به نام خدا

پروژه درس هوش محاسباتی

نام استاد : استاد تورانی

اعضای گروه: مهدی آرزومند, محمد محمدیان, رضا حیدری , روزبه رحمانی

موضوع : How to Make a Tic Tac Toe Neural Network

توضیح کلی درباره پروژه :

پروژه ی ما در واقع ایجاد یک بازی دوز است که بتوان یک به یک با کامپیوتر بازی کرد ولی به شرطی که کامپیوتر هی مرتبا بهتر وبهتر شود وباد بگیرد که چگونه هر سری بهتر از دفعه ی قبل بازی کند و در واقع این که بعد از چند بار به بهترین سطح ممکن میرسد و به سطحی میرسد که به شما نبازد وما اگر بهترین بازی ممکن را بکنیم هم نتوانیم آن را ببریم و حتی با اشتباه ما بتواند مارا ببرد.

اول با یک توضیح مختصری در مورد این که شبکه عصبی چگونه یاد میگیرد که هی بهترشود ارایه میدهیم

اطلاعات به دو طریق در شبکه‌ی عصبی جریان دارند: زمانی که در حال یادگیری است؛ یا بعد از اینکه عمل یادگیری انجام شد. در این زمان‌ها الگوهای یادگیری به‌وسیله‌ی واحدهای ورودی وارد شبکه می‌شوند و لایه‌های واحدهای مخفی را برانگیخته می‌کنند و این لایه‌ها به واحدهای خروجی می‌رسند. به این طراحی رایج، شبکه عصبی پیش‌خور **می‌گویند. همه‌ی واحدها همیشه شلیک نمی‌شوند. هر واحدی اطلاعات ورودی را از واحدهای سمت چپ خود دریافت می‌کند و ورودی‌ها در وزن اتصالات مربوطبه خود ضرب می‌شوند. هرواحدی تمامی ورودی‌هایی را که دریافت می‌کند به این طریق جمع می‌زند و (در ساده‌ترین نوع شبکه) اگر جمع بیش‌از یک مقدار** آستانه **مشخص شد، این واحد شلیک می‌کند و واحدهای متصل به‌خود را (که در سمت راست هستند) راه می‌اندازد .**

برای یادگیری یک شبکه عصبی، باید بازخورد **وجود داشته باشد؛ همان‌طور که به کودکان گفته می‌شود که چه چیزی درست است و چه چیزی غلط. درواقع همه‌ی ما همیشه از بازخورد استفاده می‌کنیم. زمانی را به‌خاطر بیاورید که می‌خواستیم برای اولین بار بازی بولینگ را یاد بگیریم. وقتی شما توپ سنگینی برمی‌دارید و آن را پرتاب می‌کنید، مغز شما به‌سرعت چگونگی حرکت توپ و مسیر آن را مشاهده می‌کند و میزان دقت شما را بررسی می‌کند. دفعه بعدی که دوباره نوبت شما رسید، اشتباهات دفعه قبلی خود را به‌یاد می‌آورید و حرکت خود را باتوجه به این اشتباهات اصلاح می‌کنید و امیدوارید که این‌بار توپ را بهتر از قبل پرتاب کنید.**
بنابراین در این مثال از بازخورد **برای مقایسه نتیجه قبلی با نتیجه دلخواه خود استفاده می‌کنید. این بازخورد تفاوت‌ها را مشخص می‌کند و تغییراتی در دستور کار شما برای دفعه بعدی ایجاد می‌کند: باشدت بیشتر پرتاب کردن؛ کمی به‌سمت چپ پرتاب کردن؛ دیرتر رها کردن، و غیره. هرچه تفاوت بین نتایج حقیقی و نتایج دلخواه بیشتر و بزرگ‌تر شود، تغییرات نیز بیشتر خواهد شد.**

بولینگ: شما باکمک شبکه عصبی داخل مغزتان یاد می‌گیرید که چگونه چنین مهارت‌هایی به دست بیاورید. هردفعه که شما توپ را اشتباه پرتاب می‌کنید، یاد می‌گیرید که چه اصلاحاتی باید برای دفعه بعد به‌کار برید .

شبکه‌های عصبی نیز به‌همین روش چیزهای مختلف را یاد می‌گیرند. یادگیری شبکه‌های عصبی با استفاده‌از یک روند بازخوردی را پس‌انتشار **گویند. این عمل عبارت است از: مقایسه‌ی خروجی تولیدی یک شبکه با خروجی که دلخواه و مورد انتظار است. از تفاوت بین این دو خروجی، برای** تغییر و اصلاح **وزن‌های اتصالات بین واحدهای شبکه استفاده می‌شود، با این تفاوت که این روش برعکس است، یعنی از واحدهای خروجی به‌سمت واحدهای مخفی و سپس از آنجا به‌سمت واحدهای ورودی می‌رویم. پس‌انتشار با کاهش تفاوت بین خروجی واقعی و خروجی دلخواه، تاحدی که این دو خروجی یکسان شوند، جلو می‌رود تا شبکه‌ی عصبی دقیقا همان‌طوری که باید و انتظار می‌رود، کار کند.**

شبکه عصبی در عمل چگونه کار می‌کند؟

زمانی که شبکه توسط نمونه‌های یادگیری کافی، آموزش داده شد، به نقطه‌ای می‌رسد که می‌توان یک سری جدید از ورودی‌ها را وارد آن کرد که قبلا آن‌ها را ندیده باشد و واکنش شبکه به این ورودی‌های جدید را مشاهده کرد. به‌عنوان مثال، فرض کنید که با نشان دادن تصاویر زیادی از صندلی و میز درحال آموزش دادن یک شبکه هستید و به‌گونه‌ای به شبکه آموزش می‌دهید که کامل مفاهیم شما را درک کند و به شما

بگوید که تصویر متعلق به صندلی است یا میز. وقتی شما به‌اندازه‌ی کافی، تصویر صندلی و میز را به این شبکه نشان دادید؛ مثلا تعداد ۲۵ میز و ۲۵ صندلی، طرح جدیدی از صندلی یا میز را به آن نشان می‌دهید که قبلا آن را ندیده باشد و می‌بینید که شبکه‌ی شما چه واکنشی نشان می‌دهد. بسته به نوع آموزش شما، شبکه تلاش می‌کند که نمونه‌ی جدید را دسته‌بندی کند و بگوید که آیا نمونه، تصویر صندلی است یا میز. شبکه کار دسته‌بندی را مانند انسان و با استفاده‌از تجارب گذشته انجام می‌دهد. درواقع شما به کامپیوتر یاد داده‌اید که چگونه لوازم را شناسایی کند.

البته این بدین معنی نیست که شبکه‌ی عصبی بتواند به نمونه‌ها نگاه کند و فورا مثل انسان به آن‌ها واکنش درست نشان دهد. مثالی را که زدیم درنظر بگیرید: این شبکه به میز یا صندلی نگاه نمی‌کند. ورودی‌های شبکه اعداد باینری هستند: هر واحد ورودی یا ۰ است یا ۱. بنابراین اگر شما ۵ واحد ورودی داشته باشید، می‌توانید اطلاعات ۵ مشخصه متفاوت صندلی‌های مختلف را با استفاده از جواب باینری (بله/خیر) پاسخ دهید. سؤالات ممکن است به این طریق باشند:

- آیا این شیء پشتی دارد؟
- آیا قسمت بالایی دارد؟
- آیا تکیه‌گاه آن نرم است؟
- آیا می‌توان برای مدت طولانی به‌طور آسوده روی آن نشست؟
- آیا می‌توان چیزهای بسیاری روی آن قرار داد؟

اگر نمونه‌ی ارائه‌شده یک صندلی معمولی باشد، جواب می‌شود: بله؛ خیر؛ بله؛ بله؛ خیر با کد باینری ۱۰۱۱۰. اگر نمونه یک میز معمولی باشد، جواب می‌شود: خیر؛ بله؛ خیر؛ خیر؛ بله با کد باینری ۰۱۰۰۱. بنابراین شبکه به اعداد باینری نگاه می‌کند و ازطریق خروجی این اعداد تشخیص می‌دهد که شی صندلی است یا میز.

حالا میرسیم به پروژه ی ما :

خوب در این پروژه ما 5تا فایل داریم که به ترتیب توضیح میدیم این فایل ها چیست :

این دو فایل :

1- tictacNET.h5

2- tictactoe-data.csv

این دو دیتا بیس پروژه ی ما هست .

tictacnet.py این فایل مربوط به هوش پروژه ی ما است .

tictactoe.py این فایل مربوط به تابع هایی است که از آن استفاده میکنیم .

play.py این فایل اصلی ما است که در آن بازی میکنیم .

tictacnet.py این فایل که شبکه عصبی توش هست به این صورت هست که از فایل **tictactoe-data.csv** یک سری دیتا هست که از روی اون میخونه که این دیتا به این صورت هست که یک سری x هستند که میشن دیتای ورودی یک سری y هستند که میشن اون خروجی که باید بهمون بده حالا این xها شرایط مختلف صحفه بازی اند که داحل اون فایل هستند به طور مثال اگه مثلا xداخل یک بود و یک xدیگر ذاخل 2 بود و 3 خالی بود حرکت درست این است که داخل 3 ,oرابگذاریم .دیتا هامون به این شکل هستند . حالا دیتا ها شکسته میشند یک سری میشوند برای ترین شبکه عصبی یک سری هم میشوند برای تست شبکه عصبی .

مفهوم ترین وتست کردن شبکه عصبی :

اول مثلا x ترین را بهش نشون میدیم میگیم yترین این شکلی است یعنی اگراین x رو گرفتی باید این y رو بدی بعد سر تست xتست را بهش میدیم بعد میگیم خب حالا چه y رو باید بدی وبعد بر اساس مقدار جواب های درستی که شبکه عصبی میده درصد دقت شبکه عصبی رو بهمون میده که این شبکه عصبی ما 92.5% دقت داشته است .

این جا جایی است که شبکه عصبی رو میسازیم

```
1-model = tf.keras.Sequential()
2-model.add(tf.keras.layers.Dense(128, activation="relu", input_dim=X.shape[1]))
3-model.add(tf.keras.layers.Dropout(0.3))
4-model.add(tf.keras.layers.Dense(64, activation="relu"))
5-model.add(tf.keras.layers.Dropout(0.3))
6-model.add(tf.keras.layers.Dense(32, activation="relu"))
7-model.add(tf.keras.layers.Dropout(0.3))
8-model.add(tf.keras.layers.Dense(moves.shape[1], activation="softmax"))
```

خط دوم : اولین لایه شبکه عصبی 128 تا نورون دارد

خط سوم : بعد یک لایه dropout

خط چهارم : یک لایه 64 تایی

بعد دوباره یک dropoutبعد یک لایه 32 تایی بعد دوباره dropout

خط آخر این رو وصل کردیم به یک تک نورون خروجی که یک عدد میده y از یک تا نه که از بین خونه یک تا نه کدومش رو باید کامپیوتر بزاره تو جدول بازی .

```
model.compile(optimizer="adam", loss="categorical_crossentropy",
metrics=["accuracy"])
```

این جا هم که همان مفهوم کامپایل است که مدل را میسازد و ایجادش میکند

```
model.fit(
  X_train,
  y_train,
  epochs=100,
  batch_size=16,
  validation_data=[X_test, y_test],
)
```

این model.fit هم مدل شروع میکنه به ترین کردن که میگه این دیتا هایی که بهت دادم 16 تا 16 جدا کن هر کدوم از این 16 تا 100 بار ببین

```
print("accuracy:", model.evaluate(X_test, y_test))
print("Custom accuracy:", move_accuracy(y_test.values, model.predict(X_test)))
```

این جا هم درصد و میزان دقت را به عنوان خروجی میدهد

```
model.save("tictacNET.h5")
```

با این خط هم اون دانشی که بدست اوورده از دیدن این xها و y ها را بدست model.save ذخیره میکند در فایل tictacNET.h5 که بعدا ما این فایل را لود میکنیم و شبکه عصبی با اون دانشی که این تو ذخیره شده میاد بازی میکنیم مثلا بهش یک x میدیم بعد یک y بر میگردونه مثلا بهش یک آرایه 9تایی میدیم این یک y ای میده مثلا 5 یعنی تو خونه 5 بزار