Neurocomputing 518 (2023) 50–59

Contents lists available at ScienceDirect

Neurocomputing

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

Classification of long sequential data using circular dilated convolutional neural networks

Lei Cheng [a], Ruslan Khalitov [a], Tong Yu [a], Jing Zhang [b], Zhirong Yang [a,*]

[a] Norwegian University of Science and Technology, Norway
[b] Beijing Jiaotong University, China

ARTICLE INFO

Article history:
Received 21 December 2021
Revised 6 September 2022
Accepted 24 October 2022
Available online 31 October 2022
Communicated by Zidong Wang

Keywords:
Classification
Sequential data
Convolutional neural networks

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).

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, 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 posi-

* Corresponding author.
E-mail address: zhirong.yang@ntnu.no (Z. Yang).

https://doi.org/10.1016/j.neucom.2022.10.054
0925-2312/

tion can receive both earlier and later information from previous layers in a circular manner. Unlike traditional 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 $x_t \in \mathbb{R}^D (1 \leqslant t \leqslant N)$ 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 rearranging 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 [29–32], while one-dimensional CNNs are commonly used for sequential data [33–35]. 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].

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 [41–45]. 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 full receptive fields 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 $(1 \leqslant t \leqslant N)$ 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[1] and $w^{(l)}$ are the convolution coefficients of the l-th layer. See Fig. 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

[1] We used $K = 3$ in all our experiments.

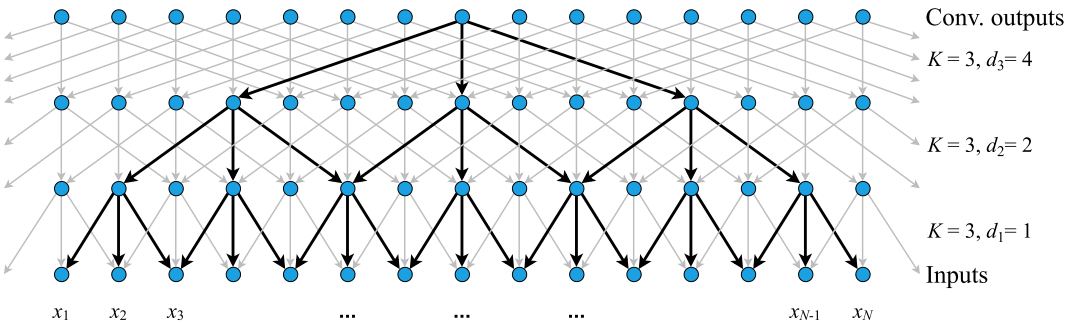

Fig. 1. Illustration of symmetric dilated convolutions. Blue nodes represent the sequence elements. Each layer keeps the same length as the input sequence. Symmetric convolutions of kernel size 3 are used in all layers. Dilation sizes are increased exponentially.

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 convolutions are shown in Fig. 2. The convolutional output b_t becomes

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

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 Fig. 3.

4. Experiments

We have compared our model with many popular models: Transformer [7], Linformer [21], Performer [23], LSTM [10], GRU [11], TCN [9]. We have also included deformable convolutional networks (Deformable) that learn the adaptive receptive field using additional offsets [50] and convolutional neural networks (CNN) with dilation 1. We used stride 1 in CNN and left out the pooling layers to ablate the effect of dilations in CDIL-CNN. In the supplemental document (Section 4), we also compared CDIL-CNN with ResNet18, a popular convolutional architecture with striding and pooling.

The compared models are tested 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 con-

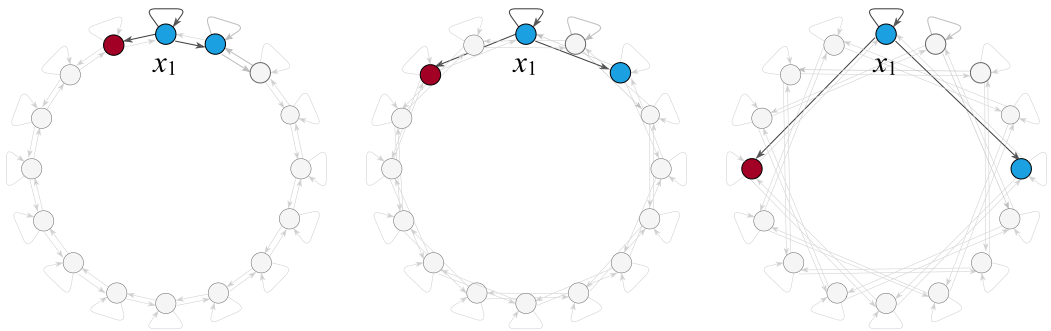

Fig. 2. Illustrations of circular mixing. (a), (b), and (c) are the first, second, and third convolutional layer, respectively. Red nodes represent convoluted positions from the other end.

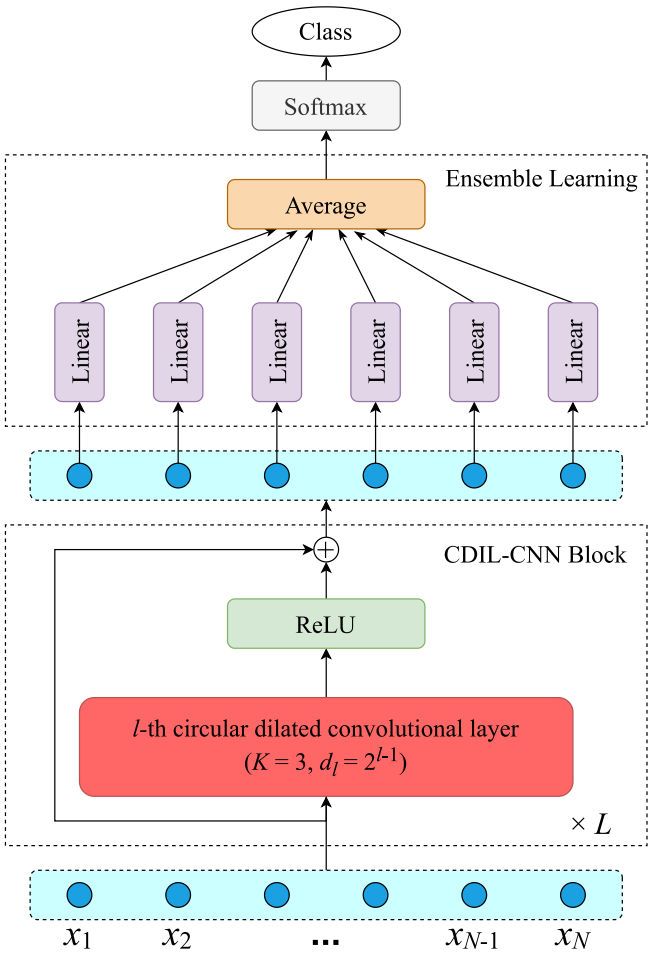

Fig. 3. Our neural network architecture for sequence classification. The model comprises L blocks of CDIL-CNN and an average ensemble learning. Each block outputs the same length as the input sequence. After the convolutions and the linear transformation, each position gives its prediction logits, and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

tain important local information and much noise. We used PyTorch[2] to implement our method. 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 classical classification problem in artificial neural network research which cannot be solved by a single perceptron [51,52]. 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 belong to the same half interval, i.e., ($X_1 < 0.5$ and $X_2 < 0.5$) or ($X_1 \geqslant 0.5$ and $X_2 \geqslant 0.5$). Otherwise, the sequence is labeled as Class 1. Fig. 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.

[2] https://pytorch.org/

We have compared our model with several popular approaches (including RNNs, Transformers, and CNNs). All convolutional networks use the $n - 1$ layers and 32 channels for a fair comparison.

The results are shown in Fig. 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.

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 [25,53–55]. 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 Fig. 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 Fig. 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 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. Especially 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 compa-

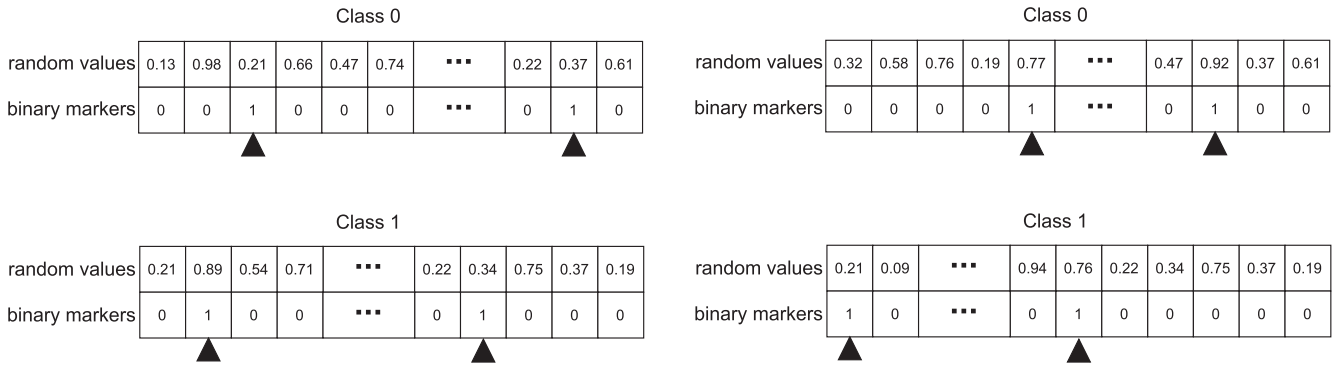

Fig. 4. Examples of the XOR problem.

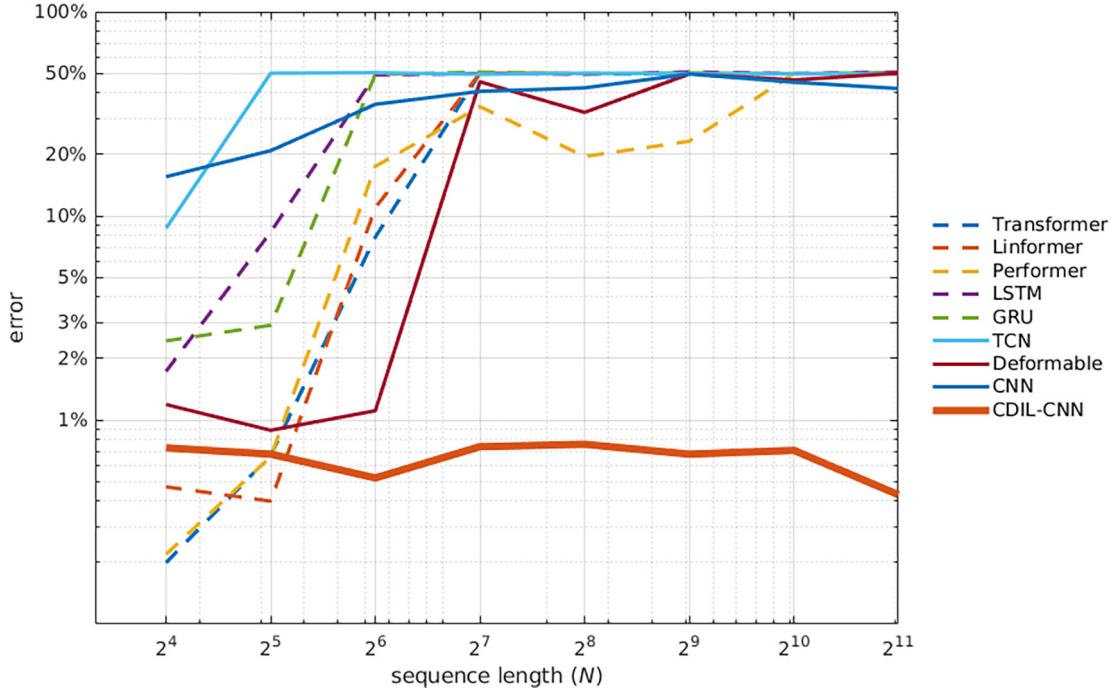

Fig. 5. Error rate for the XOR problem with increasing sequence lengths.

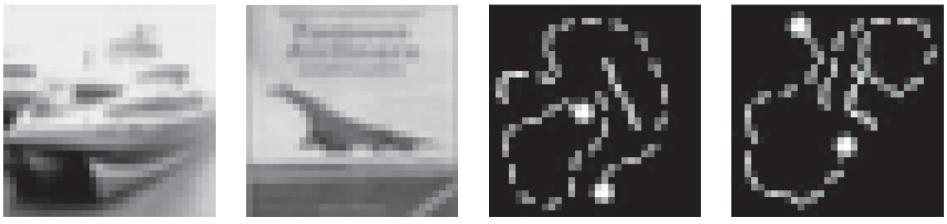

Fig. 6. Examples of Image (left two) and Pathfinder (right two).

rable 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[3] consists of various time series classification datasets [61]. Many time series classification problems can be solved by detecting local patterns [62–64]. These tasks require the model to pick out important 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

[3] http://www.timeseriesclassification.com/

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 across 5 runs. • denotes significant difference, and ○ denotes insignificance.

Model	Image N=1024	Pathfinder N=1024	Text $N = 4000$	Retrieval $N = 8000$
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 ± 5.46•	61.26 ± 12.14•	85.80 ± 0.31•	77.18 ± 0.23•
GRU	44.40 ± 1.12•	85.45 ± 0.16•	86.70 ± 0.21•	77.08 ± 0.26•
TCN	38.62 ± 0.41•	85.48 ± 0.46•	60.54 ± 0.44•	76.85 ± 0.08•
Deformable	36.57 ± 3.03•	56.14 ± 0.48•	86.91 ± 0.22•	83.69 ± 0.97○
CNN	35.85 ± 0.62•	55.95 ± 0.06•	87.29 ± 0.13○	83.33 ± 1.59○
CDIL-CNN	**64.49** ± 0.61	**91.00** ± 0.37	**87.61** ± 0.33	**84.27** ± 0.76

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 Transfomer, 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: dilated convolutions and circular mixing

Compared with conventional CNN, the proposed CDIL-CNN has two major contributed components: dilated convolutions 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.

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 across 5 runs. All observed differences are statistically significant according to paired t-test at the significance level (p-value) of 0.05.

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

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 Fig. 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 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 can-

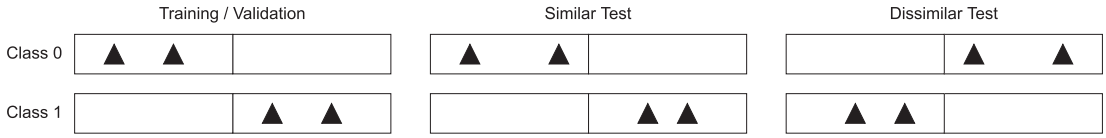

Fig. 7. Position distribution of Similar Test and Dissimilar Test. Similar Test has the same position distribution as the training and validation datasets. Dissimilar Test flips the position distribution.

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

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

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

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

not capture the long-range interaction between the marked positions. DIL, equipped with dilated convolutions, 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 convolutions 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. Fig. 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 convolutions endow 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 convolutions 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 Fig. 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 convolutions. 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 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 convolutions 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.

4.5. Ablation Study: receptive fields and ensemble learning

Here we perform extra analysis on the relation between two contributed components: 1) dilated convolutions that lead to full receptive fields and 2) ensemble learning that aggregates logits from all positions. We chose the Image task to verify their contributions to the performance improvement.

First, we study the effect of different ensemble sizes. We picked M positions, where $M = 1, 21, 41, \ldots, 1021, 1024$ and averaged their outputs to make the classification decision. The mean accuracy for each M over five runs is reported in Fig. 10. We can see that a larger ensemble size usually leads to a better accuracy.

Second, we find that both full receptive fields and ensemble learning are needed for better performance. We recapitulate the comparison among CNN, TCN, and CDIL-CNN. Similar to CDIL-CNN, the CNN method also averages the outputs from all positions, but it has only local receptive fields because it uses dilation size 1. TCN applies dilated convolutions to achieve full receptive fields,

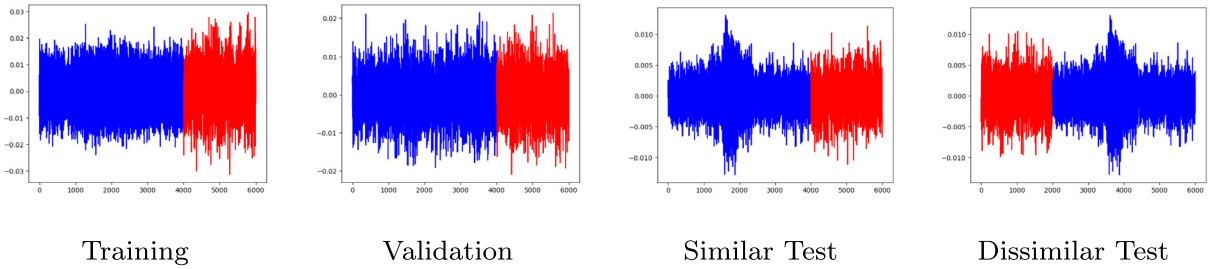

Fig. 8. Examples of noisy RightWhaleCalls. Blue is the original sequence, and red is the additional noise.

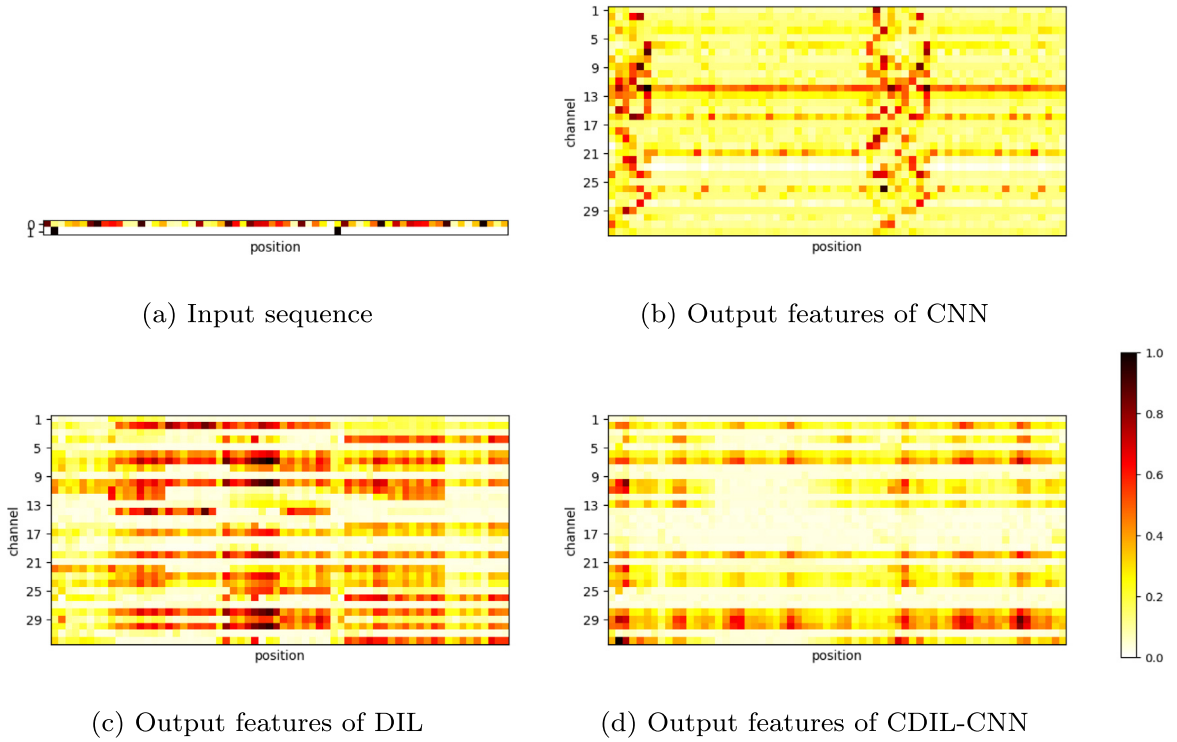

Fig. 9. Normalized matrix plots of an example input sequence and its outputs by the last layer of CNN, DIL, and CDIL-CNN. Darker colors indicate larger values.

while the classification result comes only from the last position without ensemble learning. From Table 5, we can see that using only full receptive fields or ensemble learning leads to mediocre accuracies. In contrast, CDIL-CNN, equipped with both components, has a substantially better accuracy.

Finally, we compare the computational cost of CDIL-CNN with previous convolutional networks CNN and TCN in the inference stage. All models have the same size (128.78 K parameters) for a fair comparison. We recorded their inference time for the test dataset in terms of FLOPs and seconds. For space cost, we measured their GPU memory consumption in MB. The results are shown in Table 6. We can see that CDIL-CNN has the same FLOPs as CNN and is just slightly slower than CNN due to the circular padding. TCN has much higher FLOPs and requires more computational time, probably because PyTorch does not provide causal padding. Consequently, TCN has to use longer padding and cut off the redundant right side, which is more expensive. For space cost, all compared models used the same amount of GPU memory because they used the same kernel size, the number of layers and convolution channels. In conclusion, using full receptive fields and ensemble learning does not cause much extra computational cost compared to conventional convolution networks.

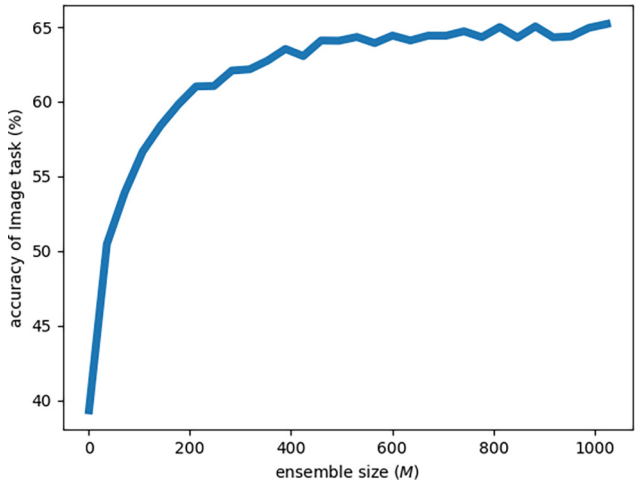

Fig. 10. Accuracy for the Image task with increasing ensemble size.

Table 5
Ablation study on full receptive fields and ensemble learning for the Image task.

Model	CNN	TCN	CDIL-CNN
Full receptive fields	×	✓	✓
Ensemble learning	✓	×	✓
Accuracy	35.85%	38.62%	64.49%

Table 6
Computational cost of three convolutional neural networks.

Model	CNN	TCN	CDIL-CNN
FLOPs (M)	113.247	503.219	113.247
Time (s)	2.49	3.61	2.80
Memory (MB)	0.496	0.496	0.496

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 apply our model to other types of long sequences, for example genome data that are known to have many long-range interactions among the sequence elements. Our model could also be used as the backbone for large-scale self-supervised pretraining (e.g., infer the masked elements from the non-masked) and then applied to few-shot or zero-shot learning, where only a few supervised labels are required in training.

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.

Acknowledgements

This work was supported by The Research Council of Norway, Grant No. 287284, and the China Scholarship Council. We acknowledge for using the IDUN computing cluster [67].

References

[1] A. Khan, B. Baharudin, L.H. Lee, K. Khan, A review of machine learning algorithms for text-documents classification, Journal of advances in information technology 1 (1) (2010) 4–20.
[2] R. Akbani, K.C. Akdemir, B.A. Aksoy, M. Albert, A. Ally, S.B. Amin, H. Arachchi, A. Arora, J.T. Auman, B. Ayala, et al., Genomic classification of cutaneous melanoma, Cell 161 (7) (2015) 1681–1696.
[3] S. Led, J. Fernández, L. Serrano, Design of a wearable device for ecg continuous monitoring using wireless technology, The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 2, IEEE, 2004, pp. 3318–3321.
[4] S. Li, W. Li, C. Cook, C. Zhu, Y. Gao, Independently recurrent neural network (indrnn): Building a longer and deeper rnn, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 5457–5466.
[5] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5 (2) (1994) 157–166.
[6] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.
[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.
[8] Y. Tay, M. Dehghani, D. Bahri, D. Metzler, Efficient transformers: A survey, arXiv preprint arXiv:2009.06732.
[9] S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv preprint arXiv:1803.01271.
[10] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.
[11] K. Cho, B. Van Merriënboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, arXiv preprint arXiv:1409.1259.
[12] S. Singh, S.K. Pandey, U. Pawar, R.R. Janghel, Classification of ecg arrhythmia using recurrent neural networks, Procedia computer science 132 (2018) 1290–1297.
[13] A.A. Sharfuddin, M.N. Tihami, M.S. Islam, A deep recurrent neural network with bilstm model for sentiment classification, in: in: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), IEEE, 2018, pp. 1–4.
[14] J. Du, C.-M. Vong, C.P. 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) (2020) 1586–1597.
[15] V. Mnih, N. Heess, A. Graves, et al., Recurrent models of visual attention, in: Advances in neural information processing systems, 2014, pp. 2204–2212.
[16] Y. Kim, C. Denton, L. Hoang, A.M. Rush, Structured attention networks, arXiv preprint arXiv:1702.00887.
[17] R. Child, S. Gray, A. Radford, I. Sutskever, Generating long sequences with sparse transformers, arXiv preprint arXiv:1904.10509.
[18] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, X. Yan, Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting, Advances in Neural Information Processing Systems 32 (2019) 5243–5253.
[19] I. Beltagy, M.E. Peters, A. Cohan, Longformer: The long-document transformer, arXiv preprint arXiv:2004.05150.
[20] M. Zaheer, G. Guruganesh, K.A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, et al., Big bird: Transformers for longer sequences., in: NeurIPS, 2020.
[21] S. Wang, B.Z. Li, M. Khabsa, H. Fang, H. Ma, Linformer: Self-attention with linear complexity, arXiv preprint arXiv:2006.04768.
[22] Y. Tay, D. Bahri, D. Metzler, D.-C. Juan, Z. Zhao, C. Zheng, Synthesizer: Rethinking self-attention for transformer models, in: International Conference on Machine Learning, PMLR, 2021, pp. 10183–10192.
[23] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, et al., Rethinking attention with performers, arXiv preprint arXiv:2009.14794.
[24] A. Katharopoulos, A. Vyas, N. Pappas, F. Fleuret, Transformers are rnns: Fast autoregressive transformers with linear attention, in: International Conference on Machine Learning, PMLR, 2020, pp. 5156–5165.
[25] H. Peng, N. Pappas, D. Yogatama, R. Schwartz, N.A. Smith, L. Kong, Random feature attention, arXiv preprint arXiv:2103.02143.
[26] N. Kitaev, Ł. Kaiser, A. Levskaya, Reformer: The efficient transformer, arXiv preprint arXiv:2001.04451.
[27] A. Roy, M. Saffar, A. Vaswani, D. Grangier, Efficient content-based sparse attention with routing transformers, Transactions of the Association for, Computational Linguistics 9 (2021) 53–68.
[28] Y. Tay, D. Bahri, L. Yang, D. Metzler, D.-C. Juan, Sparse sinkhorn attention, in: International Conference on Machine Learning, PMLR, 2020, pp. 9438–9447.
[29] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.
[30] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems 25 (2012) 1097–1105.
[31] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.
[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
[33] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, arXiv preprint arXiv:1609.03499.
[34] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. v. d. Oord, A. Graves, K. Kavukcuoglu, Neural machine translation in linear time, arXiv preprint arXiv:1610.10099.
[35] J. Gehring, M. Auli, D. Grangier, Y.N. Dauphin, A convolutional encoder model for neural machine translation, arXiv preprint arXiv:1611.02344.
[36] C. Lea, R. Vidal, A. Reiter, G.D. Hager, Temporal convolutional networks: A unified approach to action segmentation, in: European Conference on Computer Vision, Springer, 2016, pp. 47–54.
[37] C. Lea, M.D. Flynn, R. Vidal, A. Reiter, G.D. Hager, Temporal convolutional networks for action segmentation and detection, in: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 156–165.
[38] B. Martinez, P. Ma, S. Petridis, M. Pantic, Lipreading using temporal convolutional networks, in: ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020, pp. 6319–6323.
[39] P. Ma, Y. Wang, J. Shen, S. Petridis, M. Pantic, Lip-reading with densely connected temporal convolutional networks, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 2857–2866.
[40] J. Yan, L. Mu, L. Wang, R. Ranjan, A.Y. Zomaya, Temporal convolutional networks for the advance prediction of enso, Scientific reports 10 (1) (2020) 1–15.
[41] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122.
[42] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.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) (2017) 834–848.
[43] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam, Rethinking atrous convolution for semantic image segmentation, arXiv preprint arXiv:1706.05587.
[44] L.-C. Chen, M.D. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, J. Shlens, Searching for efficient multi-scale architectures for dense image prediction, arXiv preprint arXiv:1809.04184.
[45] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, G. Cottrell, Understanding convolution for semantic segmentation, in: 2018 IEEE winter conference on applications of computer vision (WACV), IEEE, 2018, pp. 1451–1460.
[46] O.S. Kayhan, J.C. v. 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, 2020, pp. 14274–14285.
[47] B. Alsallakh, N. Kokhlikyan, V. Miglani, J. Yuan, O. Reblitz-Richardson, Mind the pad–cnns can develop blind spots, arXiv preprint arXiv:2010.02178.
[48] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[49] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European conference on computer vision, Springer, 2016, pp. 630–645.
[50] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, Y. Wei, Deformable convolutional networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 764–773.

[51] M. Minsky, S. Papert, Perceptrons: An Introduction to Computational Geometry, MIT Press, Cambridge, MA, USA, 1969.
[52] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. Mcclelland (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations, MIT Press, Cambridge, MA, 1986, pp. 318–362.
[53] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, D. Metzler, Long range arena: A benchmark for efficient transformers, arXiv preprint arXiv:2011.04006.
[54] C. Zhu, W. Ping, C. Xiao, M. Shoeybi, T. Goldstein, A. Anandkumar, B. Catanzaro, Long-short transformer: Efficient transformers for language and vision, arXiv preprint arXiv:2107.02192.
[55] Y. Xiong, Z. Zeng, R. Chakraborty, M. Tan, G. Fung, Y. Li, V. Singh, Nyströmformer: A nyström-based algorithm for approximating self-attention, arXiv preprint arXiv:2102.03902.
[56] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images.
[57] R. Houtkamp, P.R. Roelfsema, Parallel and serial grouping of image elements in visual perception, Journal of Experimental Psychology: Human Perception and Performance 36 (6) (2010) 1443.
[58] D. Linsley, J. Kim, V. Veerabadran, C. Windolf, T. Serre, Learning long-range spatial dependencies with horizontal gated recurrent units, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018, pp. 152–164.
[59] A. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies, 2011, pp. 142–150.
[60] D.R. Radev, P. Muthukrishnan, V. Qazvinian, A. Abu-Jbara, The acl anthology network corpus, Language Resources and Evaluation 47 (4) (2013) 919–944.
[61] H.A. Dau, A. Bagnall, K. Kamgar, C.-C.M. Yeh, Y. Zhu, S. Gharghabi, C.A. Ratanamahatana, E. Keogh, The ucr time series archive, IEEE/CAA Journal of Automatica Sinica 6 (6) (2019) 1293–1305.
[62] P. Geurts, Pattern extraction for time series classification, in: European conference on principles of data mining and knowledge discovery, Springer, 2001, pp. 115–127.
[63] L. Ye, E. 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, 2009, pp. 947–956.
[64] B.K. Iwana, S. Uchida, Time series classification using local distance-based features in multi-modal fusion networks, Pattern Recognition 97 (2020) .
[65] K. Buza, Time series classification and its applications, in: Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, WIMS '18, Association for Computing Machinery, New York, NY, USA, 2018. doi:10.1145/3227609.3227690. URL:https://doi.org/10.1145/3227609.3227690.
[66] K. Buza, M. Antal, Convolutional neural networks with dynamic convolution for time series classification, in: K. Wojtkiewicz, J. Treur, E. Pimenidis, M. Maleszka (Eds.), Advances in Computational Collective Intelligence, Springer International Publishing, Cham, 2021, pp. 304–312.
[67] M. Själander, M. Jahre, G. Tufte, N. Reissmann, EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure (2019). arXiv:1912.05848.

Lei Cheng received her Bachelor and Master degrees from Beijing Jiaotong University in 2016 and 2019, respectively. She is working toward the Ph.D. degree at Norwegian University of Science and Technology (NTNU). Her research interests include neural modeling for long sequential data and representation learning.

Ruslan Khalitov received his Bachelor and Master degrees from Higher School of Economics in 2016 and 2018, respectively. Currently he is a Ph.D. candidate at Norwegian University of Science and Technology (NTNU). He is interested in deep learning and graph theory.

Tong Yu received his Bachelor degree from Xidian University in 2016, and Master degree from Xi'an Jiaotong University in 2019, respectively. He is doing a Ph.D. at Norwegian University of Science and Technology (NTNU). His research focuses on neural modelling for long sequences and representation learning.

Jing Zhang received his Bachelor degree in computer science and technology from Beijing Jiaotong University in 2017. He is currently pursuing the Ph.D. degree at the same school. His research interests include machine learning, deep learning and zero-shot learning.

Zhirong Yang received his Bachelor and Master degrees from Sun Yat-Sen University, China in 1999 and 2002, and his Doctoral degree from Helsinki University of Technology, Finland in 2008, respectively. Presently he is a professor with the Norwegian Open AI Lab and Department of Computer Science at Norwegian University of Science and Technology (NTNU). He leads a research group with a focus on neural modeling for very long sequential data and learning representations for structured data.