بـــه نـــام خـــدا

دانشکده مهندسی کامپیوتر

هوش مصنوعی و سیستم های خبره

پروژه سوم (برنامه نویسی ژنتیک)

دکتر آرش عبدی

پاییز ۱۴۰۲

طراحان : محمد حسین میرزائی – محمدعلی آژینی

- در صورت وجود هر گونه ابهام در سوالات تنها به طراح آن سوال پیام دهید.
- با توجه به تنظیم شدن ددلاین تمارین توسط خود شما امکان تمدید وجود ندارد.
- داک پروژه را واضح و مرتب بنویسید .
- مواردی که بعد از تاریخ فوق ارسال شوند قابل قبول نبوده و نمره ای نخواهد داشت.
- انجام تمرین تک نفره است. لطفا به تنهایی انجام شود، در غیر اینصورت نمره منفی در نظر گرفته خواهد شد.
- زبان برنامه نویسی دلخواه است. (پیشنهاد: پایتون)
- موارد ارسال شده در تاریخی که بعدا مشخص میشود به صورت آنلاین نیز تحویل گرفته خواهند شد (صرفا آنچه در کوئرا تحویل داده شده است بعدا به صورت آنلاین تست شده و توضیح داده می شود.)
- کل محتوای ارسالی را داخل فایل زیپ قرار داده و نام آن را شماره دانشجویی قرار دهید.

آیدی تلگرام طراحان :

@mim0_o
@iAmMafhoot

تقریب تابع با کمک برنامه نویسی ژنتیک(GP)

یکی از مسائل مطرح در علوم ریاضی و مهندسی تقریب تابع است. تقریب تابع یعنی با داشتن توانایی محاسبه مقدار یک تابع نامعلوم در نقاط دلخواه، ضابطه ریاضی تابع مذکور تقریب زده شود. در این پروژه برای تقریب تابع از برنامه نویسی ژنتیک (GP- Programming Genetic) استفاده می شود.

هدف پروژه: ارزیابی میزان تسلط به مفاهیم پایه الگوریتم های تکاملی (در اینجا برنامه نویسی ژنتیک)

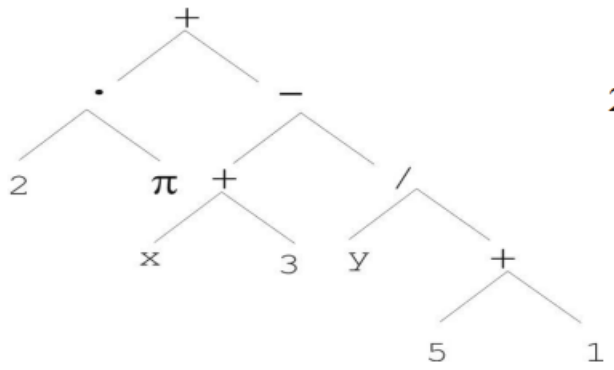
یکی از کاربرد های GP ، تقریب توابع است. در این کاربرد، هر فرد از جمعیت بیانگر یک عبارت ریاضی است که به شکل یک درخت نمایش داده میشود. عناصر میانی این درخت عملگرهای ریاضی، منطقی و یا هر گونه عملگر تعریف شده کاربر و برگ های درخت حاوی مقادیر ثابت و متغیرهای ورودی هستند. هدف از این کاربرد این است که فرمول ریاضیاتی یک تابع هدف را تخمین بزنیم. تابع هدف یک Black Box است و تنها چیزی که از آن میدانیم (یا قادریم بدست آوریم)، بدست آوردن مقدار تابع در نقاط دلخواه است. بدین ترتیب در تعدادی نقطه دلخواه مقدار تابع را یافته و از آنها به عنوان مجموعه آموزشی یاد میکنیم. برای هر درخت بر حسب مقادیری که برای نقاط آموزشی بدست می آورد و میزان اختلافش با مقادیر واقعی میتوان یک مقدار شایستگی نسبت داد و بر اساس آن الگوریتم تکاملی را دنبال کرد.

عملگرهایی که در درختان به کار برده میشود حداقل شامل موارد زیر باشد (موارد بیشتری نیز میتوانید اضافه کنید) :

عملگرهای دو عمل وندی شامل: + ، − ، × ، / ، ^ (توان)

عملگرهای تک عمل وندی شامل : Sin(x) , Cos(x)

یک نمونه از درخت:



$$2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$$

ورودی برنامه:

تعدادی نقطه آموزشی و مقدار خروجی تابع هدف در آن نقاط (برای سادگی، تابع یک بعدی در نظر گرفته میشود).

تبصره 1 :

در عمل برای تولید ورودی و ارزیابی عملکرد الگوریتم میتوان توابعی دلخواه (شامل هر عملگری که در الگوریتم پیشی بینی شده است و حتی عملگرهایی که در الگوریتم پیش بینی نشده است) در نظر گرفت و در تعدادی نقطه دلخواه مقدار تابع را محاسبه کرد و این نقاط را به عنوان ورودی الگوریتم در نظر گرفت. حتما در آزمایش ها حالتی را نیز آزمایش کنید که در آن ورودی از روی تابعی تولید شده باشد که دارای عملگری است که در فهرست عملگرهای استفاده شده در GP وجود ندارد. مثال در ضابطه تابع استفاده شده برای تولید نقاط آزمایشی از عملگر لگاریتم یا تانژانت (یا ...) استفاده کنید ولی در GP این عمگرها مجاز نباشند. یا عملگر توان را در GP حذف کنید و نقاط آموزشی را با تابعی که این عملگرها را دارند تولید کنید.

تبصره 2 :

در یکی از آزمایشها در تابع نقطه یا نقاط گسستگی ایجاد کنید و تفاوت رفتار GP را بررسی کنید. مثال متوانید برای نقاط کوچکتر از عدد c1 از ضابطه y = x^2 و در نقاط بین c1 تا c2 از ضابطه y = -x-7 و در نقاط بزرگتر از c2 از تابع دیگری برای تولید نقاط آموزشی استفاده کنید.

تبصره 3 :

در همه موارد، از مثالهای خیلی ساده شروع کنید و بعد از اطمینان از سلامت کد، و سلامت نحوه آزمایش، در حد امکان به سمت مثالهای خفن تر(!!) بروید.

تبصره 4 :

علاوه بر آزمایش روی توابعی با ورودی یک بعدی (ورودی فقط x) ، توابعی با ورودی بیش از یک بعد (حداقل 2 بعد) را نیز آزمایش کنید. در مورد توابع دو بعدی (یا با ورودی بیش از 2 بعد) صرفا توابع ساده را آزمایش کنید و نیازی به انجام نکات مذکور در تبصره 1 و 2 نیست.

تبصره 5 :

در هر نسل تاثیر جهش را بررسی کنید، حتی اگر توانستید یک الگوریتم پویا برای آن بنویسید، سعی کنید رفتار الگوریتم خود را به نحوی برنامه نویسی کنید تا بتواند بستگی به شرایط (برای مثال ثابت ماندن MSE) بتواند جهش جدیدی انجام داده و خود را بهبود دهد.

تبصره 6 :

از روند پیشرفت الگوریتم خود (برای مثال اختلاف بهترین تابع تولید شده در هر نسل و همچنین MSE کل توابع ساخته شده در آن نسل) با استفاده از کتاب خانه Plot نمودار تهیه کنید.

تبصره 7 :

سعی کنید در پایان هر نسل و در پایان اجرای الگوریتم، بهترین تابع تشخیص داده را نشان دهید و در صورت امکان فرمول ریاضی آن را ساده کنید و سعی کنید به تابع اولیه نزدیک تر کنید (اینجا بستگی به خلاقیت خودتان دارد)

تبصره 8 :

تلاش شما نمره اصلی را خواهد داشت و حتی اگر نتایج دلچسب نباشند، امکان گرفتن نمره کامل در صورت صرف وقت و یادگرفتن مشکلات راه، وجود دارد. همچنین در صورتی که نتایج کاملی از همه بخش ها به دست آورید، ممکن است باعث نمره مثبت (نمره اضافی) هم بشود.

خروجی برنامه:

نمایش فرمول متناظر با بهترین درخت تولید شده توسط الگوریتم، مقدار شایستگی بهترین درخت، تعداد نسلها، تعداد محاسبه شایستگی، زمان اجرا، نمودار پیشرفت

آنچه تحویل داده میشود :

1- کد اجرایی برنامه به همراه تست کردن چند ورودی متفاوت توسط دانشجو و نیز امکان تست کردن ورودی های دیگر در زمان تحویل برنامه .

2- گزارش از جزئیات اجرای پروژه که حداقل موارد زیر را در برمیگیرد (بدون ترتیب) :

a. نحوه دقیق نگاشت مسئله به درخت و محدودیت های احتمالی
   i. عملگرها، اعداد ثابت، و...
b. تابع شایستگی
c. تولید جمعیت اولیه
d. نحوه انتخاب والدین
e. نحوه تولید نسل بعد
   i. نحوه ترکیب متقاطع و جهش
f. شرط خاتمه الگوریتم
g. چالش های مواجه شده و روش حل آنها
h. گزارشی از آزمایش های مختلف انجام شده (حداقل 3 آزمایش برای 3 تابع مختلف ورودی) و ارزیابی و تحلیل عملکرد الگوریتم در آزمایش ها
i. جمع بندی و نتایج عملی که درباره GP در این پروژه به دست آوردید.