دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیوتر

پروژه تحقیقاتی درس شبکه‌های عصبی

ماشین تورینگ عصبی

نگارش

علیرضا مازوچی

استاد درس

دکتر رضا صفابخش

تیر ۱۴۰۱


بسم الله الرحمن الرحيم

چکیده

ماشین تورینگ عصبی یکی از انواع جدید شبکه‌های عصبی است که از یک حافظه خارجی در کنار سایر اجزای یک شبکه عصبی معمولی استفاده می‌کند. محتوای این حافظه در هنگام آموزش تغییر پیدا می‌کند و شبکه عصبی راهی برای ارتباط صحیح با حافظه یاد می‌گیرد. در این پروژه به بررسی ماشین تورینگ عصبی و افزونه‌های آن می‌پردازیم. نهایتا کاربردهای واقعی آن به همراه برخی از بهترین نتایج آن که در تحقیقات علمی حاصل شده است مورد بررسی قرار خواهد گرفت.

واژه‌های کلیدی:

ماشین تورینگ عصبی، شبکه‌های عصبی بازگشتی، ماشین تورینگ، مکانیسم توجه، یادگیری ترتیبی

فهرست مطالب

عنوان صفحه

فهرست اشکال

شکل صفحه

فهرست جداول

جدول صفحه

فهرست اختصارات

عنوان اختصاری	عنوان کامل
ماتع	ماشین تورینگ عصبی

فصل اول

مقدمه

امروزه شبکه‌های عصبی[1] به شاخه اصلی مدل‌های هوش‌مصنوعی تبدیل شده‌اند و تقریبا در تمام کاربردهای ممکن جای خود را باز کرده‌اند. در برخی از کاربردها استفاده از اطلاعات لحظات پیشین یا ورودی‌های قبلی یک دنباله از ورودی الزامی به نظر می‌رسد. برای این کاربردها باید ساز و کاری در دل شبکه عصبی طراحی شده باشد که به هر طریق ممکن اطلاعات پیشین را خود نگه دارد. شبکه‌های عصبی بازگشتی[2] چنین قابلیتی را دارند.

شبکه‌های عصبی بازگشتی دروازه‌دار[3] نظیر LSTM اگرچه قادر بودند به واسطه یک بردار داخلی اطلاعات پیشین را حفظ کنند ولی این حافظه محدود است و زمانی‌که نیاز به نگهداری اطلاعات با حجم بالا به وجود بیاید ناتوانی این دست از شبکه‌ها عیان می‌شود. در این موارد وجود یک حافظه خارجی می‌تواند راهگشا باشد.

در سال‌های جدید دو معماری شبکه عصبی برای حل این مشکل ارائه شده است که از یک حافظه در کنار شبکه عصبی بهره گرفتند:

۱. شبکه حافظه‌ای[4]: شبکه حافظه‌ای به صورت صریح تمام اطلاعات یا حقایق را که در هر دوره وجود دارد را در یک حافظه خارجی ذخیره می‌کنند و از مکانیسم مبتنی بر توجه زمانی که قصد شاخص‌گذاری آن‌ها در زمان محاسبه یک خروجی را دارند استفاده می‌کنند.

۲. ماشین تورینگ عصبی[5]: ماشین تورینگ عصبی که در ادامه آن را به اختصار ماتع می‌نامیم هر حقیقت را در یک دوره می‌خواند و تصمیم می‌گیرد که آیا آن را در حافظه خارجی قابل تمایز بنویسد، بخواند و یا هر دو کار را انجام دهد یا نه.[۳]

تفاوت مهم بین دو مدل این است که شبکه‌های حافظه‌ای مکانیسمی برای تغییر محتوای حافظه خارجی را ندارد درحالی که ماتع‌ها این قابلیت را دارند. در عمل این مسئله منجر به یادگیری ساده‌تر برای وظایف واقعی در شبکه حافظه‌ای می‌شود. در مقابل، ماتع در ابتدای معرفی عمدتا بر روی یک سری از وظایف ساختگی با مقیاس کوچک نظیر رونوشت‌گیری[6] و یادآوری انجمنی[7] ارزیابی شده است. هرچند ماتع بیان دقیق‌تری دارد؛ چراکه می‌تواند وضعیت داخلی شبکه و همچنین فرآیندهای یک دوره را ذخیره کند یا تغییر دهد و ما را قادر خواهد ساخت که از آن بدون هیچگونه تغییری بر مدل برای وظایف مختلف استفاده کنیم.[۳]

[1] Neural Network
[2] Recurrent Neural Network (RNN)
[3] Gated Recurrent Neural Network (Gated RNN)
[4] Memory Network
[5] Neural Turing Machine
[6] Copy
[7] Association Recall

ماتع در سال ۲۰۱۴ توسط گروهی از محققین گوگل به سرپرستی گریوز[8] ارائه شد [؟]. تعداد ارجاعات به این مقاله تا بدین لحظه از دو هزار گذشته است. باتوجه به قدرت بالقوه بالایی که یک ماتع دارد در این پروژه قصد داریم آن را مورد بررسی قرار دهیم. طبیعتا اگر یک شبکه عصبی بتواند به درستی با یک حافظه بزرگ تعامل برقرار کند، پیشرفت قابل ملاحظه‌ای در حوزه شبکه‌های عصبی رخ خواهد داد.

سوال مهمی که باید به آن پاسخ داده شود آن است که آیا ماتع‌ها توانسته‌اند از قدرت بالقوه خود بهره بگیرند یا نه. پیش از پاسخ به این سوال باید مجددا یادآوری کنم که ماتع در مقاله اولیه خود برای حل چندین وظیفه ساختگی ساده مورد ارزیابی قرار گرفته است؛ به عنوان مثال در وظیفه رونوشت‌گیری به طور کلی هدف آن است که ورودی مستقیما در حافظه نوشته شود و در انتها از حافظه خوانده شود و به عنوان خروجی برگردانده شود. ناگفته پیداست که کاربردهای واقعی تا این میزان سرراست نیستند و جواب مناسب بر روی وظیفه‌هایی از این دست نمی‌تواند نشان از موفقیت ماتع در مسائل واقعی باشد.

مسئله تنها به نتیجه خروجی ختم نمی‌شود. پیچیدگی‌های طبیعی این شبکه مانع مهمی برای استفاده و توسعه آن است. توسعه‌دهندگان ماتع پیاده‌سازی و جزئیات کافی برای پیاده‌سازی خود را ارائه ندادند[۱]. این مسئله قطعا تاثیر منفی‌ای برای پیشرفت این شبکه به نسبت پیچیده بود. پیاده‌سازی‌های منبع‌باز[9] اولیه آن یا سرعت پایینی داشتند و یا ممکن بود وزن‌های آن به بی‌نهایت میل کند و آموزش دچار مشکل شود. نهایتا در سال ۲۰۱۸ یعنی چهار سال بعد از معرفی ماتع کولیر[10] و بیل [11] با یک پیاده‌سازی مناسب و منبع‌باز توانستند نتایج مقاله اصلی را در زمان مناسب تکرار کنند.[۱]. به گفته آنان مقداردهی اولیه[12] حافظه نقش مهم در رسیدن به پیاده‌سازی مناسب آن‌ها داشته است.[۱].

با تفاسیر بیان‌شده شاید کاربردی بودن این شبکه مورد تردید باشد. اما باید گفت که افزونه‌های ماتع[۳][۵] در سال‌های بعد بهبودهای مهمی بر روی نسخه اولیه اعمال کردند. به علاوه مدل‌های زیادی بر پایه ماتع برای مسائل واقعی و مجموعه‌داده‌های متعارف استفاده شده است که نتایج خوبی را رقم زده است. در این گزارش پس از بررسی ماتع سنتی افزونه‌های آن معرفی می‌شود و با ارائه کاربرهای واقعی و بخشی از نتایج مرتبط به کاربرهای واقعی نشان خواهیم داد که ماتع شبکه‌ای خلاقانه و در عین حال کاربردی است. ادامه ساختار این پروژه به شرح زیر است:

- در بخش دوم ساختار یک ماتع و نحوه آموزش آن تشریح خواهد شد. وظایف ساختگی و اولیه‌ای که یک ماتع برای حل آن مناسب است بیان می‌شود.
- در بخش سوم چند تا از افزونه‌های ماتع بررسی می‌شود. باتوجه به محدودیت‌های این پروژه تنها

[8] Graves
[9] Open Source
[10] Collier
[11] Beel
[12] Initialization

شماری از این افزونه‌ها معرفی می‌شود. در این بخش ساختار و روال آموزش هر افزونه شرح داده خواهد شد.

- در بخش چهارم کاربردهای واقعی که ماتع و افزونه‌های آن توانسته‌اند در آن استفاده شوند ارائه می‌شود. برای برخی از کاربردها ماتع و افزونه‌های آن تغییراتی داشته است؛ در این موارد تغییرات اساسی تبیین می‌شود.
- در بخش پنجم بخشی از برترین نتایج گزارش‌شده در مقالات که مدلی بر پایه ماتع به دقت‌های خوبی رسیده است گلچین شده است.
- نهایتا بخش ششم مروری بر مطالب این پروژه خواهد بود.

فصل دوم

ماشین تورینگ عصبی

۲-۱ ساختار و محاسبات ماشین تورینگ عصبی

ماتع شامل دو جز است:

۱. کنترل‌گر شبکه که می‌تواند یک شبکه‌عصبی جلورو یا یک شبکه عصبی بازگشتی باشد.

۲. یک واحد حافظه خارجی که یک ماتریس حافظه $N * W$ است. N تعداد واحدهای حافظه و W ابعاد هر سلول حافظه را نمایش می‌دهد.[۱]

فارغ از آنکه کنترل‌گر بازگشتی باشد یا خیر، کل معماری بازگشتی محسوب می‌شود چراکه ماتریس حافظه در طول زمان نگهداری می‌شود. کنترل‌گر سرهای خوانده و نوشتن دارد که به ماتریس حافظه دسترسی دارد. تاثیر یک عمل خواندن یا نوشتن روی یک سلول حافظه خاص با مکانیسم توجه نرم[۱] وزن‌دهی می‌شود. این مکانیسم آدرس‌دهی مشابه مکانیسم توجه استفاده‌شده در یادگیری ماشین عصبی است به جز آنکه آدرس‌دهی وابسته به موقعیت را با آدرس‌دهی وابسته به محتوای موجود در مکانیسم توجه را ترکیب می‌کند.[۱]

به طور خاص برای یک ماتع در هر گام زمانی t برای هر سر خواندن و نوشتن کنترل‌گر یک تعدادی پارامتر را به عنوان خروجی می‌دهد. این پارامترها برای محاسبه وزن w_t بر روی N خانه حافظه در ماتریس حافظه M_t استفاده می‌شوند. نحوه محاسبه w_t در رابطه ۲-۱ آورده شده است.[۱]

$$w_t^c(i) \leftarrow \frac{exp(\beta_t K[k_t, M_t(i)])}{\sum_{j=0}^{N-1} exp(\beta_t K[k_t, M_t(j)])} \tag{۲-۱}$$

در رابطه ۲-۱ برخی از پارامترها دارای محدودیت‌هایی هستند:[۱]

- $\beta_t \leq 0$
- $g_t \in [0, 1]$
- $\sum_k s_t(k) = 1$
- $\forall_k s_t(k) \leq 0$
- $\gamma_t \leq 1$

در رابطه ۲-۱ w_t^c آدرس‌دهی وابسته به محتوا را فراهم می‌کند. k_t یک کلید جستجو در حافظه را نشان می‌دهد و K مطابق رابطه ۲-۲ یک معیار شباهت مانند شباهت کسینوسی است.[۱]

[1] Soft Attention

$$K[u,v] = \frac{uv}{||u||.||v||} \tag{۲-۲}$$

با یک سری از محاسبات مطابق روابط ۲-۳، ۲-۴ و ۲-۵ ماتع‌ها امکان تکرار بر روی وزن‌های حافظه فعلی و قبلا محاسبه‌شده را خواهند داشت. رابطه ۲-۳ به شبکه اجازه می‌دهد تا بین بردار وزن قبلی یا فعلی انتخاب کند که از کدام استفاده کند. رابطه ۲-۴ امکان تکرار از طریق حافظه با عمل کانولوشن وزن فعلی و یک کرنل کانوولوشنی جابجایی[۲] یک بعدی را فراهم می‌کند. رابطه ۲-۵ رخداد تارشدن[۳] که به واسطه عمل کانوولوشن رخداده است را اصلاح می‌کند.[۱]

$$w_t^g \leftarrow g_t w_t^c + (1g_t) w_{t1} \tag{۳-۲}$$

$$\tilde{w}_t(i) \leftarrow \sum_{j=0}^{N-1} w_t^g(j) s_t(ij) \tag{۴-۲}$$

$$w_t(i) \leftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_{j=0}^{N-1} \tilde{w}_t(j)^{\gamma_t}} \tag{۵-۲}$$

سپس بردار r_t مطابق رابطه ۲-۶ به وسیله‌ی یک سر[۴] خواندن خاص در زمان t محاسبه می‌گردد.[۱]

$$r_t \leftarrow \sum_{i=0}^{N-1} w_t(i) M_t(i) \tag{۶-۲}$$

نهایتا مطابق رابطه ۲-۷ و ۲-۸ هر سر نوشتن ماتریس حافظه را در گام t با محاسبه بردارهای جانبی پاک‌کردن یعنی e_t و جمع‌کردن یعنی a_t تغییر می‌دهد.[۱]

[2] Shift

[3] Blurring

[4] Head

$$\tilde{M}_t(i) \leftarrow M_{t-1}(i)[1 - w_t(i)e_t] \tag{۲-۷}$$

$$M_t(i) \leftarrow \tilde{M}_t(i) + w_t(i)a_t \tag{۲-۸}$$

۲-۲ وظایف یادگیری ترتیبی

برای ماتع‌ها چندین وظیفه مصنوعی در نظر گرفته شده است که تمام آن‌ها از نوع مسئله یادگیری ترتیبی[۵] است؛ زمینه‌ای که آن‌ها در آن توانمند هستند. این وظایف عبارت اند از:

- رونوشت‌گیری[۶]: برای وظیفه رونوشت‌گیری یک دنباله تصادفی از بردارهای بیت با یک نماد خاص به عنوان پایان دنباله به شبکه داده می‌شود. این وظیفه نیاز دارد تا شبکه دنباله ورودی را نگه دارد و سپس آن را از حافظه برگرداند.

- رونوشت‌گیری تکرارشونده[۷]: مشابه وظیفه رونوشت‌گیری دنباله‌ای از بردارهای بیتی تصادفی به شبکه داده می‌شود. برخلاف وظیفه رونوشت‌گیری بعد از دنباله یک عدد که نشان دهنده تعداد دفعاتی است که باید دنباله در خروجی ظاهر شود به شبکه داده می‌شود.

- یادآوری انجمنی[۸]: این وظیفه نیز یک مسئله یادگیری دنباله با دنباله‌های متشکل از بردارهای بیتی تصادفی است. در این مورد ورودی به چندین عنصر تقسیم می‌شود که هر کدام شامل بردارهای ۳×۶ بعدی است. بعد از آنکه یک دنباله از آیتم‌ها و نماد پایانی دنباله به شبکه داده می‌شود. خروجی صحیح عنصر بعدی دنباله ورودی بعد عنصر کوئری است.[۱]

[5] Sequence Learning
[6] Copy
[7] Repeat Copy
[8] Associative Recall

فصل سوم

افزونه‌ها

۳-۱ ماشین تورینگ عصبی تکاملی

تکامل عصبی توپولوژی‌های تقویت‌کننده[۱] که در ادامه آن را به اختصار تکتوت می‌نامیم با یک جمعیت[۲] از شبکه‌های عصبی ساده شروع می‌کند و سپس آن‌ها را در طی نسل‌ها[۳] با افزودن رئوس جدید و اتصالات به کمک جهش[۴] پیچیده‌تر می‌کند. با تکامل شبکه‌ها از این را لازم نیست توپولوژی شبکه‌ها از پیش دانسته شده باشد. تکتوت به طرز فزاینده‌ای در شبکه‌های پیچیده جستجو می‌کند تا یک سطح مناسب از پیچیدگی را پیدا کند. ویژگی مهم تکتوت این است که هم توپولوژی و هم وزن‌های یک شبکه را تکامل می‌دهد. چراکه به سادگی و تدریجی پیچیدگی را افزایش می‌دهد و این باعث می‌شود که یک شبکه مناسب با اندازه مینیمال حاصل شود.[۵]

بر پایه ماتع و با استفاده از تکتوت مدل ماشین تورینگ عصبی تکاملی[۵] که در ادامه آن را به طور اختصار ماتع تکاملی می‌نامیم معرفی شده است. در این روش توپولوژی و وزن‌های شبکه عصبی کنترل‌گر با کمک تکتوت یاد گرفته می‌شود. بنابراین برخلاف ماتع استاندارد نیاز به دانش پیشین[۶] نیست و شبکه می‌تواند باتوجه به پیچیدگی وظیفه رشد پیدا کند. ماتع تکاملی اغلب توپولوژی‌های فشرده برای حل یک وظیفه خاص پیدا می‌کند؛ در نتیجه جلوی جستجوی غیرضروری در فضای با ابعاد بالا گرفته می‌شود. به علاوه ماتع تکاملی قادر به جل مسائل یادگیری مستمر پیچیده است. چراکه شبکه از مشتق استفاده نمی‌کند و می‌تواند از توجه سخت[۷] و مکانیسم جابجایی استفاده کند که امکان تعمیم خوب برای دنباله‌های بلند در وظیفه رونوشت‌گیری را فراهم می‌کند. به علاوه یک نوار[۸] پویا و از نظر تئوری با اندازه بی‌نهایت قابل استفاده است.[۵]

ماتع تکاملی یک سر تکی ترکیبی خواندن/نوشتن دارد. این شبکه بردار نوشتن w با اندازه M، ورودی کنترل درون‌یابی[۹] نوشتن i، ورودی کنترل پرش محتوا j و سه ورودی کنترل جابجایی s_l (جابجایی چپ)، s_0 (بدون جابجایی) و s_r (جابجایی راست) را خروجی می‌دهد. اندازه بردار نوشتن M اندازه هر خانه حافظه بر روی نوار را مشخص می‌کند. جزء درون‌یابی نوشتن امکان مخلوط‌کردن مقادیر فعلی نوار و بردار نوشتن در موقعیت نوشتن را فراهم می‌کند. $M_h(t)$ محتوای نوار در موقعیت سر فعلی h در زمان t، i_t درون‌یابی نوشتن و w_t بردار نوشتن است. برای تمام این‌ها در زمان t رابطه ۳-۱ را خواهیم داشت.[۵]

[1] Neuroevolution of Augmenting Topologies(NEAT)
[2] Population
[3] Generations
[4] Mutation
[5] Evolvable Neural Turing Machine
[6] Prior Knowledge
[7] Hard Attention
[8] Tape
[9] Interpolation

$$M_h(t) = M_h(t1)(1i_t) + w_t i_t \quad (۱-۳)$$

پرش محتوا مشخص می‌کند که آیا سر باید به موقعیتی در حافظه حرکت کند که بیشترین شباهت را به بردار نوشتن دارد یا نه. یک پرش محتوا انجام می‌شود اگر مقدار ورودی کنترل از ۵.۰ بیشتر شود. شباهت بین بردار نوشتن w و بردار حافظه m مطابق با رابطه ۳-۲ حساب می‌شود.[۵]

$$s(w, m) = \frac{\sum_{i=1}^{M} |w_i m_i|}{M} \quad (۲-۳)$$

در گام زمانی t اقدامات زیر به ترتیب انجام می‌شود:

۱. بردار نوشتن w_t برای موقعیت فعلی h بدست می‌آید. این بردار با محتوای موجود باتوجه به درون‌یابی نوشتن i_t درون‌یابی می‌شود.

۲. اگر ورودی کنترل پرش محتوا j_t بزرگ‌تر از ۵.۰ شود، سر به مکانی در نوار که بیشترین شباهت به بردار نوشتن w_t دارد حرکت می‌کند.

۳. سر به یک موقعیت چپ‌تر، راست‌تر روی نوار حرکت می‌کند یا در همان جا ثابت می‌ماند که این وابسته به مقادیر ورودی کنترل جابجایی s_l، s_0 و s_r است.

۴. مقادیر نوار را در موقعیت جدید سر می‌خواند و بر می‌گرداند.[۵]

۳-۲ ماشین تورینگ عصبی ابر تکاملی

در کدگذاری‌های مستقیم مانند تکتوت هر بخش از نمایش جواب به یک تکه کوچک از ساختار نهایی جواب نگاشت می‌شود. عیب مهم این روش آن است که بخش‌های مختلف راه‌حل که به یک‌دیگر شبیه هستند نیز باید کد شوند و جداگانه کشف شوند. این ابراد با کدگذاری غیرمستقیم تا حد زیادی قابل حل است؛ در کدگذاری غیرمستقیم راه‌حل به شکل فشرده توصیف می‌شود و حجم اطلاعات کدشده می‌تواند کاهش بیابد. در کدگذاری غیرمستقیم به دلیل آنکه یک راه‌حل به شکل الگویی از پارامترها و نه تمام پارامترها نمایش پیدا می‌کند قدرت‌مند است.[۵]

روش ابر تکامل عصبی توپولوژی‌های تقویت‌کننده[۱۰] که در ادامه به اختصار آن را ابرتوت می‌نامیم یک افزونه از تکتوت است. در این افزونه به جای کدگذاری غیرمستقیم از کدگذاری مستقیم استفاده

[10] Hyper Neuroevolution of Augmenting Topologies(HyperNEAT)

می‌شود. در تکتوت از شبکه‌های عصبی معمولی استفاده می‌شود درحالی که در ابرتوت از شبکه‌های تولید الگوی ترکیبی[11] که در ادامه آن را به اختصار شبکه تات می‌نامیم استفاده شده است. شبکه تات برای کدگذاری ترکیب توابع طراحی شده‌اند که هر تابع در ترکیب مرتبط با یک منظم‌سازی[12] است.[۵]

حسن شبکه تات آن است که به الگوهای مکانی اجازه می‌دهد که به عنوان شبکه‌هایی از توابع ساده نمایش پیدا کنند. این یعنی تکتوت می‌تواند با شبکه تات مانند شبکه‌های عصبی تکامل پیدا کند. شبکه‌های تات مشابه شبکه‌های عصبی هستند با این تفاوت که آن‌ها متکی بر بیشتر از یک تابع فعال‌سازی هستند. کدگذاری غیرمستقیم شبکه تات می‌تواند به طور فشرده الگوها با نظم‌هایی نظیر تقارن[13]، تکرار[14] و تکرار با تغییر[15] را کد کنند. به عنوان مثال با انتخاب یک تابع گاوسین که خاصیت تقارن دارد الگوی خروجی نیز به سادگی متقارن خواهد شد. انتخاب یک تابع دوره‌ای مانند سینوس در حین تکرار قطعه‌سازی انجام می‌دهد. نهایتا تکرار با تغییر به سادگی با ترکیب یک تابع منظم (مانند سینوس یا گاوسین) با یک تابع نامنظم (مانند محور x نامتقارن) بدست می‌آید.[۵]

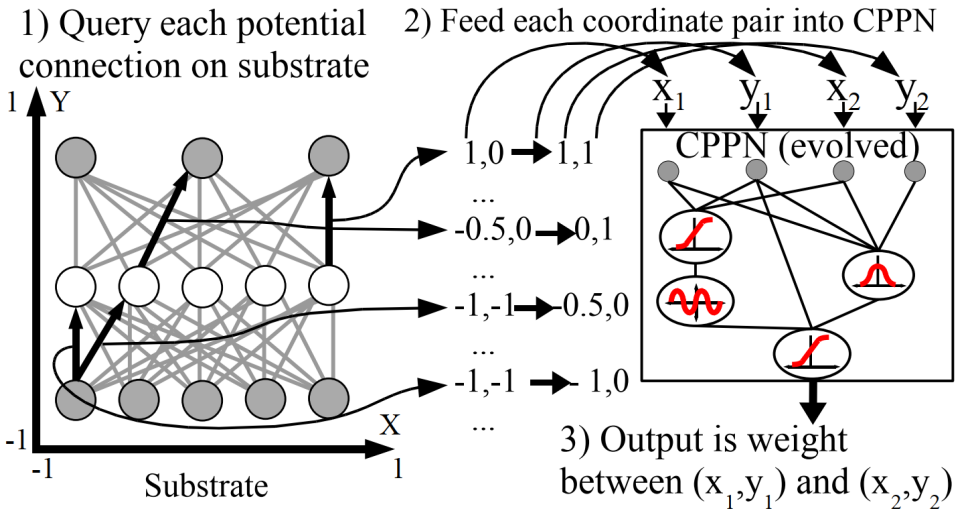
ایده اصلی ابرتوت آن است که شبکه‌های تات می‌تواند اتصال الگوها را کدکند. بدین طریق یک تکتوت می‌تواند یک شبکه تات که شبکه‌های عصبی بزرگ با منظم‌سازی‌ها و تقارن‌های خود نمایش می‌دهد را تکامل دهد.[۵]

عملکرد شبکه تات در تصویر ۳-۱ آورده شده است. شبکه‌های تات سنتی توابع هندسی هستند که الگوهای اتصال خروجی آن رئوسی در n بعد است که n تعداد ابعاد در فضای کارتزین است. یک شبکه تات که چهار ورودی با برچسب‌های x_1، y_1، x_2 و y_2 را دریافت کند و به عنوان خروجی مشخص می‌کند که اتصال بین نقاط دوبعدی (x_1, y_1) و (x_2, y_2) چه میزان است. بنابراین با داشتن یک شبکه تات آموزش یافته می‌توان با ارسال یک کوئری شامل هر دو راس در شبکه عصبی مقدار اتصال آن را بدست آورد و شبکه عصبی را ایجاد کرد.[۵]

همانطور که در توضیحات قبل بدان اشاره شد باید تمام گره‌های شبکه عصبی در یک بستر[16] قرار داده شوند. یعنی مشخص شود که هر گره در شبکه عصبی در چه مختصاتی در فضا باید قرار بگیرد. مشابه چیزی که در سمت چپ شکل ۳-۱ قابل مشاهده است. ایده اصلی در ماتع ابرتکاملی پیشنهاد یک بستر برای بهره‌مندی از شبکه تات در ماتع تکاملی بوده است.

در مدل ماتع ابرتکاملی شبکه تات نه تنها اتصال ورودی‌ها و خروجی‌های شبکه عصبی مرتبط با

[11] Compositional Pattern Producing Networks (CPPN)
[12] Regularity
[13] Symmetry
[14] Repetition
[15] Repetition with Variation
[16] Substrate



شکل ۳-۱: نحوه کارکرد شبکه تات[۵]

مجموعه‌ای از گره‌ها به مختصات بین -۱ تا +۱ در تمام ابعاد نظیر می‌شوند. (۱): هر اتصال ممکن در یک شبکه عصبی کوئری زده می‌شود تا مجاورت و وزن آن مشخص شود. خطوط جهت‌دار تیره نمایش‌داده شده در تصویر یک نمونه از اتصالاتی است که کوئری زده شده است. (۲): در درون یک شبکه تات یک گراف است که مشخص می‌کند کدام توابع فعال‌سازی به یکدیگر متصل هستند. همان‌طور که در شبکه عصبی اتصالات وزن‌دهی می‌شوند که خروجی یک تابع با چه وزنی به طرف دیگر اتصال برود، برای هر کوئری ارسال‌شده به شبکه تات جایگاه دو سر اتصال را به عنوان ورودی می‌گیرد و وزن اتصال را به عنوان خروجی می‌دهد. (۳): بنابراین شبکه تات می‌تواند الگوهای منظم از اتصالات در فضا را تولید کند.

وظیفه را مشخص می‌کند بلکه اینکه اطلاعات آمده از حافظه چگونه باید در شبکه ادغام شوند و چگونه اطلاعات در حافظه نوشته شوند را هم مشخص می‌کند. زیرا ابرتوت که می‌تواند هندسه یک وظیفه را یاد بگیرد قائدتا باید بتواند الگوی هندسی اطلاعات خوانده شده از و نوشته شده در حافظه را نیز یاد بگیرد.[۵]

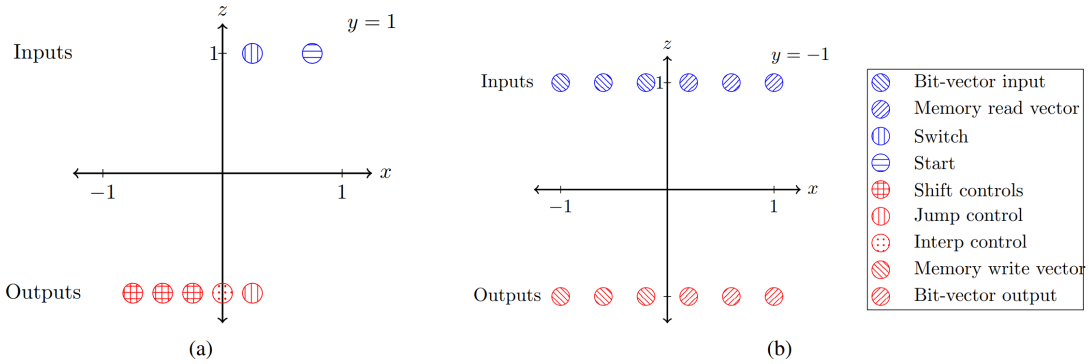
شبکه ماتع تکاملی ورودی‌های زیر را دارد:

- شروع: ورودی که هرگاه فعال می‌شود، ذخیره اعداد شروع می‌شود.
- تعویض: ورودی که هرگاه فعال شود، ذخیره اطلاعات خاتمه می‌یابد و شبکه باید مقادیر به خاطر سپرده‌شده را به یاد بیاورد.
- ورودی بردار بیتی: بردار بیتی که شبکه به عنوان ورودی می‌گیرد. توجه کنید قبل از آنکه ورودی تعویض فعال شود رنج این ورودی با بیت‌هایی که بعدا می‌خواند فعال می‌شود.
- ورودی خواندن حافظه: بردار حافظه که ماشین تورینگ در گام قبل خوانده است.[۵]

این شبکه خروجی‌های زیر را هم دارد:

- خروجی بردار بیتی: بردار بیتی که شبکه به عنوان خروجی می‌دهد. توجه کنید در حین دریافت ورودی این خروجی نادیده گرفته می‌شود.
- خروجی نوشتن حافظه: بردار حافظه‌ای که باید در حافظه نوشته شود.
- کنترل‌گرهای ماشین تورینگ: خروجی‌های کنترل مخصوص ماشین تورینگ یعنی پرش، درون‌یابی و سه کنترل جابجایی (چپ، راست و توقف)[۵]

در ادامه بستر طراحی‌شده برای وظیفه رونوشت‌گیری آورده می‌شود. این بستر در شکل ۳-۲ نشان داده شده است. این بستر طراحی شده است که گره‌های ورودی بردار بیتی مختصات x را با گره‌های نوشتن بردار حافظه به اشتراک بگذارد و بالعکس با گره‌های خواندن بردار حافظه و گره‌های خروجی بردار بیتی. به علاوه ورودی تعویض مختصات x اش را با خروجی پرش به اشتراک می‌گذارد بنابراین شبکه را می‌تواند وادار به پرش به حافظه‌ای کند که خواندن را از آن شروع کرده است. در این مقاله اندازه بردارهای حافظه برابر با اندازه بردار بیتی است. به علاوه هیچ یک از بسترها شامل گره‌های مخفی مانند آن چیزی که نشان داده شده است و ممکن است مسائل با اندازه‌های بزرگ‌تر بدون گره مخفی را حل کند نیست.[۵]



شکل ۳-۲: بستر طراحی‌شده برای شبکه تات در مدل ماتع ابرتکاملی[۵]

تمام ورودی‌ها در $z = 1$ و تمام خروجی‌ها در $z = -1$ هستند. قسمت الف تمام گره‌ها در $y = 1$ را نشان می‌دهد که ورودی‌های شروع، تعویض و کنترل‌گرهای ماشین تورینگ هستند. لازم به ذکر است که مختصات x برای ورودی تعویض و خروجی کنترل‌گر پرش یکسان است. قسمت ب گره‌ها را در $y = -1$ نشان می‌دهد که ورودی و خروجی‌های بردار حافظه و بردار بیتی را نشان می‌دهد. گره‌های ورودی بردار بیتی مختصات x را با گره‌های نوشتن بردار حافظه به اشتراک می‌گذارد، درحالی‌که گره‌های خواندن بردار حافظه مختصات x را با گره‌های خروجی بردار بیتی به اشتراک می‌گذارد.[۵]

در کنار خروجی شبکه تات که وزن هر اتصال را مشخص می‌کند، هر شبکه تات یک خروجی تابع قدم[۱۷] اضافه دارد که خروجی بیان پیوند نامیده می‌شود. این خروجی مشخص می‌کند که آیا یک اتصال

[17] Step Function

باید بیان شود یا خیر. اتصالات بالقوه برای هر ورودی در لایه‌های $y = 1$ و $y = -1$ به هر خروجی در لایه‌های $y = 1$ و $y = -1$ کوئری زده می‌شود. تعداد ورودی‌ها و خروجی‌ها در لایه $y = -1$ مطابق قسمت ب شکل ۳-۲ وابسته به اندازه بردار بینی وظیفه رونوشت‌گیری است، که در مثال نشان‌داده شده اندازه بردار بیتی برابر با ۳ است. نورون‌ها به شکل یکنواخت در بازه‌های -۱ تا -۲.۰ در مختصات x برای ورودی‌های بردار بیتی و بردار نوشتن حافظه استفاده می‌شود و بازه ۲.۰ تا ۱ برای بردار نوشتن حافظه و خروجی بردار بیتی استفاده می‌شود.[۵]

نهایتا برای تعیین میزان بایاس یک گره باید مختصات همان گره را هم به عنوان گره مبدا و هم به عنوان گره مقصد به شبکه تات کوئری زد.[۵]

۳-۳ ماشین تورینگ عصبی متوجه

در مدل ماتع متوجه حافظه با یک مکانیسم آدرس‌دهی وابسته به محتوا بازیابی می‌شود. آدرس‌دهی برپایه محتوا به صورت ضروری یک گام محاسبه شباهت میان بردار خروجی کنترل‌گر C_t و بردارهای حافظه موجود M_t است. وزن توجه در خواندن با رابطه ۳-۳ تولید می‌شود. در رابطه مذکور متغیر β می‌تواند دقت تمرکز را افزایش یا کاهش دهد و معیار شباهت استفاده‌شده معیار شباهت کسینوسی است.[۶]

$$w_t^r(i) = \frac{exp(\beta.sim(C_t, M_t(i)))}{\sum_j (exp(\beta.sim(C_t, M_t(j))))} \quad (۳-۳)$$

در حالت کلی کنترل‌گر می‌تواند با هر شبکه عصبی مصنوعی طراحی شود. در ماتع متوجه از یک لایه توجه چندسر[۱۸] استفاده کرده‌اند تا روابط بین دنباله‌های ورودی و خروجی را مدل کنند. [۶]

رابطه مربوط به لایه توجه در رابطه ۳-۴ آورده شده است. در این رابطه مطابق معمول K، Q و V به ترتیب ماتریس کوئری، کلید و مقدار است و n تعداد ابعاد است. توجه بر دنباله ورودی I اعمال می‌شود تا یک جمع وزن‌دار برای هر i_t بر پایه i_1 تا i_t اعمال شود که t زمان فعلی است.[۶]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{n}}) \quad (۴-۳)$$

به عنوان خروجی مدل یک بردار وزن مرتبط با هر تلاش را یاد می‌گیرد و بردار وزن یادگرفته‌شده

[18] Multi-Headed Attention

برای محاسبه پیش‌بینی خروجی i_t استفاده می‌شود. از تابع خطا میانگین مربعات خطا مطابق رابطه ۳-۵ استفاده می‌شود. در این رابطه a_t^k مقدار پیش‌بینی، gt_t^k مقدار واقعی برچسب، b اندازه دسته[۱۹]، k طول دنباله و t زمان است.[۶]

$$L = \sum_{b} \sum_{k=1}^{n} \sum_{t=1}^{T} MSE(a_t^k, gt_t^k) \tag{۳-۵}$$

در ماتع متوجه حافظه با مکانیسم آدرس‌دهی برپایه موقعیت بروز می‌شود. مکانیسم آدرس‌دهی برپایه موقعیت برای چندین گام طراحی شده است تا تکرار بر روی خانه‌های حافظه و پرش‌های دسترسی تصادفی را امکان‌پذیر کند. گام اول محاسبه درون‌یابی بین بردار وزن نوشتن پیشین w_{t-1} و بردار وزن محتوا تولیدشده توسط مکانیسم آدرس‌دهی محتوا w_t^c در گام زمانی فعلی با استفاده از دروازه[۲۰] درون‌یابی w_t^g مطابق رابطه ۳-۶ است.[۶]

$$w_t^g = g_t w_t^c + (1g_t) w_{t1} \tag{۳-۶}$$

بعد از درون‌یابی یک وزن‌دهی جابجایی s_t بر ماتریس وزن دروازه‌دار با یک کانولوشن دایره‌ای برای تنظیم حافظه مطابق رابطه ۳-۷ اعمال می‌شود. در این رابطه N برابر با اندازه حافظه است. نهایتا مطابق رابطه ۳-۸ عمل تیزکردن[۲۱] برای نرمال‌سازی استفاده می‌شود.[۶]

$$w_t^{ro} = \sum_{j=0}^{N-1} w_t^g(j) s_t(i-j) \tag{۳-۷}$$

$$w_t(i) = \frac{w_t^{ro}(i)}{\sum_j w_t^{ro}(j)} \tag{۳-۸}$$

[19] Batch
[20] Gate
[21] Sharpen

۳-۴ ماشین تورینگ عصبی پویا

ماتع سنتی دو مدل آدرس‌دهی را پشتیبانی می‌کند که می‌تواند به صورت همزمان استفاده شود: آدرس‌دهی بر پایه محتوا و آدرس‌دهی بر پایه موقعیت. استراتژی برپایه موقعیت خود بر پایه آدرس‌دهی خطی است که فاصله بین یک جفت از سلول حافظه متوالی همواره برابر با یک مقدار ثابت است. ماتع پویا این محدودیت را با معرفی یک بردار آدرس‌دهی قابل یادگیری برای هر سلول حافظه در ماتع که اخیرا به عنوان مکانیسم آدرس‌دهی حافظه استفاده شده است حل کرده است.[۳]

۳-۴-۱ معماری ماشین تورینگ عصبی پویا

ماتع پویا شامل یک حافظه خارجی M_t است که هر سلول حافظه i در $M_t[i]$ به دو بخش شکسته می‌شود: یک بردار آدرس آموزش‌پذیر $A_t[i]$ و بردار محتوا $C_t[i]$ است. به طور مشابه می‌توان دید که کل حافظه به یک ماتریس آدرس آموزش‌پذیر A_t و یک ماتریس محتوای C_t شکسته می‌شود. بنابراین برای هر سلول حافظه رابطه ۳-۹ و برای کل حافظه رابطه ۳-۱۰ برقرار خواهد بود.[۳]

$$M_t[i] = [A_t[i]; C_t[i]] \tag{۳-۹}$$

$$M_t = [A_t; C_t] \tag{۳-۱۰}$$

بخش آدرس A_t پارامتر مدل است که در طول یادگیری بروز می‌شود. در حین استنتاج بخش آدرس توسط کنترل‌گر تغییر پیدا نمیکند و ثابت می‌ماند. بخش محتوا C_t در حین آموزش و استنتاج توسط کنترل‌گر چه برای نوشتن و چه برای خواندن تغییر پیدا می‌کند. در ابتدای هر دوره بخش محتوای حافظه به یک ماتریس تمام صفر تغییر پیدا می‌کند. ($C_0 = 0$)این شروع باعث می‌شود که بخش قابل آموزش آدرس برای هر سلول حافظه به مدل امکان یادگیری استراتژی‌های آدرس‌دهی پیچیده مبتنی بر مکان را بدهد.[۳]

کنترل‌گر در هر گام زمانی t اقدامات زیر را انجام می‌دهد:

۱. یک مقدار ورودی x_t دریافت می‌کند.

۲. به حافظه دسترسی پیدا می‌کند و آن را می‌خواند و بردار محتوای r_t را ایجاد می‌کند.

۳. یک بخشی از اطلاعات را روی حافظه می‌نویسد.

۴. وضعیت تصادفی خود را بروز می‌کند.

۵. در صورت نیاز مقدار y_t را به عنوان خروجی می‌دهد.

در ماتع پویا هم امکان استفاده از کنترل‌گر جلورو و هم استفاده از کنترل‌گر GRU وجود دارد. نحوه محاسبه وضعیت مخفی با استفاده از این دو کنترل‌گر به ترتیب در رابطه ۳-۱۱ و ۳-۱۲ آمده است.[۳]

$$h_t = \sigma(x_t, r_t) \tag{۱۱-۳}$$

$$h_t = GRU(x_t, h_{t-1}, r_t) \tag{۱۲-۳}$$

در گام زمانی t کنترل‌گر x_t را به عنوان ورودی می‌گیرد و سپس وزن خواندن w_t^r تولید می‌شود. سپس بردار محتوا r_t مطابق رابطه ۳-۱۳ ایجاد می‌شود. نهایتا وضعیت مخفی کنترل‌گر h_t وابسته به بردار محتوای حافظه r_t و وضعیت مخفی پیشین کنترل‌گر h_{t-1} محاسبه می‌شود و مدل برچسب خروجی y_t برای ورودی را پیش‌بینی می‌کند.[۳]

$$r_t = M_t^T w_t^r \tag{۱۳-۳}$$

کنترل‌گر با پاک‌کردن محتوای قدیمی و نوشتن اطلاعات جدید حافظه را بروز می‌کند. کنترل‌گر سه بردار را محاسبه می‌کند: بردار پاک‌کردن e_t، وزن‌های نوشتن w_t^w و بردار محتوای نامزد $\bar{c}_t$. وزن‌های نوشتن w_t^w و خواندن w_t^r با سرهای مجزا محاسبه می‌شود و با شبکه پرسپترونی چندلایه [۲۲] پیاده‌سازی می‌شود. این بردارهای وزن برای تعامل با حافظه استفاده می‌شود. بردار پاک‌کردن نیز با یک شبکه پرسپترونی ساده محاسبه می‌شود که وابسته به وضعیت مخفی کنترل‌گر h_t است. بردار محتوای حافظه نامزد بر مبنای وضعیت مخفی فعلی h_t و ورودی کنترل‌گر که با یک دروازه عددی α_t مقیاس شده است خواهد بود. α_t یک تابع از وضعیت مخفی و ورودی کنترل‌گر است. در رابطه ۳-۱۴ و ۳-۱۵ به ترتیب نحوه محاسبه α_t و $\bar{c}_t$ آورده شده است. با داشتن بردارهای پاک‌کردن، نوشتن و محتوای حافظه نامزد ماترس محتوا مطابق با رابطه ۳-۱۶ قابل بروزرسانی است.[۳]

$$\alpha_t = f(h_t, x_t) \tag{۱۴-۳}$$

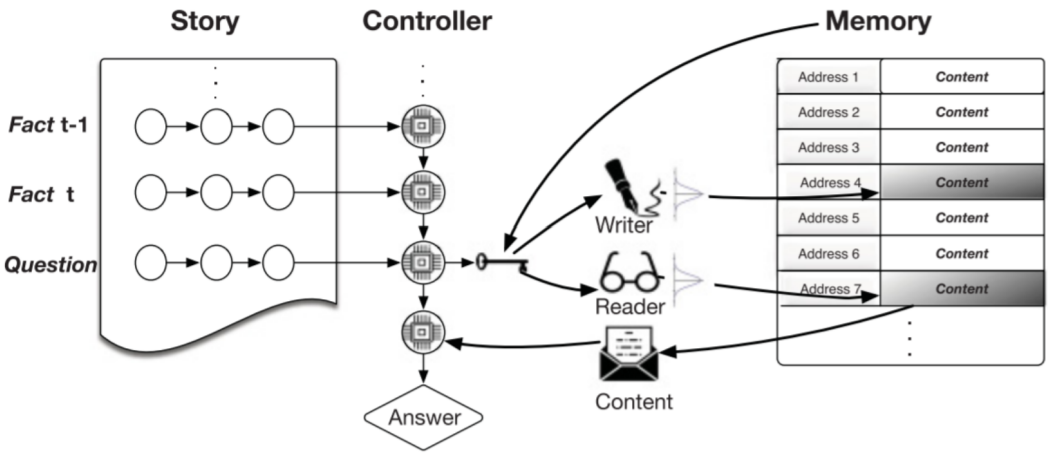
[22] Multi Layer Perceptran (MLP)

$$\bar{c}_t = ReLU(W_m h_t + \alpha_t W_x x_t) \quad (۱۵-۳)$$

$$C_t[j] = (1 - e_t w_t^w[j]) \odot C_{t-1}[j] + w_t^w[j]\bar{c}_t \quad (۱۶-۳)$$

یک عمل بی‌عملی برای کنترل‌گر که برای تنها یک مرتبه در یک زمان کاری نکند می‌تواند مفید باشد. این موقعیت با طراحی یک سلول حافظه به عنوان سلول بی‌عملی اضافی مدل شده است. کنترل‌گر زمانی که نیازی به نوشتن به یا خواندن از حافظه ندارد باید به این سلول دسترسی داشته باشد چراکه نوشتن و خواندن کاملا نادیده گرفته می‌شوند. بی‌عملی برای نوشتن در ماتع پویا معادل با یادگیری آن است که دروازه نوشتن در ماتع سنتی محتوای حافظه را بدون تغییر باقی بگذارد.[۳]

در تصویر ۳-۳ نمایش گرافیکی از ماتع پویا با کنترل‌گر بازشگتی نشان داده شده است.



شکل ۳-۳: معماری مدل ماتع پویا[۳]

کنترل‌گر یک حقیقت را به عنوان به عنوان بردار ورودی کدشده توسط شبکه عصبی بازگشتی دریافت می‌کند و وزن‌های خواندن و نوشتن برای دسترسی به حافظه را محاسبه می‌کند. اگر ماتع پویا به صورت خودکار شناسایی کند که یک کوئری دریافت شده است یک پاسخ بر می‌گرداند و کار را خاتمه می‌دهد.[۳]

۳-۴-۲ مکانیسم آدرس‌دهی

کنترل‌گر بردار کلید را مطابق با رابطه ۳-۱۷ بدست می‌آورد. سپس با کمک رابطه ۳-۱۸ وزن‌های لاجیتس[۲۳] آدرس‌دهی $z_t[i]$ حاصل می‌شود. در این رابطه S یک رابطه شباهت مانند شباهت کسینوسی(رابطه ۲-۲)است. β_t یک فاکتور تیزی[۲۴] است که نحوه محاسبه آن در رابطه ۳-۱۹ آمده است. تابع $softmax$ مطابق با رابطه ۳-۲۰ خواهد بود.[۳]

$$k_t = W_k^T h_t + b_k \tag{۳-۱۷}$$

$$z_t[i] = \beta^t S(k_t, M_t[i]) \tag{۳-۱۸}$$

$$\beta_t = softplus(u_\beta^T ht + b_\beta) + 1 \tag{۳-۱۹}$$

$$softplus(x) = log(exp(x) + 1) \tag{۳-۲۰}$$

میانگین وزن‌دار نمایی لاجیتس آدرس‌دهی v_t مطابق با رابطه ۳-۲۱ محاسبه می‌شود. سپس با کمک رابطه ۳-۲۲ و ۳-۲۳ وزن‌های آدرس‌دهی w_t حاصل می‌شود. با کمک این روابط وزن مربوط به سطرهایی از حافظه که اخیرا کمتر مورد استفاده قرار گرفته‌اند افزایش می‌یابد. تاثیر این رخداد با کمک γ_t تعیین می‌شود. این مکانیسم تعمیمی بر آدرس‌دهی برپایه محتوا سنتی است.[۳]

$$v_t = 0.1 * v_{t-1} + 0.9 * z_t \tag{۳-۲۱}$$

[23] Logits

[24] Sharpness Factor

$$\gamma_t = sigmoid(u_\gamma^T h_t + b_\gamma) \quad (۲۲-۳)$$

$$w_t = softmax(z_t - \gamma_t v_{t-1}) \quad (۲۳-۳)$$

هر سطر از ماتریس آدرس‌دهی w_t دارای مقدار مثبت است که جمع آن‌ها برابر با یک است. می‌توان بردار تک‌روشن[۲۵] $\tilde{w}_t$ را از روی آن ایجاد کرد. در زمان آموزش این بردار با رابطه ۳-۲۴ ایجاد می‌شود. این برنامه آدرس‌دهی گسسته[۲۶] و مدل هم ماتع پویای گسسته نامیده می‌شود.[۳]

$$\tilde{w}_t[k] = I(k = argmax(w_t)) \quad (۲۴-۳)$$

در انتهای این بخش باید تاکید کرد که در ماتع پویا و در هر گام زمانی کنترل‌گر می‌تواند چندین بیشتر از یک درخواست برای دسترسی به حافظه داشته باشد. این کار با یک گزینه اضافه انجام می‌شود. در ماتع سنتی با چندین سر این کار می‌توانست انجام شود.[۳]

۳-۴-۳ آموزش

تابع هزینه مورد استفاده در ماتع پویا مانند رابطه ۳-۲۵ است. ماتع پویا پیوسته[۲۷] می‌تواند مانند ماتع سنتی با انتشار به عقب [۲۸] آموزش یابد ولی در ماتع پویا گسسته به دلیل استراتژی نمونه‌برداری در زمان آموزش این امکان وجود نخواهد داشت.[۳]

$$C(\theta) = 0\frac{1}{N}\sum_{n=1}^{N} log\, p(y^{(n)}|x_1^{(n)}, ..., x_T^{(n)}; \theta) \quad (۲۵-۳)$$

برای ماتع پویا گسسته تابع سود مطابق رابطه ۳-۲۶ تعریف می‌شود و مطابق رابطه ۳-۲۷ و ۳-۲۸

[25] One-Hot
[26] Discrete
[27] Continues
[28] Backpropagation

نرمال می‌شود. در رابطه ۳-۲۸ $b(x)$ یک شبکه است که برای کاهش خطای هوبر[۲۹] آموزش می‌بینند. این خطا در رابطه ۳-۲۹ آورده شده است. مقدار z برابر با $\bar{R}(x)$ خواهد بود.[۳]

$$R(x) = log\, p(y^{(n)}|x_1^{(n)}, ..., x_T^{(n)}; \theta) \tag{۲۶-۳}$$

$$\tilde{R}(x) = \frac{R(x) - b}{\sqrt{\sigma^2 + \epsilon}} \tag{۲۷-۳}$$

$$\bar{R}(x) = \tilde{R}(x) - b(x) \tag{۲۸-۳}$$

$$H_\delta(z) = \begin{cases} z^2 & |z| \leq \delta \\ \delta(2|z| - \delta) & o.w. \end{cases} \tag{۲۹-۳}$$

تابع خطا برای حالت گسسته در رابطه ۳-۳۰ آورده شده است. در این رابطه $\mathcal{H}$ نمایانگر انتروپی[۳۰] است.

$$\begin{aligned} C^n(\theta) = &-log\, p(y|x_{1:T}, \tilde{w}^r_{1:J}, \tilde{w}^w_{1:J}) \\ &- \sum_{j=1}^{J} \bar{R}(x^n)(log\, p(\tilde{w}^r_j|x_{1:T}) + log\, p(\tilde{w}^w_j|x_{1:T})) \\ &- \lambda_H \sum_{j=1}^{J} \mathcal{H}(w^r_j|x_{1:T}) + \mathcal{H}(w^w_j|x_{1:T}) \end{aligned} \tag{۳۰-۳}$$

[29] Hubber Loss

[30] Entropy

فصل چهارم

کاربردهای واقعی

ماتع و افزونه‌هایش در سال‌های اخیر کاربردهای گوناگونی پیدا کرده‌اند. در این بخش قصد داریم به تعدادی از کاربردهای آن بپردازیم.

۴-۱ تخمین عمر مفید باقی‌مانده

فالکن[1] و همکاران[۲] در تحقیقی عمر مفید باقی‌مانده وسایل مکانیکی استفاده شده در حوزه درمان و بهداشت را بررسی کرده‌اند. تخمین عمر مفید یک وسیله مکانیکی یکی از مسائل مهم در حوزه مدیریت سلامت و پیشگیری است. توانایی تخمین قابل اطمینان بودن آن منجر به بهود در برنامه‌ریزی نگهداری و کاهش هزینه‌های مرتبط با آن می‌شود. در دسترس بودن سنسورهای با کیفیت بالا که چندین جنبه از اجزا را می‌سنجد این امکان را فراهم می‌کند که حجم زیادی از داده‌ها جمع‌آوری شود که این داده‌ها می‌تواند در تنظیم‌کردن مدل‌های برپایه داده[2] استفاده شود.[۲]

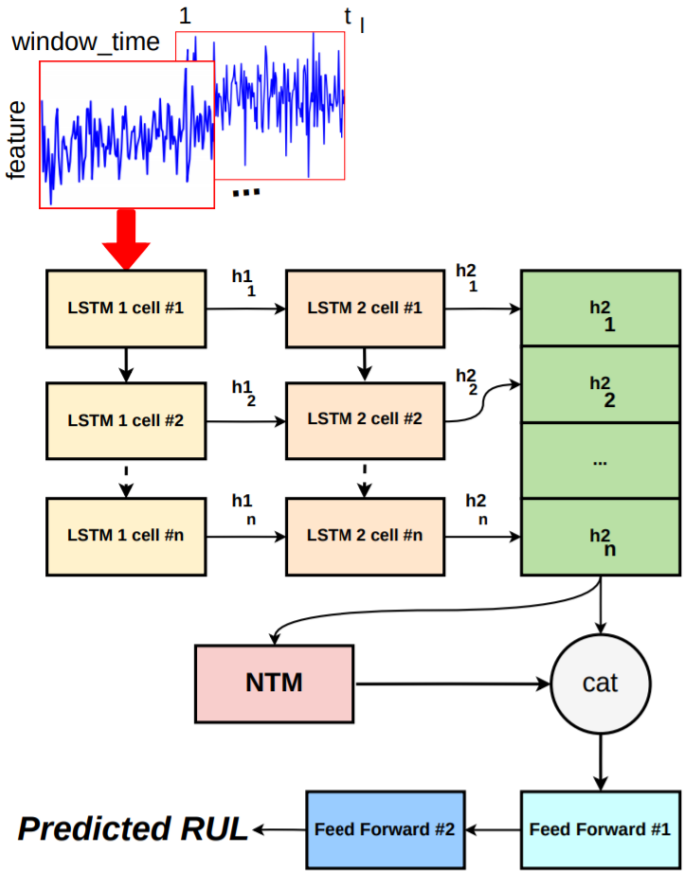
معماری مدل استفاده‌شده در شکل ۴-۱ آورده شده است. داده‌ای که در حوزه بهداشت و پیشگیری استفاده می‌شود معمولا مقادیر اندازه‌گیری‌شده طولانی‌مدت سری‌های زمانی[3] حس‌گرها هستند. سری زمانی‌های خام ورودی با پیش‌پردازش تبدیل به پنجره سری‌زمانی‌های کوچک‌تر می‌شوند. هر کدام از این پنجره‌ها به عنوان ورودی به یک شبکه از دو لایه پشته‌شده LSTM به عنوان ورودی داده می‌شود و یک دنباله از ویژگی‌های استخراج‌شده حاصل می‌گردد. این ویژگی‌ها با ویژگی‌های خروجی یک ماتع الحاق می‌شود. نتیجه نهایی بعد از عبور از یک شبکه جلورو دو لایه پشته‌شده نهایی حاصل می‌گردد.[۲]

فالکن و همکاران معتقدند که وجود یک ماتع می‌تواند کمک به فهم بهتر الگوهای مخفی در داده‌ها و ذخیره‌سازی آن شود. در پژوهش آن‌ها کنترل‌گر ماتع را از نوع شبکه‌های جلورو برگزیدند.[۲] بهبود نتایج آن‌ها که در بخش پنجم به آن اشاره خواهد شد اثبات‌کننده نقش مثبت ماتع در مقاله آنان است.

۴-۲ دسته‌بندی

ملک‌محمدی و صافی‌اصفهانی ادعا کرده‌اند که استفاده از ماتع در وظایف پیچیده‌تر نظیر دسته‌بندی مورد غفلت واقع شده است. آن‌ها توانسته‌اند با ارائه مدلی بر پایه ماتع و الگوریتم ازدحام ذرات[4] توانایی ماتع برای مسائل پیچیده را نشان دهند. علت استفاده از الگوریتم ازدحام ذرات کنترل وزن‌های شبکه بوده است.[۴]

[1] Falcon
[2] Data Driven
[3] Time Series
[4] Particle Swarm



شکل ۴-۱: معماری پیشنهادی برای کاربرد تخمین عمر مفید در کار تحقیقاتی فالکن و همکاران[۲]

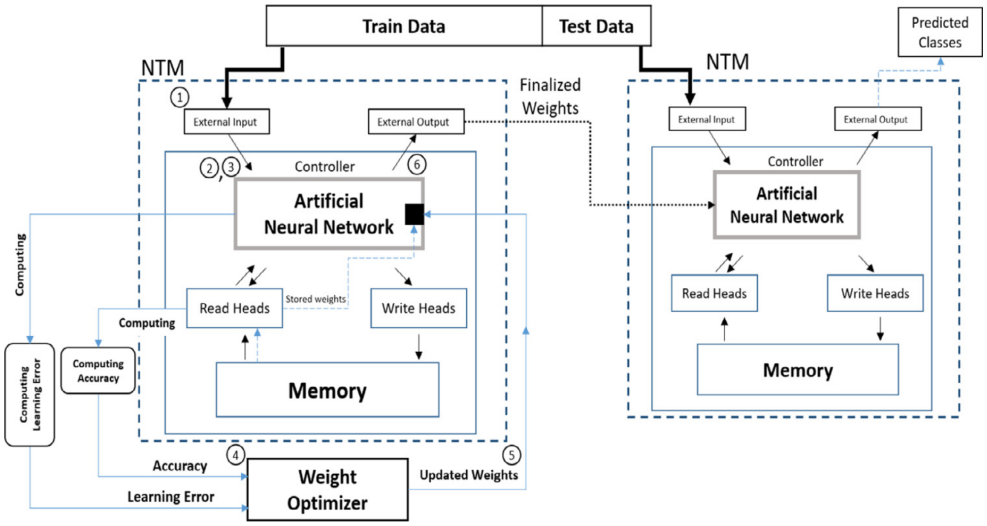
سری‌های زمانی ابتدا به پنجره‌های کوچک‌تر می‌شکنند سپس به عنوان ورودی به شبکه داده می‌شوند. ورودی از دو لایه LSTM پشته‌شده می‌گذرد و سپس با خروجی یک ماتع الحاق می‌شود. در نهایت شبکه جلورو پشته‌شده مقدار تحمین عمر مفید را ارائه می‌دهند.

در شکل ۴-۱ معماری مدل در زمان آموزش و آزمون آورده شده است. در شکل ۴-۲ نیز فلوچارت پیشنهادی آن‌ها آورده شده است. الگوریتم ازدحام ذرات در هنگام آموزش استفاده می‌شود و هدف آن پیدا کردن وزن‌های بهینه با نرخ همگرایی مناسب است. در معماری مدل استفاده شده توسط آن‌ها از یک LSTM برای پیاده‌سازی کنترل‌گر استفاده کردند.[۴]

در مجموعه‌داده‌های متعارفی که برای دسته‌بندی وجود دارد غالبا یک داده در اختیار مدل قرار داده می‌شود و از آن خواسته می‌شود تا کلاس داده را پیش‌بینی کند. کاربرد دیگری در حوزه پردازش تصویر وجود دارد که در آن پیکسل‌های یک عدد دست‌نوشته را به شکل یک دنباله از اعداد در اختیار مدل قرار می‌دهند و کلاس عدد دست‌نوشته را مورد سنجش قرار می‌دهند.

گالچره[۵] و همکاران با استفاده از ماتع پویا توانسته‌اند این مسئله دسته‌بندی خاص را نیز بهبود

[5] Gulcehre



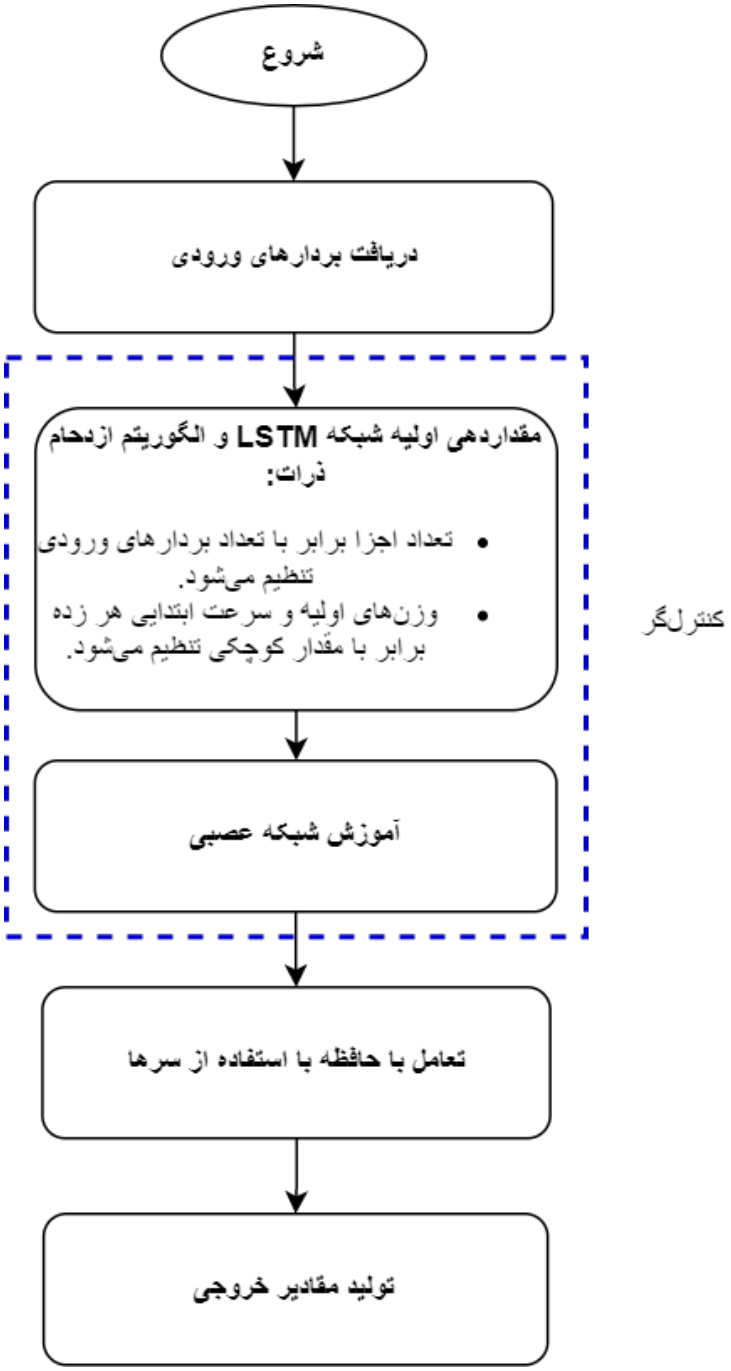
شکل ۴-۲: معماری مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۴]

دهند.[۳]

۴-۳ ردیابی دانش

ژائو[۶] و همکاران پژوهشی برای کاربرد ردیابی دانش[۷] انجام داده‌اند. در برخی از سیستم‌های یادگیری آنلاین یک پشتیبان در نظر گرفته شده است که بر اساس دانش دانشجو و وضعیت ارزیابی‌های او فعالیت‌های بعدی را برای او انتخاب می‌کند. با کمک یادگیری ماشین وضعیت دانشجو می‌تواند با مدل‌سازی رابطه بین فعالیت‌های ترتیبی یادگیری و درست‌بودن تلاش‌های یادگیری تخمین‌زده شود. وضعیت دانش تخمین‌زده شده برای استنباط تسلط بر دانش استفاده می‌شود. شروع سرد[۸] ردیابی دانش یک سناریو است که اطلاعات مشاهده‌شده از دانشجو پیش‌بینی وضعیت دانش دانشجو کافی نیست. به عنوان مثال دوره‌هایی که برای اولین بار برگزار شده‌اند و یا دانشجوهایی که به تازگی به سیستم یادگیری آنلاین پیوسته‌اند با این مشکل مواجه هستند. ژائو و همکاران در پژوهششان از ماتع متوجه برای رفع مشکل شروع سرد استفاده کرده‌اند.[۶]

[6] Zhao
[7] Knowledge Tracing
[8] Cold Start



شکل ۴-۳: فلوچارت مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۴]

۴-۴ پاسخ دوره‌ای به سوالات

یکی از حوزه‌هایی که مورد استفاده ماتع و افزونه‌هایش بوده است پاسخ به سوالات دوره‌ای[۹] است. در این کاربرد تعدادی حقیقت به مدل داده می‌شود و نهایتا یک سوال پرسیده می‌شود. مدل باید بتواند از میان حقایق خوانده‌شده پاسخ را پیدا کند.

گالچره و همکاران به عنوان توسعه‌دهندگان مدل ماتع پویا توانستند از مدل مذکور برای این کاربرد استفاده کنند و بهبود مناسبی برای آن به وجود آورند.[۳]

۵-۴ استنتاج زبان طبیعی

ماتع پویا برای حوزه استنتاج زبان طبیعی[۱۰] هم استفاده شده است. در این کاربرد رابطه بین دو قطعه از متن بررسی می‌شود. دو جمله ممکن است مستلزم یکدیگر باشند، در تضاد باهم باشند و یا رابطه‌ای خنثی داشته باشند.

[9] Episodic Question-Answering

[10] Natural Language Inference

فصل پنجم

نتایج

۵-۱ تخمین عمر مفید باقی‌مانده

در جدول ۵-۱ بخشی از نتایج مربوط به کارتحقیقاتی فالکن و همکاران که مدلی بر پایه ماتع برای تحمین عمر مفید باقی‌مانده وسایل مکانیکی پیشنهاد داده بودند آورده شده است. آن‌ها برای ارزیابی مدل از یک تابع امتیاز که در مقاله‌ای دیگر معرفی شده بود و معیار خطای ریشه میانگین مربعات خطا[۱] استفاده کرده بودند. هر چه این دو معیار برای یک مدل پایین‌تر باشد آن مدل کاراتر است.

جدول ۵-۱: نتایج کار تحقیقاتی فالکن و همکاران برای تخمین عمر مفید باقی‌مانده[۲]

مدل	امتیاز	ریشه میانگین مربعات خطا
LSTM	۳۳۹	۱۶/۱۶
LSTM + ماتع	۲۴۲	۱۲/۵۰

همانگونه که مشخص است استفاده از ماتع در کنار LSTM منجر به کاهش حدود ۲۲ درصدی خطا شده است.

۵-۲ دسته‌بندی

در جدول ۵-۲ بخشی از نتایج کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی آورده شده است. در کار تحقیقاتی آن‌ها با بهبود ماتع و استفاده از الگوریتم ازدحام ذرات مدل دسته‌بند کارایی توسعه داده‌اند. معیار ارزیابی آن‌ها صحت[۲] بوده است. برای مجموعه‌داده دسته‌بندی هم از چهار مجموعه‌داده مطرح در این حوزه استفاده کردند. نهایتا برای مدل از مدل‌های دسته‌بند بردار ماشین پشتیبان [۳]، k-نزدیک‌ترین همسایه [۴]، بیز ساده‌لوحانه[۵]، LSTM، ماتع استاندارد، کامپیوتر عصبی متمایز[۶] و نهایتا ماتع با الگوریتم ازدحام ذرات که روش پیشنهادی آن‌ها بود استفاده شده است.[۴] لازم به ذکر است کامپیوتر عصبی متمایز نیز یکی از افزونه‌های مانع است که در این گزارش به آن پرداخته نشده است. نتایج در جدول ۵-۲ آورده شده است.

همانطور که از نتایج جدول ۵-۲ بر می‌آید بهترین دقت‌های برای سه مدلی است که بر پایه ماتع طراحی شده‌اند. این‌ها نشان از کارایی ماتع و افزونه‌های آن برای کاربرد دسته‌بندی است.
گالچره و همکاران هم برای دسته‌بندی عدد از روی دنباله پیکسل‌ها نتایجی را بدست آورده‌اند که

[1] Rooted Mean Square Error (RMSE)
[2] Accuracy
[3] Support Vector Machine (SVM)
[4] K-Nearest Neighbour (KNN)
[5] Naive Bayes
[6] Differentiable Neural Computer (DNC)

جدول ۵-۲: نتایج کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی برای دسته‌بندی [۴]

مجموعه داده	ماشین پشتیبان	k- نزدیک ترین همسایه	بیز ساده لوحانه	درخت تصمیم	LSTM	ماتع	کامپیوتر عصبی متمایز	ماتع + الگوریتم ازدحام ذرات
MNIST	۹۴/۱۶	۹۶/۹	۵۶/۱۵	۶۵/۴۰	۹۶/۴۸	۹۶/۹۶	۹۹/۱۲	۹۹/۷۳
ORL	۸۴/۶۸	۹۲/۵	۷۷/۲۵	۸۸/۵	۹۴/۲	۹۵/۱۱	۹۷/۲۱	۹۷/۹
Leter	۸۴/۱۱	۸۹/۸۱	۹۳/۲۱	۸۳/۶	۹۵/۵	۹۶/۰۱	۹۸/۱۶	۹۹/۰۲
Ionosphere	۸۰/۳	۷۷/۷۸	۸۲/۶۲	۸۴/۵	۹۱/۱۶	۹۳/۴۱	۹۶/۰۲	۹۷/۱

در این بخش به آن اشاره می‌شود. آن‌ها با مدل ماتع پویا و بر روی مجموعه‌داده Sequential pMNIST کارایی ماتع را سنجیده‌اند که نتایج آن در جدول ۵-۳ آورده شده است.[۳]

۵-۳ ردیابی دانش

شکل ۵-۱ شامل بخشی از نتایج آزمایشات انجام‌شده ژائو و همکاران است. آن‌ها برای ارزیابی از معیار صحت و ناحیه زیر نمودار[۷] استفاده کرده‌اند. مدل رقیب هم شبکه LSTM بدون ماتع متوجه است. با توجه به آنکه مدل پیشنهادی آنان برای حل مشکل شروع سرد در حوزه ردیابی دانش ارائه شده است سه سناریو مرتبط با این وضعیت طراحی کرده‌اند:

۱. سناریو اول: دانشجو کم

۲. سناریو دوم: دنباله‌های یادگیری فعالیت کوتاه

۳. سناریو سوم: هم دانشجو کم و هم دنباله‌های یادگیری فعالیت کوتاه

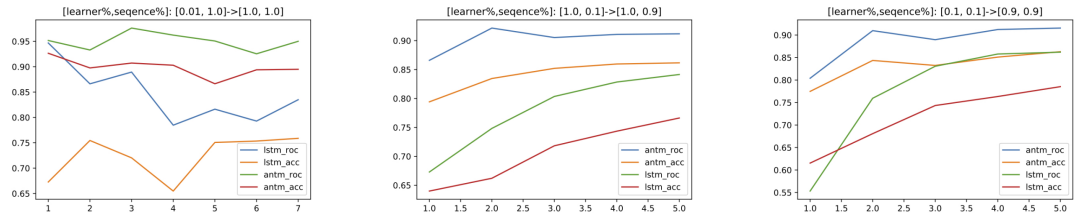
۵-۴ پاسخ به سوالات دوره‌ای

جدول ۵-۳: نتایج کار تحقیقاتی گالچره و همکاران برای پاسخ به سوالات دوره‌ای[؟]

معیار	LSTM	ماتع	ماتع پویا پیوسته	ماتع پویا گسسته
میانگین خطا	۳۶/۴۱	۳۱/۴۲	۲۴/۲۴	۲۱/۷۹
خطای بالای ۵ درصد	۱۶	۱۶	۱۶	۱۲

[7] Area Under Curve (AUC)

Scenario1	Scenario2	Scenario3



شکل ۵-۱: نتایج کار تحقیقاتی ژائو و همکاران برای ردیابی دانش[۶]

باتوجه به شکل ۵-۱ می‌توان دید که ماتع متوجه به عنوان یکی دیگر از افزونه‌های ماتع توانسته بهبودی برای شروع سرد در کاربرد ردیابی دانش ارائه دهد.

فصل ششم

جمع‌بندی و نتیجه‌گیری

منابع و مراجع

[1] Collier, Mark and Beel, Joeran. Implementing neural turing machines. In International Conference on Artificial Neural Networks, pages 94–104. Springer, 2018.

[2] Falcon, Alex, D'Agostino, Giovanni, Serra, Giuseppe, Brajnik, Giorgio, and Tasso, Carlo. A neural turing machine-based approach to remaining useful life estimation. In 2020 IEEE International Conference on Prognostics and Health Management (ICPHM), pages 1–8. IEEE, 2020.

[3] Graves, Alex, Wayne, Greg, and Danihelka, Ivo. Neural turing machines. arXiv preprint arXiv:1410.5401, 2014.

[4] Gulcehre, Caglar, Chandar, Sarath, Cho, Kyunghyun, and Bengio, Yoshua. Dynamic neural turing machine with continuous and discrete addressing schemes. Neural computation, 30(4):857–884, 2018.

[5] Malekmohamadi Faradonbe, Soroor and Safi-Esfahani, Faramarz. A classifier task based on neural turing machine and particle swarm algorithm. Neurocomputing, 396:133–152, 2020.

[6] Merrild, Jakob, Rasmussen, Mikkel Angaju, and Risi, Sebastian. Hyperentm: evolving scalable neural turing machines through hyperneat. In International Conference on the Applications of Evolutionary Computation, pages 750–766. Springer, 2018.

[7] Zhao, Jinjin, Bhatt, Shreyansh, Thille, Candace, Gattani, Neelesh, and Zimmaro, Dawn. Cold start knowledge tracing with attentive neural turing machine. In Proceedings of the Seventh ACM Conference on Learning@ Scale, pages 333–336, 2020.

واژه‌نامه‌ی فارسی به انگلیسی

ا

ابر تکامل عصبی توپولوژی تقویت‌کننده Hyper Neuroevolution of Augmenting Topologies

ازدحام ذرات . . . . . Particle Swarm

استنتاج زبان طبیعی Natural Language Inference

انتروپی . . . . . Entropy

انتشار به عقب . . . . . Backpropagation

ب

برپایه داده . . . . . Data Driven

بردار ماشین پشتیبان . . Support Vector Machine

بستر . . . . . Substrate

بیز ساده‌لوحانه . . . . . Naive Bayes

بیل . . . . . Beel

پ

پاسخ دوره‌ای به سوالات . . . . . Episodic Question-Answering

پیوسته . . . . . Continues

ت

تابع قدم . . . . . Step Function

تارشدن . . . . . Blurring

تقارن . . . . . Symmetry

تکامل عصبی توپولوژی‌های تقویت‌کننده Neuroevolution of Augmenting Topologies

تکرار . . . . . Repetition

تکرار با تغییر Repetition with Variation

توجه چندسر . Multi Headed Attention

توجه سخت . . . . . Hard Attention

توجه نرم . . . . . Soft Attention

تیزکردن . . . . . Sharpen

ج

جابجایی . . . . . Shift

جمعیت . . . . . Population

جهش . . . . . Mutation

خ

خطای هوبر . . . . . Hubber Loss

خطای ریشه مربعات خطا . Rooted Mean Square Error

د

دانش پیشین . . . . . Prior Knowledge

دروازه . . . . . Gate

درون‌یابی . . . . . Interpolation

دسته . . . . . Batch

ر

ردیابی دانش . . . . Knowledge Tracing

رونوشت‌گیری . . . . . Copy

رونوشت‌گیری تکرارشونده . Repeat Copy

ژ

ژائو . . . . . Zhao

س

سر . . . . . Head

سری‌های زمانی . . . . . Time Series

ش

شبکه پرسپترونی چندلایه . Multi Layer Perceptran

شبکه تولید الگوی ترکیبی Compositional Pattern Producing Networks

شبکه عصبی . . . . . Neural Network

شبکه عصبی بازگشتی Recurrent Neural Network

شبکه عصبی بازگشتی دروازه‌دار . . Gated Recurrent Neural Network

شبکه حافظه‌ای . . . . Memory Network

شروع سرد . . . . . Cold Start

ص

صحت . . . . . Accuracy

ف

فاکتور تیزی . . . . . Sharpness Factor

فالکن . . . . . Falcon

ک

کامپیوتر عصبی متمایز . . Differentiable Neural Computer

کولیر . . . . . Collier

گ

گالچره . . . . . Gulcehre

گریوز . . . . . Graves

گسسته . . . . . Discrete

ل

لاجیتس . . . . . Logits

م

ماشین تورینگ عصبی . . Neural Turing Machine

ماشین تورینگ عصبی تکاملی Evolvable Neural Turing Machine

مقداردهی اولیه . . . . . Initialization

منبع‌باز . . . . . Open Source

منظم‌سازی . . . . . Regularity

ن

ناحیه زیرنمودار . . . . Area Under Curve

نسل . . . . . Generation

نوار . . . . . Tape

ی

یادآوری انجمنی . . . Association Recall

یادگیری ترتیبی . . . Sequence Learning

واژه‌نامه‌ی انگلیسی به فارسی

A

Accuracy . . . . . صحت

Area Under Curve . . . . ناحیه زیرنمودار

Association Recall . . . یادآوری انجمنی

B

Backpropagation . . . . . انتشار به عقب

Batch . . . . . دسته

Beel . . . . . بیل

Blurring . . . . . تارشدن

C

Cold Start . . . . . شروع سرد

Collier . . . . . کولیر

Compositional شبکه تولید الگوی ترکیبی
Pattern Producing Networks

Continues . . . . . پیوسته

Copy . . . . . رونوشت‌گیری

D

Data Driven . . . . . برپایه داده

Differentiable . . کامپیوتر عصبی متمایز
Neural Computer

Discrete . . . . . گسسته

E

Episodic . . . . . پاسخ دوره‌ای به سوالات
Question-Answering

Entropy . . . . . انتروپی

Evolvable ماشین تورینگ عصبی تکاملی
Neural Turing Machine

F

Falcon . . . . . فالکن

G

Gate . . . . . دروازه

Gated . . شبکه عصبی بازگشتی دروازه‌دار
Recurrent Neural Network

Generation . . . . . نسل

Graves . . . . . گریوز

بیز ساده‌لوحانه . . . . . Naive Bayes

استنتاج زبان طبیعی Natural Language Inference

شبکه عصبی . . . . . Neural Network

ماشین تورینگ عصبی . . Neural Turing Machine

تکامل عصبی توپولوژی‌های تقویت‌کننده Neuroevolution of Augmenting Topologies

O

تک‌روشن . . . . . One-hot

منبع‌باز . . . . . Open Source

P

ازدحام ذرات . . . . . Particle Swarm

جمعیت . . . . . Population

دانش پیشین . . . . . Prior Knowledge

R

شبکه عصبی بازگشتی Recurrent Neural Network

منظم‌سازی . . . . . Regularity

رونوشت‌گیری تکرارشونده . Repeat Copy

تکرار . . . . . Repetition

تکرار با تغییر Repetition with Variation

خطای ریشه مربعات خطا . Rooted Mean Square Error

S

گالچره . . . . . Gulcehre

H

توجه سخت . . . . . Hard Attention

سر . . . . . Head

خطای هوبر . . . . . Hubber Loss

ابر تکامل عصبی توپولوژی تقویت‌کننده Hyper Neuroevolution of Augmenting Topologies

I

مقداردهی اولیه . . . . . Initialization

درون‌یابی . . . . . Interpolation

K

ردیابی دانش . . . . Knowledge Tracing

k-نزدیک‌ترین همسایه . . . . . K-Nearest Neighbours

L

لاجیتس . . . . . Logits

M

شبکه حافظه‌ای . . . . Memory Network

توجه چندسر . Multi Headed Attention

شبکه پرسپترونی چندلایه . Multi Layer Perceptran

جهش . . . . . Mutation

N

تابع قدم . . . . . Step Function

تقارن . . . . . Symmetry

T

نوار . . . . . Tape

سری‌های زمانی . . . . . Time Series

Z

ژائو . . . . . Zhao

یادگیری ترتیبی . . . Sequence Learning

تیزکردن . . . . . Sharpen

فاکتور تیزی . . . . . Sharpness Factor

جابجایی . . . . . Shift

توجه نرم . . . . . Soft Attention

بستر . . . . . Substrate

بردار ماشین پشتیبان . . Support Vector Machine