

Due: 16 May 2024, 23:00, to [Submit](#)

ASSIGNMENT3

AIN422



Instructors: Dr. Cemil ZALLUHOĞLU

TA: Hayriye ÇELİKBİLEK

Course: AIN422 Spring 2024

Subject: Recurrent Neural Networks

Given: 26.04.2024

Due: 16.05.2024, 23:00, to [Submit](#)

Introduction

The [Submit](#) system is our department's assignment management web application for students and instructors.

Since we handle student and instructor needed requirements privately using our services, rather than having the requirements only for developing some software, we will collect the following assignments (excluding this assignment) using the [Submit](#) webpage.

Policy

- ❖ Be sure to complete the submission deadline. The deadline is 23:00, and you see 23:59 on the live because of the advance compensation of potential problems. **Last-minute excuses will not be tolerated.**
- ❖ Save all your work until the assignment is graded.
- ❖ You can ask your questions via [Piazza](#), and you are supposed to be aware of everything discussed on Piazza.
- ❖ You must submit your work using the file hierarchy as stated below.
- ❖ No other submission methods will be accepted (mail)

Academic Integrity

All work on assignments must be done **individually** unless stated otherwise. You are encouraged to discuss the given assignments with your classmates, but these discussions should be carried out **abstractly**. Discussions about a particular solution to a problem (either in actual code or pseudocode) will not be tolerated. In short, you are turning in someone else's work (from the internet), in whole or part, as your own will be considered a **violation of academic integrity**. The former condition also holds for the material on the web, as everything on the web has been written by someone else.

References for the Academic Integrity (AI):

https://academicintegrity.ucsd.edu/AI-Handbook-for-UCSD_2019.pdf,
academicintegrity.org/resources/facts-and-statistics

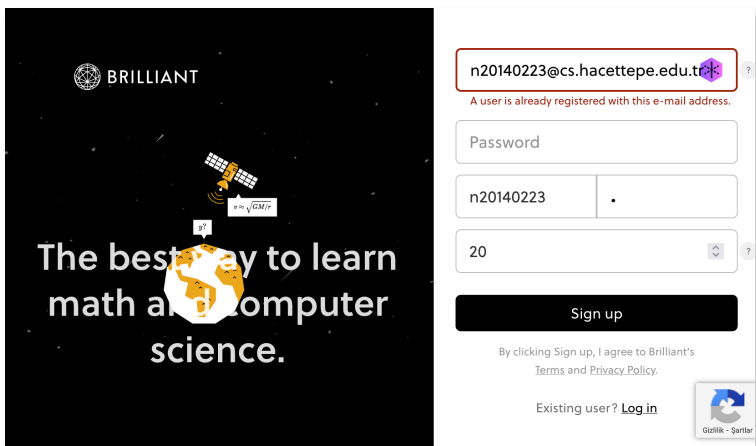
Joining Brilliant

Brilliant is an online learning platform that focuses on STEM-related topics. The 60+ courses on Science, Technology, Engineering and Math topics all offer an **interactive and hands-on learning experience**.

It is mandatory to use your **CS e-mail address** as your account, your **student ID b2XXXXXX** as your **name** and the text **'.'** (dot) as your **surname** while signing up (see Figure 1).

It is a must to use the **invitation link** to let course instructors see your solutions and time on the platform.

<https://brilliant.org/classroom/join/j72pbg/>



BRILLIANT

The best way to learn math and computer science.

n20140223@cs.hacettepe.edu.tr

A user is already registered with this e-mail address.

Password

n20140223 .

20

Sign up

By clicking Sign up, I agree to Brilliant's Terms and Privacy Policy.

Existing user? [Log in](#)

Gizlik - Şartlar

Figure 1. How to join brilliant.org

Work To Be Done

You are responsible for solving 10 of your selected lessons out of 28 in the Brilliant Classroom from the Artificial Neural Network class. (see Figure 2).



Figure 2. The overall display of the lessons.

If you cannot solve some of these examples correctly, do not worry; the interactive problems include the solutions if you make any mistakes. There will be no point deduction about the spent time and correctness in Assignment 1; we value your effort. Take your effort wisely.

These **lessons should take 5 to 15 minutes each**, and the problems are fundamental and similar to your AIN422 slide examples, so you do not have to worry.

Solving the whole lesson is recommended but optional to ensure you are comfortable with the semester.

The Data

Human species has genes (DNA) and their transcripts (RNA) as well as its metadata. In this assignment you are given two human genetic data **"GRCh38_latest_rna.fna"** and **"HGNC_results.txt"**.

The **"HGNC_results.txt"** data is a tabular metadata for human genes. It comprises ID, symbol, name, chromosome, locus group/type, and RefSeq accession information. You will try to predict the four different classes of locus groups of **given 45,729 genes** using the one-hot encoded text sequences (presumably for simplicity) as inputs of your RNN model. The ID, symbol, and name columns are human-given features of the genes. Locus group is a functional property of the individual gene; it determines how it operates. RefSeq accession is the ID for the reference sequence that you will search for in the **"GRCh38_latest_rna.fna"** data.

The **"GRCh38_latest_rna.fna"** data is a text file comprising 185,121 up-to-date human RNA sequences. You

will not use all of these sequences for your model. You will search for existing RefSeq accessions (e.g., NM_130786), and you will take the first occurrence of that sequence from the data as input (e.g.,>NM_130786.4 Homo sapiens alpha-1-B glycoprotein (A1BG), mRNA \nATTGCTGCA...). As you noticed, each sequence has a starting character of '>'; after that, it has a RefSeq ID, and then there comes a long description, which we ignore in the heading line of that sequence. After this discriptors, we reach the sequence of four alphabetical characters (A: adenine, T: thymine, G: guanine, C: cytosine). You may use other encoding techniques if you would like, but for simplicity's sake, you can take each character as a one-hot encoded vector in this sequence data. For example, you will take the vector [1,0,0,0] for character 'A'. For character 'T', the vector [0,1,0,0] and so on.

Splitting this data into training and test chunks is your responsibility. However, some of the data must be saved within chunks. Think about chunk dependencies you created in your model that differ from real-world scenarios. **It is not obligatory to use balanced splitting to avoid data bias.** Although it is necessary in real-world scenarios due to the restricted time, you do not have to apply; discussing this matter in the report is sufficient. **Using cross-validation or nested cross-validation (some sort of cross-validation) is obligatory.** Nevertheless, please do not use it because I said it without understanding. Show your knowledge and understanding in your report.

Remember that one-hot encoding, unbalanced splitting within classes, and choosing base architectures for the model could drastically affect your results negatively. You should change these in real-world scenarios; you will

discuss real-world proper methods in your report only for this assignment to make it simpler.

In Figure 3, you may find an example of older statistics about the given HGNC data. As you can see, there are **four classes** of **Locus Groups** within the data, which are **(i) protein-coding gene, (ii) non-coding RNA, (iii) pseudogene, and (iv) other**. For simplicity, we will not investigate through other Locus Type details.





















































































Statistics					
Locus Group	Total by Locus Group		Locus Type	Total by Locus Type	
protein-coding gene	19259	  	gene with protein product	19259	  
non-coding RNA	9101	  	RNA, Y	4	  
			RNA, cluster	119	  
			RNA, long non-coding	5764	  
			RNA, micro	1912	  
			RNA, misc	29	  
			RNA, ribosomal	60	  
			RNA, small nuclear	50	  
			RNA, small nucleolar	568	  
			RNA, transfer	591	  
			RNA, vault	4	  
pseudogene	14453	  	T cell receptor pseudogene	37	  
			immunoglobulin pseudogene	203	  
			pseudogene	14213	  
other	982	  	T cell receptor gene	200	  
			complex locus constituent	69	  
			endogenous retrovirus	109	  
			fragile site	116	  
			immunoglobulin gene	229	  
			readthrough	147	  
			region	38	  
			unknown	66	  
			virus integration site	8	  
Total Approved Symbols				43795	  

Figure 3. Statistics of HGNC data.

You may notice some of the data for RefSeq Accession is absent. If you can not find any accession, you will skip that gene for your recommendation system (model) and not produce an output column for the gene (corresponds to an empty string in the table).

Similarly, a sequence cannot be found for some of the given RefSeq Accession values in the metadata. You will skip those genes from the model, too.

Your model will output the "HGNC_outputs.txt", which will have **HGNC:ID** and **Prediction** column as your model's output, **do not change the original order and count**. Your prediction values are **(i) protein-coding gene, (ii) non-coding RNA, (iii) pseudogene, (iv) other, and (v) an empty string to indicate a NAN value**.

The Report

The report "bXXXXXXXX-Report.pdf" has to consist of the following sections. Each section and your overall report should have the ideal size you prefer. **Do not make it too short or too long please**. Make it a manageable length. **I am giving you feedback about your AI reports, they were primarily short reports with drastic deductions**.

- ❖ **Cover:** Your name, student number, course name, assignment name, and the delivery date.
- ❖ **Introduction:** Define the project in your own words. What is the problem description? Project goals? Definitions. Topics' narration. **A page: 250-300 words**.
- ❖ **Data:** What are your materials? Define the data, its columns, units, statistics, features, and meanings.

❖ **Method:** What are your techniques for solving the problem? Which type of RNN model have you selected? What are your architectural details? Why? Explain your method.

❖ **Development**

❖ **Plan:** What is the project plan? Your requirements. (For example, your hardware requirements might affect the number of selected hyperparameters)

❖ **Analysis:** Analyze the features of data. Which features have you selected? What are the technically related analyzed features of your problem? Why?

❖ **Design:** Which hyperparameters have you selected to tune? Have you used cross-validation or nested cross-validation? Why? Everything in the Method section will be created, analyzed, and explained. Why is your design thus and so?

❖ **Implementation:** The flow of the implementation. Explain your code parts, such as functions, selected methods, and Jupyter blocks.

❖ **Programmer Catalog:** Write down the time you spend on analysis, design, implementation, testing, and reporting. *Write down the training and testing time.* How can a programmer reuse your code for other purposes / other data? Is it reusable for a programmer? Why? Prepare the programmer manual of the program (should use unit/function explanations and usage examples) if your program is reusable. You may use the ".py" version of your project here.

❖ **User Catalog:** How can one reuse your code for other purposes / other data? Is it reusable for a regular non-programmer person? Why? Prepare the program's

user manual (you might use screenshots) if your program is reusable. State the restrictions of your project. You may use the ".py" version of your project here.

❖ **Results:** Your results. Which metrics (obligatory balanced metrics are F1 and MCC) have you used? Why? Visual statements are preferable. Performance evaluation, conclusion, comments, insights, and future work might be listed. What has RNN brought us? Is the nature of the problem compatible with the nature of the RNN? Evaluate your results.

❖ **References:** Resources you have used to prepare the project and the report. Use numbers that are indicated in the text. Realize the possible academic integrity issues.

❖ Please be aware that your points will be deducted if you miss some of the parts mentioned here or skip them too quickly without a clear sight in your report.

❖ Please be aware that the explanation of each part given here is an example to clarify its purpose so that you can understand and prepare it. Do not get too stuck to the examples here.

❖ Including at least one or, ideally, more **flowcharts** of your algorithm or project flow is **obligatory**.

The Code

The code part should consist of several sections and have to be a "bXXXXXXXXX-Supervised.ipynb" notebook file:

- ❖ Please do not forget to write your name and student number at the beginning of your code.
- ❖ Using **random seed 42** for all your random/normal operations, such as splitting, training, and testing, is **obligatory throughout your project**. Random seeding is necessary for the applicability and grading of your project. Please use random state variables wherever possible (i.e., in your RNN and splitters of sci-kit-learn).
- ❖ **Visual demonstrations are necessary and obligatory**. There is no single answer or "correct way to do it". The only correct way is for you to be able to reason what you are doing.
- ❖ You will select your architecture based on **the basic RNN architecture** space. Other complex RNN architecture space examples include traditional RNN, FRNN, GRU/LSTM, BRNN, DRNN, and other variants of RNN. Remember that these complex and deeper models could require more time than the due date for this assignment. Reference your model and design properly from your sources. Selecting a basic RNN architecture with a shallow structure is crucial to minimize the cost and fit the deadline. It could lead to insufficient results. It is fine; it is a balance between time and success. Discuss these in your report.
- ❖ Using the **matplotlib** library for your visual project parts, such as data pre-processing and evaluation of your models, is **recommended**.
- ❖ It is **obligatory to use some cross-validation** while training. Explain your cross-validation and why it is that way. Watch out, for instance, for an imbalance of classes in the data.
- ❖ You should tune an acceptable amount of parameters. Do not go for a huge parameter set; be

reasonable with your problem scale and data while preserving your time due to training properties. Staying shallow in learning during training is acceptable. You must discuss deep architectures' (non-)applicability and (dis-)advantages in the report.

- ❖ Your **evaluation metrics** have to include the selected **balanced metrics: F1-Score and MCC**. You can have other metrics.

- ❖ Lastly, you will convert your ".ipynb" file onto a ".py" file, or you will have a **split** among them that has **fundamental modules inside your ".py"** file. Do not forget to handle your design in your report.

CS Submit System

Finally, submit your **b2XXXXXXX.zip** folder with your data, notebook, code, and report inside (see Figure 4) to the [Submit](#) web page once completed (see Figure 5). These four files and their naming are obligatory to have.

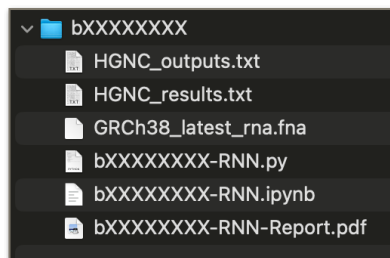


Figure 4. Showing your overall project hierarchy.

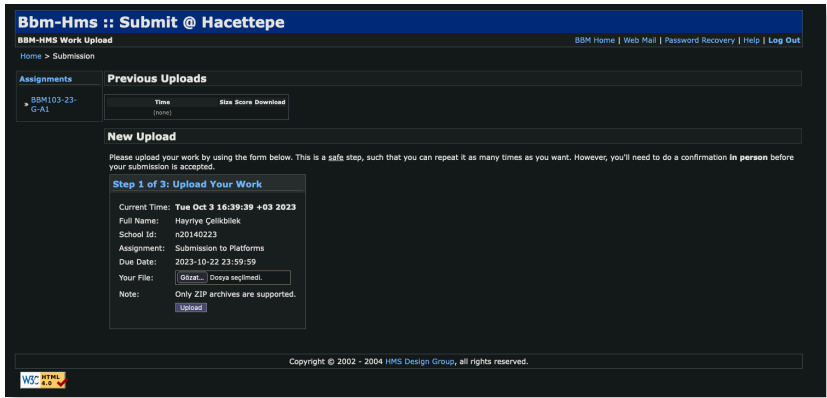


Figure 5. The overall vision of the Submit system.

Grading

In evaluating the assignment, the scoring is as follows:

Evaluation	Points	Evaluate Yourself
Solving 10 lessons from Brilliant	20	
Solving the whole 28 lessons from Brilliant	15 (Bonus)	
Wrong Folder, File names and hierarchy	-100	
Solving Brilliant without enrolling with the classroom link	-100	
No student ID signup naming in Brilliant	-100	
No student ID in code.	-100	
No student ID in report.	-100	
Missing Submit submission	-100	
Missing random seed	-100	

Evaluation	Points	Evaluate Yourself
Lack of visual demonstrations	-30	
Chosing basic metrics to evaluate your model	-40	
Report - Cover	2	
Report - Introduction, Data, Results	5	
Report - Method	8	
Report - Development Plan, Analysis	5	
Report - Development Design, Implementation	10	
Report - Programmer and User Catalogs	10	
Code	40	
Total	115/100	

You may include your evaluation guess as a file in your .zip folder named "**self-evaluation-table.pdf**" (or any other extension). This self-evaluation table is **optional** for feedback about your expectations, self-awareness, and effort.

 **GOOD LUCK
CONQUERERS OF THE
ARTIFICIAL WORLD!**

Appendix

Algorithmic Thinking, Visualization and Flowcharts:

- Python Code Debugger: visualizing code execution: pythontutor.com
- Creating flowcharts from text (pseudocode): chartmage.com/index.html
- Automatically create flowcharts from Python code:
<https://github.com/cdfmlr/pyflowchart>
<http://flowchart.js.org/>
- Creating flowcharts from code-like text: app.code2flow.com
- Creating flowcharts manually: draw.io or app.diagrams.net

Visualizing your mind and code while developing to debug or for reporting purposes is essential!

Formatting and Presentation Checklist for the Report:

- ☑ The target reader is a general high school graduate with a basic understanding of programming.
- ☑ Include a table of contents at the beginning of the report for easy navigation if your report is longer than 5-7 pages.
- ☑ Use a standard font and font size (e.g. Times New Roman, 12 pt) for consistency and readability.
- ☑ To maintain a professional and organised appearance, use a consistent formatting style throughout the report, including margins, line spacing, line breaks, and indentations.
- ☑ Use a clear, easy-to-read layout with sufficient white space to avoid clutter and make the report visually appealing. Aim for a balance between text and white space.

- ☑ Use a consistent and visually appealing style for headings and subheadings, such as bold text, italics, or larger font sizes. Ensure the hierarchy is clear and each section is easily distinguishable from the others.
- ☑ Use a clear and concise writing style, avoiding overly complex or technical language unless necessary. Use plain language wherever possible, and explain any technical terms or concepts unfamiliar to the reader.
- ☑ Break up long paragraphs into shorter ones for easier reading. Aiming for no more than five sentences per paragraph is a good guideline.
- ☑ Use (sub)headings to organize the content and guide the reader. This helps create a logical flow of information and allows the reader to find the information they need quickly.
- ☑ Use bullet points to highlight important information. This makes the information stand out and easier to read and digest.
- ☑ Use diagrams, charts or tables to explain complex concepts whenever possible. Visual aids can help to simplify complex information and make it easier for the reader to understand (such as flowcharts).
- ☑ Use consistent and appropriate terminology and abbreviations throughout the report.
- ☑ Include code snippets to illustrate specific examples or concepts and format them clearly for readability. Use a monospace font (e.g. Courier) and separate code from text using a different background colour, indentation, or border. Also, test and verify the code examples to check whether they work correctly before including them in your documentation.
- ☑ Use proper grammar, spelling and punctuation, and proofread carefully for errors to make your report well-organized. Use a

spell checker and grammar checker, but read the report out loud or have someone else proofread it to catch any mistakes.

<https://www.scribens.com/> or https://www.paperrater.com/free_paper_grader

- ☑ Provide a clear and concise conclusion summarizing the main findings or conclusions.
- ☑ Use appropriate referencing (if needed), following a recognized citation style consistently and accurately.