# CLASSIFICATION OF LONG SEQUENTIAL DATA USING CIRCULAR DILATED CONVOLUTIONAL NEURAL NETWORKS

#### Lei Cheng, Ruslan Khalitov, Tong Yu, and Zhirong Yang\*

Department of Computer Science Norwegian University of Science and Technology {lei.cheng, ruslan.khalitov, tong.yu, zhirong.yang}@ntnu.no

#### ABSTRACT

Classification of long sequential data is an important Machine Learning task and appears in many application scenarios. Recurrent Neural Networks, Transformers, and Convolutional Neural Networks are three major techniques for learning from sequential data. Among these methods, Temporal Convolutional Networks (TCNs) which are scalable to very long sequences have achieved remarkable progress in time series regression. However, the performance of TCNs for sequence classification is not satisfactory because they use a skewed connection protocol and output classes at the last position. Such asymmetry restricts their performance for classification which depends on the whole sequence. In this work, we propose a symmetric multi-scale architecture called Circular Dilated Convolutional Neural Network (CDIL-CNN), where every position has an equal chance to receive information from other positions at the previous layers. Our model gives classification logits in all positions, and we can apply a simple ensemble learning to achieve a better decision. We have tested CDIL-CNN on various long sequential datasets. The experimental results show that our method has superior performance over many state-of-the-art approaches. The model and experiments are available at https://github.com/LeiCheng-no/CDIL-CNN.

**Keywords** classification · sequential data · convolutional neural networks

#### 1 Introduction

Sequence classification is the task of predicting class labels for sequences. It is of central importance in many applications, such as document classification, genomic analysis, and health informatics. For example, classifying documents into different topic categories is a challenge for library science, especially for modern digital libraries [1]. Genomic classification help researchers to further understand some diseases [2]. Classifying ECG time series tells if someone is a healthy person or a patient with heart disease [3].

Machine Learning, especially Deep Learning, becomes widely used in end-to-end sequence classification, where a single model learns all steps between the initial inputs and the final outputs. Recurrent Neural Networks (RNNs), Transformers, and Convolutional Neural Networks (CNNs) are three primary techniques for analyzing sequential data.

RNNs use their internal states to process the sequence step by step. Despite success for short sequences, traditional RNNs cannot scale to very long sequences [4]. One reason is that they are challenging to train due to exploding or vanishing gradient problems [5]. In addition, the prediction of each timestep must wait for all its predecessors to complete, which makes RNNs difficult to parallelize. Transformers are a family of models relying on self-attention mechanism [6, 7]. They have quadratic time and memory complexities to the input sequence length because they compute pairwise dot-products. Comprehensive approximations are required to reduce the cost [8].

In contrast, CNNs are able to handle very long sequences. A convolutional layer uses sparse connections and no recurrent nodes. Therefore, CNNs are easier to train and parallelize. In addition, dilated convolutions can exponentially enlarge the receptive fields, allowing CNNs to use fewer layers to capture long-term dependencies. For example,

<sup>\*</sup>Corresponding author.

Temporal Convolutional Networks (TCNs) recently provide remarkable performance on sequence regression tasks [9]. However, the performance of TCNs for classification tasks is not satisfactory. TCNs use causal convolutions which implement a skewed connection protocol. The asymmetric design causes a tendency to focus on the latter part of a sequence.

In this paper, we propose a novel convolutional architecture named Circular Dilated Convolutional Neural Network (CDIL-CNN), which can scale to very long sequences and have superior performance on various classification tasks. Unlike TCNs, we use symmetric convolutions to mix information, and thus every position can receive both earlier and later information from previous layers in a circular manner. Unlike conventional pyramid-like CNN architecture, every position of the last convolutional layer in our design has an equal chance to receive all information from the whole sequence and gives its classification logits. Then a simple average ensemble learning helps our model achieve better accuracy.

We have tested our model on extensive sequence classification tasks, including synthetic data, images, texts, and audio series. Experimental results show that CDIL-CNN outperforms several state-of-the-art models. Our method can accurately and robustly classify across tasks with both short-term and long-term dependencies for very long sequences.

The remaining of the paper is organized as follows. We review some popular models for sequential data and their limitations in Section 2. In Section 3, we present our model, CDIL-CNN, including its connection protocol and network architecture. Experimental tasks and results are provided in Section 4, and we discover that the simple convolutional network has superior performance over other models in various scenarios. Finally, we conclude the paper in Section 5.

#### 2 Related Works

A sequence x of length N is a list of elements  $[x_1, x_2, \cdots, x_N]$ , where  $x_t \in \mathbb{R}^D$   $(1 \le t \le N)$  is the D-dimensional element at the t-th position. Given a training set  $\{x^{(i)}, y^{(i)}\}_{i=1}^I$  with I sequences and their class labels, sequence classification uses the training set to fit a model  $f: \mathbb{R}^{N \times D} \mapsto \mathbb{C}$ , where  $\mathbb{C}$  is the space of class labels. The fitted model can then be used to classify newly coming sequences.

Many deep neural networks have been proposed for various sequence classification tasks. RNNs, Transformers, and CNNs are three significant branches for learning from sequential data.

RNNs read and process inputs sequentially. At each timestep, an RNN takes the current sequence element and the hidden state as the input and outputs the next hidden state. The hidden state at a timestep is expected to act as the representation of all its earlier inputs. Because the prediction of each timestep must wait for all its predecessors to complete, the sequential process is difficult to parallelize, which makes RNNs hard to handle very long sequences. Moreover, basic RNNs suffer from vanishing and exploding gradient problems, making model training very difficult for long sequences [5]. Gated RNNs, such as Long Short-Term Memory (LSTM) [10] and Gated Recurrent Unit (GRU) [11], have been proposed to relieve the gradient problems. They have many additional gates to regulate the flow of information. The gated RNNs are used in many sequence classification tasks, such as ECG arrhythmia [12] and text [13, 14]. However, they can process only short sequences (about 500-1000 timesteps) [4].

Transformers, a family of models based on attention mechanism, quantify the interdependence within the sequence elements (self-attention). Originally, attention was used in conjunction with recurrent networks and convolutional networks [15, 16]. Later, Transformer, an architecture based solely on attention mechanism, was proposed. The vanilla Transformer computes pairwise dot-products between all sequence elements, which leads to a quadratic complexity w.r.t. the sequence length and makes it infeasible to process very long sequences. Approximated attention methods have been proposed to tackle this problem. Sparse Transformer [17], LogSparse Transformer [18], Longformer [19], and Big Bird [20] use sparse attention mechanism. Linformer [21] and Synthesizer [22] apply low-rank projection attention. Performer [23], Linear Transformer [24], and Random Feature Attention [25] rely on kernel approximation. Reformer [26], Routing Transformer [27], and Sinkhorn Transformer [28] follow the paradigm of re-arranging sequences. However, their approximation quality is questionable. Later in Section 4, we will show that their performance is inferior for long sequence classification.

CNNs are good at processing data that has a grid-like topology. Two-dimensional CNNs achieve great success in computer vision [29, 30, 31, 32], while one-dimensional CNNs are commonly used for sequential data [33, 34, 35]. Among these models, TCNs which use causal convolutions with skewed connections attempt to capture the temporal interactions and have been applied to various regression tasks, such as action segmentation and detection [36, 37], lip-reading [38, 39], and ENSO prediction [40]. The comparison of the convolutional and recurrent architectures shows that a simple TCN outperforms canonical RNNs across a wide range of sequence modeling tasks [9].

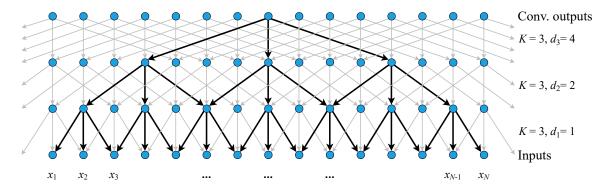


Figure 1: Illustration of symmetric dilated convolutions. Blue nodes represent the sequence elements. Each layer keeps the same size as the input sequence. Symmetric convolutions of kernel size 3 are used in all layers. Dilation sizes are increased exponentially.

#### 3 Circular Dilated CNN

Although TCN is suitable for long sequence regression, their performance for classification is not satisfactory. In this paper, we propose a new convolutional model, named CDIL-CNN, to overcome the TCN drawbacks in long sequence classification. More details are described as follows.

#### 3.1 Symmetric Dilated Convolutions

Our model uses symmetric convolutions that can receive both earlier and later information from previous layers. Because no information is allowed to be leaked from future to past in regression tasks, TCN uses causal convolutions that implement a skewed connection protocol, meaning that the output at timestep t can only receive information of t and earlier from previous layers. However, classification tasks do not have the restriction because the classification result depends on the whole sequence. Therefore, symmetric convolutions help our model better capture interactions.

Our model also uses increasing dilation sizes with the depth of the network. Dilated convolutions (or atrous convolutions) were originally introduced for dense image prediction, where they helped the model to capture multi-scale information [41, 42, 43, 44, 45]. For 1D CNNs, dilated convolutions are generally used to enlarge the receptive fields [33, 34, 37, 9]. Following these works, we increase the dilation sizes exponentially, i.e.,  $d_l = 2^{l-1}$  where  $d_l$  is the dilation size at the l-th convolutional layer. The combination of deep networks and exponentially dilated convolutions enables the receptive fields to expand quickly, which makes our model scalable to very long sequences. Our model needs  $\lceil \log_2 \frac{N}{2} \rceil$  or  $O(\log_2 N)$  layers to achieve a full receptive field for sequence length N.

To avoid notional clutter, we start from the D=1 case. Let  $[a_1,a_2,\cdots,a_N]$  denotes an 1-dimensional input sequence of the l-th convolutional layer. The convolutional output  $b_t$  at the t-th  $(1 \le t \le N)$  position is computed by  $b_t = \sum_{k=0}^{K-1} w_k^{(l)} \cdot a_{t+\left(k-\frac{K-1}{2}\right)\cdot d_l}$ , where the kernel size K is usually an odd number and  $w^{(l)}$  are the convolution coefficients of the l-th layer. See Figure 1 for an illustration of a 3-layer symmetric dilated convolutions with K=3. It is straightforward to extend the convolution with the bias term and for the D>1 cases.

#### 3.2 Circular Mixing

In traditional CNNs, zero-padding is often used for the boundary positions where the subscripts of their convoluted input positions  $\left[t+(k-\frac{K_l-1}{2})\cdot d_l\right]$  are smaller than 1 or larger than N. However, this can cause boundary effect because signals near the boundaries have to be mixed up with zeros and thus have less chance to be forwarded. The boundary effect creates blind spots and makes CNNs sensitive to absolute positions [46, 47]. For example, CNNs with zero-padding can fail to capture the useful patterns if translation exists in the test data but not in the training data (see Section 4.4).

We use a circular protocol because its corresponding circular padding can relieve the boundary effect [46, 47]. In our model, a signal on one end is no longer convoluted with zeros but with signals from the other end. Circular padding makes our model more robust to data shift and less sensitive to absolute position information. The circular dilated

<sup>&</sup>lt;sup>2</sup>We used K = 3 in all our experiments.

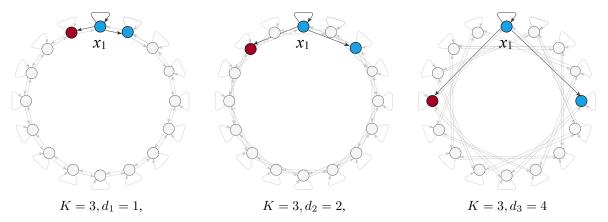


Figure 2: Illustrations of circular mixing. (a), (b), and (c) are the first, second, and third convolutional layer, respectively. Red nodes represent convoluted positions from the other end.

convolutions are shown in Figure 2. The convolutional output  $b_t$  becomes

$$b_t = \sum_{k=0}^{K-1} w_k^{(l)} \cdot a_{\left[t + \left(k - \frac{K-1}{2}\right) \cdot d_l\right] \bmod N}$$
 (1)

Using circular dilated convolutions, our model can connect boundary positions and learn long-term dependencies even in the first layer, unlike lower layers of traditional CNNs which only focus on local information. In our design, every position of the last convolutional layer has an equal chance to receive all information of the whole input sequence. Therefore, our model can apply a simple average ensemble learning as below.

#### 3.3 Ensemble Learning

We use a simple average ensemble learning to achieve better performance. RNNs and TCN assume that the last position contains all information of the whole sequence and the class decision depends only on the last position. In our model, every position of the last convolutional layer can receive all information of the whole sequence. A linear module  $\mathbb{R}^C \mapsto \mathbb{C}$ , where C is the number of convolution channels, is applied on each convolutional output position, and each position gives its preliminary class logits. Then a simple average pooling as ensemble learning aggregates the individual logits. In the implementation, we can perform the average first to speed up the network because the linear module and the average pooling are exchangeable.

Our model also uses residual connections to facilitate the training and to improve the accuracy [48, 49]. A residual block contains a skip connection where the inputs are added before the block outputs. A schematic view of our model is depicted in Figure 3.

#### 4 Experiments

We have compared our model with many popular models (including RNNs, Transformers, and CNNs) on various long sequential datasets in three groups of experiments. First, we used a synthetic dataset with increasing sequence lengths to show the scalability of our model. Then, we tested our model on the Long Range Arena (LRA) benchmark suite which contains different dependencies. Finally, we tried three time series classification datasets that contain important local information and much noise. All experiments were run on a Linux server with one NVIDIA-Tesla V100 GPU with 32 GB of memory. More details are given in the supplemental document.

#### 4.1 Synthetic Task: XOR Problem

The XOR problem is a classification problem in artificial neural network research which cannot be solved by a single perceptron [50, 51]. We created more challenging XOR tasks with increasing sequence lengths. For each length N, a sequence consists of N pairs of numbers, where the first number, called value, is randomly chosen from the interval [0,1), and the second number is used as a marker. Most markers are 0 except two 1's at randomly selected positions. Let  $X_1$  and  $X_2$  denote the two values at the 1-marked positions. A sequence belongs to Class 0 if the values

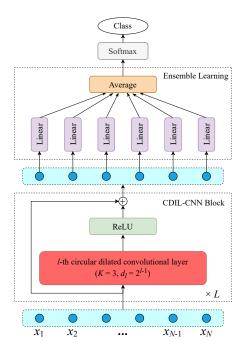


Figure 3: Our neural network architecture for sequence classification. The model comprises L blocks of CDIL-CNN and an average ensemble learning. Each block outputs the same size as the input sequence. After the convolutions and the linear transformation, each position gives its prediction logits, and the average ensemble learning aggregates the logits.

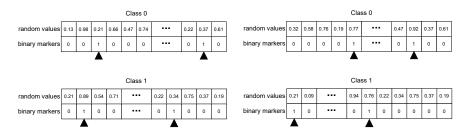


Figure 4: Examples of the XOR problem.

belong to the same half interval, i.e.,  $(X_1 < 0.5 \text{ and } X_2 < 0.5)$  or  $(X_1 \ge 0.5 \text{ and } X_2 \ge 0.5)$ . Otherwise, the sequence is labeled as Class 1. Figure 4 shows four examples of the XOR problem. We have used  $N=2^n$ , where  $n=4,\ldots,11$ . A larger N corresponds to a more challenging task. For each N, training, validation, and testing sets respectively have 10000 labeled sequences.

We have compared our model with several popular approaches: Transformer [7], Linformer [21], Performer [23], LSTM [10], GRU [11], TCN [9]. We have also included

- (Deformable): deformable convolutional networks that learn the adaptive receptive field using additional offsets [52],
- (CNN): conventional convolutional neural networks with dilation size 1,

All convolutional networks use the n-1 layers and 32 channels for a fair comparison.

The results are shown in Figure 5. Our model performs accurately for all sequence lengths, where CDIL-CNN achieves less than 1% error rate even when  $N=2^{11}$ . Transformer and its variants, RNNs, and Deformable achieve comparable error rates for short sequences. However, they turn inaccurate ( $\sim 50\%$  accuracy) when the sequences become longer than N=128. TCN and CNN perform even worse, where they respectively have 50% and 20% errors when N=32. The results indicate that CDIL-CNN is more scalable than the other compared methods.

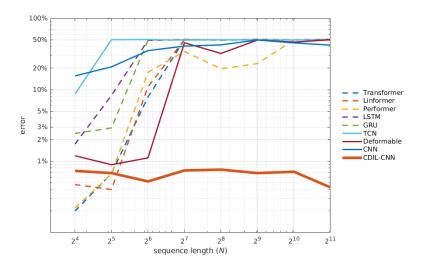


Figure 5: Error rate for the XOR problem with increasing sequence lengths.

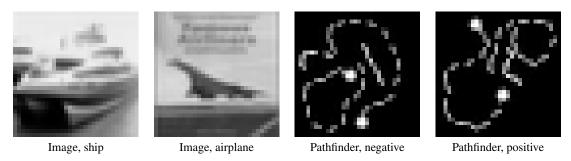


Figure 6: Examples of Image (left two) and Pathfinder (right two)

#### 4.2 Long Range Arena Benchmark

Long Range Arena is a public benchmark suite for evaluating model quality in long-context scenarios [53]. The suite consists of different data types, such as images and texts. Many Transformers have been evaluated on the suite [25, 53, 54, 55]. We compared our CDIL-CNN with other models on the following datasets:

- Image. This is a 10-class image classification task. The images come from the gray-scale version of CIFAR-10 [56], where pixel intensities (0-255) are treated as categorical values. Two example images and their labels are shown in Figure 6. Every image is flattened to a sequence of length N=1024. The task requires the model to learn the 2D spatial relations while using the 1D sequences.
- Pathfinder. This is a synthetic image task motivated by cognitive psychology [57, 58]. The task requires the model to make a binary decision whether two highlighted points are connected by a dashed path. Two example images and their labels are shown in Figure 6. Similar to the Image task, every pathfinder image is flattened to a sequence of length N=1024 with an alphabet size of 256.
- Text. This is a binary sentiment classification task of predicting whether an IMDb movie review is positive or negative [59]. The task considers the character-level sequences which generate longer inputs and make the task more challenging. We use a fixed length N=4000 for every sequence, which is truncated or padded when necessary.
- Retrieval. This is a character-level task with the ACL Anthology Network dataset [60]. The task requires the model to process a pair of documents and determine whether they have a common citation. Like the Text task, every document is truncated or padded to the sequence length of 4000, making the total length N=8000 for the pair.

For a fair comparison, we followed the same data preprocessing and training/validation/testing splitting in [53]. We quoted the results of Transformer and its variants from the literature and ran RNNs and CNNs for completeness. We

Table 1: Classification accuracy (%) of different models on LRA tasks. N is the sequence length. The dash means the result is absent in the reference paper. Means and standard deviations are computed by 5 runs.  $\bullet$  denotes significant difference, and  $\circ$  denotes insignificance.

Model	Image	Pathfinder	Text	Retrieval
	N=1024	N=1024	N = 4000	N = 8000
Transformer [53]	42.44	71.40	64.27	57.46
Transformer [55]	38.20	74.16	65.02	79.35
Transformer [54]	-	-	65.35	82.30
Local Attention [53]	41.46	66.63	52.98	53.39
Sparse Transformer [53]	44.24	71.71	63.58	59.59
Longformer [53]	42.22	69.71	62.85	56.89
Linformer [53]	38.56	76.34	53.94	52.27
Linformer [55]	37.84	67.60	55.91	79.37
Linformer [54]	-	-	56.12	79.37
Reformer [53]	38.07	68.50	56.10	53.40
Reformer [55]	43.29	69.36	64.88	78.64
Reformer [54]	-	-	64.88	78.64
Sinkhorn Transformer [53]	41.23	67.45	61.20	53.83
Synthesizer [53]	41.61	69.45	61.68	54.67
BigBird [53]	40.83	74.87	64.02	59.29
Linear Transformer [53]	42.34	75.30	65.90	53.09
Performer [53]	42.77	77.05	65.40	53.82
Performer [55]	37.07	69.87	63.81	78.62
Performer [54]	-	-	65.21	81.70
Nyströmformer [55]	41.58	70.94	65.52	79.56
Nyströmformer [54]	-	-	65.75	81.29
RFA-Gaussian [25]	-	-	66.0	56.1
Transformer-LS [54]	-	-	68.40	81.95
LSTM	$32.99 \pm 5.46 \bullet$	$61.26 \pm 12.14 \bullet$	$85.80 \pm 0.31$ $\bullet$	$77.18 \pm 0.23 \bullet$
GRU	$44.40 \pm 1.12 \bullet$	$85.45 \pm 0.16 \bullet$	$86.70 \pm 0.21 \bullet$	$77.08 \pm 0.26 \bullet$
TCN	$38.62 \pm 0.41 \bullet$	$85.48 \pm 0.46 \bullet$	$60.54 \pm 0.44 \bullet$	$76.85 \pm 0.08 \bullet$
Deformable	$36.57 \pm 3.03 \bullet$	$56.14 \pm 0.48 \bullet$	$86.91 \pm 0.22 \bullet$	$83.69 \pm 0.97 \circ$
CNN	$35.85 \pm 0.62 \bullet$	$55.95 \pm 0.06 \bullet$	$87.29 \pm 0.13$ $\circ$	$83.33 \pm 1.59 \circ$
CDIL-CNN	$64.49 \pm 0.61$	$91.00 \pm 0.37$	$87.61 \pm 0.33$	$84.27 \pm 0.76$

used one layer with a hidden size of 128 for RNNs and 64 channels for CNNs. All experiments were run five times with different random seeds, where means and standard deviations are reported in Table 1. We have used paired t-test at the significance level of 0.05 to verify whether CDIL-CNN is significantly different from RNNs or other CNNs.

Our model achieves the best mean accuracies in all tasks and is significantly better than RNNs and other CNNs in 17 out of 20 comparisons. The significant wins over all other methods hold for the Image and Pathfinder tasks. Espeicially for the Image task, CDIL-CNN achieves substantially higher mean accuracies (20.25% better than the best transformer variant, 20.09% better than the best RNN, 25.87% better than other CNNs). Deformable and CNN get comparable accuracies with CDIL-CNN for Text and Retrieval, probably because the two tasks mainly rely on local patterns.

#### 4.3 Time Series

The UEA & UCR Repository<sup>3</sup> consists of various time series classification datasets [61]. Many time series classification problems can be solved by detecting local patterns [62, 63, 64]. These tasks require the model to pick out important

<sup>&</sup>lt;sup>3</sup>http://www.timeseriesclassification.com/

Table 2: Classification accuracy (%) of different models on time series datasets. N is the sequence length. A DCNN run cannot finish in two days for the MosquitoSound dataset. Means and standard deviations are computed by 5 runs. All observed differences are statistically significant according to paired t-test at the significance level (p-value) of 0.05.

Model	FruitFlies	RightWhaleCalls	MosquitoSound		
	N = 5000	N = 4000	N = 3750		
Transformer	$55.26 \pm 1.47$	$71.84 \pm 0.65$	$32.92 \pm 0.69$		
Linformer	$81.80 \pm 1.61$	$71.17 \pm 0.84$	$60.44 \pm 0.70$		
Performer	$86.57 \pm 0.98$	$73.57 \pm 0.44$	$68.34 \pm 0.88$		
LSTM	$56.61 \pm 2.50$	$61.39 \pm 6.61$	$32.40 \pm 1.10$		
GRU	$61.47 \pm 12.35$	$63.18 \pm 8.54$	$42.44 \pm 5.66$		
TCN	$91.65 \pm 0.74$	$86.92 \pm 0.38$	$85.99 \pm 0.28$		
Deformable	$92.68 \pm 1.70$	$82.70 \pm 1.24$	$88.92 \pm 0.43$		
DCNN	$86.15 \pm 4.27$	$69.98 \pm 1.58$	-		
CNN	$95.30 \pm 0.27$	$78.34 \pm 1.05$	$89.72 \pm 0.12$		
CDIL-CNN	$97.09 \pm 0.08$	<b>91.99</b> $\pm$ 0.16	$91.54 \pm 0.22$		

local information from long sequences which contain much noise. We compared our CDIL-CNN with other popular models on three audio datasets:

- FruitFlies. The dataset comes from the same optical sensor which recorded the change in amplitude of an infra-red light as it was occluded by the wings of fruit flies during flight. The dataset contains 17259 training and 17259 testing sequences of length N=5000. The task requires the model to classify a sequence as one of three species of the fruit fly.
- **RightWhaleCalls**. Right whale calls are difficult to hear due to some low-frequency anthropogenic sounds. Up-calls are the most commonly documented right whale vocalization. The task requires the model to decide whether a sequence contains a set of right whale up-calls or not. The training and testing sizes of this dataset are 10934 and 1962, respectively. All sequences have a fixed length N=4000.
- MosquitoSound. The dataset represents the wing beat of the flying mosquito. Both training and testing sets have 139883 instances with sequence length N=3750. The task requires the model to classify each sequence into one of six species.

We split every original training set into training (70%) and validation (30%) parts, and used the original testing set for testing.

We have compared our model with Transformer, its two popular variants, RNNs, and CNNs. We also included dynamic convolutional neural networks (DCNNs) [65, 66], because it combines CNN and dynamic time warping, a widely used component in many time series classifiers. We used 32 channels for every convolutional layer. The classification results are shown in Table 2.

Our model significantly wins all three tasks with mean accuracies of 97.09%, 91.99%, and 91.54%, respectively. Transformers and RNNs struggle in the time series classification tasks. We found that convolutional networks perform better, probably because local signals are more important in these tasks. However, other CNNs are still inferior to our model.

#### 4.4 Ablation Study

Compared with conventional CNN, the proposed CDIL-CNN has two major contributed components: dilated convolution and circular mixing (padding). In this section we performed an ablation study to verify that both components are conducive to accurate and robust classifications. For this goal, we include a middle method called DIL that contains only the dilated convolution component but zero-padding. We then compare DIL with conventional CNN and CDIL-CNN.

For comparison, we first designed a more challenging XOR problem, where  $N=2^{11}$  and the test data can have the same position distribution as the training/validation data (Similar Test) or a different distribution (Dissimilar Test). See Figure 7 for illustration. In training/validation datasets, the two marked values appear in the first half for Class 0 and in the second half for Class 1. The test data follows the same pattern in the Similar Test, while the halves flip in the

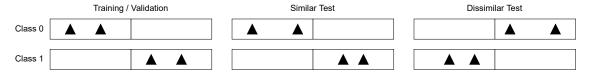


Figure 7: Position distribution of Similar Test and Dissimilar Test. Similar Test has the same position distribution as the training and validation datasets. Dissimilar Test flips the position distribution.

Table 3: Classification accuracy (%) of Similar Test and Dissimilar Test. Means and standard deviations are computed by 5 runs.

Model	Similar Test	Dissimilar Test
CNN	$50.79 \pm 0.57$	$50.46 \pm 0.54$
DIL	$99.99 \pm 0.01$	$0.81 \pm 0.42$
CDIL-CNN	$99.18 \pm 0.22$	$98.91 \pm 0.37$

Dissimilar Test. The data shift brings an extra challenge, where a non-robust model can wrongly classify the sequences by the absolute positions of the markers instead of the required XOR pattern from marked values.

The results are shown in Table 3. The CNN predictions are as bad as random guessing on both test sets, probably because it cannot capture the long-range interaction between the marked positions. DIL, equipped with dilated convolution, clearly improves the performance in Similar Test. However, DIL performs poorly on Dissimilar Test, which indicates that DIL overfits to training data and does not classify sequences by the required XOR pattern. CDIL-CNN differs from DIL by using circular padding instead of zero-padding. This change doesn't affect prediction performance in Similar Test, while achieves nearly perfect predictions in Dissimilar Test. The winning of CDIL-CNN shows that both dilated convolution and circular padding are needed for robust classification.

We also created a noisy time series classification task using RightWhaleCalls, where the test data can have the same data shift as the training/validation data (Similar Test) or different shift (Dissimilar Test). We added the Gaussian noise of length 2000 at the end of every sequence in the training/validation set. The test set in Similar Test follows the same preprocessing, while in Dissimilar Test, the Gaussian noise part is inserted in front of each original test sequence. The mean and standard deviation of Gaussian noise equal those of the original sequence. Figure 8 shows examples of noisy RightWhaleCalls.

The results are reported in Table 4, which leads to similar conclusions in the XOR problem. CNN gives mediocre accuracies in both Similar Test and Dissimilar Test. Dilated convolution endows DIL better performance in Similar Test than CNN. However, zero-padding makes DIL sensitive to the data shift and degrades its performance close to random guessing in Dissimilar Test. Equipped with both dilated convolution and circular padding, CDIL-CNN can robustly and accurately classify (higher than 91% accuracy) the time series in both cases.

Next, we visualized an input sequence of the XOR problem and its output features of CNN, DIL, and CDIL-CNN in Figure 9. The visualization helps us understand the difference among the methods in terms of receptive field and boundary effect. In conventional CNN, the important information is present locally even in the last layer, which can lead to a wrong prediction if the two markers are distant. In contrast, the receptive field in DIL and CDIL-CNN is much wider because they use dilated convolution. It is known that zero-padding can cause boundary artifacts [46, 47]. As we can see, here DIL has such artifacts in the left-most part of its visualization. Consequently, DIL probably misses the left

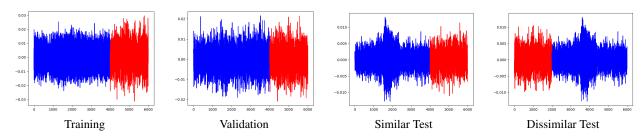


Figure 8: Examples of noisy RightWhaleCalls. Blue is the original sequence, and red is the additional noise.

Table 4: Classification accuracy (%) of noisy RightWhaleCalls. Means and standard deviations are computed by 5 runs.

Model	Similar Test	Dissimilar Test
CNN	$78.20 \pm 0.65$	$78.12 \pm 0.97$
DIL	$91.20 \pm 0.26$	$50.33 \pm 3.87$
CDIL-CNN	$91.33 \pm 0.33$	$91.39 \pm 0.37$

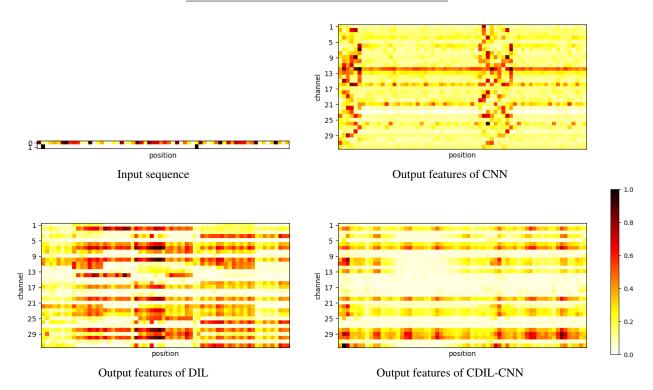


Figure 9: Normalized matrix plots of an example input sequence and its outputs by the last layer of CNN, DIL, and CDIL-CNN. Darker colors indicate larger values.

marked value and thus gives wrong classification. In comparison, CDIL-CNN with circular padding leads to more even output features across columns and does not suffer from boundary artifacts.

In summary, both dilated convolution and circular padding are useful for robust and accurate classification. With the two components, CDIL-CNN well mixes the signals from the input sequence to every output position. As a result, the subsequent averaging and linear classifier provide a good ensemble and do not lessen the contributions of the important information.

#### 5 Conclusions

We have proposed a novel convolutional model named Circular Dilated Convolutional Neural Network (CDIL-CNN) for sequence classification. Based on the characteristic of very long sequential data, we have used a design that consists of multiple symmetric and circular convolutions with exponential dilation sizes. Therefore, our model can remove boundary effect and enlarge the receptive fields quickly. In this way, every position of the last convolutional layer has an equal chance to receive all information of the whole input sequence. Finally, a simple average ensemble learning is applied to improve the accuracy. Experimental results show that our model has superior performance over all other models on various long sequential datasets.

In the future, we could add other popular modules to our model, such as absolute positional encoding [7], relative positional encoding [67], and conditional positional encoding [68], which could further improve the performance. We could also pre-train our model for few-shot or zero-shot learning, where only a few supervised labels are required in training.

#### 6 Acknowledgements

We acknowledge for using the IDUN computing cluster [69].

#### References

- [1] Aurangzeb Khan, Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology*, 1(1):4–20, 2010.
- [2] Rehan Akbani, Kadir C Akdemir, B Arman Aksoy, Monique Albert, Adrian Ally, Samirkumar B Amin, Harindra Arachchi, Arshi Arora, J Todd Auman, Brenda Ayala, et al. Genomic classification of cutaneous melanoma. *Cell*, 161(7):1681–1696, 2015.
- [3] Santiago Led, Jorge Fernández, and Luis Serrano. Design of a wearable device for ecg continuous monitoring using wireless technology. In *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 2, pages 3318–3321. IEEE, 2004.
- [4] Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5457–5466, 2018.
- [5] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.
- [6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [8] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv*:2009.06732, 2020.
- [9] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.
- [10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.
- [11] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.
- [12] Shraddha Singh, Saroj Kumar Pandey, Urja Pawar, and Rekh Ram Janghel. Classification of ecg arrhythmia using recurrent neural networks. *Procedia computer science*, 132:1290–1297, 2018.
- [13] Abdullah Aziz Sharfuddin, Md Nafis Tihami, and Md Saiful Islam. A deep recurrent neural network with bilstm model for sentiment classification. In 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pages 1–4. IEEE, 2018.
- [14] Jie Du, Chi-Man Vong, and CL Philip Chen. Novel efficient rnn and lstm-like architectures: Recurrent and gated broad learning systems and their applications for text classification. *IEEE transactions on cybernetics*, 51(3):1586–1597, 2020.
- [15] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.
- [16] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. arXiv preprint arXiv:1702.00887, 2017.
- [17] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.
- [18] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32:5243–5253, 2019.
- [19] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

- [20] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. In *NeurIPS*, 2020.
- [21] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv* preprint arXiv:2006.04768, 2020.
- [22] Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. In *International Conference on Machine Learning*, pages 10183–10192. PMLR, 2021.
- [23] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv* preprint arXiv:2009.14794, 2020.
- [24] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.
- [25] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. Random feature attention. *arXiv preprint arXiv:2103.02143*, 2021.
- [26] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. arXiv preprint arXiv:2001.04451, 2020.
- [27] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.
- [28] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR, 2020.
- [29] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
- [31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [33] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
- [34] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.
- [35] Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*, 2016.
- [36] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European Conference on Computer Vision*, pages 47–54. Springer, 2016.
- [37] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.
- [38] Brais Martinez, Pingchuan Ma, Stavros Petridis, and Maja Pantic. Lipreading using temporal convolutional networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6319–6323. IEEE, 2020.
- [39] Pingchuan Ma, Yujiang Wang, Jie Shen, Stavros Petridis, and Maja Pantic. Lip-reading with densely connected temporal convolutional networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2857–2866, 2021.
- [40] Jining Yan, Lin Mu, Lizhe Wang, Rajiv Ranjan, and Albert Y Zomaya. Temporal convolutional networks for the advance prediction of enso. *Scientific reports*, 10(1):1–15, 2020.

- [41] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [42] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.
- [43] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv* preprint arXiv:1706.05587, 2017.
- [44] Liang-Chieh Chen, Maxwell D Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jonathon Shlens. Searching for efficient multi-scale architectures for dense image prediction. *arXiv* preprint arXiv:1809.04184, 2018.
- [45] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In 2018 IEEE winter conference on applications of computer vision (WACV), pages 1451–1460. IEEE, 2018.
- [46] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14274–14285, 2020.
- [47] Bilal Alsallakh, Narine Kokhlikyan, Vivek Miglani, Jun Yuan, and Orion Reblitz-Richardson. Mind the pad–cnns can develop blind spots. *arXiv preprint arXiv:2010.02178*, 2020.
- [48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [49] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.
- [50] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.
- [51] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart and James L. Mcclelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.
- [52] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.
- [53] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.
- [54] Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. Long-short transformer: Efficient transformers for language and vision. *arXiv preprint arXiv:2107.02192*, 2021.
- [55] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. *arXiv preprint arXiv:2102.03902*, 2021.
- [56] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
- [57] Roos Houtkamp and Pieter R Roelfsema. Parallel and serial grouping of image elements in visual perception. *Journal of Experimental Psychology: Human Perception and Performance*, 36(6):1443, 2010.
- [58] Drew Linsley, Junkyung Kim, Vijay Veerabadran, Charlie Windolf, and Thomas Serre. Learning long-range spatial dependencies with horizontal gated recurrent units. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 152–164, 2018.
- [59] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.
- [60] Dragomir R Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. The acl anthology network corpus. Language Resources and Evaluation, 47(4):919–944, 2013.
- [61] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

- [62] Pierre Geurts. Pattern extraction for time series classification. In *European conference on principles of data mining and knowledge discovery*, pages 115–127. Springer, 2001.
- [63] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956, 2009.
- [64] Brian Kenji Iwana and Seiichi Uchida. Time series classification using local distance-based features in multi-modal fusion networks. *Pattern Recognition*, 97:107024, 2020.
- [65] Krisztian Buza. Time series classification and its applications. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, WIMS '18, New York, NY, USA, 2018. Association for Computing Machinery.
- [66] Krisztian Buza and Margit Antal. Convolutional neural networks with dynamic convolution for time series classification. In Krystian Wojtkiewicz, Jan Treur, Elias Pimenidis, and Marcin Maleszka, editors, *Advances in Computational Collective Intelligence*, pages 304–312, Cham, 2021. Springer International Publishing.
- [67] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv* preprint arXiv:1803.02155, 2018.
- [68] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.
- [69] Magnus Själander, Magnus Jahre, Gunnar Tufte, and Nico Reissmann. EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure, 2019.
- [70] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

## Supplemental Document. Classification of Long Sequential Data using Circular Dilated Convolutional Neural Networks

#### 1 XOR Problem

In this group of experiments, we used the categorical cross-entropy loss function and the Adam optimizer [70] with the learning rate of 0.001. We trained every model for 100 epochs using the batch size of 40. For RNNs, namely LSTM and GRU, we used 1 layer with a hidden size of 128. For Transformer, Linformer, and Performer, we used 32 dimensions, 4 layers, and 4 heads. In CNNs, we used the kernel size of 3 and 32 channels for every convolutional layer. We adopted the varying depth of CNNs so that the last position of TCN and each position of CDIL-CNN can cover the whole sequence, i.e., the depth L=n-1 for the sequence length  $N=2^n$ . Other convolutional networks used the same layers for a fair comparison. Table 1 gives the model sizes.

Table 1: The number of parameters ( $\approx K$ ) for every model on the XOR Problem. N is the sequence length. CNNs include TCN, CNN, DIL, and CDIL-CNN.

Model	$N = 2^4$	$N = 2^{5}$	$N = 2^{6}$	$N = 2^{7}$	$N = 2^{8}$	$N = 2^{9}$	$N = 2^{10}$	$N = 2^{11}$
Transformer	26.02	26.02	26.02	26.02	26.02	26.02	26.02	26.02
Linformer	45.47	47.52	51.62	59.81	76.19	108.96	174.50	305.57
Performer	101.28	101.28	101.28	101.28	101.28	101.28	101.28	101.28
LSTM	67.84	67.84	67.84	67.84	67.84	67.84	67.84	67.84
GRU	50.95	50.95	50.95	50.95	50.95	50.95	50.95	50.95
CNNs	6.69	9.83	12.96	16.10	19.23	22.37	25.51	28.64
Deformable	8.30	12.25	15.39	18.53	21.66	24.80	27.93	31.07

### 2 Long Range Arena

For this group of experiments, we quoted Transformers' results from reference papers [25, 53, 54, 55] and ran LSTM, GRU, TCN, CNN, Deformable, and CDIL-CNN for comparison. During training, we used the categorical cross-entropy loss function and the Adam optimizer with the learning rate of 0.001. For LSTM and GRU, we used 1 layer with a hidden size of 128. We used the kernel size of 3 and 64 channels for every convolutional layer. The depth was decided by the sequence length. All tasks had a vocabulary size of 256 and an embedding dimension of 64. Every model was trained for 100 epochs. More details of RNNs and CNNs are given in Table 2.

Table 2: Hyperparameters details of RNNs and CNNs for every LRA task.  $N, C, B, L, P_M$  refer to the sequence length, classes, batch size, CNN depth, and parameter size ( $\approx K$ ) of the model M, respectively. CNNs include TCN, CNN, and CDIL-CNN.

Task	N	C	В	$P_{LSTM}$	$P_{GRU}$	L	$P_{CNNs}$	$P_{Deformable}$
Image	1024	10	32	117.00	92.17	9	128.78	133.61
Pathfinder	1024	2	256	115.97	91.14	9	128.26	133.09
Text	4000	2	32	115.97	91.14	11	153.09	157.92
Retrieval	8000	2	256	116.74	91.91	11	153.47	158.30

#### 3 Time Series

In this group of experiments, we used the categorical cross-entropy loss function and the Adam optimizer with the learning rate of 0.001. We trained every model for 100 epochs using the batch size of 64. For LSTM and GRU, we used 1 layer with a hidden size of 128. For Transformer, Linformer, and Performer, we used 32 dimensions, 4 layers, and 4 heads. For CNNs, we used kernel size of 3 and 32 channels for every convolutional layer. The depth was decided by the sequence length. More details are given in Table 3.

Table 3: Hyperparameters details for every time series task. L and  $P_M$  refer to the CNN depth, and parameter size  $(\approx K)$  of the model M, respectively. CNNs include TCN, CNN, and CDIL-CNN.

Task	$P_{Trans.}$	$P_{Lin}$ .	$P_{Per.}$	$P_{LSTM}$	$P_{GRU}$	L	$P_{CNNs}$	$P_{DCNN}$	$P_{Deformable}$
FruitFlies	26.02	683.43	101.28	67.46	50.69	12	34.82	34.60	37.25
RightWhaleCalls	25.99	555.39	101.25	67.33	50.56	11	31.65	31.43	34.08
MosquitoSound	26.12	523.53	101.38	67.85	51.08	11	31.78	-	34.21