دوفصلنامه محاسبات و سامانه های توزیع شده
سال چهارم، شماره اول، شماره پیاپی ۷، صفحه ۹۰ تا ۱۰۰، سال ۱۴۰۰

مروری بر روش‌های هرس در شبکه‌های عصبی عمیق با تاکید بر روش‌های هرس پیش از آموزش

عطیه فیروزه*[۱]، مرتضی محجل[۲]، محبوبه شمسی[۲]
[۱]دانشجوی کارشناسی ارشد، دانشکده برق و کامپیوتر، صنعتی قم، قم، ایران.
[۲]استادیار دانشکده برق و کامپیوتر، صنعتی قم، قم، ایران.

چکیده

با گسترش کاربرد شبکه‌های عصبی، عمیق شدن و افزایش پارامترهای شبکه، در عین حال محدودیت منابع محاسباتی، محدودیت در حافظه و تفسیرناپذیر شدن این شبکه‌ها، فشرده‌سازی شبکه‌های عصبی مورد توجه قرار گرفته است. فشرده‌سازی می‌بایست به صورت هوشمندانه باشد، به نحوی که ما را از مزایای بهره مندی از شبکه‌های عصبی عمیق جدا نکند. هرس به عنوان یکی از روش‌های فشرده‌سازی با حذف پارامترهای غیرضروری شبکه، همواره مورد اقبال پژوهشگران بوده است، به نحوی که در پژوهش‌های اخیر سعی شده است، مرحله‌ای با عنوان هرس پیش از آموزش شبکه، در مراحل قبل از راه‌اندازی شبکه گنجانده شود تا از مزایای فشرده‌سازی و هرس در مراحل آموزش و استنتاج شبکه بهره برده شود. در مقاله پیش رو سعی شده است، مروری بر روش‌های هرس با تاکید بر هرس‌های پیش از آموزش شبکه انجام شود. در ابتدا مبانی هرس مطرح شده، سپس انواع هرس به همراه تعریف ریاضی هریک مطرح و در نهایت بررسی دقیق‌تری روی هرس‌های پیش از آموزش شبکه انجام شده است.

کلمات کلیدی: شبکه عصبی عمیق، فشرده‌سازی شبکه عصبی، فرضیه بلیط بخت‌آزمایی، هرس پیش از راه‌اندازی شبکه.

A Review on Pruning Techniques in Deep Neural Networks with Emphasis on Prune at Initial

Atieh firoozeh*, Dr.M.Mohajel, Dr.M.Shamsi
Qom University of Technology, Qom, Iran.

Abstract

With the expansion of the use of neural networks, increase deep of NN, and increase of network parameters, as well as the limitation of computational resources, the limitation of memory, and the incomprehensibility of these networks, the compression of neural networks is necessary. Compression must be intelligent, so as not to deprive us of the benefits of deep neural networks. Pruning is one of the compression methods that eliminate unnecessary network parameters. in recent research, Pruning has always been favored by researchers as far as a step called pruning at the initial design that pruned the initial network to include the benefits compression and pruning in the training and inference. this article reviews pruning techniques in deep neural networks with emphasis on Prune at initializing. First, the basics of pruning are discussed, then the types of pruning with the mathematical definition of each discussed, and finally, a more detailed study of pruning before network training has been done.

Keywords: pruning NN, Sparsification NN, Comperesion NN, LTH.

تاریخچه مقاله:

تاریخ ارسال: ۱۴۰۰/۰۲/۲۷
تاریخ اصلاحات: ۱۴۰۰/۰۳/۱۷
تاریخ پذیرش: ۱۴۰۰/۰۶/۰۲
تاریخ انتشار: ۱۴۰۰/۰۶/۳۱

Keywords:

Pruning NN,
Sparsification NN,
Comperesion NN,
LTH

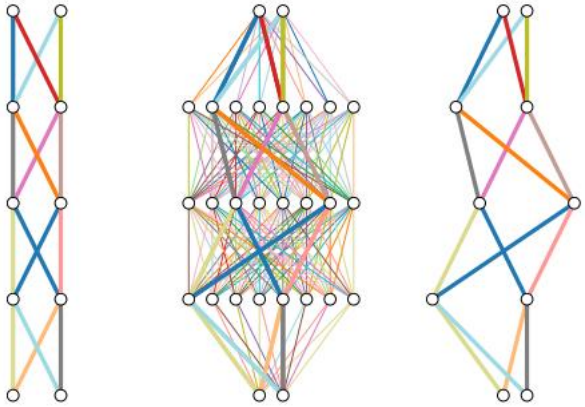
*نویسنده مسئول :
Firoozeh.a@qut.ac.ir

۱-مقدمه

با عمیق شدن شبکه‌های عصبی و افزایش چشم گیر تعداد پارامترهای آن، در عین حال محدودیت منابع محاسباتی، محدودیت حافظه و پیچیدگی بالای شبکه‌های عصبی، فشرده‌سازی یکی از دغدغه‌های اصلی در حوزه یادگیری عمیق شده است[۱].

هرس یکی از راه‌های فشرده‌سازی می‌باشد، که با کاهش اتصالات یا گره‌های شبکه، به فشرده‌سازی شبکه کمک می‌نماید. در واقع هرس به هیچ عنوان ساده سازی محاسباتی انجام نمی‌دهد، بلکه تعداد محاسبات و دسترسی به حافظه را کاهش می‌دهد. پس بر اساس نوع داده، دو مدل هرس می‌توان تعریف نمود: هرس وزن (کاهش تعداد اتصالات)و هرس نورون (کاهش تعداد گره‌ها)[۲].

یکی از نقاط قوت شبکه عصبی عمیق، رسیدن به دقت بالا با توجه به تعداد بالای لایه‌ها و پارامترهای شبکه است. در واقع هرس همین نقطه قوت را هدف گرفته و عملا سعی در کم کردن پارامترها دارد. می‌توان گفت محدودیت‌ها سبب شده، سراغ هرس شبکه‌ای برویم که بزرگی آن یکی از فاکتورهای اصلی عملکردش محسوب می‌شود. پس به صورت پیش فرض این فکر وجود داشت که هرس عملکرد شبکه را دچار اختلال خواهد کرد و از دقت در شبکه خواهد کاست[۳]. در پروسه هرس سنتی شاهد کاهش دقت شبکه بعد از هرس هستیم، به نحوی که شبکه مجددا آموزش داده می‌شود تا میزان دقت شبکه هرس شده، به دقت شبکه اصلی نزدیکتر شود. در واقع می‌توان گفت می‌بایست بین میزان هرس در شبکه عصبی و میزان دقت شبکه یک توازن برقرار نمود زیرا هرس زیاد، دقت شبکه را کاهش می‌دهد و جهت رسیدن به دقت بالا می‌بایست اتصالات با اهمیت شبکه حفظ شود[۴]. همچنین هر چه شبکه بزرگتر و پارامترهای شبکه بیشتر باشد، احتمال وجود اتصالات کم اهمیت و غیر لازم مستعد هرس شدن، بیشتر است. پس می‌توان گفت هرس درشبکه بزرگتر مثل شبکه‌های عصبی پیچشی[۱] و شبکه‌های عصبی بازگشتی[۲] در مقابل شبکه‌های عصبی متراکم[۳] نتیجه بهتری خواهد داشت و صحت و دقت بالاتری به همراه دارد[۲]. وجود اتصالات کم اهمیت در شبکه، در واقع وجود یک زیر شبکه با دقت بالا را در شبکه‌ی راه‌اندازی شده‌ی ابتدایی ثابت می‌کند. زیرشبکه‌ای که می‌تواند با شبکه هرس شده پس از آموزش کامل شبکه، متفاوت باشد [۵] (شکل-۱):.



(شکل-۱): اگر یک شبکه عصبی با وزن‌های تصادفی (گراف وسط) به اندازه کافی و بیش از حد پارامتر داشته باشد، دارای یک زیرشبکه (سمت راست) خواهد بود که به خوبی یک شبکه عصبی آموزش دیده و هرس شده (سمت چپ) با همان تعداد پارامتر عمل می‌کند[۵].

در کل هدف اصلی در روند هرس، صرفه جویی در منابع محاسباتی، صرفه جویی در حافظه و تسریع سرعت تولید خروجی می‌باشد؛ به گونه‌ای که کارایی شبکه حفظ شود، صحت و دقتی معادل عملکرد شبکه پیش از هرس ایجاد گردد و سربار محاسباتی موازی در روند هرس ایجاد نشود (زمان محاسبه شبکه هرس شده معادل زمان آموزش شبکه عمیق نشود. عملا صرفه جویی محاسباتی و زمانی مشاهده نشود). البته در برخی منابع یاد شده است که حتی اگر برای آموزش و استنتاج شبکه هرس شده معادل با شبکه اصلی منابع و زمان صرف شود؛ باز هم هرس منطقی به نظر می‌رسد، زیرا سادگی زیادی به شبکه اعمال می‌گردد و شبکه تفسیر پذیرتر می‌گردد در حالی که عملکرد شبکه عصبی عمیق همچنان وجود خواهد داشت[۱].

در پژوهش جاری ابتدا مبانی هرس و چالش‌های آن مورد بررسی قرار گرفته است، سپس انواع هرس با ارائه تاریخچه‌ای از پژوهش‌های انجام شده روی هر یک از انواع هرس‌ها به اختصار مطرح گردیده است. جهت بررسی دقیق‌تر روش‌های هرس، نیاز است زبان ریاضی تعریف هرس مطرح شود که در بخش ۰ مطرح و در همین بخش با توجه به تعاریف مطرح شده، هر یک از انواع هرس تعریف شده در بخش۰ ، به صورت فرمال تعریف شده است .در نهایت با توجه به تاکید مقاله جاری بر روی هرس‌های پیش از راه‌اندازی شبکه، این نوع هرس در بخش ۰ مورد بررسی دقیق‌تر قرار گرفته و روش‌های پر کاربرد این نوع هرس در این بخش به تفصیل بررسی و مقایسه شده است.

[1]Convolutional Neural Network (CNN)
[2]Recurrent Neural Networks (RNN)
[3]Neural Network with Dense Layer

۲-مبانی هرس

هرس در واقع حذف وزنهای بلا استفاده از شبکه‌های عصبی است. با هرس شبکه‌های عصبی تعداد پارامترهای شبکه کاهش می‌یابد، منابع محاسباتی بهینه صرف می‌شود، حافظه کمتری صرف می‌گردد، در انرژی مصرفی صرفه جویی شده و شبکه‌ای تفسیر پذیرتر ارائه می‌گردد[۱].

در بحث هرس ۴ چالش اصلی همواره مطرح است[۶].

ساختار هرس چه گونه باشد ؟[۴]

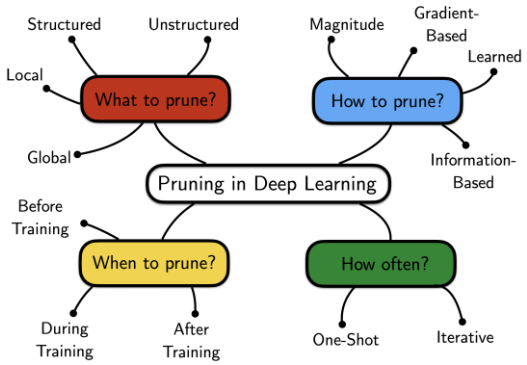
تا چه میزان هرس در شبکه عمیق صورت گیرد ؟[۵]

بر چه اساسی، موجودیت مد نظر جهت هرس انتخاب شود ؟[۶]

هرس در چه زمانی انجام شود؟[۷]

در مورد ساختار هرس دو دیدگاه عمومی هرس ساختارمند و هرس دون ساختار وجود دارد. در هرس بدون ساختار، مکان هرس به صورت تصادفی انتخاب می‌شود که بیشتر جهت حفظ منابع حافظه استفاده می‌گردد. ولی هرس ساختار مند، بنا بر قوانینی خاص هرس صورت می‌پذیرد و این نوع هرس بیشتر برای تسریع عملکرد و کم کردن تاخیر کاربرد دارد [۶]. هرس یک کانال از یک فیلتر یا کل یک فیلتر در شبکه‌های عصبی پیچشی یا کانولوشن را جزئی از طبقه‌بندی هرس‌های ساختارمند طبقه بندی می شود [۷].

نسبت هرس (شبکه تا چه حد هرس شود) می‌تواند پیش از شورع هرس شبکه به دو صورت سراسری یا به‌ازای هر لایه تعیین و یا به صورت پویا، بنا بر میزان یک متغیر خاص در زمان هرس شبکه معین شود[۶]. در هرس سراسری نرخ هرس منعطف‌تری در کل شبکه وجود دارد و رعایت نرخ خاصی از هرس در هر لایه اجباری نیست [۸]، به علاوه این روش ساده‌تر است و فرا پارامترهای[۸] کمتری در روند اجرا نیاز دارد، به این علت همواره توجه به سمت هرس سراسری بوده و اکثریت پژوهش‌ها هرس سراسری را بسیار بهتر از هرس لایه‌ای می‌دانند[۹]، البته می‌توان محاسبه نسبت هرس به صورت خودکار برای هر لایه را نیز یکی از گزینه‌های انتخابی در این بخش دانست[۱۰].



(شکل-۲): چه چیز را ، چگونه، چه زمانی و هر چند وقت یکبار هرس کنیم [۱۱]؟

اینکه چگونه آنچه باید هرس شود، انتخاب گردد، چالشی‌ترین موضوع این بحث است. راههای زیادی برای تشخیص اینکه کدام پیوند هرس شود مطرح شده است؛ مانند قدر مطلق وزن پیوند، پیوندی که کمتر روی تابع زیان تاثیر می‌گذارد و ... [۶].

زمانبندی هرس نیز می‌توان به دو صورت تک مرحله‌ای یا تکرار شونده باشد[۲, ۶]. در هرس تکرار شونده با هر مرحله هرس باز خورد از شبکه گرفته شده و ادامه مراحل هرس صورت می‌پذیرد و یکباره شبکه به نسبت هرس نهایی مد نظر نمی‌رسد. پژوهش‌ها اکثریت نشان از آن دارد که هرس تکرار شونده عملکرد بهتری دارد و دقت عملکرد بالاتری را به همراه دارد[۱].

انواع هرس: انواع هرس را در سه طبقه کلی می‌توان جای داد: هرس سنتی، هرس مبتنی بر فرضیه بلیط بخت‌آزمایی و هرس پیش از راه‌اندازی شبکه.

هرس سنتی[۹]

هرس سنتی در واقع آموزش کامل شبکه، تشخیص مقادیر کم اهمیت در شبکه و هرس آنها و آموزش مجدد شبکه است. ایده اصلی هرس شبکه‌های عصبی با هدف فشرده‌سازی و استفاده بهینه از منابع محاسباتی و حافظه به سال ۱۹۹۰ برمی‌گردد[۱۲]. در آن بازه زمانی تکیه بر این اصل بود که ضمن حفظ دقت و صحت عملکرد شبکه، پارامترها کاهش یابد. در ادامه اولین تحقیق در حوزه هرس بر مبنای مشتق دوم شکل گرفت[۱۳]. کمی بعد در سال ۲۰۱۵ روند هرس بر مبنای وزن[۱۰] شکل گرفت[۱۴]. در زمینه هرس بر مبنای مقادیر مختلف، تحقیقات زیادی در بازه زمانی سال ۲۰۱۵ تا ۲۰۱۸ انجام پذیرفته است، که هر یک الگوریتمی متفاوت برای این امر ارائه نموده‌اند.

[4]What to prune - pruning structure
[5]How many (connections or neurons) to prune - pruning ratio
[6]Which to prune - pruning criterion
[7]How to prune - pruning schedule
[8]Hyper parameter
[9]Traditional pruning
[10]Per-weight magnitude-based

رویکرد کلیه این روش‌ها آموزش شبکه سپس انجام هرس و در صورت نیاز آموزش مجدد شبکه است. الگوریتم پایه این رویکرد در [۱۷] ارائه شده است.

Algorithm 1: Pruning and fine-tuning

Input: N, the number Of iterations Of pruning, and
X, the dataset on which to train and fine-tune

1. W ← initialize()
2. W ← trainToConvergence(f(X;W))
3. M ← $1^{|W|}$
4. **For** i in 1 to N **do**
5. M ← prune(M, score(W))
6. W ← fineTune(f(X; M⊙W))
7. **End for**
8. **Return** M,W

(شکل-۳): رویکرد هرس سنتی و باز تنظیم شبکه[۱۵]

فرضیه بلیط بخت آزمایی[۱۱]

در ادامه در سال ۲۰۱۹ نظریه جدید در حوزه هرس شبکه مطرح شد، با نام فرضیه بلیط بخت‌آزمایی یا به اختصار LTH، با این مضمون که ؛شبکه عصبی متراکمی که به طور تصادفی راه‌اندازی شده است، شامل حداقل یک زیر شبکه است، که اگر به صورت جداگانه آموزش داده شود، می‌تواند با دقت تست شبکه اصلی هرس شده مطابقت داشته باشد[۱]. در واقع با این نظریه، عملا آموزش شبکه‌های پر از پارامتر، بی مفهوم گردید؛ زیرا فقط کافیست یک زیر شبکه متناسب پیدا شود و آن زیر شبکه آموزش داده شود. پس از این دوره بیشتر پژوهش‌ها به سمت یافتن زیر شبکه‌های مناسب جهت یافت (یافتن بلیط بخت‌آزمایی یا زیر شبکه‌ی هدف). اغلب روش‌های شناسایی بلیط هرس تکراری و مبتنی بر مقدار[۱۲] هستند، که روندی بسیار پرهزینه محسوب می‌شوند و بلیط‌ها قابل انتقال از یک شبکه به دیگری نیستند و برای هر شبکه متفاوت می‌باشند [۱۶].

در مطالعات بعدی همان تیم پژوهشی، مفاهیم LTH را بسط داده و مطرح کردند اگر سراغ شبکه راه‌اندازی شده نرفته و به جای جدا کردن زیر شبکه مطلوب از شبکه راه‌اندازی شده تصادفی به سراغ شبکه آموزش دیده در مرحله R رفته و زیر شبکه مذکور از این شبکه هرس و جدا شود، نتیجه‌ی بهتری گرفته خواهد شد[۱۷]. در سال‌های بعد انواع دیگری برای LTH مثل RLTH یا RingL مطرح شد که در آنها امکان بازگرداندن پیوندهای هرس شده و چرخه‌ی هرس تکراری لحاظ شده است[۱۸].

فرضیه بلیط بخت‌آزمایی در پژوهش‌های بعدی به صورت نظری و دقیق‌تر مورد مطالعه قرار گرفت، تا جایی که سعی شد بدون هیچ آموزشی زیر شبکه هدف از شبکه اولیه جدا و هرس شود. این مفهوم همان هرس پیش از راه‌اندازی شبکه است که در ادامه مطرح شده است[۱۹]. اکثریت پژوهش‌ها در LTH روی داده‌های تصویر متمرکز هستند ولی این فرضیه در یادگیری تقویتی[۱۳] و پردازش زبان طبیعی[۱۴] هم مورد بررسی قرار گرفته است[۲۰].

هرس پیش از راه‌اندازی یا در آموزش‌های اولیه[۱۵]

رویکردی دیگر در هرس شبکه‌های عصبی که تقریبا از سال ۲۰۱۹ آغاز گردیده؛ هرس در زمان راه‌اندازی یا در آموزش‌های اولیه شبکه یا به اختصار PaI است، که می‌تواند عملکردی تقریبا معادل شبکه‌یمتراکم به ارمغان آورد. این رویکرد مزایای خاص خود را دارد؛ مانند صرفه جویی در منابع در زمان آموزش و تسریع عملکرد یافتن زیر شبکه‌ی هدف. در این روش، در واقع در مراحل اولیه و راه‌اندازی شبکه، بدون آموزش کامل شبکه، تنها با در نظر گرفتن بخش کوچکی از داده ها، به کمک الگوریتم ها و معیارهای محاسباتی خاص، زیر شبکه اصلی یا همان بلیط بخت آزمایی[۱۶] مشخص شده، باقی گره ها هرس می‌شوند و مراحل آموزش شبکه با تعداد گره ها و اتصالات کمتر و به صورت منسجم‌تر آغاز و انجام می‌پذیرد. درسال ۲۰۱۹ با روش SNIP هرس وزنهای کم اهمیت و کم تاثیر روی تابع زیان در مرحله‌ی پیش از راه‌اندازی شبکه مطرح شد[۳, ۲۱]. کمی بعد در سال ۲۰۲۰ روش GraSP مطرح گردید که بیشتر از اطلاعات تابع ضرر در هنگام راه‌اندازی اولیه استفاده می‌کند و هدف حفظ جریان گرادیان در شبکه را دارد[۲۲].

در نهایت در مقاله‌ای روش‌های فوق تحت عنوان برجستگی سیناپسی جمع بندی شده و اثبات گردید، جهت حفظ جریان داده ها در شبکه، به جای هرس تک مرحله‌ای و ایستا، هرس به صورت تکرار شونده ضروری است و روش SynFlow پیشنهاد شد، که بدون نیاز به داده‌ها بر اساس برجستگی سیناپسی[۱۷] و به صورت لایه‌ای شبکه را هرس می‌کند[۹].

در نهایت توسط همان تیم ارائه‌کننده LTH در سال ۲۰۲۰ مطرح گردید که کلیه روش‌های هرس اولیه و پیش از راه‌اندازی در مقابل

[11]The lottery ticket hypothesis (LTH)
[12]Iterative Magnitude-based Pruning (IMP)
[13]Reinforcement learning (RL)
[14]Natural language processing (NLP)
[15]Pruning at Initialization (PaI)
[16]LT
[17]Synaptic saliency

روش‌های تکرار شونده و مبتنی بر مقدار[18] که در LTH استفاده می‌شوند، جای کار بیشتری دارند و عملکرد پایین‌تری ارائه می‌نمایند[۴].

بحث جدیدتری در این حوزه نیز مطرح شده است، با عنوان آموزش بر مبنای هرس پویا[19] که بر آموزش زیر شبکه‌های پراکنده از ابتدا متمرکز است؛ در حالی که به طور پویا الگوهای اتصال را تغییر می دهد. ایده اولیه این رویکرد در سال ۲۰۱۸ مطرح شد[۲۳]. در سال‌های اخیر نیز کارهای زیادی در این حوزه انجام شده است. بحث RingL که پیش‌تر مطرح گردید، را در این گروه پژوهش ها نیز می‌توان جای داد یا در پژوهشی دیگر توزیع مجدد پارامترها کار شده[۲۴]، همچنین احتمال پیوند را نیز در شاخه هرس پویا در زمان آموزش دخیل و مورد بررسی قرار داده‌اند [۲۵].

تعریف فرمال هرس: آموزش شبکه‌های عصبی با پارامترها و بردار وزن w از طریق نزول گرادیان[20]، اساساً یک دنباله مدل تولید می‌کند که در نهایت به یک مدل با عملکرد مطلوب همگرا می‌شود[۲۶, ۲۷]. اگرروند آموزش شبکه در K مرحله۲۱ با بردار وزنی به صورت زیر در نظر گرفته شود:

$$\{\boldsymbol{w}^{(0)}, \boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \dots, \boldsymbol{w}^{(k)}, \dots, \boldsymbol{w}^{(K)}\} \qquad (۱)$$

مراحل آموزش شبکه و اصلاح بردار وزن ، مجموعه از بردارهای وزن را می‌سازد که در . روش‌های هرس سنتی روی بردار وزن Wk هرس انجام می‌دهند. در واقع بردار نهایی بعد از K مرحله آموزش را جهت هرس استفاده می‌کنند. روش انجام هرس را به صورت ضرب نقطه به نقطه یک بردار تنسور پوششی[22] دو دویی در بردار وزن تعریف می‌شود[۶, ۲۸]. بود یا نبود یک اتصال را با ضرب صفر یا یک در مقدار وزن اتصال می‌توان تعیین نمود.

اینکه تنسور پوششی یا به اختصار تنسور چگونه ساخته شود، مربوط به مباحث معیار هرس۲۳می‌باشد. ولی جهت ایجاد تنسور می‌بایست از یکی از بردارهای سری وزن استفاده کرد، پس تنسور پوششی را به صورت زیر می‌توان تعریف نمود.

$$\boldsymbol{m} = \mathrm{F}_1(\boldsymbol{w}^{(\boldsymbol{k1})}; \boldsymbol{D}) \qquad (۲)$$

D مخفف مجموعه داده آموزشی است، یعنی تابع تولید تنسور پوششی می‌تواند از داده‌های موجود برای کمک به تصمیم‌گیری در مورد تنسورها استفاده کند. پس شبکه هرس شده را می‌توان حاصل ضرب نقطه به نقطه یکی از بردارهای وزن شبکه در بردار تنسور پوششی هرس دانست.. به این ترتیب شبکه هرس شده را می‌توان به فرمت زیر تعریف کرد (حاصل ضرب بردار تنسور پوششی تولیدی در یکی از بردارهای تولیدی در طی روند آموزش شبکه- بردار پایه):

$$\boldsymbol{W} = \mathrm{F}_1(\boldsymbol{w}^{(\boldsymbol{k1})}; \boldsymbol{D}) \odot \mathrm{F}_2(\boldsymbol{w}^{(\boldsymbol{k2})}; \boldsymbol{D}) \qquad (۳)$$

W بردار نهایی وزن‌های شبکه پس از هرس را نمایش می‌دهد. F1 بیان‌کننده تابع استخراج‌کننده‌ی بردار وزن مورد استفاده جهت اعمال هرس است و F2 تابع تولید تنسور جهت هرس می‌باشد. همان‌طور که مشخص است توابع تولیدکننده تنسور و بردار وزن از یکی از مجموعه بردارهای وزن و مجموعه داده‌های آموزشی استفاده می‌کنند. بنابر تعریف فرمال هرس که بیان شد و اینکه بردار پایه از کدام مرحله در روند آموزش و استنتاج اخذ شود، سه نوع هرس سنتی، LTH و PaI را می‌توان به صورت زیر تعریف نمود:

Traditional pruning : k1 = k2 = K;

Lottery ticket hypothesis (LTH)[1]: k1 = K, k2 = 0 and F2 = I (denoting the identity function)

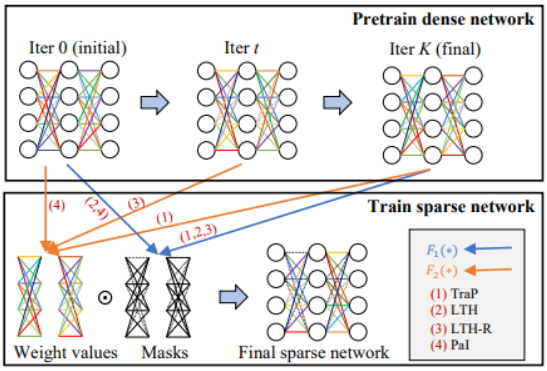
Lottery ticket rewinding (LTH-R)[17]: k1 = K, k2 = t and F2 = I;

Pruning at Initialization (PaI)[3, 9, 22]: k1 = k2 = 0

در مواردی که K1=0 است، مقدار دهی مجدد بردار شبکه باعث کاهش دقت در عملکرد شبکه هرس شده و برای رسیدن به دقت مناسب می‌بایست از بردار وزن اولیه استفاده شود تا دقت متناسبی را در عملکرد شبکه شاهد باشیم [۲۹].

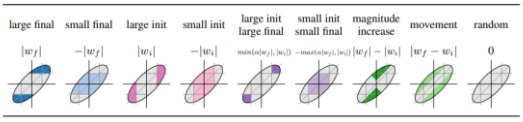
تنسور پوششی هرس از بردار وزن شبکه استفاده می‌کند. می‌توان شروع عملیات هرس را با بردار تنسور یک (برداری با کلیه عناصر یک در بردار وزن) در نظر گرفت، که پیش از آموزش شبکه در بردار وزن ضرب می‌شود و بعد از آموزش، بردار تنسور جدید ساخته شده و در مقادیر بردار وزن حاصل از یکی از مراحل آموزش شبکه (بنا بر نوع تکنیک هرس – بردار وزن اولیه یا نهایی) ضرب می‌گردد[۳۰].

[18]IMP
[19]Dynamic sparse training (DST)
[20]Stochastic Gradient Descent (SGD)
[21]Iteration
[22]Mask tensor
[23]pruning structure



(شکل-۴): عملیات هرس در دو زیر بخش، بدست آوردن تنسورها و مقادیر وزن طبقه بندی می‌شود .اینکه بردار وزن و بردار تنسور از کدام بردار وزن در طی، فرآیند آموزش شبکه استخراج شوند، تعیین کننده‌ی نوع هرس است. (*) F1 بردار تنسور پوششی به رنگ آبی و (*) F2 بردار وزن به رنگ نارنجی[۶]

نحوه ساخت تنسور از روی بردار وزن، در پژوهش‌های مختلفی مطرح شده، در زیر نه تنسور پوششی مختلف نمایش داده می‌شود، که هر کدام کاربرد خاصی دارند و در معماری‌های مختلف به کار برده می‌شوند[۳۰].



(شکل-۵): وزن‌هایی با بیشترین امتیاز (مناطق رنگی) حفظ می‌شوند و وزن‌هایی که کمترین امتیاز را دارند (مناطق خاکستری) هرس می‌شوند. محور x در هر شکل wi و محور y, wf است

وزن‌هایی که هرس می‌شوند، در واقع با ضرب نقطه به نقطه در بردار تنسور، صفر می‌شوند و کنار گذاشته می‌شوند. می‌توان وزن‌هایی که می‌بایست هرس شوند را صفر کرد یا فریز نمود. با فریز کردن وزن، وزن همان مقدار اولیه در طول آموزش باقی می‌ماند، این کار سبب می‌شود جریان صفر شدن مقادیر وزن در سطح شبکه کندتر شود. حالت دیگری نیز می‌توان در نظر گرفت: در صورت تغییر مقدار وزن در طی روند آموزش به سمت صفر، وزن مربوطه در هرس بعدی صفر شود و در صورت فاصله گرفتن وزن مربوطه از صفر، در مرحله بعدی اتصال فریز شود[۳۰].

به مرور پژوهش‌ها در این جهت حرکت کرد؛ که اگر تنسور پوششی از ابتدا و پیش از آموزش در بردار وزن ضرب شود، عملا هم راستا با آموزش شبکه و با تمرکز روی وزن‌ها و گره‌های پر اهمیت شبکه، شبکه آموزش و استنتاج خواهد شد. این ایده بیان می‌کند که ممکن است یافتن تنسور پوششی مناسب جهت هرس از آموزش کامل شبکه کم هزینه‌تر باشد. همین ایده سبب شد تا یافتن سوپر تنسورهای پوشش دهنده اتصالات مورد توجه قرار گیرد، یعنی تنسورهایی که از ابتدا خوب عمل می‌کنند و هم راستا با آموزش شبکه هستند. اعمال سوپر تنسورها به شبکه عصبی را می‌توان مثل استفاده از تکنیک هرس پیش از آموزش شبکه دانست، با این تفاوت که سوپر تنسورها همیشه به صورت تک مرحله‌ای به شبکه اعمال می‌شوند و بعد از آنها عملا آموزشی روی شبکه صورت نمی‌پذیرد و دقت و عملکرد پایین‌تری دارند[۳۰].

با ایجاد ایده تعریف سوپر تنسورها، تمایل به هرس پیش از آموزش شبکه بیشتر شد. به نحوی که هرس در زمان راه‌اندازی یا در آموزش‌های اولیه شبکه انجام گردد[۳, ۹, ۲۲]. هرس پیش از آموزش شبکه را می‌توان شامل دو بخش کلی دانست: امتیاز دهی به پارامترهای شبکه (متناسب با الگوریتم مد نظر) و حذف پارامترها با توجه به امتیاز آنها (حذف یا نگه داشتن پارامترها) [۹].

هرس‌های پیش از راه‌اندازی شبکه عموما تک مرحله‌ای هستند. در صورتی که از تکنیک هرس سراسری همزمان با آن استفاده شود، یک چالش اساسی رخ می‌دهد به نام فروپاشی لایه‌ای[۲۴] .در فروپاشی لایه‌ای صحت و دقت به یکباره دچار افت شدید می‌شود. در این حالت در واقع جریان داده‌ای در شبکه قطع می‌شود و شبکه عملا آموزش پذیری خود را از دست می‌دهد، رفع این چالش در کنار حفظ مزایای هرس تک مرحله‌ای و سراسری می‌تواند بحث چالش‌برانگیزی را فراهم نماید[۹].

در بررسی‌های بیشتر مشخص شد، تکنیک‌های هرس تکراری مبتنی بر مقدار مثل LTH عملکرد بهتری دارند و تکنیک‌های هرس پیش از راه‌اندازی شبکه، حتی اگر به صورت تکرار شونده اجرا شوند، با توجه به اعمال روی داده‌های محدود، دقت پایین‌تری دارند و همچنان جای بررسی و پیشرفت دارند[۴].

هرس پیش از راه‌اندازی[۲۵]

اگر هرسی با عملکرد مناسب و قابل رقابت با شبکه اصلی ، انجام شود، در این صورت می‌توان هرس پیش از راه‌اندازی را، همواره در مرحله‌ی راه‌اندازی شبکه می‌توان لحاظ کرد و به صورت هدفمند و بدون درگیری با اتصالات و گره‌های کم تاثیر در روند آموزش، فرآیند آموزش و استنتاج شبکه را پیش برد[۹]. همین علت سبب شده است تا پژوهشگران شانس خود را در ارایه الگوریتمی خوب جهت هرس پیش از راه‌اندازی و آموزش کامل شبکه امتحان نمایند و به این ترتیب

[24]Layer-collapse

[25]Pruning at Initialization (PaI)

این موضوع به یکی از بخش‌های پر چالش در حوزه فشرده‌سازی شبکه‌های عصبی مبدل گردیده است. در ادامه روش‌های مختلف هرس پیش از آموزش شبکه مطرح شده است. در ابتدا ایده هریک از روش‌های انتخاب کاندیدهای هرس شدن مطرح و سپس برای هر یک از روش‌های مطرح شده تعریف فرمال ارائه شده است.

روش SNIP[26] اولین روش هرس از نوع پیش از آموزش شبکه است، که از یک معیار برجستگی به نام حساسیت اتصال برای انتخاب اتصالات استفاده می‌کند و بر اساس یک ایده ساده یعنی حذف وزن‌هایی که فقدان آنها منجر به کمترین تغییر روی تابع ضرر می‌گردد، بهره می‌برد [۳]. این هرس به صورت تک مرحله‌ای پیش از آموزش کامل شبکه، صورت می‌پذیرد و تنها تاثیر بود یا نبود یک گره یا یک اتصال را بر روی میزان کل تغییر تابع زیان مد نظر قرار می‌دهد.

روش GraSP[27] تاثیر همزمان وجود یا عدم وجود گره‌ها و ارتباطات را از طریق محاسبه ماتریس هسین بر روی تابع ضرر محاسبه می‌کند و از این جهت می‌تواند بهتر از SNIP عمل نماید[۲۲].

روش Taylor-FO به صورت تکرار شونده با تاکید بر میزان تاثیر روی تابع ضرر به توان دو، تنسور پوشاننده هرس را تولید و شبکه را هرس می‌نماید. این روش حساسیتی به تغییرات در روند جریان داده در لایه‌ها ندارد[۳۱] .

روش SynFlow ضمن بهره برداری از نقاط ضعف روش‌های قبل، سعی می‌کند با تاکید بر حفظ جریان داده ها در شبکه از فروپاشی لایه‌ای[28] جلوگیری کند. این روش با محاسبه معیار هرس به صورت تکرار شونده و اعمال تاثیر حذف هر ارتباط در محاسبه مجدد معیار هرس و تعیین حد آستانه جهت جلوگیری از نزدیک شدن به فروپاشی لایه‌ای، هرس پیش از آموزش کامل شبکه را به انجام می‌رساند[۹].

روش SynFlowL2 نوع ارتقا یافته‌ی روش SynFlow است، که همان روش SynFlow را در هسته‌ی مسیر شبکه دنبال می‌کند [۳۲].

فرمول پایه معیار هرس در تمامی این روش ها را می‌توان در فرمول زیر خلاصه نمود:

$$S(\theta) = \frac{\delta R}{\delta \theta} \odot \theta \qquad (۴)$$

R تابع ضرر برای خروجی y از شبکه با پارامتر θدر صورت آموزش شبکه است. بنابراین برای SNIP معیار به صورت زیر فرمول بندی می‌گردد:

$$S_{SNIP} = \left| \frac{\delta R}{\delta \theta} \odot \theta \right| \qquad (۵)$$

و برای GraSP معیار هرس طبق فرمول (۶) قابل بیان است:

$$S_{GraSP} = -\left(H \frac{\delta R}{\delta \theta} \right) \odot \theta \qquad (۶)$$

برای Taylor-FO نیز معیار را می‌توان طبق فرمول (۷) بیان نمود:

$$S_{Taylor-FO} = \left(\frac{\delta R}{\delta \theta} \odot \theta \right)^2 \qquad (۷)$$

برای SynFlow تعریف تابع ضرر کمی متفاوت است و تابع ضرر تعریف خاص خود را دارد . معیار هرس در این روش به شرح زیر است:

$$S_{SynFlow} = \left(\frac{\delta R_{SF}}{\delta \theta} \odot \theta \right) \qquad (۸)$$

برای SynFlowL2 تعریف تابع ضرر در هسته مسیر تعریف می‌گردد و معیار کلی آن به شرح زیر است:

$$S_{SynFlowL2} = \left(\frac{\delta R_{PK}}{\delta \theta} \odot \theta \right) \qquad (۹)$$

هرس بر مبنای مسیر: شاخه‌ای از کارهای اخیر بر روی خواص همگرایی و تعمیم شبکه‌های عصبی عمیق با استفاده از تقریبهای خطی و آموزش پویا تمرکز دارد، که با عنوان هسته مماس عصبی[29] شناخته می‌شود. این مدلسازی از شبکه، بر پویایی دقیق خروجی‌های شبکه‌های عریض با عرض بی نهایت از طریق آموزش شبکه منطبق بر نزول گرادیان هسته در فضای تابع تاکید دارد[۳۳].

به صورت فرمال و ریاضی ؛ برای شبکه عصبی $\mathcal{F}$ با پارامترهای $\theta \in \mathbb{R}^m$ ، K گره خروجی و تابع ضرر $\ell : \mathbb{R}^K \times \mathbb{R}^K \rightarrow \mathbb{R}$ به صورتی است که:

$$L = \sum_{(x \in X, y \in Y)} \ell \left(\mathcal{F}_t(x, \theta), y \right) \qquad (۱۰)$$

Y برچسب خروجی و X دسته‌های آموزشی می‌باشد.

[26]Single-shot Network Pruning based on Connection Sensitivity
[27]Gradient Signal Preservation
[28]Layer-collapse
[29]Neural tangent kernel (NTK)

به ازای ورودی‌های شبکه

$$\mathcal{F}_t(x,\theta) \in \mathbb{R}^{NK} \quad (11)$$

که N تعداد نمونه‌های مرحله آموزش($x \in \mathbb{R}^N$) و K ابعاد خروجی شبکه است. در نزول گرادیان یه صورت پیوسته، تکامل پارامترها و خروجی‌ها را به صورت زیر می‌تواند بیان نمود:

$$\dot{\theta}_t = -\eta \nabla_\theta \mathcal{F}(x,\theta_t)^T \nabla_{\mathcal{F}(x,\theta_t)} L \quad (12)$$

$$\begin{aligned}\dot{\mathcal{F}}(x,\theta_t) &= \nabla_\theta \mathcal{F}(x,\theta_t)\dot{\theta}_t \\ &= -\eta \Theta_t(X,X)\nabla_{\mathcal{F}(x,\theta_t)} L\end{aligned} \quad (13)$$

ماتریس $\Theta_t(X,X) \in \mathbb{R}^{NK} \times \mathbb{R}^{NK}$ هسته مماس در زمان t است که به صورت ساختار کوواریانس پارامترها در سطح تمام نمونه‌های آموزشی تعریف می‌شود[۳۳].

$$\Theta_t(X,X) = \nabla_\theta \mathcal{F}(x,\theta_t)\nabla_\theta \mathcal{F}(x,\theta_t)^T \quad (14)$$

اگرvp حاصل ضرب وزن‌های θ باشد که در مسیر p قرار دارند. تعریف فرمال v به صورت زیر می‌شود:

$$v_p(\theta) = \prod_{j=1}^{m} \theta_j^{pj} \quad (15)$$

طبق قانون مشتق زنجیره‌ای، $\nabla_\theta \mathcal{F}(x,\theta)$را به صورت زیر می‌توان تعریف نمود:

$$\begin{aligned}\nabla_\theta \mathcal{F}(x,\theta) &= \frac{\partial \mathcal{F}(x,\theta)}{\partial \theta} \\ &= \frac{\partial \mathcal{F}(x,\theta)}{\partial v(\theta)} \frac{\partial v(\theta)}{\partial \theta} \\ &= J_v^{\mathcal{F}}(x)\, J_\theta^v\end{aligned} \quad (16)$$

که $\frac{\partial \mathcal{F}(x,\theta)}{\partial v(\theta)} = J_v^{\mathcal{F}}(x) \in \mathbb{R}^{K\times P}$ و $\frac{\partial v(\theta)}{\partial \theta} = J_\theta^v \in \mathbb{R}^{P\times m}$ می‌باشد .

پس $\Theta(X,X)$ یا هسته مماس را می‌توان به صورت زیر باز تعریف نمود:

$$\begin{aligned}\Theta(X,X) &= \nabla_\theta \mathcal{F}(x,\theta)\nabla_\theta \mathcal{F}(x,\theta)^T \\ &= J_v^{\mathcal{F}}(x)\, J_\theta^v \left(J_\theta^v\right)^T J_v^{\mathcal{F}}(x)^T \\ &= J_v^{\mathcal{F}}(x)\quad \Pi_\theta \quad J_v^{\mathcal{F}}(x)^T\end{aligned} \quad (17)$$

Π_θ را هسته‌ی مسیر[۳۰] می‌نامند. که در روش SynFlowL2 تابع ضرر در این هسته مسیر مشخص و بر اساس الگوریتم مربوطه هرس انجام می‌شود[۳۲].

با این مدلسازی، روش‌های هرس مطرح شده را می‌توان باز تعریف نمود، که به تفصیل در مقاله‌های مختلف این مهم صورت پذیرفته است. برای مثال در هرس به روش SNIP معیار هرس روی تغییرات تابع زیان مورد متمرکز است :

$$S_{SNIP}(\theta) = \frac{\partial L}{\partial \theta} \odot \theta \quad (18)$$

می‌توان $\frac{\partial L}{\partial \theta}$ را به صورت زیر جایگزین نمود و در واقع این روش را بر اساس هسته مسیر باز نویسی کرد:

$$\begin{aligned}S_{SNIP}(\theta) &= \frac{\partial L(\mathcal{F}(X,\theta),Y)}{\partial \theta} \\ &= \frac{\partial L(\mathcal{F}(X,\theta),Y)}{\partial \mathcal{F}(X,\theta)} \frac{\partial \mathcal{F}(X,\theta)}{\partial v(\theta)} \frac{\partial v(\theta)}{\partial \theta} \\ &= \frac{\partial L(\mathcal{F}(X,\theta),Y)}{\partial \mathcal{F}(X,\theta)} J_v^{\mathcal{F}}(x)\, J_\theta^v\end{aligned} \quad (19)$$

همانطور که مشخص است روش SNIP تعامل بین مسیرها و تابع ضرر را مد نظر دارد و مشخصا J_θ^v نسبت به مسیرهایی که ارزش بالایی دارند و پارامترهای بزرگتری را دارند و روی خطا تاثیر می‌گذارند، حساس‌تر است.

[30] Path kernel

روش GraSP به حفظ جریان گرادیان تاکید دارد. مشخصا پس از هرس و کاهش پارامتر، اتصالات پراکنده ایجاد می‌شود که جلوی جریان گرادیان در شبکه را می‌گیرد. این روش حفظ جریان گرادیان را با ایده اضافه کردن یک آشفتگی کوچک δ به وزن‌های اولیه پیاده سازی می‌کند و سپس از یک تقریب تیلور برای مشخص کردن چگونگی تاثیر حذف یک اتصال بر جریان گرادیان پس از هرس، استفاده می‌کند.

$$\begin{aligned} \boldsymbol{S}(\boldsymbol{\delta}) &= \Delta \boldsymbol{L}(\boldsymbol{\theta_0} + \boldsymbol{\delta}) - \Delta \boldsymbol{L}(\boldsymbol{\theta_0}) \\ &= 2\boldsymbol{\delta}^{\boldsymbol{T}} \boldsymbol{\nabla}^{\boldsymbol{2}} \boldsymbol{L}(\boldsymbol{\theta_0}) \boldsymbol{\nabla} \boldsymbol{L}(\boldsymbol{\theta_0}) \\ &\quad + \boldsymbol{\mathcal{O}}(\| \boldsymbol{\delta} \|_2^2) \\ &= 2\boldsymbol{\delta}^{\boldsymbol{T}} \boldsymbol{H} \boldsymbol{g} + \boldsymbol{\mathcal{O}}(\| \boldsymbol{\delta} \|_2^2) \end{aligned} \qquad (۲۰)$$

اگر S(δ) منفی باشد، حذف وزن‌های مربوطه باعث کاهش جریان گرادیان می‌شود، در حالی که اگر مثبت باشد، جریان گرادیان را افزایش می‌دهد. ترجیح بر حذف ابتدایی وزن‌هایی است که حذف آنها باعث کاهش جریان گرادیان نمی‌شود. پس برای هر وزن، اهمیت را می‌توان به روش زیر محاسبه کرد:

$$\boldsymbol{S}(\boldsymbol{\theta}) = \boldsymbol{\theta} \odot \boldsymbol{H} \boldsymbol{g} \qquad (۲۱)$$

در این روش ابتدا امتیازات را محاسبه کرده، بر اساس نرخ هرس P امتیاز پایین هرس می‌شود. الگوریتم بیان شده در ادامه آورده شده است .

Algorithm 2 SNIP: Single-shot Network Pruning based on Connection Sensitivity

Require: Loss function L, training dataset D, sparsity level K

Ensure: $\|W\|_0 \leq K$

1. $W \leftarrow$ VarianceScalinglnitialization
2. D^b sample a mini batch of training data
3. S_j connection sensivity
4. $\hat{S} \leftarrow$ sortDescending(s)
5. $C_j \leftarrow 1[S_j - \hat{S} \geq 0], j \in \{1 \dots m\}$
6. W^* Pruning: choose top k connection
7. $W^* \leftarrow C \odot W^*$

(شکل-۶): هرس به روش **SNIP** [۳]

Algorithm 3 Hessian-gradient Product.

Require: A batch of training data D_b, network f with initial parameters Θ_0, loss function L

1. $L(\Theta_0)$ Compute the loss and build the computation graph
2. g = grad($L(\Theta_0)$, Θ_0) Compute the gradient Of loss function With respect to Θ_0
3. Hg = grad(g^T stop_grad(g), Θ_0)
4. Return Hg

(شکل-۷): محاسبه امتیاز بر اساس هیسین [۲۲]

Algorithm 4 Gradient Signal Preservation (GraSP).

Require: Pruning ratio p, training data D, network f with initial parameters Θ_0

1. D_b Sample a collection of training examples
2. Compute the Hessian-gradient product Hg (Algorithm3)
3. $S(-\Theta_0) = -\Theta_0 \odot Hg$ Compute the importance of each weight
4. Compute P_{th} percentile of $S(-\Theta_0)$ as T
5. $M =$ of $S(-\Theta_0) < T$ Remove the weights with smallest importance
6. Train the network $f_m \odot$ on D until convergence.

(شکل-۸): هرس به روش **GraSP** [۲۲]

(جدول-۱): مقایسه روش‌های هرس

روش هرس	نوع	مزایا	معایب
سنتی	سنتی ،پس از آموزش شبکه، آموزش مجدد شبکه بعد از هرس جهت حفظ دقت ،استفاده از مقادیر آموزش دیده پارامترهای شبکه	قابلیت تفسیر بالا،ساده ، دقت بالا	هزینه بالا، نبود هرس در مرحله آموزش
LTH	بلیط بخت آزمایی،پس از آموزش شبکه، استفاده از مقادیر راه‌اندازی پارامترهای شبکه بدون هیچ آموزشی	قابلیت تفسیر بالا ، دقت بالا	هزینه بالا در یافتن LT، نبود هرس در مرحله آموزش
SNIP	PaI ، پیش از آموزش شبکه، استفاده از مقادیر راه‌اندازی پارامترهای شبکه بدون هیچ آموزشی، تک مرحله‌ای	بهینه سازی وسیع، هرس در مرحله آموزش کامل شبکه	دقت پایین
GraSP	PaI ، پیش از آموزش شبکه، استفاده از مقادیر راه‌اندازی پارامترهای شبکه بدون هیچ آموزشی	بهینه سازی وسیع، هرس در مرحله آموزش کامل شبکه	دقت پایین
SynFlow	PaI ، پیش از آموزش شبکه، استفاده از مقادیر راه‌اندازی پارامترهای شبکه بدون هیچ آموزشی، تکرار شونده، دارای حد فشرده‌سازی جهت جلوگیری از فرو پاشی لایه‌ای	بهینه سازی وسیع، هرس در مرحله آموزش کامل شبکه	دقت بیشتر نسبت به سایر روش‌های PAI و کمتر از MIG

۳-نتیجه گیری

هدف اصلی در روند هرس صرفه جویی در منابع محاسباتی، صرفه جویی درحافظه و تسریع سرعت تولید خروجی می‌باشد، به گونه‌ای که کارایی حفظ شود، صحت و دقت معادل شبکه اولیه و پیش از هرس داشته باشیم و سربار محاسباتی موازی در روند هرس ایجاد نشود (زمان محاسبه شبکه هرس شده معادل زمان آموزش شبکه عمیق نشود. عملا صرفه جویی محاسباتی و زمانی رخ ندهد.). البته در برخی منابع یاد شده است که حتی اگر برای آموزش و استنتاج شبکه هرس شده معادل با شبکه اصلی منابع و زمان صرف شود، باز هم هرس منطقی به نظر می‌رسد، زیرا سادگی زیادی به شبکه اعمال می‌گردد و شبکه تفسیر پذیرتر می‌شود، در حالی که عملکرد شبکه عمیق را همچنان داریم. هدف نهایی از هرس پیش از آموزش می‌تواند اضافه شدن هرس به عنوان یکی از مراحل راه‌اندازی شبکه عصبی باشد. در صورتی که بتوان با عملکرد مناسب و قابل رقابت با شبکه اصلی، هرس انجام داد؛ می‌توان هرس در زمان راه‌اندازی را همواره در مرحله راه‌اندازی شبکه لحاظ کرد و از مزایای آن بهره برد.

در نهایت اگر بتوان فشرده‌سازی را به نتیجه رساند، امکان استفاده از شبکه‌های عصبی در هر بستری فراهم می‌گردد. روند آموزش شبکه‌های عصبی هدفمندتر دنبال می‌گردد و دیگر زمان صرف آموزش اتصالات کم اهمیت در طی روند آموزش نمی‌شود و در هر دو مرحله آموزش و استنتاج حفظ منابع صورت می‌پذیرد.

۴-مراجع

[1] *Frankle, J. and M. Carbin, The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv preprint arXiv:1803.03635, 2018.*

[2] *Deng, L., et al., Model compression and hardware acceleration for neural networks: A comprehensive survey. Proceedings of the IEEE, 2020. 108(4): p. 485-532.*

[3] *Lee, N., T. Ajanthan, and P.H. Torr, Snip: Single-shot network pruning based on connection sensitivity. arXiv preprint arXiv:1810.02340, 2018.*

[4] *Frankle, J., et al., Pruning neural networks at initialization: Why are we missing the mark? arXiv preprint arXiv:2009.08576, 2020.*

[5] *Ramanujan, V., et al. What's hidden in a randomly weighted neural network? in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.*

[6] *Wang, H ,.et al., Emerging paradigms of neural network pruning. arXiv preprint arXiv:2103.06460, 2021.*

[7] *Mao, H., et al. Exploring the granularity of sparsity in convolutional neural networks. in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2017.*

[8] *Liu, Z., et al. Learning efficient convolutional networks through network slimming. in Proceedings of the IEEE international conference on computer vision. 2017.*

[9] *Tanaka, H., et al., Pruning neural networks without any data by iteratively conserving synaptic flow. arXiv preprint arXiv:2006.05467, 2020.*

[10] *He, Y., et al. Amc: Automl for model compression and acceleration on mobile devices. in Proceedings of the European conference on computer vision (ECCV). 2018.*

[11] *Lange, R.T. The Lottery Ticket Hypothesis: A Survey 2020; Available from: https://roberttlange.github.io/posts/2020/06/lottery-ticket-hypothesis/#yu_2019.*

[12] *LeCun, Y., J.S. Denker, and S.A. Solla. Optimal brain damage. in Advances in neural information processing systems. 1990.*

[13] *Hassibi, B. and D.G. Stork, Second order derivatives for network pruning: Optimal brain surgeon. 1993: Morgan Kaufmann.*

[14] *Han, S., et al., Learning both weights and connections for efficient neural networks. arXiv preprint arXiv:1506.02626, 2015.*

[15] *Blalock, D., et al., What is the state of neural network pruning? arXiv preprint arXiv:2003.03033, 2020.*

[16] *Chen, X., et al., The elastic lottery ticket hypothesis. Advances in Neural Information Processing Systems, 2021. 34.*

[17] *Frankle, J., et al., Stabilizing the lottery ticket hypothesis. arXiv preprint arXiv:1903.01611, 2019.*

[18] *Evci, U., et al. Rigging the lottery: Making all tickets winners. in International Conference on Machine Learning. 2020. PMLR.*

[19] *Malach, E., et al. Proving the lottery ticket hypothesis: Pruning is all you need. in International Conference on Machine Learning. 2020. PMLR.*

[20] Yu, H., et al., Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. arXiv preprint arXiv:1906.02768, 2019.
[21] Lee, N., et al., A signal propagation perspective for pruning neural networks at initialization. arXiv preprint arXiv:1906.06307, 2019.
[22] Wang, C., G. Zhang, and R. Grosse, Picking winning tickets before training by preserving gradient flow. arXiv preprint arXiv:2002.07376, 2020.
[23] Mocanu, D.C., et al., Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. Nature communications, 2018. 9(1): p. 1-12.
[24] Liu, S., et al., Selfish sparse RNN training. arXiv preprint arXiv:2101.09048, 2021.
[25] Liu, S., et al., Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. arXiv preprint arXiv:2102.02887, 2021.
[26] Bottou, L., Large-scale machine learning with stochastic gradient descent, in Proceedings of COMPSTAT'2010. 2010, Springer. p. 177-186.
[27] Robbins, H. and S. Monro, A stochastic approximation method. The annals of mathematical statistics, 1951: p. 400-407.
[28] Hayou, S., et al ,.Robust pruning at initialization. arXiv preprint arXiv:2002.08797, 2020.
[29] Renda, A., J. Frankle, and M. Carbin, Comparing rewinding and fine-tuning in neural network pruning. arXiv preprint arXiv:2003.02389, 2020.
[30] Zhou, H., et al., Deconstructing lottery tickets: Zeros, signs, and the supermask. arXiv preprint arXiv:1905.01067, 2019.
[31] Molchanov, P., et al. Importance estimation for neural network pruning. in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.
[32] Gebhart, T., U. Saxena, and P. Schrater, A Unified Paths Perspective for Pruning at Initialization. arXiv preprint arXiv:2101.10552, 2021.
[33] Jacot, A., F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks. arXiv preprint arXiv:1806.07572, 2018.

دکتر محبوبه شمسی مدرک کارشناسی خود را در رشته ریاضی از دانشگاه اصفهان ، کارشناسی ارشد خود را در رشته مهندسی نرم افزار از دانشگاه آزاد اسلامی نجف آباد، ایران و دکتری خود را در رشته مهندسی نرم افزار از دانشگاه UTM Technology مالزی، به ترتیب در سال های ۲۰۰۳، ۲۰۰۶ و ۲۰۱۱ کسب نموده است. وی در حال حاضر استادیار دانشکده مهندسی برق و کامپیوتر دانشگاه صنعتی قم است .علایق تحقیقاتی او پردازش تصویر، محاسبات نرم، رایانش ابری و پایگاه داده مرکزی و توزیع شده است.

عطیه فیروزه مدرک کارشناسی خود را درسال ۱۳۹۸ از دانشگاه الزهرا در رشته مهندسی کامپوتر گرایش نرم افزار و دوره کارشناسی ارشد خود را در سال ۱۳۹۹ در دانشگاه صنعتی قم در رشته مهندسی کامپوتر گرایش نرم افزار آغاز نموده است .زمینه ای پژوهشی مورد علاقه ایشان عبارتند از: شبکه های عصبی عمیق و یادگیری ماشین، فشرده سازی شبکه های عصبی عمیق

دکتر مرتضی محجل کفشدوز، کارشناسی ارشد و دکتری خود را به ترتیب در سالهای ۱۳۹۰ و ۱۳۹۵ از دانشگاه صنعتی شریف در رشته معماری کامپیوتر اخذ کرده است. وی در حال حاضر استادیار دانشکده مهندسی برق و کامپیوتر دانشگاه صنعتی قم است . علایق تحقیقاتی ایشان سامانه های نهفته بی درنگ، اینترنت اشیاء و یادگیری ماشین است.

روش ارجاع به مقاله : ع. فیروزه، م. محجل، م. شمسی، مروری بر روش‌های هرس در شبکه‌های عصبی عمیق با تاکید بر روش‌های هرس پیش از آموزش. دوفصلنامه محاسبات و سامانه های توزیع شده. سال چهارم، شماره اول، شماره پیاپی ۷، صفحه ۹۰ تا ۱۰۰، سال ۱۴۰۰.

How to cite: Atieh firoozeh. Morteza.Mohajel, Mahbobeh.Shamsi. Implementing a Smart Refrigerator in IoT-based House. Journal of Distributed Computing and Systems(JDCS), Vol 4, Issue 1, Page 90-100.