

سوال 1 (

الف)

$10 \times 10 \times 3$ نورون ورودی داریم یعنی 300 نورون که به طور fully connected به 2 نورون از لایه بعدی وصل شده اند یعنی 600 وزن و 1 بایاس به ازای هر نورون از لایه بعد که یعنی 602 متغیر پارامتر آموزش

ب)

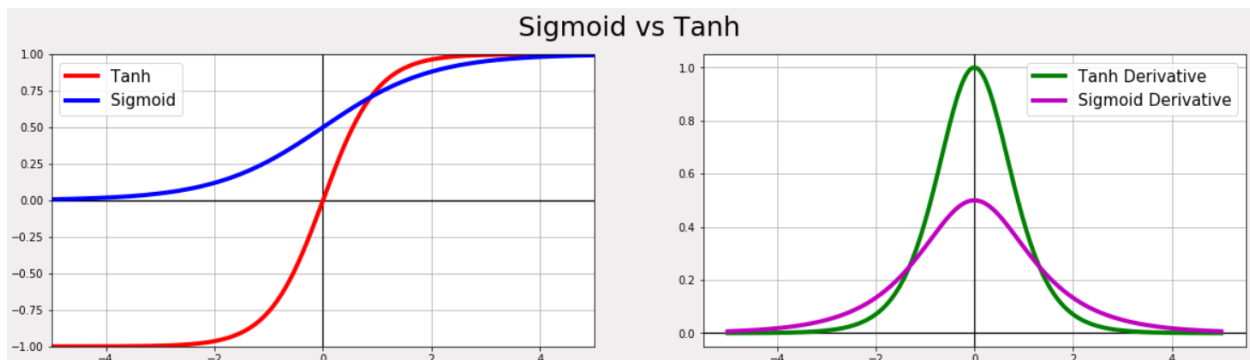
درون هر فیلتر 4 پارامتر. همچنین 3 فیلتر در لایه قبل که یعنی 12 پارامتر + 1 (یک بایاس) در هر چنل وجود دارد که یعنی در کل 39 پارامتر در این شبکه قابل آموزش هستند

پ)

چون پدینگ داریم و استراید 2 است هر 4 پیکسل ورودی را 1 پیکسل مپ میکند که یعنی سائز 10 در 10 در 3 را به 5 در 5 در 3 میرد

سوال 2 (

الف) درست. با توجه به مشتق توابع tanh و sigmoid میتوان فهمید



از آنجایی که موقع برگشتن گرادیان این مقدار در مشتق تابع فعال ساز ضرب میشود هر چقدر که تابع فعال ساز مقادیر را بیشتر به سمت 0 میل دهد امکان رخداد محو شدن گرادیان بیشتر است که میبینیم در عکس این ویژگی در sigmoid غلیظ تر است و ترجیح بر استفاده از tanh است

ب) غلط. زمان رخ دادن محو شدن گرادیان برگشتی مقدار تغییر و بهبودی که در لایه های اولیه رخ میدهد بسیار کمتر از لایه های آخر است

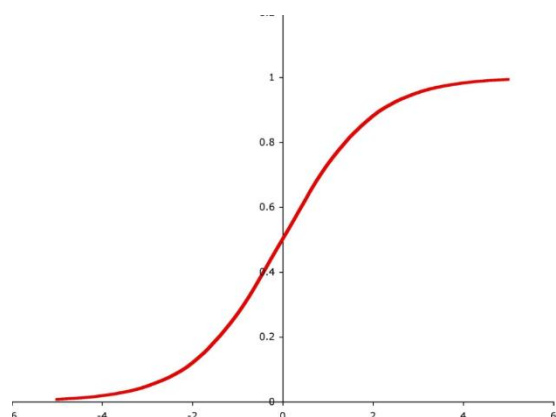
ت) درست. استفاده از فعال ساز relu به خاطر تابع مشتق اش که برای مقادیر بزرگتر از 0 همیشه 1 است و میتوان نتیجه گرفت برای حل مشکل محو شدگی گرادیان از tanh و sigmoid بهتر عمل میکند و مناسب است

ث) درست. برای پیشرفت آموزش شبکه و جلوگیری از محو شدن گرادیان آن واریانس وزن ها بهتر است بزرگتر از 0 باشد

سوال 3)

وقتی که فرایند آموزش کامل شده است و در حالت بهینه ای قرار داریم

اگر وزن ها را همگی نصف کنیم با توجه به تابع sigmoid که تصویر آن را در زیر میبینیم



با نصف کردن وزن ها خروجی هر نورون را به سمت 0.5 میریم
با این کار فرض کنید شبکه ای برای دسته بندی و classify کردن داریم

در یک شبکه آموزش داده شده برای هر ورودی بصورت احتمالی جواب های مانند زیر در لایه خروجی داریم

0.01, 0.06, 0.89, 0.04 با نصف کردن همه وزن ها این احتمالات بدست آمده را به سمت 0.5 میریم و این کار از قطعیت تصمیم گرفته شده میکاهد

که خود باعث کاهش دقت در تصمیم گیری میشود

قابل ذکر است که ما ترجیح بر این داریم که نورون های خروجی پراکندگی داشته باشند و این کار پراکندگی را کم میکند

البته خود این نکته قابل توجه است که در هر صورت زمانی که آموزش تمام شده است ما به یک حالت نسبتا بهینه از وزن ها رسیده ایم و تغییر کور کورانه ای مانند این صورت سوال تقریبا همیشه نتیجه منفی به بار میآورد

سوال 4) خیر. کلاست کردن همه وزن ها با یک عدد ثابت کار مناسبی نیست

چرا که احتمال این وجود دارد که فرایند یادگیری نتواند پیشرفت کند
میتوان فرض کرد با این کار عملاً داریم عملکرد نورون ها را یکسان در نظر میگیریم و هیچ وزنی در فرایند آموزش تغییر نمیکند (اگر دقیقاً همه وزن ها برابر باشند) که باعث میشود پیرفتی نداشته باشیم

(سوال 5)

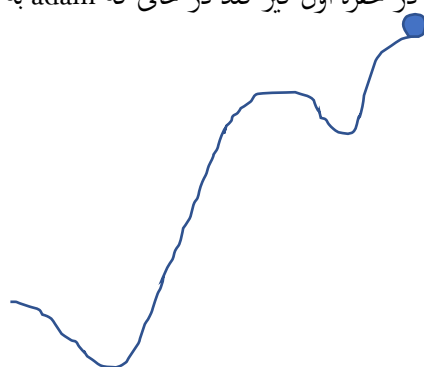
در استفاده از SGD ما هر بار یک ورودی را به نتورک داده سپس گرادیان را محاسبه کرده و خطا را پس انتشار میدهیم
در این حالت رسیدن به مینیمم ارور تضمین شده نیست (در mini batch gradient decent هم تضمین شده نیست اما جنرالیته در آن بیشتر است)
در حالی که در mini batch gradient decent یک مجموعه ای از داده ها به شبکه داده میانگین گرادیان آن ها را پس انتشار میدهیم
در اصل SGD همان mini batch gradient decent است که سایز بچ ها 1 در نظر گرفته شده است
از مشکلات این سایز اور فیت شدن و زمان طولانی آموزش است
وقتی سایز بچ ها را زیاد کرده باشیم از طرفی نیز میتوانیم پردازش موازی داشته باشیم که با بچ های 1 نمیتوانیم

(سوال 6)

Adam میتواند با استفاده از پارامتری به اسم momentum نرخ یادگیری را کنترل کنید این روش به ما اجازه میدهد که سرعت رسیدن به جواب را افزایش دهیم
برای مثال زمانی که نرخ یادگیری بیش از حد بالایی داریم که باعث پرش از جواب میشود آن را کاهش و زمانی که نرخ یادگیری بسیار کم داریم که باعث سرعت گرایش به جواب بسیار پایین میشود نرخ یادگیری را افزایش دهد

نکته مهمی که در مورد استفاده از momentum وجود دارد این است که امکان گیر کردن در local minimum در آن بسیار کمتر است

شکل زیر همان گفته بالا را نشان میدهد. Sgd ممکن است در حفره اول گیر کند در حالی که adam به global minimum میرسد



سوال 7)

یک سناریو که میتواند باعث شکست نسبی شبکه در روند یادگیری شود این است که داده ها را بدون بر زدن و بصورت concat شده به شبکه دهیم به این شکل که نصف اول داده ها از یک کلاس و نصف دوم از کلاس دیگر باشند

در این حالت اگر تعداد داده های ما به اندازه کافی زیاد باشد باعث میشود به مرور زمان در روند یادگیری شبکه به سمت اور فیت شدن روی داده های کلاس دوم پیش برود و خاصیت جنرال بودن خود را از دست بدهد


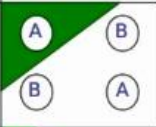
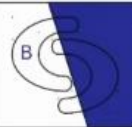

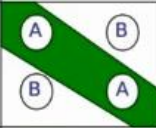


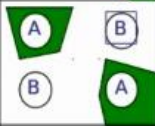

یا مثلا اگر به ترتیب زوج ها و فرد ها از یک کلاس ثابت باشند باز هم به نوعی جنرال بودن شبکه را از دست خواهیم داد

بطورکلی نباید در ترتیب قرار گیری کلاس ها قاعده ای وجود داشته باشد و باید بطور بر خورده باشند

سوال 8)

زمانی که توابع فعالیت را خطی در نظر بگیریم هر چقدر هم تعداد لایه های شبکه را زیاد کنیم و همه فعالیت هایمان خطی باشند در نهایت میتوان همه لایه ها را به نوعی در یک معادله خطی بازنویسی کرد این به این معناست که در نهایت هرچقدر هم که شبکه ما پیچیده شود تنها قدرت یک شبکه یک لایه را خواهد داشت و نه بیشتر با عکس زیر میتوان بهتر توضیح داد

اگر توابع فعالیت خطی باشند عملا تمام کلاس بندی ها مانند حالت اول میشود

Structure	Types of Decision Regions	Exclusive-OR Problem	Classes with Meshed regions
Single-Layer 	Half Plane Bounded By Hyper plane		
Two-Layer 	Convex Open Or Closed Regions		
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)		

(سوال 9)

cnn ها برای این طراحی میشوند که بتوانند از عکس ها ویژگی ها تصویری پیچیده استخراج کنند و طی چند لایه convolution پشت هم ویژگی های استخراج شده غنی تر و پیشرفته تر میشوند به نوعی میتوان گفت ساختار عکسی و ابعاد فضا عکس ها بهتر در cnn حفظ میشود و باعث استخراج بهینه تر و بهتر ویژگی ها در عکس میشود اما در شبکه های عادی با تبدیل عکس به یک وکتور اطلاعات و الگو های مهمی در عکس را از دست میدهم یکی از این الگو ها ترتیب کنار هم قرار گرفتن پیکس ها است که توی شبکه های عادی معنی خود را از دست میدهند یا الگو های پیچیده تری مانند خطوط محدب و مقعر سائز بالای عکس ها میتواند باعث اور فیت شدن در شبکه های عادی هم بشود که دیگر معایب آن است

(سوال 10)

: Convolution layers

در این لایه ها یک فیلتر با سائز مشخصی با روی قسمت های ورودی اعمال میشود خروجی این فیلتر که یک ماتریس است مشخص میکند که آیا الگوی مشخص شده در فیلتر در این قسمت از ورودی وجود داشته است یا نه (خروجی یک عدد است)

در روند آموزش این فیلتر تغییر میکند و در اصل با آموزش این فیلتر است که به شبکه یاد میدهم چه ویژگی هایی را استخراج کند

: Pooling layers

در این لایه ها کار فشرده سازی و چکیده سازی اطلاعات لایه های قبلی (عموما کانولوشن) رخ میدهد این لایه میتواند از روش های مختلفی برای اینکار استفاده کند مانند max pooling یا mean pooling

: Flatten Layer

این لایه تنها کاری که انجام میدهد این است که لایه convolution یا pooling قبلی را گرفته و تبدیل به یک وکتور تک بعدی میکند

هدف از این کار آماده سازی فیچر های استخراج شده برای ورودی دادن به ann است

روند پیشرو :

در این روند ابتدا ورودی عکس که میتواند یک یا چند چنل داشته باشد به لایه کانولوشن اول وارد میشود حال فیلتری از ساز مشخص مثلا 3 در 3 در تمام قسمت های 3 در 3 عکس ورودی اعمال میشود و یک خروجی از هر بار این عمل بیرون میاید (میتوان پدینگ داشت یا نداشت. مزیت داشتن پدینگ این است که به مرور زمان توجه شبکه به مرکز عکس جمع نخواهد شد و ویژگی ها هنوز هم در اطراف معنی خود را

حفظ میکنند. برای پدینگ میتوان یک ردیف 0 یا هر عددی دیگر را (یا حتی میانگین یا درونیایی) به اطراف عکس اضافه کرد

سپس یک لایه پولینگ اطلاعات خروجی لایه قبل را با گرفتن ماکسیمم یا میانگین چکیده میکند و خروجی با ابعاد کوچک تر برای راحت تر شدن روند آموزش و همچنین غنی تر شدن اطلاعات ساخته میشود

این روند میتواند چند مرتبه تکرار شود و هر بار لایه ها ویژگی های پیشرفته تر و غنی تری تولید میکند (بعلاوه تقریباً همیشه تعداد چنل هایی که ویژگی ها را درون خود نگه میدارند افزایش پیدا میکند چرا که با پیچیده تر شدن ویژگی ها تنوع آن ها نیز بیشتر میشود)

در مرحله آخر توسط یک لایه تک بعدی ساز تمام ویژگی ها را به یک فضای تک بعدی میرد (کانکت میشوند) تا آماده کلاس بندی شدن توسط یک شبکه عصبی تمام وصل شده شوند

روند پس انتشار خطا : در این روند خطایی که با گرادینان به عقب برمیگردد فقط روی فیلتر های درون لایه های کانولوشن اعمال میشود (پس از گذر از شبکه عصبی عادی) و مقادیر این فیلتر ها را تغییر میدهد تا ویژگی های استخراج شده بهتر و غنی تر شوند

$$\begin{bmatrix} O_{11} & O_{12} \\ O_{21} & O_{22} \end{bmatrix} = \text{Correlation} \left(\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix}, \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \right)$$

$$O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$

$$O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$

$$O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$

$$O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

(سوال 11)

(الف)

1- استفاده از زوم کردن در یک رنج معقول (اگر ضریب به درون زوم کردن زیاد باشد ممکن است ویژگی های کلیدی مانند خطوط عموی گردن زرافه را از دست داد و اگر ضریب به بیرون زوم کردن

زیاد باشد ممکن است بافت و اشکال روی پوست زرافه آنقدر کوچک شوند که قابل استخراج نباشند

2- استفاده از چرخش 180 درجه ای عکس ها (اگر 90 یا 270 درجه چرخش را بخواهیم اعمال کنیم کار معقولی نیست چرا که نه تنها در دنیای واقعی همچنین عکس هایی از یک زرافه نداریم بلکه این کار باعث میشود نیاز با استخراج فیچر های اضافی داشته باشیم)

(ب)

نقش پارامتر های الفا و بتا بنظر باید به نوعی سعی در متوازن ساختن توجه شبکه ما به دو کلاس مختلف باشد

چرا که با ضرب این دو عدد در y و $(1-y)$ عملا داریم خروجی که مربوط به دو کلاس مختلف است را به نوعی وزن دار میکنیم که اگر اینکار را درست انجام دهیم باعث میشود با داشتن حجم های نا متوازی از داده ها این مشکل را با اعمال درجه اهمیت بیشتر به داده های کم حجم تر جبران کنیم

(پ)

(د)

(سوال 12)

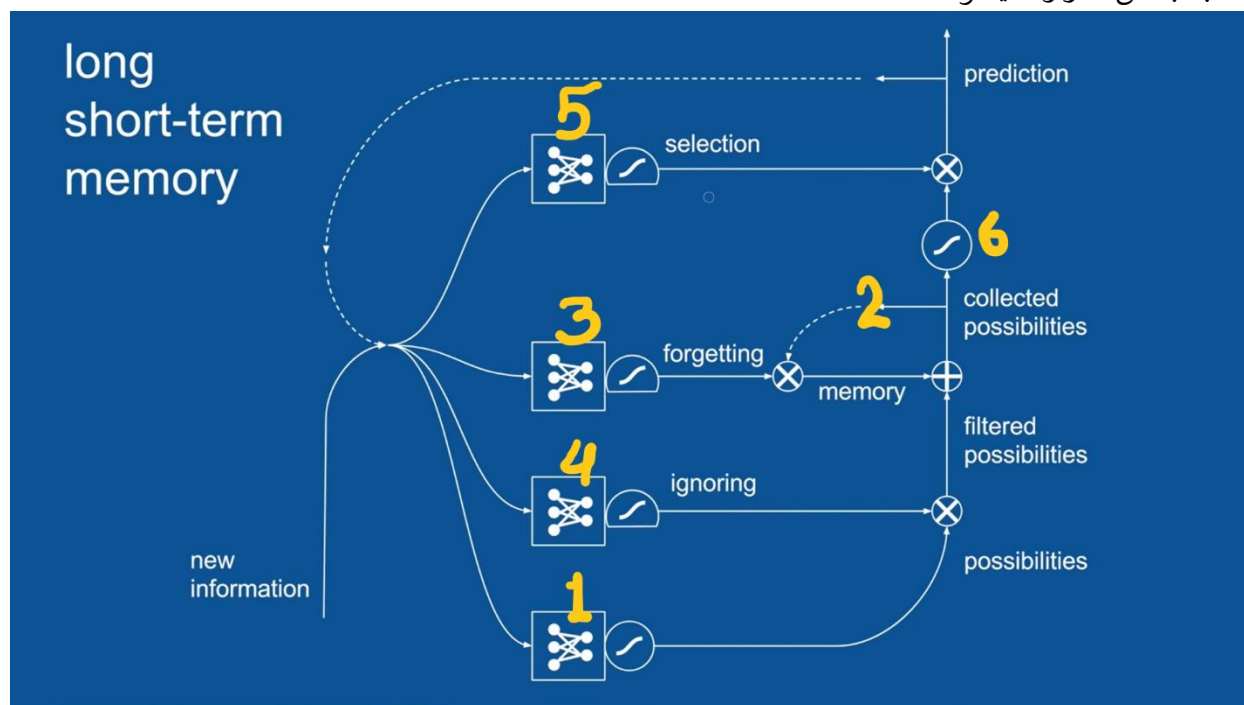
برتری اصلی بر داشتن نوعی حافظه بلند مدت است (مشکل محو شدگی گرادیان که مشکل اصلی rnn هاست در lstm ها وجود ندارد) و گیت های مختلفی که وظایف جداگانه ای دارند مانند یک گیت که تصمیم بگیرد کدام اطلاعات را فراموش کند یا یک گیت که تصمیم بگیرد کدام اطلاعات را به حافظه بلند مدت خود بسپارد و حفظ کند

تمام این گیت ها در روند آموزش یاد میگیرند کدام اطلاعات باید از کدام گیت ها عبور کنند

برای توضیح روند کار این نوع شبکه به عکس زیر توجه کنید

ابتدا اطلاعات وارد شده به 4 شبکه جداگانه (که گیت نامیده میشوند) وارد میشود و هر کدام وظیفه خاصی بر عهده دارند. در قسمت (1) که بخش اصلی شبکه است و با توجه به پیشبینی ها مرحله قبل و داده اهی جدید پیشبینی جدید رخ میدهد. در قسمت (2) یک کپی از همه داده ها به یک حافظه برگردانده میشوند. قسمتی از این کپی فراموش شده و قسمتی از آن به فرایند یادگیری باز میگردند و در (3) یک شبکه جداگانه وجود دارد که وظیفه دارد در طی آموزش یاد بگیرد که چه اطلاعاتی را در حافظه نگه داشته و کدام ها را از یاد ببرد. در (6) یک فعال ساز وجود دارد که مقدار هایی که توسط عمل جمع قبل از این فعال ساز

بدست میایند را در بازه خاصی نگه دارد. در (4) و (5) نیز که توسط دو شبکه کاملاً جداگانه آموزش داده میشوند یاد میگیرند که به ترتیب چه داده هایی را انکار و چه داده هایی را برای پیشبینی بعدی به خروجی تحویل دهد که هر دو در طی روند آموزش این کار را یاد میگیرند. هدف از (4) نوعی تمیز کردن ورودی است که به بخش 2 وارد میشود



(سوال 13)

کارکرد یک اتو انکودر به شکل زیر است :

از 3 قسمت تشکیل شده است

1 – انکودر: این قسمت ورودی را گرفته و آن را بطوری فشرده و خلاصه میکند و به فضایی به اسم latentSpace میبرد.

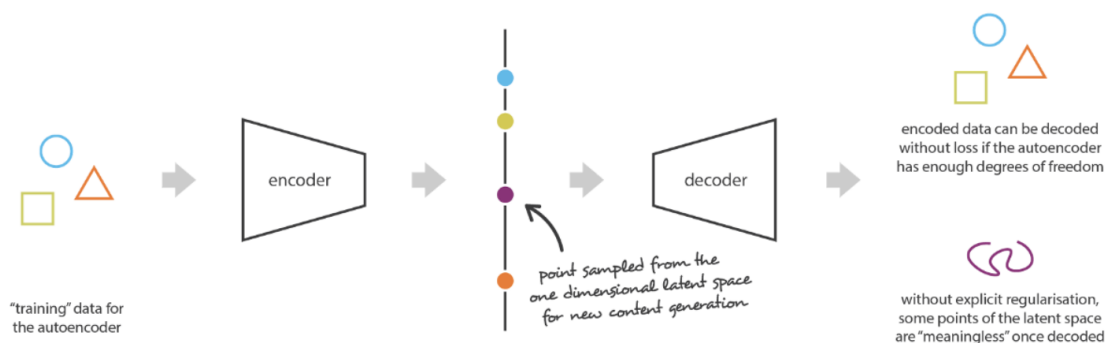
2 – latentSpace : این فضا یک وکتور است که به نوعی چکیده ورودی و دارای خصوصیات ورودی به شکلی فشرده شده است در اصل طی روند آموزش انکودر یاد میگیرد چطور به شکلی بهینه و غنی از ورودی این فضا را بسازد

پس از اتمام آموزش میتوان از این فضا استفاده های بسیاری کرد

3 – دیکودر : این قسمت در طی روند آموزش یاد میگیرد که چگونه با استفاده از فضای بالا ورودی اولیه را بازسازی کند (لزوما هدف میتواند ساخت دقیق ورودی نباشد. میتوان از این شبکه برای گرفتن نویز یا رنگی کردن تصاویر استفاده کرد)

در اصل این ساختار، یک ساختار (unsupervised) هستش. درست است که خروجی را نیز به شبکه میدهم اما خروجی در اصل خود یا حالتی از همان ورودی است

هر چند در این روند شبکه آتوانکودر ما یادمیگیرد که چگونه دیتا ها را بسازد به این دلیل که هدف اصلی شبکه بازسازی با کمترین خطا است و توجه کمتری به ساختار فضای نهان میکند باعث میشود نوعی اور فیتینگ در رابطه با ساختن دیتا های جدید از طریق این فضا رخ دهد برای مثال عکس زیر را نگاه کنید



Irregular latent space prevent us from using autoencoder for new content generation.

در این عکس میبینیم که در بازسازی داده ها خطا بسیار کم بوده اما اگر بخواهیم با تولید یک فضای نهان جدید داده ای جدید را تولید کنیم ممکن است داده تولید شده کاملاً بی معنی باشد چرا که شاید توجه کافی به توضیح مقادیر داده های آموزشی در این ساختار نشده

اینجاست که از آتوانکودر های متغیر استفاده میکنیم

کارکرد آتوانکودر متغیر :

این ساختار تقریباً شبیه به ساختار یک آتوانکودر عادی است با فرق این که به جای اینکه ورودی را با قسمت انکودر به یک وکتور خاص مپ کنیم آن را به توضیحی از فضای نهان مپ میکنیم

با این کار اتفاقی که میافتد این است که در طی روند آموزش نه تنها خطای بازسازی را کم خواهیم کرد بلکه فضای نهان را قاعده مند کرد تا در انتها موجب شود بتوان با دادن فضا های نهان جدید داده های جدید با معنا تولید کرد

(سوال 14)

ساعت کلاس را 7 صبح نداریم (:)

اما مطالب تدریس شده حداقل برای بنده بشدت مفید و جذاب بود، بابت زحمات و صبرتون ممنونم