MTSCCleav: a Multivariate Time Series Classification (MTSC)-based method for predicting human Dicer cleavage sites

First Author[1,2*], Second Author[2,3†] and Third Author[1,2†]

[1*]Department, Organization, Street, City, 100190, State, Country.
[2]Department, Organization, Street, City, 10587, State, Country.
[3]Department, Organization, Street, City, 610101, State, Country.

*Corresponding author(s). E-mail(s): iauthor@gmail.com;
Contributing authors: iiauthor@gmail.com; iiiauthor@gmail.com;
†These authors contributed equally to this work.

Abstract

Background: MicroRNAs (miRNAs) are small non-coding RNAs (ncRNAs) that regulate gene expression at the post-transcriptional level, thereby playing essential roles in diverse biological processes. The biogenesis of miRNAs requires Dicer, which is an enzyme, to cleave at specific sites on the precursor miRNAs (pre-miRNAs). These sites are called cleavage sites. Several machine learning approaches have been proposed to predict whether an input sequence contains a cleavage site. Existing studies have several limitations. They rely heavily on complex feature engineering or opaque deep neural networks. It results in a lack of generalizability and a long running time. Additionally, the probabilities of the base pairs in the predicted secondary structure have been ignored in the classification. There is a need for an alternative modeling paradigm that is accurate, fast, and simple.

Results: We proposed a novel approach to reframe the task as a multivariate time series classification problem. For this purpose, we proposed various encoding schemes to convert the RNA sequence and the secondary structure information into time series. Hence, the data can be represented in the form of a multivariate time series. We leveraged the probabilities of the base pairs in the classification, which have been long ignored in the literature. Computational experiments demonstrate that our proposed scheme can achieve better or comparable results using a simpler, more intuitive model and less computational time. Through perturbation experiments, we found that regions close to the center of pre-miRNAs are essential for predicting human dicer cleavage sites. Notably, the encoding of

RNA data into time series enables us to utilize any well-established tools from the time series community.
Conclusion: Our proposed scheme enables us to approach this problem in a novel way. By transforming the RNA sequence and its secondary structure information into a time series and using simple, state-of-the-art time series classifiers, we obtained comparable or even superior performance in a simpler and faster way. We introduced the use of probabilities of base pairs in the classification. The analysis of the importance of the subsequences suggests that the regions close to the center of the pre-miRNA are essential for this problem. Code is available at: https://github.com/cyuab/time-series-classification-cleavage.

Keywords: miRNA, Dicer Cleavage Site, Genomic signal processing (GSP), (Multivariate) time Series Classification (MTSC, TSC)

1 Background

One of the most important theories in molecular biology is the central dogma. It depicts the flow of genetic information [1, 2]. Proteins are the functional units. And the information stored in DNA is used to create them. This process is known as gene expression, which involves two key steps: transcription and translation. Genes (segments of DNA encoding proteins) in DNA are used as templates for messenger RNAs (mRNAs) synthesis. This synthesis process is called transcription. An mRNA acts as a set of instructions to assemble a chain of amino acids, which form a linear polypeptide [1]. This construction process is called translation. This chain is not yet functional. To become biologically active, this chain is folded into a specific three-dimensional structure, a proper configuration that enables it to perform its desired functions. This process is called protein folding. And this folded polypeptide is called a functional protein, or simply a protein.

This entire process closely mirrors how a computer program runs on a machine, where DNA serves as the source code, genes are the code segments, mRNA acts as the assembly code, and proteins are the executables. The source code does not function by itself. First, it is translated into assembly code (a lower-level, less human-readable form) and then into an executable file that can actually perform the intended tasks [3].

These mRNAs are called "coding RNAs" because they code for proteins. There are other genes in which the final product is the RNA molecule itself. They are called non-coding RNAs (ncRNAs). They function by themselves, such as by regulating gene expression. Two types of small ncRNAs are particularly important. They are microRNAs (miRNAs) and small interfering RNAs (siRNAs). Their discovery was recognized with the 2006 Nobel Prize in Physiology or Medicine[2], awarded for work completed only eight years prior [1]. They play crucial roles in regulating gene expression.

[1] To be accurate, the mRNA of protein-encoding genes undergoes translation to produce a protein, while many genes produce non-coding RNAs (ncRNA) that function without being translated into proteins
[2] The Nobel Prize in Physiology or Medicine 2006 - NobelPrize.org: https://www.nobelprize.org/prizes/medicine/2006/summary/ (Accessed on: 2025-06-13).

In this study, we focus on miRNAs. They are small RNAs with a length of about 22 nt. They regulate gene expression post-transcriptionally [4]. A miRNA can regulate the expression of several proteins. Hence, understanding the biogenesis of miRNAs is of great value. It involves the processing of primary miRNAs (pri-miRNAs). RNAs are 3D molecules, but it is hard to measure the 3D structure (tertiary structure). We can understand their behavior by analyzing their 1D sequence. It is easily obtained through sequencing. A predicted secondary structure of a pri-miRNA's sequence is shown in Figure 1.

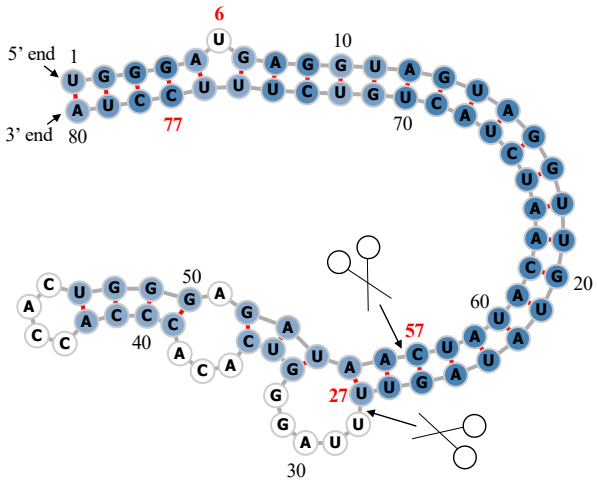


Fig. 1: Predicted secondary structure of the pri-miRNA "hsa-let-7a-1"[3]. We denote the sequence as S. Experimental evidence suggests that the two deviated mature miRNAs are $UGA\cdots GUU$ and $CUA\cdots UUC$. They are $S[6:27]$ and $S[57:77]$ (Both ends are inclusive.). Since $S[6:27]$ ($S[57:77]$) is near the 5' (3') end, we call it "5p (3p) mature miRNA". The starting and ending indices of these two subsequences are indicated in **bold**. It suggests that the two cleavage sites are the two bonds immediately after the 27^{th} nucleotide and before the 57^{th} nucleotide. The two scissors indicate the two cleavage sites. The color intensity of the nodes reflects their base pair probability in this predicted secondary structure configuration. The deeper the color, the higher the probability. The unpaired nodes are uncolored. The raw figure is generated by RNAfold web server[4].

Recall that a pri-miRNA contains a hairpin loop, also called a stem loop. A microprocessor complex comprising Drosa and DCGR8 cleaves the pri-miRNA to form a precursor miRNA (pre-miRNA) inside the nucleus. The stem-loop is still preserved, but the two arms become shorter. After that, the pri-miRNA is transported by Exportin 5 from the nucleus to the cytoplasm (i.e., outside the nucleus). It is further

[3]Its miRBase entry: https://mirbase.org/hairpin/MI0000060. (Accessed on: 2025-06-12).
[4]RNAfold web server: http://rna.tbi.univie.ac.at/cgi-bin/RNAWebSuite/RNAfold.cgi. (Accessed on: 2025-06-12). The figure is viewed in "forna". This view option can be chosen on the website.

cleaved by an enzyme called Dicer [5]. The Dicer cleaves the stem-loop from the two arms at the two cleavage sites, shown as the two scissors in figure 1. We refer to the bond between two nucleotides along the strand that is cleaved by Dicer as the Dicer cleavage site. The stem-loop is removed. It results in a short double-stranded miRNA molecule. Furthermore, these molecules may be subjected to additional trimming. Some nucleotides are removed from the two ends.

One strand of the resulting molecule is loaded into an RNA-induced silencing complex (RISC). This loaded strand guides the RISC to the target mRNA to silence it, and it results in gene silencing. This loaded strand is called the guide strand or mature miRNA. The other stand, which is called the passenger strand, is usually degraded. Note that both miRNA single strands, resulting from the unwinding of the double-stranded miRNA molecule, can become the guide strand. For example, "hsa-let-7a-1" has two guide strands or mature miRNA products.

Dicer plays an important role in the biogenesis of miRNAs. Hence, accurate cleavage of pre-miRNAs by Dicer is crucial for gene silencing. It is reasonable to argue that the structure of the pre-miRNAs informs Dicer about the cleavage process. A recent study shows that a particular secondary RNA structure, namely 22-bulge, enhances the accuracy of miRNA biogenesis experimentally [6].

It would be of great benefit to understand how dicer selects cleavage sites from the neighborhood information near the cleavage sites. Studies [7–9] revealed that the secondary structures of the sequence are essential for cleavage site determination. Hence, to predict or classify whether a subsequence, extracted from pri-miRNAs, contains a cleavage site, we need to make use of both the sequence and secondary structure information.

1.1 Related work

PHDcleav employed support vector machines (SVM), leveraging sequence and structure-based features [10] for the prediction. LBSizeCleav improved upon it by considering the loop and bulge lengths [11]. [12] proposed an ensemble learning approach, using a gradient boosting machine for better accuracy. [13] developed a deep learning model, namely DiCleave. This model used an autoencoder to learn the secondary structure embeddings of pre-miRNAs from all the species (not only human) presented in the miRBase database and leveraged this information in the prediction of human dicer cleavage sites. These methods begin with curated pre-miRNA sequences from the miRBase database. Their secondary structures were predicted. Patterns were extracted from the sequence, along with the secondary structure. These patterns are called cleavage patterns. They create the positive patterns by setting the cleavage sites at the middle of the patterns. One exception is the follow-up work of [13], which creates the cleavage pattern by allowing cleavage sites to appear at any position within the pattern, instead of the middle [14]. Then, these works employ different encoding schemes to represent these patterns and utilize various machine learning models for classification.

These models suffer several limitations. They rely heavily on complicated feature engineering and opaque deep learning models [13, 14]. It results in a lack of generalizability and a long running time. There is a need to design a simpler model so that

it can be easily extended to other prediction tasks on RNA data. One way to analyze sequence data is to transform it into time series data, where the entries are continuous. In response to this, we proposed a multivariate time series classification-based method. We introduced encoding methods to convert RNA data to time series. In addition, we leveraged the base pair probabilities in the predicted secondary structure in the transformation. To the best of our knowledge, this information has not yet been leveraged in the transformation. For the resulting time series data, we employ state-of-the-art convolution-based classifiers, whose superiority has been experimentally tested on data from diverse domains [15]. Additionally, we performed perturbation-based experiments on the classification of the resulting time series to investigate the importance of subsequences to the prediction. In summary, our contributions are shown as follows.

1. To the best of our knowledge, we are the first to frame the prediction of the cleavage sites as a multivariate time series classification problem.
2. We proposed utilizing the base-pair probabilities in the predicted secondary structure for the prediction. To our surprise, this information has been ignored in the existing works.
3. We conducted perturbation-based experiments. It shows that regions close to the cleavage sites are important for this problem. It agrees with the existing study [12].

2 Methods

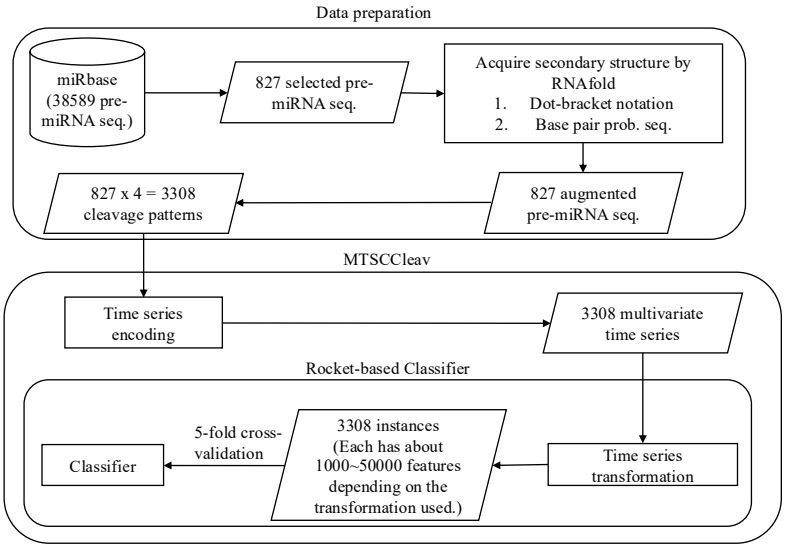


Fig. 2: The overall pipeline of this study. Symbol notations: Cylinder - Dataset, Rectangle - Process, Parallelogram - Input / Output, Rounded Rectangle - Component.

The overall pipeline of this study, including the design of MTSCCleav, is summarized in Figure 2. In this session, we first discuss data preparation. We then briefly review the concepts of time series and propose time series encoding methods. After that, time series classifiers are discussed. Finally, we discuss the evaluation metrics.

2.1 Data preparation

We used miRBase database [16][5]. The database comprises miRNA data from various organisms, including humans, mice, and C. elegans [17]. Each data entry refers to a miRNA sequence, along with other properties such as name, accession (the identifier used in miRBase), organism (its origin), and information on its derivative miRNA products. We are interested in pri-miRNA in humans. The derivative miRNA products are the mature miRNAs. The database also annotates the location of the mature miRNA within the original pri-miRNA and indicates whether its existence has experimental evidence.

Accession	Name	Organism	Sequence	Mature miRNA 1	Mature miRNA 2
MI0000001	cel-let-7	Caenorhabditis elegans	UACAC...UUCGA	cel-let-7-5p 17:38 experimental	cel-let-7-3p 60:81 experimental
MI0000060	hsa-let-7a-1	Homo sapiens	UGGGA...UCCUA	hsa-let-7a-5p 6:27 experimental	hsa-let-7a-3p 57:77 experimental
MI0000114	hsa-mir-107	Homo sapiens	CUCUC...ACAGA	hsa-miR-107 50:72 experimental	NA
MI0000238	hsa-mir-196a-1	Homo sapiens	GUGAA...UUCAC	hsa-miR-196a-5p 7:28 experimental	hsa-miR-196a-1-3p 45:65 not experimental

Table 1: Selected representative records from miRBase. For the last two columns, the first line shows the name of this mature miRNA product, the second line shows its location in the original sequence ($x : y$ denote that this product is located from the x position to the y position. Both ends are inclusive), and the third line indicates whether its existence has experimental evidence. The selected one is in **bold**.

The database contains 38589 miRNA records. Table 1 shows its four representative records. We use it to elucidate our selection criteria. The records are rows in the table. We selected the records from humans (Homo sapiens). It resulted in 1917 records. To identify the actual locations of the two cleavage sites in the pri-miRNA sequence supported by experimental evidence, we selected records that have two mature miRNAs resulting from cleavage at the 5p-arm and the 3p-arm, both of which have experimental support. According to the above selection criteria, only "MI0000060" would be selected in the table. After the selection process, we selected 827 experimental validated pre-miRNA sequences, each with its two mature miRNA products. This formed our dataset.

[5]The website is www.mirbase.org, and the newest version of the database is Release 22.1 (Accessed on 2025-06-22).

Sequence	Secondary Structure (In Dot-bracket notation)
1 UGGGA**UGAGGUAGUAGGUUGUAUAGUU** 27 28 UUAGGGUCACACCCACCACUGGGAGAU 54 55 AA**CUAUACAAUCUACUGUCUUUC**CUA 80	1 (((((**.((((((((((((((((((((((((** 27 28 UUAGGGUCACACCCACCACUGGGAGAU 54 55 **))))))))))))))))))))))))**)))) 80
Base-pair probabilities sequence (the first 10 bases)	
1 (0.549, 0.946, 0.987, 0.987, 0.904) 5 6 (**0.000**, 0.841, 0.974, 0.981, 0.890) 10	

Table 2: The whole sequence of "hsa-let-7a-1" with the locations of its two mature miRNAs and its predicted secondary structure by RNAfold. We have numbered each line with the starting and ending positions. The corresponding positions of the two mature miRNAs and the probability of the unpaired "U" are **bolded**.

The entire sequence and the necessary information for our downstream analysis of "hsa-let-7a-1" are listed in Table 2.

2.1.1 Argument the dataset with Secondary Structure information

We aim to utilize domain knowledge about pre-miRNA sequences to enhance the accuracy of our classifier. We leverage the secondary structure of these sequences to achieve this. Recall that a specific three-dimensional (3D) structure is required for DNA, RNA, and protein to perform functions [18]. However, finding these 3D structures using experimental methods such as X-ray crystallography or nuclear magnetic resonance (NMR) is costly and time-consuming. Hence, prediction methods for such 3D structures are necessary and helpful for downstream analysis. However, predicting such 3D structures is challenging. One of the reasons is that there are some "non-conventional" base-pair interactions that allow an RNA structure to fold into a 3D structure. The local structure of the 3D structures, the secondary structures, only focus on the conventional base-pair interactions [2]. It facilitates the prediction of secondary structure more easily and effectively than predicting the 3D structure. We employed RNAfold from the ViennaRNA Package[6]. RNAfold returns the secondary structure in the dot-bracket notation and a matrix of base-pair probabilities.

Definition 1 (Dot-bracket notation) Dot-bracket notation is a way of representing the secondary structure of the given string s. One of the following symbols is assigned to each base in s.

- Open parentheses "(" indicates that the base is paired with a complementary base further along in s.
- Close parentheses ")" indicates that the base is paired with a complementary base earlier in s.
- Dot "." indicates that the base is unpaired.

[6]The latest stable release is Version 2.7.0, accessed on 2025-06-22) to predict the secondary structure for a given pri-miRNA [19].

Equipped with the matrix, we can construct the base-pair probability sequence of the original sequence. An illustrative example is shown in Table 2.

2.1.2 Extract cleavage patterns

The locations of the two mature miRNAs on the main sequence indicate the probable locations of the two cleavage sites. The 5p cleavage site must be beyond and near the ending location of the 5p mature miRNA. For example, the ending position of the 5p mature miRNA for "hsa-let-7a-1" is 27. So, the 5p cleavage site would be one of the bonds beyond the 27^{th} nucleotide. We deemed the immediate bond next to the 5p mature miRNA's ending position the 5p cleavage site, with the knowledge that the actual cleavage site may not be this immediate bond but rather the nearby bonds after it. The same applies to the 3p cleavage site. It is at the immediate bond before the starting position of the 3p mature miRNA, which is 57.

We extracted a 14-string (i.e., string with length = 14) with the cleavage site located at the center. The first 7 nt (nucleotide) before the center are **bolded**. In our running example, it would be "**UAUAGUU**UUAGGU" for the 5p cleavage site and "**GAGAUAA**CUAUACA" for the 3p cleavage site. We call these 14-strings cleavage patterns as they contain the cleavage sites. We can also generate non-cleavage patterns by selecting a 14-string with the center 6 nt away from the corresponding cleavage sites towards the corresponding mature miRNA [11, 12]. It is based on the assumption that the dicer is less likely to cut the middle of the mature miRNA than the opposite side. So, in our example, the 5p non-cleavage pattern would be "**AGGUUGU**AUAGUUU". The center of the 3p non-cleavage pattern is the bond between 62^{nd} and 63^{rd} nucleuotides. The 3p non-cleavage pattern would be "**ACUAUAC**AAUCUAC".

In conclusion, for a given pri-miRNA, we can generate two cleavage patterns (positive samples) and two non-cleavage patterns (negative samples). For convenience, we also call these four patterns simply the "four strings" of a given pri-miRNA. The four strings of "hsa-let-7a-1" are listed in Table 3.

	5p-cleav	non-5p-cleav	3p-cleav	3p-non-cleav
Input strand	UAUAGUUUUAGGGU	AGGUUGUAUAGUUU	GAGAUAACUAUACA	ACUAUACAAUCUAC
Complementary strand	AUAUCAA_____UA	UCUAACAUAUCAA_	C_CUGUUGAUAUGU	UGAUAUGUUGGAUG

Table 3: The first row shows the four strings of "hsa-let-7a-1". Their complementary strands are shown in the second row.

We could construct the complementary strand of each of the strands in the "four strings" by finding the corresponding paired base for each of the bases in the input strand by considering the secondary structure information. We use '_' to denote the unpaired base in the complementary strand. For example, in Figure 1, the sub-string "UUAGG" in the 5p-cleavage pattern "UAUAGUUUUAGGGU" is unpaired while other bases do pair with some bases in the complementary strand, the resulting complementary strand is "AUAUCAA____UA". There is a loop/ budge there.

We call the four original input strands and the constructed four complementary strands together as the "eight strings" of the input pre-miRNA, also shown in Table 3.

2.2 Time Series Encoding

Definition 2 (Time Series and Subsequence) A time series $T = t_1, t_2, ..., t_n$ is a sequence of real-valued numbers with length $= n$. A short contiguous region of T is called a subsequence. A subsequence $T(i : j)$ of a time series T is a shorter time series that starts from position i and ends at position j.

2.2.1 Transform strings into time series

Strings and time series are temporal sequences. The primary difference between strings and time series lies in their behavioral attributes [20]. For strings, also known as words, a y-value is a symbol from a predefined set called the alphabet. Thus, we also refer to the symbols as letters. For example, the alphabet is $\{A, C, G, T\}$ in the DNA string, while $\{A, C, G, U\}$ in the RNA string. For time series, a y-value is a scalar number. The number can be an integer or a real number. Unlike real numbers, there is no ordering in the alphabet unless some external domain knowledge is introduced.

In the bioinformatics community, the study of applying signal processing techniques to genomic data, which includes DNA and RNA strings, is called "Genomic Signal Processing" (GSP) [21]. In the field of GSP, the time series representations of DNA strings are called DNA numeric representations (DNR). Many DNRs have been proposed in the field of GSP, with applications including identifying protein-coding regions in DNA sequences [22], biological sequence querying [23], and finding similarities between DNA sequences [24]. We noted that DNA strings and RNA strings are equivalent from a computational standpoint. Many transformation methods designed for DNA are applicable to RNA by simply substituting T for U. We present the nine encoding methods. The relationship among them is shown in Figure 3.

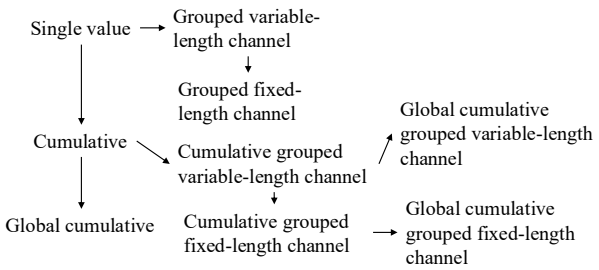


Fig. 3: Relationship of the proposed transformation methods.

One of the simple, if not the simplest, transformations is to map the letters in the alphabet into integer numbers without considering any domain knowledge about the nucleotides. This method or approach is called the "Single value mapping" [21, 25–28]. One single value is assigned to each of the letters. Domain knowledge can be

	Name	Numeric representation	Example for $s = G, A, G, A, U, A, A, C, U, A$
1	Single value mapping	for $i = 1$ to $\|s\|$: $t_i = \begin{cases} 2 & \text{if } s_i = A \\ 1 & \text{if } s_i = G \\ -1 & \text{if } s_i = C \\ -2 & \text{if } s_i = U \end{cases}$ return t	$t = 1, 2, 1, 2, -2, 2, 2, -1, -2, 2$
2	Grouped variable-length channel mapping	$j = 1, k = 1$ for $i = 1$ to $\|s\|$: $t^1_j = \begin{cases} 1, & j = j + 1 \quad \text{if } s_i = A \\ -1, & j = j + 1 \quad \text{if } s_i = U \end{cases}$ $t^2_k = \begin{cases} 1, & k = k + 1 \quad \text{if } s_i = G \\ -1, & k = k + 1 \quad \text{if } s_i = C \end{cases}$ return $t^1[1:j], t^2[1:k]$	$t^1 = 1, 1, -1, 1, 1, -1, 1$ $t^2 = 1, 1, -1$
3	Grouped fixed-length channel mapping	for $i = 1$ to $\|s\|$: $t^1_i = \begin{cases} 1, & j = j + 1 \quad \text{if } s_i = A \\ -1, & j = j + 1 \quad \text{if } s_i = U \\ 0 & \text{otherwise} \end{cases}$ $t^2_i = \begin{cases} 1, & k = k + 1 \quad \text{if } s_i = G \\ -1, & k = k + 1 \quad \text{if } s_i = C \\ 0 & \text{otherwise} \end{cases}$ return t^1, t^2	$t^1 = 0, 1, 0, 1, -1, 1, 1, 0, -1, 1$ $t^2 = 1, 0, 1, 0, 0, 0, 0, -1, 0, 0$
4	Cumulative mapping	$t_1 = 0$ for $i = 1$ to $\|s\|$: $t_{i+1} = \begin{cases} t_i + 2 & \text{if } s_i = A \\ t_i + 1 & \text{if } s_i = G \\ t_i - 1 & \text{if } s_i = C \\ t_i - 2 & \text{if } s_i = U \end{cases}$ return t	$t = 0, 1, 3, 4, 6, 4, 6, 8, 7, 5, 7$
5	Cumulative grouped variable-length channel mapping	$t^1_1 = 0, t^2_1 = 0$ $j = 1, k = 1$ for $i = 1$ to $\|s\|$: $t^1_{j+1} = \begin{cases} t^1_j + 1, & j = j + 1 \quad \text{if } s_i = A \\ t^1_j - 1, & j = j + 1 \quad \text{if } s_i = U \end{cases}$ $t^2_{k+1} = \begin{cases} t^2_k + 1, & k = k + 1 \quad \text{if } s_i = G \\ t^2_k - 1, & k = k + 1 \quad \text{if } s_i = C \end{cases}$ return $t^1[1:j], t^2[1:k]$	$t^1 = 0, 1, 2, 1, 2, 3, 2, 3$ $t^2 = 0, 1, 2, 1$
6	Cumulative grouped fixed-length channel mapping	$t^1_0 = 0, t^2_0 = 0$ for $i = 1$ to $\|s\|$: $t^1_{i+1} = \begin{cases} t^1_i + 1 & \text{if } s_i = A \\ t^1_i - 1 & \text{if } s_i = U \\ t^1_i & \text{otherwise} \end{cases}$ $t^2_{i+1} = \begin{cases} t^2_i + 1 & \text{if } s_i = G \\ t^2_i - 1 & \text{if } s_i = C \\ t^2_i & \text{otherwise} \end{cases}$ return t^1, t^2	$t^1 = 0, 0, 1, 1, 2, 1, 2, 3, 3, 2, 3$ $t^2 = 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1$

Table 4: Time series transformation for RNA string s. The ending index in indexing is exclusive. s is the first ten nucleotides of the non-cleavage pattern on the 3p-arm of "hsa-let-7a-1"

utilized during assignments. For example, [29] employs the atomic number of each nucleotide as the transformed values, where $\{G : 78, A : 70, C : 58, T : 66\}$. [30] uses "electron-ion" interaction potential representation (EIIP) as such values, where $\{G : 0806, A : 1260, C : 1340, T : 1335\}$. Our goal is to transform the input strand and its complementary strand into two time series, aiming to capture the information contained in these sequences and their secondary structure. So, we need to employ the complementarity property during the transformation [31]. Recall that in the base-pairing rules, "A" pairs with "U"[7] to form two hydrogen bonds while "G" pairs with "C" to form three hydrogen bonds. Hence, "A" ("C") can be regarded as the "inverse" of "U" ("G") [8]. We can preserve these base-pairing rules in the time series representation by assigning A (G) and U (C) opposite values. 'A' and 'G' have a two-ring structure. They are purines. 'U' and 'C' have a one-ring structure. They are pyrimidines. We put 'A' and 'G' ('U' and 'C') on the same side of the number line with zero in the middle.

Now, the only remaining question is which nucleotide on the same side we should assign a larger absolute value to. We adopted the "A = 2 and G = 1" assignment. The reasoning is as follows. The main goal of this manuscript is to find the two cleavage sites on the pre-miRNA. The cleavage sites are the bonds along the strands, the phosphodiester linkages. After the pre-miRNA is cleaved, the resulting double-stranded miRNA molecule is unwound to form the guide strand and the passenger strand. The stability of the double strand would affect the unwinding process. There are three hydrogen bonds in C-G pairs and two in A-U pairs. Hence, C-G pairs are more stable than A-U pairs. This means that in regions with more C-G pairs, the double strands hold more tightly, and the unwinding process is less likely to occur than in regions with more A-U pairs. So, we want to emphasize the existence of such less stable A-U pairs in the double strands in our time series representations. We assign A and U with larger absolute values than those of G and C. With this reasoning, we propose our first transformation method, as shown in row 1 of Table 4.

We can also transform each RNA sequence into a multivariate time series using grouped binary encoding, where nucleotides are grouped into (A, U) and (G, C). Each time series channel indicates the presence of a nucleotide from the respective group at each time step. The kind of methods releases our assumption that A (U) has a larger absolute value than G (C). Noted that each channel may differ in length depending on group-specific occurrences, we propose two variations. The first one allows the output to be variable-length sequences per channel. The second one always returns two resulting sequences of fixed length. These methods are rows 2 and 3 in Table 4.

The "Single value mapping" and its two variations allow us to focus on the absolute values of the alphabet and ignore what has happened in the past.

2.2.2 Cumulative mapping

The "cumulative version" allows us to focus on analyzing the "trend", such as increasing and decreasing, by accumulating what has happened in the past [32, 33]. We show the "Cumulative mapping" in row 4 in the Table 4. We can also use two channels

[7] In DNA, "A" pairs with "T"
[8] Recall that we call "-1' as the inverse of "1"' and vice versa under addition in Algebra.

to represent the time dynamics of a nucleotide group in "Cumulative mapping". For the output, we can allow variable-length channels or fixed-length channels. These two variations are illustrated in rows 5 and 6 of the table.

2.2.3 Incorporating base-pair probabilities

We can incorporate the base-pair probabilities in the encoding by multiplying the base-pair probability of the nucleotide with the assigned value of that nucleotide during encoding. They are treated as the weight or confidence of that numerical assignment of the nucleotide. Table 5 incorporates the base-pair probability time series into the transformation methods listed in Table 4.

2.2.4 Accumulating from the beginning of the pre-miRNA sequence

In cumulative encoding, it would return a different result if we choose different starts for the accumulation. We can start the accumulation from the beginning of the full sequence or just from the beginning of the selected local subsequence. The first one preserves the global context by recording the previous counts of the nucleotide up to the selected subsequence. It can be useful when early elements (those before the selected subsequence) influence later interpretation. In other words, the previous nucleotides may affect the chemical property of the selected subsequence. The second one only focuses on the local history by ignoring the global history. It is useful if the previous nucleotides do not affect the chemical property of the selected subsequence.

Considering the time series representation of the running example $s = GAGAUAACUA$ in "Cumulative mapping" in Table 4 $t = 0, 1, 3, 4, 6, 4, 6, 8, 7, 5, 7$, it accumulates from zero. According to Figure 1, $s = S[50 : 60]$ In other words, it begins at the 50^{th} position of the whole pre-miRNA sequence, as shown in Table 2. If we start the accumulation from the first entry $S[1]$ of the whole pre-miRNA sequence instead, it will yield a different result. Suppose the last entry of the resulting time series of S[1:49] with Cumulative mapping encoding is 2, the resulting time series of s would be $2, 3, 5, 6, 8, 6, 8, 10, 9, 7, 9]$ if we start the "Cumulative" at the beginning at S[1]. Note that these two time series have the same trend, but they start at different values. The first one starts at 2 while the latter one starts at 0. We name the first approach as "Global Cumulative" and the latter one as "Local Cumulative" or simply "Cumulative". We can apply the "Global" notion to every cumulative method in the Table 4

2.2.5 Transforming the secondary string into time series

We can apply a similar idea to transform the secondary string in the dot-bracket notation into a time series by "Single value mapping", where "(" maps to 1, "." maps to 0, and ")" maps to -1.

2.3 Time series classification

In univariate time series classification, an instance in the dataset consists of a time series $x = x_1, x_2, ..., x_m$ with m observations and a discrete class label y, which takes

	Name	Numeric representation	Example for $s = C, \_, C, U, G, U, U, G, A, U$ $s^P = 0.843, 0.000, 0.807, 0.807, 0.793, 0.914, 0.982, 1.000, 0.999, 0.999$
1	Single value mapping [21, 25–28]	for $i = 1$ to $\|s\|$: $t_i = \begin{cases} 2 \cdot s_i^P & \text{if } s_i = A \\ 1 \cdot s_i^P & \text{if } s_i = G \\ -1 \cdot s_i^P & \text{if } s_i = C \\ -2 \cdot s_i^P & \text{if } s_i = U \\ 0 & \text{if } s_i = \_ \end{cases}$ return t	Without base-pair probability: $t = -1, 0, -1, -2, 1, -2, -2, 1, 2, -2$ With base-pair probability: $t = -0.843, 0.000, -0.807, -1.614, 0.793, -1.829, -1.963, 1.000, 1.999, -1.998$
2	Grouped variable-length channel mapping	$j = 1, k = 1$ for $i = 1$ to $\|s\|$: $t_j^1 = \begin{cases} 1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = A \\ -1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = U \\ 0, & j = j + 1 & \text{if } s_i = \_ \end{cases}$ $t_k^2 = \begin{cases} 1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = G \\ -1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = C \end{cases}$ return $t^1[1:j], t^2[1:k]$	Without base-pair probability: $t^1 = 0, -1, -1, -1, 1, -1$ $t^2 = -1, -1, 1, 1$ With base-pair probability: $t^1 = 0.000, -0.807, -0.914, -0.982, 0.999, -0.999$ $t^2 = -0.843, -0.807, 0.793, 1.000$
3	Grouped fixed-length channel mapping	for $i = 1$ to $\|s\|$: $t_i^1 = \begin{cases} 1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = A \\ -1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = U \\ 0 & & \text{otherwise} \end{cases}$ $t_i^2 = \begin{cases} 1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = G \\ -1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = C \\ 0 & & \text{otherwise} \end{cases}$ return t^1, t^2	Without base-pair probability: $t^1 = 0, 0, 0, -1, 0, -1, -1, 0, 1, -1$ $t^2 = -1, 0, -1, 0, 1, 0, 0, 1, 0, 0$ With base-pair probability: $t^1 = 0.000, 0.000, 0.000, -0.807, 0.000, -0.914, -0.982, 0.000, 0.999, -0.9999$ $t^2 = -0.843, 0.000, -0.807, 0.000, 0.793, 0.000, 0.000, 1.000, 0.000, 0.000$
4	Cumulative mapping [32, 33]	$t_1 = 0$ for $i = 1$ to $\|s\|$: $t_{i+1} = \begin{cases} t_i + 2 \cdot s_i^P & \text{if } s_i = A \\ t_i + 1 \cdot s_i^P & \text{if } s_i = G \\ t_i - 1 \cdot s_i^P & \text{if } s_i = C \\ t_i - 2 \cdot s_i^P & \text{if } s_i = U \\ t_i & \text{if } s_i = \_ \end{cases}$ return t	Without base-pair probability: $t = 0, -1, -1, -2, -4, -3, -5, -7, -6, -4, -6$ With base-pair probability: $t = 0.000, -0.843, -0.843, -1.650, -3.265, -2.471, -4.300, -6.263, -5.264, -3.265, -5.263$
5	Cumulative grouped variable-length channel mapping	$t_1^1 = 0, t_1^2 = 0$ $j = 1, k = 1$ for $i = 1$ to $\|s\|$: $t_{j+1}^1 = \begin{cases} t_j^1 + 1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = A \\ t_j^1 - 1 \cdot s_i^P, & j = j + 1 & \text{if } s_i = U \\ t_j^1, & j = j + 1 & \text{if } s_i = \_ \end{cases}$ $t_{k+1}^2 = \begin{cases} t_k^2 + 1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = G \\ t_k^2 - 1 \cdot s_i^P, & k = k + 1 & \text{if } s_i = C \end{cases}$ return $t^1[1:j], t^2[1:k]$	Without base-pair probability: $t^1 = 0, -1, -2, -3, -2, -3$ $t^2 = 0, -1, -2, -1, 0$ With base-pair probability: $t^1 = 0.000, -0.807, -1.722, -2.703, -1.704, -2.703$ $t^2 = 0.000, -0.843, -1.650, -0.857, 0.143$
6	Cumulative grouped fixed-length channel mapping	$t_0^1 = 0, t_0^2 = 0$ for $i = 1$ to $\|s\|$: $t_{i+1}^1 = \begin{cases} t_i^1 + 1 \cdot s_i^P & \text{if } s_i = A \\ t_i^1 - 1 \cdot s_i^P & \text{if } s_i = U \\ t_i^1 & \text{otherwise} \end{cases}$ $t_{i+1}^2 = \begin{cases} t_i^2 + 1 \cdot s_i^P & \text{if } s_i = G \\ t_i^2 - 1 \cdot s_i^P & \text{if } s_i = C \\ t_i^2 & \text{otherwise} \end{cases}$ return t^1, t^2	Without base-pair probability: $t^1 = 0, 0, 0, 0, -1, -1, -2, -3, -3, -2, -3$ $t^2 = 0, -1, -1, -2, -2, -1, -1, -1, 0, 0, 0$ With base-pair probability: $t^1 = 0.000, 0.000, 0.000, 0.000, -0.807, -0.807, -1.722, -2.703, -2.703, -1.704, -2.703$ $t^2 = 0.000, -0.843, -0.843, -1.650, -1.650, -0.857, -0.857, -0.857, 0.143, 0.143, 0.143$

Table 5: Time series transformation for RNA complementary string s with its probability time series s^p. The ending index in indexing is exclusive.

c possible values. If $c = 2$, we refer to these problems as binary classification problems. If $c > 2$, we refer to these problems as multiclass classification problems. In multivariate time series classification, the time series is not a single sequence but a list of vectors over d dimensions and m observations. $x =< x_1, x_2, ..., x_d >$, where $x_k = x_{1,k}, x_{2,k}, ..., x_{m,k}$. We also denote the j^{th} of the i^{th} case of dimension k as a scalar number $x_{i,j,k}$.

There are many classifiers defined for time series data, including distance-based, feature-based, interval-based, shapelet-based, dictionary-based, convolution-based, and deep learning-based classifiers. Additionally, two or more of the above approaches can be combined, resulting in hybrid approaches [15]. For a complete review of time series classification, we would like to direct the readers to [15, 34, 35] for details. As our downstream classifier, we employ convolution-based classifiers due to their simplicity and accuracy.

2.3.1 Convolution-based classification

Convolution-based classifiers first use randomly parameterized kernels to perform convolutions on the original time series. Each convolution is performed by sliding a kernel across the original time series and computing a dot product. The output of this process is another time series, namely an activation map.

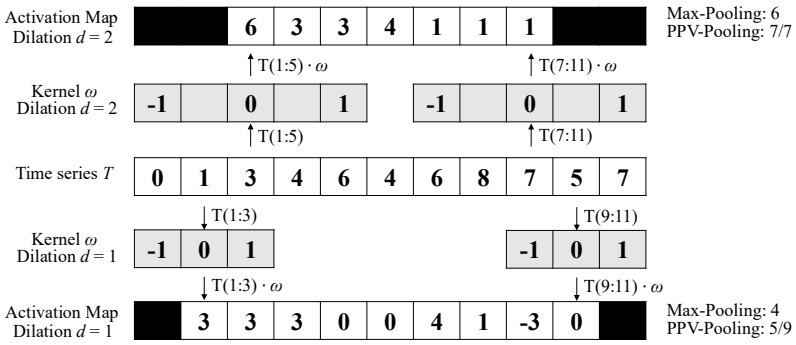


Fig. 4: Features generation in the transformation

Figure 4 shows two kernels ω_1 and ω_2 , each of which performs convolution with T. For example, ω_1 performs dot product with $T(1 : 3)$ and result 3 as the first entry of the activation map. By sliding the kernel one time stamp at a time, an activation map is produced for each kernel. Then, the summary features are derived from the activation map, such as the maximum and proportion of positive values. The most popular convolution-based approach is the Random Convolutional Kernel Transform (ROCKET) [36]. It generates a large number of randomly parameterized kernels, ranging from thousands to tens of thousands. The kernel's parameters include length (It defines the kernel's length), weights (the entries inside the kernel), bias (valued added to the result of the convolution operation). Besides, padding can be applied on the input series at the start and the end, such that the activation map has

the same length as the input. The summary statistics are obtained through two pooling operations: the max and the proportion of positive values (PPV). Hence, for k kernels, the transformed data has $2k$ features. The default value of k is 10000. There are two extensions of ROCKET. They are MiniRocket [37] and MultiRocket [38]. MiniRocket removes unnecessary operations and many of the random components of ROCKET. It speeds up Rocket by over an order of magnitude with no significant difference in accuracy, making the classifier almost deterministic. For example, the kernel length is fixed, and only two weight values are used. Only PPV is used for the summary statistics. MultiRocket is extended from MiniRocket. The main improvement of it is to extract features from first-order differences and add three new pooling operations. The three added operations are mean of positive values (MPV), mean of indices of positive values (MIPV) and longest stretch of positive values (LSPV).

The HYbrid Dictionary-ROCKET Architecture (Hydra) is a time series classification model that combines dictionary-based and convolution-based models [39]. Similar to ROCKET-family classifiers, it uses random 1D convolutional kernels to extract features from the input time series. But it groups the kernels into g groups of k kernels each. Each time series is passed through all the groups. For each group of kernels, we slide them across T and compute the dot product at each timestamp. Recall that the dot product of two input vectors has the maximum value when the two vectors align in the same direction and the minimum value when they are oriented in different directions. We record the kernel that best matches the subsequence of T at each timestamp. This results in a k-dimensional count vector for each of the g groups. It results in a total of $g \times k$ features, with default values of $g = 64$ and $k = 8$. In addition to recording the kernel with the maximum response, we can also record the kernel with the minimum response, with the knowledge that this kernel would be the best match with the "inverted" subsequence of T. Hydra is applied to both the original time series and its first-order differences. [39] concatenate features from Hydra with features from MultiRocket to obtain the best result. This classifier is called MultiROCKET-Hydra.

Typically, convolution-based approaches employ a simple design pattern, which involves the overproduction of features followed by a selection strategy. A large number of features are generated for each instance. The derived features from the transformation methods are then fed into a simple linear classifier to obtain the final classification result. Then, a simple classifier such as a linear ridge classifier determines which features are most useful. A ridge classifier is used in this study. It is a linear classifier that extends ridge regression to classification tasks by applying a threshold to the predicted values. It uses L2 regularization to prevent overfitting. The regularization strength is selected by internal cross-validation. A Ridge classifier is suggested for small datasets, as in our case, while a logistic regression classifier is suggested for large datasets [15].

2.4 Evaluation metrics

To evaluate the performance of our time series-based classification model, we adopted five standard classification metrics. They are Accuracy (Acc), Specificity (Sp),

Sensitivity (Sn), F1 score (F1), and Matthews Correlation Coefficient (MCC) [40].

$$
\begin{aligned}
Acc &= \frac{TP + TN}{TP + TN + FP + FN} \\
Sp &= \frac{TN}{TN + FP} \\
Sn &= \frac{TP}{TP + FN} \\
F1 &= \frac{2 \times TP}{2 \times TP + FP + FN} \\
MCC &= \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\end{aligned}
$$

Where TP, TN, FP, and FN are the number of true positives, true negatives, false positives, and false negatives, respectively.

To extend a binary metric to multiclass problems, we can treat the data as a collection of binary problems, one for each class. One class is treated as positive while the other classes are treated as negative. Then, the multiclass metrics can be obtained by averaging binary metric calculations across the set of classes. There are different ways to do the averaging. Here, we adopted a macro-averaging approach. It treats each class equally and calculates the mean of the binary metrics. To use MCC in the multiclass case, it can be defined in terms of a confusion matrix C for K classes, where $C_{i,j}$ is the number of observations that are actually in class i and predicted to be in class j [41].

$$
MCC_{multi} = \frac{c \times s - \sum_k^K p_k \times t_k}{\sqrt{(s^2 - \sum_k^K p_k^2) \times (s^2 - \sum_k^K t_k^2)}}
$$

Where $t_k = \sum_i^K C_{ik}$ (The number of times class k actually occurred), $p_k = \sum_i^K C_{ki}$ (The number of times class k was predicted), $c = \sum_k^K C_{kk}$ (The total number of samples correctly predicted) and $s = \sum_i^K \sum_j^K C_{ij}$ (The total number of samples).

3 Results

In all experiments, the models were trained and evaluated using 5-fold cross-validation. We retrieved 827 empirically validated sequences of pre-miRNAs. There are 5p-arm and 3p-arm in each sequence. For each arm, we define a cleavage pattern and a non-cleavage pattern. Three datasets, namely "5p-arm", "3p-arm", and "multi-class" were constructed by these patterns. The 5p-arm dataset comprises 827 positive instances and an equal number of negative instances. We refer to the cleavage patterns as positive instances and the non-cleavage patterns as negative instances. The 5p-arm and 3p-arm datasets are binary-class datasets. The multi-class dataset comprises all patterns from both the 5p-arm and the 3p-arm. There are 827 "5p" instances (cleavage patterns from the 5p-arm), 827 "3p" instances, and 1654 negative instances (non-cleavage patterns).

For every fold in 5-fold cross-validation, the dataset was divided into a training set and a test set with sizes of 80% and 20% of the whole dataset, respectively. We kept the class distribution approximately the same in each fold, as it is in the original dataset. In each fold derived from the 5p-arm and 3p-arm datasets, the training set has a size of 1323, and the test set has a size of 331. In each fold derived from the multi-class dataset, the training set has a size of 2262, and the test set has a size of 662. We reported the average of the five classification metrics.

The ROCKET-based classifiers require all channels in the multivariate time series to have equal length. We applied padding to shorter channels using the constant value 100, which does not appear in any resulting time series from the encodings. It ensures the padding does not introduce ambiguity or interfere with the semantic meaning of the encoded nucleotide signals.

3.1 Channel ablation study

We utilized three types of data as the input features for each instance. They are (1) the RNA sequence, which consists of the primary strand and its complementary strand, (2) the secondary structure information, and (3) the base-pair probability sequence. To input the data into our time series-based classifiers, we converted the data into multivariate time series. The RNA strand and its complementary strand are each encoded into one or two channels. For example, single value mapping encodes each of them in one channel, while grouped variable-length channel mapping employs two channels to encode each of them. The secondary structure information is converted into a multivariate time series. The base-pair probability sequence is already in numerical form and does not require further transformation. It can be used either as a standalone channel or incorporated into the encoding of the complementary strand, as described by the encoding methods in Table 5. We performed a channel ablation study to determine the most informative combination of the above channels.

We referred to the multivariate time series that consists of the channels from the RNA sequence only as the baseline setting. We added the other channels to this baseline. It leads to the following configurations (cfgs):

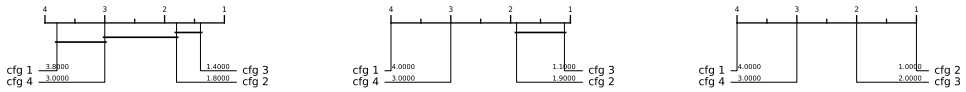
- (cfg 1) Baseline: Time series derived only from the RNA sequence.
- (cfg 2) Baseline + Secondary Structure (SS): Baseline plus the time series representation of the secondary structure.
- (cfg 3) Baseline + Base-pair probability (Standalone as a separated channel): Baseline plus the base-pair probability sequence as a separated channel.
- (cfg 4) Baseline + Base-Pair Probability (Incorporated): Baseline with the base-pair probability sequence incorporated into the encoding of the complementary strand.

We used the single value mapping as the encoding method in this experiment. Table 6 shows the result. From the table, we can see that the addition of secondary structure, base-pair probability as a standalone channel, and base-pair probability incorporated in the encoding can improve the performance. We plotted the critical difference (CD) diagram to visualize Table 6 to make the performances of different combinations more obvious. In CD diagrams, lower-ranked methods (toward the right)

	Classifier	5p arm					3p arm					Multi				
		Acc	Sp	Sn	F1	MCC	Acc	Sp	Sn	F1	MCC	Acc	Sp	Sn	F1	MCC
Baseline (cfg 1)	ROCKET	0.781	0.743	0.819	0.789	0.563	0.790	0.773	0.807	0.793	0.580	0.717	0.838	0.685	0.700	0.538
	MiniROCKET	0.755	0.728	0.782	0.762	0.512	0.788	0.781	0.794	0.789	0.576	0.685	0.823	0.653	0.662	0.486
	MultiROCKET	0.784	0.767	0.801	0.787	0.569	0.803	0.792	0.814	0.805	0.606	0.691	0.830	0.667	0.672	0.501
	Hydra	0.830	0.800	0.860	0.835	0.663	0.808	0.797	0.820	0.810	0.617	0.731	0.844	0.696	0.712	0.560
	MultiROCKET-Hydra	0.796	0.778	0.815	0.800	0.594	0.807	0.767	0.816	0.808	0.614	0.701	0.836	0.681	0.686	0.520
Baseline + Secondary Structre (cfg 2)	ROCKET	**0.847**	**0.832**	0.862	**0.849**	**0.695**	**0.855**	0.842	**0.868**	**0.857**	**0.711**	**0.836**	**0.907**	**0.828**	**0.833**	**0.736**
	MiniROCKET	0.825	0.807	0.843	0.827	0.652	0.822	0.802	0.843	0.826	0.646	0.823	0.900	0.812	0.818	0.715
	MultiROCKET	0.812	0.803	0.822	0.814	0.626	0.824	0.809	0.839	0.826	0.649	0.796	0.888	0.791	0.792	0.673
	Hydra	0.845	0.816	**0.873**	**0.849**	0.691	0.846	0.817	0.874	0.850	0.693	0.830	0.901	0.814	0.826	0.724
	MultiROCKET-Hydra	0.817	0.809	0.826	0.819	0.635	0.825	0.816	0.834	0.826	0.652	0.803	0.891	0.798	0.800	0.684
Baseline + Base-pair probability (Standalone) (cfg 3)	ROCKET	0.842	0.828	0.855	0.844	0.684	**0.855**	**0.856**	0.854	0.855	0.710	0.795	0.885	0.783	0.789	0.670
	MiniROCKET	0.817	0.820	0.814	0.816	0.634	0.836	0.834	0.838	0.836	0.673	0.772	0.872	0.757	0.764	0.632
	MultiROCKET	0.822	0.813	0.832	0.824	0.645	0.825	0.831	0.820	0.824	0.651	0.758	0.866	0.747	0.750	0.612
	Hydra	0.846	0.827	0.865	**0.849**	0.693	0.851	0.840	0.861	0.852	0.702	0.789	0.879	0.769	0.780	0.658
	MultiROCKET-Hydra	0.822	0.809	0.834	0.824	0.644	0.835	0.840	0.830	0.834	0.670	0.759	0.866	0.746	0.750	0.611
Baseline + Base-pair probability (Incorporated) (cfg 4)	ROCKET	0.799	0.771	0.827	0.805	0.600	0.809	0.786	0.832	0.813	0.619	0.737	0.850	0.712	0.724	0.573
	MiniROCKET	0.776	0.756	0.797	0.781	0.554	0.801	0.808	0.794	0.799	0.603	0.705	0.835	0.675	0.684	0.521
	MultiROCKET	0.814	0.801	0.828	0.817	0.630	0.816	0.812	0.820	0.816	0.634	0.726	0.848	0.706	0.712	0.556
	Hydra	0.822	0.787	0.857	0.828	0.647	0.834	0.828	0.840	0.835	0.669	0.759	0.862	0.734	0.746	0.608
	MultiROCKET-Hydra	0.814	0.802	0.820	0.817	0.629	0.820	0.825	0.816	0.819	0.642	0.736	0.853	0.717	0.723	0.874

Table 6: Channel ablation study.

are better. A horizontal bar connecting combinations indicates no statistically significant difference. From Figure 5, we can see that including time series derived from



(a) 5p (b) 3p (c) multi

Fig. 5: CD diagrams of channel ablation study.

Secondary Structure information and base-pair probability as a separate channel can significantly improve the performance of the classifiers. Incorporating the base-pair probability sequence in the time series encoding of the complementary strand can also improve the classifier, but to a minor degree compared to serving as a separate channel. In our downstream analysis, we adopted the combination of strand time series, secondary structure time series, and base-pair probability time series as our multivariate time series input.

3.2 Predictive performance

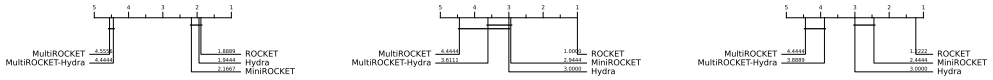
The experiment was conducted on three datasets: the 5p-arm, the 3p-arm, and the multi-class datasets. Recall that we have 9 encoding methods and 5 ROCKET-based classifiers. It results in 45 combinations of encoding methods and classifiers.

The result is shown in Table 7. The best combination of encoding method and classifier is shown in Table 8. For the 5p-arm dataset, the best combination is "Global Cumulative grouped fixed-length channel mapping + ROCKET". For all five classification metrics, it outperforms the state-of-the-art (SOTA) method, DiCleave. For the 3p-arm dataset, the best combination is "Global Cumulative grouped fixed-length channel mapping + ROCKET". Out of the five classification metrics, it outperforms

	Classifier	5p arm					3p arm					Multi				
		Acc	Sp	Sn	F1	MCC	Acc	Sp	Sn	F1	MCC	Acc	Sp	Sn	F1	MCC
Single value mapping (enc 1)	ROCKET	0.849	0.842	0.857	0.851	0.699	0.863	0.854	0.873	0.865	0.727	0.853	0.917	0.847	0.851	0.764
	MiniROCKET	0.823	0.809	0.837	0.825	0.647	0.823	0.828	0.817	0.822	0.647	0.835	0.906	0.828	0.833	0.735
	MultiROCKET	0.821	0.802	0.840	0.824	0.643	0.839	0.826	0.852	0.841	0.679	0.811	0.894	0.806	0.809	0.697
	Hydra	0.843	0.820	0.867	0.847	0.688	0.838	0.819	0.857	0.841	0.677	0.831	0.901	0.815	0.827	0.727
	MultiROCKET-Hydra	0.820	0.803	0.837	0.823	0.640	0.840	0.830	0.850	0.841	0.680	0.816	0.896	0.810	0.814	0.704
Grouped variable-length channel mapping (enc 2)	ROCKET	0.835	0.826	0.844	0.836	0.670	0.855	0.849	0.861	0.856	0.710	0.846	0.913	0.839	0.844	0.752
	MiniROCKET	0.843	0.833	0.853	0.844	0.686	0.831	0.821	0.842	0.833	0.663	0.837	0.907	0.828	0.834	0.737
	MultiROCKET	0.819	0.809	0.828	0.820	0.638	0.817	0.814	0.820	0.818	0.634	**0.890**	0.894	0.806	0.808	0.695
	Hydra	0.825	0.780	0.869	0.832	0.653	0.811	0.769	0.854	0.819	0.626	0.818	0.892	0.765	0.812	0.705
	MultiROCKET-Hydra	0.818	0.814	0.822	0.819	0.636	0.831	0.825	0.837	0.832	0.662	0.820	0.900	0.815	0.818	0.710
Grouped fixed-length channel mapping (enc 3)	ROCKET	0.851	0.843	0.859	0.852	0.702	0.863	0.850	0.875	0.864	0.726	0.849	0.915	0.843	0.847	0.757
	MiniROCKET	0.844	0.836	0.853	0.845	0.689	0.840	0.826	0.855	0.843	0.682	0.851	0.915	0.844	0.849	0.760
	MultiROCKET	0.831	0.815	0.848	0.834	0.663	0.824	0.813	0.836	0.826	0.649	0.811	0.896	0.808	0.808	0.698
	Hydra	0.848	0.816	0.880	0.853	0.699	0.862	0.839	0.884	0.864	0.724	0.843	0.908	0.837	0.839	0.746
	MultiROCKET-Hydra	0.836	0.813	0.859	0.839	0.672	0.833	0.820	0.845	0.835	0.665	0.828	0.905	0.824	0.826	0.725
Cumulative mapping (enc 4)	ROCKET	0.850	0.834	0.866	0.852	0.701	0.863	0.855	0.871	0.864	0.726	0.852	0.915	0.842	0.850	0.762
	MiniROCKET	0.840	0.821	0.860	0.843	0.682	0.840	0.837	0.844	0.841	0.682	0.843	0.911	0.835	0.840	0.747
	MultiROCKET	0.822	0.809	0.834	0.824	0.644	0.832	0.830	0.834	0.832	0.665	0.820	0.898	0.810	0.816	0.709
	Hydra	0.848	0.819	0.878	0.853	0.698	0.853	0.856	0.869	0.855	0.705	0.845	0.910	0.830	0.841	0.749
	MultiROCKET-Hydra	0.824	0.811	0.856	0.825	0.647	0.838	0.833	0.843	0.839	0.677	0.821	0.898	0.810	0.817	0.711
Cumulative grouped variable-length channel mapping (enc 5)	ROCKET	0.843	0.821	0.866	0.847	0.688	0.856	0.840	0.871	0.857	0.712	0.855	0.916	0.843	0.851	0.766
	MiniROCKET	0.845	0.826	0.865	0.848	0.691	0.836	0.833	0.838	0.836	0.672	0.840	0.909	0.833	0.838	0.742
	MultiROCKET	0.826	0.814	0.838	0.828	0.653	0.815	0.820	0.810	0.814	0.631	0.826	0.902	0.820	0.824	0.721
	Hydra	0.850	0.819	**0.880**	0.854	0.701	0.834	0.807	0.861	0.838	0.669	0.833	0.903	0.818	0.829	0.731
	MultiROCKET-Hydra	0.824	0.810	0.838	0.826	0.649	0.833	0.833	0.833	0.833	0.666	0.830	0.903	0.821	0.827	0.726
Cumulative grouped fixed-length channel mapping (enc 6)	ROCKET	0.856	0.836	0.876	0.858	0.712	**0.870**	**0.861**	**0.879**	**0.871**	**0.741**	0.863	**0.921**	**0.852**	**0.860**	**0.780**
	MiniROCKET	0.856	0.837	0.874	0.858	0.712	0.842	0.839	0.845	0.843	0.685	0.845	0.912	0.837	0.843	0.751
	MultiROCKET	0.820	0.802	0.839	0.824	0.642	0.798	0.798	0.798	0.798	0.597	0.809	0.894	0.806	0.807	0.694
	Hydra	0.850	0.814	0.885	0.855	0.701	0.855	0.840	0.869	0.857	0.711	0.847	0.910	0.831	0.843	0.752
	MultiROCKET-Hydra	0.820	0.801	0.839	0.823	0.641	0.807	0.813	0.802	0.806	0.615	0.821	0.900	0.817	0.819	0.713
Global Cumulative mapping (enc 7)	ROCKET	0.850	0.834	0.866	0.852	0.701	0.863	0.855	0.871	0.864	0.726	0.852	0.915	0.842	0.850	0.762
	MiniROCKET	0.847	0.832	0.862	0.849	0.695	0.848	0.839	0.857	0.850	0.697	0.845	0.911	0.836	0.843	0.750
	MultiROCKET	0.827	0.819	0.834	0.828	0.653	0.847	0.842	0.853	0.848	0.695	0.825	0.901	0.817	0.822	0.718
	Hydra	0.851	0.821	0.880	0.855	0.703	0.861	0.848	0.874	0.863	0.722	0.847	0.911	0.834	0.844	0.753
	MultiROCKET-Hydra	0.829	0.823	0.834	0.830	0.658	0.843	0.838	0.849	0.844	0.688	0.832	0.905	0.823	0.829	0.730
Global Cumulative grouped variable-length channel mapping (enc 8)	ROCKET	0.840	0.814	0.867	0.844	0.682	0.853	0.838	0.867	0.854	0.706	0.856	0.917	0.845	0.853	0.768
	MiniROCKET	0.848	0.834	0.862	0.850	0.697	0.841	0.824	0.859	0.844	0.683	0.844	0.911	0.856	0.842	0.748
	MultiROCKET	0.834	0.828	0.839	0.834	0.668	0.831	0.821	0.842	0.833	0.663	0.828	0.904	0.823	0.826	0.724
	Hydra	0.857	0.821	0.894	0.862	**0.717**	0.822	0.786	0.857	0.828	0.645	0.826	0.898	0.806	0.820	0.717
	MultiROCKET-Hydra	0.837	0.834	0.839	0.837	0.674	0.834	0.827	0.840	0.835	0.668	0.835	0.907	0.828	0.832	0.734
Global Cumulative grouped fixed-length channel mapping (enc 9)	ROCKET	0.856	0.836	0.876	0.858	0.712	**0.870**	**0.861**	**0.879**	**0.871**	**0.741**	0.863	**0.921**	**0.852**	**0.860**	**0.780**
	MiniROCKET	**0.857**	**0.845**	0.870	**0.859**	0.715	0.840	0.821	0.859	0.843	0.681	0.844	0.911	0.837	0.842	0.749
	MultiROCKET	0.829	0.825	0.833	0.830	0.658	0.820	0.816	0.823	0.820	0.640	0.819	0.900	0.816	0.817	0.710
	Hydra	0.856	0.817	0.894	0.861	0.713	0.859	0.838	0.880	0.862	0.719	0.846	0.911	0.832	0.843	0.752
	MultiROCKET-Hydra	0.829	0.824	0.834	0.830	0.658	0.822	0.825	0.819	0.821	0.644	0.827	0.904	0.823	0.824	0.722

Table 7: Performance on the 45 combinations between encoding methods and the transformation methods.

DiCleave, except in specificity. For the multi-class dataset, the best combination is "Global Cumulative grouped fixed-length channel mapping + ROCKET". For all five classification metrics, it outperforms DiCleave.



(a) 5p (b) 3p (c) multi

Fig. 6: CD diagrams to compare different transformation methods.

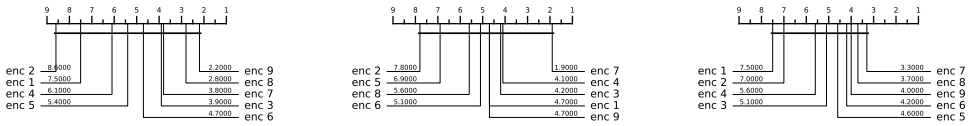
To summarize the result, we plot the CD diagrams to find the general best classifier, as shown in Figure 6 and the general best encoding method, as shown in Figure 7.

3.3 Running time analysis

To compare the computational efficiency of MTSCCleav and DiCleave, we conducted a comparative analysis of their running times. For DiCleave, we employed the code from

Dataset	Methods	Acc	Sp	Sn	F1	MCC	Time (s)
5p-arm	enc 9 + MiniROCKET	**0.857**	**0.845**	**0.870**	**0.859**	**0.715**	**0.787**
	DiCleave	0.818	0.790	0.846	0.822	0.653	21.249
3p-arm	enc 9 + ROCKET	**0.870**	0.861	**0.879**	**0.871**	**0.741**	4.311
	enc 7 + MiniROCKET	0.848	0.839	0.857	0.850	0.697	**0.989**
	DiCleave	0.854	**0.891**	0.817	0.847	0.715	15.919
multi-class	enc 9 + ROCKET	**0.863**	**0.921**	**0.852**	**0.860**	**0.780**	12.208
	enc 3 + MiniROCKET	0.851	0.915	0.844	0.849	0.760	**4.550**
	DiCleave	0.820	0.895	0.804	0.815	0.710	131.151

Table 8: Comparative analysis between MTSCCleav with the best combination of the encoding method and transformation method, with the SOTA, DiCleave, on the three datasets. The best results of using MiniROCKET have also been shown to compare the computational efficiency.



(a) 5p (b) 3p (c) multi

Fig. 7: CD diagrams to compare different encoding methods.

its supporting website, without any modifications. All experiments were conducted on the same machine (a personal laptop equipped with an Apple M1 Pro chip and 16 GB of memory) and using the same splits of the training and test datasets under 5-fold cross-validation to ensure fairness. The reported running times are the averages of the five runs. The timing results were measured from the training phase to the return of the five classification metrics. The result is shown in Table 8. MiniROCKET is the most computationally efficient of the five rocket-based classifiers. We also included its best result, along with the corresponding encoding method, even though this combination may not be the best overall. From Table 7, the best results on the 5p-arm, 3p-arm, and multi-class datasets are from the combinations of "enc 9 + MiniROCKET", "ecv 9 + ROCKET" and "enc 9 + ROCKET", respectively. Note that for the 3p-arm and the multi-class datasets, the combination of "enc 6 + ROKCET" also returns the same result.

MTSCCleav demonstrated a significant advantage in computational efficiency, achieving an average 27.0X, 3.7X, and 10.7X speedup over DiCleave, for the 5p-arm, 3p-arm, and multi-class datasets, respectively. If we also consider using the MiniROCKET in the case of 3p-arm and multi-class datasets, it achieves 16.1X and 28.8X speedup. To note, in the case of the 3p-arm dataset, the performance of MiniROCKET is only slightly lower than DiCleave. In the case of the multi-class dataset, even the performance of MiniROCKET is better than DiCleave. DiCleave

is a deep learning-based method that requires substantial time for model inference, while MTSCleav leverages efficient ROCKET-based classifiers. This significant reduction in runtime makes MTSCCleav more suitable for large-scale data and real-time applications.

3.4 Subsequence importance

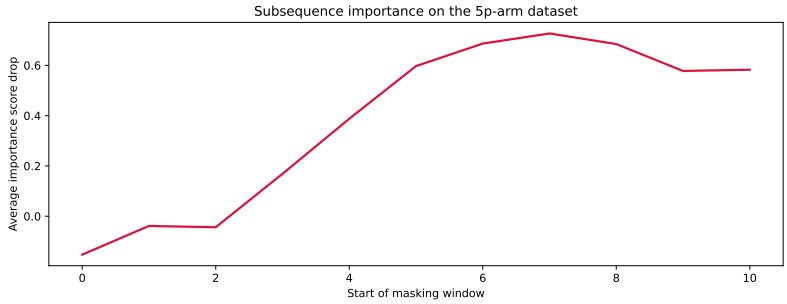
To evaluate the sensitivity of MTSCCleav to subsequences of the original input, we conducted a perturbation experiment to evaluate the importance of subsequences based on masking windows. The goal of this experiment is to identify which subsequences of the entire time series are critical for classification. The primary objective is to examine how various modifications to the original input impact model performance. It suggests which features are essential for classification.

The model is trained on the original training dataset. For each instance in the test dataset, we measure its original score and the masked score. We slid a masking window w with a fixed length $|w|$ over the input time series T. $|w|$ is set to 4. For each window position $i \in \{1, 2, ..., |T| - |w| + 1\}$, we masked all entries across all the channels of T within the window. Hence, we removed or hid that portion of information from the model during inference. The changes in classification performance in terms of accuracy relative to the unmasked original input of each i are recorded. Intuitively, if the information of a subsequence is critical for the classification, the masking of this subsequence would lead to a great drop in classification performance. We aggregated the importance score across the test dataset. The result of the perturbation experiment for the 5p-arm and 3p-arm datasets is shown in Figure 8. For the encoding methods, we cannot use the methods derived from the cumulative mapping because the accumulation would leak information from the masked region. We adopted "Grouped fixed-length channel mapping" as the encoding method and ROCKET as the classifier. "Grouped fixed-length channel mapping" is the general best encoding, other than the methods derived from the cumulative mapping, in all datasets. ROCKET is the overall best classifier.

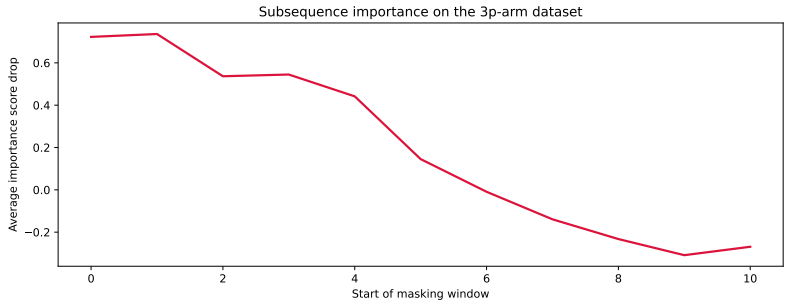
In the 5p-arm dataset, we found that masking subsequences at the end caused a significant drop in the importance score, as shown in Figure 8 (a). In the 3p-arm dataset, we found that masking subsequences at the beginning caused a significant drop in the importance score, as shown in Figure 8 (b).

4 Discussion

Our results show that MTSCCleav outperforms the SOTA, DiCleave, in both accuracy and efficiency. The channel ablation study reveals that the involvement of the time series derived from the secondary structure can improve accuracy. It suggests the importance of RNA folding in Dicer processing. Additionally, we found that the base-pair probability sequence of the secondary structure can also enhance accuracy. To the best of our knowledge, it is a novel application of the base-pair probability sequence. We can make use of the probability sequence either by involving it in the time series encoding of the complementary strand or as a standalone channel in the multivariate time series. Experiments show that using the probability sequence as an additional



(a) 5p-arm



(b) 3p-arm

Fig. 8: Results of the perturbation experiment on the 5p-arm and the 3p-arm datasets.

channel can enhance accuracy more than involving it in the encoding. It is likely because keeping it as an additional channel can preserve more information, of both the probability sequence itself and the encoded complementary strand.

Out of the three datasets, the best classifier is ROCKET. The ranking of the five classifiers by performance, starting from the best, is as follows: ROCKET, Hydra, MiniROCKET, MultiROCKET-Hydra, MultiROCKET. It indicates that the features created from the pooling operations of the activation map, which are only in MultiROCKET but not in MiniROCKET, confuse the final classifier. They are mean of positive values (MPV), mean of indices of positive values (MIPV) and longest stretch of positive values (LSPV) [38]. In contrast, the pooling operator, which is only present in ROCKET but not in MiniROCKET, enhances the classification performance. It is maximum (MAX). For the encoding methods, we have the following observations. Fixed-length grouped channel mappings outperform variable-length counterparts with one exception in the multi-class dataset, likely because fixed-length schemes better preserve the original positional information of nucleotides within the sequence. Global cumulative methods, which compute cumulative mappings starting from the beginning of the sequence, consistently yield better performance than local cumulative methods, with one exception in the 3p dataset. It suggests that the upstream information

of the cleavage pattern plays a critical role in identifying cleavage sites. Cumulative-based encodings perform better than single-value mappings, with one exception in the 3p dataset, suggesting that the accumulated nucleotide signal is more informative for cleavage site prediction than the local or isolated presence of nucleotides. In the 5p dataset, encoding in two channels rather than one channel appears to worsen the result. This suggests that the same grouping method used for the 5p-arm dataset cannot be applied to the 3p-arm dataset.

One limitation of DiCleave is overfitting during training because of the relatively small size of the dataset [13]. DiCleave is a deep learning-based method. Deep learning models typically require a large amount of training data to generalize effectively. They are data-hungry. In contrast, MTSCCleav leverages ROCKET-based methods for the classification. They rely on random convolutional feature extraction followed by a linear classifier. A ridge classifier was used in this study. Ridge classifiers are less data-hungry compared to deep learning methods due to their use of L2 regularization and the simplicity of their linear model nature. It allows ROCKET-based classifiers, and hence MTSCCleav, to maintain strong predictive performance even in settings with a relatively small dataset size.

The subsequence importance of MTSCCleav reveals some connections between RNA secondary structure and human dicer cleavage site prediction. The perturbation experiment shows that the leading part of 5p-arm and the tailing part of 3p-arm are important for the classification. These parts are close to the center of the RNA secondary structure of pre-miRNA. It indicates that the center region is more crucial for human dicer cleavage site prediction. It agrees with the previous study [12].

5 Conclusions

We proposed a simple, fast, and accurate multivariate time series classification-based method, termed MTSCCleav, for predicting human dicer cleavage sites by transforming nucleotide sequences and the secondary structures into time series. Base-pair probability sequences of the secondary structures have also been used in the method. MTSCCleav consists of three parts: time series encoding, time series transformation, and classification. ROCKET-based methods were used for time series transformation. Ridge Classifier was used for classification. For the computational experiments, we evaluated nine time series encoding methods with five time series transformation methods. For the three datasets: 5p-arm, 3p-arm, and multi-class, MTSCCleav outperformed the SOTA methods in all five evaluation metrics for the 5p-arm and multi-class datasets, and four of the metrics for the 3p-arm dataset. In terms of computational efficiency, MTSCCleav with the optimal setting achieves an average 3.7X to 27.0X speedup over the SOTA method on the three datasets. With the use of a less accurate but faster time series transformation method, MTSCCleav achieves an average speedup of 16.1X to 28.8X, respectively. We analyzed the subsequence importance of the input multivariate time series. The results show that subsequences near the center of the pre-miRNA sequences are more important. This aligns with the findings from previous work.

Overall, this study demonstrates that time series analysis provides a powerful alternative to conventional modeling in the context of RNA processing. This framework may be extended to other RNA-processing tasks.

References

[1] Urry, L.A., Cain, M.L., Wasserman, S.A., Minorsky, P.V., Orr, R.B., Campbell, N.A.: Campbell Biology, Twelfth edition edn., New York, NY (2020)

[2] Alberts, B.: Molecular Biology of the Cell, Seventh edition edn., New York (2022)

[3] Cohen, W.W.: A Computer Scientist's Guide to Cell Biology: A Travelogue from a Stranger in a Strange Land, New York, NY (2007)

[4] Bartel, D.P.: Micrornas: Genomics, biogenesis, mechanism, and function. Cell **116**(2), 281–297 (2004)

[5] Lee, Y., Jeon, K., Lee, J.-T., Kim, S., Kim, V.N.: Microrna maturation: Stepwise processing and subcellular localization. The EMBO Journal **21**(17), 4663–4670 (2002)

[6] Nguyen, T.D., Trinh, T.A., Bao, S., Nguyen, T.A.: Secondary structure rna elements control the cleavage activity of dicer. Nature Communications **13**(1), 2138 (2022)

[7] Gu, S., Jin, L., Zhang, Y., Huang, Y., Zhang, F., Valdmanis, P.N., Kay, M.A.: The loop position of shrnas and pre-mirnas is critical for the accuracy of dicer processing in vivo. Cell **151**(4), 900–911 (2012)

[8] Feng, Y., Zhang, X., Graves, P., Zeng, Y.: A comprehensive analysis of precursor microrna cleavage by human dicer. RNA **18**(11), 2083–2092 (2012)

[9] MacRae, I.J., Zhou, K., Doudna, J.A.: Structural determinants of rna recognition and cleavage by dicer. Nature Structural & Molecular Biology **14**(10), 934–940 (2007)

[10] Ahmed, F., Kaundal, R., Raghava, G.P.: Phdcleav: A svm based method for predicting human dicer cleavage sites using sequence and secondary structure of mirna precursors. BMC Bioinformatics **14**(14), 9 (2013)

[11] Bao, Y., Hayashida, M., Akutsu, T.: Lbsizecleav: Improved support vector machine (svm)-based prediction of dicer cleavage sites using loop/bulge length. BMC Bioinformatics **17**(1), 487 (2016)

[12] Liu, P., Song, J., Lin, C.-Y., Akutsu, T.: Recgbm: A gradient boosting-based method for predicting human dicer cleavage sites. BMC Bioinformatics **22**(1), 63 (2021)

[13] Mu, L., Song, J., Akutsu, T., Mori, T.: Dicleave: A deep learning model for predicting human dicer cleavage sites. BMC Bioinformatics **25**(1), 13 (2024)

[14] Mu, L., Akutsu, T.: DiCleavePlus: A Transformer-Based Model to Detect Dicer Cleavage Sites within Cleavage Patterns (2025)

[15] Middlehurst, M., Schäfer, P., Bagnall, A.: Bake off redux: A review and experimental evaluation of recent time series classification algorithms. Data Mining and Knowledge Discovery **38**(4), 1958–2031 (2024)

[16] Griffiths-Jones, S., Saini, H.K., van Dongen, S.: mirbase: Tools for microrna genomics. Nucleic Acids Research **36**(suppl_1), 154–158 (2008)

[17] Xu, T., Su, N., Liu, L., Zhang, J., Wang, H., Zhang, W., Gui, J., Yu, K., Li, J., Le, T.D.: mirbaseconverter: An r/bioconductor package for converting and retrieving mirna name, accession, sequence and family information in different versions of mirbase. BMC Bioinformatics **19**(19), 514 (2018)

[18] Zvelebil, M.J., Baum, J.O., Zvelebil, M.: Understanding Bioinformatics, New York (2008)

[19] Lorenz, R., Bernhart, S.H., Höner zu Siederdissen, C., Tafer, H., Flamm, C., Stadler, P.F., Hofacker, I.L.: Viennarna package 2.0. Algorithms for Molecular Biology **6**(1), 26 (2011)

[20] Aggarwal, C.C.: Data Mining: The Textbook, Cham (2015)

[21] Mendizabal-Ruiz, G., Román-Godínez, I., Torres-Ramos, S., Salido-Ruiz, R.A., Morales, J.A.: On dna numerical representations for genomic similarity computation. PLOS ONE **12**(3), 0173288 (2017)

[22] Sahu, S.S., Panda, G.: Identification of protein-coding regions in dna sequences using a time-frequency filtering approach. Genomics, Proteomics & Bioinformatics **9**(1), 45–55 (2011)

[23] Ravichandran, L., Papandreou-Suppappola, A., Spanias, A., Lacroix, Z., Legendre, C.: Time-frequency based biological sequence querying. In: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4174–4177 (2010)

[24] Yin, C., Yin, X.E., Wang, J.: A novel method for comparative analysis of dna sequences by ramanujan-fourier transform. Journal of Computational Biology **21**(12), 867–879 (2014)

[25] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., Keogh, E.: Searching and mining trillions of time series subsequences

under dynamic time warping. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '12, pp. 262–270, New York, NY, USA (2012)

[26] Cristea, P.D.: Conversion of nucleotides sequences into genomic signals. Journal of Cellular and Molecular Medicine **6**(2), 279–303 (2002)

[27] Chakravarthy, N., Spanias, A., Iasemidis, L.D., Tsakalis, K.: Autoregressive modeling and feature analysis of dna sequences. EURASIP Journal on Advances in Signal Processing **2004**(1), 1–16 (2004)

[28] Zhao, J., Yang, X.W., Li, J.P., Tang, Y.Y.: Dna sequences classification based on wavelet packet analysis. In: Wavelet Analysis and Its Applications, pp. 424–429 (2001)

[29] Holden, T., Subramaniam, R., Sullivan, R., Cheung, E., Schneider, C., Jr, G.T., Flamholz, A., Lieberman, D.H., Cheung, T.D.: Atcg nucleotide fluctuation of deinococcus radiodurans radiation genes. In: Instruments, Methods, and Missions for Astrobiology X, vol. 6694, pp. 402–411 (2007)

[30] Nair, A.S., Sreenadhan, S.P.: A coding measure scheme employing electron-ion interaction pseudopotential (eiip). Bioinformation **1**(6), 197–202 (2006)

[31] Akhtar, M., Epps, J., Ambikairajah, E.: On dna numerical representations for period-3 based exon prediction. In: 2007 IEEE International Workshop on Genomic Signal Processing and Statistics, pp. 1–4 (2007)

[32] Zhang, R., Zhang, C.-T.: Z curves, an intutive tool for visualizing and analyzing the dna sequences. Journal of Biomolecular Structure and Dynamics **11**(4), 767–782 (1994)

[33] Berger, J.A., Mitra, S.K., Carli, M., Neri, A.: Visualization and analysis of dna sequences using dna walks. Journal of the Franklin Institute **341**(1), 37–53 (2004)

[34] Ruiz, A.P., Flynn, M., Large, J., Middlehurst, M., Bagnall, A.: The great multivariate time series classification bake off: A review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **35**(2), 401–449 (2021)

[35] Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **31**(3), 606–660 (2017)

[36] Dempster, A., Petitjean, F., Webb, G.I.: Rocket: Exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery **34**(5), 1454–1495 (2020)

[37] Dempster, A., Schmidt, D.F., Webb, G.I.: Minirocket: A very fast (almost) deterministic transform for time series classification. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. KDD '21, pp. 248–257, New York, NY, USA (2021)

[38] Tan, C.W., Dempster, A., Bergmeir, C., Webb, G.I.: Multirocket: Multiple pooling operators and transformations for fast and effective time series classification. Data Mining and Knowledge Discovery **36**(5), 1623–1646 (2022)

[39] Dempster, A., Schmidt, D.F., Webb, G.I.: Hydra: Competing convolutional kernels for fast and accurate time series classification. Data Mining and Knowledge Discovery **37**(5), 1779–1805 (2023)

[40] Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA) - Protein Structure **405**(2), 442–451 (1975)

[41] Gorodkin, J.: Comparing two k-category assignments by a k-category correlation coefficient. Computational Biology and Chemistry **28**(5), 367–374 (2004)