0.06 G**
SepNet-C1	95.43%	1.47M	0.18 G

We trained the SepNet-L1 and SepNet-C1 using batch size 32 for 30 epochs with 5-fold cross-validation. The RMSprop optimizer at learning rate 0.0003 was used to minimize the cross-entropy loss. The learning rate was scheduled to reduce by a factor of 0.5 every 8 epochs.

Results. We next compared the performance of SepNet with four state-of-the-art image classification models: MobileNetV2, Vgg16, ResNet50, and EffienctNetB0. Among these models, MobileNetV2 was most efficient and converged fastest, but EfficientNetB0, which was based on the inverted bottleneck residual blocks of MobileNetV2, unexpectedly did not converge as quickly. VGG16 and ResNet50 had better performance but used substantially more parameters. These results suggested that the performance of SepNet was comparable with the best of these models, while having the simplest architecture and the fewest parameters (Table 3).

What if. In both SepNet-L1 and SepNet-C1, the sizes of the directional convolutional kernels in dimension 1 were small and were based on the width of the high-correlation band. While the band width could vary several pixels depending on the threshold we picked, we found that the performance of the networks were not particularly sensitive to the kernel sizes as long as the sizes were reasonably close to the estimate. However, we also found that if the sizes of the kernel were way off, the performance could be very poor. In SepNet-L1, we tested the opposite strategy of using directional linear operator in dimension 1, and directional convolutional operators with small kernels in dimension 2, resulting in a network that performed poorly and converged very slowly. In fact, with the same hyperparameters, the test accuracy only reached 69% after 30 epochs.

6.2. Physical activity dataset

Our second experiment used the PAMAP2 Physical Activity Monitoring dataset (Reiss & Stricker, 2012). PAMAP2 describes 18 different physical activities performed by 9 different subjects (one female and eight males) wearing 3 inertial measurement units and a heart rate monitor. These data were represented as 3D-time series.

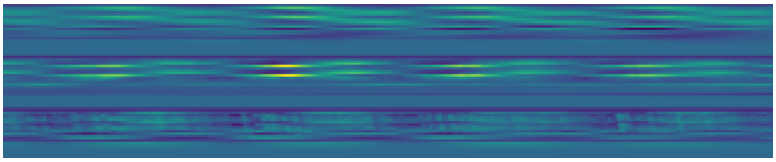

Fig. 7. The PAPAM2 data reveals relatively high correlation in three horizontal bands.

Table 4
SepNet-L2 architecture.

Layer	Basic operator	Output shape
1	$L_0[4, 4] \circ L_2^2[52, 40] \circ L_1^2[256, 40]$	$4 \times 40 \times 40$
2	$L_0[4, 8] \circ L_2^2[40, 30] \circ L_1^2[40, 30]$	$8 \times 30 \times 30$
3	$L_0[8, 16] \circ L_2^2[30, 16] \circ L_1^2[30, 16]$	$16 \times 16 \times 16$
4	$L_0[16, 32] \circ L_2^2[16, 8] \circ L_1^2[16, 8]$	$32 \times 8 \times 8$
5	$L_0[32, 64] \circ L_2^2[8, 4] \circ L_1^2[8, 4]$	$64 \times 4 \times 4$
6	$L_0[64, 128] \circ L_2^2[4, 2] \circ L_1^2[4, 2]$	$128 \times 2 \times 2$

Table 5
SepNet-C2 architecture.

Layer	Basic operator	Additional operator	Output shape
1	$L_0[81, 81] \circ L_1^{81}[256, 20]$	Dropout(0.1)	$81 \times 20 \times 52$
2	$L_0[81, 100] \circ C_2^9[4, 1, 9]$		$100 \times 20 \times 49$
3	$L_0[100, 100] \circ C_2^{10}[6, 1, 10]$		$200 \times 20 \times 44$
4	$L_0[100, 300] \circ L_1^{100}[20, 1]$	MaxPool2d(2)	$300 \times 1 \times 22$
5	$L_0[300, 400] \circ L_2^{300}[22, 1]$		$400 \times 1 \times 1$

Data preparation. In order to directly compare our results with the best-known model for this dataset, MC-DNN (Zheng, Liu, Chen, Ge, & Zhao, 2014), we limited our analysis to a subset of the data to match that used in MC-DNN. In particular, we (1) chose the same 4 physical activities (among the 18 total) for analysis: 'standing', 'walking', 'ascending stairs', and 'descending stairs'; (2) excluded the same two subjects previously excluded (*e.g.* on the basis of using a different dominant hand or foot); and (3) reduced the sampling rate of each time series from the original 10 ms to 2.56 s via averaging. The resulting data subset consisted of 52 reduced 3D-time series. For each physical activity, we normalized the data using the transform: $x \mapsto (x - \mu)/\sigma$, where μ and σ are the mean and standard deviation, respectively. Finally, these time series were divided into smaller-sized batches to facilitate model training. After this preparation, the input tensor for SepNet is of the shape $1 \times 256 \times 52$.

Determining kernel size. A correlation analysis of the input tensor revealed a stereotypical banding pattern (Fig. 7). The high-correlation region roughly consists of three sets of bands. Along dimension 1 (the horizontal direction in the figure), these bands spread out across almost the entire dimension. Thus, we use directional linear operators. In dimension 2 (the vertical direction in the figure), they roughly occupy three distinct zones, indicating the presence of teleconnections. For this dimension, we evaluated two different strategies: (1) directional linear operators, as in dimension 1, and (2) alternating use of directional convolutional operators and directional linear operators. These two models were named SepNet-L2 and SepNet-C2, respectively, and their architectures are specified below.

Architecture and hyperparameters. SepNet-L2 uses directional linear operators, permuted among the two dimensions and the channel dimension, with the following architecture (see Tables 4 and 5): ReLU activation was used before each composition operator and after each layer. Finally, a maximal pooling is used, followed by a fully-connected layer.

By contrast, SepNet-C2 uses linear operators in dimension 1, but mixes use of linear and convolutional layers in dimension 2, with the following architecture: In layers 1, 4, and 5, the Seagull (Gao & Zhang, 2022) activation function $\log(1 + x^2)$ was used before the composition operator. In layers 2–3, ReLU activation was used before the composition operator, and batch normalization and Softplus were used after the composition operator. Finally, a fully connected layer was used at the end.

We trained both models using batch size 1 for 100 epochs, with early stopping and 20-fold cross validation. Cross entropy loss and an Adam optimizer with learning rate 0.0001 were used. For SepNet-L2, we used the consecutive weight regularization strategy introduced in Gao (2022) with weight decay patience 10. For SepNet-C2, we used L^2 weight regularization with patience 10. The early stopping patience for SepNet-C2 was adjusted to 20 such that the training process stopped around 50 epochs for both models.

Results. SepNet-L2 achieved a 98.10% test accuracy, and SepNet-C2 achieved a 96.16% test accuracy. Both models performed significantly better than all models discussed in MC-DNN (Zheng et al., 2014), in which the five best models 1-NN-DTW-%5, MLP, 1-NN-ED, MC-DCNN1, and MC-DCNN2 have accuracies of 83.61%, 84.83%, 82.28%, 90.53%, and 93.36%, respectively. The model 1-NN-ED was built based k-Nearest Neighbor classification (Batista, Wang, & Keogh, 2011), and 1-NN-DTW-%5 combined k-Nearest Neighbor classification with Dynamic Time Warping (Rakthanmanon, Campana, Mueen, Batista, Westover, Zhu, Zakaria, & Keogh, 2012). Thus, we could not evaluate the efficiency of the models by simply comparing the number of parameters with SepNet models. MPL was a multi-layer perceptron model, and it did have simpler architecture than SepNet, but it performed the worst. The MC-DCNN1 and MC-DCNN2 (Zheng et al., 2014) were multi-channels deep convolution neural networks models in which each channel of which takes a single dimension of multivariate time series as input and learns features individually. Then the MC-DCNN models combined the learnt features of each channel and feeds them into a MLP to perform classification finally. The construction was a bit more complicated than SepNet. The number of parameters in MC-DCNNs would be comparable to that of SepNet, but we had difficulties to accurately estimate these numbers for the models in Zheng et al. (2014). Thus, instead of comparing the number of parameters and FLOP as we did in Experiment 1, we report our the learning curve in Fig. 8, as comparison to Figure 5 of Zheng et al. (2014).

7. Discussion

SepNet uses an empirical correlation matrix to design a neural network architecture that enables substantially improved efficiency for applications of practical interest. Our results suggest a general guideline for best performance: (1) use of directional linear operators or directional group convolutional operators with larger kernels in the dimensions where the region of high correlation is spread out, and (2) use of directional group convolutional operators with small kernels in the dimensions where the correlation is localized. A violation of this guideline will generally lead to poor performance.

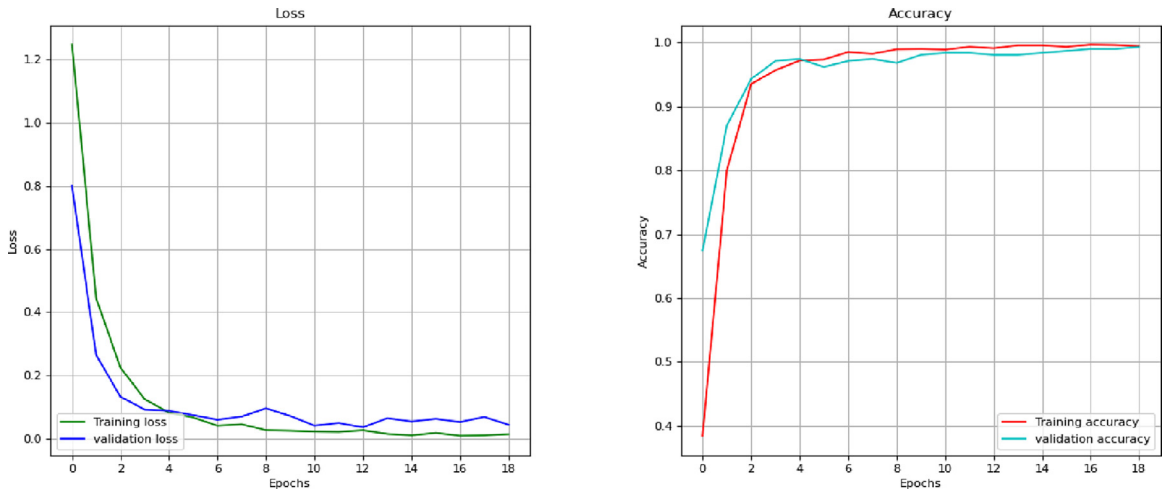

Fig. 8. Learning curve of SepNet-L2 in experiment 2 on PAMAP2 data.

SepNet performs well for data in which the high-correlation region is a union of vertical or horizontal bands (cf Experiment 2). It may also work well when the high-correlation region is more generally of the form $\{(x_1, x_2, \ldots, x_d) : x_i \in A_i, 1 \leq i \leq d\}$, where A_i are one-dimensional sets. However, for data with more complicated teleconnections, SepNet is not likely to perform well, and in this case, we believe that better performance can be achieved using GNNs that are designed according to the correlation structure of the specific data. Finally, as SepNet requires correlation information, it will be valuable to develop a variant of SepNet that can dynamically adapt its architecture during training, so that it can be used for online learning where empirical correlation is not accessible at the beginning, but can be better estimated during training.

8. Conclusion

Current deep neural network models have high computational complexity. Our method develops the most efficient deep neural network framework (SepNet) for multidimensional tensor data, which performs well on directionally correlated data, a special yet very common type of data. The efficiency improvement enabled by SepNet highlights the potential of architecting neural networks according to specific structure information within the data.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by National Science Foundation, United States of America grant OAC1940270, National Science Foundation, United States of America grant #2019609, and NIH, United States of America grant P20GM104420.

References

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473.

Batista, G., Wang, X., & Keogh, E. (2011). A complexity-invariant distance measure for time series. In *Proceedings of the 11th SIAM international conference on data mining, SDM 2011* (pp. 699–710). http://dx.doi.org/10.1137/1.9781611972818.60.

Cho, H., & Choi, I. S. (2018). Three-dimensionally embedded graph convolutional network (3DGCN) for molecule interpretation. arXiv abs/1811.09794.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1800–1807).

Fukushima, K. (2004). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*, 193–202.

Gao, F. (2021). Relation between variable correlation and neural connectivity. (Preprint).

Gao, F. (2022). Ae^{2}I: A double autoencoder for imputation of missing values. *Transactions on Computational Science & Computational Intelligence*, (in press).

Gao, F., & Zhang, B. (2022). Data-aware customization of activation functions reduces neural network error. *Transactions on Computational Science & Computational Intelligence*, (in press).

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). *Identity mappings in deep residual networks, Vol. 9908* (pp. 630–645). http://dx.doi.org/10.1007/978-3-319-46493-0_38.

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 770–778).

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700–4708).

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. http://dx.doi.org/10.1109/5.726791.

Lee, J., Bang, J., & Yang, S.-I. (2017). Object detection with sliding window in images including multiple similar objects. In *2017 international conference on information and communication technology convergence (ICTC)* (pp. 803–806).

Minami, K.-i., Nakajima, H., & Toyoshima, T. (1999). Real-time discrimination of ventricular tachyarrhythmia with Fourier-transform neural network. *IEEE Transactions on Biomedical Engineering*, *46*(2), 179–185.

Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. *Advances in Neural Information Processing Systems*, *3*.

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., et al. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings. International Conference on Knowledge Discovery and Data Mining 2012* (pp. 262–270). https://eurekamag.com/research/069/459/069459412.php.

Reiss, A., & Stricker, D. (2012). Creating and benchmarking a new dataset for physical activity monitoring. In *PETRA '12*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. http://dx.doi.org/10.1038/323533a0, URL http://www.nature.com/articles/323533a0.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 4510–4520). http://dx.doi.org/10.1109/CVPR.2018.00474.

Sanyal, S., Balachandran, J., Yadati, N., Kumar, A., Rajagopalan, P., Sanyal, S., et al. (2018). MT-CGCNN: Integrating crystal graph convolutional neural network with multitask learning for material property prediction. arXiv abs/1811.05660.

Schmidt, J., Pettersson, L., Verdozzi, C., Botti, S., & Marques, M. A. L. (2021). Crystal graph attention networks for the prediction of stable materials. *Science Advances*, *7*(49), eabi7948. http://dx.doi.org/10.1126/sciadv.abi7948.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-ResNet and the impact of residual connections on learning. In *AAAI'17*, *Proceedings of the thirty-first AAAI conference on artificial intelligence* (pp. 4278–4284). AAAI Press.

Székely, G. J., & Rizzo, M. L. (2009). Brownian distance covariance. *The Annals of Applied Statistics*, *3*, 1236–1265.

Székely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, *35*, 2769–2794.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008). URL http://arxiv.org/abs/1706.03762.

Wan, M., Chen, X., Zhan, T., Xu, C., Yang, G., & Zhou, H. (2021). Sparse fuzzy two-dimensional discriminant local preserving projection (SF2DDLPP) for robust image feature extraction. *Information Sciences*, *563*, http://dx.doi.org/10.1016/j.ins.2021.02.006.

Wan, M., Lai, Z., Yang, G., Yang, Z., Zhang, F., & Zheng, H. (2016). Local graph embedding based on maximum margin criterion via Fuzzy Set. *Fuzzy Sets and Systems*, *318*, http://dx.doi.org/10.1016/j.fss.2016.06.001.

Wan, M., Yao, Y., Zhan, T., & Yang, G. (2022). Supervised low-rank embedded regression (SLRER) for robust subspace learning. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(4), 1917–1927. http://dx.doi.org/10.1109/TCSVT.2021.3090420.

Warden, P. (2018). Speech commands: A dataset for limited-vocabulary speech recognition.

Weston, J., Chopra, S., & Bordes, A. (2015). Memory networks. CoRR abs/1410.3916.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24. http://dx.doi.org/10.1109/TNNLS.2020.2978386.

Xia, X., Xu, C., & Nan, B. (2017). Inception-v3 for flower classification. In *2017 2nd international conference on image, vision and computing (ICIVC)* (pp. 783–787). http://dx.doi.org/10.1109/ICIVC.2017.7984661.

Xie, S., Girshick, R. B., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 5987–5995).

Xie, T., & Grossman, J. C. (2018). Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Physical Review Letters*, *120 14*, Article 145301.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., et al. (2015). Show, attend and tell: Neural image caption generation with visual attention. In F. Bach, D. Blei (Eds.), *Proceedings of machine learning research*: *vol. 37*, *Proceedings of the 32nd international conference on machine learning* (pp. 2048–2057). Lille, France: PMLR, URL https://proceedings.mlr.press/v37/xuc15.html.

Yu, W., Teng, X., & Liu, C. (2006). Face recognition using discriminant locality preserving projections. *Image and Vision Computing*, *24*(3), 239–248. http://dx.doi.org/10.1016/j.imavis.2005.11.006, URL https://www.sciencedirect.com/science/article/pii/S0262885605002039.

Zhang, Q., & Benveniste, A. (1992). Wavelet networks. *IEEE Transactions on Neural Networks*, *3*(6), 889–898.

Zhang, B., Wang, S., & Gao, F. (2021). Contrastive metric learning for lithium super-ionic conductor screening. (Preprint).

Zhang, B., Zhou, M., Wu, J., & Gao, F. (2022). Predicting the materials properties using a 3d graph neural network with invariant representation. *IEEE Access*, 1-1, http://dx.doi.org/10.1109/ACCESS.2022.3181750.

Zheng, Y., Liu, Q., Chen, E., Ge, Y., & Zhao, J. L. (2014). Time series classification using multi-channels deep convolutional neural networks. In *WAIM*.