به نام خدا

رباتیک اجتماعی و شناختی

مدرس : علیرضا طاهری

نیمسال دوم ۰۲-۱۴۰۲

دانشگاه صنعتی شریف
دانشکده‌ی مهندسی مکانیک

تمرین سری اول	" شبکه‌های عصبی مصنوعی"	مهلت ارسال: ۱۷ فروردین ۱۴۰۳

- هم فکری و هم کاری شما در انجام تمارین مانعی ندارد اما پاسخ ارسالی هر فرد در نهایت حتما باید توسط خود او حل و نوشته شده باشد.
- در صورت هم فکری و یا استفاده از منابع خارج درسی، نام هم‌فکران و آدرس منابع مورد استفاده برای حل سوال مورد نظر را ذکر کنید. کمک گرفتن از LLM ها در حل تمارین مجاز است!!
- نتایج و پاسخ های خود را در یک فایل فشرده (ترجیحاً به نام HW1-Name-StudentNumber) در سامانه قرار دهید. پیشنهاد می شود برای بخش نرم افزاری از زبان برنامه نویسی پایتون در یکی از محیط های Jupyter notebook و یا Google Colab برای کدنویسی و تست کدهای خود استفاده نموده و فایل کدها را به فرم IPYNB ارسال کنید.

سوالات تشریحی (۲۵۰ نمره)

۱- با دانش خود و یا بهره گیری از جستجوهای اینترنتی یا سایر منابع، به پرسش‌های زیر پاسخ دهید:

الف) به دلخواه خود، جزییات یکی از روش های بهینه سازی Adam، Adadelta یا Adagrad را توضیح دهید.

ب) به صورت مختصر توضیح دهید در یادگیری ماشین، L2 Regularization چیست و چگونه به ما در فرآیند هموارسازی کمک می کند.

پ) یک شبکه عصبی تا حد امکان ساده پیشنهاد دهید که با گرفتن یک ورودی و همچنین استفاده از توابع فعال سازی ReLu برای نرون های آن، تابع sigmoid را با دقت مناسب در خروجی ایجاد کند. وزن ها و بایاس های شبکه پیشنهادی شما چه مقایری خواهند داشت؟ مشتق در مبدا و مقادیر در کران‌های اشباع تابع شما چقدر است؟ (لزوماً نیازی به کدنویسی نیست و احتمالاً با نوشتن روابط ریاضی به صورت دستی، می توانید شبکه مناسبی را بیابید.)

ت) یکی از مواردی که همواره در عملکرد شبکه های عصبی اهمیت خود را نشان می دهد، نحوه انتخاب پارامتر های اصلی (Hyper Parameters Tuning) است. تحقیق کنید چه روش هایی برای انتخاب بهینه پارامتر های اصلی شبکه وجود دارد (راهنمایی: دو مورد از معروف ترین این روش ها *Mesh Grid Search* و *Random Search* می باشد.)

ث) در مورد روش‌های مقداردهی اولیه شبکه‌های عصبی تحقیق نموده و توضیحات مختصری در مورد کارایی سه روش بالقوه ۱) وزن دهی یکسان به همه نرون ها، ۲) نمونه گیری از توابع گوسی، و ۳) روش زیویر (Xavier initialization) ارائه دهید.

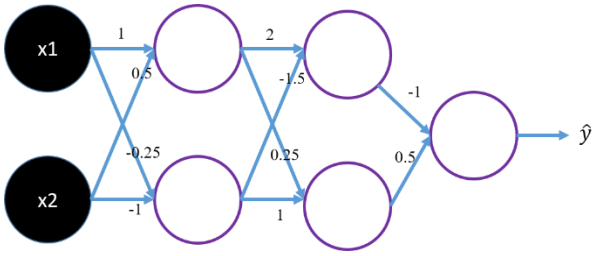
ج) روش k-fold cross validation را توضیح دهید.

سوالات عملی (۷۵۰ نمره)

۱- برای انجام یک طبقه بندی دو کلاسه با داده های آموزش داده شده، می خواهیم از شبکه عصبی ساده سه لایه زیر استفاده کنیم. با درنظر گرفتن تابع فعال سازی tanh برای نرون های لایه های (مخفی) اول و دوم و تابع فعال سازی sigmoid برای نرون لایه سوم (خروجی) و همچنین تابع هزینه میانگین مربعات خطا (MSE)، مطلوبست:

الف) محاسبه عبارت های $\frac{\partial Loss}{\partial w_{11}^{[3]}}$ ، $\frac{\partial Loss}{\partial w_{11}^{[2]}}$ و $\frac{\partial Loss}{\partial b_{1}^{[1]}}$ به صورت پارامتری.

Sample #	x1	x2	y
1	1	0	1
2	-1	0	1
3	0	1	1
4	0	-1	1
5	2	0	0
6	-2	0	0
7	0	2	0
8	0	-2	0
9	0	0	1
10	2	2	0
11	2	-2	0
12	-2	2	0
13	2	-2	0
14	4	0	1
15	-4	0	1
16	0	4	1
17	0	-4	1



ب) هدف از این بخش، آشنا شدن شما با جزییات پیاده سازی روش انتشار به عقب است. بدون بهره گیری از کتابخانه های آماده و موجود شبکه های عصبی و با کدنویسی از ابتدا (در متلب، پایتون یا هر زبان برنامه نویسی دیگر)، شبکه عصبی داده شده را برای ۸ داده اول ارائه شده در جدول آموزش دهید (در این قسمت به داده های ۹ تا ۱۷ کاری نداشته باشید!). با وزن های تصادفی داده شده و با نرخ آموزش $\eta = 0.001$ و بهره گیری از روش SGD، الگوریتم انتشار به عقب (BP) را ابتدا به میزان یک دوره (epoch) اجرا کرده و تمامی وزن ها (و بایاس ها) را به روزرسانی و گزارش کنید (مقادیر تمامی بایاس ها در حالت اول صفر فرض شده است). سپس تعداد دوره ها را افزایش دهید تا دقت شبکه روی داده های آموزش داده شده، حداکثر مقدار خود شود. مقادیر نهایی وزن های شبکه را گزارش نمایید. همچنین هندسه مرزهای تفکیکی دو دسته شبکه خود را در طول دوره آموزش و پس از به روزرسانی وزن ها رسم نمایید (به عنوان نمونه به ازای ۵ ایپاک دلخواه با فاصله های مساوی در کل طول دوره آموزش).

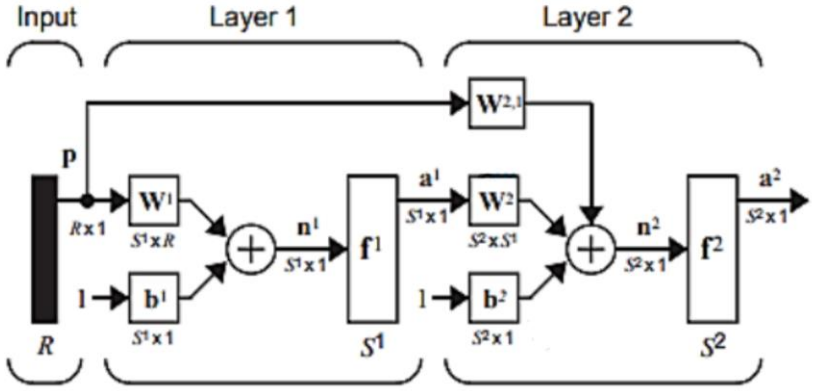
پ) حال با بهره گیری از کتابخانه های موجود شبکه های عصبی نظیر کراس، پایتورچ یا تولباکس شبکه عصبی متلب، برای کل ۱۷ داده آموزشی ارائه شده در جدول، شبکه ای سه لایه را آموزش دهید. در شبکه مذکور، تعداد نرون های لایه مخفی شما بین ۱ تا ۴ متغیر بوده و برای این ۱۶ حالت ممکن، از توابع فعال سازی tanh و ReLu استفاده نمایید. مرزهای نهایی هر یک از شبکه های آموزش دیده شده را برای این مسئله جداسازی رسم نمایید. هدف از این بخش، مشاهده تفاوت تعداد نرون ها در ایجاد مرزهای غیرخطی در شبکه های عصبی است. در صورتی که هیچ یک از شبکه های آموزش دیده نتوانستند دسته بندی شما را انجام دهند، می توانید تعداد نرون های لایه مخفی تان را از ۴ بیشتر نمایید.

۲- شبکه عصبی زیر، یک شبکه پیشرو (feed forward) دو لایه است؛ با این تفاوت که یک ارتباط از ورودی به لایه دوم نیز اضافه شده است! هدف از این تمرین، به دست آوردن تمامی آپدیت‌های الگوریتم عقب گرد برای تمامی پارامترهای شبکه است. در واقع بعد

از انجام این تمرین، انتظار می رود که شما بتوانید مفهوم الگوریتم بازگشت به عقب (BP) را برای هر شبکه با تغییرات جزئی در معماری یا توابع فعال ساز آن پیاده‌سازی کنید.

(راهنمایی: ابتدا فرمول‌های forward pass را به شکل زیر در نظر گرفته و سپس مشتقات $\frac{\partial F}{\partial w_{ij}^{m}}$ ، $\frac{\partial F}{\partial b_{i}^{m}}$ و $\frac{\partial n_i^2}{\partial w_{ij}^{2,1}}$ را برحسب آن‌ها محاسبه کنید. در نهایت هم معادلات آپدیت را به فرم ماتریسی بنویسید.)

$$n^1 = W^1 p + b^1$$
$$a^1 = f^1(n^1) = f^1(W^1 p + b^1)$$
$$n^2 = W^2 a^1 + W^{2,1} p + b^2$$
$$a^2 = f^2(n^2) = f^2(W^2 a^1 + W^{2,1} p + b^2)$$



۳- **مسئله رگرسیون:** یک شبکه عصبی چندلایه پیشنهاد دهید که با آن بتوانیم تابع $y = 2\sin(x) + \sin(2x)$ را بازتولید کنیم (ورودی شبکه: اسکالر x؛ و خروجی شبکه: اسکالر y). خروجی شبکه آموزش دیده تان را در بازه $x\epsilon[0,4\pi]$ رسم نمایید. در نظر داشته باشید که داده های آموزش فقط از مجموعه $x\epsilon\{0,\frac{\pi}{20},\frac{2\pi}{20},\frac{3\pi}{20},\dots,2\pi\}$ (و y های مربوطه به عنوان برچسب) به عنوان ورودی به شبکه داده شوند. آیا شبکه شما در درون یابی و برون یابی تابع مورد نظر، قدرت های یکسانی دارد؟ متریک پیشنهادی برای این گونه مسائل، MAE است.

۴- **تشخیص الفبای لاتین با حرکات دست:** در این سوال قصد داریم که شبکه‌ای برای تشخیص حروف لاتین با توجه به حرکت دست انسان آموزش دهیم. دو فایل CSV برای آموزش و تست در اختیار شما قرار گرفته است. ستون اول در هر فایل نشان‌دهنده‌ی برچسب آن سطر می‌باشد و ۷۸۴ ستون بعدی، نشان دهنده مقادیر پیکسل‌های هر تصویر می‌باشد. برچسب هر کلاس هم به این صورت می‌باشد که عدد ۰ معادل کلاس A، عدد ۱ معادل کلاس B و به همین ترتیب عدد ۲۴ معادل کلاس Y می‌باشد. توجه داشته باشید که تصاویر به صورت تک کاناله (طیف خاکستری) می‌باشد.

الف) ابتدا ۵ داده تصادفی از دیتاست آموزش را بعد از استاندارد سازی در قالب تصویر به همراه کلاس مربوطه به هر تصویر نمایش دهید.

ب) حال داده‌های اعتبارسنجی (validation) را از داده‌های آموزش با نسبت ۱ به ۶ جدا نموده و یک شبکه MLP را آموزش دهید. می‌توانید در آموزش از تابع هدف categorical cross-entropy استفاده نمایید. در انتخاب تعداد لایه و تعداد نرون موجود در هر لایه کاملاً آزاد هستید. استفاده از هر کتابخانه keras و یا pytorch نیز آزاد است. مشخصات و ویژگی های بهترین شبکه ای که در قسمت های قبل آموزش دادید را بیان کنید (حداقل درصد قابل قبول دقتی بین بازه‌ی ۷۵ تا ۸۰ درصد بر روی داده های تست است.)

۵- یکی از قابلیت های مورد انتظار از یک ربات اجتماعی، توانمندی تشخیص حالت چهره افراد (نظیر شادی، غم، خشم و ...) می باشد. در این مسئله، مجموعه عکس هایی مستخرج از پایگاه داده CK+ به عنوان داده های آموزش در اختیار شما قرار می گیرد. شما می بایست **یک شبکه عصبی چندلایه** طراحی کنید که تا حد امکان، طبقه بندی ۸ کلاسه شناسایی حالت های چهره کاربران را به خوبی انجام دهد. عکس ها در ابعاد ۴۹۰×۶۴۰ پیکسل (عمدتاً در محدوده طیف خاکستری) می باشند و در هشت پوشه مختلف قرار داده شده اند؛ به عبارتی، نام پوشه ها، معادل برچسب عکس موجود در آن ها است. همچنین به دلخواه خود، وزن های منتهی به حداقل ۱۰ نرون از لایه های مخفی یا پایانی شبکه تان را در قالب عکس ارائه دهید. آیا وزن های ارائه شده شما، ویژگی قابل فهمی را ارائه می نمایند؟

ما در نهایت کدهای شما را روی مجموعه دادگانی به عنوان داده های تست (که هم اکنون در اختیار شما نیست) نیز پیاده سازی خواهیم کرد تا میزان توانمندی شبکه پیشنهادی شما را در طبقه بندی داده های جدید محک بزنیم.

لینک دانلود داده های آموزش:

https://drive.google.com/file/d/1XL0W5YdJjdE4WVfo9YHjddH4V6CJ6H10/view?usp=sharing

توضیح تکمیلی ۱: در صورتی که عکس رنگی در مجموعه دادگان وجود داشت، با دستورات نمونه زیر از کتابخانه opencv می توانید آن را به طیف خاکستری تبدیل نمایید:

```
import cv2
image= cv2.imread("picture.jpg",0)
```

توضیح تکمیلی ۲: با استفاده از مجموعه دستورات نمونه زیر می توان با مشخص کردن مسیر پوشه مورد نظر، تمامی تصاویر موجود در آن پوشه با فرمت مشخص را خواند و در یک آرایه ذخیره نمود:

```
import cv2
import glob

image_dir = "./image/"
cv_img = []
for img in glob.glob(image_dir+"*.jpg"):
    image= cv2.imread(img)
    cv_img.append(image)
```

البته راه های دیگری نیز در این خصوص وجود دارد که شما می توانید با کمک دستیاران آموزشی درس و یا جستجوهای اینترنتی آن ها را بیابید.

۶- هنگامی که مغز فرمان حرکت عضله ای را می دهد، سیگنالی از مغز به عضله داده می شود که ناشی از آن فعل و انفعالاتی در عضله ایجاد شده و عضله به حرکت در می آید. ناشی از این فعل و انفعالات درون عضله ای، بار های الکتریکی ایجاد می شوند که دارای اطلاعات مربوط به فرامین مغزی هستند. به این سیگنال ها، سیگنال های الکترومایوگرافی گفته می شود. دیتاست زیر حاوی اطلاعات مربوط به سیگنال های الکترومایوگرافی چهار کانال عضله ای پا، برای افراد سالم و همچنین افراد دارای محدودیت می باشد:

https://archive.ics.uci.edu/ml/machine-learning-databases/۰۰۲۷۸/SEMG_DB۱.rar

* پوشه N_TXT حاوی سیگنال های الکترومایوگرافی افراد نرمال و A_TXT حاوی سیگنال های افراد داری محدودیت می باشد. ۴ ستون اول دیتاست، داده های sEMG و ستون پنجم داده زاویه می باشد.

الف-۱) به منظور دسته بندی داده های افراد نرمال و دارای محدودیت (دو کلاس)، شبکه عصبی دلخواهی را طراحی کنید که ۳ لایه پنهان داشته باشد. نمودار هدررفت (loss) و دقت را گزارش کنید. ورودی این شبکه، داده های مربوط به ۴ کانال سیگنال EMG (۴ ستون اول مجموعه دادگان) و خروجی آن نرمال بودن یا نبودن افراد است.

الف-۲) شبکه را با سه تابع فعال ساز ReLU، PReLU و ELU تست کنید و به صورت جداگانه نتایج را گزارش نمایید. کدام یک بهتر عمل کرده اند؟

ب-۱) حال به سراغ مسئله تخمین زاویه زانو (ستون پنجم مجموعه دادگان) می رویم! چهار کانال sEMG را به عنوان ورودی به شبکه داده و با یک شبکه دلخواه (با ۳ لایه پنهان)، زاویه زانو را تخمین بزنید. برای تخمین زاویه نیاز است که ابتدا بر روی دادگان سری زمانی موجود، پنجره (Data Frame) مناسبی استخراج کنید.Data Frame ها را به نحوی استخراج کنید که از گذشته ۲ داده توسط شبکه دیده شود (به عبارتی ۸ داده ورودی شامل ۲ بسته ۴ تایی از سیگنال ها) و در آینده ۱ مرحله تخمین زده شود. نمودار Loss و متریک MAE را برای کل شبکه گزارش کنید.

- پارامتر های شبکه را به کمک یکی از روش های Random Search و یا Mesh Grid تنظیم کرده و دلیل انتخاب خود بین این روش ها را بیان کنید.

ب-۲) در پنجره زنی بخش قبل، مدت زمان را از ۲ به ۱۰ افزایش داده و موارد خواسته شده در بخش قبل را گزارش کنید. نتایج بدست آمده را با بخش ب-۱ مقایسه کنید.

۷- در کلاس درسی، با مفهوم PINN (Physics-Informed Neural Network) آشنا شدید. حال در این تمرین به پیاده سازی آن خواهید پرداخت. شما به عنوان یک مهندس مکانیک، با هر زمینه تحقیقاتی اعم از سیالات، جامدات و کنترل رباتیک، در درجه اول با معادلات دینامیکی سیستم ها مواجه می شوید. این معادلات دیفرانسیل ممکن است از نوع ODE یا PDE باشند. می دانید که بعضی از معادلات PDE ساده دارای حل دقیق هستند. اما اگر این معادلات کمی پیچیده شوند، چه؟!! در این صورت، از روش های حل عددی این معادلات کمک می گیریم. استفاده از شبکه های عصبی (عمیق) نیز یک روش حل عددی برای حل این معادلات به شمار می رود؛ اگرچه روش های حل عددی امروزی هنوز از PINN استفاده نمی کنند. مبحث PINN یک موضوع تحقیقاتی بسیار جدید، حتی در میان موضوعات هوش مصنوعی است و احتمالاً به زودی جایگزین روش های حل عددی مرسوم امروزی خواهد شد. شما قرار است قدم به قدم با علم روز، یک معادله PDE را حل کنید!

در این تمرین یکی از معروف ترین معادلات PDE یعنی معادله موج یک بعدی را برای حل انتخاب کرده ایم. علت این انتخاب این است که می توانید حل PINN را با حل دقیق معادله موج که در دسترس است مقایسه کنید؛ اگرچه قدرت روش PINN در جایی است که از حل دقیق ناتوانیم! برای سادگی کار شما، خودمان حل دقیق آن را با روش جداسازی متغیرها پیاده سازی کرده ایم (فایل با نام wave در پوشه utils). اما حل آن با استفاده از روش PINN به عهده شما خواهد بود. دقت کنید که در نهایت پاسخ شما با پاسخ حل دقیق باید یکسان باشد، وگرنه پیاده سازی شما صحیح نیست. به این منظور پیاده سازی فایل های model، pinn و train در پوشه utils می بایست تماماً توسط شما **کامل** گردد. بقیه فایل ها کامل هستند و نیازی به تکمیل ندارند. در پایان می توانید با اجرای نوتبوک wave_equation_۱d نتایج خود را ملاحظه کنید و ببینید که آیا با حل دقیق نتایج مشابهی گرفته اید یا خیر. به دلیل چالشی بودن

این تمرین، لطفا وقت خود را برای حل آن مدیریت کنید. اگرچه این تمرین کمی چالشی است، اما موضوعی جدید و پرثمر و آینده دار در علوم مهندسی و پایه به خصوص مهندسی مکانیک و فیزیک محض می باشد.

به صورت خلاصه، نیاز به تغییر فایل های wave (فایل مربوط به حل دقیق) و vis (فایل مربوط به توابع سازنده پلات ها برای مقایسه حل دقیق و حل PINN) ندارید. همچنین نیازی به تغییر نوتبوک wave_equation_۱d ندارید. فایل ها را از پوشه utils خارج نکنید. در نهایت نوتبوک wave_equation_۱d به عنوان موتور اصلی باید ران شود و فایل های پایتون در پوشه utils برای ران شدن نیستند. تنها تکمیل کردن فایل های model و pinn و train به عهده شما است که در زیر در رابطه با هر کدام توضیح داده شده است. به این توضیحات اکتفا نکنید؛ در خود نوتبوک نیز توضیحاتی داده شده که می توانید مطالعه کنید و از همه مهمتر، جستجو کردن و مطالعه درباره موضوع PINN را فراموش نکنید. منابع بسیار زیاد و مناسبی در اینترنت موجود است.

فایل pinn: در این فایل باید کلاس PINN و تابع compute_grad را کامل کنید.

برای تکمیل compute_grad می توانید از تابع آماده torch.autograd.grad برای پایتورچ و یا tf.GradientTape برای تنسورفلو استفاده کنید (پیشنهاد می شود که برای این تمرین از پایتورچ استفاده کنید). توجه کنید که مدل PINN برای حل معادله موج u_tt=c^۲ u_xx قرار است که به شما u را خروجی دهد؛ اما دستیابی به u_tt و u_xx از طریق استفاده از تابع compute_grad صورت می گیرد.

و اما در رابطه با کلاس PINN، همانطور که ملاحظه می کنید، سه تابع زیان مختلف داریم که تابع زیان نهایی از مجموع آنها بدست می آید. همانطور که می دانید، هر صورت سوال PDE سه قسمت دارد: خود معادله PDE، شرایط مرزی و شرایط اولیه. مدل پیش بینی کننده PINN قرار است مقادیر x و t را به عنوان ورودی گرفته و یک u خروجی بدهد که هر سه اینها را با هم ارضا کند. به همین جهت، سه تابع زیان خواهیم داشت تا هر کدام مدل را به سمتی ببرد تا این سه مورد را ارضا کند.

برای تابع زیان برآمده از خود معادله PDE، خروجی u را از مدل شبکه عصبی بگیرید و سپس مقادیر u_tt و u_xx را بدست آورید و سپس یک تابع زیان بنویسید که معادله را ارضا کند. برای تابع زیان نیاز به یک مقدار پیش بینی شده و یک مقدار برچسب داریم. حال چالش شما این است که برچسب چه چیزی باشد تا معادله u_tt=c^۲ u_xx با در دست داشتن مقادیر پیش بینی شده u_tt و u_xx ارضا شود.

از سمتی دیگر، شرایط مرزی را دارید. می دانیم که شرایط مرزی به ازای تمامی زمان ها با شرایط مرزی خاص داده شده، باید در معادله صدق کند. مقادیر داده شده برای شرایط مرزی به عنوان برچسب برای مقدار u استفاده می شود! حال همین شرایط را (تمامی زمان ها با شرایط مرزی خاص) به عنوان ورودی به مدل بدهید و خروجی آن را بگیرید. تابع زیان میان این دو تعریف می شود.

همچنین، شرایط اولیه را دارید که به ازای تمامی مکان ها و زمان صفر در معادله صدق می کند. روند شرایط مرزی در این جا نیز صادق است. برچسب توسط صورت سوال به عنوان شرایط مرزی داده شده و مقدار پیش بینی شده توسط مدل با ورودی تمامی مکان ها و زمان صفر بدست خواهد آمد. تابع زیان مربوطه نیز از مقایسه آنها تعریف می شود.

مجموع این سه زیان، زیان کل را خواهد ساخت که آن را در تابع physics_loss بدست آورید. همچنین تابع data_loss نیز وجود دارد. این زیان که به عنوان زیان چهارم به کار می رود، زمانی است که شما جدا از معادله، شرایط مرزی و شرایط اولیه، یک سری داده دارید که مثلا از طریق تجربی و آزمایش بدست آمده اند (در اینجا مکان x، زمان t و مقدار u). می توانید برای این داده ها نیز یک تابع زیان بنویسید (ورودی های مکان x، زمان t را دارید که توسط مدل u را پیش بینی می کند و برچسب را نیز خود داده به عنوان مقدار u در اختیار شما گذاشته است). از آنجا که داده ای جداگانه برای حل معادله در اختیار نداریم، در این مسئله (خوشبختانه!!) به این تابع زیان کاری نداریم. اما کماکان نوشتن کد تابع زیان آن لازم است.

در تابع make_collocation باید ورودی مناسب برای سه تابع زیان را بسازید. برای ایجاد داده، می توانید از مقادیر تصادفی در یک بازه مکانی و زمانی، یا مقادیر با گام ثابت در یک بازه مکانی و زمانی استفاده کنید. و سپس آنها را به ابعاد مناسب reshape کرده و به صورت یک دیکشنری خروجی بدهید.

در تابع forward باید به صورت واضح روند محاسبه مقدار u توسط مدل بدست آید (یک گام رو به جلو که مکان x، زمان t را ورودی می گیرد و u را خروجی می دهد). حال مدل از کجا می آید؟! مدل از کلاس make_fc_model که ایمپورت شده است، فراخوانی می شود (که این کلاس در فایل model قرار دارد و توسط شما نوشته می شود) و باید در قسمت init کلاس PINN مقداردهی اولیه شود. همچنین ضریب c و بازه های زمانی و مکانی و تعداد داده ها را نیز می توانید از کلاس Wave_۱D_Equation که ایمپورت شده است فراخوانی کنید (که این کلاس در فایل wave شامل حل دقیق موجود است و این به منظور یکسان کردن متغیرها برای هر دو حل است).

فایل model: در این فایل باید تابع make_fc_model را تکمیل کنید. این مدل یک شبکه عصبی MLP خواهد بود. برای پیاده سازی مدل از توابع آماده استفاده کنید. تابع فعال ساز به دلخواه خودتان است. شبکه بیش از سه چهار لایه نیاز نیست.

فایل train: در این فایل باید حلقه آموزش مدل را بنویسید. ابتدا تابع زیان را از physics_loss در فایل pinn فراخوانی کنید. این کار را در test_pinn انجام دهید که اگر صرفاً خواستید در آینده مدل را تست کنید، از همین تابع استفاده کنید. سپس از مقادیر زیان بدست آمده از test_pinn در تابع train_pinn استفاده کنید و حلقه آموزش را کامل کنید. می توانید از توابع آماده مانند loss.backward استفاده کنید. ضمناً آموزش مدل بسیار کوتاه خواهد بود و وقت گیر نیست.

فراموش نکنید که در نوتبوک wave_equation_۱d باید آدرس جایی که در آن فایل utils قرار دارد را بدهید (ولی نیاز به تکمیل هیچ چیز دیگری را در این نوتبوک ندارید.). مثلا فایل utils در ‘content/my_notebooks/’ قرار دارد. نحوه نوشتن آن:

path = ‘content/my_notebooks/’

لینک مستقیم دانلود فایل های مربوط به این سوال:

https://drive.google.com/file/d/18i6L4fsp20CQa03_H82UUMdATo5j7ErL/view?usp=sharing

نکات مورد توجه در زمینه ارائه فایل های سوالات عملی:

- ترجیحاً صفحات html از صفحه گوگل کولب یا جوپیتر که هم دارای کدها و هم پاسخ ها در زیر سلول های مربوطه می باشند را برای ما ذخیره و ارسال نمایید.
- پیشنهاد می شود بهترین مدل خود در هر سوال عملی را با فرمت h5. ذخیره کنید تا در صورت نیاز ما در آینده به آن ها، دسترسی ها مقدور باشد.
- در بخش عملی الزامی به ارائه گزارش نبوده و تنها کافی است توضیحات و فرضیات مورد نظر و همچنین بحث روی نتایج حاصله، به صورت مختصر و قابل فهم در کد پایتون نوشته شود.