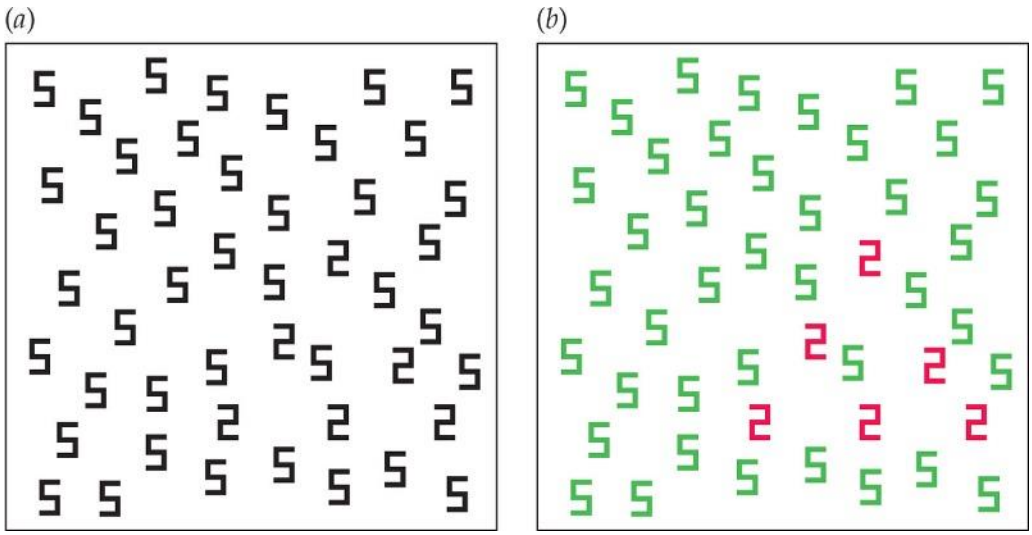
یادم تو را فراموش!*

خوب بچه‌ها می‌خواهیم با چند فوت و فن ریاضی آشنا بشیم که ممکنه در شبیه‌سازی حافظه به شما کمک کنه. یادتون باشه که وقتی ما از فرمولهای ریاضی برای شناخت یک پدیده استفاده می‌کنیم همیشه به‌این معنی نیست که خود پدیده هم از فرمول های ریاضی استفاده می‌کنه! مثلاً می‌تونیم حرکت زمین به‌دور خورشید رو با دقت زیاد به وسیله چند فرمول ریاضی توصیف کنیم ولی زمین برای گردش دور خورشید لازم نیست فرمول ریاضی حل کنه! حالا این که رابطه‌ی ریاضی و مغز چیه؟ سؤال خیلی جالبیه که هنوز چیز زیادی در موردش نمی‌دونیم، اما بد نیست بدونین که در مغز یه ناحیه پیدا شده که با اعداد سروکار داره، یعنی وقتی مثلاً به شمارش اعداد مشغولید سلول‌های این ناحیه فعال می‌شن. هنوز نمی‌دونیم مغز ما چطور مفهوم پیچیده‌ای مثل عدد رو یاد گرفته؟ ولی تقریباً ثابت شده که حیوانی مثل گربه حداکثر تا عدد سه رو می‌تونه یاد بگیره و این توانایی در میمون به‌سختی به عدد پنج می‌رسه! اینکه چه مکانیزمی در مغز ماست که به راحتی می‌تونیم تا بی نهایت عدد رو شمارش کنیم هنوز برامون معماست (شاید شما جزو کسانی باشید که در آینده‌ی نزدیک بتونن این مکانیزم رو کشف کنن). بعضی وقت‌ها یک معمای حل نشده در مغز مثل شناخت اعداد به معمای دیگه‌ای تبدیل می‌شه که شاید از اولی جالب‌تر باشه. مثلاً معلوم شده که ناحیه اعداد، کنار ناحیه‌ای از مغز است که با شناسایی و تشخیص رنگ سروکار داره. از قضا دربعضی افراد فعالیت این دو ناحیه باهم تداخل پیدا می‌کنه و پدیده‌ی جالبی اتفاق می‌افته که مثلاً یه بیمار ممکنه عدد 5 را به‌رنگ آبی و عدد ۷ را به‌رنگ قرمز ببینند. به‌این پدیده اصطلاحاً سینتسزی (Synesthesia حس آمیزی) می‌گویند. این افراد می‌تونند به‌آسونی آزمایشی انجام بدن که برای افراد "معمولی" خیلی سخته مثلاً جای اعداد ۲ ای را که لابه‌لای تعداد زیادی پنج پنهان شده فوراً پیدا کنند (فرض کنید که تعداد زیادی ۵ به‌رنگ خاکستری و به‌طور پراکنده روی صفحه داشته باشیم و فقط تعدادی عدد ۲ لابه لای اون‌ها قرار گرفته باشه پیدا کردن این دو ها کار مشکل و زمان‌بریست. اما اگه همین پنج و دوها به رنگ‌های متفاوت باشن، به راحتی شما می‌تونین با یک نگاه اون‌ها رو از هم تشخیص بدید. میتونید چند آزمایش انجام بدین و این پدیده رو در بین دوستان خودتون مطالعه کنین.)

---

*حتما در کودکی با دوستان یا یکی از اعضای خانوادتون "یادم تورا فراموش" بازی کردید. این بازی، از بازی های قدیمی و جالب ایرانیه! که به طرز جالبی از بسیاری توانایی‌های سیستم عصبی استفاده می‌کنه. توانایی هایی مثل حافظه و صد البته توجه!!! اینکه چطور سیستم عصبی می‌تونه چنین توانایی‌هایی داشته باشه، از چالش‌های بزرگ دانشمندانی هست که به مغز و نحوه‌ی کار اون علاقه‌مندند. در این بخش به‌طور مختصر راجع به حافظه بحث می‌کنیم



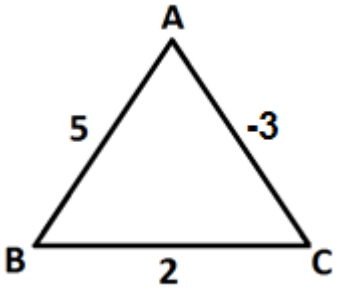
نکته‌ی اگه به عکس (a) نگاه کنید پیدا کردن تمام ۲ ها بین این همه ۵ کار سختی به نظرمیاد، اما اگه اعداد رنگی باشن (مثل عکس (b)) می‌بینین که با یه نگاه سریع به راحتی همه ۵ ها از ۲ ها قابل تشخیص هستن.

برگردیم سر اصل مطلب! اول از همه باید منظور خودمون رو از فعالیت یک سلول عصبی مشخص کنیم. در ساده‌ترین حالت فعالیت یک سلول عصبی با دو عدد صفر و یک نشون داده می‌شه، صفر به‌معنی خاموش و یک به‌معنی روشن. پس اگر مثلاً پنج سلول داشته باشیم حداکثر ۳۲ حالت خواهیم داشت و اگر تعداد این سلول‌ها به فقط صد سلول برسه، تعداد کل حالت‌هایی که این سیستم می‌تونه به خودش بگیره چیزی حدود 2^{100} حالت میشه که نزدیک به تعداد کل اتم‌های جهان ماست (درمورد یک سوسک عادی که حدود یک میلیون نورون داره، اصلاً فکرش را هم نکنید که این نورونها می‌تونن چه تعداد حالت مختلف به خودشون بگیرن؟!).

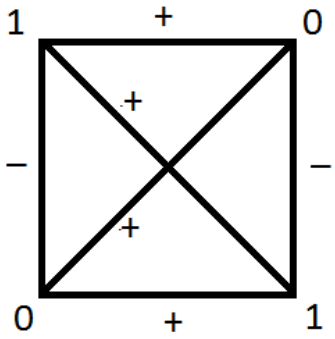
سلولهای عصبی یا نورون‌ها به هم متصلند و اتفاقا یکی از پیچیدگی‌های مغز همین نحوه‌ی اتصال نورون‌ها به یکدیگر است. بعضی نورون ها به نورون های زیادی متصلند و بعضی‌ها فقط با تعداد کمی اتصال دارند. بعضی اتصال‌ها شکل منظمی دارن و به‌نظر می‌رسد که بعضی اتصالات هم تصادفی هستند. اگر کلمه‌ی گراف به گوشتان خورده باشد (اگر هم نخورده مهم نیست!) می‌بینید که می‌شه یک دسته نورون رو با یک گراف نشون داد. هر نورون به عنوان یه راس گراف در نظر گرفته میشه و بین نورون هایی که به هم متصلند یه خط رسم می کنیم. به عبارت دیگه اتصال بین نورون ها با یال های گراف نشون داده می‌شن. در این مجموعه بعضی نورون‌ها با فعالیتشون بقیه نورون‌ها رو تحریک

به فعالیت می کنن (باعث می‌شن که بقیه نورون‌ها از حالت صفر به یک تغییر وضعیت بدن). به این نورون‌ها، نورون‌های تحریکی یا تقویتی گفته می شه و بعضی از نورون ها هم وقتی فعال می شن بقیه نورون‌ها رو تضعیف یا مهار می‌کنند (به این نورون‌ها، نورون‌های مهاری گفته میشه). یک اصل مهم می‌گوید که یک نورون در تمامی اتصالات خود یا تنها نقش تحریکی دارد یا درتمامی آنها نقش مهاری رو ایفا میکنه. شما می‌توانید درباره‌ی این اصل که به اصل دیل (Dale) معروف است، بیشتر تحقیق کنید ولی باعرض پوزش ما دراین‌جا این اصل را زیر پا می‌گذاریم! و نورون‌هایی که در نظر می‌گیریم می‌تونن بعضی سلول‌ها را تقویت کنند و بعضی را تضعیف (بی‌خود نیست که به‌کار ریاضی دانان وقتی مسائل رو ساده می‌گیرند با تردید نگاه می‌کنند ولی ناراحت نباشید چون در این حالت توجیه خوبی وجود داره و این ساده‌سازی مشکل ساز نیست. مثلاً هروقت یک نورون تقویتی بخواد نورون دیگه‌ای رو تضعیف کنه می‌تونه با کمک یک نورون واسطه این کار رو انجام بده. آیا می تونید حدس بزنید چطوری؟)

پیش از این‌که اولین بازی ساده‌ی خودمون رو شروع کنیم خودتون رو جای یکی از این نورون ها قرار بدین (تقویتی یا تضعیفی فرقی نداره). اگر در لحظه‌ی فعلی تقویت بشین طبیعی است که خودتون رو فعال یا روشن نگه دارین و در غیر این صورت ترجیح بدین که غیرفعال یا خاموش بمونید. اگر نیروی تقویت‌کننده با نیروی تضعیف‌کننده برابر شه چی؟ خوب اگر آدم مثبتی باشید شاید تصمیم بگیرید که روشن باشید و اگر منفی باشید برعکس عمل می‌کنید. در شکل زیر سه نورون A, B و C داریم بطوری‌که A و B به اندازه‌ی ۵ واحد همدیگه رو تقویت می‌کنند. به‌این عدد میگن وزنه یا وزن اتصال (وزنه در واقع ضریب اثرگذاری دو نورون A و B روی یکدیگره، به این معنی که وضعیت A در یک مرحله با چه ضریبی روی وضعیت مرحله بعد B اثر گذاره و بر عکس) و همینطور که درشکل پیداست وزنه‌ی اتصال A و C برابر ۱- و B و C برابر ۲ است. فرض کنید در لحظه‌ی فعلی A خاموش (A=0) و B و C هر دو روشن باشن (C=1, B=1) اگر نورن ها بتوانند به‌طور هم زمان وضعیت خودشون‌رو در لحظه‌ی بعد تعیین کنند چه اتفاقی خواهد افتاد؟ خودتون رو قانع کنید که B و C روشن باقی می‌مونه و A نیز ترجیح می ده از حالت خاموش به‌حالت روشن تبدیل بشه. از آن جالب‌تر این‌که این وضعیت برای همیشه ثابت می‌مونه!! (تمرین: یال AC را از ۱- به ۳- تبدیل کنید و سعی کنید یک نقطهٔ ثابت جدید پیدا کنید).

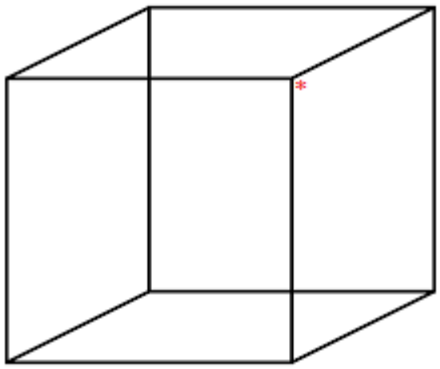


در مثال زیر چهار نورون داریم و فرض می‌کنیم وزن‌های نورون‌ها باهم برابر است و فقط علامت‌هاشون باهم تفاوت داره. وضعیت اولیه شبکه به این صورت است : A=0، B=1، C=0، D=1 که می‌توان آن را به صورت (۰و۱و۰و۱) نشان داد. حالا مثل قبل خود را قانع کنید که وضعیت (۰و۰و۰و۰) و (۱و۱و۱و۱) همیشه ثابت می‌مونه.



اما وضعیت‌های جالب دیگه‌ای هم می تونن اتفاق بیافتن، مثلاً اگر وزن‌ها را از حالت تقارن خارج کنیم ($W_{ij} \neq W_{ji}$)، می‌تونیم کاری کنیم که از حالت(۰و۰و۱و۱) شروع کنیم، به وضعیت (۱و۱و۰و۰) برسیم و بعد دوباره به‌حالت (۰و۰و۱و۱) برگردیم و همین طور الی آخر. به عبارت دیگر ما یک وضعیت تناوبی داریم که مدام تکرار می‌شه(سعی کنین وزن‌های مناسب برای چنین سیستم متناوبی رو بیابید. می‌تونید برای شروع با ۳ نورون کار رو آغاز کنید). البته چون در مثال فعلی تعداد حالت‌ها خیلی کم است به‌آسونی همه ی نقاط ثابت و تناوبی آن قابل محاسبه است. ولی اگر صد نورون داشته باشیم کارمون دشوار می‌شه. به‌طورکلی وقتی شما پدیده‌ای رو مشاهده می‌کنید که با زمان تغییر می‌کند (مثل دو مثال قبلی) اولین سؤالی که مطرح می‌شه اینه که آیا به یک نقطه‌ی تعادل یا ثابت می‌رسه یا نه؟ سؤال بعدی اینه‌که آیا دور تناوبی هم داره یا نه؟ البته مسئله پایداری هم خیلی مهمه چون اگر پایداری نباشه با کوچک ترین اختلالی نقطه ثابت یا دور تناوبی به‌هم می‌ریزه. تصور کنید که اگر دور تناوبی منظومه شمسی پایدار

نباشه چه خواهد شد؟ خوشبختانه براساس یک سری مطالعات پیچیده ریاضی حداقل تا میلیون‌ها سال دیگه این دور تناوب ادامه داره و شما می‌توانید شب‌ها راحت بخوابید. این مثال مارو با شاخه‌ی مهمی در علم ریاضی به‌نام سیستم-های دینامیکی آشنا می‌کنه و به همین خاطر وقتی ما مغز را به عنوان یک سیستم دینامیکی درنظر می‌گیریم با ریاضیات هیجان‌انگیزی روبه رو هستیم که هنوز پدیده‌های ناشناخته زیاد داره و شاید روزی مجبور شوید ریاضیات جدیدی کشف کنید تا مغز را بهتر بشناسید. البته این ریاضیات جدید می‌تونه در دل مسئله‌ای پنهان شده باشه که خیلی به‌نظر ساده بیاد. مثلاً چرا وقتی به‌شکل یک مکعب نگاه می‌کنیم گاهی‌اوقات یک رأس مکعب به‌نظر نزدیکتر میاد و بعد دورتر می‌شه. به‌نظر می‌رسد که در مغز یک رقابت وجود داره و این دوری و نزدیکی به‌طور تناوبی تکرار می‌شه. تاکنون مدل‌های ریاضی جالبی در توصیف این پدیده ارائه شده که هیچکدام هنوز جنبه ی قطعی نداره !!



به نظر شما راس ستاره‌دار مکعب مربوط به وجه نزدیک مکعب به شماست (جلویی) یا وجه دورتر(پشتی) ؟

اگربرای مدتی به تصویر نگاه کنید احتمالا جای این دو حالت تغییر می‌کند!

این‌که یک پدیده‌ی ثابت بیرونی، مثل شکل یک مکعب می‌تواند به دو ادراک بینایی متفاوت منجر بشه مسئله جالبی است که به ادراک دوبینی یا رقابت دوبینی (Binocular Rivalry) مشهور است و با آزمایش‌های بسیار هوشمندانه‌ی سایکوفیزیکی و مدل‌های ریاضی مهیج، دانشمندان علوم شناختی را به‌خود مشغول کرده (شما هم می‌توانید دراین باره بیشتر تحقیق کنید و مطلب جدید کشف کنید). حالا برگردیم به مسئله‌ی سیستم‌های دینامیکی و این‌که چرا نباید مسائل به‌ظاهر ساده رو دست کم گرفت؟ یک مسأله جالب که شاید در کتاب‌ها خوانده باشید مدل کردن رشد جمعیت در یک گونه‌ی جانوری است. برای لحظه‌ی شروع، دو حالت متمایز با جواب نهایی بدیهی می‌توان درنظر گرفت. حالت اول این‌که جمعیت صفر باشه که بدیهی هست که رشدی هم در کار نخواهد بود و جمعیت صفر

می‌مونه. حالت دوم وقتی است که جمعیت به‌قدری زیاد بشه که منبع غذایی در دسترس نباشه و در نتیجه همه از گرسنگی تلف می‌شن. فرض کنید جانوری پیدا بشه که رشد جمعیت‌اش با تابع ساده زیر قابل توصیف باشد: $f(x) = ۲x\ (۱-x)$. استفاده از این تابع برای مدل کردن رشد جمعیت به‌این معنی است که مثلاً اگر جمعیت درلحظه‌ی فعلی $\frac{1}{3}$ واحد است (واحد را معادل ۹۰۰۰۰۰ در نظر بگیرید) مقدار تابع رو به ازای $x=\frac{1}{3}$ (یعنی جمعیت در لحظه فعلی برابر ۳۰۰۰۰۰ است) محاسبه می‌کنیم که می‌شود $f\left(\frac{1}{3}\right) = \frac{2}{3} \times \frac{2}{3} = \frac{4}{9}$ (دراین‌جا واحد زمان ممکن است سال یا ماه باشد، یعنی در واحد بعدی زمان جمعیت به ۴۰۰۰۰۰ خواهد رسید). البته همانطور که انتظار داریم اگر جمعیت در لحظه شروع را صفر و یا یک قرار بدین نتیجه این خواهد بود که درلحظه بعد جمعیت صفر می‌شه و با صحبت‌هایی که تابه‌حال داشتیم این یعنی که صفر یک نقطه ثابت برای این مدل جمعیت است. یه سوال جالب اینه که آیا نقطه‌ی ثابت دیگه‌ای هم هست یا نه؟ با کمی فکر کردن می‌بینید که نقطه $\frac{1}{2}$ هم یک نقطهٔ ثابت است (یعنی $f(\frac{1}{2}) = \frac{1}{2}$) پس اگر جمعیت ۴۵۰/۰۰۰ تا بشه روی همین تعداد باقی می مونه). البته شاید جانوری درهیچ مکانی پیدا نشه که چنین مدلی در موردش صدق کنه ولی خوب می‌شه با این مدل بازی کرد و نتایجی بدست آورد که کاملاً غیرمنتظره است.

تغییرات ضریب تابع هم باعث تغییرات جالبی در رفتار جمعیت خواهد شد، مثلا اگر به‌جای ۲ عددهای بزرگتر قرار بدین و همین آزمایش را تکرار کنید، (بله! با ریاضیات میشه پدیده‌های جدیدی کشف کرد که به اثبات قضایای مهم در ریاضی منجر می‌شه) اتفاقات جالبی خواهید دید. مثلاً اگر به‌جای ۲ عدد ۳,۱ قرار دهید، $f(x) = 3.1x\ (۱-x)$) علاوه بر نقطه ثابت نقاطی با تناوب ۲ ظاهر می‌شود (یعنی اگر از یکی از این نقاط شروع کنید بعد از گذشت دو واحد زمانی، باز به همان نقطه شروع بر می‌گردید. نقطه ثابت در واقع نقطه‌ای است با تناوب یک، یعنی بعد از هر بار جابه‌جایی(گذشت هر واحد زمانی) به خودش می‌رسد). وقتی عددی تقریباً به اندازه‌ی ۳/۴۵ را برای ضریب انتخاب کنید، نقطه‌ای با تناوب ۴ ظاهر می‌شه و برای ضرایب $\geq$۳,۴۵ ضریب$\geq$۴ ، ظهور مدارهای تناوبی با شدت بیشتری مشاهده می‌شه تا جایی که برای $n=4$ یک قضیه نسبتاً سخت ریاضی ثابت می‌کنه که بینهایت مدار متناوب با تناوبی از جنس اعداد اول ظاهر خواهد شد. حالا کی فکرش را می‌کرد که تابع ساده‌ای مثل $f(x) = \alpha x(\beta-x)$ ، این همه رفتار پیچیده از خودش نشون بده (تا چند دهه قبل هیچکس!!). این تابع ($f(x) = \alpha x\ (\beta-x)$) که به تابع لجستیک (Logistic function) معروفه، خیلی‌ها رو غافلگیر کرد، (به‌طوری که در لینک زیر:

http://www.mathematica-journal.com/2013/05/using-the-logistic-map-to-generate-scratching-sounds /

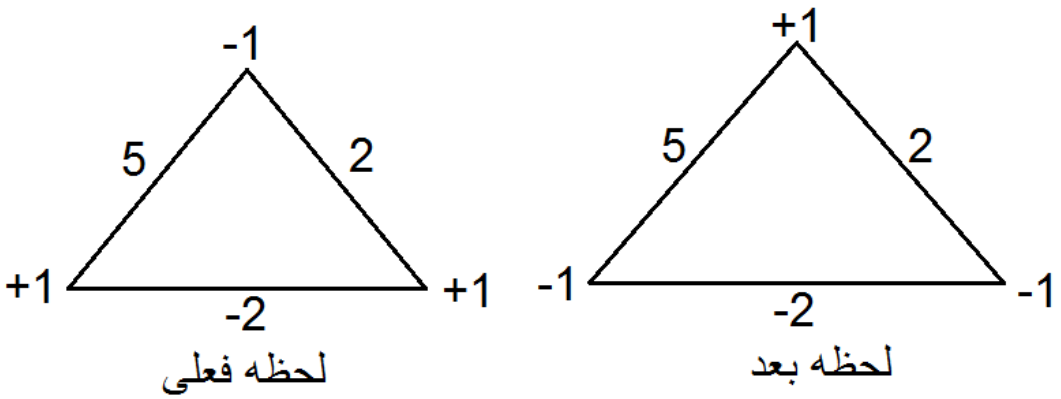
مدلی که این تابع رو توصیف می کنه در نرم‌افزار Mathematica نشان داده شده و خروجی تابع به شکل یک فایل صوتی پخش می‌شه که صدای شبیه یک طوفان شدید یا گردبادی هولناک داره). جالبه بدونید مدلی که برای شبیه سازی رفتار نورون‌ها به کار میره و به مدل هاجکین– هاکسلی معروفه از این دست پیچیدگی‌ها زیاد داره (البته هنوز به معنی دقیق ریاضی آشوبناک بودن این مدل به اثبات نرسیده. این دو دانشمند به‌خاطر مدل زیبایی که برای نورونها بدست آوردند موفق شدند جایزه نوبل بگیرند). پس باید خودمون رو آماده کنیم که در آینده‌ی نه چندان دور با مدل‌های شگفت‌انگیزتری از طرز کار مغز برخورد کنیم.

سوالات بنیادی زیادی راجع به مغز هنوز بی‌پاسخند، مثلاً درمورد اینکه اطلاعات در مغز چگونه ذخیره می‌شود و اینکه اصلاً حافظه یعنی چه؟ هنوز خیلی کم می‌دونیم، اما انتظار داریم با پیشرفت‌های زیادی که در علوم تصویربرداری و تحریک مغز به‌دست اومده مدل‌های بهتری از حافظه بسازیم. البته اصلاً انتظار نداریم که وقتی تصویر مغز را خیلی بزرگ کنیم یک کتابخانه‌ی بزرگ ببینیم. تا اینجا شواهد نشون می‌دن که اوضاع عجیب و غریب‌تر از این حرف‌هاست. بیشتر شبیه این است که اگر هرخاطره‌ی ما مثل یه کتاب باشه، "صفحات" این خاطره در حافظه به‌طور درهم و برهم در جاهای مختلف مغز پراکنده شدن و هرزمان که خاطره ای به یادمان میاد این صفحات مثل یک شعبده بازی در کنار هم با همان نظم و ترتیب اولیه ظاهر می‌شوند. شاید آخرین چیزی که به ذهنمون برسه این باشه که چنین پدیده‌ی حیرت‌آوری را هم می‌شه با یک مدل ریاضی توصیف کرد. به همین خاطر وقتی فیزیکدانی به اسم هاپفیلد برای اولین بار مدل ساده‌ای برای حافظه ارائه کرد، تعجبی نداشت که بسیاری از فیزیکدان‌ها و ریاضیدانان علاقمند به مدل‌سازی حافظه شدند. شما هم باید وقتی این نوشته رو تا آخر مطالعه کردید به قدر کافی اطلاعات کسب کنید تا بتونید مدل‌های مختلفی از حافظه بسازید که طرز کارشون شبیه حافظه در مغز باشد. برای این کار بازی با یک دسته نورون‌های خاموش و روشن رو ادامه می‌دیم. با این تغییر کوچک که به جای صفر و یک از نماد −۱ و +۱ استفاده می‌کنیم*

---

*(فیزیکدان‌ها این نحوه‌ی نمایش را بیشتر دوست دارند چون به ذرات فیزیکی، اسپین نسبت می‌دن که می‌تونه جهت‌های بالا و پائین اختیار کنه (اسپین یک خاصیت مغناطیسی ذرات هست که در حالتی که اسپین بالا باشه با +۱ و وقتی پایین باشه را با −۱ نشونش میدن) میدونیم که میدان مغناطیسی یه ذره با میدان مغناطیسی ذره‌های دیگه برهم کنش میکنه و روی وضعیت مغناطیسی اونا هم اثر میذاره، جالبه که برای نورون ها هم دو وضعیت داریم و هر نورون می‌تونه ←

حال مانند مثال‌های قبل اگر سه تا نورون داشته باشید طوری که در لحظه فعلی نورون اول مقدارش ۱- باشد (s=1) و نورون دوم و سوم هردو ۱+ باشن و اتصال بین نورونها را هم داشته باشیم می‌تونیم مقدار نورونها در لحظه بعدی رو مثل شکل زیر پیدا کنیم:



در واقع برای هر نورون یک ورودی در نظر می‌گیریم که مساوی است با حاصل جمع مقدار هر نورون ضربدر وزن اتصالش و بعد به علامت مثبت یا منفی توجه می‌کنیم:

نورون بالایی : ۰ < (۱×۲) + (۱×۵) پس از ۱- به ۱+ تبدیل می‌شه.

نورون سمت چپ : ۰ < (۱×۲-) + (۱-×۵) پس از ۱+ به ۱- تبدیل می‌شه.

نورون سمت راست : ۴- = (۱×۲-) + (۱-×۲) (در حالتی ورودی به یک نورون صفر شد قرار داد می کنیم که حالت نورون را به ۱+ تغییر بدیم!) حالا یک قانون جدید به بازی اضافه می‌کنیم که مارو یک قدم به بحث حافظه نزدیکتر می‌کند. این قانون به زبان خیلی نادقیق میگه که اگر دوتا نورون در یک لحظه هم‌علامت باشند، مثلاً هردو

⟵روی رفتار بقیه نورون‌ها تاثیر بذاره. پس دور از ذهن نخواهد بود اگه فرض کنیم که نورون‌ها هم یک نوع ذره‌ی فیزیکی باشن و (مثل الکترون که اسپین ±۱ داره) دو حالت داشته باشن و از مدل‌هایی که برای بررسی رفتار اون ذره ها استفاده می‌کنن برای مدل کردن رفتار نورون‌ها استفاده کنیم.)

نکته‌ی جالبی که در مورد این نحوه‌ی نمایش اینه که با وجود اینکه در وضعیت خاموش به جای صفر از منهای یک استفاده می‌کنیم، اما در نتیجه نهایی تفاوت چشمگیری به وجود نمی‌آید !!!

۱+ یا هردو ۱- باشند به اندازه‌ی یک واحد به وزن اتصالشون اضافه می‌شه و در غیر این‌صورت به اندازه یک واحد از وزن اتصالشون کم می‌شه. پس در مثال بالا با استفاده از این قاعده -که قانون هب (Hebb) نام داره- اتصال‌ها در لحظه بعدی این چنین تغییر می‌کند: ۵ می‌شد ۴ چون نورون اول و دوم هم‌علامت نیستند، ولی ۲- می‌شه ۱- چون نورون دوم و سوم هم‌علامت هستند و ۲ می‌شه ۱+ چون بازه دوباره نورون اول و سوم هم‌علامت نیستند. البته کاری که کردیم خیلی درست نیست، چون خیلی عجیبه که اتصال نورون‌ها به‌طور لحظه ای تغییر کنه (اتفاقاً این روزها بعضی‌ها فکر می‌کنند که چنین چیزی دقیقاً ممکن است در مغز اتفاق بیوفته که فعلاً به آن کاری نداریم). دوم این که اگر بخواهیم به‌طور همزمان هم اتصالها و هم فعالیت نورونها تغییر کنه کارمون خیلی سخت می‌شه. به‌همین خاطر در شبیه‌سازی‌ها که خود شما هم آن‌را انجام خواهید داد زمان خاصی را به تغییر اتصالها در نظر می‌گیرند که زمان یادگیری نامیده می‌شود و وقتی می‌خواهند اطلاعات ذخیره شده را بازیابی کنند اتصال‌ها را ثابت می‌گیرند. اما خیلی مهمه که وقتی آقای هب، نوروفیزیولوژیست (اگر این کلمه را بار اول خوب تلفظ کردید جایزه دارید!) برجسته‌ی کانادایی، در اویل قرن بیستم مسأله‌ی تغییرات اتصال بین نورونها را مطرح کرد، تعداد کمی از دانشمندان علوم اعصاب چنین عقیده‌ای را قبول داشتند و همه فکر می‌کردند که اتصالات مغز ما از بدو تولد تا سن پیری ثابت باقی می‌مانند. اما این روزها مسأله‌ی تغییرپذیری یا انعطاف‌پذیری اتصال‌های مغزی یک امر پذیرفته شده است.

برای این‌که ارتباط بین تغییر اتصال‌ها و حافظه را بهتر درک کنیم اول یک مثال ساده می‌زنیم. فرض کنید چند تصویر دیجیتالی از حروف A و F و C دارید که می‌خواهید در یک شبکه‌ی عصبی ذخیره کنید. بدیهی است که هرچقدر تعداد پیکسل‌های صفحه‌ی نمایش بیشتر باشد، این حروف قشنگتر دیده می‌شن اما حتی در یک تصویر ۱۰۰× ۱۰۰پیکسلی هم (مجموعا ده‌هزار پیکسل). می‌تونید این حروف را به راحتی از هم تشخیص بدین.

هر چشم ما معادل صفحه ای است که تقریباً ۱۰۰ میلیون پیکسل یا سنسور نوری داره، یعنی در هرفریم می‌تونه به اندازه‌ی صد مگابایت اطلاعات به مراکز بینایی مغز مخابره کنه. در مورد تصاویر متحرک با فرض ۲۴ فریم در ثانیه این یعنی حداقل دو گیگابایت اطلاعات در ثانیه آنهم برای تصاویر سیاه و سفید (معلوم می‌شه مرکز مخابرات مغز کارش رو خوب بلده!). اگر به هر پیکسل یک نورون اختصاص بدیم تعداد اتصال‌ها از مرتبه‌ی صد میلیون به توان دو یا ده گیگ می‌شه که این روزها عدد بزرگی به‌حساب نمی‌یاد ولی با این حال شبیه‌سازی چنین شبکه‌ای روی کامپیوترهای شخصی هنوز راحت نیست. به‌هرحال همان تصاویر صد در صد پیکسل برای منظور ما کافی است و می تونیم با روشن و خاموش کردن پیکسلها یک تصویر سیاه و سفید نسبتاً خوب از این حروف بسازیم.

حالا می‌خواهیم یک شبکه ی هاپفیلدی درست کنیم که اولاً به تعداد پیکسلها نورون داشته باشد (در این حالت می‌شه ده هزار نورون) ثانیاً با استفاده از قانون هب اتصال‌هایش را طوری تغییر دهیم که این حروف در حافظه‌ی شبکه ذخیره بشه (توجه کنید که در این حالت ما نزدیک صد میلیون (۱۰۰۰۰×۱۰۰۰۰) اتصال داریم که باید تغییر کند). حال اگر شما آدم خیلی شکاکی باشید خواهید گفت که این کار غیر ممکنه چون به فرض این که ما اتصالها را طوری تغییر بدیم که حرف A در شبکه ذخیره بشه به محض اینکه بخواهم مثلاً حرف F یا C را ذخیره کنیم دوباره اتصالها تغییر می‌کند و شبکه حرف A را فراموش می‌کنه و همینطور تا آخر. واقعاً حق دارید که شک کنید چون همانطور که قبلاً اشاره کردیم این کار شبیه یک چشم‌بندی است و شبیه این می‌مونه که صفحات هر کتاب یک کتابخانه به‌طور دلخواه با کتاب‌های دیگه عوض بشه ولی هروقت بخواهیم کتابی را جستجو کنیم کل صفحات کتاب از جاهای مختلف کتابخانه جمع آوری بشه و صحیح و کامل به شکل اولیه خودش ظاهر بشه!! باور کنید که اوضاع از این هم عجیب تره! در واقع نه فقط صفحات بلکه کلمات و حروف کتاب‌ها هم در این کتابخانه پخش و پلا شده!!!! یه قضیه در مورد شبکه‌ی هاپفیلد ثابت شده که میگه ظرفیت حافظه در این شبکه‌ها به اندازه ده درصد تعداد نورون‌هاست. یعنی وقتی ده هزار نورون داریم باید بتونیم در حدود هزارتا الگوی حافظه مثل A,F,C و یا هرچیز دیگری مثل تصویر میز و صندلی و درخت، میوه‌های مختلف و ... را در این شبکه ذخیره کنیم. شما الآن حق دارید شک کنید که چطور این شبکه می‌تونه دو حرف A و F را به‌طور همزمان در خودش نگه داره. اگر فعلاً حوصله شبیه‌سازی کامپیوتری ندارید، می‌تونید با مداد و کاغذ چند تا دست گرمی بزنید. مثلاً یک شبکه با ۱۰ نورون انتخاب کنید طوری که دو الگوی زیر را با کمک قانون هب در خودش ذخیره کنه:

$$P_1 = (1 , 1 , -1 , 1 , -1 , -1 , 1 , -1 , 1 , 1)$$

$$P_2 = (-1 , -1 , 1 , -1 , 1 , 1 , 1 , -1 , -1 , -1)$$

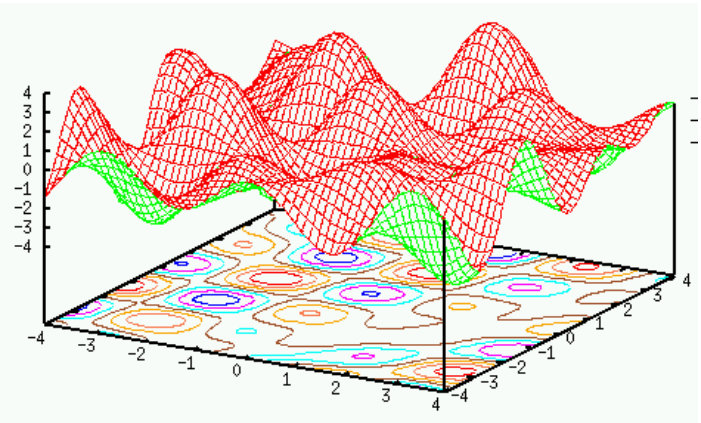
البته چون ده درصد عدد ده میشه یک، شاید موفق نشید! شاید هم موفق بشید! چون گفتیم تقریباً ده درصد (البته بر اساس یک قضیه دقیق فیزیک آماری برای شبکه‌های خیلی بزرگ این ده درصد به عدد جالب سیزده درصد نزدیک می‌شه). هنوز خوب توضیح ندادیم که وقتی می‌گیم دوالگوی P_1 و P_2 در شبکه ذخیره شده یعنی چه؟ ولی با صحبت‌هایی که قبلاً کردیم کم‌کم باید حدس بزنید که منظور ما چیه؟ باید یک شبکه‌ای با ده نورون درست کنیم به‌طوری که وقتی P_1 را به شبکه می‌دیم مقدار شبکه در همین نقطه ثابت بمونه و برای P_2 هم همینطور! از همه مهم تر انتظار داریم که وقتی یک الگوی نزدیک به شبکه بدیم، مثلاً الگویی که فقط در دوجا با P_1 متفاوت باشد بعد از چند لحظه دوباره به همان نقطه‌ی P_1 برسیم. حالا اتصالها را چطور باید تغییر بدیم؟ به جایگاه اول و پنجم نگاه کنید

چون هم علامت نیستند بر طبق قانون هب یک واحد از اتصالشان کم می‌شود (فرض کنید در لحظه ی شروع تمام اتصالات گراف صفر باشه. با اولین بار که فرایند رو اجرا کنیم وزن‌های جدید در سیستم تعریف خواهد شد و همینطور الی آخر...) و چون در مورد P_2 نیز اوضاع به همین شکل است یک واحد دیگه از اتصال نورون اول و پنجم کم می‌کنیم (می‌تونیم بگیم پنجم و اول، فرقی نداره). پس اگر در ابتدا و قبل از مرحله‌ی یادگیری اتصال این دو نورون یک و پنج برابر صفر بوده باشد، بعد از یک دور تکرار وزن این اتصال برابر ۲- واحد می‌شه. این کار را می‌تونیم برای همه نورون‌ها انجام بدیم و در عرض چند دقیقه کل وزن‌های شبکه رو که صدتا بیشتر نیست بدست بیاریم! (توجه کنید که وزن‌ها تقارن دارند، یعنی مثلاً اتصال نورون سه و دو با دو و سه برابر خواهد شد و چون فرض می‌کنیم هیچ نورونی با خودش اتصال ندارد فقط باید در مجموع ۴۰ اتصال را محاسبه کنیم). چون ده نورون بیشتر نداریم این شبکه در مجموع $۲^{۱۰}=۱۰۲۴$ حالت بیشتر ندارد و اگر شانس داشته باشیم، خیلی از حالت‌ها یا به P_1 میل می‌کند یا به P_2. حالا شاید الگویی مثل P_3 باشد که به اندازه ی کافی از P_1 و P_2 دور باشد و از بدشانسی ما P_3 نیز نقطه ثابت این شبکه باشد (مثل این می‌مونه که آدم توهم زده باشه و خاطره‌ای یادش بیاد که قبلاً تجربه نکرده باشه. تا حالا شده جای جدیدی برید و قسم بخورین که قبلا در آنجا بودین؟ فکر می‌کنین که این پدیده با حالتی که الان صحبت کردیم، یعنی حالت P_3 ارتباطی داره یا نه ؟)

الان شاید هوس کنید که یک شبکه‌ی صد نورونی بسازید و با همین حقه‌ای که یاد گرفتید شش هفت تا الگوی دبش توش ذخیره کنید. ولی خوب باید حداقل پنج‌هزار اتصال رو محاسبه کنید و یک شب تا صبح بیدار بمونید. پس خوبه یکی از دوستانتون که با زبان برنامه‌نویسی آشناست در این بازی شرکت کنه تا بتونید آزمایش‌های جالب‌تری انجام بدین. یادتون باشه که وقتی صد نورون دارید تعداد حالت‌ها یک عدد نجومی است (دو به توان صد!!) و ممکنه حالت های جالبی در شبکه پیدا بشه که اصلاً فکرش را هم نکرده باشید. خوشبختانه یک قضیه جالب وجود داره که می‌گه از هر حالتی شروع کنید دیر یا زود به یک نقطه ثابت می‌رسید. در غیر این‌صورت ممکن بود مجبور باشید تا آخر الزمان صبر کنید و به نقطه‌ی ثابتی نرسید! (این اتفاقاتی است که ممکنه برای یک شبکه آشوبناک پیش بیاد ولی درمورد شبکه‌ی هاپفیلد چنین اتفاقی نخواهد افتاد. به بچه‌هایی که به ریاضی و فیزیک علاقه دارند راهش رو یاد می‌دیم که در کمتر از چند خط این مسأله‌ی جالب را برای خودشون اثبات کنند). جالب این بود که آقای هاپفیلد همه را قانع کرد که چنین سیستمی شبیه یک پدیده‌ی فیزیکی است که همیشه در جهتی حرکت می‌کند که مقدار انرژی خودش رو به کمترین مقدار برسونه. مثل این می‌مونه که همیشه داریم سرازیری حرکت می‌کنیم و گاهی اوقات سر

راه به دره‌ای برخورد می‌کنیم و مجبوریم تا ته دره سرازیری بریم و وقتی به ته دره رسیدیم چون امکان بالا رفتن نداریم (طبق قواعد فیزیکی) همانجا ثابت می‌مانیم.

تصویر جالبی که باید در ذهن شما نقش بسته باشه اینه که حافظه‌های ذخیره شده در واقع نقاط ته دره هستند. به زبان فیزیکی به این نقاط می‌گویند می‌نیمم‌های تابع انرژی سیستم. شکل زیر نمونه ای از یک تابع انرژی و دره‌هایی است که جای ذخیره حافظه است (حالا اگر از دره خوشتون نمیاد می تونید هر تصویری در ذهنتون دارید رو در یک عدد منفی ضرب کنید و به بلندیها و قله‌های کوه فکر کنید). به‌هر حال یادتون باشه که تعداد زیادی دره‌های ناخواسته هم داریم یعنی آموزه‌هایی که قصد نداشتیم در سیستم ذخیره کنیم. حالا ممکنه این‌ها توهم نباشه و اسمشون رو بگذاریم خلاقیت ذهنی! البته اگر خیلی اصرار داشته باشید که در این دره‌های ناخواسته گیر نکنید می‌تونیم راه‌های جالبی به شما پیشنهاد کنیم که اتفاقاً تعبیر فیزیکی هم دارد. ولی قبل از این که راه سربالایی رفتن را به شما یاد بدیم اول باید ببینیم که این تعداد ناچیز حافظه (یعنی ده درصد کل تعداد نورون‌ها) ارزش این‌همه دردسر کشیدن رو داره یا نه؟ بله، اگر فقط صد یا هزارتا نورون داشته باشیم ده درصد آن که می‌شه ده یا صد قلم حافظه اصلاً به‌چشم نمیاد. درسته که شبیه‌سازی همین تعداد هم کلی به آدم چیز یاد میده ولی بدرد مدل‌سازی یک حافظه‌ی واقعی نمی‌خوره. ولی اگر این تعداد چند میلیون و یا چند میلیارد باشه کم کم می‌شه حرف های جالبی زد. دقت کنید که هر قلم حافظه در چنین سیستمی صدها و هزارها قلم حافظه‌ی نزدیک به آن را هم پوشش می‌ده.



گودی‌های (مینیمم) نمودار کمترین انرژی را دارند و میشه معادل خاطره حسابشون کرد

به نظرتون برای سیستمی که نمودار انرژیش اینه، چند تا خاطره توش ذخیره شده؟

مثلاً اگر ما تصویر یک صندلی یا درخت نوعی رو در حافظه‌ی خود ذخیره کنیم مثل این می‌مونه که هزارها وسیله‌ی دیگه که شبیه صندلی یا درخت هستند هم در حافظه ما ذخیره شده و هروقت به این اشیاء نگاه کنیم آنها را صندلی یا درخت می‌بینیم. پس هر قلم حافظه در چنین سیستمی دسته‌ی بزرگی را تشکیل می‌ده که نماینده همه چیزهای شبیه به آن است. حالا شاید قانع شوید که اگر بتونیم در یک سیستم صد میلیارد نورونی در حدود صد میلیون قلم حافظه داشته باشیم اصلاً عدد کمی نیست. حتی می‌تونیم به ده میلیون تا یعنی یک درصد کل نورونها هم راضی بشیم!! یکی از نواحی مغز به اسم هیپوکامپ، نقش مهمی در به خاطرسپاری تجربیات روزمره داره به‌طوری که افرادی که این ناحیه‌شان آسیب جدی دیده باشه ممکنه فقط حافظه چند ساعت قبل را به یاد بیاورند. اخیرا هیپوکامپ به شدت مورد توجه پژوهشگران قرار گرفته و تحقیقات زیادی روش انجام می‌شه. در یک مورد خاص، بیمار جالبی به اسم HM که در موردش هم یه فیلم درست شده، برای نزدیک ۴۰ سال هر روز که دکترش را می دید از او می‌خواست خودش را معرفی کنه. این بیمار تقریباً همه حافظه‌ی دراز مدت خود را از دست داده بود و به قولی فقط در زمان حال زندگی می کرد. بیماری آلزایمر هم که این روزها خیلی از آن صحبت می‌شه ارتباط زیادی با آسیب‌دیدگی ناحیه هیپوکامپ داره. جالب است بدونید که یکی از مدل‌هایی که در مورد هیپوکامپ خیلی روی آن کار شده مدل هاپفیلد است. شما می‌تونید با دوستانتون که اطلاعات زیستی مناسبی دارند، مطالعات بیشتری در مورد این ناحیه‌ی جالب مغز انجام بدین و با نحوه‌ی مدل‌سازی آن آشنا بشید. البته چون این ناحیه در انسان نزدیک به چند صد میلیارد نورون داره، هیچ‌کس توقع نداره که به این زودی‌ها یک مدل واقعی از هیپوکامپ انسان (و یا حتی هیپوکامپ موش) داشته باشه.

همانطور که گفتیم یکی از جذابیت‌های شبکه‌ی هاپفیلد برای فیزیکدان‌ها اینه که می‌تونن به این شبکه انرژی نسبت بدن که پر از پستی و بلندی است. بعضی از دره‌ها ممکنه خیلی عمیق باشه و دیگه نشه از اون‌ها بیرون پرید. اما از بعضی دره‌ها که عمق کمتری دارند می‌تونیم ازشون خارج بشیم. یک راهش اینه که سیستم را کمی داغ کنیم! برای این که تصویر خوبی داشته باشید فرض می‌کنیم یک ماهی‌تابه دارید که ته اش به جای این‌که صاف باشد مقدار زیادی پستی و بلندی داره و مقدار ذرت در این ظرف ریختید و روی آتش گذاشتید. حالا هرچقدر شعله‌ی آتش رو زیادتر کنید ذرت‌هایی که در ماهی‌تابه هستند بیشتر به هوا می‌پرند و از یک گودی خارج می‌شوند و احتمالاً در گودی دیگری داخل می‌شوند. این همان کاری است که شما می‌تونید با حافظه‌های ذخیره شده در شبکه هاپفیلد انجام بدین. به این‌صورت که اولاً به جای این که نورون‌ها به‌طور قطعی در حالت خاموش و روشن قرار بگیرند یعنی وقتی بیشتر تقویت شدند حتماً روشن بشن و برعکس وقتی بیشتر تضعیف شدند حتماً خاموش بشن! حالا با یک احتمال تصمیم

می‌گیرند که خاموش یا روشن بشن. البته اگر یک نورون خیلی داره تقویت می‌شه عاقلانه نیست که خودش رو خاموش نگه داره ولی اگر هنوز مقدار کمی تقویت می‌شه این احتمال وجود داره که خودش رو خاموش نگه داره. در این احتمال یک عامل دخالت داره که همان دماست. هرچقدر این دما را زیادتر کنیم بیشتر احتمال داره که نورون‌ها علیرغم میل باطنی خودشون روشن یا خاموش بمانند. در بخش بعدی که می‌خواهیم یک پروژه برای شما تعریف کنیم این مطالب را دقیق‌تر بیان می‌کنیم.

پروژه ها

در این پروژه شما می‌توانید چند آزمایش خوب با مدل هاپفیلد انجام بدین و اگر کمی ابتکار به خرج بدین حتما نتایج جالب و تازه‌ای در مورد این‌که حافظه‌ی شما چطور کار می‌کند به‌دست می‌آورید. خوشبختانه مدل هاپفیلد به‌قدری ساده است که با چند خط کدنویسی می‌توانید همه‌ی رفتارهای جالبی که در این بخش خواندید را مشاهده کنید. ولی اول باید با چند نماد ریاضی آشنا بشید تا توضیح مسئله آسون‌تر بشه. مثلا الگوهایی که می‌خواهید در شبکه ذخیره کنید رو با نماد $P^1, P^2, ..., P^m$ (البته قصد آزار نداریم اما نوشتن شماره الگو در جای توان دلیلی داره که بعد بهش اشاره خواهیم کرد!) نشون می‌دیم. هر یک از این الگوها بسته به اینکه تعداد نورون‌های شبکه چندتا باشه به‌صورت یک رشته از اعداد ۱ و ۱− نوشته می‌شود. همان‌طور که قبلا دیدیم اگر شبکه‌ی ما مثلا ۱۰ تا نورون داشته باشد، الگوهای ما رشته‌های ده‌تایی از ۱ و ۱− هستند. (در این نماد P مخفف pattern یعنی همان الگو است ولی اگر از این نماد خوشتون نمیاد می‌تونید از هر نماد دیگه‌ای که دوست دارید استفاده کنید، بعضی نیز از نماد ξ استفاده می‌کنند که شبیه کرم است و یونانی‌ها بهش «اگزی» می‌گن و تازه شماره‌ی الگو را هم در بالاش قرار می‌دهند و مثلا الگوی شماره‌ی ۵ را اینطور نشون میدن ξ^5 ! خوشتان آمد؟)

همان‌طور که دیدیم مقدار نورون در شبکه‌ی هاپفیلد در هر لحظه یا یک یا منهای یکه و آن را با نماد S (مخفف Spin در فیزیک ذرات) نشون میدن و مثلا اینکه نورون شماره ۵ در لحظه‌ی t مقدارش منهای یکه این‌طور نوشته می‌شه: $S_5(t) = -1$ و یا به طور خلاصه‌تر می‌نویسند $S_5 = -1$. این در حالتی است که خودمونیم می‌دونیم در چه لحظه‌ای هستیم و دیگه لازم نیست از حرف t برای مشخص کردن زمان استفاده کنیم. (البته ریاضی‌دان‌ها خیلی با نماد S احساس راحتی نمی‌کنند و ترجیح میدن از نماد آشنای X استفاده کنند، ولی ما اینجا به احترام آقای هاپفیلد از همان نماد S استفاده می‌کنیم). خوشبختانه در انتخاب نماد t برای زمان اختلافی نظری وجود نداره! ولی مواظب باشید که برنامه‌نویسان حرفه‌ای معمولا زمان را با نماد s کوچک (مخفف step به معنای قدم) نشون می‌دن و اگه کسی قصد آزار و اذیت داشته باشه!! می‌تونه وقتی از S بزرگ برای نمایش فعالیت نورون استفاده می‌کنیم از s کوچک برای زمان استفاده کنه!

به هر صورت اگه شما مفهوم رو درست بلد باشید نباید از انتخاب نمادهای عجیب که براتون آشنا نیست هراسی داشته باشید، مثلا در این پروژه برای نشون دادن اتصال نورون‌ها از نماد W (مخفف Weight یعنی وزن) استفاده می‌کنیم.

وقتی می‌خواهیم بگیم اتصال نورون پنجم و دوم برابر با $\frac{1}{4}$ است، می‌نویسیم: $W_{25} = \frac{1}{4}$ و البته در مدل هاپفیلد چون اتصال‌ها متقارنند، اتصال نورون دوم و پنجم هم برابر با $\frac{1}{4}$ است. یعنی $W_{52} = \frac{1}{4}$. تقارن یکی از مهم‌ترین خاصیت‌های شبکه‌ی هاپفیلد است. (مدل‌هایی که این خاصیت را ندارند رفتارهای پیچیده‌تری دارند مثل رفتار تناوبی که در متن به آن اشاره کردیم و بعدها با آن آشنا خواهید شد) و همین تقارن تضمین می‌کنه که شبکه دیر یا زود به تعادل می‌رسه و می‌شه از این شبکه به عنوان یک سیستم ذخیره‌ی حافظه استفاده کرد. پس در این مدل فرقی نداره که بگیم اتصال نورون پنجم و دوم و یا دوم و پنجم (اگه حوصله داشته باشید می‌تونید به‌طور دست گرمی با چند مدل ساده که این شرط را ندارند چندتا آزمایش انجام بدین). حالا می‌رسیم به مهم‌ترین قسمت کار، یعنی نشون دادن قانون هب به کمک همین نمادهایی که معرفی کردیم. فقط این یادتون باشه که از اینجا به بعد به‌جای اسم بردن نورون‌ها از لفظ کلی‌تر نورون i ام یا j ام استفاده کنیم و مثلا می‌گیم که وقتی الگوی P رو به شبکه می‌دیم چطور باید اتصال نورون iام و j ام رو تغییر بدیم. اینجا شاید ساده‌تر باشه که یک‌بار دیگه قانون هب رو در ذهنمون مرور کنیم و بعد به فکر یک روش ساده برای نشون دادن این قانون باشیم. کاری که می‌کنیم اینه که اول یک الگو انتخاب می‌کنیم و بعد به جایگاه i ام و j ام این الگو نگاه می‌کنیم. اگر این دو جایگاه هم‌علامت بودند (یعنی هردو ۱+ یا هر دو ۱−) به اندازه‌ی یک واحد به اتصال نورون i ام و j ام اضافه می‌کنیم، و اگر نبودند یک واحد از این اتصال کم می‌کنیم، این کار را برای تک‌تک الگوهای P^1 تا P^m را انجام می‌دیم، حاصلش مقداری می‌شه که برای اتصال نورون j و i داریم (یعنی W_{ij}) همین‌طور که می‌بینیم با این روش خواهیم داشت $W_{ij} = W_{ji}$. اگر همین رو به یک برنامه‌نویس مبتدی توضیح بدین به سرعت برای شما کدی می‌نویسه که با هر تعداد نورون والگوی داده شده بتونید اتصال‌ها را به‌دست بیارید. اما نوشتن این مطلب ساده به زبان ریاضی کمی دردسر داره، چون هم باید شماره‌ی الگوها رو داشته باشید (مثلا P^4 یا P^5) و هم شماره‌ی جایگاه مربوط به هر الگویی که در لحظه‌ی فعلی نگاه می‌کنید. شاید دوست نداشته باشید از چنین نمادی استفاده کنید، ولی چاره‌ای جز تسلیم ندارید و باید شماره‌ی الگوها را در بالا و به شکل توانی بنویسید یعنی P^1، P^2 ،... ، P^m (البته این اصلا معنی توان نمی‌ده و فقط از روی ناچاری است).

حالا اگر بخواهید وزن اتصال نورون سوم و چهارم را تغییر بدین، به جایگاه سوم و چهارم P^1 نگاه می‌کنید. اگر در هر دو جایگاه هم علامت باشند(مثلا هردو ۱+ یا ۱− باشند) یک واحد به W_{34} اضافه می‌کنیم و اگر علامت آنها متفاوت باشد(یکی ۱+ و دیگری۱− باشد)یک واحد از W_{34} کم می‌کنیم و همین کار را برای الگوهای P^2 تا P^m ادامه می‌دهیم و نتیجه را با هم جمع می کنیم تا مقدار نهایی W_{34} به دست آید.

حالا یک کمی قیافه‌ی متفکرانه به خودتون بگیرید. همین حرف‌هایی که زدیم (قانون هب) را به صورتی که در زیر نوشته شده بررسی کنید:

$$W_{ij} = P\,{}_i^1 P_j^1 + P\,{}_i^2 P_j^2 + \ \ldots + P\,{}_i^m P_j^m$$

منظور از $P^3{}_i$ و $P\,{}^3{}_j$ روشنه یعنی جایگاه iام و j ام در الگوی سوم. حالا چرا ما این‌ها را در هم ضرب کردیم؟ کمی فکر کنید حتما خودتون متوجه می‌شید!

یه موضوع مهم دیگه هم هست که حتما باید بهش توجه کنیم. برای اینکه اتصال نورون‌ها زیادی بزرگ نشه و خدای نکرده مغزمون متلاشی نشه، حاصل به‌دست آمده رو معمولا به کل تعداد نورون‌ها که با N نشون می‌دیم، تقسیم می‌کنیم. که می‌شه:

$$W_{ij} = \frac{1}{N}\left(P\,{}_i^1 P_j^1 + P\,{}_i^2 P_j^2 + \ \ldots + P\,{}_i^m P_j^m \right)$$

اگه شما هم مثل بقیه‌ی دانشمندان موافق صرفه‌جوئی در کاغذ باشید می‌تونید با کمک یک نماد جالب ریاضی که به شکل Σ نوشته می‌شه حاصل بالا را به این شکل زیبا نشون بدین:

$$W_{ij} = \frac{1}{N}\sum_{l=1}^{m} p_i^l p_j^l$$

(اگه این شکل به نظرتون زیبا نیست، فورا برگردید به همان تعریف اولیه‌ی قانون هب، به راحتی می‌تونید این نحوه‌ی نمایش رو درک کنید) به این ترتیب شما می‌تونید براساس یک‌سری الگوی که توضیح داد شد و استفاده از قانون هب یک شبکه‌ی هاپفیلد بسازید و حالا باید آزمایش کنید که آیا این شبکه کاری رو که باید انجام بده (ذخیره‌ی الگو) رو درست انجام میده یا نه؟ برای این کار یکی از الگوها رو در لحظه‌ی فعلی به شبکه میدین و بعد مشاهده می‌کنید که نورون‌ها در لحظه‌ی بعد چطور تغییر می‌کنن و اگه دوباره همون الگوی قبلی را مشاهده کردید نفس راحت می‌کشید که تا اینجای کار درست است. حالا مثلا این الگو را با یک تغییر کوچک به شبکه می‌دین و باز انتظار می‌کشید که شبکه برای چند لحظه کار خودش رو انجام بده و نورون‌ها از یک حالت به حالت دیگه تغییر کنن و اگه دوباره به همان الگوی اول رسیدید حالا یک هورا می‌کشید و این خبر خوب رو به دوستان خودتون اطلاع می‌دین. ولی خیلی عجله نکنید چون باید این کار رو برای همه‌ی الگوهای ذخیره شده انجام بدین و اینجا شاید همیشه خوش‌شانس نباشید! بالاخره این کار ارزش زیاد امتحان کردن رو داره، به‌خصوص اگه این الگوها مربوط به تصاویر چهره‌هایی باشه که دوستشون دارید.

با اینکه فکر می‌کنیم الان شما خوب میدونید که چطور در هر زمان وضعیت نورون‌ها رو محاسبه کنید، ولی برای دقیق شدن مسأله لازم است که مراحل کار رو کاملا مشخص کنیم!

مرحله‌ی اول: در هر لحظه مجموع اطلاعات دریافتی یا ورودی به نورون i ام را طبق رابطه‌ی زیر محاسبه کنید:

$$\text{ام i = مجموع ورودی‌ها به نورون } W_{i1}S_1 + W_{i2}S_2 + \ldots + W_{iN}S_N$$

که باز برای صرفه‌جوئی در مصرف کاغذ سمت چپ عبارت را با نماد h نشون می‌دیم و از علامت Σ برای جمع بستن مقادیر سمت راست استفاده می‌کنیم.

$$h_i = \sum_{j=0}^{n} W_{ij}S_j$$

مرحله‌ی دوم: به علامت h_i نگاه می‌کنیم، اگر علامت مثبت شد وضعیت نورون i ام ۱+ می‌شه و اگه علامت h_i منفی شه، وضعیت نورون i ام ۱− می‌شه و اگر صفر شد همون طور که در متن اشاره شد قرار داد می‌کنیم که وضعیت نورون ۱+ بشه. همه‌ی این حرف‌ها رو می‌شه به‌صورت زیر خلاصه کرد:

$$S_i \text{ در هر لحظه} = \begin{cases} +1 & h_i \geq 0 \\ -1 & h_i < 0 \end{cases}$$

حالا شما همه‌ی مراحل کار با شبکه‌ی هاپفیلد رو یاد گرفتید. دو مرحله دارید که شبکه رو بسازید، یعنی انتخاب الگوها و محاسبه‌ی اتصال‌ها براساس قانون هب. دو مرحله هم داریم برای این که به اصطلاح استارت سیستم را بزنید و ببینید که نورون‌ها چطور تغییر وضعیت میدن؟ در اینجا چند توصیه به شما می‌کنیم که شاید در آزمایش‌های اولیه به شما کمک کنه. اول برای دست گرمی چند مدل خیلی کوچک در حد سه یا چهار نورون در نظر بگیرید و همه‌ی مراحل کار رو با مداد و کاغذ محاسبه کنید. وقتی احساس کردید تا حدود خوبی بر اوضاع مسلط هستید دست به یک شبیه‌سازی ساده بزنید. مثلا یک شبکه‌ی صد نورونی را در نظر بگیرید و برای آنکه در روزهای اول از نتایج به‌دست آمده خیلی دلسرد نشوید حداکثر دو یا سه الگوی حافظه بیشتر انتخاب نکنید و با همین تعداد کم شروع به آزمایش کنید. حالا سؤالات زیر رو با دقت بررسی کنید و پروژه‌ی خود را کامل کنید:

۱–شبکه‌ی هاپفیلد را در دو حالت آزمایش کنید. حالت اول همه‌ی نورون‌ها را به‌طور همزمان تغییر وضعیت بدین و در حالت دوم، یک نورون رو به‌طور تصادفی انتخاب کنید و تغییر وضعیت بدین و بعد نورون دیگه‌ای انتخاب کنید همین طور الی آخر. (توجه کنید که در این حالت وقتی می‌خواهید وضعیت نورن بعدی را مشخص کنید باید وضعیت جدید نورون قبلی را در رابطه‌ی (۱) لحاظ کنید). آیا تفاوتی در نتیجه‌ی کار مشاهده می‌کنید یا نه؟

۲– معمولا الگوها را در مدل هاپفیلد طوری انتخاب می‌کنند که خیلی شبیه به هم نباشند. در اینجا حالتی را آزمایش کنید که الگوها خیلی شبیه به هم باشند و نیتجه‌ی آزمایش را گزارش کنید. آیا توضیح جالبی دارید که چرا الگوها را تا حد امکان مستقل از هم می‌گیرند.

۳– درصد کمی از اتصالات به‌دست آمده را صفر کنید. مثلا سه تا پنج درصد. (یا اصطلاحا یالهای گراف را پاره کنید) و دوباره شبکه را با همان الگوهای قبلی آزمایش کنید. اگر انتظارمون درست باشه نتایج نباید فرق زیادی کنه. در مورد مغز هم وقتی آسیب‌ها جزئی باشه انتظار نداریم همه‌ی حافظه از دست برود. برای کسانی که واژه‌ی هولگرام به گوششان خورده باید بگیم که این نوع حافظه شبیه به ذخیره‌ی اطلاعات در تصاویر هولگرام است، که اگر قسمت‌هایی اطلاعات از بین بره باز هم تصاویر اولیه قابل بازسازی است. در موردتصاویر معمولی این‌طور نیست و اگر قسمتی از تصویر را از بین ببرید دسترسی شما به آن اطلاعات برای همیشه از بین می‌رود. مثل این می‌مونه که شما روی یک عکس کاغذی خانوادگی خراشی ایجاد کنین و مثلا چهره‌ی برادرتون دیده نشه. اما شما می‌تونید این خراش رو در شبکه‌ی هاپفیلد ایجاد کنید (مثل پاره کردن بعضی از اتصال‌ها) و هنوز هم با کمال تعجب مشاهده کنید که تصویر برادرتون مثل روز اولش قابل بازیابی است!!!

۴– اگر آزمایش‌های بالا رو خوب انجام داده باشید، کم‌کم دارید در شبکه‌های هاپفیلد مهارت پیدا می‌کنید. حالا می‌تونید با اضافه کردن دو سه خط برنامه‌نویسی آزمایش زیر را انجام دهید و نشون بدین که هربار نورونی تغییر وضعیت می‌ده مقدار رابطه‌ی زیر که همان انرژی تعریف شده برای شبکه‌ی هاپفیلد است کاهش پیدا می‌کنه.

$$E = -1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij} S_i S_j$$

البته اگه تعداد نورون‌ها خیلی کم باشه می‌تونید مقدار E را با مداد و کاغذ هم حساب کنید.

نموداری رسم کنید که محور افقی‌اش زمان و محور عمودی‌اش مقدار E باشد. وقتی شبکه را از یک حالت اولیه به‌طور دلخواه آغاز می‌کنید در نقطه‌ی خاصی از صفحه قرار دارید و اگر برنامه‌ی شما درست کار کند این نقطه همواره به شکل نزولی حرکت می‌کند و زمانی که به یکی از الگوهای ذخیره شده می‌رسد باید در جای خود ثابت بمونه. اگه شبکه‌ی شما نسبتا بزرگ باشه حتما به نقاط ثابتی می‌رسید که مربوط به هیچ‌کدام از الگوهای حافظه‌ی پیش‌بینی شده نیست و همان‌طور که قبلا گفتیم به این‌ها حافظه‌های ناخواسته می‌گویند. آیا می‌تونید ویژگی بعضی از این نقاط ثابت ناخواسته را بیان کنید؟ در آزمایش بعدی روشی یاد می‌گیرید که چطوری این نقاط رو از بین ببرید.

۵- مدل‌هایی که تابه‌حال ازشون صحبت کردیم همه مدل‌های قطعی بودند، یعنی ما با قطعیت کامل می‌تونیم وضعیت بعدی سیستم رو با دونستن وضعیت فعلی تعیین کنیم. همان‌طور که قبلا اشاره کردیم ما می‌تونیم به زبان احتمالات صحبت کنیم و بگیم با توجه به میزان ورودی هر نورون احتمال اینکه وضعیت نورون در لحظه‌ی بعد ۱+ شود چقدر است. برای این کار اول مقدار h را به‌دست می‌آورند و به‌جای این که فقط به علامتش نگاه کنند، اول آن را در یک تابع خوش‌رفتاری مثل $P(S_i) = \frac{1}{1+e^{-h}}$ قرار می‌دهند. از شکل‌اش پیداست که h هرچه بزرگ‌تر و مثبت باشد مقدار تابع $P(S_i)$ به عدد یک نزدیک‌تر می‌شه و هرچه بزرگ‌تر و منفی باشه $P(S_i)$ به عدد صفر نزدیک می‌شه و برای یک h متوسط این تابع مقداری بین صفر و یک اختیار می‌کند ($0 < P(S_i) < 1$)، و بنابراین می‌تونیم احتمال تغییر وضعیت نورون به ۱+ را این‌طور حساب کنیم و بگیم با احتمال $P(S_i) = \frac{1}{1+e^{-hi}}$، در مرحله بعدی $S_i = +1$ می‌شه (به زبان ریاضی این رو این‌طور می‌نویسند: $(P(S_i = +1) = \frac{1}{1+e^{-hi}})$ حالا می‌تونید یک پارامتری مثل دما (T) هم به این مدل اضافه کنید و نتیجه‌ی آزمایش رو با مقادیر مختلف T بسنجید: $(P(S_i = +1) = \frac{1}{1+e^{-Thi}})$ (یادتون باشه که سیستم رو خیلی داغ نکنید چون آن‌وقت هیچ الگویی حتی الگوی مورد علاقه‌تون در شبکه ثابت نخواهد موند). روش جالبی که مردم استفاده می‌کنند اینکه اول دستگاه رو کمی داغ می‌کنند (T بزرگ) و بعد به‌تدریج دستگاه رو خنک می‌کنند. شما می‌تونید برای خودتون آزمایش‌های جالبی طرح کنید و نتیجه رو اعلام کنید. شاید هم به یافته‌های جدیدی برسید که بقیه هنوز کشف نکردند. البته ما هنوز خیلی چیزها در مورد شبکه‌ی هاپفیلد به شما نگفتیم، ولی اگر تا اینجا مراحل رو با موفقیت طی کرده باشید حتما خودتون بهتر از ما کار رو ادامه خواهید داد.

در ادامه با دو پروژه‌ی هیجان انگیز اما اختیاری آشنا می‌شید و اگه از اون‌ها خوشتون اومد می‌تونین به این مسائل هم فکر کنید!

۶- یک مسأله‌ی حل نشده! (اختیاری)

تا این‌جا در تمام مسائلی که مطرح شد، وزن‌ها متقارن بود ($W_{ij} = W_{ji}$). حالا بزارید ببینیم اگر وزن‌ها متقارن نباشد چه اتفاقی می‌افته! هیچ تضمینی نیست که شبکه به نقطه‌ی پایدار برسه و برعکس حتی ممکنه تعداد زیادی حالات تناوبی داشته باشه. پیچیده ترین حالت شبکه اینه که ساختار کاملا تصادفی باشه و هیچ نظمی در اتصال بین نورون‌ها مشاهده نشه. حتی ممکنه از یه حالت شروع کنه و از همه‌ی حالت‌ها گذر کنه تا بالاخره برسه به همون حالتی که ازش شروع کرده (تصور کنین فقط صد تا نورون باشه چقدر باید انتظار کشید تا این مسأله معلوم بشه). اما گاهی اوضاع به این پیچیدگی نیست و کمی نظم در ساختار شبکه وجود داره (مثلا در ساختار سیستم عصبی بین دو ناحیه به صورت غیر تصادفی و جهت دار اتصالاتی وجود داره که شبکه رو از حالت تصادفی کامل خارج می‌کنه). سوال سخت اینه که چطور می‌تونیم با دانستن حالت‌های شبکه در یک زمان محدود به نحوه‌ی اتصالات شبکه دست پیدا کنیم . تا به حال الگوریتم های مختلفی پیشنهاد شده که بتونه این مسأله رو در بعضی موارد خاص حل کنه. به این نوع مسأله‌ها، "مسأله‌های وارون" می‌گن. چون به جای اینکه با دونستن وزن اتصالات، بخواهیم حالت‌های بعدی شبکه رو پیدا کنیم(مثل مثال‌های قبل) فرض می‌کنیم حالت ها رو داریم و می‌خواهیم وزن اتصالات رو پیدا کنیم. اگر شما مدعی باشید که الگوریتم خوبی پیدا کردید می‌تونید روی مثال‌هایی که جواب اون معلومه، آن را آزمایش کنید و بعد که مطمئن شدید برای بعضی مثال‌ها خوب جواب می‌ده، الگوریتم خود را به همه معرفی کنین. شاید معروف شدید!

۷- مساله‌ی کمی نامربوط به حافظه!!! (اختیاری)

خیلی از شما حتما مسئله‌ی معروف فروشنده‌ی دوره‌گرد رو شنیدین :

در این سوأل چند تا شهربا فواصل معین داریم و یه فروشنده که باید به همه‌ی این شهرها سرکشی کنه! البته یه سری شروطی هم هست مثل اینکه از هیچ شهری نباید دوباره عبور کنه! و کمترین فاصله ممکن رو هم طی کنه! حل این مسئله وقتی تعداد شهرها زیاده خیلی دشوار می‌شه و قوی‌ترین الگوریتم‌های موجود هم وقتی تعداد شهرها خیلی زیاده جوابگوی حل این مسئله نیست! اگر بخواهیم این مسئله رو با شبکه‌ی هاپفیلد حل کنید، چه خواهید کرد؟ لابد خواهید گفت که به تعداد شهرها نورون انتخاب خواهید کرد و اتصال بین شهر ها رو هم از روی فاصله‌شون تعیین می کنین. محض راهنمایی به شما توصیه می‌کنم که کمی بیشتر ابتکار به خرج بدید و قبل از اینکه به گوگل مراجعه کنین، کمی فکر کنید. شاید تونستید یه راه جدید برای حل مسئله پیدا کنید! این دسته از مسائل تحت عنوان مسائل بهنیه‌سازی(

optimization) از اهمیت زیادی برخورداره و خیلی‌ها امرار معاششان به حل چنین مسائلی بستگی داره! حالا این شما و این هم **معاش**(م. مسابقات .ع. علوم .ا. اعصاب .ش.شناختی)

موفق باشید.