Санкт-Петербургский государственный университет

Технологии программирования

Кафедра Системного Программирования

Группа 21.Б09-мм

Чистякова Анна Артуровна

Улучшение качества предсказания вторичной структуры РНК

Отчет по учебной практике

в форме «Эксперимент»

Научный руководитель:
кандидат физико-математических наук, доцент кафедры информатики
С. В. Григорьев

Санкт-Петербург
2022

Содержание

Введение

Современная наука сталкивается с множеством прикладных задач в области биоинформатики. Одним из ее направлений является исследование макромолекулы РНК. Ни для кого не секрет, что РНК выполняет множество важнейших функций в организме для обеспечения функционирования тканей и органов. Поэтому она является важным объектом для изучения.

Молекула РНК представляет собой цепочку, образованную четырьмя нуклеотидами: аденином, гуанином, цитозином и урацилом. При этом взаимодействовать между собой они могут не только последовательно, образуя первичную (линейную) структуру РНК, но и через некоторое количество нуклеотидов, образуя при этом вторичную (пространственную) структуру (рис. 1), которая содержит в себе информацию о функциях клетки и об эволюционном происхождении организма. Также вторичная структура РНК играет важную роль во внутриклеточных процессах. Так, например, одна из важнейших ролей вторичной структуры мРНК – регуляция синтеза белка [3].

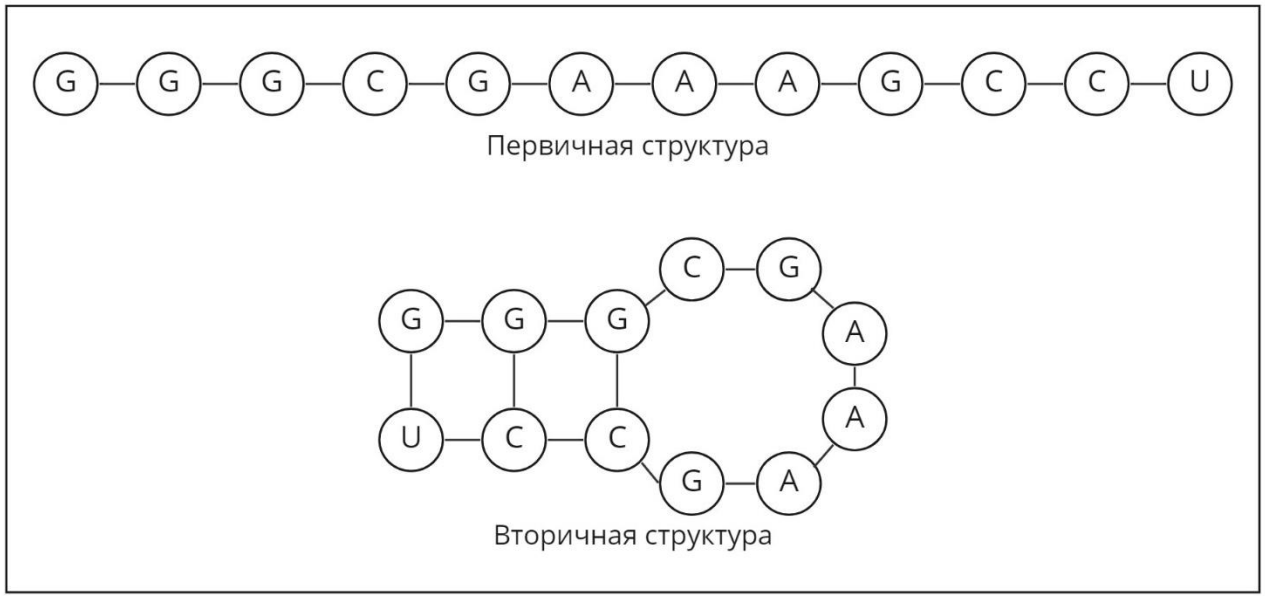

Рис. 1 Примеры первичной и вторичной структур РНК

Так как вторичная структура РНК несет в себе много полезной информации, ее определение является популярным направлением современной науки. Для этого существуют разные способы [6]: вычислительные и экспериментальные. Самые точные результаты достигаются в лабораторных условиях, однако такие эксперименты требуют большого количества ресурсов и времени, поэтому активно развиваются альтернативные методы, основанные на вычислениях. Одним из таких методов является использование машинного обучения.

Чтобы развивать эту область с помощью нейронных сетей, нужно иметь достаточно большой и точный набор данных для обучения и валидации модели. Как уже было сказано ранее, наибольшей точностью обладают вторичные структуры, полученные в лабораторных условиях, но из-за сложности проведения экспериментов, данных, полученных таким способом, не так много. Есть вероятность, что с увеличением их объема может также увеличится и точность предсказания.

Данная работа посвящена увеличению точности определения пространственной структуры РНК с помощью машинного обучения, основываясь на уже существующем решении [1,4], путем расширения имеющегося набора данных.

1. Обзор

Данная работа основана на уже созданном ранее инструменте [4], поэтому этот раздел посвящен обзору его архитектуры и подготовки данных.

1.1 Архитектура решения

Предложенный инструмент работает следующим образом. Сначала цепочка РНК обрабатывается с помощью синтаксического анализа, который преобразует ее в черно-белое изображение, где пиксели на главной диагонали – это нуклеотиды, обозначенные разными цветами, а белые пиксели в строке i и столбце j – потенциальные связи между нуклеотидами, находящимися в соответствующих строке и столбце. Эти связи устанавливаются на основе Уотсон-Криковских правил и задачи поиска рекурсивных композиций шпилек высотой больше трех. Затем полученные изображения обрабатываются нейронной сетью, которая убирает ненужные связи и добавляет нужные. Схема работы инструмента представлена на рис. 2.

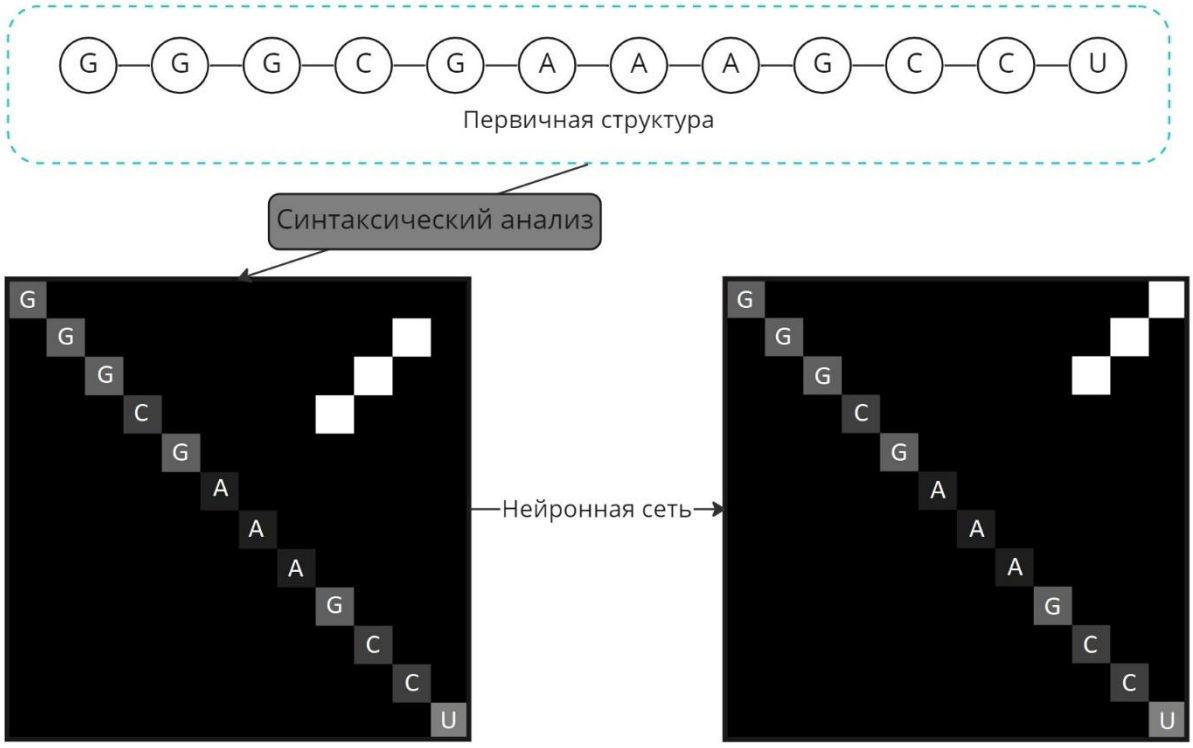

Рис. 2 Пример работы описываемого инструмента

Модель имеет параллельную остаточную архитектуру (рис. 3), представляющую собой 4 остаточные сети, которые обучаются параллельно и состоят из 5 остаточных блоков.

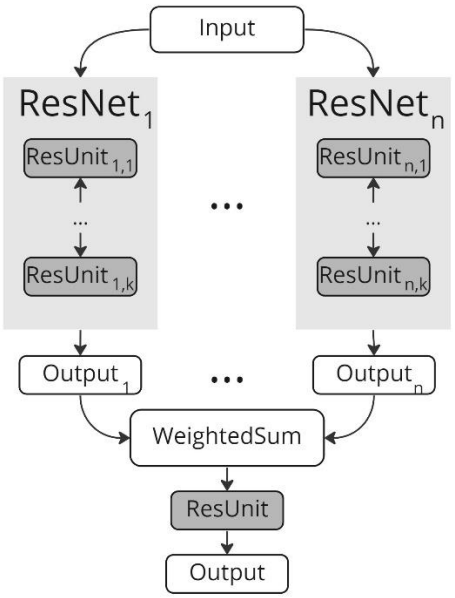

Рис. 3 Архитектура нейронной сети

1.2 Подготовка данных

Исходная нейронная сеть была обучена на данных из базы RNAstrand [2], которая содержит информацию, проверенную путем лабораторных экспериментов и эволюционного анализа вторичными структурами. В наборе содержится 801 цепочка, длина которых варьируется от 8 до 100 нуклеотидов. При этом последовательностей разного размера содержится неодинаковое количество, распределение длин в выборке можно увидеть на рис. 4. Перед каждой эпохой данные распределяются по батчам размера 4, содержащим последовательности одной длины. Если количество цепочек определенного размера оказывается меньше четырех или не кратно его размеру, то данные дублируются до необходимого объема.

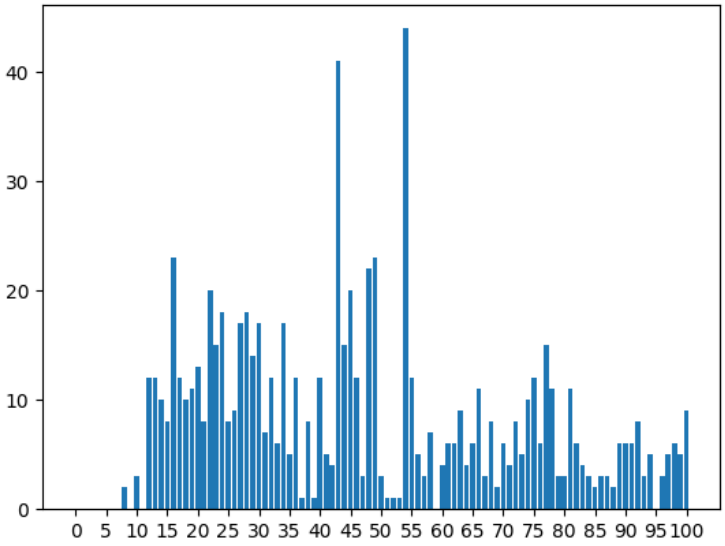

Рис. 4 Распределение длин последовательностей в исходной выборке

1.3 Обучение модели

Оценка качества модели в ходе обучения осуществлялась с помощью трех метрик: Precision, Recall и F1. Значения Precision и Recall вычислялись исходя из попиксельной разницы между входным и эталонным изображениями. Далее в формулах обозначения TP (true positive), FP (false positive) и FN (false negative) означают количество правильно определенных белых пикселей, ошибочно определенных белых пикселей и ошибочно определенных черных пикселей соответственно.

$Precision = \frac{TP}{TP+FP}$ – доля найденных контактов, которые действительно есть в эталонном изображении;

$Recall = \frac{TP}{TP+FN}$ – доля найденных связей среди всех искомых;

$F1 = 2 * \frac{Precision*Recall}{Precision+Recall}$ – гармоническое среднее Precision и Recall.

2. Постановка задач

Целью данной работы является увеличение точности предсказания вторичной структуры РНК. Для её достижения были поставлены следующие задачи.

1) Реализация алгоритма для расширения набора данных путем добавления полей к изображениям.
2) Расширение набора данных.
3) Проведение экспериментов с различными гиперпараметрами модели.

3. Расширение набора данных

Исходя из особенностей описанного ранее набора данных, использованного для обучения исходной модели, смысл его расширения заключался не только в том, чтобы увеличить количество данных, но и чтобы минимизировать дублирование входных изображений перед формированием батча.

Для начала было решено сгенерировать несколько десятков тысяч изображений 4-х размеров (16x16, 32x32, 64x64 и 128x128) и провести эксперименты с таким набором данных. При этом планировалось, что сгенерированные изображения будут размером не больше, чем 2n-1, где n – размер исходного изображения.

Исходный набор данных был разделен на 2 части: тренировочные данные (~70%) и тестовая выборка (~30%). Затем набор для обучения был разделен на 4 части: последовательности размером от 8 до 14, от 15 до 30, от 31 до 62 и от 63 до 100 и к ним были добавлены поля с разных сторон так, чтобы получилось изображение нужного размера. При этом учитывалось количество имеющихся цепочек данного размера, и если, например, цепочек было слишком мало, то к изображению добавлялось больше вариантов различных полей, увеличивая при этом количество экземпляров данной длины.

На рис. 5 можно увидеть фрагмент кода, в котором рассчитывается нужное количество изображений, исходя из количества цепочек. Функция получает на вход словарь, в котором длинам цепочек сопоставляются списки с названиями изображений данной длины (pictures_info), количество различных размеров цепочек в данной части набора данных (len_num) и нужное количество изображений (required_pictures_num), которое должно получиться из этой части исходной выборки. На выходе получается словарь, содержащий информацию о том, сколько картинок каждой длины нужно сгенерировать. Сначала каждой длине сопоставляется целая часть частного нужного количества картинок (required_pictures_num) и количества вариантов длин (len_num), а затем, если сумма всех получившихся значений меньше, чем заданное количество изображений, то к ним циклически прибавляется по единице до достижения нужной суммы всех значений. Таким образом, в каждой части набора данных количество изображений цепочек различных длин отличается не больше, чем на один, но при этом изображения, полученные из одной и той же цепочки имеют неодинаковые поля, то есть по сути являются разными входными данными.

```python
def get_required_num(pictures_info, len_num, required_pictures_num):
    required_num = {}
    for seq_len in pictures_info:
        required_num[seq_len] = required_pictures_num // len_num
    for seq_len in required_num:
        if sum(required_num.values()) != required_pictures_num:
            required_num[seq_len] += 1
    return required_num
```

Рис. 5 Функция, определяющая нужное количество изображений

Всего изображений размера 16x16 получилось 160, 32x32 – 768, 64x64 – 372, а 128x128 – 20000. При генерировании полей возникло ограничение на количество возможных изображений, которое регулировалось количеством последовательностей определенного размера и желаемым распределением. Например, в первой части исходной выборки (диапазон длин цепочек 8-14) изображений размера 14x14 8 штук. Из каждого изображения такого размера можно получить 4 различных картинки размером 16x16. Всего в этой части набора данных 5 вариантов длин последовательностей. Значит, чтобы количество картинок цепочек различных длин отличалось не более, чем на одну, можно сгенерировать 8*4*5=160 изображений. Аналогично были получены значения для 2 и 3 частей выборки, а для четвертой части значение было уменьшено для того, чтобы длинных цепочек было не сильно больше, чем остальных. Но, как видно из распределения, изображенного на рис. 6, длинных цепочек все равно получилось значительно больше, что было сделано для достижения достаточного объема набора данных, так как планировалось сгенерировать несколько десятков тысяч изображений.

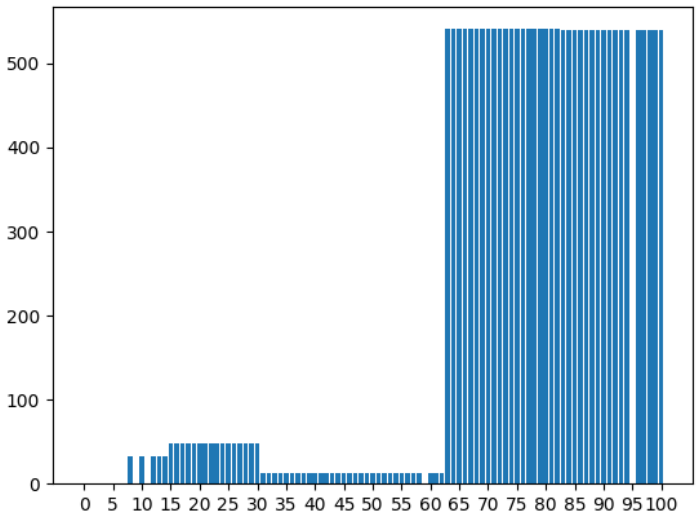

Рис. 6 Распределение длин последовательностей в расширенной выборке

Для преобразования изображений была использована библиотека Pillow [5], которая содержит в себе весь необходимый функционал и является простой в использовании.

4. Проведение экспериментов

В ходе работы было проведено несколько экспериментов. Для оценки модели во время их проведения были использованы те же метрики, что и в ранее созданной модели (Precision, Recall и F1). Для вычисления ошибки использовалось значение корня среднеквадратической ошибки (RMSE).

4.1 Результаты

Один из показателей того, что точность предсказания вторичной структуры после обучения может увеличиться – постоянное уменьшение значения ошибки и увеличение метрики F1 с каждой эпохой. Этому критерию соответствовал эксперимент с периодическим изменением показателя метода исключения (Dropout).

Сначала значение параметра было равным 0,65 в параллельных остаточных сетях и 0,4 – в последнем остаточном блоке. Обучение продолжалось 16 эпох, так как примерно после 16 эпохи начиналось переобучение модели (уменьшение значения ошибки, и в то же время уменьшение метрики F1). Затем этот показатель был снижен до 0,5 и 0,1 соответственно, и обучение длилось 24 эпохи. После чего везде было установлено значение 0,1 и обучение длилось 50 эпох. В ходе вышеописанного процесса удалось достичь значения F1≈0,24 на валидационной выборке. Хоть модель и показывала прогресс на валидационном наборе данных с увеличением количества эпох, эксперимент было решено не продолжать, так как обученная модель показывала худшие результаты относительно исходной модели на тестовой выборке.

Также проводились эксперименты с изменением показателей регуляризации и распределения расширенного набора данных, но они не показали положительных результатов. Значения метрик в ходе обучения оставались постоянными, и на тестовой выборке модель, соответственно, показывала низкую точность предсказания.

Заключение

В данной работе было проведено исследование, направленное на улучшение качества предсказания вторичной структуры РНК, и были получены следующие результаты:

- Реализован алгоритм для расширения набора данных путем добавления полей к изображениям;
- Получен более объемный набор данных;
- Проведены эксперименты с применением расширенной выборки и с изменением гиперпараметров модели на основе существующего инструмента.

Ссылка на исходный код алгоритма для расширения набора данных: https://github.com/Anya497/DatasetExpander

Ссылка на инструмент, использованный для проведения экспериментов: https://github.com/LuninaPolina/SecondaryStructureAnalyzer/tree/master/secondary-structure-prediction/dataset-sizes

Развитие исследования

В ходе проведения исследования с учетом всех полученных показателей был намечен дальнейший план для достижения поставленной цели. Так как эксперименты с расширенным набором данных не показали положительных результатов, планируется использование графовых нейронных сетей. Основным их преимуществом является увеличение значимой информации в данных относительно незначимой, так как в случае с изображениями основная их часть – не имеющие значения черные пиксели, в то время как информации, имеющей значение, в процентном соотношении очень мало, что приводит к маленькому значению ошибки, но в то же время низкому показателю точности. С использованием графовых нейронных сетей вся информация в данных будет нести в себе полезную информацию, что гипотетически сможет улучшить точность предсказания вторичной структуры. Также в ходе частичного анализа этой области было обнаружено решение задачи предсказания связей в графе на основе библиотеки PyTorch Geometric [7], которое может оказаться полезным в ходе дальнейших исследований.

Список источников

1. Лунина П. С. Комбинирование нейронных сетей и синтаксического анализа для обработки вторичной структуры последовательностей. – 2019.
2. Andronescu M. et al. RNA STRAND: the RNA secondary structure and statistical analysis database //BMC bioinformatics. – 2008. – Т. 9. – №. 1. – С. 1-10.
3. Vandivier L. E. et al. The conservation and function of RNA secondary structure in plants //Annual review of plant biology. – 2016. – Т. 67. – С. 463.
4. Secondary Structure Analyzer. [Электронный ресурс]. URL: https://github.com/LuninaPolina/SecondaryStructureAnalyzer (дата обращения: 06.12.22).
5. Clark A. et al. Pillow //URL https://pillow. readthedocs. io/.(Visited on 06/12/22). – 2021.
6. Seetin M. G., Mathews D. H. RNA structure prediction: an overview of methods //Bacterial regulatory RNA. – 2012. – С. 99-122.
7. Fey M., Lenssen J. E. Fast graph representation learning with PyTorch Geometric //arXiv preprint arXiv:1903.02428. – 2019.
8. Dataset Expander. [Электронный ресурс]. URL: https://github.com/Anya497/DatasetExpander (дата обращения: 06.12.22).