

باسمه تعالی



دانشگاه صنعتی شریف

دانشکده مهندسی برق

درس علوم اعصاب: یادگیری، حافظه، شناخت

تمرین مطالعاتی سری اول

استاد درس: دکتر کربلایی آقاجان

امیرحسین افشارراد

۹۵۱۰۱۰۷۷

۸ آبان ۱۳۹۷

۱ مقاله اول

۱.۱ علل استفاده از تابع فعال‌سازی خطی یکسوسده

تابع فعال‌سازی خطی یکسوسده (Rectified Linear Units - ReLU) با توجه به ضابطه‌ی ساده‌تر خود (در مقایسه با توابع فعال‌سازی دیگری مانند sigmoid) پیاده‌سازی روش gradient-descent را آسان‌تر می‌کند؛ و به تبع آن باعث افزایش سرعت فرایند آموزش (training) می‌شود. ضمناً لازم به ذکر است که در عمل، مشاهده می‌شود که استفاده از این تابع فعال‌سازی نتایج عملیاتی بسیار خوبی دارد و این موضوع در کنار سرعت بالا، از علل استفاده از این تابع فعال‌سازی است.

۲.۱ مقایسه‌ی شبکه‌های عمیق و کم‌عمق

طبق تعریف، شبکه‌ی کم عمق شبکه‌ای است که تنها یک لایه‌ی پنهان داشته باشد، در حالی که شبکه‌ی عمیق بیش از یک لایه‌ی پنهان دارد.

از نظر قدرت محاسبه، می‌توان گفت که هر دو نوع شبکه قادر به تخمین زدن هر تابعی هستند. شبکه‌ی کم عمق می‌تواند هر تابع پیچیده‌ای را به شکل یک تابع چندضابطه‌ای مدل کند؛ اما ممکن است تعداد ضوابط بسیار زیاد شود. ضمناً شبکه‌ی عمیق می‌تواند عملکردی دقیقاً مشابه با شبکه‌ی کم عمق را پیاده‌سازی کند.

با این حال، تفاوت مهم این دو نوع شبکه آن است که شبکه‌ی عمیق قادر است توابع پیچیده را به شکلی مختصرتر و با تعداد واحدهای محاسباتی کمتری مدل کند. علت مهم این امر، آن است که شبکه‌ی عمیق می‌تواند خروجی‌های لایه‌های مختلف را (به صورت سلسله‌مراتبی) مجدداً مورد پردازش قرار دهد و نتایج را گسترش دهد؛ در حالی که این کار در شبکه‌ی کم عمق امکان‌پذیر نیست.

۳.۱ مقایسه‌ی شبکه‌های recurrent و feedforward

شبکه‌های مستقیم (feedforward) تنها قادر به محاسبه و تخمین توابع استاتیک (غیر متغیر با زمان) هستند. در طرف مقابل، شبکه‌های بازگشتی (recurrent) می‌توانند دینامیک متغیر با زمان سیستم‌ها را محاسبه کرده و تخمین بزنند و در لحظات مختلف، حالت سیستم را در خود نگه‌داری کنند. در این نوع شبکه‌ها، اتصالات تنها از یک لایه به لایه‌ی بعدی برقرار نیست، بلکه هر لایه می‌تواند اتصالاتی به خود، و نیز به لایه‌های قبلی داشته باشد. به این ترتیب، مفهوم لایه‌ها تقریباً معنای اصلی خود را از دست می‌دهند، چرا که می‌توان تمامی لایه‌های پنهان را به شکل یک لایه در نظر گرفت که درون خود دارای اتصالاتی است.

برای تعیین وزن‌های هر دو نوع شبکه از روش انتشار به عقب (backpropagation) استفاده می‌شود، و می‌توان شبکه‌های بازگشتی را نیز به نوعی مشابه با شبکه‌های مستقیم در نظر گرفت، تنها به این تفاوت که در بُعد زمان گسترش یافته‌اند. یک تفاوت قابل توجه در آموزش برخی از شبکه‌های بازگشتی (echo-state networks) آن است که وزن بین لایه‌های پنهان مورد آموزش قرار نمی‌گیرد و به صورت تصادفی انتخاب می‌گردد، تا اثر موج‌گونه‌ی ورودی متغیر با زمان و الگوهای زمانی موجود در آن، در نهایت بر شبکه حاکم شود.

۴.۱ مفاهیم محو و انفجار گرادیان در شبکه‌های عصبی

در آموزش شبکه‌های عصبی بازگشتی (recurrent)، بعضاً مشاهده می‌شود که مقدار گرادیان بسیار کوچک یا صفر می‌شود (و جهت ادامه‌ی فرایند نامشخص می‌گردد، یا به عبارت دیگر وزن مورد نظر دیگر تغییر نمی‌کند و نمی‌تواند بهبود یابد) یا مقدار آن بسیار بزرگ می‌شود (منفجر می‌گردد).

علت این امر آن است که برای یادگیری وابستگی‌های زمانی طولانی‌مدت در شبکه‌های بازگشتی، محاسبه‌ی گرادیان (از رابطه‌ی مشتق زنجیری) از حاصل ضرب تعداد زیادی عبارت محاسبه می‌گردد که امکان صفر یا بسیار بزرگ شدن را ایجاد می‌کند.

برای حل این مشکل از ساختار Long Short-Term Memory (LSTM) استفاده می‌شود. در این ساختار، واحدهای محاسباتی قادر به نگه‌داری خاطرات کوتاه‌مدت برای هر زمان طولانی و دل‌خواهی هستند که این امر باعث می‌شود حساسیت و مشکل ناشی از اختلاف زمانی طولانی بین الگوهای بامعنی از بین برود.

۵.۱ شباهت شبکه‌های مصنوعی تشخیص تصاویر با شبکه‌های زیستی مغز

در شبکه‌های مصنوعی پردازشگر تصویر، لایه‌های ابتدایی عملکردی مشابه با نورون‌های V1 (primary visual cortex) دارند. این لایه‌ها به خصوصیت‌هایی از جنس فیلترهای gabor حساسیت دارند؛ یعنی الگوهای ساده‌ای به شکل بافت‌های تکرارشونده (مانند خطوط سیاه و سفید موازی با فرکانس مکانی ثابت) را تشخیص می‌دهند.

۶.۱ Adversarial Attacks

- برخی روش‌های دفاع در مقابل حمله‌های adversarial:

۱ – Network Distillation

در این روش تلاش می‌شود تا اطلاعات از شبکه‌ی عصبی اولیه به یک شبکه‌ی عصبی کوچک‌تر منتقل شود؛ و با کاهش حساسیت نسبت به تغییرات و اغتشاشات کوچک در ورودی، مقاومت شبکه‌ی عصبی در مقابل حملات adversarial بیشتر شود.

۲ – Adversarial (Re)training

در این روش، در هر مرحله از فرایند یادگیری شبکه، مثال‌هایی از نوع adversarial تولید شده و در میان داده‌های یادگیری قرار می‌گیرد تا فرایند آموزش شبکه‌ی عصبی شامل مثال‌های متنوعی از نوع adversarial باشد و به این ترتیب، شبکه‌ی عصبی در نهایت نسبت به این نوع از ورودی مقاومت باشد.

۳ – Adversarial Detecting

در این روش، یک شبکه‌ی کمکی جداگانه وظیفه‌ی تعیین adversarial بودن یا نبودن را به عهده می‌گیرد، به گونه‌ای که پاسخ آن همواره به شکل باینری بوده و صرفاً مشخص می‌کند که آیا ورودی داده‌شده، مجاز است یا از نوع adversarial می‌باشد. در این حالت، با استفاده از یک شبکه‌ی کمکی، تلاش می‌شود تا از ورود نمونه‌های adversarial به شبکه‌ی اصلی به طور کلی جلوگیری شود.

۴ – Input Reconstruction

در این روش، با استفاده از رویکردهای مختلفی، سعی می‌شود تا با انجام یک تبدیل بر روی ورودی adversarial آن را به یک ورودی سالم و مجاز تبدیل کنند. در این حالت، بعضاً شبکه‌های عصبی مجزایی آموزش داده می‌شوند تا ورودی‌های adversarial را به ورودی‌های سالم تبدیل کرده و در ادامه، این ورودی‌های فیلترشده و سالم به شبکه‌ی اصلی وارد شوند. (denoising autoencoder networks)

۵ – Network Verification

این روش تأکید بر مطالعه و شناخت دقیق شبکه‌ی عصبی آموزش داده‌شده دارد؛ به گونه‌ای که بتوان به صورت تئوری واکنش و رفتار آن را به ورودی‌های مختلف پیش‌بینی و مشخص کرد. در نتیجه، می‌توان مطمئن بود که رفتار شبکه در مقابل ورودی‌های adversarial چگونه است و در صورت امکان، رفتار آن را بهبود بخشید؛ و اگر

ممکن نیست، در مورد ورودی‌هایی که شبکه را به خطا می‌اندازند، آگاه باشیم.

۶- Ensembling Defenses

این روش در واقع روش جدیدی نیست؛ و صرفاً بیانگر آن است که بهتر است برای مقاوم‌سازی هرچه بیشتر شبکه‌های عصبی در برابر حملات adversarial از ترکیب و ادغام چندین روش مختلف دفاعی (نظیر روش‌های مذکور) استفاده کنیم.

مراجع:

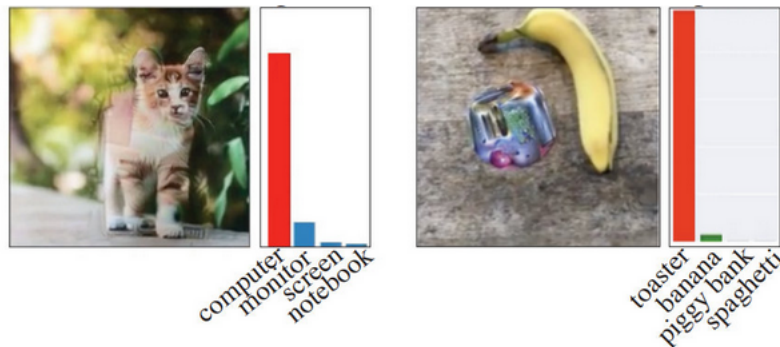
1. Xiaoyong Yuan, Pan He, Qile Zhu, Xiaolin Li, "Adversarial Examples: Attacks and Defenses for Deep Learning", 2018.
2. N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in Security and Privacy (SP), 2016 IEEE Symposium on . IEEE, 2016, pp. 582–597.
3. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv preprint arXiv:1412.6572, 2014.
4. J. Lu, T. Issaranon, and D. Forsyth, "Safetynet: Detecting and rejecting adversarial examples robustly," ICCV, 2017.
5. J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," Proceedings of 5th International Conference on Learning Representations (ICLR), 2017.
6. S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," Proceedings of the International Conference on Learning Representations (ICLR), 2015.
7. G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," arXiv preprint arXiv:1702.01135 , 2017.
8. Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," arXiv preprint arXiv:1710.10766 , 2017.

● روش‌های تولید تصاویری که علاوه بر شبکه‌های عصبی، انسان را نیز به اشتباه بیندازند:

تولید تصاویر adversarial به گونه‌ای که انسان را به اشتباه بیندازند، در مقایسه با حملات adversarial عادی که شبکه‌ی عصبی مصنوعی را به اشتباه می‌اندازند، به مراتب پیچیده‌تر است. یک نکته آن است که اصولاً تعریف اولیه‌ی ما از این نوع تصاویر، مثال‌هایی است که اگر چه برای انسان واضح هستند و با تصویر اصلی تفاوتی نمی‌کنند، اما شبکه‌ی عصبی را به اشتباه می‌اندازند. با توجه به این توضیح، مثال‌های رایج و معروفی که برای خطای دید انسان وجود دارند نمی‌توانند پاسخ مناسبی برای این قسمت باشند، چرا که اصولاً هدف در این بخش تولید تصاویری است که اگر چه تفاوت خاصی با تصویر اولیه ندارند، اما اشتباه طبقه‌بندی می‌شوند و همین هدف با توجه به توضیحات داده‌شده سؤال برانگیز است، چرا که اگر تعریف ما از عدم ایجاد تغییر خاص در تصویر معادل با درک انسانی باشد، دیگر نمی‌توان انتظار داشت که چنین تصویری درک انسانی را به اشتباه بیندازد!

با همه‌ی این توضیحات، اگر به مقالات مراجعه کنیم و به بررسی پیاده‌سازی حملات adversarial بر روی انسان بپردازیم، یک روش رایج آن است که مثال‌هایی از تصاویر adversarial به دست آورند که قادر به فریب دادن تعداد قابل توجهی از شبکه‌های عصبی مصنوعی باشند. چنین مثال‌هایی احتمالاً بتوانند انسان را نیز دچار اشتباه کنند؛ اما یک نکته‌ی مهم آن است که برای انجام چنین آزمایشی باید زمان بسیار کوتاهی به فرد داد تا بین دو دسته تصمیم بگیرد (تا بررسی کنیم که آیا تصمیم درست است یا فرد فریب تصویر adversarial را خورده است)، چرا که با گذشت زمان، تفاوت‌های بنیادین شبکه عصبی مصنوعی و طبیعی خود را نشان می‌دهند و فرد در بیشتر موارد قادر به تشخیص دسته‌بندی صحیح خواهد بود.

همان طور که گفته شد، تصاویری می‌توانند گزینه‌های احتمالی فریب انسان باشند که شبکه‌های عصبی متعددی را فریب داده باشند. ادعا می‌شود که در چنین تصاویری، نشانه‌هایی ظاهر می‌شود که توسط انسان نیز محسوس است. به عنوان مثال، شکل ۱ دو نمونه از چنین تصاویری را نشان می‌دهد.



شکل ۱: نمونه‌هایی از تصاویری که قادر به فریب تعداد زیادی شبکه عصبی مصنوعی بوده و گزینه‌های احتمالی فریب انسان هستند

تصویر سمت چپ گربه‌ای است که توسط شبکه‌های عصبی به عنوان کامپیوتر طبقه‌بندی می‌شود و تصویر دوم عکسی از یک موز است که توسط این شبکه‌ها به عنوان توستر طبقه‌بندی می‌شود. همان طور که گفته شد، نشانه‌هایی از لبه‌های تیز کامپیوتر در تصویر سمت چپ، و موجودی در وسط تصویر سمت راست که بی‌شباهت به توستر نیست، از جمله مواردی است که برای انسان محسوس است؛ با این حال در این مورد نمی‌توان ادعا کرد که انسان این تصاویر را به اشتباه تشخیص می‌دهد.

نمونه‌ی موجود در شکل ۲ بیشتر قابل اعتنا است. تصویر سمت چپ نشان‌دهنده‌ی یک گربه است، و تصویر سمت راست نیز همان گربه است اما با این تفاوت که عکس دچار حمله adversarial شده است. در این حالت مشاهده می‌شود که امکان تشخیص اشتباه توسط انسان به مراتب قابل توجه است؛ به خصوص که (به دلایل ذکر شده) برای آزمایش انسان زمان بسیار کمی برای انتخاب یکی از دو طبقه‌بندی (سگ یا گربه) به وی می‌دهند، و مشاهده می‌شود که اشتباه در طبقه‌بندی بسیار زیاد است.

مراجع:

1. Gamaleldin F. Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alex Kurakin, Ian Goodfellow, Jascha Sohl-Dickstein, "Adversarial Examples that Fool both Computer Vision and Time-Limited Humans", 2018.
2. <https://spectrum.ieee.org/the-human-os/robotics/artificial-intelligence/hacking-the-brain-with-adversarial-images>



شکل ۲: تصویر چپ: تصویر اصلی گربه / تصویر راست: تصویر گربه پس از حمله adversarial که به اشتباه به عنوان سگ طبقه‌بندی می‌شود

۷.۱ بررسی ناحیه‌ی IT در مغز و شباهت‌های آن با شبکه‌ها عصبی

ناحیه‌ی IT (Inferior Temporal cortex) معروف به طبقه‌بندی داده‌ها به گروه‌های مختلف است. این طبقه‌بندی کاری سطح بالاست و فراتر از تشخیص‌های الگو و فرایندهای پردازشی اولیه است؛ و در واقع کاری نظیر طبقه‌بندی محتوایی انجام می‌دهد.

در مطالعات انجام‌شده مشاهده می‌شود که تنها شبکه‌های عصبی عمیق (عمدتاً از نوع مبتنی بر کانولوشن) قادر به توجیه و مدل‌کردن رفتار IT هستند و می‌توانند طبقه‌بندی‌های بین گروهی و درون‌گروهی مشابه با IT را انجام دهند. همچنین مشاهده می‌شود که هر چه به سمت لایه‌های بالاتر و عمیق‌تر این شبکه‌ها حرکت می‌کنیم، این شباهت بیشتر می‌شود، و می‌توان گفت که تنهای لایه‌های بالایی (انتهایی) شبکه‌های عصبی عمیق می‌توانند عملکردی مشابه IT داشته باشند.

۲ مقاله دوم

۱.۲ ارجحیت مدل ارائه شده در مقایسه با مدل های معمول RNN

تفاوت مشخص مدل ارائه شده در این مقاله با مدل های رایج مربوط به در نظر گرفتن نوروهای excitatory و inhibitory است. اصولاً شبکه های عصبی متعددی می توانند با ساختارهای داخلی و دینامیک های متفاوت به نتایج یکسان (و صحیح) برسند. در این گونه موارد، مشاهده می شود که عملکرد و خروجی شبکه ی عصبی مصنوعی با آنچه که در واقعیت (و مثلاً در مغز انسان) اتفاق می افتد یکسان است، اما نحوه ی عملکرد داخلی آن ها ممکن است کاملاً متفاوت باشد. بدیهی است عملکرد داخلی چنین شبکه هایی نمی تواند برای شبیه سازی و شناخت بیشتر رفتار درونی مغز مناسب باشد، بنابراین برای رسیدن به این هدف جدید، باید شبکه هایی را مورد بررسی و آموزش قرار داد که از لحاظ عملکرد درونی نیز مشابه با شبکه های طبیعی عمل کنند.

با توجه به این توضیحات، ارزش کار این مقاله آن است که با ارائه ی مدلی مبتنی بر وجود نوروهای excitatory و inhibitory قادر است شبکه های عصبی ای را تولید کند که علاوه بر ارائه ی خروجی های مطلوب، از نظر دینامیک داخلی نیز عملکردی مشابه با مغز را دارا باشند و بتوانند در بررسی و شبیه سازی رفتار داخلی مغز مؤثر واقع شوند.

۲.۲ روش آموزش شبکه

روش مورد استفاده در این مقاله، روش «کاهش گرادیان تصادفی» – minibatch Stochastic Gradient Descent (SGD) – است. علت استفاده از این روش، مزیت هایی است که در مقایسه با سایر روش های رایج دارد. در متن مقاله، این روش را با دو روش Hessian-Free (HF) و First-Order Reduced and Controlled Error (Force) مقایسه کرده اند. روش SGD بر خلاف FORCE (و مشابه با HF) امکان فرمول بندی آسان مسأله ی آموزش RNN را به صورت یک مسأله ی بهینه سازی تابع هدف در اختیار قرار می دهد، که با تغییر دادن آن می توان به جواب های مختلفی دست یافت. همچنین روش SGD بر خلاف HF (و مشابه با FORCE) می تواند پارامترهای محاسبه شده را بعد از هر trial به روزرسانی کند، و مجبور به دریافت تمامی داده ها در یک مرحله نمی باشد. به عبارت دیگر، امکان یادگیری به صورت آنلاین (online learning) وجود دارد. بدین ترتیب، امکان بررسی اثرات بین آزمایشی across-trial به وجود می آید.

البته باید توجه داشت که هیچ کدام از روش های مذکور دقیقاً آنچه در مغز اتفاق می افتد یکسان نیستند. روش مورد استفاده در مغز، Spike-Timing Dependent Plasticity (STDP) نام دارد که بر اساس آن، اتصالات بین نوروها بر مبنای زمان بندی اسپایک زدن آن ها و سنکرون بودن این زمان بندی ها تنظیم می شود.

۳.۲ چگونگی اضافه کردن نوروهای excitatory و inhibitory به RNN های معمولی

مدل اصلی و ابتدایی شبکه در معادلات ۱ تا ۳ مشاهده می شود که در آن τ ثابت زمانی، \mathbf{x} بردار جریان های ورودی، W^{rec} ماتریس اتصالات بازگشتی (recurrent)، \mathbf{r} بردار firing rate، \mathbf{z} بردار خروجی شبکه، W^{in} ماتریس وزن ورودی، \mathbf{u} بردار ورودی های شبکه، و ξ نویز است.

$$\tau \dot{\mathbf{x}} = -\mathbf{x} + W^{rec} \mathbf{r} + W^{in} \mathbf{u} + \sqrt{2\tau\sigma_{rec}^2} \xi \quad (۱)$$

$$\mathbf{r} = [\mathbf{x}]_+ \quad (۲)$$

$$\mathbf{z} = W^{out} \mathbf{r} \quad (۳)$$

برای تکمیل این مدل به منظور توجیه نوروهای excitatory و inhibitory، باید به این نکته توجه داشت که اگر اتصال از نورون j به نورون i excitatory باشد، خواهیم داشت: $W_{ij}^{rec} > 0$
 همچنین اگر اتصال مذکور inhibitory باشد، داریم: $W_{ij}^{rec} < 0$
 همچنین به طور کلی اگر نورون j یک نورون excitatory باشد خواهیم داشت: $\forall i : W_{ij}^{rec} \geq 0$
 همچنین در مورد نورون inhibitory نیز به طریق مشابه داریم: $\forall i : W_{ij}^{rec} \leq 0$

حال برای کلی سازی مدل به منظور توجیه نوروهای excitatory و inhibitory، ماتریس W^{rec} را به شکل حاصل ضرب دو ماتریس دیگر در نظر می گیریم:

$$W^{rec} = W^{rec,+} D \quad (4)$$

که $W^{rec,+}$ ماتریسی نامنفی و D ماتریسی قطری با مقادیر 1 یا -1 است. هر عنصر از ماتریس D متناظر با یک نورون است که excitatory یا inhibitory بودن آن با 1 یا -1 مشخص می شود. همچنین در اثر ضرب ماتریس ها، ماتریس W^{rec} به دست می آید که ستون های متناظر با نوروهای excitatory همگی دارای مقادیر مثبت (یا صفر) و ستون های متناظر با نوروهای inhibitory همگی دارای مقادیر منفی (یا صفر) هستند: (در معادله ۵، عناصر خالی معادل صفر هستند)

$$\underbrace{\begin{bmatrix} & + & + & + & - \\ + & & + & + & - \\ + & + & & + & - \\ + & + & + & & - \\ + & + & + & + & \end{bmatrix}}_{W^{rec}} = \underbrace{\begin{bmatrix} & + & + & + & + \\ + & & + & + & + \\ + & + & & + & + \\ + & + & + & & + \\ + & + & + & + & \end{bmatrix}}_{W^{rec,+}} \underbrace{\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & -1 \end{bmatrix}}_D \quad (5)$$

۴.۲ مشکل انفجار گرادیان

رابطه ی مورد استفاده در هر گام روش کاهش گرادیان در معادله ۶ مشاهده می شود. (θ بردار پارامترهای مدل است)

$$\theta^{(i)} = \theta^{(i-1)} + \delta\theta^{(i-1)} \quad (6)$$

که در آن، با محاسبه ی گرادیان گام بعدی به دست می آید:

$$\delta\theta^{(i-1)} = -\eta \nabla \mathcal{E}^{(i-1)} \quad (7)$$

در این روش، هنگامی که مقدار گرادیان بسیار بزرگ می شود (انفجار گرادیان)، روش مورد استفاده آن است که اصطلاحاً گرادیان را «می بُرنند»، یا به عبارت دیگر اجازه ی عبور آن از یک مقدار مشخص G را نمی دهند:

$$\delta\theta^{(i-1)} = \begin{cases} -\eta \nabla \mathcal{E}^{(i-1)} \times \frac{G}{|\nabla \mathcal{E}^{(i-1)}|} & |\nabla \mathcal{E}^{(i-1)}| > G \\ -\eta \nabla \mathcal{E}^{(i-1)} & \text{otherwise} \end{cases} \quad (8)$$

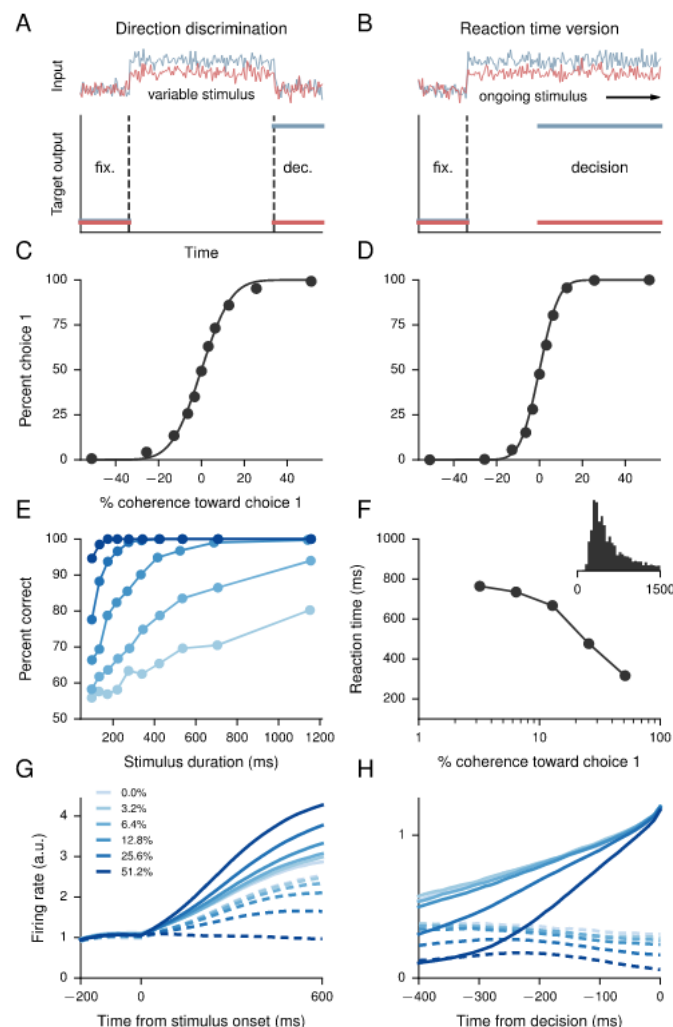
۵.۲ بررسی تسک های مختلف مبتنی بر مدل ارائه شده

۱.۵.۲ تسک تصمیم گیری ادراکی

این تسک مبتنی بر فرایند تصمیم گیری و انتخاب توسط شبکه ی عصبی است. این تصمیم گیری بین دو گزینه صورت می گیرد، به گونه ای که شواهدی در ارتباط با گزینه های موجود به شبکه ارائه می شود و در ادامه، شبکه ی عصبی باید یکی از این دو گزینه را (بر اساس شواهد) انتخاب کند. منظور از شواهد، دو تحریک، یکی مربوط به گزینه ۱ و دیگری مربوط به گزینه ۲ است. تحریک ها از آزمایش تشخیص حرکت تصادفی نقطه گرفته شده اند. ضمناً برای اعمال تحریک ها دو روش مختلف اتخاذ شده است؛ در روش اول دو تحریک برای مدت زمان معینی داده می شوند، سپس شبکه اقدام به تصمیم گیری می نماید؛

و در روش دوم تحریک‌ها برای زمانی نامحدود موجود هستند و شبکه در هر زمان قادر به تصمیم‌گیری است، اما سرعت واکنش شبکه نیز مورد بررسی و ارزیابی قرار می‌گیرد. ضمناً در هر دو حالت، در زمان ابتدایی آزمایش تحریکی وجود ندارد و انتظار داریم که خروجی شبکه‌ی عصبی در این زمان به ازای هر دو خروجی مقدار صفر (منظور از مقدار صفر، مقدار پایین یا به عبارت دیگر مقدار صفر منطقی است) داشته باشد. در ادامه و در زمان تصمیم‌گیری، شبکه با بالا بردن خروجی متناظر با گزینه‌ی مد نظر، آن گزینه را انتخاب می‌کند.

نتایج حاصل از این آزمایش قابل توجه هستند و عملکرد خوبی از شبکه‌های عصبی را به نمایش می‌گذارند. شکل ۳ نتایج مختلفی از دو آزمایش را نشان می‌دهد که در ادامه به توضیح آن می‌پردازیم. دقت کنید که ستون سمت چپ مربوط به روش اول اعمال تحریک (با مدت زمان مشخص) و ستون سمت راست مربوط به روش دوم اعمال تحریک (با زمان نامتناهی) می‌باشد.



شکل ۳: نتایج آزمایش تصمیم‌گیری ادراکی

پیش از توضیح نتایج، لازم به ذکر است که منظور از پارامتر $\text{coherence toward choice } 1/2$ آن است که دو تحریک داده‌شده، تا چه میزان تفاوت قابل توجهی میان دو انتخاب دارند، به گونه‌ای که هر چه مقدار این پارامتر (مثلاً در جهت گزینه ۱) بیشتر باشد، یعنی تحریک مربوط به این گزینه تفاوت معنادارتری نسبت به تحریک دیگر داشته و شواهد بیشتری برای انتخاب آن داریم.

شکل‌های A و B نشان‌دهنده‌ی فرایند انجام آزمایش هستند که پیش‌تر توضیح داده شد.

شکل های C و D درصد انتخاب گزینه ۱ را بر حسب coherence در جهت این گزینه نشان می دهند، که می بینیم با افزایش coherence در جهت این انتخاب، دقت پاسخ گویی نیز بسیار بالا می رود.

شکل E درصد پاسخ صحیح را بر حسب مدت زمان اعمال تحریک (در روش اعمال تحریک با زمان محدود) نشان می دهد که نمودارهای مختلف آن مربوط به coherence های متفاوت هستند. مشاهده می شود که در coherence های بالا، تقریباً درصد پاسخ گویی شبکه مستقل از طول تحریک و در حدود 100% است.

شکل F زمان واکنش شبکه را بر حسب coherence نشان می دهد. (برای روش دوم اعمال تحریک) که مشاهده می شود با افزایش coherence، سرعت واکنش و تصمیم گیری شبکه افزایش می یابد.

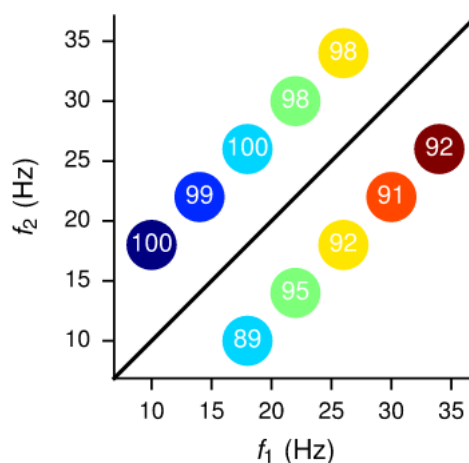
نهایتاً شکل های G و H نیز نشان دهنده firing rate بر حسب زمان اعمال تحریک می باشند. نمودارهای مختلف مربوط به coherence های مختلف هستند که مجدداً مشاهده می شود coherence بیشتر باعث عملکرد سریع تر و بهتر شبکه ی عصبی می شود.

نهایتاً به عنوان جمع بندی می توان گفت که عملکرد شبکه ی عصبی در انجام تصمیم گیری معقول و قابل قبول بوده و نتایج با انتظارات ما در تطابق نسبی هستند.

۲.۵.۲ تسک پارامتری حافظه ی کاری (Parametric working memory task)

هدف از این تسک سنجش حافظه ی شبکه ی عصبی است، به گونه ای که مدتی پس از دریافت یک ورودی، بتواند آن را مورد بررسی و مقایسه قرار دهد. در این تسک، دو ورودی با فرکانس های متفاوت و با اختلاف زمانی مشخصی (حدود ۳ ثانیه) به شبکه ی عصبی داده می شوند، و سپس شبکه باید در خروجی، مشخص کند که فرکانس کدام ورودی بیشتر بوده است.

شکل ۴ درصد پاسخ گویی این شبکه ی عصبی را در تشخیص فرکانس بالاتر بر حسب فرکانس های داده شده نشان می دهد؛ و مشخص است که نتایج قابل قبولی به دست آمده اند. ضمناً بنابر گزارش مقاله، تمامی درصدهای مشاهده شده بیشتر از 85% بوده اند.



شکل ۴: درصد پاسخ صحیح شبکه ی عصبی در تشخیص فرکانس بالاتر

۳ مقاله سوم

۱.۳ انواع کدهای نورونی

• rate coding: در این روش کدگذاری، مبنای انتقال اطلاعات آهنگ (فرکانس) اسپایک زدن نورون‌ها است، و الگوی دقیق زمانی آن‌ها مد نظر نیست.

• temporal coding: در این روش کدگذاری، هنگامی که دقت زمانی بالایی مد نظر است؛ الگوی زمانی اسپایک‌ها و فاصله‌ی زمانی‌های آن‌ها در کد حامل اطلاعات است.

به عنوان مثال، دو کد 000111000111 و 00110011 از منظر rate coding یکسان هستند، چرا که آهنگ نوسان اسپایک‌ها در هر دو ثابت است؛ اما این دو از منظر temporal coding متفاوت هستند، چرا که رفتار زمانی آن‌ها به وضوح متفاوت است.

رویکرد این مقاله در طراحی و مدل‌سازی، استفاده از مدل temporal coding بوده است.

۲.۳ مدل‌سازی محاسباتی

در مدل محاسباتی ارائه‌شده، برای نورون j ، مجموعه Γ_j نشان‌گر مجموعه‌ی نورون‌هایی است که به این نورون متصل هستند و اسپایک زدن آن‌ها بر نورون j اثرگذار است؛ بنابراین نورون مورد نظر مجموعه‌ای از اسپایک‌ها را در زمان‌های t_i دریافت می‌کند که $i \in \Gamma_j$.

در مدل ارائه‌شده، هر نورون در هر بازه‌ی شبیه‌سازی حداکثر یک اسپایک می‌زند، و شرط وقوع اسپایک نیز آن است که متغیر حالت داخلی این نورون به حد آستانه‌ی v برسد. متغیر حالت مربوط به نورون j تحت تأثیر اسپایک‌های ورودی قرار دارد، به گونه‌ای اثر آن از طریق یک تابع پاسخ اسپایک $\varepsilon(t)$ اعمال می‌شود. همچنین اثر هر کدام از اسپایک‌های ورودی به وسیله‌ی وزن w_{ij} نیز تنظیم می‌شود، بنابراین به معادله‌ی ۹ دست خواهیم یافت.

$$x_j(t) = \sum_{i \in \Gamma_j} w_{ij} \varepsilon(t - t_i) \quad (9)$$

همچنین تابع پاسخ اسپایک نیز از معادله ۱۰ محاسبه می‌شود که در آن، منظور از $u(t)$ تابع پله واحد است و τ نیز ثابت زمانی نزول پتانسیل غشاء نورون است.

$$\varepsilon(t) = \frac{t}{\tau} e^{1 - \frac{t}{\tau}} u(t) \quad (10)$$

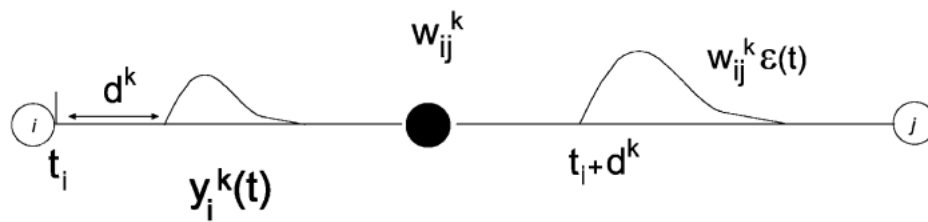
همچنین برای تقویت مدل، هر اتصال بین نورونی را شامل m اتصال سیناپسی در نظر می‌گیرند که هر ترمینال آن وزن و تأخیر متفاوتی دارد. معادله ۱۱ اثر هر ترمینال سیناپسی را (بدون در نظر گرفتن وزن) نشان می‌دهد.

$$y_i^k(t) = \varepsilon(t - t_i - d^k) \quad (11)$$

منظور از d^k در معادله ۱۱ تأخیر ترمینال سیناپسی k ام است، که به معنی اختلاف زمانی اسپایک زدن نورون قبلی و زمان شروع به افزایش پتانسیل نورون مورد نظر است. شکل ۵ نشان‌گر این تأخیر است. در نهایت برای تکمیل مدل، کافی است اثر ترمینال‌های سیناپسی مختلف را در شکل‌دهی متغیر حالت در نظر بگیریم. معادله ۱۲ شکل کلی این مدل‌سازی را نشان می‌دهد.

$$x_j(t) = \sum_{i \in \Gamma_j} \sum_{k=1}^m w_{ij}^k y_i^k(t) \quad (12)$$

همچنین همان طور که پیش‌تر اشاره شد، شرط اسپایک زدن هر نورون آن است که مقدار متغیر حالت آن، از یک مقدار آستانه بیشتر شود؛ یعنی داشته باشیم $x_j(t) \geq v$ که مقدار v برای تمامی نورون‌ها ثابت و مشترک است.



شکل ۵: تأخیر زمانی ترمینال‌های سیناپسی

۳.۳ backpropagation

هدف از عملیات backpropagation دستیابی به مجموعه‌ای از «زمان‌های اسپایک زدن» است. برای دستیابی به این هدف، وزن‌ها (w_{ij}^k) را در هر مرحله تغییر می‌دهند.

- یکی از چالش‌هایی که در فرایند backpropagation و در این مدل به آن برمی‌خورند، آن است که ممکن است تغییرات ایجاد شده در وزن‌های مربوط به نورون به گونه‌ای به وقوع بپیوندند که پتانسیل غشاء نورون دیگر هیچ‌گاه به حد آستانه نرسد، و دیگر هیچ اسپایکی نزدند. در این صورت تخمین‌های مورد استفاده (تخمین مربوط به متغیر حالت x_j و حد آستانه) دیگر معتبر نخواهند بود. روش پیشنهادی برای حل این مشکل آن است که ورودی‌ها به گونه‌ای در شبکه کد شوند که اسپایک‌های زود هنگام به صورت اتوماتیک اهمیت بیشتری نسبت به اسپایک‌های دیر هنگام پیدا کنند.
- چالش دیگری که در این فرایند به وجود می‌آید آن است که اگر یک نورون برای هیچ الگوی ورودی اسپایک نزند، دیگر مکانیزمی برای تنظیم وزن‌های آن نورون وجود ندارد. روشی که در این آزمایش برای حل این مشکل در نظر گرفته‌اند، آن است که وزن‌های ابتدایی هر نورون را به گونه‌ای تنظیم کرده‌اند تا حداقل به بخشی از الگوهای ورودی پاسخ بدهد. با این کار، دیگر مشکلی در ارتباط با نورون‌های خاموش مشاهده نشد.

۴.۳ کدگذاری متغیرها

برای کدگذاری متغیرها با استفاده از روش زمانی (temporal coding) دو روش موجود است: اول استفاده از اختلاف زمانی‌های بسیار کم و رزولوشن زمانی بسیار بالا؛ و دوم استفاده از تعداد زیادی نورون به جای بالا بردن بیش از اندازه دقت زمانی.

با توجه به محدودیت‌های زیستی در مورد بالا بردن دقت زمانی و تلاش سازندگان مدل برای تعهد به این محدودیت‌ها، روش دوم را برای کد کردن ورودی انتخاب کرده‌اند؛ روش استفاده از جمعیت نورون‌ها.

برای کدگذاری در این روش از مفهوم receptive field (RF) استفاده می‌شود. منظور از receptive field هر نورون، الگویی از ورودی است که نورون مربوطه به آن حساس است و در صورت مشاهده‌ی چنین الگویی در ورودی اسپایک می‌زند. بدیهی است نورون مد نظر در اثر ورودی‌های دیگر اسپایک نمی‌زند (یا به تعبیر دقیق‌تر، نرخ اسپایک زدن کمتری دارد). برای کدگذاری زمانی ورودی، نورون‌هایی که (با توجه به receptive field خود) در اثر ورودی بیشتر برانگیخته می‌شوند، در زمان‌های زودتر، و نورون‌هایی که کمتر برانگیخته می‌شوند، در زمان‌های دیرتر اسپایک می‌زنند (یا به کلی اسپایک نمی‌زنند).

نحوه‌ی کدگذاری خروجی در حالت طبقه‌بندی نیز از winner-take-all paradigm پیروی می‌کند؛ به این معنی که نورونی که نشان‌گر کلاس انتخابی طبقه‌بندی است زودتر اسپایک می‌زند، و تمامی نورون‌های مربوط به کلاس‌های دیگر با اختلاف زمانی معناداری دیرتر شروع به اسپایک زدن می‌کنند.