



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

پروژه تحقیقاتی درس شبکه‌های عصبی

ماشین تورینگ عصبی

نگارش

علیرضا مازوچی

استاد درس

دکتر رضا صفابخش

تیر ۱۴۰۱

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

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. نهایتاً 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، شبکه عصبی بازگشتی، ماشین تورینگ، مکانیسم توجه، یادگیری ترتیبی

فهرست مطالب

صفحه

عنوان

۱	مقدمه	۱
۵	ماشین تورینگ عصبی	۲
۶	۱-۲ ساختار و محاسبات ماشین تورینگ عصبی	
۸	۲-۲ وظایف یادگیری ترتیبی ساختگی	
۹	افزونه‌ها	۳
۱۰	۱-۳ ماشین تورینگ عصبی تکاملی	
۱۲	۲-۳ ماشین تورینگ عصبی ابر تکاملی	
۱۲	۱-۲-۳ ابر تکامل عصبی توپولوژی‌های تقویت‌کننده	
۱۳	۲-۲-۳ بستر وظیفه رونوشت‌گیری	
۱۶	۳-۳ ماشین تورینگ عصبی متوجه	
۱۷	۴-۳ ماشین تورینگ عصبی پویا	
۱۸	۱-۴-۳ معماری ماشین تورینگ عصبی پویا	
۲۱	۲-۴-۳ مکانیسم آدرس‌دهی	
۲۲	۳-۴-۳ آموزش	
۲۴	کاربردهای واقعی	۴
۲۵	۱-۴ تخمین عمر مفید باقی‌مانده	
۲۵	۲-۴ دسته‌بندی	
۲۷	۳-۴ ردیابی دانش	
۲۹	۴-۴ پاسخ دوره‌ای به سوالات	
۲۹	۵-۴ استنتاج زبان طبیعی	
۳۰	نتایج	۵
۳۱	۱-۵ تخمین عمر مفید باقی‌مانده	
۳۱	۲-۵ دسته‌بندی	
۳۲	۳-۵ ردیابی دانش	
۳۳	۴-۵ پاسخ به سوالات دوره‌ای	
۳۴	جمع‌بندی و نتیجه‌گیری	۶
۳۵	منابع و مراجع	
۳۷	واژه‌نامه‌ی فارسی به انگلیسی	

۴۰ واژه‌نامه‌ی انگلیسی به فارسی
----	------------------------------------

شکل	فهرست اشکال	صفحه
۱-۳	نحوه کارکرد شبکه تات [۶]	۱۴
۲-۳	بستر طراحی شده برای شبکه تات در مدل مائع ابر تکاملی [۶]	۱۵
۳-۳	معماری مدل مائع پویا [۴]	۲۰
۱-۴	معماری پیشنهادی برای کاربرد تخمین عمر مفید در کار تحقیقاتی فالکن و همکاران [۲]	۲۶
۲-۴	معماری مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۵]	۲۷
۳-۴	فلوچارت مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۵]	۲۸
۱-۵	نتایج کار تحقیقاتی ژائو و همکاران برای ردیابی دانش [۷]	۳۳

صفحه	فهرست جداول	جدول
۳۱	۱-۵ نتایج کار تحقیقاتی فالکن و همکاران برای تخمین عمر مفید باقی مانده [۲]	
۳۲	۲-۵ نتایج کار تحقیقاتی ملک محمدی و صافی اصفهانی برای دسته بندی [۵]	
۳۲	۳-۵ نتایج کار تحقیقاتی گالچره و همکاران برای پاسخ به سوالات دوره ای [۴]	

فهرست اختصارات

عنوان اختصاری	عنوان کامل
ماتع	ماشین تورینگ عصبی
ماتع تکاملی	ماشین تورینگ عصبی تکاملی
ماتع ابر تکاملی	ماشین تورینگ عصبی ابر تکاملی
ماتع متوجه	ماشین تورینگ عصبی متوجه
ماتع پویا	ماشین تورینگ عصبی پویا
تکتوت	تکامل عصبی توپولوژی‌های تقویت کننده
ابرتوت	ابر تکامل عصبی توپولوژی‌های تقویت کننده
شبکه تات	شبکه تولید الگوی ترکیبی

فصل اول

مقدمه

امروزه شبکه‌های عصبی^۱ به شاخه اصلی مدل‌های هوش مصنوعی تبدیل شده‌اند و تقریباً 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^۲ چنین قابلیت را دارند.

شبکه‌های عصبی بازگشتی دروازه‌دار^۳ نظیر LSTM اگرچه قادر بودند به واسطه یک بردار داخلی اطلاعات پیشین را حفظ کنند ولی این حافظه محدود است و زمانی که 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^۴: 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^۵: ماشین تورینگ عصبی که در ادامه آن را به اختصار م‌ت‌م‌م 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.^[۴]

تفاوت مهم بین دو مدل این است که شبکه‌های حافظه‌ای مکانیسمی برای تغییر محتوای حافظه خارجی را ندارد درحالی که م‌ت‌م‌ها این قابلیت را دارند. در عمل این مسئله منجر به یادگیری ساده‌تر برای وظایف واقعی در شبکه حافظه‌ای می‌شود. در مقابل، م‌ت‌م‌ در ابتدای معرفی عمدتاً بر روی یک سری از وظایف ساختگی با مقیاس کوچک نظیر رونوشت‌گیری^۶ و یادآوری انجمنی^۷ ارزیابی شده است. هرچند م‌ت‌م‌ 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.^[۴]

¹Neural Network

²Recurrent Neural Network (RNN)

³Gated Recurrent Neural Network (Gated RNN)

⁴Memory Network

⁵Neural Turing Machine

⁶Copy

⁷Association Recall

ماتع در سال ۲۰۱۴ توسط گروهی از محققین گوگل به سرپرستی گریوز^۸ ارائه شد [۳]. 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. طبیعتاً 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 مجدداً 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 مستقیماً 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 سراسر 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 ماتع مهمی برای استفاده و توسعه آن است. توسعه‌دهندگان ماتع پیاده‌سازی و جزئیات کافی برای پیاده‌سازی خود را ارائه ندادند [۱]. این مسئله قطعاً تأثیر منفی‌ای برای پیشرفت این شبکه به نسبت پیچیده بود. پیاده‌سازی‌های منبع‌باز^۹ اولیه آن یا سرعت پایینی داشتند و یا ممکن بود وزن‌های آن به بی‌نهایت میل کند و آموزش دچار مشکل شود. نهایتاً در سال ۲۰۱۸ یعنی چهار سال بعد از معرفی ماتع کولیر^{۱۰} و بیل^{۱۱} با یک پیاده‌سازی مناسب و منبع‌باز توانستند نتایج مقاله اصلی را در زمان مناسب تکرار کنند. [۱]. به گفته آنان مقداردهی اولیه^{۱۲} 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 [۴][۶] در سال‌های بعد بهبودهای مهمی بر روی نسخه اولیه اعمال کردند. به علاوه مدل‌های زیادی بر پایه ماتع برای مسائل واقعی و مجموعه داده‌های 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 کاربردهای واقعی و بخشی از نتایج مرتبط به کاربردهای 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

⁸Graves

⁹Open Source

¹⁰Collier

¹¹Beel

¹²Initialization

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 مانع و افزونه‌های آن توانسته‌اند در آن استفاده شوند ارائه می‌شود. برای برخی از کاربردها مانع و افزونه‌های آن تغییراتی داشته است؛ در این موارد تغییرات اساسی تبیین می‌شود.

- در بخش پنجم بخشی از برترین نتایج گزارش شده در مقالات که مدلی بر پایه مانع به دقت‌های خوبی رسیده است گلچین شده است.

- نهایتاً بخش ششم مروری بر مطالب این پروژه خواهد بود.

فصل دوم

ماشین تورینگ عصبی

در این فصل قرار است ساختار و محاسبات ماتع را مورد بررسی قرار دهیم. پس از آن مجموعه‌ای وظایف یادگیری ترتیبی^۱ ساختگی که ماتع سنتی برای انجام آن توانمند است را معرفی خواهیم کرد.

۱-۲ ساختار و محاسبات ماشین تورینگ عصبی

ماتع شامل دو جز است:

۱. کنترل گر^۲ شبکه که می‌تواند یک شبکه عصبی جلورو^۳ یا یک شبکه عصبی بازگشتی باشد.
 ۲. یک واحد حافظه خارجی که یک ماتریس $N * W$ است. N تعداد واحدهای حافظه و W ابعاد هر سلول حافظه را نمایش می‌دهد.^[۱]
- فارغ از آنکه کنترل گر بازگشتی باشد یا خیر، کل معماری بازگشتی محسوب می‌شود چراکه ماتریس حافظه در طول زمان نگهداری می‌شود. کنترل گر سر^۴های خواندن و نوشتن دارد که به ماتریس حافظه دسترسی دارد. تاثیر یک عمل خواندن یا نوشتن روی یک سلول حافظه خاص با مکانیسم توجه نرم^۵ 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 کنترل گر 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^c یعنی آدرس‌دهی وابسته به محتوا در رابطه ۱-۲ آورده شده است.^[۱]

$$w_t^c(i) = \frac{\exp(\beta_t K[k_t, M_t(i)])}{\sum_{j=0}^{N-1} \exp(\beta_t K[k_t, M_t(j)])} \quad (1-2)$$

در رابطه ۱-۲ یک کلید جستجو در حافظه را نشان می‌دهد و K مطابق رابطه ۲-۲ یک معیار شباهت^۶ مانند شباهت کسینوسی^۷ است.^[۱]

¹Sequence Learning

²Controller

³Feedforward Neural Network

⁴Head

⁵Soft Attention

⁶Similarity Metric

⁷Cosine Similarity

$$K[u, v] = \frac{uv}{||u|| \cdot ||v||} \quad (2-2)$$

با یک سری از محاسبات مطابق روابط ۲-۳، ۲-۴ و ۲-۵ ماتعها امکان تکرار بر روی وزنهای حافظه فعلی و قبلا محاسبه شده را خواهند داشت. رابطه ۲-۳ به شبکه اجازه می دهد 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 هسته^۸، کانولوشنی جابجایی^۹ یک بعدی را فراهم می کند. رابطه ۲-۵ رخداد تارشدن^{۱۰} که به واسطه عمل کانولوشن رخ داده است را اصلاح می کند. [۱]

$$w_t^g = g_t w_t^c + (1 - g_t) w_{t-1} \quad (3-2)$$

$$\tilde{w}_t(i) = \sum_{j=0}^{N-1} w_t^g(j) s_t(i - j) \quad (4-2)$$

$$w_t(i) = \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_{j=0}^{N-1} \tilde{w}_t(j)^{\gamma_t}} \quad (5-2)$$

سپس بردار r_t مطابق رابطه ۲-۶ به وسیله ی یک سر خواندن خاص در زمان t محاسبه می گردد. [۱]

$$r_t = \sum_{i=0}^{N-1} w_t(i) M_t(i) \quad (6-2)$$

نهایتا مطابق رابطه ۲-۷ و ۲-۸ هر سر نوشتن ماتریس حافظه را در گام t با محاسبه بردارهای جانبی پاک کردن یعنی e_t و جمع کردن یعنی a_t تغییر می دهد. [۱]

⁸Kernel

⁹Shift

¹⁰Blurring

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

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

در روابط این بخش برخی از پارامترها دارای محدودیت‌هایی هستند که باید به آن اشاره شود: [۱]

$$\bullet \beta_t \geq 0$$

$$\bullet g_t \in [0, 1]$$

$$\bullet \sum_k s_t(k) = 1$$

$$\bullet \forall_k s_t(k) \geq 0$$

$$\bullet \gamma_t \geq 1$$

۲-۲ وظایف یادگیری ترتیبی ساختگی

برای ماتع‌ها چندین وظیفه ساختگی در نظر گرفته شده است که تمام آن‌ها از نوع مسئله یادگیری ترتیبی است؛ زمینه‌ای که آن‌ها در آن توانمند هستند. این وظایف عبارت‌اند از:

- رونوشت‌گیری: 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^{۱۱}: مشابه وظیفه رونوشت‌گیری دنباله‌ای از بردارهای بیتی تصادفی به شبکه داده می‌شود. برخلاف وظیفه رونوشت‌گیری بعد از اتمام دنباله یک عدد که نشان دهنده تعداد دفعاتی است که باید دنباله در خروجی ظاهر شود به شبکه داده می‌شود.
- یادآوری انجمنی^{۱۲}: این وظیفه دارای دنباله با دنباله‌های متشکل از بردارهای بیتی تصادفی است. بعد از آنکه یک دنباله از آیتم‌ها و نماد پایانی دنباله به شبکه داده می‌شود. خروجی صحیح عنصر بعدی دنباله ورودی بعد از عنصر کوثری است. [۱]

¹¹Repeat Copy

¹²Associative Recall

فصل سوم

افزونه‌ها

در فصل قبل معماری و روال آموزش مانع استاندارد را بررسی کردیم. در این فصل قرار است برخی از افزونه‌های آن که در سال‌های بعد از ارائه مانع استاندارد توسعه داده شدند را تشریح کنیم. فاعداً امکان بررسی تمام افزونه‌ها با جزئیات کامل وجود ندارد ولی می‌توان برخی از آن‌ها را جزئیات مناسب ارائه داد.

۳-۱ ماشین تورینگ عصبی تکاملی

در این بخش قرار است با افزونه ماشین تورینگ عصبی تکاملی^۱ که در ادامه آن را به طور اختصار مانع تکاملی می‌نامیم آشنا شویم. قبل از بررسی مدل باید با یک الگوریتم تکاملی^۲ به نام تکامل عصبی توپولوژی‌های تقویت‌کننده^۳ که در ادامه آن را به اختصار تکتوت می‌نامیم آشنا شویم. تکتوت یک الگوریتم تکاملی در شبکه عصبی است که یک رکن مهم از افزونه را تشکیل می‌دهد. ابتکار مانع تکاملی صرفاً به استفاده از یک الگوریتم تکاملی ختم نمی‌شود و تغییرات دیگری در قسمت آموزش مدل دارد که قدرت آن را بیش از پیش زیاد می‌کند.

تکتوت با یک جمعیت^۴ از شبکه‌های عصبی ساده شروع می‌کند و سپس آن‌ها را در طی نسل‌ها^۵ با افزودن گره‌های جدید و بال‌ها^۶ به کمک جهش^۸ پیچیده‌تر می‌کند. با تکامل شبکه‌ها از این راه 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 به تدریج پیچیدگی را افزایش می‌دهد و این باعث می‌شود که یک شبکه مناسب با اندازه مینیمال حاصل شود.^[۶]

بر پایه مانع و با استفاده از تکتوت مدل مانع تکاملی معرفی شده است. در این روش توپولوژی و وزن‌های شبکه عصبی کنترل‌گر با کمک تکتوت یاد گرفته می‌شود. بنابراین برخلاف مانع استاندارد نیاز به دانش پیشین^۹ نیست و شبکه می‌تواند با توجه به پیچیدگی وظیفه رشد پیدا کند. مانع 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 مانع تکاملی قادر به حل مسائل یادگیری مستمر پیچیده است. چراکه شبکه از مشتق استفاده نمی‌کند و می‌تواند از توجه سخت^{۱۰} و مکانیسم جابجایی بهره ببرد.

¹Evolvable Neural Turing Machine

²Evolutionary Algorithm

³Neuroevolution of Augmenting Topologies (NEAT)

⁴Population

⁵Generations

⁶Node

⁷Edge

⁸Mutation

⁹Prior Knowledge

¹⁰Hard Attention

در این شرایط امکان تعمیم خوب برای دنباله‌های بلند در وظیفه رونوشت‌گیری فراهم می‌شود. به علاوه یک نوار^{۱۱} پویا و از نظر تئوری با اندازه بی‌نهایت قابل استفاده شده است.[۶]

مانع تکاملی یک سر تکی ترکیبی خواندن-نوشتن دارد. این شبکه بردار نوشتن w با اندازه M ، ورودی کنترل درونیابی^{۱۲} 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 مخلوط کردن مقادیر فعلی نوار و بردار نوشتن در موقعیت نوشتن را فراهم می‌کند. در لحظه t $M_h(t)$ محتوای نوار در موقعیت سر فعلی i_t ، h درونیابی نوشتن و w_t بردار نوشتن است. برای تمام این‌ها رابطه ۳-۱ را خواهیم داشت.[۶]

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

پرش محتوا مشخص می‌کند که آیا سر باید به موقعیتی در حافظه حرکت کند که بیشترین شباهت را به بردار نوشتن دارد یا نه. اگر مقدار ورودی کنترل از 0.5 بیشتر شود، یک پرش محتوا انجام می‌شود. شباهت بین بردار نوشتن w و بردار حافظه m مطابق با رابطه ۳-۲ حساب می‌شود.[۶]

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

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 با توجه به درونیابی نوشتن i_t درونیابی می‌شود.
۲. اگر ورودی کنترل پرش محتوا j_t بزرگ‌تر از 0.5 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.[۶]

¹¹Tape

¹²Interpolation

۲-۳ ماشین تورینگ عصبی ابر تکاملی

در بخش پیش مانع تکاملی معرفی شد. بر پایه این افزونه، افزونه دیگری به نام ماشین تورینگ عصبی ابر تکاملی^{۱۳} که در ادامه آن را به اختصار مانع ابر تکاملی می‌نامیم توسعه یافته است. در این مدل، روش تکتوت بهبود پیدا کرده است. مدل مانع ابر تکاملی می‌تواند مقیاس‌پذیری^{۱۴} مانع را بهبود دهد. یعنی به کمک مانع ابر تکاملی می‌توان با داده‌های کم و برای ابعاد پایین‌تر مدل را آموزش داد و سپس مدل را حتی بدون هیچ آموزش اضافی در شرایط با مقیاس بالاتر قرار داد!^[۶]

۱-۲-۳ ابر تکامل عصبی توپولوژی‌های تقویت‌کننده

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^{۱۵} 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.^[۶]

روش ابر تکامل عصبی توپولوژی‌های تقویت‌کننده^{۱۶} 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 می‌شود. در تکتوت از شبکه‌های عصبی معمولی استفاده می‌شود در حالی که در ابرتوت از شبکه تولید الگوی ترکیبی^{۱۷} 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 نوع منظم‌سازی^{۱۸} است.^[۶]

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

¹³Hyper Evolvable Neural Turing Machine

¹⁴Scalability

¹⁵Encoding

¹⁶Hyper Neuroevolution of Augmenting Topologies (HyperNEAT)

¹⁷Compositional Pattern Producing Networks (CPPN)

¹⁸Regularity

تقارن^{۱۹}، تکرار^{۲۰} و تکرار با تغییر^{۲۱} 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. نهایتاً تکرار با تغییر به سادگی با ترکیب یک تابع منظم (مانند سینوس یا گاوسین) با یک تابع نامنظم (مانند محور x نامتقارن) بدست می‌آید.^[۶]

ایده اصلی ابرتوت آن است که شبکه تات می‌تواند یال الگوها را کد کند. بدین طریق یک تکتوت می‌تواند یک شبکه تات که شبکه‌های عصبی بزرگ با منظم‌سازی‌های خود را نمایش می‌دهد تکامل دهد.^[۶]

عملکرد شبکه تات در تصویر ۱-۳ آورده شده است. برای استفاده از شبکه تات ابتدا باید هر گره موجود در شبکه در یک فضای چندبعدی قرار بگیرد. قوانین مربوط به آنکه گره‌های شبکه عصبی در کجای این فضا قرار بگیرد اصطلاحاً یک بستر^{۲۲} را شکل می‌دهد. شبکه‌های تات استاندارد توابع هندسی هستند که الگوهای یال خروجی آن رئوسی در n بعد است که n تعداد ابعاد در فضای کارترین است. یک شبکه تات چهار ورودی با برچسب‌های x_1, y_1, x_2, y_2 را دریافت می‌کند و به عنوان خروجی مشخص می‌کند که یال 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 یال آن را بدست آورد و شبکه عصبی را ایجاد کرد.^[۶]

۲-۲-۳ بستر وظیفه رونوشت‌گیری

همانطور که در توضیحات قبل بدان اشاره شد باید تمام گره‌های شبکه عصبی در یک بستر 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 ۱-۳ 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 یال ورودی‌ها و خروجی‌های شبکه عصبی مرتبط با 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 منطقاً باید بتواند الگوی هندسی اطلاعات خوانده شده از و نوشته شده در حافظه را نیز یاد بگیرد.^[۶]

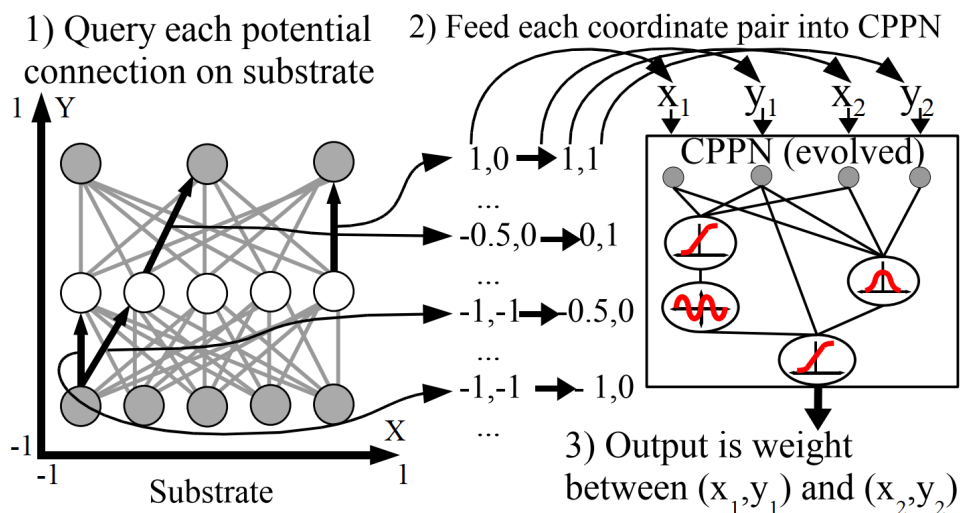
شبکه ماتع تکاملی ورودی‌های زیر را دارد:

¹⁹Symmetry

²⁰Repetition

²¹Repetition with Variation

²²Substrate



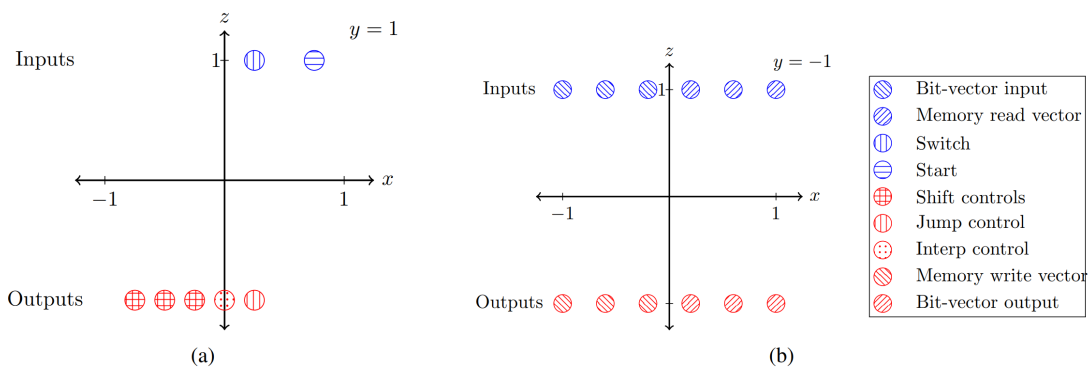
شکل ۳-۱: نحوه کارکرد شبکه تات [۶]

مجموعه‌ای از گره‌ها به مختصات بین -۱ تا +۱ در تمام ابعاد نظیر می‌شوند. (۱): برای هر یال ممکن در یک شبکه عصبی کوثری زده می‌شود تا مجاورت و وزن آن مشخص شود. خطوط جهت‌دار تیره نمایش داده شده در تصویر یک نمونه از یال‌هایی است که کوثری 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 یال‌ها وزن‌دهی می‌شوند که خروجی یک تابع با چه وزنی به طرف دیگر یال برود، برای هر کوثری ارسال شده به شبکه تات جایگاه دو سر یال را به عنوان ورودی می‌گیرد و وزن یال را به عنوان خروجی می‌دهد. (۳): بنابراین شبکه تات می‌تواند الگوهای منظم از یال‌ها در فضا را تولید کند. [۶]

- شروع: ورودی که هرگاه فعال می‌شود، ذخیره اعداد شروع می‌شود.
- تعویض^{۲۳}: 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 بعداً می‌خواند فعال می‌شود.
- ورودی خواندن حافظه: بردار حافظه که ماشین تورینگ در گام قبل خوانده است. [۶]
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.
- کنترل گره‌های ماشین تورینگ: خروجی‌های کنترل مخصوص ماشین تورینگ یعنی پرش، درون‌یابی و سه کنترل جابجایی (چپ، راست و توقف) [۶]

²³Switch

در ادامه بستر طراحی شده برای وظیفه رونوشت‌گیری آورده می‌شود. این بستر در شکل ۳-۲ نشان داده شده است. در این بستر گره‌های ورودی بردار بیتی دارای مختصات یکسان x با گره‌های نوشتن بردار حافظه است. با رویکرد مشابه گره‌های خواندن بردار حافظه و گره‌های خروجی بردار بیتی نیز دارای مختصات x یکسانی هستند. به علاوه مقدار x ورودی تعویض با مقدار x خروجی پرش یکسان است. این مسئله باعث می‌شود پس از آمدن ورودی تعویض شبکه پرش به حافظه‌ای کند که خواندن را از آن شروع کرده است. در ماتع ابرتکاملی اندازه بردارهای حافظه برابر با اندازه بردار بیتی است. نهایتاً باید گفت که در ابرماتع تکاملی گره مخفی ندارد و در بستر هم نشان داده نشده است. [۶]



شکل ۳-۲: بستر طراحی شده برای شبکه تات در مدل ابرتکاملی [۶]

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 کنترل گره‌های ماشین تورینگ هستند. لازم به ذکر است که مختصات x برای ورودی تعویض و خروجی کنترل گر پرش یکسان است. قسمت ب گره‌ها را در $y = -1$ نشان می‌دهد که ورودی‌ها 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. [۶]

در کنار خروجی شبکه تات که وزن هر یال را مشخص می‌کند، هر شبکه تات یک خروجی تابع قدم^{۲۴} اضافه دارد که خروجی بیان پیوند نامیده می‌شود. این خروجی مشخص می‌کند که آیا یک یال باید بیان شود یا خیر. یال‌های بالقوه برای هر ورودی در لایه‌های $y = 1$ و $y = -1$ به هر خروجی در لایه‌های $y = 1$ و $y = -1$ کوثری زده می‌شود. تعداد ورودی‌ها و خروجی‌ها در لایه $y = -1$ مطابق قسمت ب شکل ۳-۲ وابسته به اندازه بردار بیتی وظیفه رونوشت‌گیری است، که در مثال نشان داده شده اندازه بردار بیتی برابر با ۳ است. نوره‌ها به شکل یکنواخت در بازه‌های -1 تا -0.2 در مختصات x برای ورودی‌های بردار بیتی و بردار نوشتن حافظه استفاده می‌شود و بازه 0.2 تا 1 برای بردار نوشتن حافظه و خروجی بردار بیتی استفاده می‌شود. [۶]

نهایتاً 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 کوثری زد. [۶]

²⁴Step Function

۳-۳ ماشین تورینگ عصبی متوجه

یکی دیگر از افزونه‌های مائع، ماشین تورینگ عصبی متوجه که در ادامه آن را به اختصار مائع متوجه^{۲۵} می‌نامیم هست. ایده اصلی مائع متوجه استفاده از مکانیسم توجه^{۲۶} بوده است.

در مدل مائع متوجه مانند مائع استاندارد حافظه با یک مکانیسم آدرس‌دهی وابسته به محتوا بازیابی می‌شود. آدرس‌دهی برپایه محتوا به صورت ضروری یک گام محاسبه شباهت میان بردار خروجی کنترل گر 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. [۷]

$$w_t^r(i) = \frac{\exp(\beta \cdot \text{sim}(C_t, M_t(i)))}{\sum_j (\exp(\beta \cdot \text{sim}(C_t, M_t(j))))} \quad (3-3)$$

در حالت کلی کنترل گر می‌تواند با هر شبکه عصبی مصنوعی طراحی شود. در مائع متوجه از یک لایه توجه چندسر^{۲۷} استفاده کرده‌اند تا روابط بین دنباله‌های ورودی و خروجی را مدل کنند. [۷]

رابطه مربوط به لایه توجه در رابطه ۳-۴ آورده شده است. در این رابطه مطابق معمول 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 وزن دار برای هر i_t بر پایه i_1 تا i_t اعمال شود که t زمان فعلی است. [۷]

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{n}}\right) \quad (4-3)$$

به عنوان خروجی مدل یک بردار وزن مرتبط با هر تلاش را یاد می‌گیرد و بردار وزن یادگرفته‌شده 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^{۲۸}، k طول دنباله و t زمان است. [۷]

²⁵ Attentive Neural Turing Machine (ANTM)

²⁶ Attention Mechanism

²⁷ Multi-Headed Attention

²⁸ Batch

$$L = \sum_b \sum_{k=1}^n \sum_{t=1}^T MSE(a_t^k, gt_t^k) \quad (5-3)$$

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^{۲۹} درونیابی w_t^g مطابق رابطه ۶-۳ است. [۷]

$$w_t^g = g_t w_t^c + (1 - g_t) w_{t1} \quad (6-3)$$

بعد از درونیابی یک وزن‌دهی جابجایی s_t بر ماتریس وزن دروازه‌دار با یک کانولوشن دایره‌ای برای تنظیم حافظه مطابق رابطه ۷-۳ اعمال می‌شود. در این رابطه N برابر با اندازه حافظه است. نهایتاً مطابق رابطه ۸-۳ عمل تیزکردن^{۳۰} برای نرمال‌سازی استفاده می‌شود. [۷]

$$w_t^{ro} = \sum_{j=0}^{N-1} w_t^g(j) s_t(i - j) \quad (7-3)$$

$$w_t(i) = \frac{w_t^{ro}(i)}{\sum_j w_t^{ro}(j)} \quad (8-3)$$

۴-۳ ماشین تورینگ عصبی پویا

ماتع استاندارد 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. ماشین تورینگ عصبی پویا^{۳۱} که در ادامه به اختصار آن را ماتع پویا می‌نامیم برای بهبود در این زمینه معرفی

²⁹Gate

³⁰Sharpen

³¹Dynamic Neural Turing Machine (DNTM)

شد. ماتع پویا این محدودیت را با معرفی یک بردار آدرس‌دهی قابل یادگیری برای هر سلول حافظه در ماتع که اخیراً به عنوان مکانیسم آدرس‌دهی حافظه استفاده شده است حل کرده است. [۴]

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. [۴]

$$M_t[i] = [A_t[i]; C_t[i]] \quad (۳-۹)$$

$$M_t = [A_t; C_t] \quad (۳-۱۰)$$

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. [۴]

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 کنترل گر جلورو و هم استفاده از کنترل گر GRU وجود دارد. نحوه محاسبه وضعیت مخفی^{۳۲} با استفاده از این دو کنترل گر به ترتیب در رابطه ۱۱-۳ و ۱۲-۳ آمده است.[۴]

$$h_t = \sigma(x_t, r_t) \quad (11-3)$$

$$h_t = GRU(x_t, h_{t-1}, r_t) \quad (12-3)$$

در گام زمانی t کنترل گر x_t را به عنوان ورودی می‌گیرد و سپس وزن خواندن w_t^r تولید می‌شود. سپس بردار محتوا r_t مطابق رابطه ۱۳-۳ ایجاد می‌شود. نهایتاً وضعیت مخفی کنترل گر h_t وابسته به بردار محتوای حافظه r_t و وضعیت مخفی پیشین کنترل گر h_{t-1} محاسبه می‌شود و مدل برچسب خروجی y_t برای ورودی را پیش‌بینی می‌کند.[۴]

$$r_t = M_t^T w_t^r \quad (13-3)$$

کنترل گر با پاک کردن محتوای قدیمی و نوشتن اطلاعات جدید حافظه را بروز می‌کند. کنترل گر سه بردار را محاسبه می‌کند: بردار پاک کردن 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 پرسپترون چندلایه^{۳۳} پیاده‌سازی می‌شود. این بردارهای وزن برای تعامل با حافظه استفاده می‌شود. بردار پاک کردن نیز با یک شبکه پرسپترون ساده محاسبه می‌شود که وابسته به وضعیت مخفی کنترل گر h_t است. بردار محتوای حافظه نامزد بر مبنای وضعیت مخفی فعلی h_t و ورودی کنترل گر که با یک دروازه عددی α_t مقیاس شده است خواهد بود. α_t یک تابع از وضعیت مخفی و ورودی کنترل گر است. در رابطه ۱۴-۳ و ۱۵-۳ به ترتیب نحوه محاسبه α_t و \bar{c}_t آورده شده است. با داشتن بردارهای پاک کردن، نوشتن و محتوای حافظه نامزد ماترس محتوا مطابق با رابطه ۱۶-۳ قابل بروزسانی است.[۴]

$$\alpha_t = f(h_t, x_t) \quad (14-3)$$

³²Hidden State

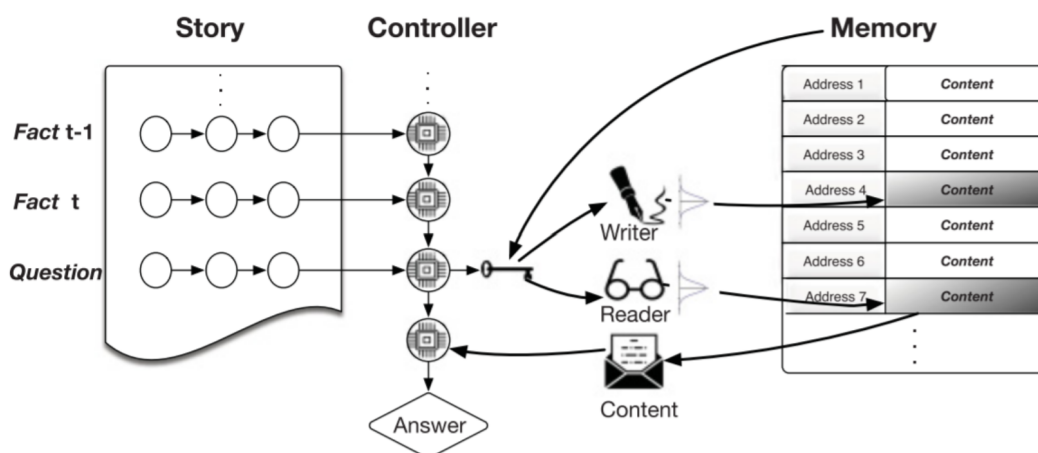
³³Multi Layer Perceptron (MLP)

$$\bar{c}_t = \text{ReLU}(W_m h_t + \alpha_t W_x x_t) \quad (15-3)$$

$$C_t[j] = (1 - e_t w_t^w[j]) \odot C_{t-1}[j] + w_t^w[j] \bar{c}_t \quad (16-3)$$

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 کاملاً نادیده گرفته می‌شوند. بی‌عملی برای نوشتن در ماتع پویا معادل با یادگیری آن است که دروازه نوشتن در ماتع استاندارد محتوای حافظه را بدون تغییر باقی بگذارد. [۴]

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



شکل ۳-۳: معماری مدل ماتع پویا [۴]

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. [۴]

۲-۴-۳ مکانیسم آدرس‌دهی

کنترل‌گر بردار کلید را مطابق با رابطه ۱۷-۳ بدست می‌آورد. سپس با کمک رابطه ۱۸-۳ وزن‌های لاجیتس^{۳۴} 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^{۳۵} است که نحوه محاسبه آن در رابطه ۱۹-۳ آمده است. تابع $softmax$ مطابق با رابطه ۲۰-۳ خواهد بود. [۴]

$$k_t = W_k^T h_t + b_k \quad (۱۷-۳)$$

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

$$\beta_t = softplus(u_\beta^T h_t + b_\beta) + 1 \quad (۱۹-۳)$$

$$softplus(x) = \log(\exp(x) + 1) \quad (۲۰-۳)$$

میانگین وزن‌دار نمایی لاجیتس آدرس‌دهی v_t مطابق با رابطه ۲۱-۳ محاسبه می‌شود. سپس با کمک رابطه ۲۲-۳ و ۲۳-۳ وزن‌های آدرس‌دهی w_t حاصل می‌شود. با کمک این روابط وزن مربوط به سطرهایی از حافظه که اخیراً کمتر مورد استفاده قرار گرفته‌اند افزایش می‌یابد. تاثیر این رخداد با کمک γ_t تعیین می‌شود. این مکانیسم تعمیمی بر آدرس‌دهی برپایه محتوا استاندارد است. [۴]

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

^{۳۴}Logits^{۳۵}Sharpness Factor

$$\gamma_t = \text{sigmoid}(u_\gamma^T h_t + b_\gamma) \quad (22-3)$$

$$w_t = \text{softmax}(z_t - \gamma_t v_{t-1}) \quad (23-3)$$

هر سطر از ماتریس آدرس‌دهی w_t دارای مقدار مثبت است که جمع آن‌ها برابر با یک است. می‌توان بردار تک‌روشن^{۳۶} \tilde{w}_t را از روی آن ایجاد کرد. در زمان آموزش این بردار با رابطه ۳-۲۴ ایجاد می‌شود. این برنامه آدرس‌دهی گسسته^{۳۷} و مدل هم مانع پویای گسسته نامیده می‌شود. این درحالی است که بدون این مکانیسم مدل مانع پویا پیوسته^{۳۸} نام خواهد گرفت.^[۴]

$$\tilde{w}_t[k] = I(k = \text{argmax}(w_t)) \quad (24-3)$$

در انتهای این بخش باید تاکید کرد که در مانع پویا و در هر گام زمانی کنترل‌گر می‌تواند بیشتر از یک درخواست برای دسترسی به حافظه داشته باشد. این کار با یک گزینه اضافه انجام می‌شود. در مانع استاندارد با چندین سر این کار می‌توانست انجام شود.^[۴]

۳-۴-۳ آموزش

تابع هزینه مورد استفاده در مانع پویا پیوسته مطابق رابطه ۳-۲۵ است. مانع پویا پیوسته می‌تواند مانند مانع استاندارد با انتشار به عقب^{۳۹} آموزش یابد ولی در مانع پویا گسسته به دلیل استراتژی نمونه‌برداری در زمان آموزش این امکان وجود نخواهد داشت.^[۴]

$$C(\theta) = -\frac{1}{N} \sum_{n=1}^N \log p(y^{(n)} | x_1^{(n)}, \dots, x_T^{(n)}; \theta) \quad (25-3)$$

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

³⁶One-Hot

³⁷Discrete

³⁸Continues

³⁹Backpropagation

نرمال می‌شود. در رابطه ۲۸-۳ $b(x)$ یک شبکه است که برای کاهش خطای هوبر^{۴۰} آموزش می‌بیند. این خطا در رابطه ۲۹-۳ آورده شده است. مقدار z برابر با $\bar{R}(x)$ خواهد بود.^[۴]

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

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

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

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

تابع خطا برای حالت گسسته در رابطه ۳۰-۳ آورده شده است. در این رابطه \mathcal{H} نمایانگر انتروپی^{۴۱} است.

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

⁴⁰Hubber Loss

⁴¹Entropy

فصل چهارم

کاربردهای واقعی

در دو فصل قبل ماتب و شماری از افزونه‌های آن معرفی شد. ماتب و افزونه‌هایش در سال‌های اخیر کاربردهای گوناگونی پیدا کرده‌اند. کاربردهایی که نه به صورت ساختگی که به صورت واقعی و بر اساس مجموعه داده‌ها و معیارهای ارزیابی شناخته شده تحقق پیدا کرده‌اند. در این فصل بخش قصد داریم به تعدادی از کاربردهای آن بپردازیم. برای شماری از کاربردها که معماری مخصوص یا آموزش خاصی مورد استفاده بوده است مختصراً موارد ویژه بیان خواهد شد.

۴-۱ تخمین عمر مفید باقی مانده

فالکن^۱ و همکاران در تحقیقی عمر مفید باقی مانده 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 بهبود در برنامه ریزی 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 تنظیم کردن مدل‌های برپایه داده^۲ استفاده شود.[۲]

معماری مدل استفاده شده در شکل ۴-۱ آورده شده است. داده‌ای که در حوزه بهداشت و پیشگیری استفاده می‌شود معمولاً مقادیر اندازه‌گیری شده طولانی مدت سری‌های زمانی^۳ حس گرها هستند. سری زمانی‌های خام ورودی با پیش پردازش تبدیل به پنجره سری‌های زمانی‌های کوچک‌تر می‌شوند. هر کدام از این پنجره‌ها به عنوان ورودی به یک شبکه از دو لایه پشته شده LSTM به عنوان ورودی داده می‌شود و یک دنباله از ویژگی‌های استخراج شده حاصل می‌گردد. این ویژگی‌ها با ویژگی‌های خروجی یک ماتب الحاق می‌شود. نتیجه نهایی بعد از عبور از یک شبکه جلورو دو لایه پشته شده نهایی حاصل می‌گردد.[۲]

فالکن و همکاران معتقدند که وجود یک ماتب می‌تواند کمک به فهم بهتر الگوهای مخفی در داده‌ها و ذخیره سازی آن کند. در پژوهش آن‌ها کنترل گر ماتب را از نوع شبکه‌های جلورو برگزیدند.[۲] بهبود نتایج آن‌ها که در بخش پنجم به آن اشاره خواهد شد اثبات کننده نقش مثبت ماتب در مقاله آنان است.

۴-۲ دسته بندی

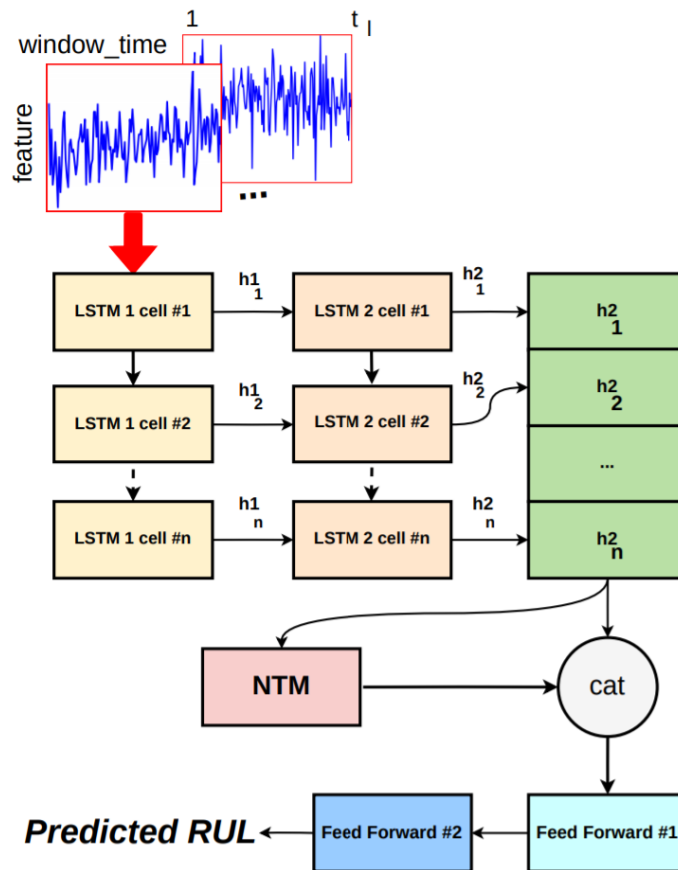
ملک محمدی و صافی اصفهانی ادعا کرده‌اند که استفاده از ماتب در وظایف پیچیده تر نظیر دسته بندی مورد غفلت واقع شده است. آن‌ها توانسته‌اند با ارائه مدلی بر پایه ماتب و الگوریتم ازدحام ذرات^۴ توانایی ماتب برای

¹Falcon

²Data Driven

³Time Series

⁴Particle Swarm

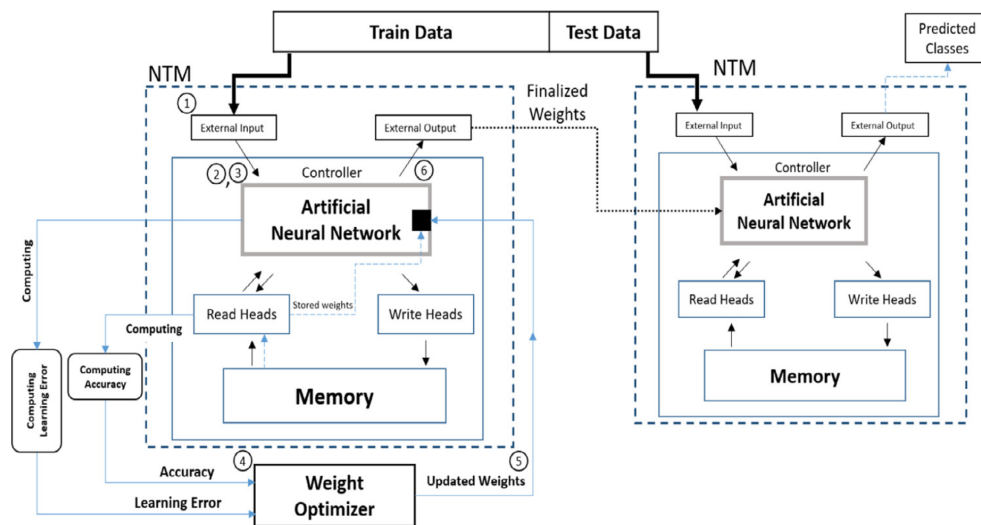


شکل ۴-۱: معماری پیشنهادی برای کاربرد تخمین عمر مفید در کار تحقیقاتی فالکن و همکاران [۲]. سری‌های زمانی ابتدا به پنجره‌های کوچک‌تر می‌شکنند سپس به عنوان ورودی به شبکه داده می‌شوند. ورودی از دو لایه LSTM پشته‌شده می‌گذرد و سپس با خروجی یک مانتع الحاق می‌شود. در نهایت شبکه جلورو پشته‌شده مقدار تخمین عمر مفید را ارائه می‌دهند. [۲]

مسائل پیچیده را نشان دهند. علت استفاده از الگوریتم ازدحام ذرات کنترل وزن‌های شبکه بوده است. [۵]

در شکل ۴-۲ معماری مدل در زمان آموزش و آزمون آورده شده است. در شکل ۴-۳ 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. [۵]

در مجموعه داده‌های متعارفی که برای دسته‌بندی وجود دارد غالباً 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



شکل ۴-۲: معماری مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۵]

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 دهند. [۴]

۴-۳ ردیابی دانش

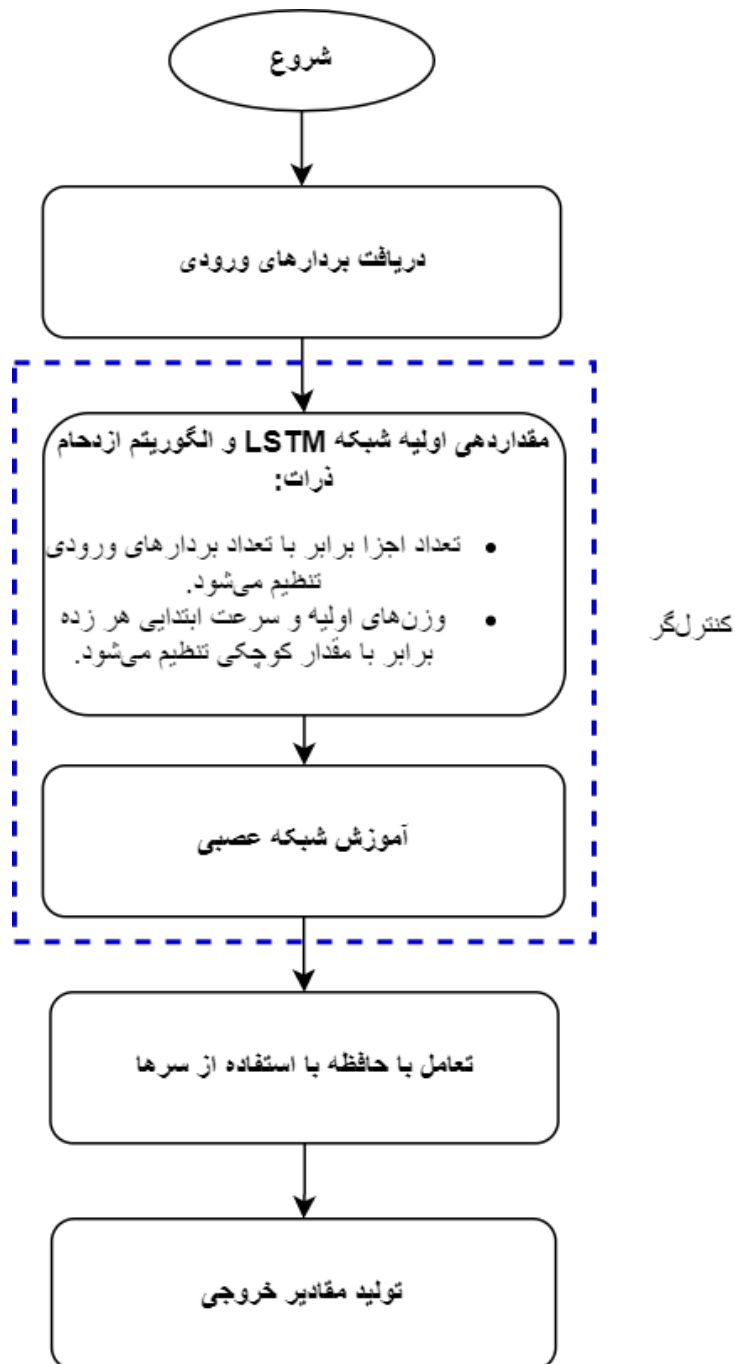
ژائو^۶ و همکاران پژوهشی برای کاربرد ردیابی دانش^۷ انجام داده‌اند. در برخی از سیستم‌های یادگیری آنلاین یک پشتیبان در نظر گرفته شده است که بر اساس دانش دانشجو و وضعیت ارزیابی‌های او فعالیت‌های بعدی را برای او انتخاب می‌کند. با کمک یادگیری ماشین وضعیت دانشجو می‌تواند با مدل‌سازی رابطه بین فعالیت‌های ترتیبی 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 برای پیش‌بینی وضعیت دانش دانشجو کافی نیست. به عنوان مثال دوره‌هایی که برای اولین بار برگزار شده‌اند و یا دانشجویایی 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. [۷]

⁵Gulcehre

⁶Zhao

⁷Knowledge Tracing

⁸Cold Start



شکل ۴-۳: فلوچارت مدل پیشنهادی برای کاربرد دسته‌بندی در کار تحقیقاتی ملک‌محمدی و صافی‌اصفهانی [۵]

۴-۴ پاسخ دوره‌ای به سوالات

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

گالچره و همکاران به عنوان توسعه‌دهندگان مدل مائع پویا توانستند از مدل مذکور برای این کاربرد استفاده کنند و بهبود مناسبی برای آن به وجود آورند.^[۴]

۵-۴ استنتاج زبان طبیعی

مائع پویا برای حوزه استنتاج زبان طبیعی^{۱۰} هم استفاده شده است.^[۴] 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.

^۹Episodic Question-Answering

^{۱۰}Natural Language Inference

فصل پنجم

نتایج

در فصل قبل بیان شد که ماتع و افزونه‌های آن در کاربردهای مختلف استفاده شده‌اند. در این بخش قصد داریم شماری از بهترین نتایج ماتع و افزونه‌های آن در این کاربردها را بررسی کنیم.

۱-۵ تخمین عمر مفید باقی‌مانده

در جدول ۱-۵ بخشی از نتایج مربوط به کار تحقیقاتی فالکن و همکاران که مدلی بر پایه ماتع برای تخمین 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 کارتر است.^[۲]

جدول ۱-۵: 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.

۲-۵ دسته‌بندی

در جدول ۲-۵ 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 مجموعه داده دسته‌بندی هم از چهار مجموعه داده مطرح در این حوزه استفاده کردند. نهایتاً برای مدل از مدل‌های دسته‌بند بردار ماشین پشتیبان^۳، k-نزدیک‌ترین همسایه^۴، بیز ساده لوحانه^۵، LSTM، ماتع استاندارد، کامپیوتر عصبی متمایز^۶ و نهایتاً ماتع با الگوریتم ازدحام ذرات که روش پیشنهادی آن‌ها بود استفاده شده است.^[۵] 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 ۲-۵ آورده شده است.

^۱Rooted Mean Square Error (RMSE)

^۲Accuracy

^۳Support Vector Machine (SVM)

^۴K-Nearest Neighbour (KNN)

^۵Naive Bayes

^۶Differentiable Neural Computer (DNC)

جدول ۵-۲: نتایج کار تحقیقاتی ملک محمدی و صافی اصفهانی برای دسته‌بندی [۵]

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

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. بعد از این مدل‌ها، مدل LSTM به عنوان یک مدل شبکه عصبی بازگشتی بهترین نتایج را گرفته است.

گالچره و همکاران هم برای دسته‌بندی عدد از روی دنباله پیکسل‌ها نتایجی را بدست آورده‌اند. آن‌ها با مدل ماتع پویا و بر روی مجموعه داده Sequential pMNIST کارایی ماتع را سنجیده‌اند که نتایج آن در جدول ۵-۳ آورده شده است. [۴]

جدول ۵-۳: نتایج کار تحقیقاتی گالچره و همکاران برای پاسخ به سوالات دوره‌ای [۴]

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

مطابق با جدول ۵-۳ می‌توان دید که چه ماتع استاندارد و چه افزونه‌های آن نتایج بهتری نسبت به LSTM بدست آورده‌اند. بر اساس جدول ۵-۲ و ۵-۳ می‌توان مشاهده کرد مدل افزونه نسبت به مدل استاندارد بهبود قابل ملاحظه‌ای داشته است.

۵-۳ ردیابی دانش

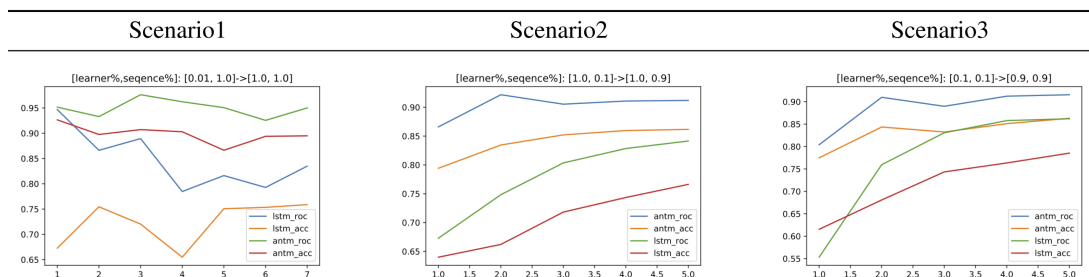
شکل ۵-۱ شامل بخشی از نتایج آزمایشات انجام شده ژائو و همکاران است. آن‌ها برای ارزیابی از معیار صحت و ناحیه زیر نمودار^۷ استفاده کرده‌اند. مدل رقیب هم شبکه LSTM بدون ماتع متوجه است. با توجه به آنکه مدل پیشنهادی آنان برای حل مشکل شروع سرد در حوزه ردیابی دانش ارائه شده است سه سناریو مرتبط با این وضعیت طراحی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

^۷Area Under Curve (AUC)

۱. سناریو اول: دانشجو کم

۲. سناریو دوم: دنباله‌های یادگیری فعالیت کوتاه

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



شکل ۵-۱: نتایج کار تحقیقاتی ژائو و همکاران برای ردیابی دانش [۷]

باتوجه به شکل ۵-۱ می‌توان دید که مائع متوجه به عنوان یکی دیگر از افزونه‌های مائع 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, Soroosh 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

Beel بیل	۱
پ	
Episodic پاسخ دوره‌ای به سوالات Question-Answering	ابر تکامل عصبی توپولوژی تقویت‌کننده Hyper Neuroevolution of Augmenting Topologies
Continues پیوسته	ازدحام ذرات Particle Swarm
ت	
Step Function تابع قدم	Natural Language استنتاج زبان طبیعی Inference
Blurring تارشدن	Evolutionary الگوریتم تکاملی Algorithm
Switch تعویض	Entropy انتروپی
Symmetry تقارن	Backpropagation انتشار به عقب
تکامل عصبی توپولوژی‌های تقویت‌کننده Neuroevolution of Augmenting Topologies	ب
Repetition تکرار	Data Driven برپایه داده
Repetition with Variation تکرار با تغییر	Support Vector . . بردار ماشین پشتیبان Machine
Multi Headed Attention . توجه چندسر	Substrate بستر
Hard Attention توجه سخت	Naive Bayes بیز ساده‌لوحانه
Soft Attention توجه نرم	

Time Series سری‌های زمانی	Sharpen تیز کردن
ش	ج
Cosine Similarity شباهت کسینوسی	Shift جابجایی
Multi Layer شبکه پرسپترون چندلایه	Population جمعیت
Perceptron	Mutation جهش
Compositional شبکه تولید الگوی ترکیبی	خ
Pattern Producing Networks	Hubber Loss خطای هوبر
Neural Network شبکه عصبی	Rooted Mean خطای ریشه مربعات
Recurrent Neural شبکه عصبی بازگشتی	Square Error
Network	د
Gated شبکه عصبی بازگشتی دروازه‌دار	Prior Knowledge دانش پیشین
Recurrent Neural Network	Gate دروازه
Feedforward Neural شبکه عصبی جلورو	Interpolation درونیابی
Network	Batch دسته
Memory Network شبکه حافظه‌ای	ر
Cold Start شروع سرد	Knowledge Tracing ردیابی دانش
ص	رونوشت‌گیری
Accuracy صحت	Copy رونوشت‌گیری
ف	Repeat Copy رونوشت‌گیری تکرارشونده
Sharpness Factor فاکتور تیزی	ژ
Falcon فالکن	ژائو Zhao
ک	س
Differentiable کامپیوتر عصبی متمایز	Head سر
Neural Computer	
Encoding کدگذاری	
Controller کنترل‌گر	

Area Under Curve ناحیه زیرمنمودار	Collier کولیر
Generation نسل	گ
Tape نوار	Gulcehre گالچره
ن	Node گره
□□□□□ □□□□□ . □□□□□ □□□□□□	Graves گریوز
ه	Discrete گسسته
Kernel هسته	ل
ی	Logits لاجیتس
Association Recall . . . یادآوری انجمنی	م
Sequence Learning . . . یادگیری ترتیبی	ماشین تورینگ عصبی . . Neural Turing Machine
Edge یال	ماشین تورینگ عصبی ابر تکاملی . . Hyper Evolvable Neural Turing Machine
	ماشین تورینگ عصبی پویا . . . Dynamic Neural Turing Machine
	ماشین تورینگ عصبی تکاملی Evolvable Neural Turing Machine
	معیار شباهت Similarity Metric
	مقداردهی اولیه Initialization
	مقیاس‌پذیری Scalability
	مکانیسم توجه . . Attention Mechanism
	منبع‌باز Open Source
	منظم‌سازی Regularity

ن

واژه‌نامه‌ی انگلیسی به فارسی

A	رونوشت‌گیری Copy
Accuracy صحت	Cosine Similarity . . شباهت کسینوسی
Area Under Curve . . . ناحیه زیرنمودار	D
Association Recall . . یادآوری انجمنی	Data Driven برپایه داده
Attention Mechanism . . مکانیسم توجه	Differentiable . . کامپیوتر عصبی متمایز
B	Neural Computer
Backpropagation انتشار به عقب	Discrete گسسته
Batch دسته	Dynamic . . . ماشین تورینگ عصبی پویا
Beel بیل	Neural Turing Machine
Blurring تارشدن	E
C	Edge یال
Cold Start شروع سرد	Episodic پاسخ دوره‌ای به سوالات
Collier کولیر	Question-Answering
Compositional شبکه تولید الگوی ترکیبی	Encoding کدگذاری
Pattern Producing Networks	Entropy انتروپی
Continues پیوسته	Evolutionary الگوریتم تکاملی
Controller کنترل‌گر	Algorithm

ماشین تورینگ عصبی تکاملی Evolvable Neural Turing Machine	مقداردهی اولیه Initialization
F	درون‌یابی Interpolation
Falcon	K
شبکه عصبی جلورو Feedforward Neural Network	هسته Kernel
G	ردیابی دانش Knowledge Tracing
Gate	k -Nearest
Gated	Neighbours
شبکه عصبی بازگشتی دروازه‌دار . . Recurrent Neural Network	L
Generation	لاجیتس Logits
Graves	M
Gulcehre	شبکه حافظه‌ای Memory Network
H	توجه چندسر Multi Headed Attention .
Hard Attention	شبکه پرسپترونی چندلایه Multi Layer .
Head	Perceptron
Hidden State	Mutation
Hubber Loss	N
Hyper	بیز ساده‌لوحانه Naive Bayes
Evolvable Neural Turing Machine	استنتاج زبان طبیعی Natural Language Inference
ابر تکامل عصبی توپولوژی تقویت‌کننده Hyper Neuroevolution of Augmenting Topologies	شبکه عصبی Neural Network
I	ماشین تورینگ عصبی Neural Turing Machine
	تکامل عصبی توپولوژی‌های تقویت‌کننده Neuroevolution of Augmenting Topologies
	گره Node
	O

One-hot تک‌روشن	Sequence Learning . . . یادگیری ترتیبی
Open Source منبع‌باز	Sharpen تیزکردن
P	Sharpness Factor فاکتور تیزی
Particle Swarm ازدحام ذرات	Shift جابجایی
Population جمعیت	Similarity Metric معیار شباهت
Prior Knowledge دانش پیشین	Soft Attention توجه نرم
R	Substrate بستر
Recurrent Neural Network شبکه عصبی بازگشتی	Support Vector Machine بردار ماشین پشتیبان
Regularity منظم‌سازی	Step Function تابع قدم
Repeat Copy . رونوشت‌گیری تکرارشونده	Switch تعویض
Repetition تکرار	Symmetry تقارن
Repetition with Variation تکرار با تغییر	T
Rooted Mean Square Error خطای ریشه مربعات خطا	Tape نوار
S	Time Series سری‌های زمانی
Scalability مقیاس‌پذیری	Z
	Zhao ژائو