- مشکل گرادیان محو شونده یک مسئله رایج است که در شبکه های عصبی عمیق رخ می دهد. این زمانی اتفاق می‌افتد که شیب‌های تابع loss کوچک می‌شوند، زیرا در طول آموزش در لایه‌های شبکه منتشر می‌شوند، و در نهایت منجر به این می‌شود که وزن لایه‌های قبلی اصلاً به‌روز نمی‌شود. این می تواند باعث شود که این لایه ها اساساً "محو شوند" یعنی همانطور که لایه های بیشتری با استفاده از توابع فعال سازی خاص به شبکه های عصبی اضافه می شوند، گرادیان loss function به صفر نزدیک می شود و آموزش شبکه را سخت می کند. این به این دلیل است که برخی از توابع فعال سازی، مانند تابع سیگموئید، یک فضای ورودی بزرگ را به یک فضای ورودی کوچک بین ۰ و ۱ نظیر می کنند. بنابراین، تغییر زیاد در ورودی تابع سیگموئید باعث تغییر کوچکی در خروجی می شود. از این رو، مشتق کوچک می شود. برای شبکه های کم عمق با تنها چند لایه که از این فعال سازی ها استفاده می کنند، این مشکل بزرگی نیست. با این حال، زمانی که از لایه‌های بیشتری استفاده می‌شود، می‌تواند باعث شود که گرادیان برای اینکه بتواند آموزش به طور موثر کار کند بسیار کوچک باشد. گرادیان های شبکه های عصبی با استفاده از پس انتشار یافت می شوند. به زبان ساده، پس انتشار مشتقات شبکه را با حرکت لایه به لایه از لایه نهایی به لایه اولیه پیدا می کند. بر اساس قانون زنجیره، مشتقات هر لایه در شبکه ضرب می شوند (از لایه نهایی تا لایه اولیه) تا مشتقات لایه های اولیه را محاسبه کنند. با این حال، وقتی n لایه پنهان از یک فعال سازی مانند تابع سیگموئید استفاده می کنند، n مشتق کوچک با هم ضرب می شوند. بنابراین، با انتشار به لایه های اولیه، گرادیان به صورت تصاعدی کاهش می یابد. یک گرادیان کوچک به این معنی است که وزن ها و سوگیری های لایه های اولیه به طور موثر در هرtrain به روز نمی شوند. از آنجایی که این لایه های اولیه اغلب برای شناسایی عناصر اصلی داده های ورودی بسیار مهم هستند، می تواند منجر به عدم دقت کلی کل شبکه شود.

راه حل ها:

ساده ترین راه حل استفاده از توابع فعال سازی دیگر است، مانند ReLU، که مشتق کوچکی ایجاد نمی کند. لایه های عادی سازی دسته ای batch normalization layers نیز می توانند مشکل را حل کنند. همانطور که قبلا گفته شد، مشکل زمانی ایجاد می شود که یک فضای ورودی بزرگ به یک فضای کوچک نگاشت می شود و باعث ناپدید شدن مشتقات می شود. تکنیک‌های منظم‌سازی regularization techniques مانند انصراف dropout یا منظم‌سازی L1/L2 به نوبه خود می‌تواند به بهبود عملکرد شبکه و کاهش گرادیان‌های محوشونده کمک کند. راه حل دیگر استفاده از تکنیک های اولیه سازی وزن است که می تواند به اطمینان حاصل شود که گرادیان ها ناپدید نمی شوند. یکی از روش های رایج، مقداردهی اولیه Xavier نام دارد که وزن اولیه هر نورون را به گونه ای تنظیم می کند که به جلوگیری از کوچک شدن گرادیان ها کمک می کند.

- انفجار گرادیان مشکلی است که در یادگیری عمیق زمانی رخ می دهد که شیب ها در طول back propagation بیش از حد بزرگ می شوند. این می تواند باعث شود که وزن شبکه در هر تکرار به شدت به روز شود که می تواند منجر به بی ثباتی و عملکرد ضعیف شود. در شبکه‌های عمیق یا شبکه‌های عصبی مکرر، گرادیان‌های خطا می‌توانند در طول به‌روزرسانی جمع شوند و منجر به گرادیان‌های بسیار بزرگ شوند. اینها به نوبه خود منجر به به روز رسانی های بزرگ برای وزن شبکه و به نوبه خود یک شبکه ناپایدار می شود. در نهایت، مقادیر وزن‌ها می‌توانند آنقدر بزرگ شوند که سرریز شده و به مقادیر NaN منجر شوند. انفجار از طریق رشد نمایی با ضرب مکرر گرادیان در لایه های شبکه که مقادیر بزرگتر از ۱.۰ دارند، رخ می دهد. در شبکه‌های عصبی مکرر، شیب‌های انفجاری می‌تواند منجر به شبکه‌ای ناپایدار شود که قادر به یادگیری از داده‌های آموزشی نیست و در بهترین حالت شبکه‌ای که نمی‌تواند از طریق توالی‌های ورودی طولانی اطلاعات یاد بگیرد.

راه حل ها:

استفاده از برش گرادیان gradient clipping که شامل تعیین حداکثر آستانه برای گرادیان ها است. اگر گرادیان ها از این آستانه فراتر رفت، آنگاه آنها را کاهش می دهند تا باعث به روز شدن بیش از حد وزن ها نشوند. روش دیگر استفاده از نرمال سازی دسته ای batch normalization است که فعال سازی هر لایه را قبل از انتقال به لایه بعدی نرمالایز می کند. این می تواند به کاهش بزرگی گرادیان ها و بهبود پایداری شبکه کمک کند. همچنین می‌توان از الگوریتم‌های بهینه‌سازی مختلف مانند Adam یا Adagrad استفاده کرد، که برای مدیریت موثرتر انفجارهای گرادیان نسبت به روش‌های بهینه‌سازی سنتی طراحی شده‌اند. یکی از این روش‌ها تنظیم وزن weight regularization است که شامل اضافه کردن یک عبارت جریمه به تابع loss است که مانع از وزنه‌های بزرگ می‌شود.(L1 (absolute weights) or an L2 (squared weights) penalty)) این می تواند به جلوگیری از برازش بیش از حد شبکه کمک کند و شیب ها را تحت کنترل نگه دارد. در

شبکه‌های عصبی عمیق، گرادیان‌های انفجاری ممکن است با طراحی مجدد شبکه برای داشتن لایه‌های کمتر مورد بررسی قرار گیرند. همچنین ممکن است استفاده از اندازه دسته کوچکتر در حین آموزش شبکه مزایایی داشته باشد. در شبکه‌های عصبی مکرر، به‌روزرسانی در گام‌های کمتر زمانی قبلی در طول آموزش، که به آن truncated Backpropagation در طول زمان گفته می‌شود، ممکن است مشکل گرادیان انفجاری را کاهش دهد. شیب های انفجاری را می توان با استفاده از واحدهای حافظه بلند مدت (LSTM) و احتمالاً ساختارهای نورونی نوع دروازه ای مرتبط کاهش داد. اتخاذ واحدهای حافظه LSTM بهترین روش جدید برای شبکه های عصبی مکرر برای پیش بینی توالی است.

-۲

اگر مقدار عددی دقیق یک ویژگی مهم نیست و تأثیر آن فقط به محدوده ای که در آن قرار دارد بستگی دارد، می توانیم از مقیاس گذاری ویژگی برای تغییر مقادیر ویژگی استفاده کنیم. مقیاس‌بندی ویژگی تکنیکی است که برای استاندارد کردن دامنه ویژگی‌ها در یک مجموعه داده استفاده می‌شود، به طوری که اندازه‌های مشابهی داشته باشند.

روش های مختلفی برای مقیاس بندی ویژگی وجود دارد، از جمله نرمال سازی و استانداردسازی.

عادی سازی شامل مقیاس بندی مقادیر یک ویژگی در محدوده ای بین ۰ و ۱ با استفاده از فرمول است:

$$X' = (X - X\_min) / (X\_max - X\_min)$$

که در آن X مقدار جدید مقیاس شده، X مقدار اصلی، X_min حداقل مقدار ویژگی، و X_max حداکثر مقدار ویژگی است.

استانداردسازی شامل مقیاس بندی مقادیر یک ویژگی به گونه ای است که میانگین ۰ و انحراف معیار ۱ با استفاده از فرمول:

$$X' = (X - mean(X)) / std(X)$$

که در آن X مقدار جدید مقیاس شده، X مقدار اصلی، mean(X) میانگین ویژگی، و std(X) انحراف استاندارد ویژگی است.

نرمال سازی و استانداردسازی هر دو می توانند در تغییر مقادیر ویژگی موثر باشند به طوری که اندازه های مشابهی داشته باشند و راحت تر قابل مقایسه و تجزیه و تحلیل باشند. انتخاب روش مورد استفاده به نیازهای خاص مسئله و ماهیت مجموعه داده بستگی دارد.

-۳

مشکل مجموعه داده های نامتعادل، این است که مدل های یادگیری ماشینی می توانند نسبت به طبقه اکثریت (در این مورد، افراد سالم) تعصب داشته باشند. این به این دلیل است که مدل در معرض داده‌های طبقه اکثریت قرار می‌گیرد و ممکن است تشخیص صحیح طبقه اقلیت را نیاموزد. این می تواند منجر به عملکرد ضعیف در هنگام پیش بینی طبقه اقلیت شود، که اغلب طبقه مورد علاقه است (در این مورد، افراد مبتلا به بیماری).

برای رفع این مشکل می توان از چندین تکنیک استفاده کرد:

کم نمونه برداری: حذف تصادفی برخی از نمونه های کلاس اکثریت به طوری که مجموعه داده متعادل شود. این ممکن است منجر به از دست دادن اطلاعات شود، اما می تواند به مدل کمک کند تا طبقه اقلیت را تشخیص دهد.

Oversampling: ایجاد نمونه های مصنوعی از کلاس اقلیت برای افزایش نمایش آن در مجموعه داده. این را می توان با استفاده از تکنیک هایی مانند SMOTE (تکنیک نمونه برداری بیش از حد اقلیت مصنوعی) یا ADASYN (نمونه گیری مصنوعی تطبیقی) انجام داد. نمونه برداری بیش از حد می تواند از نظر محاسباتی گران باشد، اما می تواند به مدل کمک کند تا طبقه اقلیت را بهتر تشخیص دهد.

یادگیری حساس به هزینه: تغییر تابع ضرر الگوریتم یادگیری ماشین برای در نظر گرفتن عدم تعادل در مجموعه داده. این را می توان با اختصاص وزن های بالاتر به نمونه های کلاس اقلیت انجام داد.

روش‌های مجموعه: ترکیب چندین مدل یادگیری ماشین آموزش‌دیده بر روی نسخه‌های مختلف مجموعه داده (مثلاً با تکنیک‌های مختلف کم‌نمونه‌سازی یا بیش‌نمونه‌سازی) برای بهبود عملکرد کلی.

توجه به این نکته مهم است که انتخاب تکنیک به مجموعه داده‌ها و مسئله خاص در دست بستگی دارد، و ممکن است برای یافتن بهترین راه‌حل نیاز به آزمایش با رویکردهای مختلف باشد.

-۴

روش های مختلفی برای این منظور وجود دارد.

One-hot/dummy encoding

داده های طبقه بندی شده به صورت بردارهای صفر و یک نمایش داده می شوند. این کار با استفاده از یک متغیر ساختگی مجزا برای هر دسته انجام می شود و در صورتی که مشاهده متعلق به آن دسته باشد، مقدار متغیر ساختگی را ۱ و در غیر این صورت ۰ قرار می دهیم. برای داده های NOMINAL بسیار خوب و محبوب است.

Label / Ordinal encoding

این نوع رمزنگاری برای داده هایی که نسبت به هم اولویت دارند به کار میرود(همان مثال تهران ۱۰) و برای هر دسته یک عدد درنظر گرفته میشود.

سایر روش ها::

Frequency / count encoding-Binary Encoding-Feature Hashing-Target Encoding...و

با عدد دادن به شهر ها عملا داریم برتری ناخواسته ایجاد میکنیم و مدل را به اشتباه می اندازیم. چرا که مدل اعداد را بر اساس مقدارشان در نظر میگیرد نه به عنوان ایندکس. پس اگر م به تهران عدد ۱۰ بدهیم و به اردبیل عدد ۱ بدهیم و به همدان عدد ۳۲ بدهیم یعنی ارزش خانه ها در همدان بیشتر از تهران و بیشتر از اردبیل است که این درست نیست و حتی اگر بخواهیم با مفداردهی این کار را انجام دهیم باید بر اساس ملاک هایی مثل کلان شهر بودن، پایتخت بودن و... این اعداد داده شود. یعنی اگر همه موارد دو خانه یکسان باشند و تنها شهرشان فرق کند قیمت کدام بیشتر خواهد بود و بر این اساس شماره بدهیم.

-۵

جلوگیری از بیش‌برازش(overfitting) :

۱ Hold-out (data) .

به جای استفاده از تمام داده‌هایمان برای آموزش، می‌توانیم به سادگی مجموعه داده‌های خود را به دو مجموعه تقسیم کنیم: آموزش و تست. یک نسبت تقسیم معمول ۸۰ درصد برای آموزش و ۲۰ درصد برای تست است. ما مدل خود را تا زمانی آموزش می دهیم که نه تنها در مجموعه آموزشی بلکه برای مجموعه تست نیز عملکرد خوبی داشته باشد. این نشان دهنده قابلیت تعمیم خوب است زیرا مجموعه آزمایشی داده های دیده نشده را نشان می دهد که برای آموزش استفاده نشده اند. با این حال، این رویکرد به یک مجموعه داده به اندازه کافی بزرگ برای آموزش حتی پس از تقسیم نیاز دارد.

۲ Cross-validation (data) .

ما می‌توانیم مجموعه داده‌های خود را به k گروه تقسیم کنیم . (k-fold cross-validation) در این روش اجازه می دهیم یکی از گروه ها مجموعه تست باشد و بقیه به عنوان مجموعه آموزشی، و این روند را تا زمانی که هر گروه جداگانه به عنوان مجموعه تست استفاده شود) به عنوان مثال، kتکرار (تکرار شود. برخلاف Hold-out ، این روش اجازه می‌دهد تا در نهایت از تمام داده‌ها برای آموزش استفاده شود، اما همچنین از نظر محاسباتی گران‌تر از Hold-out است.

۳Data augmentation (data) .

یک مجموعه داده بزرگتر باعث کاهش بیش از حد بیش‌برازش می شود. اگر نمی‌توانیم داده‌های بیشتری جمع‌آوری کنیم و محدود به داده‌هایی هستیم که در مجموعه داده فعلی خود داریم، می‌توانیم افزایش داده‌ها (data augmentation) را برای افزایش مصنوعی اندازه مجموعه داده‌مان اعمال کنیم. به عنوان مثال، اگر در حال آموزش برای یک کار طبقه بندی تصویر هستیم، می توانیم تغییر شکل های تصویری مختلفی را در مجموعه داده های تصویر خود انجام دهیم (به عنوان مثال، چرخش، تغییر مقیاس، شیفت).

۴. Feature selection (data) .

اگر فقط تعداد محدودی از نمونه‌های آموزشی داریم که هر کدام دارای تعداد زیادی ویژگی هستند، باید فقط مهم‌ترین ویژگی‌ها را برای آموزش انتخاب کنیم تا مدل ما نیازی به یادگیری برای این همه ویژگی نداشته باشد. ما می‌توانیم به سادگی ویژگی‌های مختلف را آزمایش کنیم، مدل‌های فردی را برای این ویژگی‌ها آموزش دهیم، و قابلیت‌های تعمیم را ارزیابی کنیم، یا از یکی از روش‌های مختلف انتخاب ویژگی استفاده کنیم.

۵. L1 / L2 regularization (learning algorithm) .

رگولاریزیشن تکنیکی برای محدود کردن شبکه ما از مدلی است که بسیار پیچیده است. در رگولاریزیشن L1 یاL2 ، می‌توانیم یک عبارت جریمه را بر روی تابع هزینه اضافه کنیم تا ضرایب تخمینی را به سمت صفر برسانیم (و شدیدتر را نگیریم). رگولاریزیشن L2 به وزن‌ها اجازه می‌دهد تا به صفر کاهش یابند، اما نه به صفر، در حالی که L1 به وزن‌ها اجازه می‌دهد تا به صفر برسند.

۶. Remove layers / number of units per layer (model) .

همانطور که در رگولاریزیشن L1 یا L2 ذکر شد، یک مدل بیش از حد پیچیده ممکن است دچار بیش‌برازش شود. بنابراین، ما می توانیم به طور مستقیم پیچیدگی مدل را با حذف لایه ها کاهش دهیم و اندازه مدل خود را کاهش دهیم. ما ممکن است پیچیدگی را با کاهش تعداد نورون ها در لایه های کاملاً متصل کاهش دهیم .

۷. Dropout (model) .

با اعمالdropout ، که نوعی منظم‌سازی است، در لایه‌های خود، زیر مجموعه‌ای از واحدهای شبکه خود را با احتمال مجموعه نادیده می‌گیریم. با استفاده ازdropout ، می‌توانیم یادگیری وابسته به هم را در بین واحدها کاهش دهیم، که ممکن است به بیش‌برازش منجر شده باشد. با این حال، باdropout ، ما به آموزش بیشتری برای همگرایی مدل خود نیاز خواهیم داشت.

۸. Early stopping (model) .

توقف زودهنگام یک تکنیک منظم‌سازی (رگولاریزیشن) برای شبکه‌های عصبی عمیق است که هنگامی که به‌روزرسانی‌های پارامتر در مجموعه اعتبارسنجی دیگر شروع به بهبود نمی‌کنند، آموزش را متوقف می‌کند. در اصل، ما بهترین پارامترهای فعلی را در طول آموزش ذخیره و به روز می کنیم، و زمانی که به روز رسانی پارامترها دیگر بهبودی حاصل نمی شود (پس از یک تعداد تکرار مشخص)، آموزش را متوقف می کنیم و از آخرین بهترین پارامترها استفاده می کنیم. این روش با محدود کردن روند بهینه‌سازی به حجم کمتری از فضای پارامتر، به عنوان یک تنظیم‌کننده عمل می‌کند.

جلوگیری از کم‌برازش(underfitting) :

۱. افزایش پیچیدگی مدل

مدل ممکن است صرفاً به این دلیل که برای ثبت الگوها در داده ها به اندازه کافی پیچیده نیست، مناسب نیست. استفاده از یک مدل پیچیده تر، به عنوان مثال با تغییر از یک مدل خطی به یک مدل غیر خطی یا با افزودن لایه های پنهان به شبکه عصبی، اغلب کمک می کند.

۲. افزایش داده ها:

افزایش داده ها شامل ایجاد داده های آموزشی جدید با اعمال تبدیل به داده های موجود است. به عنوان مثال، می توانید تصاویر را به صورت افقی یا عمودی ورق بزنید، به داده ها نویز اضافه کنید، یا تصاویر را برای ایجاد نمونه های آموزشی جدید برش دهید. این می تواند به مدل کمک کند تا الگوهای پیچیده تر را یاد بگیرد و بهتر تعمیم دهد.

۳. تغییر معماری مدل:

به جای دستکاری مدل موجود، می توان از یک مدل متفاوت استفاده کرد. برای مثال، اگر از مدل رگرسیون خطی استفاده می شود، از یک شبکه عصبی یا درخت تصمیم استفاده کنیم.

۴. تنظیم Hyperparameter:

فراپارامترها پارامترهایی هستند که قبل از آموزش مدل تنظیم می شوند. تنظیم این فراپارامترها می تواند به بهبود عملکرد مدل کمک کند. برای مثال، می‌توانید نرخ یادگیری، قدرت منظم‌سازی یا تعداد دوره‌ها را تنظیم کنید.

۵. مهندسی ویژگی:

مهندسی ویژگی شامل انتخاب و تبدیل ویژگی‌های مرتبط در داده‌های شما برای بهبود توانایی مدل برای گرفتن الگوها است. می‌توانید ویژگی‌های جدیدی را اضافه کنید که ممکن است با مشکل مرتبط باشند، ویژگی‌های موجود را ترکیب کنید، یا ویژگی‌های نامربوط یا اضافی را حذف کنید.

هنگام آموزش یک مدل یادگیری ماشین، استفاده از مجموعه داده های آموزشی و اعتبارسنجی برای اطمینان از دقت و اثربخشی مدل ضروری است. مجموعه داده های آموزشی برای آموزش مدل استفاده می شود، به این معنی که الگوریتم الگوها و روابط را از این مجموعه داده یاد می گیرد. استفاده از مجموعه داده های آموزشی به اندازه کافی بزرگ و متنوع برای اطمینان از اینکه مدل به اندازه کافی آموزش داده شده است و می تواند به داده های جدید و دیده نشده تعمیم یابد، ضروری است. از سوی دیگر، مجموعه داده های اعتبارسنجی برای ارزیابی عملکرد مدل در طول آموزش استفاده می شود. این مجموعه داده برای نظارت بر دقت مدل در طول فرآیند آموزش و اطمینان از عدم تناسب بیش از حد مدل با مجموعه داده های آموزشی استفاده می شود. تطبیق بیش از حد زمانی اتفاق می‌افتد که مدل بیش از حد پیچیده باشد و مجموعه داده‌های آموزشی را به خاطر بسپارد، به جای یادگیری الگوهای کلی که می‌توانند روی داده‌های جدید اعمال شوند.هنگام تقسیم داده های عملیاتی، از مجموعه داده های آموزشی برای آموزش مدل استفاده می شود، در حالی که مجموعه داده های اعتبارسنجی برای ارزیابی دقت مدل در طول آموزش استفاده می شود. عملکرد مدل در مجموعه داده های اعتبارسنجی را می توان برای تنظیم پارامترهای مدل و جلوگیری از برازش بیش از حد استفاده کرد. هنگامی که مدل آموزش دید و اعتبار سنجی شد، می توان از آن برای پیش بینی داده های جدید و دیده نشده در مجموعه داده های عملیاتی استفاده کرد. بنابراین، استفاده از هر دو مجموعه داده آموزشی و اعتبارسنجی هنگام تقسیم داده های عملیاتی ضروری است تا اطمینان حاصل شود که مدل به اندازه کافی آموزش داده شده است و بر روی داده های جدید و دیده نشده به خوبی عمل می کند.