

دانشگاه صنعتی خواجه نصیرالدین طوسی

دانشکده مهندسی برق

پایان نامه مقطع کارشناسی مهندسی برق گرایش الکترونیک

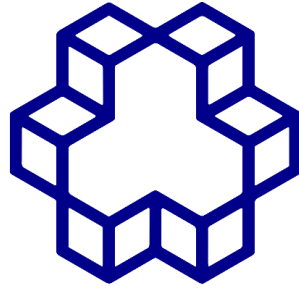
**طراحی و پیاده سازی یک پردازنده نورومورفیک تمام دیجیتال**

دانشجو: امیررضا بهرامنی

استاد راهنما: دکتر امیرمسعود سوداگر

بهمن ۱۴۰۰

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



دانشگاه صنعتی خواجه نصیرالدین طوسی

## تأییدیه هیئت داوران

هیئت داوران پس از مطالعه پایان نامه پیش رو و شرکت در جلسه دفاع از پایان نامه تهیه شده تحت عنوان:

**طراحی و پیاده سازی یک پردازنده نورومورفیک تمام دیجیتال**

نوشته آقای امیررضا بهرامنی صحت و کفایت تحقیق انجام شده را برای اخذ مدرک کارشناسی در رشته مهندسی برق مورد تأیید قرار دادند.

استاد راهنما: آقای دکتر امیرمسعود سوداگر

امضا و تاریخ: ۳۰ / بهمن / ۱۴۰۰

استاد ارزیاب: آقای دکتر مهدی علیاری شوره دلی

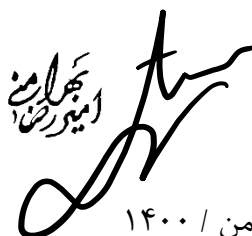
امضا و تاریخ: ۳۰ / بهمن / ۱۴۰۰

## اظهارنامه دانشجو

اینجانب امیررضا بهرامنی دانشجوی مقطع کارشناسی رشته مهندسی برق گواهی می‌نمایم که مطالب ارائه شده در این پروژه با عنوان

### طراحی و پیاده‌سازی یک پردازنده نورومورفیک تمام دیجیتال

با راهنمایی استاد محترم جناب آقای دکتر امیرمسعود سوداگر توسط شخص اینجانب انجام شده است. صحت و اصالت مطالب نوشته شده در این پروژه تأیید می‌شود و در تدوین متن پروژه قالب مصوب دانشگاه را رعایت کرده‌ام.



امضا دانشجو:

تاریخ: ۳۰ / بهمن / ۱۴۰۰

## حق طبع، نشر و مالکیت

۱- حق چاپ و تکثیر این پروژه متعلق به نویسنده و استاد راهنمای آن است. هرگونه تصویربرداری از کل یا بخشی از پروژه تنها با موافقت نویسنده یا استاد راهنما یا کتابخانه دانشکده‌های مهندسی برق و کامپیوتر دانشگاه صنعتی خواجه‌نصیرالدین طوسی مجاز است.

۲- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی خواجه‌نصیرالدین طوسی است و بدون اجازه کتبی دانشگاه قابل‌واگذاری به شخص ثالث نیست.

۳- استفاده از اطلاعات و نتایج موجود پروژه بدون ذکر مرجع مجاز نیست.

## تشکر و قدردانی

خدا را شکر بابت خلقت بی‌بدیلش که فرصت تفکر در آن را برای ما بندگان پدید آورده است.

از استاد عزیزم آقای دکتر امیرمسعود سوداگر بسیار سپاسگزارم که در طول یک سال گذشته با راهنمایی‌های فراوانشان بسیار به من آموختند. همچنین از دیگر اساتیدم خصوصاً در دانشگاه خواجه‌نصیرالدین طوسی متشکرم که در طول این سال‌ها مرا با بسیاری از مطالب مهندسی برق و دیگر حوزه‌ها آشنا کردند.

از دوستانم متشکرم که در مراحل مختلف کار با نظر دادن و تشویق‌هایشان مرا یاری نموده‌اند.

در نهایت از خانواده عزیزم تشکر می‌کنم که همیشه از من و علایقم پشتیبانی کرده‌اند.

## چکیده

ساخت پردازنده‌ای که قدرتمندتر و کم‌مصرف‌تر از پردازنده‌های امروزی باشد کار چالش برانگیزی است که نیاز به پیشرفت در حوزه فناوری ساخت مدارهای مجتمع دارد. این چالش برانگیز بودن از آن رو است که به باور بسیاری فناوری به مرز کوچک کردن ترانزیستورها بر روی یک تراشه رسیده است؛ بنابراین، ارائه راه‌های جایگزین برای بهبود عملکرد پردازنده‌های الکترونیکی امری ضروری و اجتناب‌ناپذیر به نظر می‌رسد.

مغز، هوشمندترین پردازنده‌ای است که انسان تاکنون به آن برخورد کرده است. پس اگر از ساختار مغز الهام گرفته و با آن یک پردازنده بسازیم، شاید بتوانیم راه‌حلی برای چالش فوق ارائه دهیم. به آن دسته از معماری‌ها که ساختار و عملکردهای مغز را تقلید می‌کنند، نورومورفیک می‌گویند. پس هدف ما در این پایان‌نامه، بررسی پردازنده‌های نورومورفیک و طراحی نمونه ساده‌ای از آن‌ها برای انجام یک کاربرد یادگیری ماشینی است. برای این منظور ابتدا ساختار فیزیولوژیک مغز بررسی شده و در ادامه برای پیاده‌سازی پردازنده نورومورفیک خودمان از مدل‌های محاسباتی استفاده کرده‌ایم. در این مسیر مدل‌های مناسب از عناصر سازنده شبکه، نورون و سیناپس، و همچنین الگوریتم یادگیری، ارائه شده و به‌صورت کامل توضیح داده شده‌اند. در ادامه مقایسه‌ای بین شبکه‌های عصبی/اسپایکی و شبکه‌های عصبی مصنوعی ارائه شده است.

پس از آن مزایا و معایب پردازنده‌های نورومورفیک ارائه شده است. پس از آن یک پردازنده نورومورفیک تحت زبان برنامه‌نویسی پایتون پیاده‌سازی شده. این پردازنده برای تشخیص ارقام انگلیسی دست‌نوشته موجود در دادگان MNIST طراحی و پیاده‌سازی شده است. در نهایت، عملکرد این پردازنده مورد ارزیابی قرار گرفته و راه‌های ممکن برای بهبود عملکرد آن پیشنهاد شده است.

**کلیدواژه‌ها:** نورومورفیک، شبکه عصبی اسپایکی، یادگیری ماشینی، علوم اعصاب محاسباتی، تشخیص رقم،

MNIST

## فهرست مطالب

فصل اول: مقدمه‌ای بر هوشمندی طبیعی و مصنوعی .....	۱
۱-۱ مقدمه .....	۱
۲-۱ هوش، ذهن و مغز .....	۱
۳-۱ ساخت سیستم هوشمند .....	۲
۱-۳-۱ منطق .....	۲
۲-۳-۱ معماری .....	۴
۴-۱ آینده پردازش .....	۴
۱-۴-۱ قانون مور .....	۵
۲-۴-۱ راه‌حل‌ها .....	۷
۵-۱ نتیجه‌گیری .....	۸
فصل دوم: مقدمه‌ای بر علوم اعصاب محاسباتی .....	۹
۲-۱ مقدمه .....	۹
۲-۲ علوم اعصاب .....	۹
۱-۲-۲ ساختار نورون .....	۹
۳-۲-۲ سیناپس .....	۱۶
۴-۲-۲ شبکه‌های نورونی .....	۱۸
۵-۲-۲ سطوح بررسی .....	۱۹
۳-۲ مدل‌سازی نورون .....	۲۰
۱-۳-۲ مدل هاجکین - هاکسلی .....	۲۰
۲-۳-۲ مدل ایژیکویچ .....	۲۲
۳-۳-۲ مدل LIF .....	۲۳
۴-۳-۲ مدل پرسپترون .....	۲۴
۴-۲ مدل‌سازی سیناپس .....	۲۵
۱-۴-۲ سیناپس به‌عنوان یک ضریب .....	۲۵



- ۲۵..... ۲-۴-۲ مدلی دقیق تر از سیناپس
- ۲۶..... ۵-۲ شبکه عصبی
- ۲۶..... ۱-۵-۲ شبکه عصبی مصنوعی
- ۲۷..... ۲-۵-۲ شبکه عصبی اسپایکی
- ۲۸..... ۳-۵-۲ برخی دیگر از شبکه‌های عصبی
- ۲۸..... ۶-۲ مدل سازی فرایند یادگیری
- ۲۹..... ۱-۶-۲ انواع یادگیری در یادگیری ماشینی
- ۳۰..... ۲-۶-۲ الگوریتم پس انتشار
- ۳۰..... ۳-۶-۲ یادگیری در مغز
- ۳۰..... ۴-۶-۲ پلاستیسیته وابسته به زمان اسپایک (STDP)
- ۳۱..... ۷-۲ نتیجه گیری
- ۳۲..... فصل سوم: مقدمه‌ای بر محاسبات نورومورفیک**
- ۳۲..... ۱-۳ مقدمه
- ۳۲..... ۲-۳ کارکرد پژوهشی
- ۳۳..... ۳-۳ کارکرد محاسباتی
- ۳۴..... ۴-۳ مغز در برابر کامپیوتر
- ۳۵..... ۵-۳ شباهت به بیولوژی یا اهمیت محاسبه؟
- ۳۶..... ۶-۳ انتخاب مدل‌های مناسب
- ۳۶..... ۱-۶-۳ انتخاب مدل نرون
- ۳۷..... ۲-۶-۳ انتخاب مدل سیناپس
- ۳۸..... ۳-۶-۳ انتخاب مدل شبکه
- ۳۸..... ۳-۶-۴ انتخاب الگوریتم یادگیری
- ۳۹..... ۷-۳ زبان مغز
- ۴۰..... ۸-۳ بسترهای پیاده‌سازی
- ۴۳..... ۹-۳ کاربردها

۴۴	..... نتیجه‌گیری ۱۰-۳
۴۵	..... فصل چهارم: پردازنده نورومورفیک برای تشخیص ارقام دست‌نوشته
۴۵	..... ۱-۴ مقدمه
۴۵	..... ۲-۴ معرفی MNIST
۴۶	..... ۳-۴ توضیح شبکه
۴۶	..... ۱-۳-۴ مدل نورون
۴۷	..... ۲-۳-۴ مدل سیناپس
۴۷	..... ۳-۳-۴ ساختار شبکه
۴۸	..... ۴-۳-۴ الگوریتم یادگیری
۴۹	..... ۵-۳-۴ هم‌ایستایی
۴۹	..... ۴-۴ شیوه کارکرد شبکه
۴۹	..... ۵-۴ بستر پیاده‌سازی
۵۰	..... ۶-۴ نتایج
۵۰	..... ۱-۶-۴ نتایج مقاله
۵۱	..... ۲-۶-۴ نتایج کار انجام شده
۵۶	..... ۷-۴ نتیجه‌گیری
۵۷	..... فصل پنجم: نتیجه‌گیری و پیشنهادها
۵۷	..... ۱-۵ بررسی کار انجام شده
۵۷	..... ۲-۵ درنگ فلسفی
۵۸	..... ۳-۵ راه‌های توسعه در آینده
۶۰	..... پیوست اول: معرفی Google Colab
۶۱	..... پیوست دوم: کدهای پروژه
۶۲	..... مراجع

## فهرست جداول

- جدول ۱: مقایسه کامپیوتر و مغز..... ۳۴
- جدول ۲: مقایسه نتایج گوگل کولب و کامپیوتر شخصی ..... ۵۱
- جدول ۳: مقایسه تعداد کلاس‌ها در لایه دوم و داده‌های آموزشی و تست ..... ۵۵

## فهرست شکل‌ها

- شکل ۱: سه گیت اصلی منطقی ..... ۳
- شکل ۲: نمودار تعداد ترانزیستورها بر حسب سال ..... ۶
- شکل ۳: مدل سیستماتیک یک نورون در حالت کلی ..... ۹
- شکل ۴: ساختار بیولوژیک نورون ..... ۱۰
- شکل ۵: کانال‌های یونی در غشا آکسون نورون ..... ۱۱
- شکل ۶: کانال یونی وابسته به فشار و کشش و کانال یونی وابسته به ولتاژ ..... ۱۲
- شکل ۷: برخی از انواع نورون شامل تک‌قطبی، دوقطبی و چند قطبی ..... ۱۲
- شکل ۸: مراحل ایجاد پتانسیل عمل یا اسپایک ..... ۱۳
- شکل ۹: مراحل مختلف ایجاد پتانسیل عمل و عامل زیستی آن ..... ۱۴
- شکل ۱۰: پاسخ یک نورون به تحریک‌های متفاوت ..... ۱۵
- شکل ۱۱: تصویری از سیناپس الکتریکی ..... ۱۶
- شکل ۱۲: نحوه کار سیناپس شیمیایی ..... ۱۷
- شکل ۱۳: تصویر واقعی از نورون و یک سیناپس که توسط میکروسکوپ الکترونی گرفته شده ..... ۱۷
- شکل ۱۴: سطوح متفاوت بررسی علوم اعصاب ..... ۱۹
- شکل ۱۵: مدار معادل مدل هاجکین-هاکسلی ..... ۲۱
- شکل ۱۶: نتیجه شبیه سازی نورون LIF در پایتون ..... ۲۳
- شکل ۱۷: مدل پرسپترون ..... ۲۴
- شکل ۱۸: مدل مداری سیناپس و پاسخ آن ..... ۲۵
- شکل ۱۹: شبکه عصبی مصنوعی سه لایه ..... ۲۷
- شکل ۲۰: مقایسه شبکه عصبی پیچشی و ساختار کورتکس بینایی ..... ۲۸
- شکل ۲۱: نمودار تغییر وزن‌ها طبق الگوریتم STDP ..... ۳۱
- شکل ۲۲: دقت بایولوژیک مدل‌های نورونی در قیاس به سختی پیاده‌سازی آن‌ها ..... ۳۶
- شکل ۲۳: انواع مدل‌های نورونی استفاده شده در نورومورفیک ذیل خانواده‌شان ..... ۳۷
- شکل ۲۴: برخی از کرنل‌های شبکه‌ای مهم ..... ۳۸
- شکل ۲۵: انواع الگوریتم‌های یادگیری استفاده شده در نورومورفیک ذیل خانواده‌شان ..... ۳۹
- شکل ۲۶: نگاهی به دسته‌بندی بسترهای متفاوت پیاده‌سازی پردازنده‌های نورومورفیک ..... ۴۰
- شکل ۲۷: مدار یک نورون در PSpice ..... ۴۱
- شکل ۲۸: پاسخ مدار شکل ۲۷ (Vinside - Voutside) ..... ۴۲
- شکل ۲۹: برخی کاربردهای مهم نورومورفیک ..... ۴۳

- شکل ۳۰: برخی از داده‌های MNIST ..... ۴۶
- شکل ۳۱: ساختار شبکه عصبی اسپایکی استفاده شده ..... ۴۷
- شکل ۳۲: نتایج مقاله ..... ۵۰
- شکل ۳۳: نمودار وزن‌های لایه اول به دوم ..... ۵۲
- شکل ۳۴: تعداد اسپایک‌های نورون‌های لایه دوم ..... ۵۲
- شکل ۳۵: وزن‌های لایه دوم به سوم و لایه سوم به دوم ..... ۵۳
- شکل ۳۶: ماتریس کانفیوژن ..... ۵۳
- شکل ۳۷: پوروتوتایپ‌های ذخیره شده در وزن‌های لایه اول به دوم ..... ۵۴
- شکل ۳۸: تصویری از محیط گوگل کولب ..... ۶۰

## فصل اول: مقدمه‌ای بر هوشمندی طبیعی و مصنوعی

### ۱-۱ مقدمه

در این فصل ابتدا از رابطه ذهن و مغز، و اینکه چه ارتباطی با هوشمندی ما دارد، خواهیم گفت. در ادامه شیوه‌های انتقال هوشمندی به ماشین‌ها و ایجاد هوش مصنوعی<sup>۱</sup> را بررسی خواهیم کرد و در نهایت درباره آینده پردازش صحبت می‌کنیم.

### ۱-۲ هوش، ذهن و مغز

تنها موجوداتی در عالم که از هوشمندی‌شان مطمئن هستیم فقط خودمان هستیم، ما انسان‌ها! درباره هوشمندی موجودات دیگر از حیوانات گرفته تا ماشین‌ها اختلاف نظرهای فراوانی وجود دارد [1].

اما اساساً شناخت و هوش چیست؟ برای این منظور بهتر است ابتدا تعریفی از ذهن و مغز ارائه دهیم، چراکه می‌دانیم هوشمندی ما بخشی از ذهن ما است و آن نیز - تا حدی یا به صورت کامل - برآمده از مغز ما است.

تعریف ذهن کاری بسیار دشوار و تقریباً غیرممکن است چرا که تمام تجربیات ما (حتی خود تجربه ذهن‌مندی) در قالب ذهن هستند، اما با مسامحه می‌توان گفت که ذهن آن بخشی از فرد است که او را قادر به هوشیار بودن نسبت به تجربه‌هایش و جهان بیرون می‌سازد، تجربه‌هایی از قبیل تفکر، احساسات، توجه و... نکته مهم دیگر ذهن، رابطه آن با خودآگاهی است.

برخلاف ذهن، تعریف مغز بسیار ساده است. اندامی در جمجمه همه ما با مشخصات فیزیکی مشخص<sup>۲</sup>.

همان‌طور که پیش‌تر اشاره شد درباره رابطه ذهن با مغز یا به صورت کلی بدن، اختلافات بسیاری وجود دارد که در فلسفه ذهن<sup>۳</sup> به مسئله ذهن - بدن<sup>۴</sup> معروف است. در حالت کلی دو دیدگاه درباره رابطه بین ذهن و مغز وجود دارد، (۱) این دو، جوهرهایی کاملاً متفاوت و مجزا هستند [2]، (۲) ذهن همان مغز است [3]. البته که دیدگاه‌ها بسیار بیشتر از این دو هستند اما این نوشته محل بحث این موضوع نیست.

<sup>۱</sup> Artificial Intelligence

<sup>۲</sup> در اینجا منظور از مغز، دستگاه عصبی می‌باشد. دستگاه عصبی = مغز (مخ+مخچه+ساقه مغز) + نخاع + اعصاب محیطی

<sup>۳</sup> Philosophy of Mind

<sup>۴</sup> Mind-body problem

نکته‌ای که در اینجا برای ما حائز اهمیت است و باتوجه به یافته‌های اخیر در علوم اعصاب<sup>۱</sup> درباره صحت آن شکی وجود ندارد، این است که بخشی از ذهن ما - شامل بخشی از رفتار و هوش ما - برآمده از فعالیت‌های مغزمان است.

### ۱-۳ ساخت سیستم هوشمند

یکی از اهداف مهمی که انسان سال‌ها آن را دنبال کرده ساخت سیستم هوشمند<sup>۲</sup> است. مزایای این سیستم‌ها بسیارند که نیازی به بیان آن نیست، کافی است به اطراف خود نگاهی بیندازید! مشخصاً این سیستم‌ها مانند کامپیوترها، وسایل نقلیه، شبکه اینترنت و... زیست ما را آسان‌تر کرده و ما را قادر به انجام کارهایی کرده‌اند که هیچ‌وقت تصورش را هم نمی‌کردیم. (البته می‌توان درباره معایب این سیستم‌ها و آسیب‌هایشان به ما نیز بسیار بررسی کرد که موضوع این نوشته نیست)

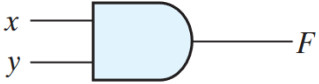
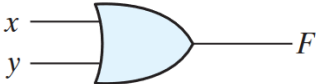
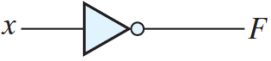
بنابراین طبیعی است که در حدود یک قرن اخیر توان فراوانی بر روی ساخت و بهبود سیستم‌های هوشمند گذاشته شده و بخش اصلی بسیاری از رشته‌های تحصیلی فنی مهندسی - از جمله مهندسی برق - در سراسر دنیا به این موضوع پرداخته‌اند. حال سؤال این است که چگونه می‌توانیم سیستمی هوشمند را بسازیم؟ به نظر می‌رسد که باتوجه به بخش قبل دوره اصلی برای انتقال هوشمندی خودمان به ماشین‌ها و نهایتاً ساخت سیستم هوشمند داریم که در ادامه آن‌ها را بررسی خواهیم کرد.

#### ۱-۳-۱ منطق

راه اول این است که بدون توجه به مغزمان، سعی کنیم آن جنبه از ذهنمان که منجر به هوشمندی می‌شود، یعنی تفکر، را به ماشین منتقل کنیم. برای این منظور لازم است که قالبی برای تفکر پیدا کرده تا با پیاده‌سازی آن قالب به خواسته خود برسیم. خوشبختانه هزاران سال است که متفکرین بر روی این مطلوب کار کرده‌اند و ثمره آن را منطق نامیده‌اند. اما انواع زیادی از منطق ارائه شده، از پی‌ریزی نخست آن توسط ارسطو تا منطق جدید و حتی بخش‌های زیادی از ریاضیات.

اما منطقی که امروزه بیشترین کاربرد را در ساخت سیستم‌های هوشمند دارد توسط جرج بول<sup>۱</sup> ارائه شد [4]. این منطق که ما آن را با نام منطق بولی می‌شناسیم شامل دو عنصر سازنده (صفر و یک) و همچنین سه عملیات اصلی و<sup>۲</sup>، یا<sup>۳</sup> و نقیض<sup>۴</sup> می‌باشد.

پس اگر بتوانیم این نظام منطقی را بر روی یک بستر مناسب پیاده کنیم خواهیم توانست لااقل بخشی از قدرت تفکر و هوشمندی خود را به ماشین منتقل کنیم. این کار بسیار مهم توسط کلود شانون<sup>۵</sup> در مقاله تاریخ سازش [5] انجام شده. البته از آن زمان تاکنون راه زیادی طی شده و امروزه این نظام را بصورت الکترونیک دیجیتال [6] پیاده می‌کنیم. در شکل ۱ سه گیت منطقی اصلی به همراه جدول صحت و رابطه جبری‌شان آورده شده‌اند.

Name	Graphic symbol	Algebraic function	Truth table															
AND		$F = x \cdot y$	<table border="1"> <thead> <tr> <th>x</th> <th>y</th> <th>F</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	x	y	F	0	0	0	0	1	0	1	0	0	1	1	1
x	y	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = x + y$	<table border="1"> <thead> <tr> <th>x</th> <th>y</th> <th>F</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	x	y	F	0	0	0	0	1	1	1	0	1	1	1	1
x	y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
Inverter		$F = x'$	<table border="1"> <thead> <tr> <th>x</th> <th>F</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </tbody> </table>	x	F	0	1	1	0									
x	F																	
0	1																	
1	0																	

شکل ۱: سه گیت اصلی منطقی [6]

منطق بولی قابلیت پیاده‌سازی روی هر بستری را ندارد و آن بستر باید قابل کنترل باشد. به همین دلیل است که نمی‌توانیم فقط با استفاده از منابع جریان و ولتاژ، مقاومت، خازن و سلف این منطق را پیاده‌سازی کنیم و نیاز به ادوات الکترونیکی مانند ترانزیستور [7] داریم.

George Boole<sup>۱</sup>

AND<sup>۲</sup>

OR<sup>۳</sup>

NOT<sup>۴</sup>

Claude Shannon<sup>۵</sup>



اوج این پیاده‌سازی، معماری فون‌نویمان<sup>۱</sup> است [8] که در اکثر کامپیوترهای امروزی به کار گرفته شده است. این معماری قابلیت اجرای هرکاری را دارد چراکه یک ماشین تورینگ<sup>۲</sup> است اما مشکلاتی نیز دارد که عموماً ناشی از ارتباط بین واحد پردازش و حافظه می‌باشد. به هر حال بنظر می‌رسد که این معماری دیگر برای انجام همه عملیات‌های محاسباتی در دنیای امروز کافی نیست که در ادامه به آن بیشتر می‌پردازیم.

پیاده‌سازی منطق با راه‌های بسیار فراوان دیگری نیز دارد که مجال برای بررسی آن‌ها نیست. فقط قابل ذکر است که در صورت استفاده از بسترهای آنالوگ، باید به سراغ ریاضیات برویم [9].

### ۱-۳-۲ معماری

در بخش قبل‌تر دیدیم که مغز عامل ایجاد تفکر و هوشمندی است. طبیعتاً این ویژگی باید از طریق خواص فیزیکی مغز و خصوصاً معماری آن ناشی شده باشد. پس اگر بتوانیم به نحوی این معماری را بر روی سیستمی پیاده‌سازی کنیم خوش‌بینانه به نظر می‌رسد که از آن سیستم از خود هوشمندی بروز دهد.

پس اگر هدف کپی‌کردن معماری مغز باشد، ما به دنبال پیاده‌سازی پردازنده‌ای هستیم که بتواند شبیه سیستم عصبی رفتار کند. به این پردازنده‌ها، نورومورفیک<sup>۳</sup> می‌گویند. در ادامه این نوشته مفصلاً با پردازنده‌های نورومورفیک آشنا شده و نمونه‌ای از آنها را پیاده خواهیم کرد.

اما استفاده از این روش - یعنی انتقال هوشمندی با استفاده از پیاده‌سازی معماری - نیازمند شناختی دقیق از سیستم عصبی و به‌صورت خاص مغز است. به این مهم در فصل بعد پرداخته شده است. پیش از اتمام این فصل لازم است تا کمی درباره آینده پردازش و مشکلات پیشروی آن صحبت کنیم.

### ۱-۴ آینده پردازش

بررسی آینده پردازش از آن جهت بسیار اهمیت دارد که امروزه نیاز به پردازنده‌های قوی‌تر و درعین حال کم‌مصرف‌تر با توجه به تولید روزافزون داده حس می‌شود؛ بنابراین باید از اکنون به فکر آینده باشیم و برای حل مشکلات برنامه‌ریزی کنیم.

---

<sup>۱</sup> Von Neumann architecture

<sup>۲</sup> Turing Machine

<sup>۳</sup> Neuromorphic

## ۱-۴-۱ قانون مور

گوردون مور<sup>۱</sup> یکی از مؤسسان برجسته شرکت اینتل<sup>۲</sup> در سال ۱۹۶۵ پیش‌بینی کرد که باتوجه‌به سرعت رشد صنعت الکترونیک، احتمالاً شاهد آن خواهیم بود که تعداد ترانزیستورها روی یک تراشه با مساحت ثابت، هر دو سال یکبار تقریباً دوبرابر می‌شوند. [10] این پیش‌بینی به قانون مور<sup>۳</sup> معروف است. همانطور که در شکل ۲ [11] می‌بینید تا به امروز این پیش‌بینی تا حد بسیار خوبی صادق بوده است، اما اعتبار آن رو به اتمام است!

از آنجاکه امروزه تکنولوژی ساخت ترانزیستورها به حدود ۳ نانومتر رسیده است، در حال نزدیک شدن به مرز محدودیت‌های فیزیکی ناشی از اثرات کوانتومی هستیم. جالب است بدانید که قطر اتم سیلیکون ۰.۲ نانومتر است. پس نمی‌توانیم ترانزیستورها را هرچقدر که می‌خواهیم کوچک کنیم و توان پردازشی بیشتری تولید کنیم! در نتیجه باید به فکر راه‌حل‌های دیگری برای آینده باشیم، چراکه باتوجه‌به افزایش روزافزون نیاز به پردازش بیشتر، باید پردازنده‌هایمان را نیز بهبود دهیم و کوچک کردن ترانزیستور دیگر پاسخگو نیست!

---

Gordon Moore<sup>۱</sup>

Intel<sup>۲</sup>

Moore's law<sup>۳</sup>



## ۱-۴-۲ راه حل‌ها

به نظر می‌رسد که در کل سه‌راه برای حل این مشکل داریم که در ادامه این راه‌ها را بررسی خواهیم کرد. راه اول) استفاده حداکثری از توان محاسباتی: طبیعتاً اولین راهی که به ذهن می‌رسد این است که تا می‌توانیم با بهینه‌کردن الگوریتم‌ها و روش‌های برنامه‌نویسی، از تمام توان موجود در پردازنده استفاده کنیم. البته که این راه تا حد بسیار خوبی جلو رفته و دانشمندان علوم کامپیوتر بر روی آن تمرکز زیادی دارند. اما این راه هرچقدر هم که جلو برود باز نمی‌تواند پاسخ نهایی باشد.

راه دوم) تغییر بستر محاسباتی: در این راستا نیز تلاش‌های زیادی در حال انجام است که به نظر می‌رسد آینده محاسبات را خواهد ساخت و راه حل نهایی خواهد بود. معروف‌ترین ایده در این بخش محاسبات کوانتومی<sup>۱</sup> است. در محاسبات کوانتومی با کاهش دما و استفاده از ذرات کوانتومی به جای بیت، شاهد پدید آمدن برخی خواص بی‌نظیر مکانیک کوانتومی نظیر برهم‌نهی<sup>۲</sup> و درهم‌تنیدگی کوانتومی<sup>۳</sup> هستیم که با استفاده از آن‌ها می‌توانیم به جای منطق دوارزشی بولی، از منطق چند ارزشی کوانتومی استفاده کنیم. نشان داده شده که این منطق امکان می‌دهد که در برخی محاسبات بسیار پیچیده سرعت انجام محاسبات به صورت نمایی افزایش یابد. کامپیوترهای کوانتومی دو مشکل عمده دارند، اولاً پیاده کردن دمایی نزدیک به صفر کلون کاری هزینه بر و سخت است و ثانیاً این نوع محاسبات باعث افزایش سرعت در همه الگوریتم‌ها نمی‌شود و هنوز کاربرد اصلی آن‌ها مشخص نشده است. [12]

راه سوم) تغییر معماری: راه حل آخر تغییر معماری پردازنده است. این راه نکات مثبت فراوانی دارد، اولاً که با استفاده از تکنولوژی‌های الکترونیکی امروزه نیز قابل پیاده‌سازی است، ثانیاً محل بسیار جذابی برای بیان کردن ایده‌های جدید است. با توجه به نکاتی که در فصل قبل گفته شد طبیعتاً یکی از بهترین گزینه‌ها (از نظر بسیاری، بهترین!) استفاده از معماری مغز است. اینجاست که معماری نورومورفیک وارد می‌شود. درباره چيستی این معماری و مزایای آن در ادامه این نوشته بسیار صحبت خواهیم کرد. در اینجا خوب است اشاره کنم که ایده استفاده از معماری مغز در کامپیوترها اصلاً ایده جدیدی نیست و قدمت آن به اندازه خود کامپیوترها است. در واقع جان فون‌نویمان<sup>۴</sup> این ایده را در سال‌های ابتدایی توسعه کامپیوترها مطرح کرده بود. [9] لازم به ذکر

---

Quantum Computing<sup>۱</sup>

Superposition<sup>۲</sup>

Quantum entanglement<sup>۳</sup>

John von Neumann<sup>۴</sup>

است که در این بخش راه‌های بسیار دیگری نیز وجود دارد. یکی از معروفترین آنها معماری FPGA<sup>۱</sup> می‌باشد که کاربردهای زیادی نیز پیدا کرده است.

## ۱-۵ نتیجه‌گیری

در این فصل ابتدا راجع به ارتباط هوش با ذهن و مغز صحبت کردیم و متوجه شدیم دوره کلی برای ساخت سیستم‌های هوشمند داریم. در ادامه با بررسی قانون مور فهمیدیم که نیازمند راه‌های دیگری برای افزایش توان محاسباتی خود هستیم. دیدیم که یکی از این راه‌ها معماری است و یکی از جذاب‌ترین این معماری‌ها نورومورفیک است که در ادامه با آن آشنا خواهیم شد.

---

<sup>۱</sup> Field-programmable gate array

## فصل دوم: مقدمه‌ای بر علوم اعصاب محاسباتی

### ۲-۱ مقدمه

در فصل قبل دیدیم که معماری نورومورفیک یکی از راه‌های مناسب برای حل مشکل آینده محاسبات است. همچنین دیدیم که برای پیاده‌سازی معماری سیستم عصبی باید آن را خوب بررسی کنیم و بشناسیم، بنابراین در ادامه ابتدا مقدمه‌ای درباره علوم اعصاب خواهیم گفت، سپس خواهیم دید که چگونه می‌توان بخش‌های مختلف مغز را به صورت ریاضیاتی مدل‌سازی کرد. بحث اخیر موضوع علوم اعصاب محاسباتی است.

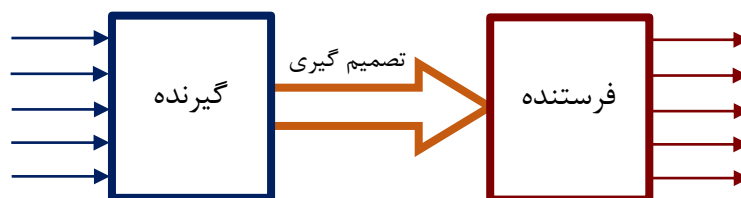
### ۲-۲ علوم اعصاب

هدف اصلی علوم اعصاب همان‌طور که از نام آن مشخص بررسی سیستم عصبی انسان و موجودات دیگر است. این بررسی در سطوح متفاوتی انجام می‌شود (که در ادامه معرفی خواهند شد). علاوه بر این روش‌های متفاوتی نیز برای بررسی وجود دارد، از روش‌های آزمایشگاهی تا مدل‌سازی و روش‌های نظری و محاسباتی. درباره اهمیت علوم اعصاب و تأثیر بی‌بدیل آن در آینده علم و فناوری بسیاری می‌توان صحبت کرد که در اینجا مجال برای آن نیست.

لازم به ذکر است که تمامی بخش‌های علوم اعصاب که در ادامه آورده شده‌اند بر اساس [13]، [14] و [15] هستند. علاوه بر این باید اشاره کرد که بخش پیش رو به هیچ وجه کامل نیست و صرفاً مقدمه‌ایست برای فهم ادامه مباحث. اما نباید به آن بی‌توجهی کرد چرا که کلید فهم صحیح محاسبات نورومورفیک، بدون شک فهم درست سیستم عصبی است.

### ۲-۲-۱ ساختار نورون

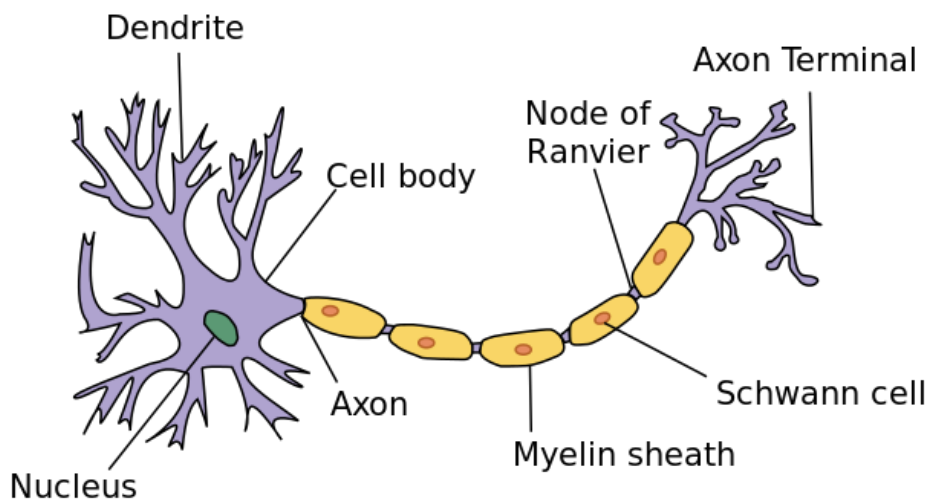
به سلول عصبی که واحد اصلی سازنده آن است نورون<sup>۱</sup> می‌گویند. نورون‌ها به صورت کلی از سه بخش اصلی ساخته شده‌اند که در ادامه به توضیح آنها می‌پردازیم. اما در حالت ساده می‌توان نورون را مانند زیر مدل کرد.



شکل ۳: مدل سیستماتیک یک نورون در حالت

همان طور که از شکل بالا مشخص است نورون ورودی‌های زیادی دارد که در گیرنده گرفته می‌شود و در ادامه یک تصمیم‌گیری ساده انجام می‌شود و در فرستنده سیگنالی تولید می‌شود و آن سیگنال به ترمینال‌های خروجی فرستاده می‌شود.

حال باتوجه به این نکته ساختار بیولوژیک نورون را بررسی می‌کنیم. همان طور که از شکل زیر مشخص



شکل ۴: ساختار بیولوژیک نورون

است نورون شامل چهاربخش اصلی دندریت<sup>۱</sup>، جسم سلولی (سوما<sup>۲</sup>) و آکسون<sup>۳</sup> و ترمینال‌های پیش‌سیناپسی<sup>۴</sup> است. در شکل علاوه بر این چهاربخش، بخش‌های دیگری نیز مشخص شده‌اند که در ادامه به آن‌ها اشاره می‌کنیم. نورون‌ها ابعاد بسیار متفاوتی دارند و طول آکسون برخی از آن‌ها می‌تواند به بیش از یک متر نیز برسد! اما اندازه عموم آن‌ها - مخصوصاً آن‌هایی که اهمیت محاسباتی دارند - در رنج چند ده میکرومتر است.

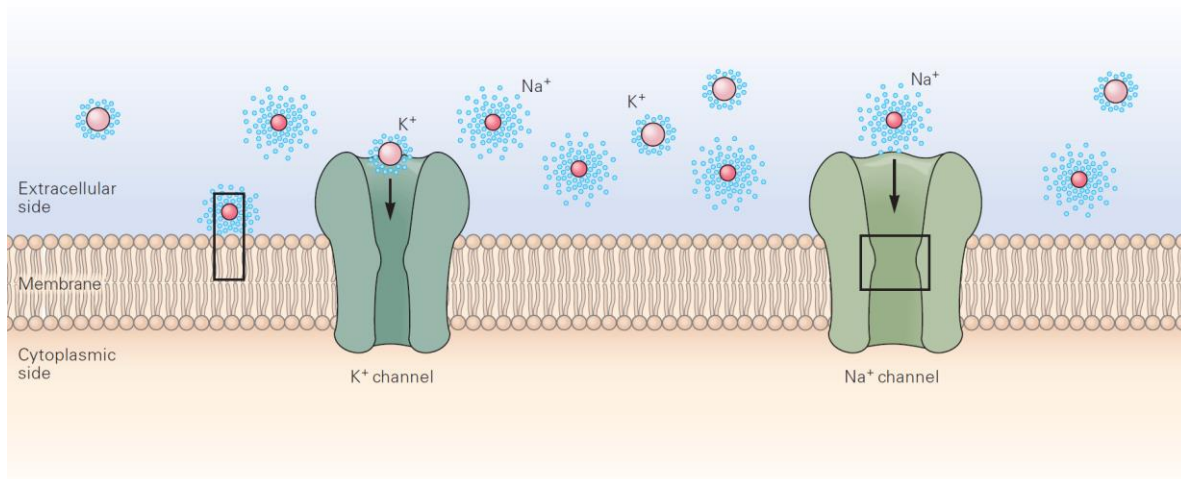
برای توضیح ابتدا از سوما شروع می‌کنیم، این بخش شامل اکثر اندامک‌های سلولی نورون است که مهم‌ترین آن‌ها هسته است. هسته شامل DNA است که تعیین‌کننده ساختار سلول و بخش زیادی از عملکرد آن است. باقی اندامک‌های درون سوما وظیفه تولید انرژی، ساخت پروتئین‌های موردنیاز و... را بر عهده دارند.

بخش مهم بعدی دندریت‌ها هستند. این بخش از نورون درواقع گیرنده‌های آن است که سیگنال تولید شده توسط نورون‌های دیگری که به نورون موردنظر متصل‌اند را دریافت می‌کند. در بخش‌های بعدی درباره چگونگی دریافت سیگنال در دندریت‌ها صحبت خواهیم کرد.

- 
- Dendrite <sup>۱</sup>
  - Soma <sup>۲</sup>
  - Axon <sup>۳</sup>
  - Presynaptic terminals <sup>۴</sup>

اما مهم‌ترین بخش نورون که ویژگی‌های بی‌نظیر محاسباتی، ناشی از آن است، آکسون است. در این بخش با جزئیات بیشتری درباره ساختار آکسون صحبت خواهیم کرد چرا که در ادامه به این جزئیات نیاز است. قطر آکسون در انسان بین ۱ تا ۲۵ میکرومتر است.

ابتدای آکسون بخشی وجود دارد به نام برآمدگی آکسون<sup>۱</sup> که سیگنال‌های ورودی - عمدتاً دریافت شده از دندریت - در آن جمع (جمع مکانی و زمانی) شده و اگر از یک مقدار آستانه مشخص بیشتر شوند نورون اصطلاحاً/اسپایک<sup>۲</sup> می‌زند، پس در واقع این بخش تصمیم‌گیری فعال‌شدن یا نشدن نورون را بر عهده دارد. درباره اسپایک زدن نورون‌ها در بخش توضیحات بیشتری خواهیم داد. در صورتی که نورون اسپایک بزند، آکسون مسئول ساخت و انتقال این اسپایک است. به این منظور، آکسون از ساختارهای پروتئینی خاصی به نام کانال‌های یونی<sup>۳</sup> و پمپ‌های یونی<sup>۴</sup> استفاده می‌کند.



شکل ۵: کانال‌های یونی در غشا آکسون نورون [14]

همان‌طور که در شکل بالا مشخص است درون نورون (آکسون) توسط غشا<sup>۵</sup> از بیرون آن جدا شده. خود غشا به تنهایی، با توجه به ساختارش نسبت به عبور یون‌ها مقاوم است و هیچ بار الکتریکی‌ای نمی‌تواند از آن عبور کند. به همین دلیل از لحاظ الکتریکی می‌توان آن را نارسانا فرض کرد. اما همان‌طور که در شکل پیداست کانال‌های یونی در صورت بازبودن اجازه عبور برخی یون‌ها را می‌دهند. هر کانال اجازه عبور یون خاص خود را می‌دهد. مثلاً همان‌طور که در شکل مشخص است کانال سدیم، اجازه عبور سدیم و کانال پتاسیم، اجازه

Axon hillock<sup>۱</sup>

Spike<sup>۲</sup>

Ion channels<sup>۳</sup>

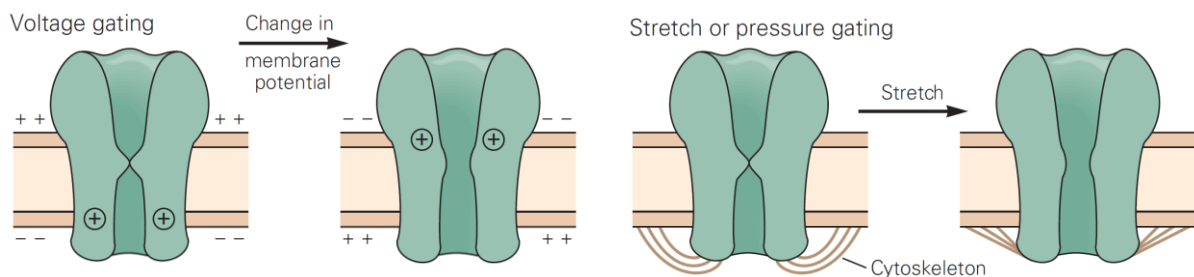
Ion pumps<sup>۴</sup>

Membrane<sup>۵</sup>



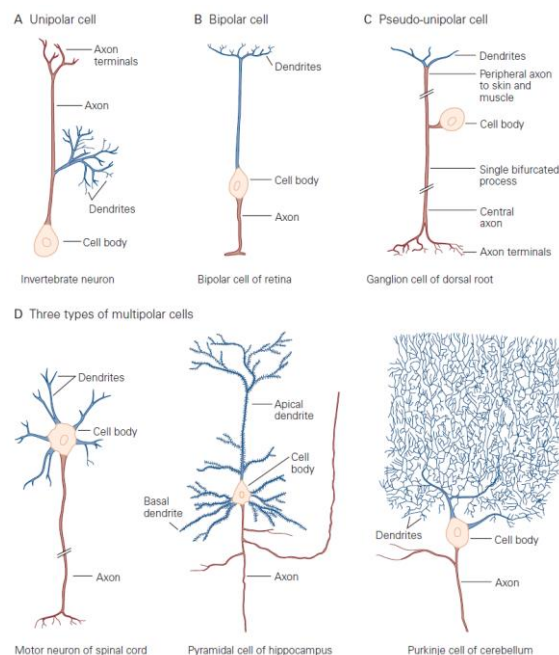
عبور پتاسیم را می‌دهد. اینکه کانال اجازه عبور چه یونی را می‌دهد و با چه مکانیسمی باز و بسته می‌شود کاملاً به ساختار پروتئینی آن وابسته است که خود بحثی مفصل است.

کانال‌های یونی روش‌های متفاوتی برای باز و بسته شدن دارند. مثلاً برخی از آنها (شکل ۶ سمت راست) حساس به فشار و کشش هستند. برخی دیگر که مهم‌ترین نوع کانال‌ها در ایجاد اسپایک هستند، حساس به ولتاژ هستند (شکل ۶ سمت چپ). در این نوع از کانال‌ها، باز و بسته شدن کانال وابسته به ولتاژ است. به این صورت که اگر اختلاف ولتاژ دو سر کانال (بیرون و درون سلول) به عدد خاصی برسد آن کانال با تغییر ساختار فضایی پروتئینی خود، باز یا بسته می‌شود. (احتمالاً به یاد مقاومت وابسته به ولتاژ افتاده‌اید! بله درست فکر می‌کنید، در ادامه خواهیم دید که دقیقاً می‌توان این کانال‌ها را با مقاومت وابسته به ولتاژ مدل کرد)



شکل ۶: کانال یونی وابسته به فشار و کشش در سمت راست و کانال یونی وابسته به ولتاژ در سمت چپ [14]

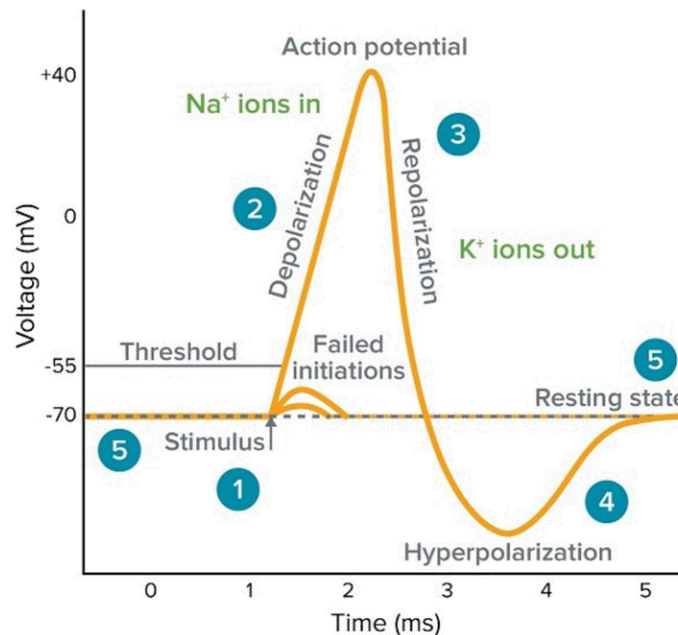
بخش چهارم که در واقع انتهای آکسون است، ترمینال‌های پیش‌سیناپسی هستند. این ترمینال‌ها در انتقال سیگنال بین نورون‌ها نقش دارند. درباره این ترمینال‌ها در بخش سیناپس بیشتر صحبت خواهیم کرد. در نهایت لازم به ذکر است که این توضیحات بسیار ساده‌سازی شده‌اند و اجزا نورون بسیار پیچیده‌تر هستند و نورون‌ها انواع بسیار متفاوتی دارند. در شکل ۷ برخی از این انواع را مشاهده می‌کنید.



شکل ۷: برخی از انواع نورون شامل تک‌قطبی، دوقطبی و چند قطبی

## ۲-۲-۲ پتانسیل عمل

ویژگی اصلی نوروها قدرت آنها در محاسبات و پردازش اطلاعات است. اطلاعات در سیستم عصبی به صورت الکتریکی است؛ بنابراین آن بخش از رفتار نورو که در بررسی رفتار محاسباتی آن بیشترین تأثیر را دارد، رفتار الکتریکی نورو است. در شکل زیر رفتار الکتریکی نورو که به آن پتانسیل عمل<sup>۱</sup> یا اسپایک می‌گویند آورده شده. در ادامه ابتدا بخش‌های مختلف شکل توضیح داده می‌شود و سپس درباره چگونگی ایجاد هر بخش نیز توضیحاتی داده می‌شود.



شکل ۸: مراحل ایجاد پتانسیل عمل یا اسپایک

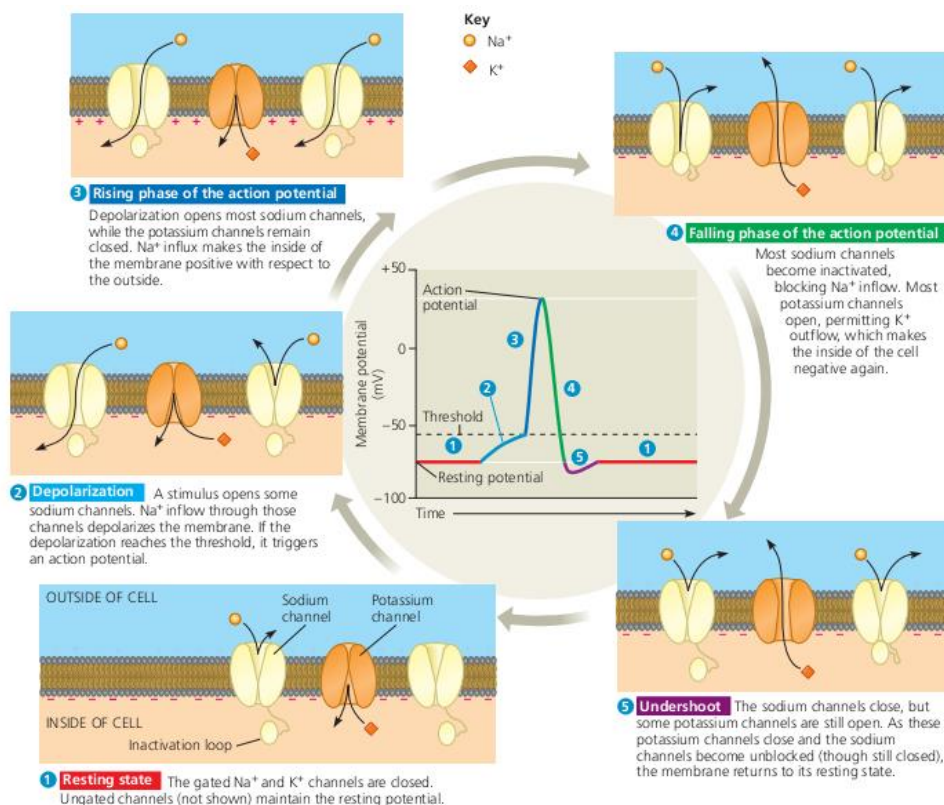
شکل ۸ اختلاف ولتاژ دو سر غشا را در زمان‌های متفاوت نشان می‌دهد. اختلاف ولتاژ دو سر غشا مطابق رابطه زیر تعریف می‌شود:

$$V_m = V_{in} - V_{ex} \quad (\text{معادله ۱})$$

که در آن  $V_m$  اختلاف ولتاژ غشا،  $V_{in}$  ولتاژ درون سلول<sup>۲</sup> و  $V_{ex}$  ولتاژ خارج سلول<sup>۳</sup> است. این مقدار در حالت استراحت<sup>۴</sup> به دلیل اختلاف غلظت یون‌ها در خارج و درون سلول حدوداً برابر منفی هفتاد میلی‌ولت است که نشان می‌دهد درون سلول نسبت به بیرون آن ولتاژ کمتری دارد.

- 
- Action Potential<sup>۱</sup>
  - Intracellular<sup>۲</sup>
  - Extracellular<sup>۳</sup>
  - Resting state<sup>۴</sup>

حال باتوجه به شکل ۹ مراحل مختلف ایجاد اسپایک را توضیح می دهیم:



شکل ۹: مراحل مختلف ایجاد پتانسیل عمل و عامل زیستی آن

**مرحله اول:** حالت استراحت: همان طور که گفته شد در حالت استراحت نوروں در واقع در تعادل است و اکثر کانال ها بسته هستند<sup>۱</sup> همچنین  $V_m$  نیز ثابت و برابر حدود منفی هفتاد میلی ولت است.

**مرحله دوم:** دپلاریزاسیون<sup>۲</sup>: در این مرحله تحریک ورودی باعث باز شدن برخی از کانال های سدیمی می شود و چون غلظت یون سدیم بیرون سلول حدود ۱۰ برابر بیشتر از داخل سلول است، به دلیل قانون دوم ترمودینامیک، جریانی از یون های سدیم به درون سلول برقرار می شود و  $V_m$  افزایش می یابد.

**مرحله سوم:** اسپایک زدن: اگر در مرحله قبل تحریک ورودی به اندازه کافی قوی باشد (بیشتر از آستانه<sup>۳</sup>) و تعداد کافی کانال سدیمی باز شود باز هم یون های سدیم بیشتری به درون سلول می روند و در نتیجه  $V_m$  به بیشترین مقدار خود می رسد و اصطلاحاً نوروں اسپایک می زند. (ولتاژ آستانه در حدود منفی ۵۵ و ولتاژ اسپایک در حدود مثبت ۵۰ میلی ولت است).

<sup>۱</sup> البته کانال های دیگری نیز وجود دارند که آن ها بسته نیستند. در اینجا نیازی به بررسی آن ها نیست.

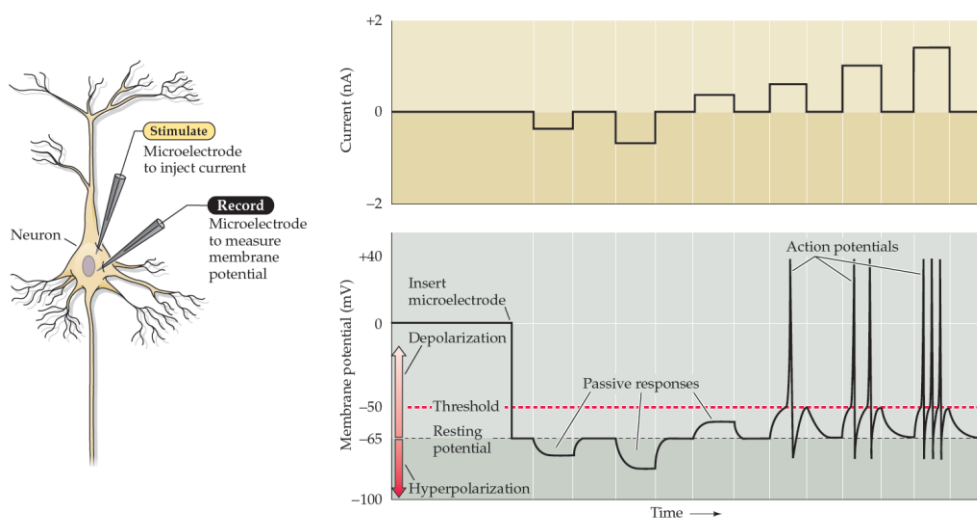
<sup>۲</sup> Depolarization

<sup>۳</sup> Threshold

**مرحله چهارم: رپلاریزاسیون<sup>۱</sup>:** پس از اسپایک زدن کانال‌های سدیمی بسته می‌شوند و کانال‌های پتاسیمی که به ولتاژ باز شدن خود رسیده‌اند، باز می‌شوند. اما برعکس سدیم، غلظت یون پتاسیم درون سلول حدود ۲۰ برابر بیشتر از خارج سلول است، پس بازهم باتوجه به قانون دوم ترمودینامیک جریانی از یون‌های پتاسیم از درون سلول به خارج آن جاری می‌شود که طبعاً باعث کاهش  $V_m$  می‌شود.

**مرحله پنجم: هایپرپلاریزاسیون<sup>۲</sup>:** از آنجاکه کانال‌های پتاسیمی با تأخیر بسته می‌شوند، بنابراین بازهم ولتاژ غشا کاهش می‌یابد تا اینکه به حدود منفی ۸۰ میلی‌ولت می‌رسد. سپس کانال‌های پتاسیمی نیز بسته می‌شوند و ولتاژ غشا به کمک پمپ‌های سدیم - پتاسیم و صرف انرژی به همان ولتاژ تعادل خود برمی‌گردد.

برای اضافه کردن برخی نکات و جمع‌بندی این بخش به شکل ۱۰ توجه کنید.



شکل ۱۰: پاسخ یک نورون به تحریک‌های متفاوت [15]

همان‌طور که مشخص است دو میکرو الکتروود یکی برای ثبت و دیگری برای تحریک [16]، درون نورون کاشته شده‌اند. نمودار بالا جریان تحریک وارد شده به سلول را نشان می‌دهد و نمودار پایین نشان دهنده پاسخ ضبط شده می‌باشد. در ابتدا قبل از شروع تحریک ولتاژ غشا در حالت استراحت خود است. سپس دو تحریک با مقادیر کمتر از صفر وارد می‌شوند، هیچکدام از این دو تحریک (حتی آنی که قدر مطلق ولتاژی که ایجاد می‌کند بیشتر از ولتاژ آستانه است) قادر به ایجاد اسپایک نیستند چراکه مقدار آن‌ها منفی است. سپس تحریکی داریم با مقدار مثبت اما چون به آستانه نمی‌رسد، اسپایک تولید نمی‌کند. اما سه تحریک بعدی باعث ایجاد ولتاژی بیشتر از آستانه می‌شوند پس اسپایک تولید می‌کنند. دقت کنید که هرچه مقدار تحریک بیشتر

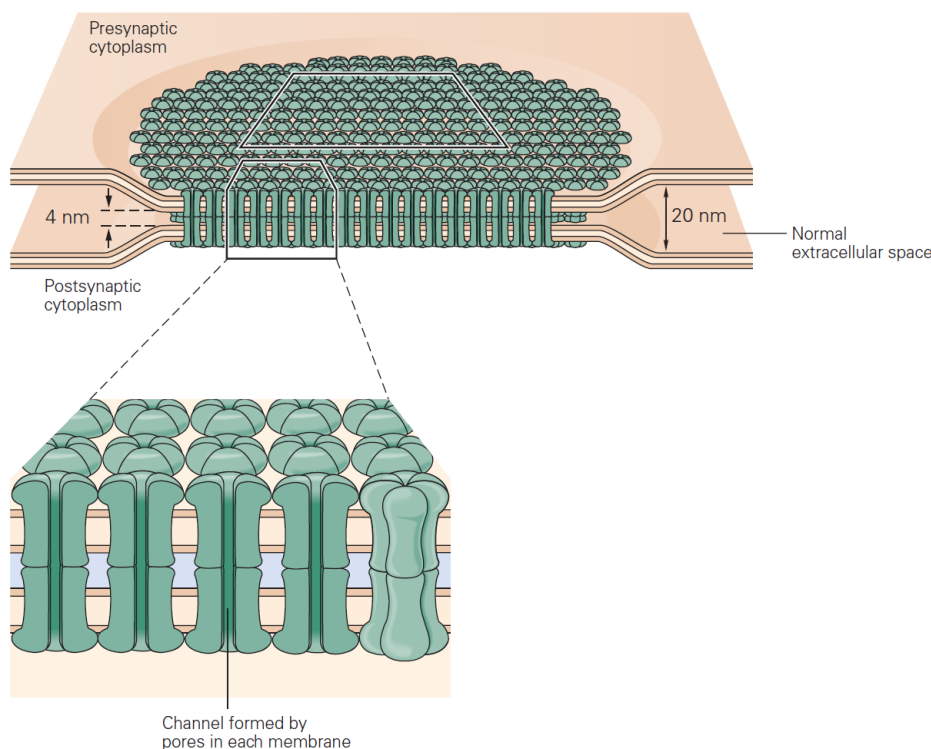
<sup>۱</sup> Repolarization  
<sup>۲</sup> Hyperpolarization

باشد، فرکانس اسپایک زدن افزایش می‌یابد. قابل ذکر است که نورون ویژگی‌های مهم دیگری مانند ادپتیشن نیز دارد که در اینجا مجال برای پرداختن به آنها نیست.

### ۲-۲-۳ سیناپس

سیناپس<sup>۱</sup> محل اتصال عملکردی دو نورون با یکدیگر است که در آن سیگنال از نورون پیش‌سیناپسی<sup>۲</sup> به نورون پس‌سیناپسی<sup>۳</sup> منتقل می‌شود. این انتقال می‌تواند به دو صورت باشد: الکتریکی یا شیمیایی. در ادامه این مدل را بررسی می‌کنیم.

**سیناپس الکتریکی<sup>۴</sup>:** در این مدل از سیناپس همان‌طور که در شکل ۱۱ مشخص است در این نوع از سیناپس ارتباط مستقیمی بین دو نورون برقرار می‌شود و یونها مستقیماً از نورون پیش‌سیناپسی از طریق کانال‌هایی به نام کانکسون<sup>۵</sup>، به نورون پس‌سیناپسی منتقل می‌شوند. به این نوع اتصال اصطلاحاً *gap junction* می‌گویند. سرعت انتقال اطلاعات در این سیناپس بسیار زیاد است و عموماً کاربرد آن

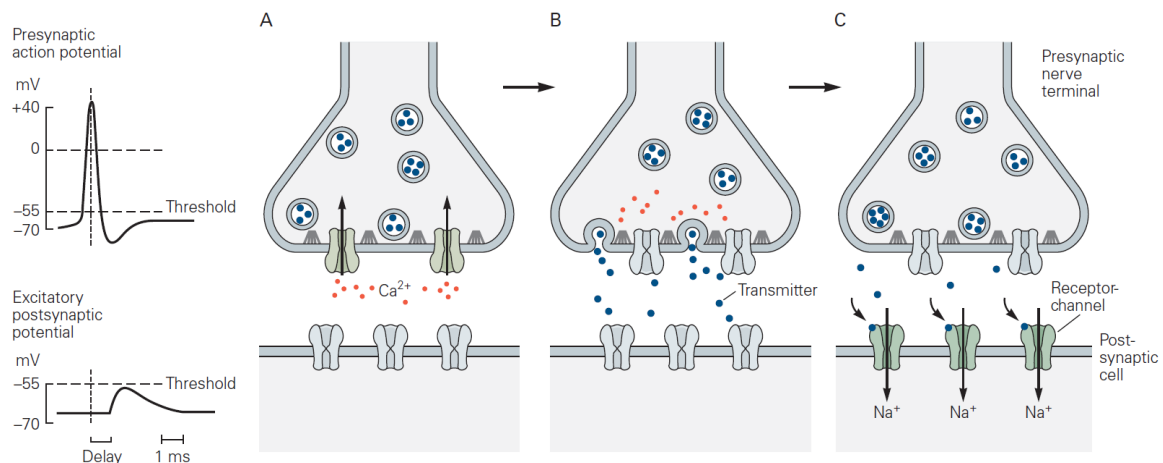


شکل ۱۱: تصویری از سیناپس الکتریکی [14]

- Synapse<sup>۱</sup>
- Presynaptic<sup>۲</sup>
- Postsynaptic<sup>۳</sup>
- Electric Synapse<sup>۴</sup>
- Connexon<sup>۵</sup>

سنکرون سازی نورون ها با یکدیگر است. البته این نوع اتصال ویژگی های بسیار جالب نیز دارد، از جمله اینکه دوطرفه هستند.

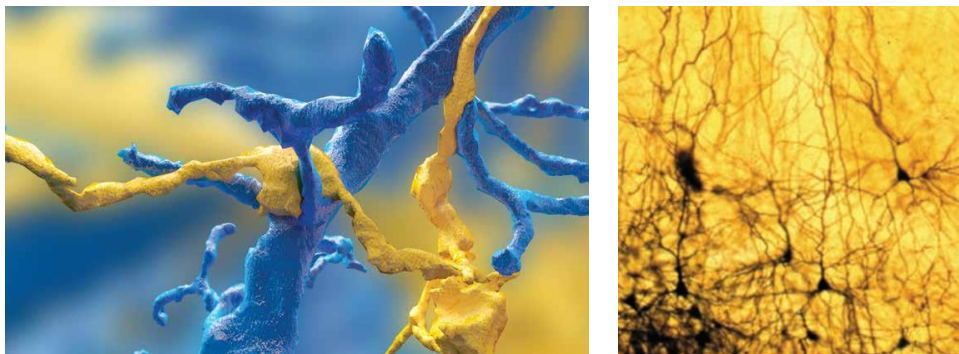
**سیناپس شیمیایی:** اما نوع رایج تر، سیناپس شیمیایی است. این مدل نسبت به سیناپس الکتریکی سرعت کمتری - در حدود ۱ تا ۵ میلی ثانیه - دارد و برای انتقال سیگنال از مواد شیمیایی استفاده می کند.



شکل ۱۲: نحوه کار سیناپس شیمیایی [14]

همان طور که در شکل ۱۲ مشخص است نحوه کار یک سیناپس شیمیایی را می توان به سه مرحله تقسیم نمود. ابتدا با رسیدن اسپایک به ترمینال نورون پیش سیناپسی، کانال های حساس به ولتاژ یون کلسیم باز می شوند و مقدار زیادی یون کلسیم وارد ترمینال آکسون نورون پیش سیناپسی می شود. سپس این یون ها باعث باز شدن وزیکل ها می شوند. این وزیکل ها که حاوی نوروترنسمیترها هستند در فضای بین سیناپسی رها می شوند. سپس نوروترنسمیترها باعث باز شدن کانال های یون سدیم در دندریت نورون پس سیناپسی می شوند و سپس با ورود یون های سدیم به درون نورون دوم تحریکی به وجود می آید، در صورتی که این تحریک به همراه باقی تحریک های هم زمان از آستانه بیشتر باشند نورون دوم اسپایک می زند.

باتوجه به نمودار سمت راست شکل ۱۲، می بینید که دامنه ولتاژ ایجاد شده توسط یک ورود از یک دندریت بسیار کم است و نسبت به اسپایک اصلی تأخیر دارد؛ بنابراین می توان گفت که سیناپس شیمیایی علاوه بر اعمال تأخیر به سیگنال ورودی، دامنه آن را نیز به شدت کاهش می دهد.



شکل ۱۳: تصویر واقعی از نورون و یک سیناپس که توسط میکروسکوپ الکترونی گرفته شده و رنگ آمیزی شده. آکسون به رنگ زرد و دندریت به رنگ آبی هستند [13] و [15]

پیش از شروع بخش بعد لازم به ذکر است که تمام فرایندهای گفته شده مربوط به نورون‌های تحریکی بودند، در مغز نورون‌های دیگری نیز وجود دارند که مهاری هستند. تفاوت آنها این است که فعال‌شدنشان و انتقال فعالیت از طریق سیناپس به نورون بعدی باعث مهار آن نورون می‌شود. می‌توان این‌طور گفت که ولتاژ پس‌سیناپسی که این نورون‌ها تولید می‌کنند ولتاژ پس‌سیناپسی نورون‌های دیگر را خنثی می‌کند. (کمی مانند وزن منفی در شبکه عصبی مصنوعی)

## ۲-۲-۴ شبکه‌های نورونی

طبیعتاً نورون‌ها به‌تنهایی نمی‌توانند قدرت محاسباتی خاصی داشته باشند و برای این منظور با یکدیگر تشکیل شبکه می‌دهند. شبکه‌های نورونی ساختارهای بسیار متفاوت و پیچیده‌ای دارند. مغز انسان شامل  $8.6 \times 10^9$  نورون می‌باشد که هر کدام از این نورون‌ها خود صدها و در مواردی هزاران سیناپس دارند. تعداد دقیقی برای سیناپس‌ها بدست نیامده اما تعداد تخمینی آنها در حدود  $1.5 \times 10^{14}$  می‌باشد! خود این اعداد نشان دهنده پیچیدگی غیرقابل توصیف مغز می‌باشد. در نظر بگیرید که تعداد زیادی از این اتصالات فیدبک هستند که این موضوع خود پیچیدگی را دو چندان می‌کند. موضوع دیگر غیرخطی بودن ذاتی مغز است که آن هم به پیچیدگی آن دامن می‌زند. [17] دقیقاً به دلیل همین پیچیدگی‌های بسیار زیاد شبکه‌های نورونی است که امروزه یکی از قدرتمندترین ابزارهای تحلیل ریاضی شبکه‌های نورونی منیفلدها<sup>۱</sup> هستند. [18]

نکته دیگر تفاوت ساختار شبکه‌های نورونی در بخش‌های مختلف مغز است. مثلاً نورون‌های کورتکس<sup>۲</sup> سیناپس‌های بسیاری دارند و به همین جهت عموم فرایندهای محاسباتی در کورتکس انجام می‌شود. یا به‌عنوان مثال نورون‌های مربوط به حافظه فیدبک‌های بسیار زیاد و متنوعی دارند.

در نهایت باید گفت که همه اینها تنها بخشی از پیچیدگی شبکه‌های نورونی هستند چراکه علاوه بر نورون سلول‌های دیگری مثل گلیا<sup>۳</sup> نیز در مغز وجود دارند. به این سلول‌ها تا به امروز در مدل‌سازی‌های محاسباتی علوم اعصاب توجه زیادی نمی‌شد، اما در سال‌های اخیر کشفیاتی مبتنی بر نقش آنها در عملیات‌های پردازشی نیز به‌دست آمده که در نوع خود بسیار جالب است و می‌تواند اثرگذار باشد.

البته این پیچیدگی بسیار زیاد نباید ما را برای الهام گرفتن از ساختار شبکه‌های نورونی جهت ساخت پردازنده ناامید کند. همان‌طور که در ادامه خواهیم دید بسیار از مدل‌های بسیار ساده‌سازی شده از مغز، توان پردازشی بی‌نظیری ارائه داده‌اند.

---

Manifolds<sup>۱</sup>

Cortex<sup>۲</sup>

Glia<sup>۳</sup>

## ۲-۲-۵ سطوح بررسی

در نهایت خوب است تا درباره سطوح بررسی‌ای که در علوم اعصاب وجود دارد نیز کمی صحبت کنیم. همان‌طور که در شکل ۱۴ مشخص است می‌توان ۵ سطح را برای بررسی مطالعات علوم اعصاب در نظر گرفت.

[13]



شکل ۱۴: سطوح متفاوت بررسی علوم اعصاب

- **علوم اعصاب مولکولی**<sup>۱</sup>: بنیادی‌ترین سطح بررسی که درباره اثرگذاری برهم‌کنش‌های مولکولی بر سیستم عصبی است.
- **علوم اعصاب سلولی**<sup>۲</sup>: بررسی در سطح نورون، مانند توضیحاتی که داده شد.
- **علوم اعصاب سیستم**<sup>۳</sup>: بررسی شبکه‌های نورونی و اینکه چه عملکردی دارند. مثلاً شبکه بینایی
- **علوم اعصاب رفتاری**<sup>۴</sup>: بررسی در سطح رفتار و زیرساخت‌های عصبی آن
- **علوم اعصاب شناختی**<sup>۵</sup>: بررسی رفتارهای سطح بالاتر شناختی و پیدا کردن فرایندهای عصبی‌ای که باعث ایجاد آن‌ها می‌شود.

عموم بحث‌های ما مربوط به سطح سلولی و سیستم است.

<sup>۱</sup> Molecular Neuroscience

<sup>۲</sup> Cellular Neuroscience

<sup>۳</sup> Systems Neuroscience

<sup>۴</sup> Behavioral Neuroscience

<sup>۵</sup> Cognitive Neuroscience



## ۲-۳ مدل سازی نورون

حال که با ساختار بیولوژیک عناصر اصلی سازنده مغز آشنا شدیم فرصت آن رسیده که به سراغ مدل سازی ریاضی آنها برویم. اهمیت مدل سازی ریاضی مشخص است چراکه فقط با وجود مدل های ریاضی است که می توان یک نظریه علمی کامل درست کرد و به صورت کامل مغز را تبیین کرد. شاخه ای که مدل سازی در آن بررسی می شود علوم اعصاب نظری<sup>۱</sup> یا علوم اعصاب محاسباتی<sup>۲</sup> نام دارد. هدف نهایی علوم اعصاب نظری ارائه مدلی منسجم است که بتواند نحوه پدید آمدن رفتارها و شناخت، از فعالیت های عصبی را به صورت کامل تبیین کند. مباحثی که در این بخش و بقیه بخش های مدل سازی آورده شده اند از [19]، [20] و [21] هستند. ابتدا با انواع روش های مدل سازی نورون شروع می کنیم. مدل هایی که در ادامه آورده شده اند بر اساس مدل سازی با کمک مدارهای الکتریکی هستند، در حالی که می توان نورون ها را با روش های دیگری مانند استفاده از سیستم های دینامیک [17]، باندگراف [22] یا روش های دیگری نیز مدل کرد.

### ۲-۳-۱ مدل هاجکین - هاکسلی

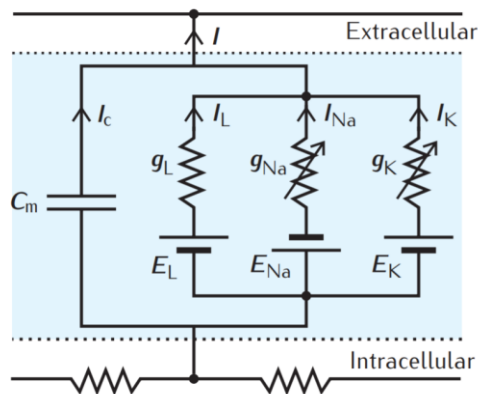
مدل هاجکین - هاکسلی<sup>۳</sup> (HH) که توسط دو دانشمند به همین نام در سال ۱۹۵۲ ارائه شد [23] و جایزه نوبل فیزیولوژی سال ۱۹۶۳ را برایشان به ارمغان آورد، هنوز هم بعد از گذشت نیم قرن یکی از کاملترین مدل های ارائه شده از نورون است.

برای توضیح این مدل ابتدا باید درباره مدل سازی یک کانال یونی صحبت کنیم. از بخش های قبل به یاد بیاورید که گفته شد هر دسته کانال های یونی اجازه عبور یک یون خاص را می دهند و یا بسته هستند یا باز. اما از آنجایی که تعداد زیادی از هر کدام از انواع کانال های یونی وجود دارد پس می توان گفت در هر لحظه از زمان برای هر یون، یک رسانایی وجود دارد. مقدار این رسانایی که در واقع نشان دهنده راحتی عبور آن یون از غشا است، همان طور که گفته شد وابسته به عوامل مختلفی مخصوصاً ولتاژ غشا است. باتوجه به موارد گفته شده می توان برای هر کدام از انواع کانال های یونی رابطه زیر را نوشت:

$$I_{ion} = g_{ion}(V_m - E_{ion}) \quad (\text{معادله ۲})$$

که در آن سمت چپ جریان الکتریکی ناشی از عبور آن یون مورد نظر،  $g_{ion}$  رسانایی آکسون برای یون مورد نظر و  $E_{ion}$  و ولتاژ حالت تعادل آن یون است.

مدل HH ناشی از استخراج این معادلات برای هر کانال یونی و سپس جمع کردن آنها باهم است. برای فهم بهتر لازم است تا مدل سازی مداری مدل را نیز ببینیم، چون همان طور که مشخص است این رابطه یک معادله قانون اهم [24] است.



شکل ۱۵: مدار معادل مدل هاجکین-هاکسلی [21]

در مدار شکل ۱۵ هر شاخه نشان دهنده یک یون است. همان طور که در شکل می بینید شاخه هر یون شامل یک مقاومت متغیر و یک منبع ولتاژ است. البته شاخه L مربوط به بخش نشستی جریان ناشی از نشستی غشا است. اگر بار دیگر به شکل ۵ دقت کنید می بینید که در بارهای مثبت و منفی در دو سمت غشا جمع شده اند پس غشا رفتاری خازنی دارد که در مدار شکل ۱۵ با خازن  $C_m$  مدل شده است. همان طور که گفته شد رسانایی هر کانال یونی وابسته به ولتاژ غشا است. در ادامه روابط هاجکین - هاکسلی آورده شده است.

$$I = C_m \frac{dV_m}{dt} + \bar{g}_K n^4 (V_m - E_K) + \bar{g}_{Na} m^3 h (V_m - E_{Na}) + \bar{g}_L (V_m - E_L) \quad (\text{معادله ۳})$$

$$\frac{dn}{dt} = \alpha_n (1 - n) - \beta_n n$$

$$\frac{dm}{dt} = \alpha_m (1 - m) - \beta_m m$$

$$\frac{dh}{dt} = \alpha_h (1 - h) - \beta_h h$$

$$\alpha_n = 0.01 \frac{V_m + 55}{1 - \exp\left(\frac{-(V_m + 55)}{10}\right)}$$

$$\alpha_m = 0.1 \frac{V_m + 40}{1 - \exp\left(\frac{-(V_m + 40)}{10}\right)}$$

$$\alpha_h = 0.07 \exp\left(\frac{-(V_m + 65)}{20}\right)$$

$$\beta_n = 0.125 \exp\left(\frac{-(V_m + 65)}{80}\right)$$

$$\beta_m = 4 \exp\left(\frac{-(V_m + 65)}{18}\right)$$

$$\beta_h = \frac{1}{1 + \exp\left(\frac{-(V_m + 35)}{10}\right)}$$

مقادیر پارامترهایی که هاجکین و هاگسلی برای معادلات بالا به دست آوردند [23]:

$$C_m = 1 \mu F \text{ cm}^{-2}$$

$$E_{Na} = 50 \text{ mV}, \quad \overline{g_{Na}} = 120 \text{ mS cm}^{-2}$$

$$E_K = -77 \text{ mV}, \quad \overline{g_K} = 36 \text{ mS cm}^{-2}$$

$$E_L = -54.4 \text{ mV}, \quad \overline{g_L} = 0.3 \text{ mS cm}^{-2}$$

همان طور که احتمالاً از معادلات بالا حدس می‌زنید شبیه‌سازی کامپیوتری آنها بسیار سنگین است و برای مصارف محاسباتی مناسب نیست، اما به جهت دقت شبیه‌سازی مدل HH بی‌نظیر است که با توجه به ابداع آن در سال ۱۹۵۲ و امکانات آن زمان نشان‌دهنده ارزش بی‌نظیر کار هاجکین و هاگسلی است.

نکته مهم دیگری که باید با آن اشاره کرد این است که این مدل نورون‌ها را به‌عنوان یک نقطه بدون بعد و صلب در نظر می‌گیرد و خواص مکانی را لحاظ نمی‌کند. این در حالی است که خواص مکانی نیز در انتشار اسپایک اثرگذارند.

### ۲-۳-۲ مدل ایژیکویچ

مدل/ایژیکویچ<sup>۱</sup> که توسط ریاضی‌دانی با همین نام ارائه شده [25] در واقع نسخه‌ای است ساده شده از مدل HH. در این مدل تعداد معادلات دیفرانسیل به ۲ عدد کاهش یافته اما همچنان خروجی دقت بسیار خوبی دارد. در ادامه رابطه ریاضی این مدل آمده است.

$$\begin{cases} \frac{dV_m}{dt} = k(V_m - E_m)(V_m - V_{th}) - u + I \\ \frac{du}{dt} = a(b(V_m - E_m) - u) \end{cases} \quad (\text{معادله ۴})$$

$$\text{if } V_m \geq 30 \text{ mV, then } \begin{cases} V \text{ is set to } c \\ u \text{ is reset to } u + d \end{cases}$$

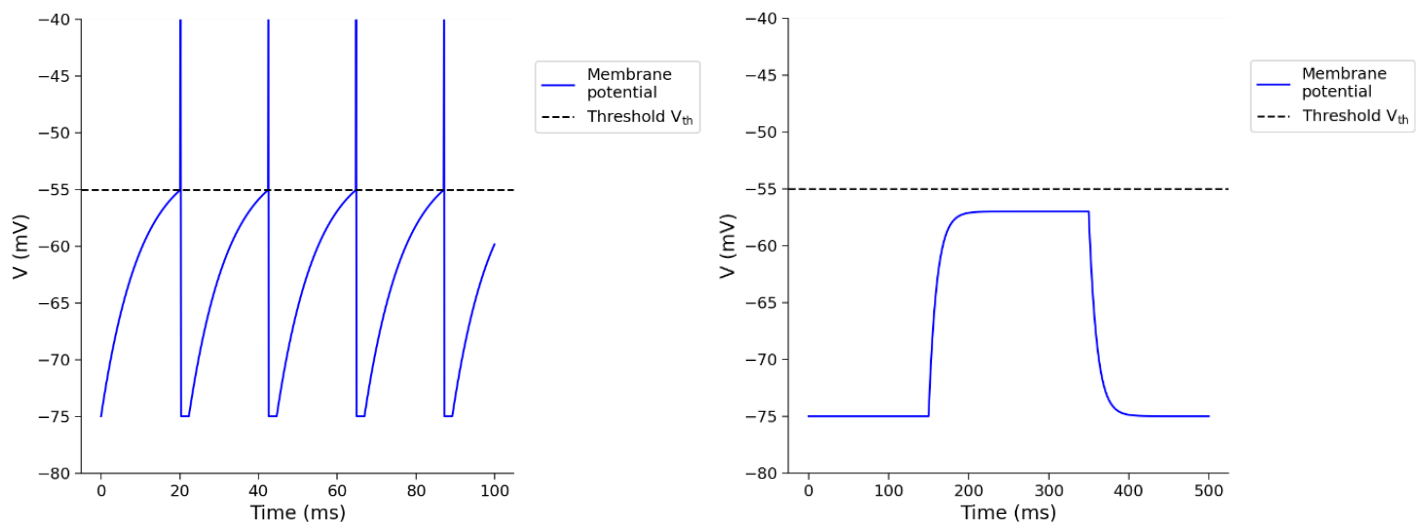
در این مدل پارامترهای  $k, a, b, c, d$  با توجه به نورونی که دنبال مدل کردن آن هستیم انتخاب می‌شوند. این مدل بین دانشمندان محبوبیت بالایی دارد زیرا هم دقیق است و هم آسان.

### ۲-۳-۳ مدل LIF

مدل  $LIF$  ساده‌ترین مدل نورونی است که کماکان شکل موج کلی اسپایک بیولوژیک را تا حدی حفظ می‌کند و در عین حال شبیه‌سازی آن بسیار ساده است، چراکه فقط با یک معادله دیفرانسیل توصیف می‌شود. این مدل هم حالت بسیار ساده‌ای از مدل هاجکین - هاگسلی است. به این صورت که تمام پارامترهای کانال‌های یونی از آن حذف شده‌اند و فقط پارامتر نشتی مانده است و عملکرد اسپایک زدن توسط یک شرط انجام می‌شود.

$$R_L I = \tau_m \frac{dV_m}{dt} + (V_m - E_{rest}) \quad (\text{معادله ۵})$$

$$\text{if } V_m = V_{th}, \text{ then fire and } V_m(t + \Delta) = E_{rest}$$



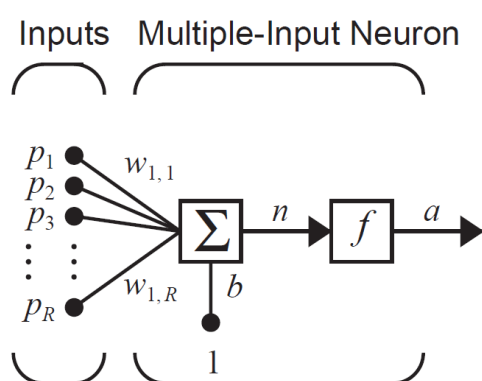
شکل ۱۶: نتیجه شبیه‌سازی نورون LIF در پایتون

که در آن  $\tau_m$  ثابت زمانی خازن مدل کننده غشا است و دلتا مقدار زمانی است که نوروں پس از اسپایک خود نمی‌تواند اسپایک دیگری بزند که اصطلاحاً به آن دوره مقاومتی<sup>۱</sup> می‌گویند. دوره مقاومتی نام دیگری است برای هایپرپلاریزاسیون که پیش‌تر آن را معرفی کردیم.

این مدل به زبان پایتون<sup>۲</sup> در فایل LIFNeuronModel شبیه‌سازی شده است که در شکل ۱۶ نتیجه آن را مشاهده می‌کنید. در سمت راست می‌بینید که ولتاژ غشا به آستانه خود نرسیده بنابراین اسپایک نمی‌زند. این بخش رفتاری کاملاً مشابه خازنی دارد که شارژ و سپس با قطع شدن تحریک، دشارژ می‌شود. اما در شکل سمت چپ حالتی را مشاهده می‌کنید که ولتاژ غشا از آستانه بیشتر شده، بنابراین نوروں اسپایک زده است.

## ۲-۳-۴ مدل پرسپترون

پرسپترون<sup>۳</sup> ساده‌ترین و رایج‌ترین مدل نورونی است [26] که عملکرد اصلی آن استفاده در مسائل محاسباتی خصوصاً یادگیری ماشینی<sup>۴</sup> می‌باشد [27] و می‌توان گفت برای شبیه‌سازی فرآیندهای زیستی هیچ ارزشی ندارد.



شکل ۱۷: مدل پرسپترون [27]

شکل ۱۷ مدل پرسپترون را نشان می‌دهد. عملکرد این مدل بسیار ساده است. به این صورت که ابتدا هر ورودی در ضریب وزن خاص خود (در واقع مدل‌سازی قدرت سیناپس) ضرب می‌شود، سپس همه این اعداد با یکدیگر جمع شده و از تابع فعال‌سازی<sup>۵</sup> می‌گذرند. با توجه به توضیحات می‌توان رابطه زیر را نوشت:

$$a = f \left( \sum_{i=1}^R p_i w_{1,i} \right) \quad (\text{معادله ۶})$$

<sup>۱</sup> Refractory period

<sup>۲</sup> Python

<sup>۳</sup> Perceptron

<sup>۴</sup> Machine Learning

<sup>۵</sup> Activation function

توابع متفاوتی به عنوان تابع فعال سازی پیشنهاد شده اند که از معروف ترین آن ها می توان به تابع خطی، تابع پله<sup>۱</sup>، تابع رلو<sup>۲</sup> و تابع سیگموئید<sup>۳</sup> اشاره کرد.

مشکل اصلی پرسپترون این است که در مدل سازی اش برای زمان هیچ نقشی قائل نشده. در حالی که زمان در دینامیک نورونی بسیار مهم است. نکته مهم دیگر این است که پرسپترون با اعداد کار می کند اما مدل های دقیق تر، مانند آن ها که پیش از این ارائه شدند، با اسپایک کار می کنند.

## ۲-۴ مدل سازی سیناپس

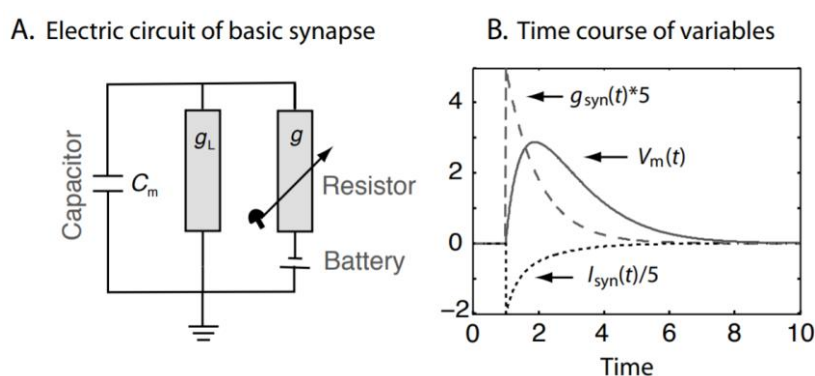
در این بخش به برخی روش های مدل سازی سیناپس می پردازیم. در ابتدا ممکن است تصور شود که مدل سازی سیناپس کار بسیار ساده ای است اما واقعیت این است که اگر به دنبال مدل سازی دقیق باشیم، از نورون به مراتب دشوارتر است.

### ۲-۴-۱ سیناپس به عنوان یک ضریب

ساده ترین مدلی که می توان برای سیناپس در نظر گرفت یک ضریب بین یک تا منفی یک است. این در واقع همان روش مدل سازی سیناپس در پرسپترون است. این مدل سازی خاصیت تضعیف سیناپس و همچنین سیناپس های منفی را در نظر می گیرد (به خاطر حضور اعداد منفی). اما خبری از مدل سازی تأخیر زمانی سیناپس و دینامیک کانال ها نیست.

### ۲-۴-۲ مدلی دقیق تر از سیناپس

در این روش سیناپس مانند مدار شکل ۱۸ مدل می شود که در آن، خازن تأخیر زمانی و مقاومت متغیر باز و بسته شدن کانال های سیناپسی توسط نوروترنسمیترها را مدل می کند.



شکل ۱۸: مدل مداری سیناپس و پاسخ آن [19]

<sup>۱</sup> Step function

<sup>۲</sup> ReLU function

<sup>۳</sup> Sigmoid function

بار دیگر می‌توان مانند مدل نورونی HH، از قانون جریان کیرشهف استفاده کرد و جریان گذرنده از سیناپس را نوشت:

$$I_{syn} = C_m \frac{dV_m}{dt} + g_L V_m - g_{syn} (V_m - E_{syn}) \quad (\text{معادله ۷})$$

که در آن دینامیک رسانایی سیناپس ( $g_{syn}$ ) توسط رابطه زیر تعریف می‌شود:

$$\tau_{syn} \frac{dg_{syn}}{dt} = -g_{syn} \quad (\text{معادله ۸})$$

رفتار  $g_{syn}$  در شکل مشخص است. ابتدا با باز شدن کانال‌ها، بالا می‌رود و سپس به صورت نمایی کم می‌شود. در حالت ساده‌تر می‌توان سیناپس را فقط با  $g_{syn}$  مدل کرد.

## ۲-۵ شبکه عصبی

حال که مدل سازی نورون و سیناپس را بررسی کردیم وقت آن رسیده که با استفاده از این دو عنصر شبکه عصبی<sup>۱</sup> یا نورومورفیک بسازیم. برای ساخت شبکه عصبی انتخاب سه مورد ضروری است: انتخاب مدل نورون، انتخاب مدل سیناپس و انتخاب خود شبکه یعنی شیوه اتصال این عناصر به یکدیگر. درباره این سه انتخاب در فصل بعد مفصلاً توضیح خواهیم داد. اما در اینجا خوب است که چند شبکه عصبی معروف را معرفی کنیم.

### ۲-۵-۱ شبکه عصبی مصنوعی

رایج‌ترین نوع شبکه عصبی که فقط استفاده‌های محاسباتی دارد، شبکه عصبی مصنوعی<sup>۲</sup> (ANN) نام دارد [27]. در این نوع شبکه از مدل نورونی پرسپترون استفاده میشود و سیناپس‌های نیز صرفاً وزن هستند. اما این نوع شبکه می‌تواند معماری‌های متفاوتی داشته باشد.

در شکل ۱۹ یک ANN سه‌لایه را می‌بینید. اگر تعداد لایه‌ها افزایش یابد به این نوع شبکه‌ها اصطلاحاً شبکه عصبی عمیق و به تکنیک یادگیری آن‌ها، یادگیری عمیق<sup>۳</sup> [28] می‌گویند. امروزه این نوع شبکه‌های عصبی کاربردها بی‌شماری در یادگیری ماشینی از خود نشان داده‌اند و یک از داغ‌ترین مباحث حوزه‌های

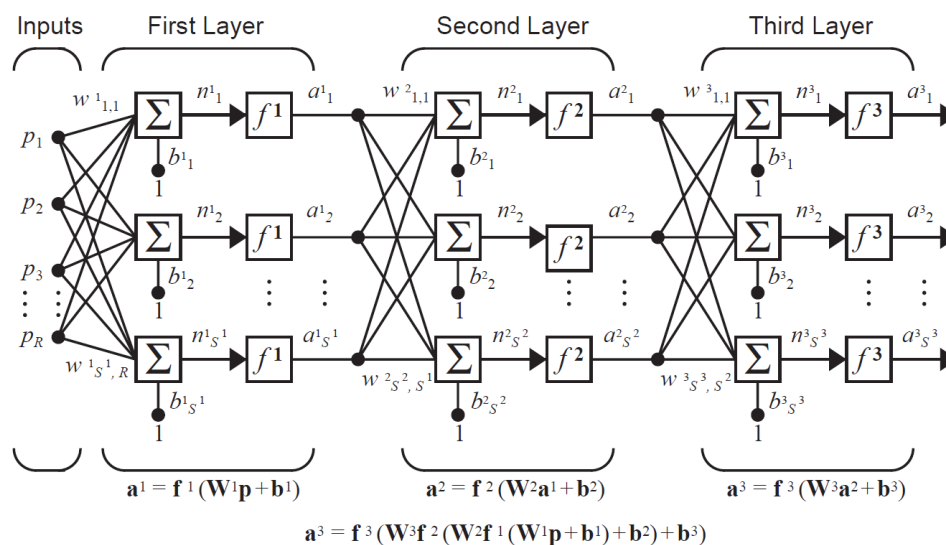
---

<sup>۱</sup> Neural Network

<sup>۲</sup> Artificial Neural Network

<sup>۳</sup> Deep Learning

پژوهشی و فناوری هستند. بررسی این موفقیت‌ها موضوع این نوشته نیست. در صورت علاقه می‌توانید به [29] مراجعه کنید. دقت کنید که همچنان این شبکه‌ها قدرت مدل‌سازی زمانی فعالیت عصبی را ندارند.



شکل ۱۹: شبکه عصبی مصنوعی سه لایه [27]

## ۲-۵-۲ شبکه عصبی اسپایکی

شبکه عصبی/اسپایکی<sup>۱</sup> (SNN) [30] که موضوع اصلی بحث ما هستند و واژه نورومورفیک بیشتر به آن‌ها اطلاق می‌شود، قدرت مدل‌سازی بسیار بالایی دارند چرا که در آن‌ها از مدل‌های نورونی دقیق‌تری مانند مدل ایژیکویچ و مدل LIF استفاده می‌شود. همچنین مدل سیناپسی آن‌ها نیز صرفاً یک وزن ساده نیست و از مدل‌های دقیق‌تری مانند آن که در بخش‌های قبل توضیح داده شد استفاده می‌شود. مدل‌هایی که SNN‌ها برای شبکه استفاده می‌کنند بسیار متنوع هستند و عموماً از مدل‌های واقعی در مغز الهام گرفته‌اند.

نکته مهم دیگر این است که شیوه کدینگ اطلاعات در SNN مانند مغز، قطار/اسپایک<sup>۲</sup> است.

در فصل سوم کاربردهای این نوع شبکه عصبی بیشتر بیان خواهد شد. همچنین نمونه یک SNN کامل و قدرت پردازشی آن را در فصل چهارم خواهید دید.

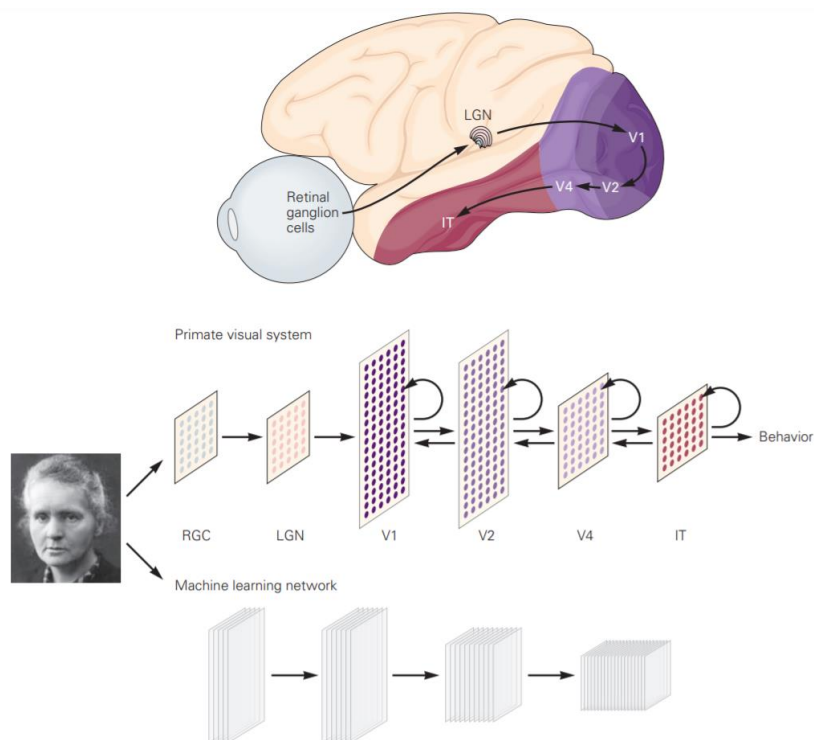
<sup>۱</sup> Spiking Neural Network

<sup>۲</sup> Spike train



## ۲-۵-۳ برخی دیگر از شبکه‌های عصبی

مدل‌های بسیار دیگری نیز برای شبکه‌های عصبی معرفی شده‌اند و در کل این حوزه بسیار مهم و داغ است. یکی از مدل‌هایی که اشاره به آن می‌تواند کمک‌کننده باشد شبکه عصبی پیچشی<sup>۱</sup> است. این نوع شبکه از سیستم بینایی الهام گرفته است و موفقیت‌های بسیار خوب آن در حوزه بینایی ماشینی<sup>۲</sup> [31]، نشان دهنده اهمیت الهام گرفتن از مغز برای پردازش می‌باشد. شمایی از این مدل را در کنار منبع الهام آن (سیستم بینایی در مغز) در شکل ۲۰ مشاهده می‌کنید.



شکل ۲۰: مقایسه شبکه عصبی پیچشی و ساختار کورتکس بینایی [14]

## ۲-۶ مدل‌سازی فرایند یادگیری

یکی از مهم‌ترین و چالش‌برانگیزترین عملکردهای اساسی مغز که نیاز است تا به مدل خود (خصوصاً برای کارهای یادگیری ماشینی) اضافه کنیم، مبحث الگوریتم‌های یادگیری<sup>۳</sup> هستند. اما پیش از آن لازم است تا نگاه کوتاهی داشته باشیم به انواع الگوریتم‌های یادگیری در حوزه یادگیری ماشینی و شناسایی الگو<sup>۴</sup>.

<sup>۱</sup> Convolutional Neural Networks

<sup>۲</sup> Machine Vision

<sup>۳</sup> Learning Algorithms

<sup>۴</sup> Pattern Recognition

## ۲-۶-۱ انواع یادگیری در یادگیری ماشینی

در یادگیری ماشینی، یادگیری به سه بخش: *Supervised*، *Unsupervised* و سایر روش‌ها تقسیم می‌شود [32]. در ادامه اجمالاً هر کدام از این سه دسته را معرفی می‌کنیم.

- یادگیری *Supervised*: در این نوع یادگیری داده‌هایی از قبل در اختیار ماشین قرار گرفته‌اند و ماشین پاسخ دسته‌بندی صحیح آن‌ها را نیز می‌داند. سپس با توجه به این داده‌ها که اصطلاحاً به آن‌ها *داده‌های آموزشی*<sup>۱</sup> می‌گویند، ماشین فرایند آموزش را انجام می‌دهد. سپس این آموزش با داده‌های دیگری که اصطلاحاً *داده‌های تست*<sup>۲</sup> نام دارند ارزیابی می‌شوند. ذکر مثالی می‌تواند مفید باشد. فرض کنید می‌خواهیم ماشینی بسازیم که بتواند تشخیص دهد آیا در یک تصویر چهره‌ای وجود دارد یا خیر. برای این منظور اگر روش *Supervised* را انتخاب کنیم، باید ابتدا تعداد زیادی از تصاویر را به ماشین نشان بدهیم و بگوییم که تصویر چهره دارد یا خیر. پس از آن اگر فرایند آموزش ماشین موفقیت‌آمیز باشد، می‌تواند پس از گرفتن تصویر حضور چهره در آن را با دقت خوبی پیش‌بینی کند. از معروف‌ترین الگوریتم‌های این روش می‌توان به *ماشین بردار پشتیبان*<sup>۳</sup> (SVM) [33] و بسیاری از شبکه‌های عصبی مصنوعی اشاره کرد.
  - یادگیری *Unsupervised*: در این روش برخلاف روش قبل، داده‌های آموزشی وجود ندارند و ماشین باید داده‌ها را بر اساس میزان شباهتشان با یکدیگر خوشه‌بندی کند. اگر همان مثال قبل را بخواهیم با این روش حل کنیم باید تمام عکس‌ها را به ماشین داده و انتظار داشته باشیم تا بتواند بر اساس میزان شباهت آن‌ها با یکدیگر خوشه‌بندی را انجام دهد. اما بزرگ‌ترین چالش در این الگوریتم سؤال مهم *کدام شباهت* است. برخی از انواع شبکه عصبی زیرمجموعه این دسته قرار می‌گیرند. دیگر روش محبوب در این دسته، خوشه‌بند *k-means* است [34].
  - سایر روش‌های یادگیری: علاوه بر دو روش قبل، روش‌های بسیار زیاد دیگری نیز برای آموزش یک ماشین وجود دارند که یا مانند یکی از روش‌های قبل هستند اما با کمی تغییر (مانند آنچه در فصل چهارم خواهید دید) یا تلفیقی از دو روش قبلی هستند (مانند *Semi-Supervised*) یا روش عملکرد آن‌ها به کل متفاوت است (مانند *یادگیری تعاملی*<sup>۴</sup> [35]).
- پیش از پرداختن به یادگیری در مغز، به یکی از روش‌های مهم یادگیری در ANN‌ها می‌پردازیم.

---

<sup>۱</sup> Training Data

<sup>۲</sup> Test Data

<sup>۳</sup> Support-Vector Machine

<sup>۴</sup> Reinforcement Learning

## ۲-۶-۲ الگوریتم پس انتشار

الگوریتم پس انتشار<sup>۱</sup> [27] مهم‌ترین و رایج‌ترین الگوریتم یادگیری در شبکه‌های عصبی مصنوعی می‌باشد. در این روش ابتدا یک تابع خطا تعریف می‌شود و با توجه به آن و همچنین خروجی شبکه، مقدار خطا محاسبه می‌شود. سپس سعی می‌شود تا با تغییر وزن‌ها، این تابع خطا به کمینه خود برسد. پس در این روش ما با یک مسئله بهینه‌سازی مواجه هستیم. برای این منظور باید تابع خطا را برحسب وزن‌ها نوشته و برای رسیدن به کمینه از آن مشتق بگیریم. پرواضح است که این روش بار محاسباتی زیادی دارد و ارتباطی با روش یادگیری در مغز ندارد. همچنین باید گفت که این روش Supervised است.

## ۲-۶-۳ یادگیری در مغز

ریشه یادگیری در مغز پدیده‌ای است به نام نوروپلاستیسیته<sup>۲</sup> [36]. این پدیده بی‌نظیر به مغز امکان انعطاف پذیری می‌دهد. به این معنا که مغز می‌تواند ساختار خود (ارتباطات سیناپسی و...) را تغییر دهد. این موضوع ریشه اصلی فرآیندهای حافظه و یادگیری در می‌باشد. پس مغز ما ثابت نیست و هر روز تغییر می‌کند! برای فهم بهتر این ویژگی خوب است تا به نام کتابی اشاره کنیم که دیوید ایگلمن<sup>۳</sup>، یکی از عصب‌پژوهان برجسته معاصر درباره این پدیده نوشته است [37]. ایگلمن از واژه Livewired برای نام گذاری کتابش استفاده کرده. همانطور که مشخص است این واژه ترکیب live به معنای زنده و پویا، و همچنین wired به معنای سیم‌کشی شده می‌باشد. پس همانطور که ایگلمن سعی دارد بگوید، سیم‌کشی مغز ما پویا، زنده و متغیر است. در ادامه نگاهی خواهیم داشت به یکی از مدل‌سازی‌های مهم از فرآیند یادگیری در مغز.

## ۲-۶-۴ پلاستیسیته وابسته به زمان اسپایک (STDP)

پلاستیسیته وابسته به زمان اسپایک<sup>۴</sup> (STDP) [38] یکی از روش‌های مدل‌سازی فرآیند نوروپلاستیسیته و یادگیری در مغز است. این روش انواع و اقسام بسیار متنوع و گوناگونی دارد که فهرست خوبی از آن‌ها در [20] تهیه شده است. در اینجا ما یکی از اقسام ساده‌تر و کاربردی آن را با فرمول‌بندی [21] ارائه می‌دهیم.

پیش از ارائه فرمول‌بندی، خوب است تا مفهوم STDP را بیان کنیم. یک سیناپس تقویت می‌شود، اگر نورون پیش‌سیناپسی در حدود نهایتاً ۵۰ میلی‌ثانیه قبل از نورون پس‌سیناپسی، اسپایک بزند. متقابلاً اگر در

---

<sup>۱</sup> Backpropagation

<sup>۲</sup> Neuroplasticity

<sup>۳</sup> David Eagleman

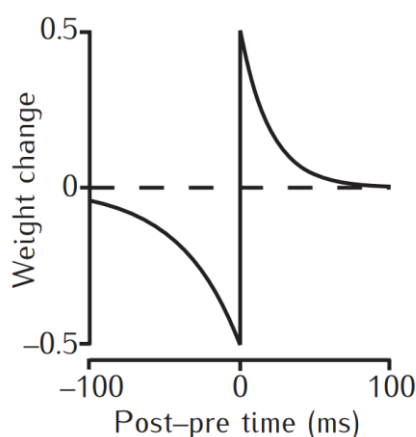
<sup>۴</sup> Spike-timing-dependent plasticity

زمان کوتاهی بعد از نورون پس‌سیناپسی اسپایک بزند، سیناپس تضعیف می‌شود. در واقع این پدیده که بارها در مغز کودکان (مغز در حال رشد و تغییر فراوان) و همچنین مغز بزرگسالان دیده شده سعی دارد تو سیناپس‌های اثرگذار رو قوی‌تر کرده و سیناپس‌های بی‌اثر را تضعیف کند. در زیر رابطه STDP آورده شده است:

$$\text{STDP: } \begin{cases} \text{LTP: } \Delta w_{ij} = A_{LTP} \exp\left(\frac{-\Delta t}{\tau_{LTP}}\right), & \text{if } \Delta t \geq 0 \\ \text{LTD: } \Delta w_{ij} = -A_{LTD} \exp\left(\frac{\Delta t}{\tau_{LTD}}\right), & \text{if } \Delta t \leq 0 \end{cases} \quad \text{(معادله ۹)}$$

,where  $\Delta t = t_{post} - t_{pre}$

رابطه بالا دو بخش دارد، یکی مربوط به *long-term potentiation* یا به اختصار LTP که عامل تقویت وزن سیناپسی است و دیگری *long-term depression* یا به اختصار LTD که عامل تضعیف وزن سیناپسی است. در این رابطه  $\Delta t$  اختلاف بین زمان اسپایک نورون پس‌سیناپسی با نورون پیش‌سیناپسی است. متغیر  $w_{ij}$  وزن سیناپسی اتصال نورون  $i$  به نورون  $j$  است. این عدد در واقع همان مقدار بیشینه  $g_{syn}$  در شکل ۱۸ است که پس از اسپایک نورون پیش‌سیناپسی، مقدار  $g_{syn}$  به آن رسیده و سپس به صورت نمایی افت می‌کند. البته دقت کنید که در این روابط  $\Delta w_{ij}$  یعنی مقدار تغییر آن محاسبه می‌شود مقادیر  $A_{LTP}$ ،  $A_{LTD}$  دامنه این



شکل ۲۱: نمودار تغییر وزن‌ها طبق الگوریتم STDP [21]

اثرگذاری هستند (معمولاً  $A_{LTP} > A_{LTD}$ ). همچنین  $\tau_{LTP}$  و  $\tau_{LTD}$  نیز ثابت زمانی‌های این رابطه هستند.

## ۷-۲ نتیجه‌گیری

در این فصل در راستای نتیجه‌گیری فصل قبلی مبنی بر اهمیت شناخت سیستم عصبی، ابتدا درباره بیولوژی سیستم عصبی با تمرکز بر نورون و سیناپس صحبت کردیم. سپس تلاش کردیم تا مدل‌سازی‌های معروف این دو را معرفی کرده و آن‌ها کاملاً شرح دهیم. پس از آن کمی درباره انواع شبکه‌های عصبی و همچنین الگوریتم‌های یادگیری در یادگیری ماشینی و مغز صحبت کرده و یکی از مهم‌ترین آن‌ها در مغز را مدل کردیم

## فصل سوم: مقدمه‌ای بر محاسبات نورومورفیک

### ۳-۱ مقدمه

تا به اینجا به ساختار سیستم عصبی و نحوه مدل‌سازی آن و همچنین برخی معماری شبکه‌ها آشنا شده‌ایم. حال می‌توانیم به هدفی که در فصل اول برای خود تعریف کردیم، یعنی ساختن پردازنده‌ای با الهام گرفتن از معماری مغز بپردازیم. در این فصل این اطلاعات را جمع‌بندی کرده و به صورت خاص درباره کارکردها، مزایا و معایب، طراحی و ساخت پردازنده نورومورفیک صحبت خواهیم کرد [39].

واژه نورومورفیک اولین بار در سال ۱۹۹۰ توسط کارور مید<sup>۱</sup> استفاده شد [40]. مید این واژه را برای آی‌سی  $VLSI$ <sup>۲</sup> آنالوگی به کار برد که رفتار سیستم عصبی را تقلید میکرد. به همین دلیل از واژه نورومورفیک استفاده کرد. البته همانطور که پیش‌تر گفته شد ایده استفاده از معماری مغز به فون‌نویمان [9] و تورینگ [41] بر میگردد. کار بر روی نورومورفیک نیازمند ورود دانشمندان و مهندسانی از بسیاری حوزه‌ها از جمله علوم اعصاب، مهندسی برق و کامپیوتر، شیمی و مواد می‌باشد. چرا که می‌خواهیم با کمک عصب پژوهان مغز را برای گرفتن ایده‌های نورومورفیکی بهتر، دقیق‌تر بشناسیم. سپس با کمک مهندسی کامپیوتر این پردازنده را طراحی کرده و در نهایت توسط مهندسی برق آن را پیاده کنیم. البته در این مسیر نیاز به تحقیقات برای ساختن مواد جدید نیز می‌باشد.

پیش از شروع لازم است تا این نکته نیز ذکر شود که طبق تعریف هر ساختار الهام گرفته از مغز، نورومورفیک است؛ بنابراین حتی ANN هم نورومورفیک حساب می‌شود. اما معمولاً واژه نورومورفیک برای ساختارهایی استفاده می‌شود که بر اساس انواعی از SNNها هستند.

ساخت پردازنده نورومورفیک می‌تواند برای رسیدن به اهداف متفاوتی باشد. به طور کلی می‌توان دو کارکرد اساسی برای پردازنده‌های نورومورفیک تعریف کرد.

### ۳-۲ کارکرد پژوهشی

یکی از کاربردهای مهم نورومورفیک که مخصوص به آن است، استفاده به عنوان ابزاری پژوهشی در علوم اعصاب است. چرا که در علوم اعصاب لازم است تا با شبیه‌سازی سیستم عصبی کارکردهای آن را بررسی کرده و اعتبار نظریه‌های خود را بسنجیم. اما شبیه‌سازی سیستم عصبی بادقت کافی بر روی معماری فون‌نویمان بسیار کند است. اما چون پردازنده نورومورفیک فی‌الذات ساختاری شبیه مغز دارد می‌تواند سرعت پردازش

<sup>۱</sup> Carver Mead

<sup>۲</sup> Very Large-Scale Integration

و شبیه‌سازی را به شدت افزایش دهد [42]. دقیقاً به همین علت است که بسیاری از پروژه‌های بزرگ حوزه علوم اعصاب از جمله *Human Brain Project* اتحادیه اروپا [43]، تعداد زیادی پروژه‌های نورومورفیک را راه‌اندازی کرده‌اند.

نکته جالب بعدی دیدن چگونگی کارکرد پردازنده‌های نورومورفیک و الهام گرفتن از آن‌ها برای توضیح مغز است که می‌تواند برای دانشمندان حوزه علوم اعصاب نظری و محاسباتی بسیار جذاب باشد.

### ۳-۳ کارکرد محاسباتی

دیگر دلیل سرمایه‌گذاری بر روی نورومورفیک، ویژگی‌های خاص محاسباتی آن است. همان گونه که در فصل اول توضیح داده شد به دلیل رو به اتمام بودن اعتبار قانون مور لازم است تا راه حلی برای آن پیدا کنیم. علاوه بر این موضوع خود پردازنده‌های مبتنی بر معماری فون‌نویمان نیز مشکلات فراوان خود را دارند مانند پهنای باند کم بین CPU و حافظه. پس این نیز خود دلیلی برای سرمایه‌گذاری روی نورومورفیک است [44]. پردازنده‌های نورومورفیک چند ویژگی مهم محاسباتی دارند.

(۱) توان مصرفی کم: این ویژگی مهم‌ترین نکته سرمایه‌گذاری بر روی پردازنده‌های نورومورفیک در سال‌های اخیر است. ریشه این توان مصرفی کم به خود مغز برمی‌گردد. توان مصرفی مغز ما با تمام قدرت‌های پردازشی بی‌نظیرش فقط در حدود ۲۰ تا ۲۵ وات است! این ویژگی نشان‌دهنده کم‌مصرف بودن این معماری در ذات خود است. برخی از پردازنده‌های نورومورفیک ساخته شده برای هر اسپایک فقط در حدود چند نانو ژول یا حتی در برخی موارد چند پیکو ژول انرژی مصرف می‌کنند که بسیار عالی است [45].

(۲) پردازش موازی: یکی از اولین انگیزه‌ها برای توسعه پردازنده‌های نورومورفیک، توانایی آن‌ها در پردازش موازی بود. این توانایی با دیدن مثال فصل بعد بیشتر مشخص می‌شود اما اجمالاً با نگاه به ساختار شبکه عصبی می‌بینید که نورون‌ها از طریق راه‌های متفاوت امکان پردازش هم‌زمان ورودی‌ها را دارند.

(۳) مقیاس‌پذیری: دیگر ویژگی خاص نورومورفیک مقیاس‌پذیری<sup>۱</sup> آن است. به این معنا که چون ساختار شبکه‌ای دارد، بزرگ کردن ابعاد آن کار راحتی است. مانند راحتی اضافه کردن لایه‌های پنهان در ANN.

۴) یادگیری آنلاین<sup>۱</sup> و عملکرد در لحظه<sup>۲</sup>: یکی از نیازهای بسیار مهم در دنیای امروزه، استفاده از پردازنده‌هایی است که قدرت عملکرد در لحظه و همچنین یادگیری آنلاین را دارند. بازم به اگر به رفتار موجودات هوشمند توجه کنیم می‌بینیم که هر دوی این ویژگی‌ها در رفتار موجودات وجود دارد. پس عجیب نیست اگر فکر کنیم که ساختار نورومورفیک نیز این قابلیت را دارد.

۵) غیره: ویژگی‌های بسیار دیگری از جمله سرعت بیشتر، ابعاد کوچک‌تر و ضریب خطای کمتر نیز وجود دارد که به آن‌ها نمی‌پردازیم.

پس برخی از ویژگی‌های مهم نورومورفیک که انگیزه‌های اصلی محاسباتی هستند را دیدیم.

### ۳-۴ مغز در برابر کامپیوتر

پیش از توضیح روش طراحی یک پردازنده نورومورفیک، خوب است تا مقایسه‌ای داشته باشیم بین مغز و کامپیوتر. (در اینجا منظور از کامپیوتر، کامپیوترهای مبتنی بر معماری فون‌نویمان هستند). در جدول ۱ مقایسه مهم‌ترین تفاوت‌های مغز و کامپیوتر را مشاهده می‌کنید:

جدول ۱: مقایسه کامپیوتر و مغز

حافظه	تخصص	پارادایم پردازش	سرعت	تعداد المان	
جدا	منطقی و ریاضی	ترتیبی و بر اساس بیت	۳.۲ گیگاهرتز (پردازنده Apple M1 Max)	۵۷ میلیارد ترانزیستور (پردازنده Apple M1 Max)	کامپیوتر
مجتمع	شناختی <sup>۴</sup>	موازی و بر اساس اسپایک	حداکثر ۴۵۳ هرتز در نوروهای <sup>۳</sup> FS [46]	۸۶ میلیارد نرون + صد هزار برابر سیناپس	مغز

- تعداد المان: همان‌طور که مشاهده می‌شود تعداد ترانزیستورهای یک IC می‌تواند به تعداد نرون‌ها در مغز نزدیک شده و حتی در مواردی (مانند برخی حافظه‌های پرسرعت) از آن بیشتر بشود. اما نکته مهم درباره مغز این است که فهم ساختار و پیچیدگی آن از نظر تعداد المان، بدون در نظر گرفتن اتصالات (سیناپس‌ها) بی‌معنی است. همان‌طور که می‌بینید و در بخش‌های قبل نیز گفته شد، تعداد سیناپس‌ها آن قدر زیاد است که به عقیده بسیاری مغز، با فاصله‌ی فراوان پیچیده‌ترین سیستم کشف شده توسط انسان در کل عالم است. این مقایسه از منظر دیگری نیز به نفع مغز است چراکه قدرت پردازشی یک نرون به مراتب از یک ترانزیستور بیشتر است.

<sup>۱</sup> Online Learning

<sup>۲</sup> Real-time Performance

<sup>۳</sup> Cortical fast-spiking neurons

<sup>۴</sup> Cognitive

- سرعت: همان طور که مشخص است سرعت ترانزیستورها با سرعت مغز قابل قیاس نیست! مغز به دلیل ساختار بیولوژیک و همچنین استفاده از یون‌ها به‌عنوان حامل و در نهایت سیناپس‌های شیمیایی به نسبت کامپیوترها بسیار کندتر است. البته این کندی با خاصیت پردازش موازی بهبود می‌یابد.
- پارادایم پردازش: احتمالاً مهم‌ترین تفاوت مغز و کامپیوتر، پارادایم پردازش آن‌هاست. کامپیوترها بر اساس بیت کار می‌کنند و پردازش آن‌ها نیز موازی است. اما در مغز همان طور که پیش‌تر گفته شد به دلیل ساختار شبکه‌ای و گرافی آن، پردازش به‌صورت موازی انجام می‌شود. علاوه بر این تمام پردازش‌های مغز بر اساس اسپایک است و اطلاعات در ویژگی‌های فرکانسی و زمانی قطار اسپایک‌ها کد می‌شود. در این باره بیشتر خواهیم گفت.
- تخصص: تخصص کامپیوترها منطق و ریاضیات است. (البته ذکر کردن منطق کافی است چراکه ریاضیات بر خواسته از منطق است) دلیل آن نیز همان طور که در فصل اول دیدیم بیس منطق محور طراحی کامپیوترها است. ما تخصص مغز در عملیات‌های شناختی مانند ادراک، تشخیص چهره، زبان و... است.
- حافظه: در کامپیوترها حافظه به‌صورت جداگانه در کنار CPU قرار می‌گیرد که این موضوع خود از مهم‌ترین مشکلات معماری فون‌نویمان است<sup>۱</sup>. اما در مغز، حافظه نیز به‌صورت توأمان درون ساختار شبکه نورونی قرار می‌گیرد. البته که بخش‌هایی از مغز به‌صورت خاص برای حافظه بهینه‌سازی شده‌اند. اما نکته مهم این است که همه شبکه‌های نورونی در مغز هم ویژگی‌های حافظه محور دارند (به‌واسطه پلاستیسیته سیناپس‌ها) و هم خواص محاسباتی. این ویژگی از مغز بسیار منحصر به فرد است و می‌تواند (در صورت برنامه‌نویسی صحیح) راندمان را بسیار افزایش دهد. البته که این ویژگی چالش بزرگی نیز دارد زیرا فهم آن بسیار سخت است و امروزه مسئله حافظه یکی از مهم‌ترین مسائل علوم اعصاب است.

یک پردازنده نورومورفیک ایدئال می‌تواند در کنار داشتن تمام ویژگی‌های مثبت ذکر شده، سرعت بالا را نیز به این مجموعه اضافه کند. (البته اینکه بتوانیم با تکنولوژی الکترونیک فعلی پیچیدگی مغز از نظر تعداد سیناپس‌ها را مدل کنیم، ناممکن به نظر می‌رسد. مگر استفاده از مغز موجودات ضعیف‌تر)

### ۳-۵ شباهت به بیولوژی یا اهمیت محاسبه؟

سؤال مهمی که ارزش مطرح کردن دارد این است که چرا باید پردازنده را شبیه مدل بیولوژیک آن کنیم؟ در بحث خاص ما می‌توان این سؤال را این‌گونه پرسید: حال که به ویژگی‌ها خاص معماری شبکه عصبی پی

<sup>۱</sup> به این مشکل اصطلاحاً von Neumann bottleneck می‌گویند.



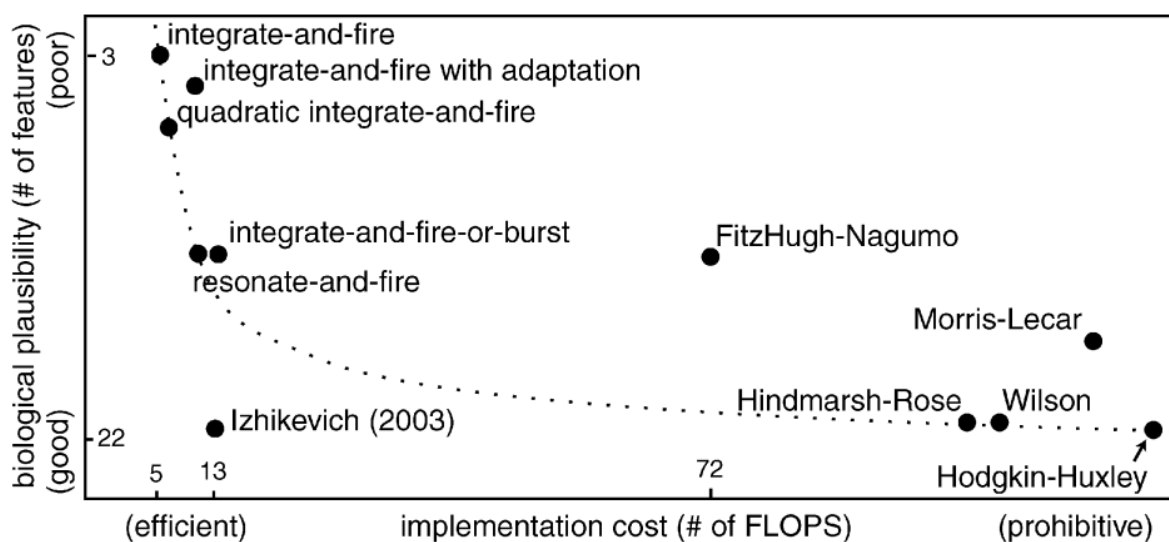
برسیم، چرا از شبکه عصبی اسپایکی استفاده کنیم؟ چرا خودمان با الهام گرفتن از آن شبکه‌هایی که از نظر پردازشی بهینه‌تر هستند نسازیم؟ این سؤالات در واقع در دو سر یک طیف قرار می‌گیرند. طیفی که در یک سر آن اهمیت شباهت به بیولوژی یا اصطلاحاً **Biologically Plausible (BP)** قرار دارد و در سر دیگر آن به صرفه بودن از نظر محاسباتی قرار دارد. این بحث یک منازعه طولانی است و نمی‌توان رأی نهایی داد و در واقع باید هر دو موضوع را در نظر گرفت، اما نکته مهم این است که BP امتحان خودش را در طبیعت پس داده و همچنین دارای ویژگی‌های محاسباتی خاصی مانند آنچه که در بخش‌های قبل ذکر شد است. در کنار این موضوع اهمیت آن در پژوهش‌های علوم اعصاب نیز مهم است. پس در نتیجه استفاده از SNN‌هایی که به بیولوژی شباهت دارند دارای ارزش است و به حتی اعتقاد برخی نسل بعد شبکه‌های عصبی هستند که انقلابی در هوش مصنوعی ایجاد خواهند کرد [30].

### ۳-۶ انتخاب مدل‌های مناسب

حال اگر بخواهیم پردازنده نورومورفیک بسازیم، گام مهم بعدی انتخاب مدل‌ها مناسب است. در فصل قبل چندی از مدل‌های نورونی، سیناپسی و شبکه‌ای را معرفی کردیم. در این بخش آن‌ها را مقایسه می‌کنیم.

#### ۳-۶-۱ انتخاب مدل نورون

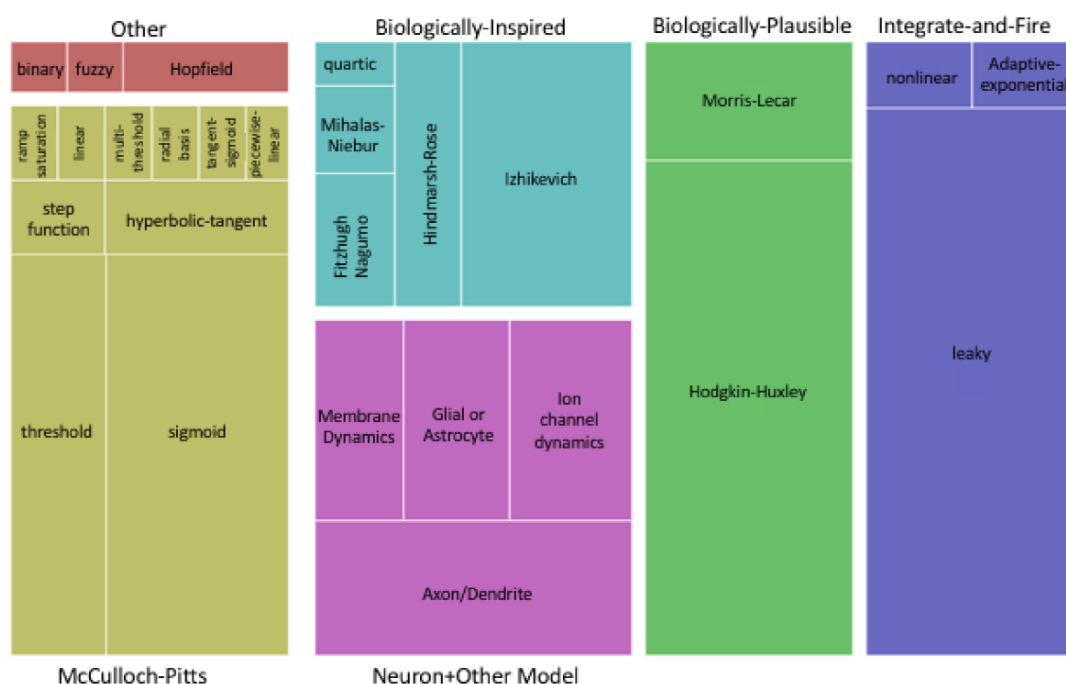
حال که قصد طراحی SNN را داریم بدیهی است که باید مدل نورونی‌ای را انتخاب کنیم که اسپایک بزند.



شکل ۲۲: دقت بایولوژیک مدل‌های نورونی در قیاس به سختی پیاده‌سازی آن‌ها (مقدار محاسبات لازم) [47]

ایژیکویچ در [47] مقایسه خوبی بین انواع مدل‌های نورونی متفاوت انجام داده. شکل ۲۲ که برگرفته از مقاله او است نشان می‌دهد که معمولاً هرچه مدل نورون از نظر بایولوژیک دقت بالاتری داشته باشد و رفتار آن به نورون واقعی نزدیک‌تر باشد، (در رأس آن‌ها مدل HH) پیاده‌سازی آن دشوارتر بوده و توان محاسباتی بیشتری می‌خواهد.

همان طور که از شکل ۲۲ پیداست نورون‌های دسته IF - از جمله LIF که فصل قبل آن را به صورت کامل توضیح دادیم - در عین اینکه همچنان اسپایک می‌زنند دارای کمترین وزن محاسباتی هستند و همین عامل باعث شده تا کاربرد بسیار زیادی در نورومورفیک داشته باشند. این موضوع به خوبی در شکل ۲۳ مشخص است. (اندازه مستطیل‌ها نشان‌دهنده میزان استفاده از آن مدل نورونی است). طبق شکل ۲۳ مدل دیگری که طرفداران زیادی دارد، مدل ایژیکویچ است. همان طور که در شکل ۲۲ دیدیم این مدل در عین نیاز به توان محاسباتی کم، دقت بیولوژیک بالایی دارد. البته باید دقت کرد که مدل ایژیکویچ، فقط معادلات خود را به رفتار نورون فیت کرده و برخلاف مدل HH معادلات آن مدل کننده شیوه ایجاد اسپایک نیست.



شکل ۲۳: انواع مدل‌های نورونی استفاده شده در نورومورفیک ذیل خانواده‌شان. ابعاد مستطیل هر مدل نشان‌دهنده میزان استفاده از آن است [39]

در نهایت باید گفت، طراح باید با توجه به کاربری و امکاناتی که دارد بین Biologically Plausible بودن و بهینه بودن از نظر محاسباتی دست به انتخاب بزند.

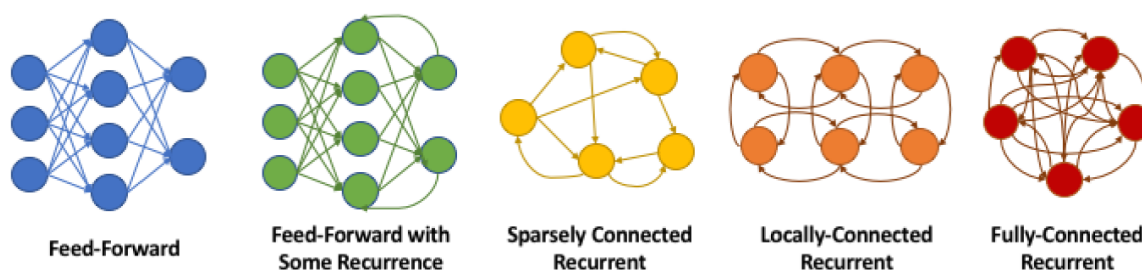
### ۳-۶-۲ انتخاب مدل سیناپس

این انتخاب بسیار به توان محاسباتی موجود وابسته است چرا که تعداد سیناپس‌ها بسیار زیاد خواهد بود. از طرف دیگر این انتخاب بستگی به سخت‌افزار پیاده‌سازی نیز دارد چون ممکن است سخت‌افزار موجود خودش المان‌هایی با خواص ذاتی سیناپس گونه داشته باشد که در ادامه درباره آن توضیح می‌دهیم. نکته مهم دیگر در انتخاب مدل سیناپس، رابطه آن با مدل یادگیری است چون در فرایند یادگیری، سیناپس تغییر می‌کند.

و اما نکته آخر در انتخاب سیناپس، بحث دینامیک زمانی است. اگر شما نیاز به شبکه‌ای دارید که زمانمند باشد، (همه SNNها) مدل سیناپس آن نیز باید دینامیک زمانی داشته باشد. مانند مدل ارائه شده در فصل قبل.

### ۳-۶-۳ انتخاب مدل شبکه

همان طور که در فصل قبل توضیح دادم، انتخاب مدل شبکه اهمیت فراوانی دارد. شبکه مهم‌ترین نقطه‌ای است که سیستم نورومورفیک می‌تواند هر چه بیشتر خود را شبیه مغز بیولوژیک کند. البته که انتخاب مدل شبکه، کاملاً توسط انتخاب سخت‌افزار پیاده‌سازی محدود می‌شود. چراکه بستر پیاده‌سازی تعیین‌کننده حداکثر تعداد اتصالات به هر نورون و همچنین نحوه قرارگیری آن‌ها است.



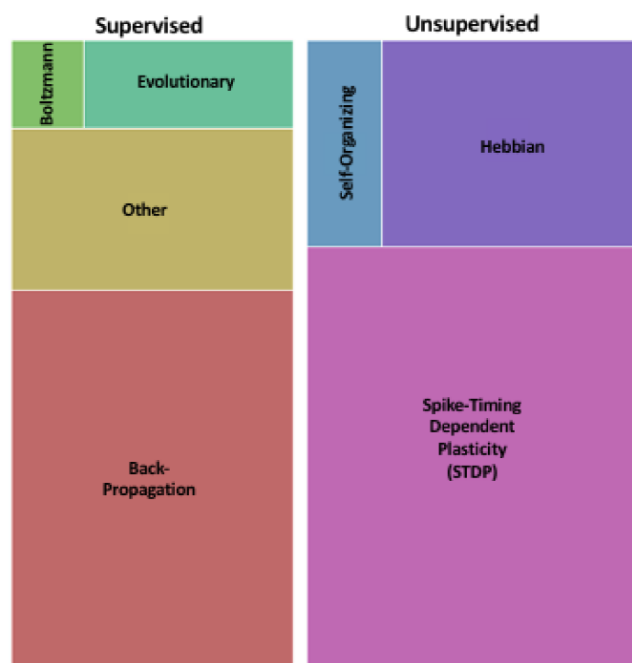
شکل ۲۴: برخی از کرنل‌های شبکه‌ای مهم [39]

شکل ۲۴ نشان‌دهنده چند کرنل شبکه‌ای مهم است. دقت کنید که شبکه می‌تواند ترکیبی از چند کرنل باشد. (مانند شبکه پیاده شده در فصل بعد). نکته مهم دیگر اهمیت وجود فیدبک در شبکه است چرا که وجود آن در پیچیده‌ای است به سمت پیاده‌سازی حافظه در ساختار شبکه.

در پایان باید گفت که تعامل با علوم اعصاب می‌تواند به طراحی شبکه‌های جدید بسیار کمک کند، چراکه یکی از مهم‌ترین وظایف علوم اعصاب کشف ریز مدارها و شبکه‌های مغز است. مثلاً شبکه‌های نورونی استفاده شده در سیستم عصبی، الهام‌بخش بسیاری از شبکه‌های نورومورفیک بوده‌اند.

### ۳-۶-۴ انتخاب الگوریتم یادگیری

مهم‌ترین نکته‌ای که در انتخاب این بخش اثر دارد این است که آیا می‌خواهیم فرایند یادگیری روی خود سخت‌افزار نورومورفیک انجام شود یا آنکه بیرون سخت‌افزار انجام شده و شبکه‌ای با پارامترهای ثابت پیاده شود. نکته بعدی انتخاب بین supervised بودن یا نبودن الگوریتم است که در شکل ۲۵ تقسیم‌بندی آن را مشاهده می‌کنید.



شکل ۲۵: انواع الگوریتم‌های یادگیری استفاده شده در نورومورفیک ذیل خانواده‌شان. ابعاد مستطیل هر مدل نشان‌دهنده میزان استفاده از آن است [39]

یکی از روش‌های رایج در نورومورفیک، ساختن یک ANN و پیاده‌سازی کامل الگوریتم یادگیری روی آن است که سپس با روش‌هایی از جمله تغییر تابع فعال‌سازی به SNN تبدیل می‌شود. اما مهم‌ترین روش‌ها، Unsupervised هستند چون همان‌طور که پیش‌تر گفته شد خود مغز نیز این‌گونه است. از بین روش‌های Unsupervised همان‌گونه که در شکل ۲۵ مشخص است، STDP رایج‌ترین روش است که آن را به طور کامل در فصل قبل بررسی کردیم. البته دوباره لازم به یادآوری است که STDP انواع گوناگونی دارد. البته باید گفت که هنوز برای STDP مجموعه کاربردهایی که در آن‌ها بسیار بهتر از دیگر روش‌ها عمل کند پیدا نشده. طبق گفته دلیلی که ما هنوز نتوانستیم کاربرد اصلی‌ای که نورومورفیک برای آن بهینه‌ترین است را پیدا کنیم، ضعف ما در توسعه الگوریتم یادگیری مناسب است. در واقع ما برای توسعه الگوریتم یادگیری همچنان به سبک معماری فون‌نویمان فکر می‌کنیم. اما لازم است تا از این چارچوب خارج شده و به سبک خود نورومورفیک فکر کنیم و در تئوری محاسباتمان اصطلاحاً یک شیفت پارادایم<sup>۱</sup> ایجاد کنیم. در اینجا هم احتمالاً علوم اعصاب می‌تواند بسیار کمک‌کننده باشد.

### ۳-۷ زبان مغز

همان‌طور که بارها گفته شد زبان مغز و به طبع آن SNN ها، قطار اسپایک است. این نکته شاید بزرگ‌ترین چالش نورومورفیک و از آن مهم‌تر، بزرگ‌ترین چالش علوم اعصاب نظری در فهم مغز است. در واقع در باب

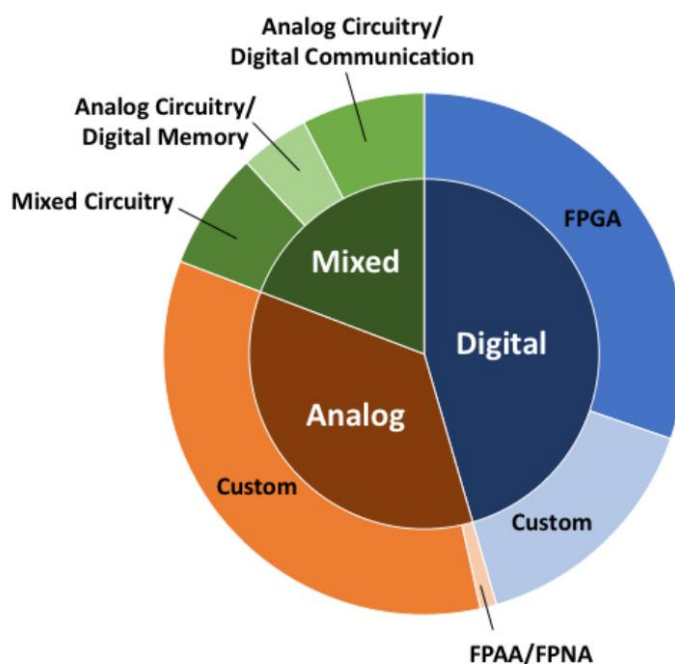
<sup>۱</sup> Paradigm Shift

اهمیت رمزگشایی کامل زبان اسپایک‌ها می‌توان این‌گونه گفت که هر شخصی موفق به دستیابی به این مهم شود عملاً تبدیل به نیوتن علوم اعصاب می‌شود!

این موضوع برای نورومورفیک از دو جنبه چالش‌زاست. اولاً چون زبان پردازش سیستم اسپایکی است، طبیعتاً ورودی‌ها و خروجی‌ها سیستم نیز باید اسپایکی باشند، اما در سنسورهای ما به صورت عدد ثبت می‌گیرند و سایر سیستم‌ها نیز با اعداد - در قالب بیت - کار می‌کنند. پس لازم است تا اعداد را با اسپایک و بالعکس تغییر دهیم. نمونه‌ای از این تغییر را در فصل بعد خواهید دید. چالش بعدی بحث زبان برنامه‌نویسی است. متأسفانه هنوز هیچ چارچوب<sup>۱</sup> برنامه‌نویسی خوبی برای SNN تعریف نشده. در واقع همان‌طور که مایک دیویس<sup>۲</sup> (مسئول آزمایشگاه نورومورفیک شرکت اینتل) می‌گوید، ما هنوز نمی‌دانیم باید دقیقاً به چه شکل برای پردازنده‌های نورومورفیک برنامه‌نویسی کنیم که این بزرگ‌ترین مشکل کنونی این حوزه است [48]. بنظر می‌رسد باید به سراغ تغییر شیوه تفکر خود و نوعی شیفت پارادایم باشیم.

### ۳-۸ بسترهای پیاده‌سازی

نکته مهم دیگر، بستر پیاده‌سازی شبکه نورومورفیک است.



شکل ۲۶: نگاهی به دسته‌بندی بسترهای متفاوت پیاده‌سازی پردازنده‌های نورومورفیک [39]

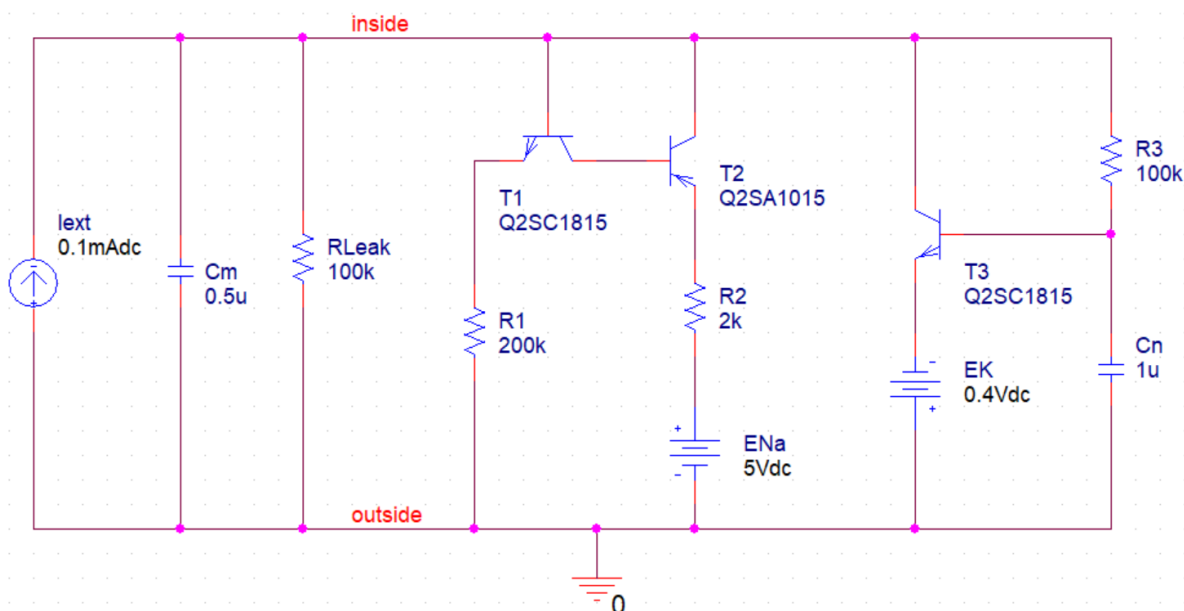
<sup>۱</sup> Framework

<sup>۲</sup> Mike Davies

همان‌طور که در شکل ۲۶ مشخص است به‌صورت کلی می‌توان بسترهای پیاده‌سازی نورومورفیک را به آنالوگ، دیجیتال و ترکیبی تقسیم کرد. در [45] بسیاری از این پیاده‌سازی‌ها را می‌توانید ببینید.

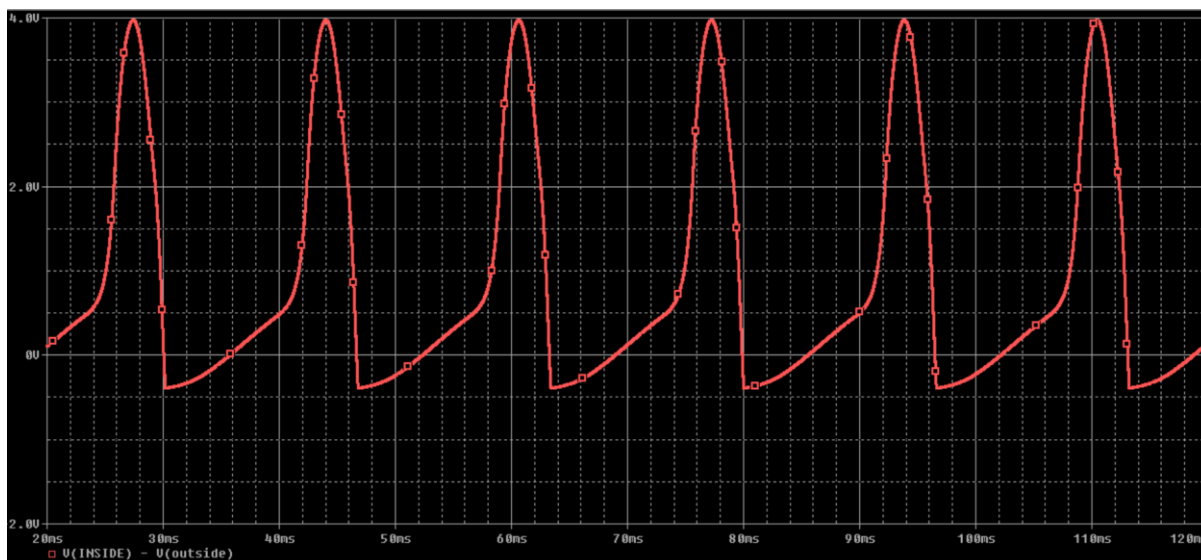
**آنالوگ:** نورومورفیک در ذات خود آنالوگ است چراکه مغز نیز آنالوگ است. اما کامپیوترهای آنالوگ مشکلات فراوانی دارند و به همین دلیل است که امروزه کاربرد بسیار کمی دارند و جای خود را به پردازنده‌های دیجیتال داده‌اند. مشکلاتی از قبیل ادوات نویزی، عدم پایداری و آسنکرونی<sup>۱</sup>. اما به عقیده برخی، معماری نورومورفیک می‌تواند مشکلات گفته شده را حل کرده و کامپیوترهای آنالوگ را دوباره احیا کند [49].

در شکل ۲۷ مداری را بر اساس [50] در PSpice پیاده‌سازی کرده‌ام. این نورون بسیار ساده است و مشکلاتی از قبیل ولتاژ بسیار بالا برای اسپایک را نیز دارد، اما نشان می‌دهد که می‌توان تنها ۳ ترانزیستور رفتار اسپایک گونه‌ای را تولید کرد که در نوع خود بسیار جالب است.



شکل ۲۷: مدار یک نورون در PSpice

در مدار شکل ۲۷، ترانزیستورهای T1 و T2 به همراه R1، R2 و ENa مدل کننده کانال‌های سدیمی هستند و ترانزیستور T3 به همراه EK، R3 و Cn مدل کننده کانال‌های پتاسیمی هستند. RLeak مدل کننده جریان نشت شده از غشا و Cm نیز مدل کننده خاصیت خازنی آن است. منبع جریان Iext نیز به‌عنوان منبع تحریک نورون عمل می‌کند.



شکل ۲۸: پاسخ مدار شکل ۲۷ (Vinside - Voutside)

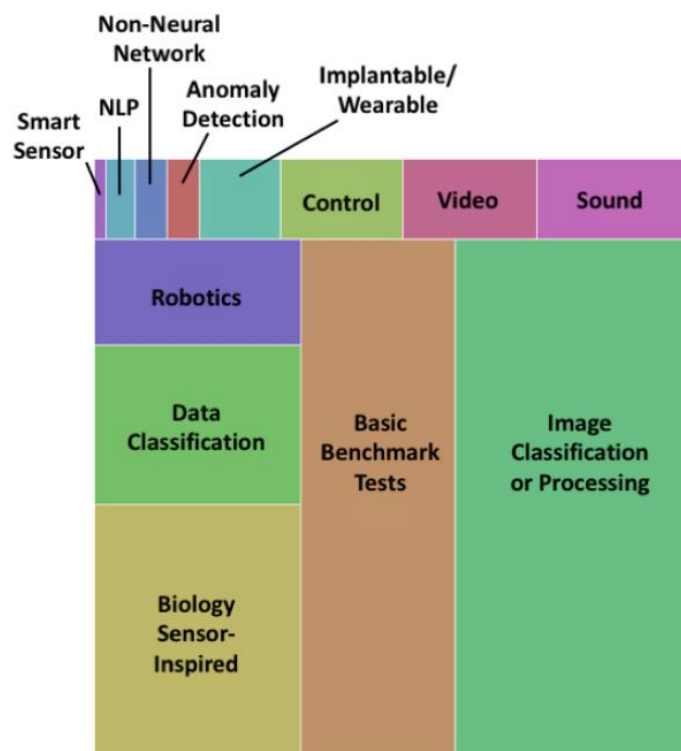
در شکل ۲۸ پاسخ این نورون را به جریان تحریک یک‌دهم میلی‌آمپری مشاهده می‌کنید. همان‌طور که مشخص شکل بسیار شبیه به اسپایک واقعی است و از نظر زمانی نیز مطلوب است. اما بزرگ‌ترین مشکل آن دامنه اسپایک است که به صورت غیرطبیعی‌ای بزرگ است. البته مدل‌های آنالوگ بسیار پیشرفته‌تری نیز ارائه شده و این حوزه محل تحقیقات فراوان است. مثالی از یک مدل نورونی آنالوگ جدید را می‌توانید در [51] ببینید.

**دیجیتال:** پیاده‌سازی دیجیتال رایج‌ترین نوع پیاده‌سازی نورومورفیک است. چراکه در اکثر موارد نیاز به طراحی در سطح سخت‌افزاری ندارد. متداول‌ترین نوع پیاده‌سازی دیجیتال، FPGA است. البته باید دقت کرد که کاربرد اصلی FPGA افزایش سرعت نسبت به شبیه‌سازی نرم‌افزاری است اما برای محصول نهایی اگر می‌خواهیم از مزایای نورومورفیک مانند توان کم و ابعاد کوچک استفاده کنیم، FPGA گزینه مناسبی نیست [52]. بسیاری از طراحی‌های موفق نورومورفیک مانند تراشه TrueNorth ساخت شرکت IBM از یک طراحی ASIC<sup>۱</sup> کاملاً سفارشی‌سازی شده استفاده می‌کند [53]. مثال دیگر تراشه SpiNNaker است که در این تراشه نیز از یک طراحی دیجیتال سفارشی‌سازی شده استفاده شده است [54]. تراشه TrueNorth فقط از یک مدل نورون و سیناپس پشتیبانی می‌کند و انعطاف زیادی ندارد، اما در عوض برای همان مدل‌ها بسیار بهینه است به طوری که هر اسپایک در آن حدود ۲۵ پیکوژول انرژی مصرف می‌کند. اما SpiNNaker از چندین مدل متفاوت نورون، سیناپس و الگوریتم یادگیری پشتیبانی می‌کند. به همین دلیل انرژی مصرفی آن به ازای هر اسپایک حدود ۱۰ نانوژول است [55].

نکته آخری که باید در این بخش به آن اشاره کرد این است که در علم افزاره‌های نیمه‌هادی و مهندسی موارد تلاش‌های زیادی انجام می‌شود تا برای برخی نورو و سیناپس به‌جای استفاده از ادوات الکترونیکی مشخص (مانند ترانزیستور و...)، ادوات جدیدی مختص آن‌ها ساخته شود. مهم‌ترین مسیر پژوهشی در این حوزه تلاش برای ساخت ممریستور<sup>۱</sup> است [56]. این وسیله رفتاری بسیار شبیه به سیناپس دارد. (به زبان ساده و غیرفنی، مقاومت الکتریکی آن وابسته به فعالیت آن است) در واقع اثبات شده است که می‌توان الگوریتم STDP را با ادوات ممریستوری ساخت [57]. بنابراین به عقیده بسیاری کلی حل مشکلات پیاده‌سازی نورومورفیک در ساخت ممریستور است.

### ۳-۹ کاربردها

در آخرین بخش این فصل درباره کاربردهای نورومورفیک صحبت می‌کنیم. واقعیت این است که هنوز آن کاربردی که نورومورفیک در آن بهترین باشد پیدا نشده است. اما پژوهشگران در کاربردهای بسیار وسیعی از نورومورفیک استفاده می‌کنند تا کاربرد اصلی آن پیدا شود. با توجه به نکات گفته شده در بخش‌های قبلی این تلاش ارزش زیادی دارد چون نباید فراموش کنیم که ما در حال استفاده از ساختار مغز گونه برای حل مسائل هستیم! در شکل ۲۹ برخی کاربردهای رایج نورومورفیک در ادبیات این بحث آورده شده است.



شکل ۲۹: برخی کاربردهای مهم نورومورفیک. ابعاد مستطیل هر مدل نشان‌دهنده میزان استفاده از آن است [38]



### ۳-۱۰ نتیجه‌گیری

در این فصل ابتدا راجع به کارکردهای کلی نورومورفیک صحبت کردیم. سپس انواع مدل‌هایی که در پیاده‌سازی نورومورفیک استفاده می‌شود را بیان کرده و آن‌ها را باهم مقایسه کردیم. بعد از آن مغز را با کامپیوتر مقایسه کرده و مفصلاً درباره چالش‌ها و نکات قوت نورومورفیک صحبت کردیم. در نهایت نیز بسترهای پیاده‌سازی نورومورفیک را بررسی کرده و به کاربردهای این تکنولوژی اشاره کردیم.

## فصل چهارم: پردازنده نورومورفیک برای تشخیص ارقام دست‌نوشته

### ۴-۱ مقدمه

در این فصل یک پردازنده نورومورفیک را بر اساس مقاله [58] پیاده‌سازی کرده و آن را بررسی می‌کنیم. لازم به ذکر است که این مقاله در حوزه استفاده از SNN در یادگیری ماشینی بسیار اثرگذار بوده است و از سال ۲۰۱۵ تاکنون بیش از ۶۰۰ بار به آن ارجاع داده شده‌است.

همان‌طور که پیش‌تر گفته شد استفاده از SNN برای مقاصد یادگیری ماشینی دو جنبه مثبت دارد. اولاً می‌تواند یک راه‌حل سریع و بهینه را ارائه کرده و عملاً تبدیل به نسل سوم شبکه‌های عصبی شود. ثانیاً به ما در فهم مغز و اینکه چگونه کارهای شناختی را انجام می‌دهد کمک می‌کند. در بسیاری از مدل‌های SNN ابتدا پارامترها در یک ANN یادگیری شده سپس این شبکه تبدیل به نسخه اسپایکی خود می‌شود [59]. به این مدل‌ها اصطلاحاً rate-based گفته می‌شود. اما این شبکه‌ها BP خوبی ندارند چراکه در طبیعت یادگیری نیز در قالب خود شبکه عصبی اسپایکی (یعنی مغز) انجام می‌شود. اما در طراحی پیش‌رو همه بخش‌ها با SNN پیاده شده، از جمله یادگیری.

ابتدا مسئله یادگیری ماشینی که به دنبال آن هستیم را معرفی می‌کنیم. سپس شبکه استفاده شده را به‌صورت کامل توضیح می‌دهیم. پس از آن به معرفی بستر نرم‌افزاری پیاده‌سازی پرداخته و نتایج را بررسی می‌کنیم. در نهایت انتقادات و پیشنهادهای خود درباره این کار را بیان خواهیم کرد.

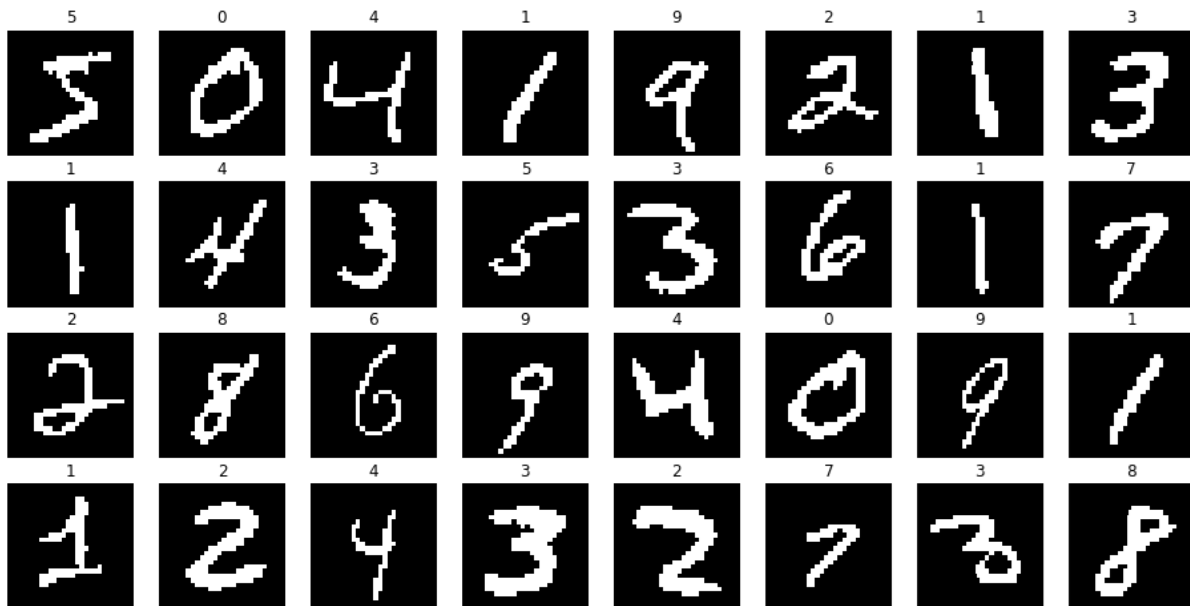
### ۴-۲ معرفی MNIST

دادگانی که عملیات یادگیری ماشینی بر روی آن انجام شده دادگان بسیار معروف MNIST<sup>۱</sup> است [60]. این دادگان در ادبیات یادگیری ماشینی جایگاه ویژه‌ای داشته و معمولاً از آن برای مقایسه الگوریتم‌های متفاوت استفاده می‌شود.

این دادگان شامل ۶۰۰۰۰ تصویر ۲۸ در ۲۸ پیکسل است که در هرکدام از آن‌ها یک رقم دست‌نوشته از بین ارقام ۰ تا ۹ وجود دارد. این ۶۰۰۰۰ تصویر داده‌های آموزشی هستند. علاوه بر آن‌ها این دادگان شامل ۱۰۰۰۰ تصویر به‌عنوان داده تست نیز است.

در شکل ۳۰ برخی از این داده‌ها را می‌بینید.

<sup>۱</sup> برای دانلود MNIST به لینک <http://yann.lecun.com/exdb/mnist> مراجعه کنید.



شکل ۳۰: برخی از داده‌های MNIST

همان‌طور که پیش‌تر گفته شد لازم است تا داده‌ها را که در قالب  $28 \times 28$  عدد برای هر رقم هستند را به زبان SNN یا همان قطار اسپایک درآوریم. هر کدام از این ۷۸۴ عدد تبدیل به یک قطار اسپایک می‌شوند که فرکانس اسپایک زدن آن متناسب با شدت پیکسلش<sup>۱</sup> است. بیشترین شدت پیکسل ۲۵۵ است، پس همه آن‌ها را تقسیم بر ۴ می‌کنیم. بدین ترتیب فرکانس‌های اسپایک زدن ورودی‌ها بین ۰ تا  $63.75$  هرتز خواهد بود. لازم به ذکر است که هر داده ورودی برای  $350$  میلی‌ثانیه به شبکه ارائه می‌شود.

### ۴-۳ توضیح شبکه

#### ۴-۳-۱ مدل نورون

برای مدل نورون‌های این شبکه از مدل LIF استفاده شده است. این مدل را در فصل‌های قبل به صورت کامل توضیح داده‌ایم. در این شبکه از هر دو نوع نورون مهاری و تحریکی استفاده شده است. همچنین طبق مشاهدات علوم اعصاب، ثابت زمانی برای نورون‌های تحریکی بیشتر از مهاری است. در زیر رابطه نورون‌ها آورده شده است.

$$\tau \frac{dV_m}{dt} = (E_{rest} - V_m) + g_{exc}(E_{exc} - V_m) + g_{inh}(E_{inh} - V_m)$$

$$\text{if } V_m = V_{th}, \text{ then fire and } V_m(t + \Delta) = E_{rest} \quad (\text{معادله ۱۰})$$

---

<sup>۱</sup> Pixel intensity

دقت کنید که رابطه بالا کمی با LIF گفته شده در فصل‌های پیشین متفاوت است. در این رابطه علاوه بر بخش مربوط به نشت و پتانسیل استراحت، رسانایی و پتانسیل تعادل سیناپس‌های مهارتی و تحریکی نیز آورده شده است تا نقش آن‌ها در تحریک و مهار نورون و در واقع اسپایک زدن آن مشخص شود.

#### ۲-۳-۴ مدل سیناپس

برای مدل سیناپس از مدلی کاملاً مشابه با مدل گفته شده در بخش ۲-۴-۲ استفاده شده. در ادامه رابطه این مدل آورده شده است.

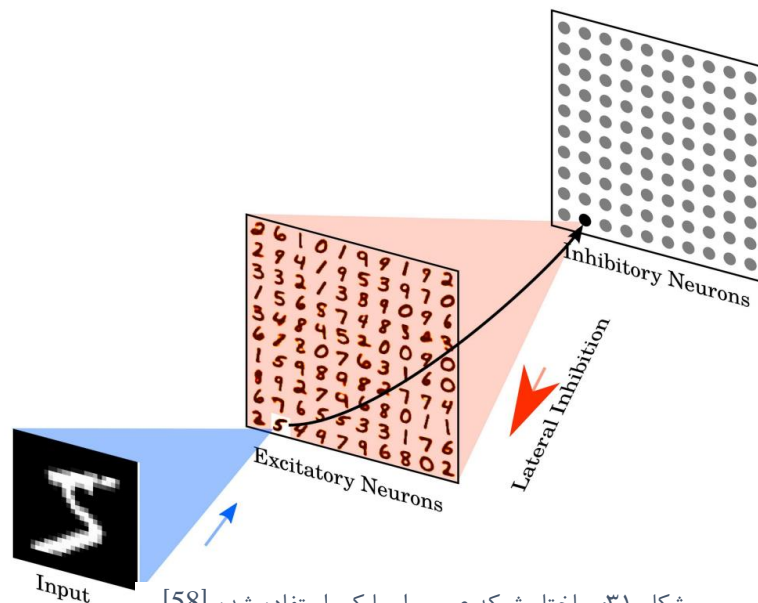
$$\begin{cases} \tau_{g_{exc}} \frac{dg_{exc}}{dt} = -g_{exc} \\ \tau_{g_{inh}} \frac{dg_{inh}}{dt} = -g_{inh} \end{cases} \quad (\text{معادله ۱۱})$$

دقت کنید که در این شبکه هم سیناپس‌های مهارتی داریم و هم سیناپس‌های تحریکی.

برای تمامی پارامترهای مدل نورون، سیناپس و الگوریتم یادگیری از مقادیری در رنج مقدار واقعی مشاهده شده در طبیعت استفاده شده است [61]. در این بین اما ثابت زمانی غشا نورون‌های تحریکی استثناست. افزایش این ثابت زمانی از حدود ۱۰ تا ۲۰ میلی‌ثانیه - که مقدار واقعی آن است - به ۱۰۰ میلی‌ثانیه دقت دسته‌بندی را بسیار افزایش داد [58]. ریشه این امر به الگوریتم کد کردن ورودی به قطار اسپایک برمی‌گردد. هرچه نورون‌ها برای ثابت زمانی بیشتری داشته باشند، می‌توانند تخمین بهتری از چیستی ورودی اعمال شده به‌شان بزنند. درباره علت این موضوع در بخش انتقادات و پیشنهادات بحث خواهیم کرد.

#### ۳-۳-۴ ساختار شبکه

ساختار این شبکه را می‌توانید در شکل ۳۱ مشاهده کنید.



شکل ۳۱: ساختار شبکه عصبی اسپایکی استفاده شده [58]

همان طور که در شکل ۳۱ مشخص است این شبکه متشکل از سه لایه است.

**لایه اول:** لایه اول، نورون‌های ورودی هستند که طبق توضیحات گفته شده، فرکانس اسپایک زدن آن‌ها متناسب با شدت پیکسل متناظر با آن‌ها است. پس بدیهی است که تعداد نورون‌های لایه ورودی ثابت و برابر  $28 \times 28 = 784$  است. همه نورون‌های این لایه به همه نورون‌های لایه دوم متصل هستند (اتصال آبی‌رنگ). وزن این اتصال همان وزنی است که شبکه روی آن آموزش می‌بیند و در واقع پارامترهای آزاد شبکه هستند.

**لایه دوم:** در لایه دوم نورون‌های تحریکی قرار دارند. هر نورون در این لایه ورودی را از همه نورون‌های لایه اول دریافت کرده و در صورتی که این ورودی‌ها بتوانند نورون را به آستانه‌اش برسانند، اسپایک می‌زند. خروجی هر کدام از نورون‌های این لایه به یک نورون متناظرش در لایه سوم متصل می‌شود (اتصال با فلش مشکی). پس هر اسپایک در نورون‌های این لایه، باعث به‌وجود آمدن اسپایک در نورون متناظرش در لایه سوم می‌شود. وزن این اتصالات (از لایه دوم به سوم) ثابت هستند (البته در اینجا منظور این است که  $w$  آن‌ها ثابت است، و گر نه  $g_{syn}$  آن‌ها طبق صحبت‌ها گفته شده در بخش قبل تغییر می‌کند). تعداد نورون‌های این لایه یکی از متغیرهای شبکه است که در ادامه اثر تعداد آن‌ها را خواهیم دید. در کاری که من انجام داده‌ام تعداد نورون‌های این لایه ۴۰۰ است.

**لایه سوم:** تعداد نورون‌های لایه سوم همیشه برابر تعداد نورون‌های لایه دوم است. (چون اتصال از لایه دوم به سوم، یک - به - یک است) اما نورون‌های این لایه مهاری هستند. نورون‌های لایه سوم ورودی خود را از نورون‌های لایه دوم می‌گیرند و در ساختاری فیدبکی، به همه نورون‌های لایه دوم خروجی می‌دهند، به جز آن نورونی که از آن ورودی گرفتند. (این اتصال با رنگ قرمز نمایش داده شده) این وزن‌ها نیز همانند وزن‌های لایه دوم به سوم ثابت هستند.

لازم به ذکر است، در واقع آن لایه‌ای که نتیجه کلاس‌بندی را مشخص می‌کند نورون‌های لایه دوم هستند که در ادامه شیوه این دسته‌بندی را خواهیم دید. پس می‌توان گفت که ساختار شبکه ارائه شده نوعی شبکه winner-take-all است.

#### ۴-۳-۴ الگوریتم یادگیری

در این شبکه تمام سیناپس‌ها در اتصال لایه اول به دوم توسط الگوریتم STDP آموزش می‌بینند. در مقاله اصلی از انواع متفاوتی از STDP استفاده شده. (power-law dependence STDP, exponential weight STDP, pre-and-post STDP, triplet STDP) اما در این کار از همان مدلی استفاده شده که در بخش ۲-۶-۴ به صورت کامل توضیح داده شده است.

#### ۴-۳-۵ هم ایستایی

یکی دیگر از الگوریتم‌های مهم بکار گرفته شده در این شبکه هم/ایستایی<sup>۱</sup> است. آستانه هر نورون مقدار ثابتی ندارد و به جای مقدار  $V_{th}$  مقدار آن برابر  $V_{th} + \theta$  است. تنها هر بار که نورون اسپایک می‌زند افزایش یافته و سپس به صورت نمایی به مقدار صفر برمی‌گردد. پس مشخص است که یک نورون نمی‌تواند به راحتی بارها پشت هم اسپایک بزند، چون پس از هر اسپایک آستانه آن افزایش یافته و فعال شدن آن نیاز به ورودی قوی‌تری دارد. با استفاده از این روش نرخ اسپایک زدن نورون‌ها نمی‌تواند به هر میزانی بالا رود و سیستم پایدار می‌ماند. برای پیاده‌سازی این الگوریتم از [62] استفاده شده.

#### ۴-۴ شیوه کارکرد شبکه

تمامی ۶۰۰۰۰ داده آموزشی MNIST به شبکه نشان داده می‌شوند. هر داده برای ۳۵۰ میلی‌ثانیه و سپس بعد از آن ۱۵۰ میلی‌ثانیه داده‌ای نشان داده نمی‌شود تا تمامی مقادیر (مانند  $g_{syn}$  و  $\theta$ ) به مقدار اولیه خود بازگردند. پس از آنکه تمامی داده‌ها به شبکه نشان داده شدند و فرایند یادگیری انجام شد، به تاریخچه فعالیت نورون‌های لایه دوم نگاه می‌کنیم. هر نورون در این لایه نماینده یک رقم (کلاس‌های MNIST) است. برای تشخیص اینکه هر نورون نماینده کدام رقم است ابتدا روی شدت فعالیت (نرخ اسپایک زدن) همه رقم‌ها زمانی که به آن نورون نشان داده شده‌اند میانگین می‌گیریم. سپس هر رقمی که بیشترین میانگین را دارد رقم خاص آن نورون است.

حال که وزن‌ها آموزش دیده و مشخص شده‌اند، اگر تصویر جدیدی (از مجموعه داده‌های تست) به ورودی شبکه داده شود، پس از طی شدن زمان ۳۵۰ میلی‌ثانیه تحریک و ۱۵۰ میلی‌ثانیه استراحت، از میزان فعالیت نورون‌های هر کلاس میانگین گرفته و این اعداد را با یکدیگر مقایسه می‌کنیم. بزرگ‌ترین این اعداد نشان‌دهنده کلاس موردنظر است.

مشخص است طبق فرایندی که توضیح داده شد، الگوریتم یادگیری این شبکه Unsupervised است.

#### ۴-۵ بستر پیاده‌سازی

ساختار گفته شده در خود مقاله اصلی در محیط Brian پیاده‌سازی شده است. Brian یک شبیه‌ساز نورونی تحت پایتون است که از آن برای شبیه‌سازی مدل‌های نورونی مخصوصاً در علوم اعصاب بسیار استفاده می‌شود [63]. اما در این کار از Brian2 [64] استفاده شده که نسخه بروزتر همان Brian است. لازم به

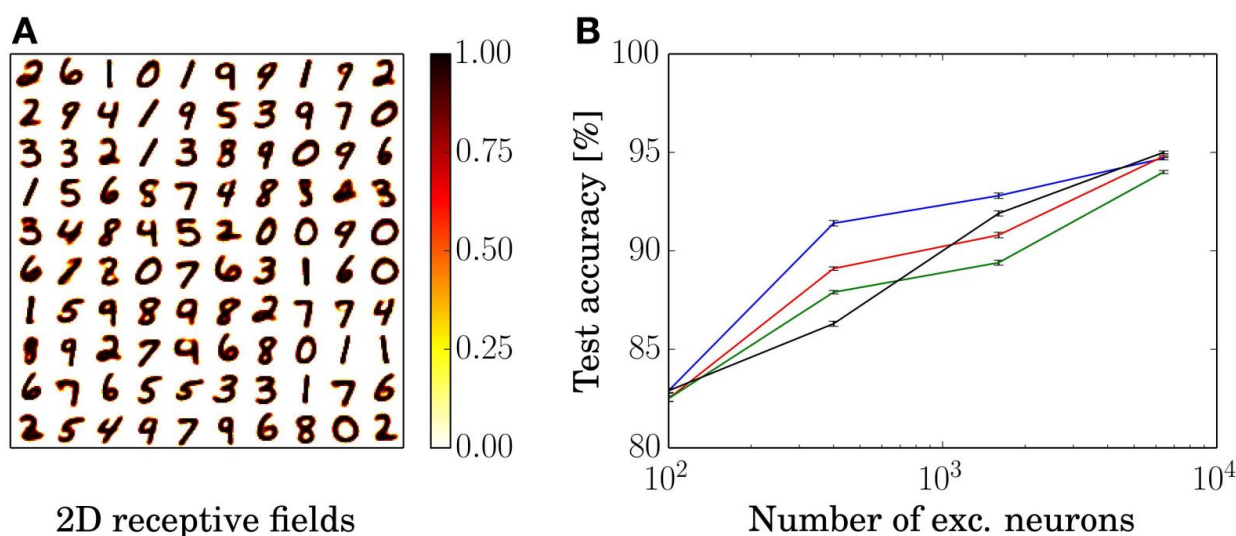
ذکر است که در نوشتن کد در بستر Brian2 از کد نویسندگان<sup>۱</sup> که در نسخه قدیمی نوشته بودند کمک گرفته شده است.

## ۴-۶ نتایج

در این بخش ابتدا نتایج به دست آمده توسط مقاله را بررسی کرده و سپس نتایج به دست آمده در این پایان نامه را گزارش خواهیم کرد.

### ۴-۶-۱ نتایج مقاله

در شکل ۳۲ نتایج مقاله آورده شده است.



شکل ۳۲: نتایج مقاله [58]

نمودار B شکل ۳۲ نشان دهنده میزان دقت<sup>۲</sup> به ازای تعداد نورون‌های لایه دوم و الگوریتم‌های یادگیری متفاوت است. در اینجا به تفاوت دقت ناشی از الگوریتم‌های متفاوت STDP نمی‌پردازیم، اما باید ذکر کرد که بهترین دقت (خط آبی‌رنگ) برای الگوریتم triplet STDP [65] است. اندازه‌گیری‌ها به ازای ۱۰۰، ۴۰۰، ۱۶۰۰ و ۶۴۰۰ نورون در لایه دوم (و به طبع آن سوم) انجام شده است. مشخص است که هرچه تعداد نورون‌ها بیشتر شود، دقت نیز افزایش می‌یابد و می‌تواند به ۹۵ درصد هم برسد. نکته دیگری که می‌توان گفت این است که با افزایش تعداد نورون‌ها اثر تکنیک‌های متفاوت STDP کاهش می‌یابد.

<sup>۱</sup> کد نویسندگان از آدرس <https://github.com/peter-u-diehl/stdp-mnist> قابل دسترسی است.

<sup>۲</sup> Accuracy

نمودار A شکل ۳۲ اما مفهوم زیباتری دارد. این نمودار برای شبکه‌ای با ۱۰۰ نورون تحریکی ترسیم شده است. این تصویر متشکل از ۱۰۰ مربع  $28 \times 28$  پیکسل به‌ازای هر نورون لایه دوم است که رنگ هر پیکسل آن متناسب است با وزن اتصال آن پیکسل به نورون موردنظر. رقمی که مشخص شده نیز نشان می‌دهد آن نورون نماینده کدام کلاس است. پس در واقع می‌توان این‌گونه تفسیر کرد که تصاویری از پوروتوتایپ‌های<sup>۱</sup> هر رقم در وزن‌های بین لایه اول و دوم ذخیره شده است. در اینجا مفهوم بسیار مهمی که پیش‌تر بارها به آن اشاره کردم مشخص می‌شود: در معماری نورومورفیک حافظه با پردازش مجتمع است. پس در واقع شبکه چندین تصویر پوروتوتایپ از هر رقم را در وزن‌های سیناپسی خود ذخیره کرده و با مقایسه ورودی جدید با آن‌ها درباره کلاس آن تصمیم می‌گیرد (شبیه به فیلتر گابور<sup>۲</sup>)

#### ۴-۶-۲ نتایج کار انجام شده

من محاسبات را با استفاده از کامپیوتر شخصی خودم (که CPU آن Intel Core i7-10510U) است به زبان پایتون و در *VsCode*<sup>۳</sup> انجام داده‌ام. همچنین بار دیگر همه محاسبات را با گوگل کولب<sup>۴</sup> نیز انجام دادم تا بین سرعت آن‌ها مقایسه کنم. این مقایسه در ادامه گزارش شده<sup>۵</sup>.

به دلیل حجم محاسباتی بسیار زیاد، من موفق به آموزش شبکه نشدم. در واقع آموزش شبکه‌ای با ۴۰۰ نورون و استفاده از تنها یک‌سوم داده‌های آموزشی MNIST چیزی بیش از ۲۴ ساعت در کولب به طول می‌انجامد؛ بنابراین من شبکه‌ای پیاده کردم تا با استفاده از وزن‌های موجود، دقت را بر روی ۱۰۰۰ داده‌های تست MNIST بیاید. البته لازم به ذکر است که کد بخش یادگیری به‌صورت کامل پیاده شده است.

نتایج ذیل برای شبکه‌ای با ۴۰۰ نورون در لایه دوم (و طبیعتاً سوم) است که بر روی ۱۰۰۰۰ داده تست MNIST به‌دست آمده است.

جدول ۲: مقایسه نتایج گوگل کولب و کامپیوتر شخصی

مدت زمان انجام محاسبات	دقت	پردازشگر
۳ ساعت و ۳۵ دقیقه	۹۱.۴۳٪	Google Colab
۴ ساعت و ۵۳ دقیقه	۹۱.۴۳٪	Intel Core i7-10510U with VSCode

<sup>۱</sup> Prototype

<sup>۲</sup> Gabor filter

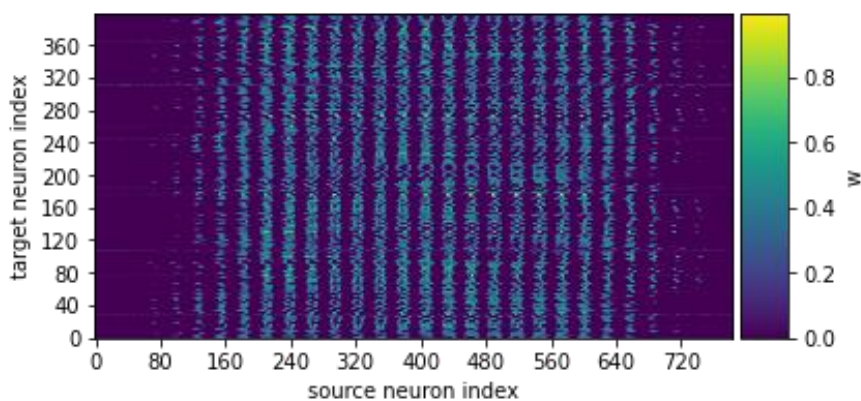
<sup>۳</sup> Visual Studio Code یک IDE محبوب برای زبان پایتون است.

<sup>۴</sup> Google Colab

<sup>۵</sup> توضیحاتی درباره گوگل کولب و ورژن کتابخانه‌های استفاده شده در پیوست‌ها آمده است.

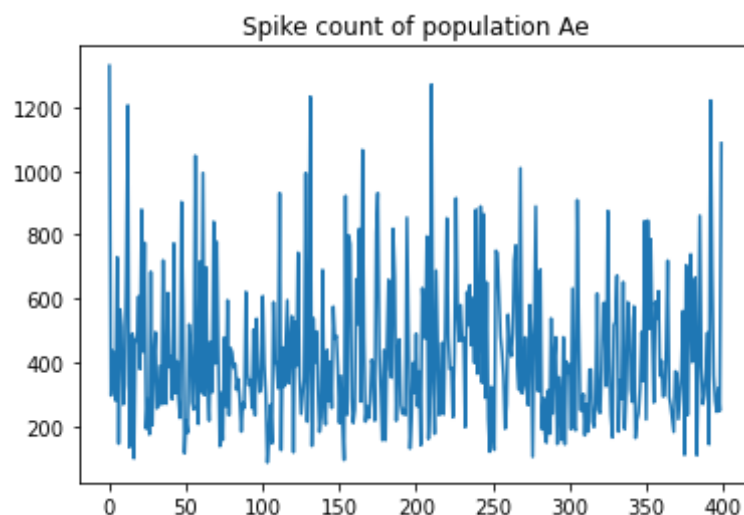


همان‌طور که مشخص است به دلیل استفاده از عدد رندوم یکسان، دقت‌ها برابر شده است. اما سرعت محاسبات در کولب از کامپیوتر شخصی بیشتر است. البته لازم به ذکر است که نه در کولب و نه در کامپیوتر شخصی از GPU استفاده نشده چون Brian2 با آن تطابق ندارد.



شکل ۳۳: نمودار وزن‌های لایه اول به دوم

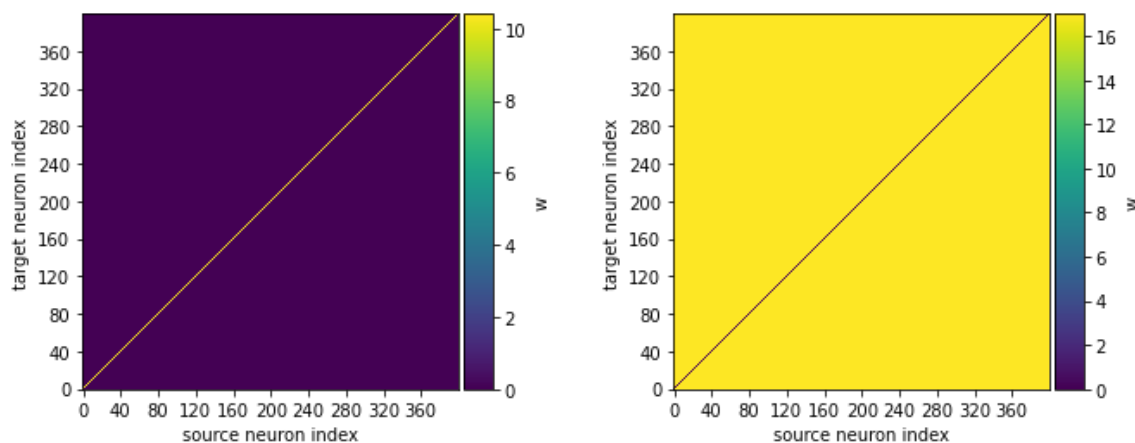
در شکل ۳۳ نمودار وزن‌های آموزش‌دیده از لایه اول به دوم را مشاهده می‌کنید. علت تناوبی بودن نسبی این نمودار این است که اطلاعات مهم هر داده در پیکسل‌های وسطی آن قرار دارد و همان‌طور که از شکل ۳۰ مشخص است پیکسل‌ها کناری معمولاً محتوای مفیدی ندارند. به همین دلیل است که هر بار به نورونی‌هایی می‌رسیم که نماینده گوشه‌های پیکسل هستند، وزن آن‌ها صفر است و این حالت تناوبی پدید می‌آید.



شکل ۳۴: تعداد اسپایک‌های نورون‌های لایه دوم

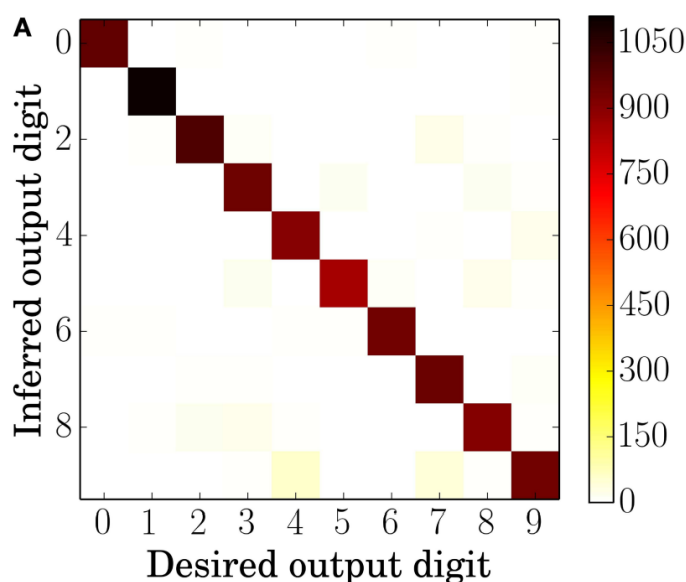
شکل ۳۴ نشان دهنده تعداد اسپایک‌های نورون‌های لایه دوم است. همان‌طور که مشخص است تعداد اسپایک‌ها نزدیک به هم و حدود ۶۰۰ است. دلیل اینکه برخی نورون‌ها بسیار بیشتر اسپایک زدند را می‌توان اینگونه تعبیر کرد که آن‌ها نورون‌هایی هستند مربوط به ارقامی که شکل هندسی آن‌ها، برخی ارقام

دیگر را نیز دربرمیگیرد. مثل رقم 0 که شکل آن می‌تواند ارقام 6، 9 و تاحدی 8 را دربربگیرد. پس در واقع تصاویر بیشتری وجود دارند که می‌توانند آن نورون را تحریک کنند.



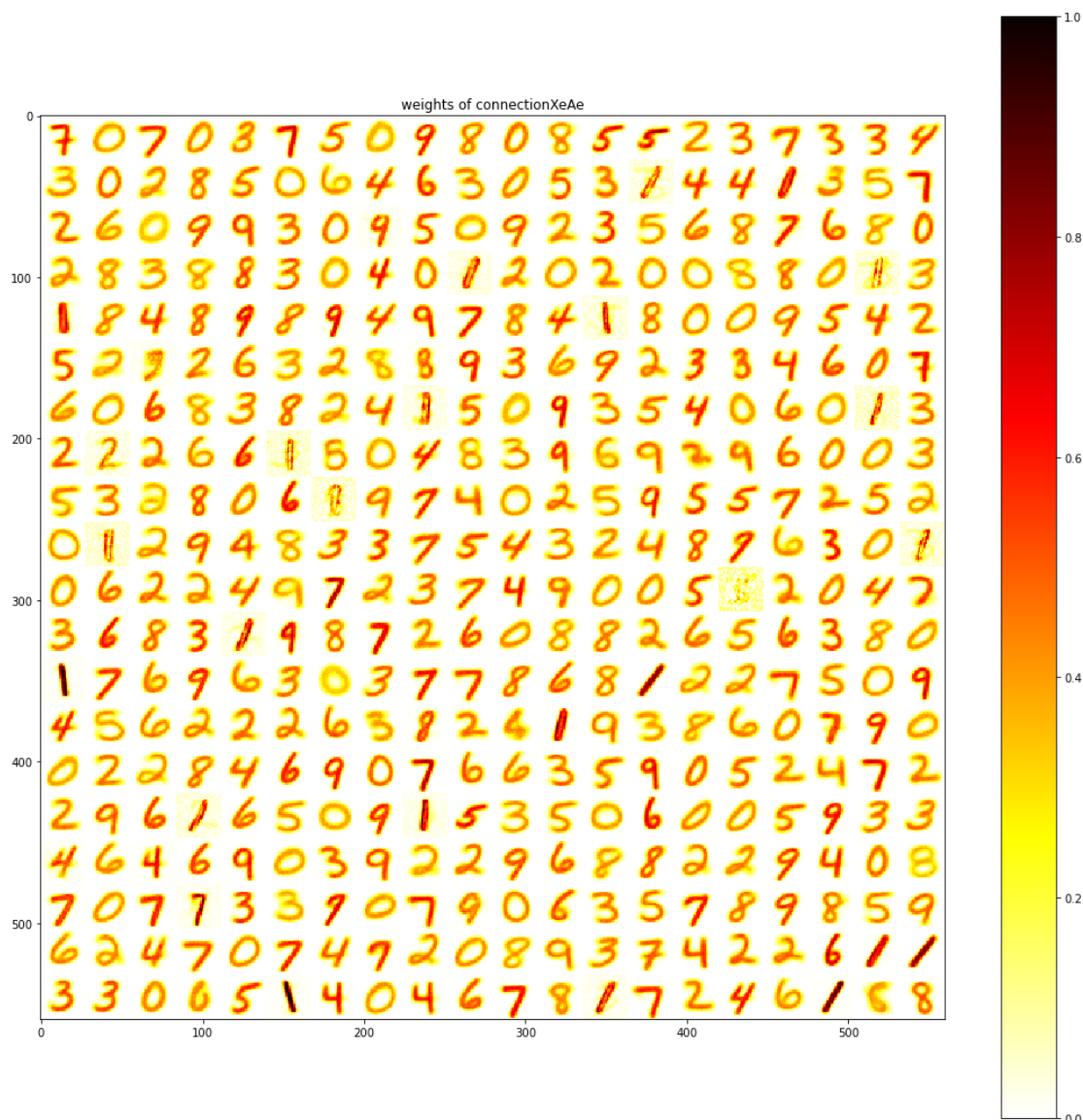
شکل ۳۵: وزن‌های لایه دوم به سوم در سمت چپ و لایه سوم به دوم در سمت راست

شکل ۳۵ سمت چپ نشان‌دهنده وزن‌های لایه دوم به سوم است. همان گونه که در بخش ساختار شبکه توضیح داده شد، هر نورون فقط به نورون متناظرش در لایه سوم متصل می‌شود. پس همان‌طور که مشخص است همه وزن‌ها به جز وزن‌های روی قطر اصلی، صفر هستند. در سمت راست نیز وزن‌های اتصالات لایه سوم به دوم را می‌بینید که این بار طبق توضیحات داده شده فقط وزن‌های اتصال هر نورون به نورون متناظرش در لایه دوم صفر است و باقی وزن‌ها برابر و غیر صفر هستند.



شکل ۳۶: ماتریس کانفیوژن

در شکل ۳۶ ماتریس کانفیوژن کلاس‌بندی را می‌بینید. مثلاً اگر دقت کنید بیشترین خطا در کلاس‌بندی بین رقم 7 و 9 پیش‌آمده که به دلیل ظاهرشان - مخصوصاً دست‌نویس - کاملاً منطقی است. در کل ۸۵۷ نمونه اشتباه کلاس‌بندی شده‌اند.



شکل ۳۷: پوروتوتایپ‌های ذخیره شده در وزن‌های لایه اول به دوم

شکل ۳۷ در واقع همان شکل ۳۲ بخش A، برای شبکه پیاده شده است (پوروتوتایپ‌های ذخیره شده در وزن‌های لایه اول به دوم). طبیعتاً این بار به جای ۱۰۰ نورون، ۴۰۰ نورون داریم. تمام توضیحات شکل ۳۲ برای این شکل نیز صادق است.

نتیجه دیگری که می‌تواند در فهم ما از شبکه بیشتر کمک کند تعداد نورون‌های هر کلاس است. همان‌طور که پیش‌تر مفصلاً توضیح داده شد هر نورونی در لایه دوم نماینده یکی از ده کلاس ما است پس می‌توانیم با دیدن تعداد نورون‌های هر کلاس، شبکه را از این منظر نیز بررسی کنیم. در جدول ۳ تعداد نورون‌های هر کلاس را مشاهده می‌کنید. علاوه بر آن تعداد داده‌های هر کلاس در مجموعه آموزشی و تست را نیز در کنار آن‌ها می‌بینید.

جدول ۳: مقایسه تعداد کلاس‌ها در لایه دوم و داده‌های آموزشی و تست

کلاس	تعداد در لایه دوم شبکه	تعداد در داده‌های آموزشی	تعداد در داده‌های تست
0	۵۷	۵۹۲۳	۹۸۰
1	۲۲	۶۷۴۲	۱۱۳۵
2	۴۹	۵۹۵۸	۱۰۳۲
3	۴۷	۶۱۳۱	۱۰۱۰
4	۳۲	۵۸۴۲	۹۸۲
5	۳۴	۵۴۲۱	۸۹۲
6	۴۴	۵۹۱۸	۹۵۸
7	۳۴	۶۲۶۵	۱۰۲۸
8	۴۲	۵۸۵۱	۹۷۴
9	۳۹	۵۹۴۹	۱۰۰۹
مجموع	۴۰۰	۶۰۰۰۰	۱۰۰۰۰

همان‌طور که از مقادیر جدول مشخص است تعداد نورون‌های هر کلاس در لایه دوم تقریباً برابر است اما در این بین دو کلاس هستند که وضعیت متفاوتی دارند. اولاً کلاس 1 که تعداد نورون‌های آن از باقی کلاس‌ها بسیار کمتر است، ثانیاً کلاس 0 که تعداد نورون‌های آن بیشتر است. با توجه کردن به دو ستون دیگر معلوم می‌شود که این تفاوت ناشی از تفاوت تعداد داده‌های آموزشی برای کلاس‌ها نیست. یکی از راه‌های تفسیر این تفاوت دقت کردن به شکل نوشته شدن این ارقام است. شکل رقم 0 در برگزیده ارقام دیگری مانند 6، 8 و 9 است که نشان می‌دهد این کلاس با ورودی‌های بیشتری تحریک می‌شود. اما شکل 1 شباهت زیادی به باقی ارقام ندارد. با توجه به این نکته می‌توان حدس زد که در صورت استفاده از داده‌گان MNIST فارسی، بیشترین نورون‌ها به رقم ۱ تعلق بگیرند چراکه شکل آن در ارقام دیگر زیاد تکرار می‌شود.

مسئله مهم دیگر بحث همگرایی این شبکه است. متأسفانه این موضوع در [58] نیز بررسی نشده، اما می‌توان گفت که مشکل اصلی در بررسی همگرایی این شبکه نبود مرجعی برای فهمیدن میزان همگرا شدن است. خصوصاً که شیوه یادگیری نیز *unsupervised* است.

## ۴-۷ نتیجه گیری

پیش از شروع این بخش لازم است اشاره کنم که در [58] مقایسه‌ای بین این شبکه و برخی SNN‌های دیگر انجام شده است که می‌تواند مفید باشد.

مهم‌ترین انتقاد به این شبکه عدم تطابق برخی از ویژگی‌ها آن با مغز بیولوژیک است. مهم‌ترین عدم تطابق مربوط به شیوه کد کردن ورودی به زبان اسپایک است. در واقع در مغز بیولوژیک ساختار ورودی سیستم بینایی بیشتر شبیه به شبکه عصبی پیچشی (شکل ۲۰) است که قبل از اعمال ورودی به شبکه، از آن استخراج ویژگی می‌شود. به نظر می‌رسد که عدم تطابق ثابت زمانی غشا نورون‌های تحریکی با نسخه بیولوژیک نیز ناشی از همین نکته است.

نکته مهم دیگر این است که باید الگوریتم STDP بر روی همه وزن‌ها اعمال شود. چرا که در مغز STDP روی بیشتر سیناپس‌ها وجود دارد و باعث تغییر آن‌ها می‌شود.

نقد دیگری که در خود مقاله نیز مطرح شده این است که در مغز بیولوژیک تعداد نورون‌های لایه مهاری (لایه سوم) حدوداً یک‌چهارم لایه تحریکی (لایه دوم) است و همه نورون‌های لایه دوم به همه نورون‌های لایه سوم متصل هستند.

پیشنهادی که به نظر می‌رسد این است که اولاً با سعی در رفع مشکلات بالا و همچنین استفاده از یک مدل نورونی دقیق‌تر (مانند ایژیکویچ) ساختار شبکه خود را بیشتر با بیولوژی نزدیک کنیم. همچنین خوب است تا با بیشتر کردن تعداد لایه‌های اولیه، نوعی انتخاب ویژگی را در شبکه به وجود آوریم.

در نهایت مهم این است که توانستیم با ساختاری شبیه به مغز، ارقام دست‌نوشته MNIST را با دقت بالایی تشخیص دهیم. البته که این دقت در مقایسه با باقی روش‌های یادگیری ماشینی مانند SVM مطلوب نیست اما باید توجه داشت که در طراحی این شبکه اولویت با تقلید مغز بوده و الگوریتم یادگیری آن نیز شبیه بیولوژی بوده است.

## فصل پنجم: نتیجه‌گیری و پیشنهادها

### ۵-۱ بررسی کار انجام شده

در این پایان‌نامه ما دیدیم که با الهام گرفتن از ساختار مغز، یک مسئله واقعی را حل کرد. دقت کنید که در مدل ارائه شده هیچ ریاضیاتی به‌صورت هدفمند برای حل مسئله به کار گرفته نشد. اگر لحظه‌ای درنگ کنیم درمی‌یابیم که این نکته واقعاً عجیب است. می‌توان این‌گونه بیان کرد که انگار در ذات معماری نورومورفیک نوعی هوشمندی قرار دارد که باعث می‌شود حتی با تعداد کمی نورون نیز یک مسئله شناختی مانند تشخیص ارقام دست‌نوشته را تا حد نسبتاً خوبی انجام داد. البته که بدون شک ما هنوز در ابتدای مسیر هستیم و برای رسیدن به پردازنده نورومورفیکی که بتواند از پس انجام بسیاری از عملیات‌ها بربیاید راه زیادی باقی‌مانده.

نکته دیگر اهمیت الگوریتم یادگیری ما یعنی STDP است. باید دقت کرد که این الگوریتم ساده، Unsupervised بوده و بدون کمک گرفتن از هیچ سیگنال اصلاح یا خطایی توانسته به‌نوعی از اشکال ورودی تقلید کرده و شمایی از آن‌ها را مانند یک شابلون در وزن‌های خود ذخیره کند. این نشان می‌دهد که یک قانون ساده می‌تواند پیچیدگی‌های بسیار جالبی ایجاد کند.

بار دیگر اشاره به این نکته مهم است که نباید نقش کمک‌کننده نورومورفیک به فهم بهتر علوم اعصاب را نادیده بگیریم. ما درواقع برای طراحی این نوع شبکه‌ها و حل مسائل شناختی، می‌توانیم درباره چگونگی حل این مسائل در مغز بیولوژیک هم نظریه‌پردازی کنیم.

### ۵-۲ درنگ فلسفی

در اینجا لازم است تا به یک موضوع فلسفی مهم نیز اشاره شود. سؤالی که وجود دارد این است: آیا موفقیت نورومورفیک نشان‌دهنده این موضوع است که تمام قوای شناختی ما فقط برخاسته از مغز ماست؟ باید در نظر داشت، طبق چیزی که فصول قبل دیدیم، میزان پیچیدگی مغز با میزان پیچیدگی‌ای که ما می‌توانیم به‌وسیله معماری نورومورفیک پیاده کنیم به‌هیچ‌وجه قابل قیاس نیست. در واقع به عقیده بسیاری همین پیچیدگی می‌تواند باعث پدیدآمدن ویژگی‌هایی باشد که به‌صورت تقلیل‌گرایانه قابل توضیح نیستند. اما اگر با هر روشی روزی بتوان پردازنده نورومورفیکی ساخت که اکثر رفتارهای انسانی را تولید کند، آن موقع می‌توان گفت که این اختراع قطعاً نقطه مثبتی به نفع نظریه این‌همانی ذهن و مغز [3] است.

جالب است بدانید که برخی فیلسوفان بزرگ مانند جان سرل<sup>۱</sup> استدلال می‌کنند که ساخت ماشینی که هوش مصنوعی قوی<sup>۲</sup> داشته باشد (به این معنا که تمام ویژگی‌های انسانی را شامل باشد) غیرممکن است. یکی از این استدلال‌ها به نام *اتاق چینی*<sup>۳</sup> معروف است [66].

اما در سمت دیگر فیلسوفانی نیز هستند که ساختن هوش مصنوعی قوی خصوصاً با استفاده از ساختار نورومورفیک را راهی برای اثبات نظریه این‌همانی ذهن و مغز می‌دانند.

در نهایت باید منتظر آینده بمانیم و ببینیم کدام یک از این نظرات (شاید هم هیچ‌کدام!) موفق خواهند بود. این امر نشان‌دهنده تأثیرگذاری علم و فناوری بر روی نظرات فلسفی است که در نوع خود بسیار جالب است.

### ۵-۳ راه‌های توسعه در آینده

این پروژه را در آینده می‌توان از مسیرهای گوناگونی توسعه داد. در ادامه برخی از آن‌ها را ذکر می‌کنم.

(۱) MNIST فارسی: یکی از اولین کارها در راستای توسعه این پروژه، استفاده از MNIST فارسی روی این شبکه است.

(۲) ایجاد یک چارچوب برنامه‌نویسی: کتابخانه Brian2 که از آن برای شبیه‌سازی این شبکه استفاده شد در عین داشتن ویژگی‌های مثبت، برای کارهای یادگیری ماشینی بهینه نیست. در اینجا می‌توان پروژه‌هایی برای ایجاد کتابخانه‌هایی برای SNNها تعریف کرد. یکی از این نمونه‌های موفق که در دانشگاه تهران انجام شده، [67] است.

(۳) پیاده‌سازی بر روی FPGA: یکی دیگر از راه‌های مهم توسعه، پیاده‌سازی شبکه فوق - در ابتدا بدون یادگیری و در گام بعد با یادگیری - بر روی FPGA است. این کار می‌تواند سرعت شبیه‌سازی را نیز افزایش داده و راه‌های توسعه دیگر را هموار سازد.

(۴) پیاده‌سازی بر روی IC: اما راه توسعه اصلی پیاده‌سازی شبکه بر روی IC است. البته این شبکه ارزش پیاده‌سازی بر روی IC - که کاری سخت و هزینه‌بر است - را ندارد اما می‌تواند بستری باشد برای ساخت شبکه‌های دیگر که ارزش پیاده‌سازی بر روی IC را دارند.

(۵) تعریف کاربرد: شاید مهم‌ترین و اثرگذارترین کار این باشد که به دنبال کاربردهایی در مشکلات واقعی برای پردازنده‌های نورومورفیک باشیم. مثلاً یکی از این کاربردها می‌تواند Spike Sorting باشد. چون این عملیات معمولاً بر روی تراشه کاشته شده در مغز انجام می‌شود، نیاز است تا با کمترین توان

---

John Searle<sup>۱</sup>

Strong A.I.<sup>۲</sup>

Chinese room<sup>۳</sup>

مصرفی پیاده شود. از طرف دیگر Spike Sorting مانند کاری که انجام دادیم، یک عملیات Unsupervised است. پس به نظر می‌رسد که نورومورفیک (به شرطی که در انتها در سطح IC پیاده شود) می‌تواند گزینه خوبی برای حل این مشکل باشد.

در پایان امیدوارم که این پایان‌نامه بتواند نقطه آغاز مناسبی برای علاقه‌مندان به محاسبات نورومورفیک، علوم اعصاب محاسباتی و یادگیری ماشینی باشد و در آینده عامل موفقیت‌های فراوانی شود.

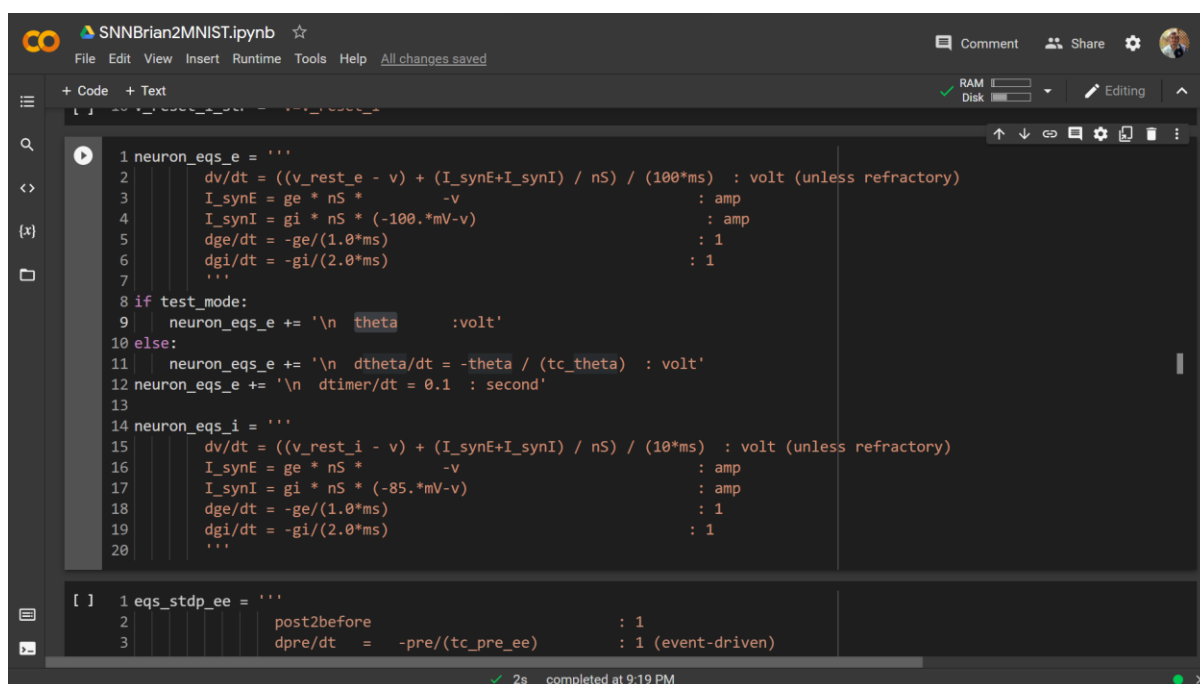


## پیوست اول: معرفی Google Colab

گوگل کولب بستری است که شرکت گوگل برای برنامه‌نویسی به زبان پایتون فراهم دیده. با استفاده از کولب شما می‌توانید بدون نیاز به سخت‌افزار قدرتمند (حتی بر روی گوشی هوشمند خود) و فقط بر روی یک مرورگر وب، بسیاری از پردازش‌های سنگین را انجام دهید.

درواقع گوگل به شما پردازنده و حافظه‌ای می‌دهد که شما از آن استفاده می‌کنید. در کولب علاوه بر CPU، دو پردازنده دیگر یعنی GPU و TPU نیز وجود دارند که در صورتی که کتابخانه‌هایی که شما استفاده می‌کنید از آن‌ها پشتیبانی کنند (مانند کتابخانه TensorFlow)، سرعت پردازش را چندین برابر افزایش می‌دهد.

مجموعه ارزشمندی از آموزش‌های گوگل کولب توسط گوگل بر روی یوتیوب قرار داده شده است.<sup>۱</sup>



```

1 neuron_eqs_e = '''
2     dv/dt = ((v_rest_e - v) + (I_synE+I_synI) / nS) / (100*ms) : volt (unless refractory)
3     I_synE = ge * nS * -v : amp
4     I_synI = gi * nS * (-100.*mV-v) : amp
5     dge/dt = -ge/(1.0*ms) : 1
6     dgi/dt = -gi/(2.0*ms) : 1
7     '''
8 if test_mode:
9     neuron_eqs_e += '\n theta :volt'
10 else:
11     neuron_eqs_e += '\n dtheta/dt = -theta / (tc_theta) : volt'
12     neuron_eqs_e += '\n dtimer/dt = 0.1 : second'
13
14 neuron_eqs_i = '''
15     dv/dt = ((v_rest_i - v) + (I_synE+I_synI) / nS) / (10*ms) : volt (unless refractory)
16     I_synE = ge * nS * -v : amp
17     I_synI = gi * nS * (-85.*mV-v) : amp
18     dge/dt = -ge/(1.0*ms) : 1
19     dgi/dt = -gi/(2.0*ms) : 1
20     '''

[ ] 1 eqs_stdp_ee = '''
2     post2before : 1
3     dpre/dt = -pre/(tc_pre_ee) : 1 (event-driven)

```

شکل ۳۸: تصویری از محیط گوگل کولب

<sup>۱</sup> <https://www.youtube.com/watch?v=inN8seMm7UI>

## پیوست دوم: کدهای پروژه

همه کدهای این پروژه در <https://github.com/bahramani/SNN-Brian2-MNIST> قابل دسترسی هستند.<sup>۱</sup>

متأسفانه نسخه‌های متفاوت کتابخانه‌های پایتون گاهی با یکدیگر مشکل دارند و برای آنکه بتوان از آنها در کنار یکدیگر استفاده کرد باید نسخه‌های مشخصی را نصب کرد. به همین جهت در جدل ۳ نسخه کتابخانه‌هایی که برای این پروژه از آنها استفاده کردم را آورده‌ام. اکیداً پیشنهاد می‌کنم که برای نصب کتابخانه‌ها بر روی کامپیوتر خود از نرم‌افزار <sup>۲</sup>Anaconda استفاده کنید.

Library	Version
<b>Python</b>	3.9.9
<b>Brian2</b>	2.5.0.2
<b>Brian2tools</b>	0.3
<b>Numpy</b>	1.22.0
<b>SciPy</b>	1.7.3
<b>Matplotlib</b>	3.5.1
<b>Keras</b>	2.6.0

<sup>۱</sup> در صورت داشتن سوال یا پیشنهادی می‌توانید از طریق [bahramani77@gmail.com](mailto:bahramani77@gmail.com) نکات خود را مطرح بفرمایید.

<sup>۲</sup> <https://www.anaconda.com/>

- [1] S. Bringsjord and N. S. Govindarajulu, “Artificial Intelligence,” in *The Stanford Encyclopedia of Philosophy*, Summer 2020., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2020. Accessed: Jan. 27, 2022. [Online]. Available: <https://plato.stanford.edu/archives/sum2020/entries/artificial-intelligence/>
- [2] H. Robinson, “Dualism,” in *The Stanford Encyclopedia of Philosophy*, Fall 2020., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2020. Accessed: Jan. 28, 2022. [Online]. Available: <https://plato.stanford.edu/archives/fall2020/entries/dualism/>
- [3] J. J. C. Smart, “The Mind/Brain Identity Theory,” in *The Stanford Encyclopedia of Philosophy*, Spring 2017., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2017. Accessed: Jan. 28, 2022. [Online]. Available: <https://plato.stanford.edu/archives/spr2017/entries/mind-identity/>
- [4] G. Boole, “THE MATHEMATICAL THEORIES OF LOGIC AND PROBABILITIES.,” p. 344.
- [5] C. E. Shannon, “A symbolic analysis of relay and switching circuits,” *Trans. Am. Inst. Electr. Eng.*, vol. 57, no. 12, pp. 713–723, Dec. 1938, doi: 10.1109/T-AIEE.1938.5057767.
- [6] M. M. Mano and M. D. Ciletti, *Digital design: with a introduction to the verilog hdl*, 5th ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2013.
- [7] B. Razavi, *Fundamentals of microelectronics*, Second edition. Hoboken, NJ: Wiley, John Wiley & Sons, Inc, 2014.
- [8] J. von Neumann, “First draft of a report on the EDVAC,” *IEEE Ann. Hist. Comput.*, vol. 15, no. 4, pp. 27–75, 1993, doi: 10.1109/85.238389.
- [9] J. Von Neumann and R. Kurzweil, *The computer & the brain*, 3rd ed. New Haven, Conn. ; London: Yale University Press, 2012.
- [10] G. E. Moore, “Cramming more components onto integrated circuits,” vol. 38, no. 8, p. 4, 1965.
- [11] “Moore’s law,” *Wikipedia*. Jan. 18, 2022. Accessed: Jan. 28, 2022. [Online]. Available:

[https://en.wikipedia.org/w/index.php?title=Moore%27s\\_law&oldid=1066544125](https://en.wikipedia.org/w/index.php?title=Moore%27s_law&oldid=1066544125)

- [12] Committee on Technical Assessment of the Feasibility and Implications of Quantum Computing, Computer Science and Telecommunications Board, Intelligence Community Studies Board, Division on Engineering and Physical Sciences, and National Academies of Sciences, Engineering, and Medicine, *Quantum Computing: Progress and Prospects*. Washington, D.C.: National Academies Press, 2019, p. 25196. doi: 10.17226/25196.
- [13] M. F. Bear, B. W. Connors, and M. A. Paradiso, *Neuroscience: exploring the brain*, Fourth edition. Philadelphia: Wolters Kluwer, 2016.
- [14] E. R. Kandel, J. Koester, S. Mack, and S. Siegelbaum, Eds., *Principles of neural science*, Sixth edition. New York: McGraw Hill, 2021.
- [15] D. Purves, Ed., *Neuroscience*, Sixth edition. New York: Oxford University Press, 2018.
- [16] K. D. Wise, A. M. Sodagar, Ying Yao, M. N. Gulari, G. E. Perlin, and K. Najafi, “Microelectrodes, Microelectronics, and Implantable Neural Microsystems,” *Proc. IEEE*, vol. 96, no. 7, pp. 1184–1202, Jul. 2008, doi: 10.1109/JPROC.2008.922564.
- [17] E. M. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*. The MIT Press, 2006. doi: 10.7551/mitpress/2526.001.0001.
- [18] R. Chaudhuri, B. Gerçek, B. Pandey, A. Peyrache, and I. Fiete, “The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep,” *Nat. Neurosci.*, vol. 22, no. 9, pp. 1512–1520, Sep. 2019, doi: 10.1038/s41593-019-0460-x.
- [19] T. P. Trappenberg, *Fundamentals of computational neuroscience*, 2nd ed. Oxford ; New York: Oxford University Press, 2010.
- [20] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge: Cambridge University Press, 2014. doi: 10.1017/CBO9781107447615.
- [21] D. Sterratt, *Principles of computational modelling in neuroscience*. Cambridge; New York: Cambridge University Press, 2011. Accessed: Jan. 29, 2022. [Online]. Available:

<http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=361614>

[22] ه. خوش آمدی، "مدل‌سازی نورون عصبی به روش باندگراف"، دانشگاه صنعتی خواجه‌نصیرالدین طوسی، ۲۰۱۶.

[23] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *J. Physiol.*, vol. 117, no. 4, pp. 500–544, 1952, doi: 10.1113/jphysiol.1952.sp004764.

[24] W. H. Hayt, J. E. Kemmerly, and S. M. Durbin, *Engineering circuit analysis*, 8th ed. New York: McGraw-Hill, 2012.

[25] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003, doi: 10.1109/TNN.2003.820440.

[26] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

[27] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jesús, *Neural network design*, 2nd edition. s.L: Martin T. Hagan, 2014.

[28] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/j.neunet.2014.09.003.

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, Massachusetts: The MIT Press, 2016.

[30] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, Dec. 1997, doi: 10.1016/S0893-6080(97)00011-7.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[32] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, 4th ed. Burlington, MA London: Academic Press, 2009.

[33] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

- [34] S. Lloyd, “Least squares quantization in PCM,” *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982, doi: 10.1109/TIT.1982.1056489.
- [35] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, Second edition. Cambridge, Massachusetts: The MIT Press, 2018.
- [36] “Neuroplasticity,” *Wikipedia*. Jan. 13, 2022. Accessed: Jan. 29, 2022. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=Neuroplasticity&oldid=1065348525>
- [37] D. EAGLEMAN, *LIVEWIRED: the inside story of the ever-changing brain*. S.I.: CANONGATE BOOKS LTD, 2021.
- [38] G. Bi and M. Poo, “Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type,” *J. Neurosci.*, vol. 18, no. 24, pp. 10464–10472, Dec. 1998, doi: 10.1523/JNEUROSCI.18-24-10464.1998.
- [39] C. D. Schuman *et al.*, “A Survey of Neuromorphic Computing and Neural Networks in Hardware,” *ArXiv170506963 Cs*, May 2017, Accessed: Jan. 26, 2022. [Online]. Available: <http://arxiv.org/abs/1705.06963>
- [40] C. Mead, “Neuromorphic Electronic Systems,” *Proc. IEEE*, vol. 78, p. 8, 1990.
- [41] A. M. Turing, “Computing Machinery and Intelligence,” *Mind New Ser.*, vol. 59, no. 236, pp. 433–460, 1950.
- [42] J. Schemmel, D. Briiderle, A. Griibl, M. Hock, K. Meier, and S. Millner, “A wafer-scale neuromorphic hardware system for large-scale neural modeling,” in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, Paris, France, May 2010, pp. 1947–1950. doi: 10.1109/ISCAS.2010.5536970.
- [43] H. Markram, “The Human Brain Project,” *Sci. Am.*, vol. 306, no. 6, pp. 50–55, May 2012, doi: 10.1038/scientificamerican0612-50.
- [44] D. Monroe, “Neuromorphic computing gets ready for the (really) big time,” *Commun. ACM*, vol. 57, no. 6, pp. 13–15, Jun. 2014, doi: 10.1145/2601069.
- [45] G. Indiveri *et al.*, “Neuromorphic Silicon Neuron Circuits,” *Front. Neurosci.*, vol. 5, 2011, doi: 10.3389/fnins.2011.00073.

- [46] B. Wang *et al.*, “Firing Frequency Maxima of Fast-Spiking Neurons in Human, Monkey, and Mouse Neocortex,” *Front. Cell. Neurosci.*, vol. 10, Oct. 2016, doi: 10.3389/fncel.2016.00239.
- [47] E. M. Izhikevich, “Which Model to Use for Cortical Spiking Neurons?,” *IEEE Trans. Neural Netw.*, vol. 15, no. 5, pp. 1063–1070, Sep. 2004, doi: 10.1109/TNN.2004.832719.
- [48] “Mike Davies: Realizing the Promise of Spiking Neuromorphic Hardware - YouTube.” <https://www.youtube.com/watch?v=jhQgElvtb1s> (accessed Feb. 02, 2022).
- [49] G. Indiveri, “Computation in Neuromorphic Analog VLSI Systems,” in *Neural Nets WIRN Vietri-01*, R. Tagliaferri and M. Marinaro, Eds. London: Springer London, 2002, pp. 3–20. doi: 10.1007/978-1-4471-0219-9\_1.
- [50] Y. Maeda and H. Makino, “A pulse-type hardware neuron model with beating, bursting excitation and plateau potential,” *Biosystems*, vol. 58, no. 1–3, pp. 93–100, Dec. 2000, doi: 10.1016/S0303-2647(00)00111-8.
- [51] A. Rubino, C. Livanelioglu, N. Qiao, M. Payvand, and G. Indiveri, “Ultra-Low-Power FDSOI Neural Circuits for Extreme-Edge Neuromorphic Intelligence,” *IEEE Trans. Circuits Syst. Regul. Pap.*, vol. 68, no. 1, pp. 45–56, Jan. 2021, doi: 10.1109/TCSI.2020.3035575.
- [52] J. Liu and C. Wang, “A Survey of Neuromorphic Engineering--Biological Nervous Systems Realized on Silicon,” in *2009 IEEE Circuits and Systems International Conference on Testing and Diagnosis*, Chengdu, China, Apr. 2009, pp. 1–4. doi: 10.1109/CAS-ICTD.2009.4960772.
- [53] F. Akopyan *et al.*, “TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, Oct. 2015, doi: 10.1109/TCAD.2015.2474396.
- [54] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, “The SpiNNaker Project,” *Proc. IEEE*, vol. 102, no. 5, pp. 652–665, May 2014, doi: 10.1109/JPROC.2014.2304638.
- [55] S. Furber, “Large-scale neuromorphic computing systems,” *J. Neural Eng.*, vol. 13, no. 5, p. 051001, Oct. 2016, doi: 10.1088/1741-2560/13/5/051001.

- [56] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, “The missing memristor found,” *Nature*, vol. 453, no. 7191, pp. 80–83, May 2008, doi: 10.1038/nature06932.
- [57] B. Linares-Barranco and T. Serrano-Gotarredona, “Memristance can explain Spike-Time-Dependent-Plasticity in Neural Synapses,” *Nat. Preced.*, Mar. 2009, doi: 10.1038/npre.2009.3010.1.
- [58] P. U. Diehl and M. Cook, “Unsupervised learning of digit recognition using spike-timing-dependent plasticity,” *Front. Comput. Neurosci.*, vol. 9, Aug. 2015, doi: 10.3389/fncom.2015.00099.
- [59] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, “Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing,” in *2015 International Joint Conference on Neural Networks (IJCNN)*, Killarney, Ireland, Jul. 2015, pp. 1–8. doi: 10.1109/IJCNN.2015.7280696.
- [60] Y. Lecun, “Gradient-Based Learning Applied to Document Recognition,” *Proc. IEEE*, vol. 86, no. 11, p. 47, 1998.
- [61] F. Jug, “On Competition and Learning in Cortical Structures,” ETH Zurich, 2012. doi: 10.3929/ETHZ-A-007140134.
- [62] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, “Immunity to Device Variations in a Spiking Neural Network With Memristive Nanodevices,” *IEEE Trans. Nanotechnol.*, vol. 12, no. 3, pp. 288–295, May 2013, doi: 10.1109/TNANO.2013.2250995.
- [63] D. Goodman, “Brian: a simulator for spiking neural networks in Python,” *Front. Neuroinformatics*, vol. 2, 2008, doi: 10.3389/neuro.11.005.2008.
- [64] M. Stimberg, R. Brette, and D. F. Goodman, “Brian 2, an intuitive and efficient neural simulator,” *eLife*, vol. 8, p. e47314, Aug. 2019, doi: 10.7554/eLife.47314.
- [65] J.-P. Pfister, “Triplets of Spikes in a Model of Spike Timing-Dependent Plasticity,” *J. Neurosci.*, vol. 26, no. 38, pp. 9673–9682, Sep. 2006, doi: 10.1523/JNEUROSCI.1425-06.2006.
- [66] J. R. Searle, “Minds, brains, and programs,” *Behav. Brain Sci.*, vol. 3, no. 3, pp. 417–424, Sep. 1980, doi: 10.1017/S0140525X00005756.
- [67] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, and T. Masquelier, “SpykeTorch: Efficient Simulation of Convolutional Spiking Neural Networks



With at Most One Spike per Neuron,” *Front. Neurosci.*, vol. 13, p. 625, Jul. 2019, doi: 10.3389/fnins.2019.00625.

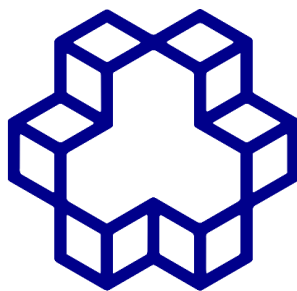
## Abstract

The realization of a processor that is more powerful and consumes less power than what we have got today is a challenging task that requires the advancement of micro-fabrication technology. This is because it is believed that technology has almost reached the boundary of shrinking transistors on a chip. Therefore, providing alternative ways for the enhancement of the performance of electronic processors seems to be both necessary and inevitable.

The brain is the most intelligent processor humans have ever encountered. Therefore, if we take inspiration from the architecture of the brain to design a processor, we might be able to provide a solution to the aforementioned challenge. Architectures that mimic the structure and functions of the brain are called *Neuromorphic* architecture. Thus, our goal in this thesis is to first study neuromorphic processors in general and then design a simple neuromorphic processor for machine learning applications. For this purpose, first, the physiological structure of the brain has been studied, and then we have used computational models for the implementation of our neuromorphic processor. This way, proper models of network components (*i.e.*, *neuron* and *synapse*) as well as learning algorithms are presented and fully explained. Subsequently, a comparison between *spiking neural networks* and *artificial neural networks* is presented.

The pros and cons of neuromorphic processors are then presented. Following that, a neuromorphic processor is implemented using the Python programming language. This processor is designed and implemented for English handwritten digit recognition (MNIST dataset). Finally, the performance of this processor is evaluated and possible ways to enhance and improve its performance are suggested.

**Keywords:** Neuromorphic, Spiking Neural Network, Machine Learning, Computational Neuroscience, Digit recognition, MNIST



*Electrical Engineering Department*  
*K. N. Toosi University of Technology*  
*Tehran*

*Submitted as thesis in partial fulfillment*  
*of the requirements for Electrical Engineering bachelor's degree*

**Design and Implementation of a Fully-Digital  
Neuromorphic Processor**

*By Amirreza Bahramani*

*Under Supervision of Dr. Amir M. Sodagar*

**February 2022**