آشنایی با محیط توسعه یکپارچه

گوگل کولب

به همراه پیاده‌سازی الگوریتم‌های کاربردی یادگیری عمیق

گردآوری و تدوین:

حسین صدر

مژده نظری

زینب خداوردیان

انتشارات رئا

سرشناسه	:	صدر،حسین،۱۳۶۵
عنوان و نام پدیدآور	:	آشنایی با محیط توسعه یکپارچه گوگل کولب به همراه پیاده سازی الگوریتم‌های کاربردی یادگیری عمیق/ حسین صدر , مژده نظری سلیماندارابی , زینب خداوردیان.
مشخصات نشر	:	تهران : رئا وابسته به موسسه مروارید درخشان اندیشه ، ۱۴۰۲.
مشخصات ظاهری	:	۳۳۴ ص.:مصور
شناسه مجوز	:	۹-۷۷۵۹۳۶-۱۳۴۲۰۷
شابک	:	۹۷۸-۶۲۲-۹۰۰۸۰-۲-۷
وضعیت فهرست‌نویسی	:	فیپا
یادداشت	:	کتابنامه
موضوع	:	شبکه‌های عصبی (کامپیوتر) Neural networks (Computer science) پایتون (زبان برنامه‌نویسی کامپیوتر) Python (Computer program language) فراگیری ماشینی Machine learning
شناسه افزوده	:	نظری سلیماندارابی،مژده ،۱۳۶۸
شناسه افزوده	:	خداوردیان، زینب، ۱۳۷۰-
رده‌بندی کنگره	:	QA76/۸۷
رده‌بندی دیویی	:	۰۰۶/۳۲
شماره‌کتاب‌شناسی ملی	:	۹۳۷۲۵۷۳
اطلاعات رکورد کتاب شناسی	:	فیپا

آشنایی با محیط توسعه یکپارچه گوگل کولب به همراه پیاده سازی الگوریتمهای کاربردی یادگیری عمیق

مولف	:	**حسین صدر، مژده نظری سلیماندارابی ، زینب خداوردیان**
طراح جلد	:	**سهیل فاخری**
شابک	:	**۹۷۸-۶۲۲-۹۰۰۸۰-۲-۷**
نوبت چاپ	:	**اول**
سال چاپ	:	۱۴۰۲
شمارگان	:	۵۰۰ نسخه
قیمت	:	**۲۰۰۰۰۰۰ ریال**

فهرست‌نویسی پیش از انتشار کتابخانه ملی جمهوری اسلامی ایران

نشانی: تهران-بلوار عدل- بالاتر از چهارراه عدل-کوچه گلزار یکم- پلاک۳۶-مروارید درخشان اندیشه

تارنما:**https://mdapubs.com**

پیشگفتار

عصر حاضر سرشار از تکنولوژی‌هایی است که هوش مصنویی جزو لاینفک آن‌ها می‌باشد و روز به روز بر تعداد محققان، پژوهشگران و دانشجویان این حوزه افزوده می‌شود. یکی از شاخه‌های قابل توجه هوش مصنوعی، الگوریتم‌های یادگیری عمیق است که پیاده‌سازی این مدل‌ها بر روی مجموعه داده‌ها از دغدغه‌های اساسی پژوهشگران این حوزه می‌باشد که مهمترین دلیل آن کمبود منابع سخت افزاری بخصوصGPU و از طرفی محدودیت‌های دسترسی به آن‌ها در دانشگاه‌ها و مراکز تحقیقاتی است. برای تشریح بهتر موضوع در ذهن خود مجسم کنید که به یک سیستم قدرتمند لینوکسی مجهز به GPU با مشخصات NVIDIA GeForce RTX 3090 با قیمت تقریبی حدود سه هزار و پانصد دلار، دسترسی دارید که بی‌شمار کاربرد خواهد داشت و براحتی می‌توان کدها را بروی آن اجرا کرد. حال کمی واقع‌بینانه به موضوع بنگریم؛ خرید و دسترسی به منابع سخت افزاری به‌روز که پاسخگوی نیازمندی‌های مدل‌های پیچیده امروزی باشد کار چندان ساده‌ای نیست. می‌توان گفت یکی از چالش‌های مهم این روزهای دنیای برنامه‌نویسان به‌ویژه در حوزه هوش مصنوعی و یادگیری عمیق محدودیت منابع سخت‌افزاری بخصوص GPU است. برای مواجهه با این چالش، شرکت گوگل، سرویس گوگل‌کولب که یکی از مجهزترین سیستم‌های پردازشگر ابری مجازی دنیاست را عرضه نموده است.

گوگل‌کولب[1] یک سرویس ابری است که سخت‌افزارهایی همچون GPU را بصورت رایگان، با سرعت بالا ارایه می‌دهد که به‌راحتی می‌توان به آن دسترسی داشت. برای استفاده از این سرویس تنها چیزی که مورد نیاز است داشتن اینترنت و حساب گوگل است که از طریق ثبت‌نام سرویس جی‌میل به کاربران داده می‌شود. با ساخت حساب کاربری فضای ۲۵ گیگابایتی در گوگل درایو[2] به کاربران اختصاص داده می‌شود که می‌توان با آپلود داده‌های خود در این محیط یا دانلود داده‌ها از سایت‌های دیگر به صورت مستقیم از کولب استفاده کرد.

کولب یک محیط مشابه ژوپیتر نوت‌بوک[3] است و برنامه‌نویس را قادر می‌سازد تا با استفاده مرورگر و بدون هیچ گونه پیش نیازی بتوان با محبوب ترین زبان برنامه نویسی یعنی زبان برنامه نویسی پایتون کدهای خود را نوشت و پکیج (کتابخانه) جدیدی را در آن اجرا کرد. کولب برای هر کاربر سیزده گیگابایت حافظه اصلی[4] (به صورت اختصاصی البته موقت) اختصاص می‌دهد که می‌تواند کدهای اجرایی را از گیت‌هاب و یا گوگل

---

1 Google Colab
2 Google Drive
3 Jupyter Notebook
4 RAM

درایو، فراخوانی یا به آنها ارسال کند و برای حوزه یادگیری ماشین[1] و علوم داده[2] مناسب‌ترین ابزار پیاده سازی است. از جمله پکیج‌ها و ابزارهای پیش‌فرض یادگیری عمیق که می‌توان به آن اضافه نمود می‌توان به OpenCV، PyTorch، Keras وTensorflow اشاره کرد.

با این‌حال بسیاری از پژوهشگران با امکانات این سرویس آشنایی ندارند. برهمین اساس، تصمیم گرفتیم آموزش گوگل کولب را در غالب کتابی آموزشی تدوین نماییم که شامل دو بخش مجزا می‌باشد در بخش ابتدایی مفاهیم تئوری یادگیری ژرف می‌باشد که چند مدل رایج در یادگیری ژرف تشریح می‌گردد؛ در بخش دوم کتاب به معرفی عملیاتی گوگل کولب پرداخته می‌شود و در ادامه این بخش پیاده‌سازی چند کاربرد قابل توجه در پردازش تصویر و پردازش متن همچون تشخیص چهره، شناسایی اعداد دستنویس، شناسایی اثر انگشت، تجزیه و تحلیل احساس، خلاصه سازی متن ارایه خواهد شد.

این اثر برای کلیه دانشجویان و پژوهشگران حوزه برنامه‌نویسی یادگیری عمیق که با زبان پایتون فعالیت می‌کنند، مناسب است و خیلی سریع بدون نصب نرم افزاری نحوه کدنویسی در کولب را می‌آموزید. در پایان، امیدواریم این اثر، منبع مفید و موثر برای محققان، پژوهشگران و دانشجویان عزیز باشد و از خوانندگان محترم درخواست می‌شود به منظور ارتقای کیفی محتوی آن در چاپ‌های بعدی، مولفان را از پیشنهادات و انتقادات سازنده خود از طریق پست الکترونیکی به آدرس Sadr@qiau.ac.ir مطلع فرمایند.

مولفان

۱۴۰۱/۰۱۰/۱

1 machine learning
2 data science

فهرست مطالب

فصل اول: مقدمات یادگیری ژرف ۱

۱-۱- دیباچه ۱

۱-۲- یادگیری سطحی ۲

۱-۳- یادگیری ژرف ۳

۱-۴- یادگیری ژرف چگونه کار می‌کند؟ ۸

۱-۴-۱- مسیر عقب رو (پس انتشار) ۱۰

۱-۵- توابع زیان و طبقه‌بندی کننده بیشینه هموار ۱۲

۱-۵-۱- زیان میانگین مربع خطا ۱۲

۱-۵-۲- زیان آنتروپی متقاطع ۱۳

۱-۵-۳- طبقه‌بندی کننده بیشینه هموار ۱۳

۱-۶- تکنیک‌های بهینه سازی مبتنی بر گرادیان کاهشی ۱۵

۱-۶-۱- گرادیان کاهشی (GD) دسته‌ای ۱۶

۱-۶-۲- گرادیان کاهشی تصادفی ۱۶

۱-۶-۳- گرادیان کاهشی دسته‌ای کوچک ۱۷

۱-۷- بهبود گرادیان کاهشی به منظور همگرایی سریع‌تر ۱۷

۱-۷-۱- AdaGrad ۱۸

۱-۷-۲- AdaDelta ۱۸
۱-۷-۳- RMSProp ۱۹
۱-۷-۴- AdaDelta ۲۰
۱-۸- چالش‌های آموزش شبکه‌های ژرف ۲۱
۱-۸-۱- محو شدگی گرادیان ۲۱
۱-۸-۲- اندازه داده آموزشی ۲۲
۱-۸-۳- بیش برازش و کم برازش ۲۳
۱-۸-۴- سخت افزار با کارایی بالا ۲۶
۱-۹- تکنیک‌های مقداردهی اولیه وزن ۲۶
۱-۹-۱- مقدار اولیه تمام وزن‌ها را صفر قرار دهید ۲۷
۱-۹-۲- مقداردهی اولیه تصادفی ۲۷
۱-۹-۳- وزن‌های تصادفی از توزیع احتمال ۲۸
۱-۹-۴- انتقال یادگیری ۲۸
۱-۱۰- خلاصه ۲۹
فصل دوم: معرفی شبکه عصبی پیچشی ۳۱
۲-۱- دیباچه ۳۱
۲-۲- شبکه عصبی پیچشی ۳۲

۲-۲-۱- عملیات پیچشی ۳۳
۲-۲-۲- معماری شبکه عصبی پیچشی ۳۴
۲-۲-۳- لایه پیچشی ۳۶
۲-۲-۴- تابع فعال‌سازی (ریلو) ۳۹
۲-۲-۵- لایه ادغام ۴۲
۲-۲-۶- لایه تمام متصل ۴۳
۲-۲-۷- حذف تصادفی ۴۴
۲-۳- معرفی چندین معماری از شبکه عصبی پیچشی ۴۵
۲-۳-۱- معماری LeNet-5 ..... ۴۵
۲-۳-۲- معماری AlexNet ۴۸
۲-۳-۳- معماری ZFNet ۵۱
۲-۳-۴- معماری VGGNet ۵۳
۲-۳-۵- معماری GoogleNet ۵۶
۲-۳-۶- معماری ResNet ۶۰
۲-۳-۷- شبکه پیچشی متصل شده به صورت متراکم (DenseNet) ۶۴
۲-۳-۸- شبکه‌ی کپسولی ۶۷

۲-۴- خلاصه ۷۰

فصل سوم: معرفی شبکه عصبی برگشتی ۷۲

۳-۱- دیباچه ۷۲

۳-۲- شبکه عصبی برگشتی ۷۳

۳-۳- مفهوم محو شدگی گرادیان ۷۸

۳-۴- شبکه‌های حافظه طولانی کوتاه مدت ۸۱

۳-۵- واحد برگشتی دروازه‌دار ۸۴

۳-۶- خلاصه ۸۹

فصل چهارم: معرفی شبکه عصبی مولد متخاصم ۹۰

۴-۱- دیباچه ۹۰

۴-۲- شبکه عصبی مولد متخاصم ۹۱

۴-۳- فرآیند آموزش شبکه عصبی مولد متخاصم ۱۰۰

۴-۳- انواع شبکه‌های عصبی مولد متخاصم ۱۰۳

۴-۴- کاربردهای شبکه عصبی مولد متخاصم ۱۰۷

۴-۵- خلاصه ۱۰۹

فصل پنجم: معرفی گوگل کولب و امکانات آن ۱۱۰

۵-۱- دیباچه ۱۱۰
۵-۲- معرفی گوگل کولب ۱۱۰
۵-۲-۱- Google Colaboratory چیست؟ ۱۱۱
۵-۲-۲- بیان مزایای استفاده از سرویس کولب و ذکر محدودیت‌ها ۱۱۲
۵-۲-۳- انواع پردازشگر قابل استفاده در کولب ۱۱۵
۵-۲-۴- چرا برای آموزش مدل‌های یادگیری ژرف به واحد پردازش گرافیکی نیاز است؟ ۱۱۶
۵-۲-۵- محدودیت منابع پردازشی در کولب گوگل ۱۱۸
۵-۳- شروع کار با گوگل کولب ۱۱۹
۵-۳-۱- آشنایی با ژوپیتر نوت‌بوک و اجزای آن ۱۲۰
۵-۳-۲- آشنایی با بخش‌های مختلف گوگل کولب ۱۲۳
۵-۴- نوشتن یک برنامه در کولب ۱۳۰
۵-۴-۱- کار با سلول کد ۱۳۱
۵-۴-۲- کار با سلول متنی ۱۳۲
۵-۴-۳- رسم نمودار در کولب ۱۳۵
۵-۴-۴- کلیدهای میانبر ۱۳۶
۵-۴-۵- استفاده از دستورات لینوکسی در گوگل کولب ۱۳۷

۵-۶- اتصال کولب به گوگل درایو ۱۴۲

۵-۷- همراهی مخزن هوش مصنوعی گوگل با کولب ۱۴۴

۵-۸- اتصال کولب به کگل ۱۴۵

۵-۹- دانلود پروژه گیت هاب در کولب ۱۴۸

۵-۱۰- نصب پکیج یا کتابخانه جدید در گوگل کولب ۱۴۹

۵-۱۱- مدیریت فایل در کولب ۱۵۰

۵-۱۲- خلاصه ۱۵۲

فصل ششم: پروژه شناسایی اعداد دست نویس در گوگل کولب ۱۵۳

۶-۱- دیباچه ۱۵۳

۶-۲- پروژه شناسایی اعداد دست نویس در گوگل کولب ۱۵۴

۶-۲-۱- معرفی مجموعه داده شناسایی اعداد دست نویس ۱۵۵

۶-۲-۲- وارد کردن داده ورودی و پیش‌پردازش آن ۱۵۹

۶-۲-۳- معرفی معماری مدل ژرف(شبکه عصبی پیچشی) و ساخت آن ۱۶۶

۶-۲-۴- ارزیابی مدل ۱۷۵

۶-۳- خلاصه ۱۸۵

فصل هفتم: پروژه شناسایی اثر انگشت در گوگل کولب ۱۸۶

۷-۱- دیباچه ۱۸۶
۷-۲- مراحل کار سیستم تشخیص اثر انگشت ۱۸۷
۷-۳- پروژه شناسایی اثر انگشت در گوگل کولب ۱۹۳
۷-۳-۱- معرفی مجموعه داده مناسب برای تشخیص اثر انگشت ۱۹۳
۷-۳-۲- وارد کردن مجموعه داده ۱۹۵
۷-۲-۳- پیش‌پردازش داده ورودی ۱۹۸
۷-۲-۴- ساخت و آموزش مدل ۲۰۲
۷-۲-۵- ارزیابی مدل پیشنهادی ۲۰۵
۷-۳- خلاصه ۲۰۷
فصل هشتم: پروژه شناسایی چهره در گوگل کولب ۲۰۹
۸-۱- دیباچه ۲۰۹
۸-۲- مراحل کار سیستم تشخیص چهره ۲۱۰
۸-۳- پروژه شناسایی چهره در گوگل کولب ۲۱۲
۸-۳-۱- معرفی مجموعه داده مناسب برای تشخیص چهره ۲۱۲
۸-۳-۲- وارد کردن مجموعه داده ۲۱۶
۸-۳-۳- پیش‌پردازش داده ورودی ۲۲۷

۸-۳-۴- ساخت مدل و آموزش آن ۲۲۸

۸-۳-۵- ارزیابی مدل ۲۳۲

۸-۷- خلاصه ۲۳۷

فصل نهم: پروژه خلاصه‌سازی متن در کولب ۲۳۸

۹-۱- دیباچه ۲۳۸

۹-۲- خلاصه‌سازی متن ۲۳۹

۹-۲-۱- خلاصه‌سازی استخراجی متن ۲۴۰

۹-۲-۲- خلاصه‌سازی چکیده‌ای متن ۲۴۱

۹-۲-۲- خلاصه‌سازی ساخت یافته متن ۲۴۳

۹-۳- پروژه خلاصه‌سازی چکیده‌ای متن در گوگل کولب ۲۴۴

۹-۳-۱- معرفی مجموعه داده غذای آمازون ۲۴۸

۹-۳-۲- پیاده‌سازی معماری دنباله به دنباله با استفاده از مکانیسم توجه ۲۵۰

۹-۴- خلاصه ۲۷۴

فصل دهم: پروژه تحلیل احساسات در گوگل کولب ۲۷۵

۱۰-۱- دیباچه ۲۷۵

۱۰-۲- تحلیل احساسات ۲۷۶

۱۰-۳- پروژه تحلیل احساسات در گوگل کولب ۲۸۰

۱۰-۳-۱- معرفی مجموعه داده مناسب برای تحلیل احساسات ۲۸۰

۱۰-۳-۲- وارد کردن مجموعه داده ۲۸۲

۱۰-۳-۳- پیش‌پردازش داده ورودی ۲۸۷

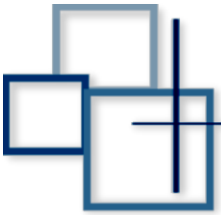
۱۰-۳-۴- ساخت مدل و آموزش آن ۳۰۲

۱۰-۳-۵- ارزیابی مدل ۳۰۹

۱۰-۴- خلاصه ۳۱۱

منابع ۳۱۳

فصل اول
مقدمات یادگیری ژرف



۱-۱- دیباچه

هدف سیستم‌های یادگیری ماشین، با معماری‌های سطحی یا ژرف این است که به یک ماشین اجازه داده شود تا اطلاعات مفید را از داده‌های خام یاد بگیرد. در سطح انتزاعی، یادگیری ماشین می‌تواند با استفاده از رویکردهای زیر انجام شود:

- **یادگیری نظارت شده**[1] **:** یک سیستم را به گونه‌ای تطبیق می‌دهد که برای ورودی‌های داده شده، یک خروجی هدف تولید کند. داده‌های یادگیری از مواردی تشکیل شده است (ویژگی‌ها و برچسب) که "ویژگی‌ها" نمایانگر داده‌های ورودی هستند و "برچسب" بیانگر خروجی هدف. هدف در اینجا سازگارکردن سیستم است تا بتواند برای ورودی جدید، خروجی هدف را پیش‌بینی کند. یادگیری با ناظر می تواند از هر دو نوع پیوسته و گسسته از داده‌های ورودی استفاده کند.
- **یادگیری نظارت نشده**[2] **:** شامل داده‌هایی است که از بردارهای ورودی بدون هیچگونه خروجی هدفی تشکیل شده است. اهداف مختلفی در یادگیری بدون ناظر مانند خوشه‌بندی، برآورد توزیع و بصری سازی وجود دارد. هدف اصلی خوشه‌بندی کشف گروه‌هایی از اقلام داده‌های متشابه براساس مطابقت‌ها یا شباهت‌های اندازه‌گیری شده یا درک شده بین

[1] Supervised learning
[2] Unsupervised learning

آن‌هاست (به عبارتی اعضای یک خوشه دارای بیشترین شباهت هستند). هدف از برآورد تراکم، تعیین توزیع داده‌ها در فضای ورودی است. در بصری سازی، داده‌ها از یک فضای با ابعاد بالا به دو یا سه بعد مشاهده‌ی موارد مشابه کاهش می یابند.

یادگیری نیمه نظارتی[1] **:** ابتدا از داده‌های بدون برچسب برای یادگیری بازنمایی ویژگی داده‌های ورودی و سپس از نمایش ویژگی یادگرفته شده برای حل وظیفه نظارت شده استفاده می‌کند. مجموعه داده‌های آموزشی می‌توانند به دو بخش تقسیم شوند: نمونه داده‌هایی با برچسب مربوطه و نمونه داده‌هایی که برچسب آن‌ها شناخته شده نیست. یادگیری نیمه نظارتی می‌تواند شامل اختصاص پاداش و جزا باشد که نشان می‌دهد چگونه سیستم باید رفتار خود را تغییر دهد و گاهی اوقات به عنوان یادگیری تقویتی[2] نامیده می‌شود. یادگیری تقویتی در کاربردهای متنوعی چون "پرواز هلی‌کوپتر خودگردان"، "حرکات بات"، "مسیریابی شبکه تلفن همراه"، "انتخاب استراتژی بازاریابی" و "ایندکس کردن کارآمد صفحه وب" مورد استفاده قرار می‌گیرد.

۱-۲- یادگیری سطحی[3]

معماری‌های سطحی در بسیاری از مسائل رایج یادگیری ماشین عملکرد خوبی دارند و هنوز هم در اکثریت قریب به اتفاق کاربردهای امروزه یادگیری ماشین استفاده می‌شوند. با این حال اخیراً علاقه به معماری‌های ژرف بیش‌تر شده است همچنین امید به یافتن راهی برای حل مسائل پیچیده‌تر در دنیای واقعی (به عنوان مثال، تجزیه و تحلیل تصویر یا درک زبان طبیعی) که معماری‌های سطحی قادر به یادگیری کافی آن مدل‌ها نیستند.

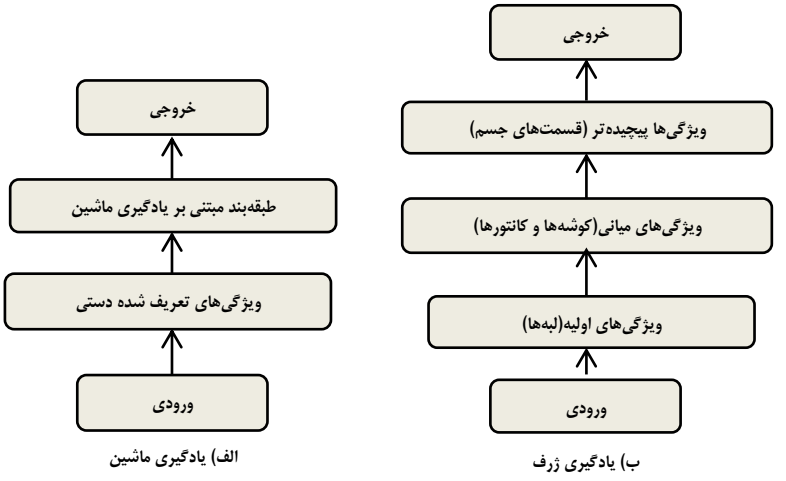
[1]Semi-supervised learning
[2]Reinforcementlearning
[3]Shallow Learning

۱-۳- یادگیری ژرف[1]

یادگیری ژرف حوزه جدیدی از یادگیری ماشین است که اخیرا محبوبیت پیدا کرده است. یادگیری ژرف به معماری‌هایی گفته می‌شود که شامل چندین لایه پنهان (شبکه‌های ژرف) برای یادگیری ویژگی‌های مختلف با چندین سطح انتزاع هستند. الگوریتم‌های یادگیری ژرف قادر به استخراج ویژگی‌ها به صورت اتوماتیک هستند که این ویژگی‌ها در سطح بالاتری از ویژگی‌های اولیه هستند و حاوی اطلاعات دقیق‌تری از داده ورودی می‌باشند. در تکنیک های متداول یادگیری ماشین، پردازش داده‌های طبیعی به منظور استخراج الگو، بر اساس ویژگی‌های اولیه تعریف شده برای داده‌ها می‌باشد. برای دهه‌ها ساخت یک سیستم تشخیص الگو با استفاده از تکنیک‌های یادگیری ماشین، نیازمند تخصص دقیق در مهندسی ویژگی به منظور استخراج ویژگی‌های مناسب از داده‌های خام بود، به گونه‌ای که طبقه‌بند بتواند الگوی داده ورودی را تشخیص داده و آن را طبقه بندی کند. یادگیری ژرف به داده‌های خام ورودی اجازه می‌دهد (پیکسل‌های یک مورد داده تصویر) تا الگوریتم را بدون استخراج اولیه ویژگی‌ها یا تعریف یک بردار ویژگی یاد بگیرند. الگوریتم‌های یادگیری ژرف می‌توانند مجموعه‌ای از ویژگی‌ها را به درستی یاد بگیرند و این کار را به روش بسیار بهتری نسبت به استخراج این ویژگی‌ها با استفاده از کدنویسی دستی انجام می‌دهد. به عنوان مثال، در یک تصویر داده شده، مقادیر شدت پیکسل به عنوان ورودی‌های سیستم یادگیری ژرف فرستاده می‌شوند. تعدادی از لایه‌های پنهان ویژگی‌ها را از تصویر ورودی استخراج می‌کنند. این لایه‌های پنهان به گونه‌ای سلسله مراتبی بر روی یکدیگر بنا شده‌اند. در ابتدا، لایه‌های سطح پایین شبکه فقط نواحی لبه مانند را تشخیص می‌دهند. این نواحی لبه برای تعریف گوشه‌ها[2] (جایی که لبه‌ها با هم تلاقی می‌کنند) و کانتورها[3] (رئوس یا طرح کلی اجسام) استفاده می‌شود. لایه‌ها در سطح بالاتر گوشه‌ها و کانتور را با هم ترکیب می‌کنند تا منجر به انتزاع بیش‌تر "قسمت‌های جسم" در لایه بعدی شوند. جنبه اصلی یادگیری ژرف این است که این لایه‌ها از ویژگی‌ها به دست مهندسان انسانی ساخته و طراحی نشده‌اند. (ویژگی‌ها به صورت دستی توسط یک فرد متخصص برای نمونه‌ها تعریف نشده است) بلکه به تدریج با استفاده از یک روش یادگیری همه منظوره، از داده‌ها آموخته می‌شوند.

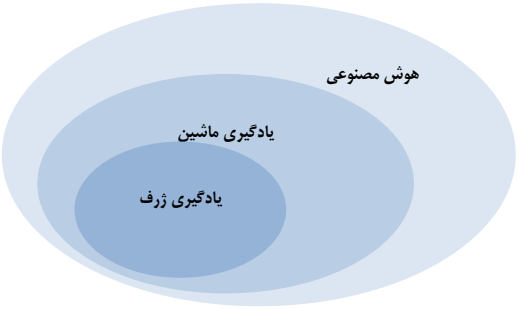
[1] Deep Learning
[2] Corners
[3] Contours

سرانجام، لایه خروجی، تصویر را طبقه‌بندی کرده و برچسب کلاس خروجی را بدست می‌آورد. خروجی بدست آمده در لایه خروجی به طور مستقیم با تمام نودها یا نورون های لایه قبل در ارتباط است. این فرآیند را می‌توان به عنوان یادگیری سلسله مراتبی مشاهده کرد زیرا هر لایه در شبکه از خروجی لایه‌های قبلی به عنوان "بلوک‌های سازنده" برای ساخت مفاهیم پیچیده‌تر به طور فزاینده‌ای در لایه‌های بالاتر استفاده می‌کند. شکل ۱ رویکرد یادگیری ماشین سنتی مبتنی بر ویژگی‌های دست ساز (ویژگی‌های تعریف شده به صورت دستی) را با رویکرد یادگیری ژرف مبتنی بر یادگیری بازنمایی سلسله مراتبی مقایسه می‌کند. به‌طور خاص در یادگیری ژرف با تأکید بر تهیه نگاشت پیچیده با استفاده از یک سری نگاشت‌های ساده، نمایش معناداری از داده‌های ورودی یاد گرفته می‌شود. کلمه "ژرف" به یادگیری لایه‌های متوالیِ بازنمایی معنادار از داده‌های ورودی اشاره دارد. تعداد لایه‌هایی که برای مدل‌سازی داده‌ها استفاده می‌شوند، عمق مدل را تعیین می‌کند. یادگیری ژرف کنونی غالباً شامل یادگیری ده‌ها یا حتی صدها لایه متوالی نمایش از داده‌های آموزشی به صورت خودکار است. رویکردهای متداول یادگیری ماشین اغلب بر یادگیری فقط یک یا دو لایه نمایش داده متمرکز است، چنین رویکردهایی اغلب به عنوان یادگیری سطحی دسته‌بندی می‌شوند.



شکل۱-(الف) یادگیری ماشین متداول و (ب) رویکرد یادگیری ژرف

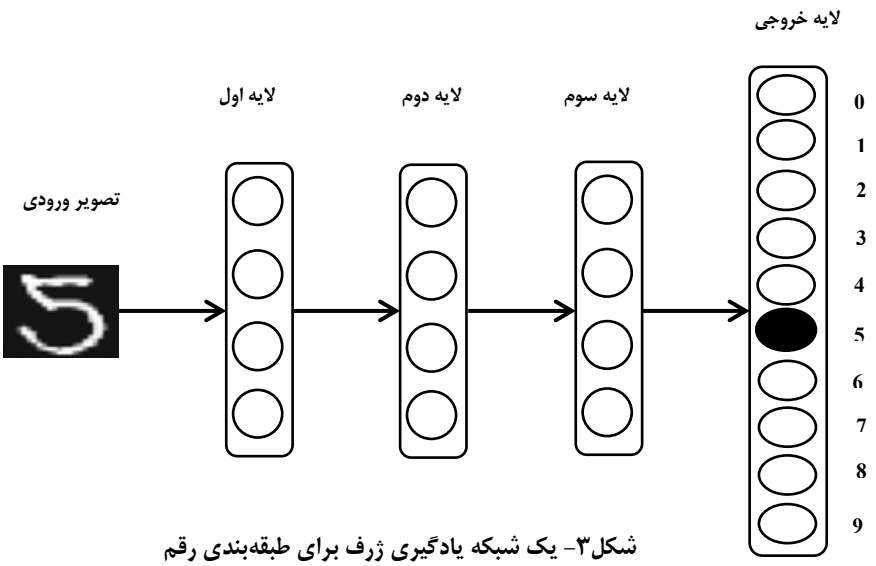
یادگیری ژرف و یادگیری ماشین زیرمجموعه‌های هوش مصنوعی هستند. شکل ۲ رابطه بین هوش مصنوعی، یادگیری ماشین و یادگیری ژرف را نشان می‌دهد.



شکل۲- رابطه بین هوش مصنوعی، یادگیری ماشین و یادگیری ژرف

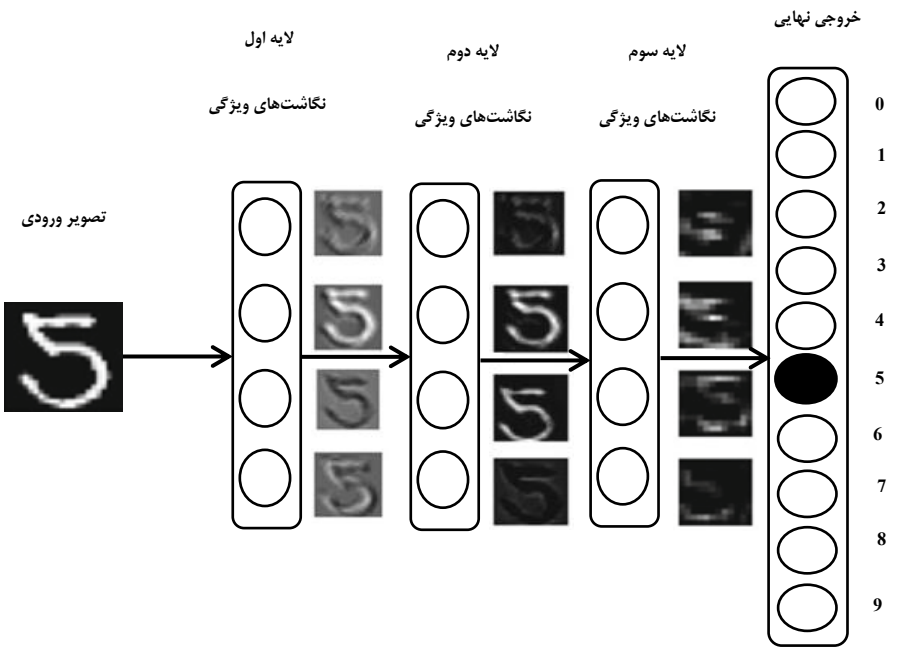
با توجه به اینکه یک شبکه ژرف معمولاً لایه‌ها و پارامترهای بیش‌تری دارد، این پتانسیل را دارد که ورودی‌های پیچیده‌تری را نشان دهد. اگرچه یادگیری ژرف از دهه ۱۹۸۰ وجود داشته است، اما از آنجا که زیرساخت‌های محاسباتی(سخت افزار و نرم افزار) کافی نبود و مجموعه داده‌های در دسترس در مقیاس کوچکی بودند، برای چندین سال از محبوبیت آن کاسته شد. با کاهش محبوبیت شبکه‌های عصبی متداول، اخیراً شبکه‌های ژرف با دستیابی به نتایج چشم‌گیر در تشخیص گفتار و وظایف بینایی کامپیوتر، ظهور بزرگی پیدا کردند.

اصطلاح شبکه‌های عصبی مصنوعی اشاره به علوم اعصاب دارد اما شبکه‌های یادگیری ژرف مدل‌هایی از مغز نیستند. با این حال، تکنیک‌های یادگیری ژرف تنها با الهام از عملکرد بیولوژیکی مغز، تنظیم و فرموله می‌شوند. همه اجزای مدل‌های ژرف الهام گرفته از علوم اعصاب نیستند، برخی از آن‌ها ناشی از اکتشاف تجربی، نظریه و شهود است. فعالیت عصبی در مغز انسان بسیار پیچیده‌تر از آن است که ممکن باشد با مطالعه ساده سلول‌های عصبی مصنوعی پیشنهاد شود. مکانیسم‌های یادگیری استفاده شده توسط مدل‌های یادگیری ژرف به هیچ وجه با مغز انسان قابل مقایسه نیستند، اما می‌تواند به عنوان یک چارچوب ریاضی برای یادگیری بازنمایی داده‌ها توصیف شود. شکل ۳ یک مثال از معماری یادگیری ژرف را نشان می‌دهد که برای شناسایی کاراکتر می‌تواند مورد استفاده قرار گیرد.



شکل۳- یک شبکه یادگیری ژرف برای طبقه‌بندی رقم

شکل ۴ بازنمایی‌هایی را نشان می‌دهد که توسط شبکه یادگیری ژرف یادگرفته می‌شوند. شبکه ژرف از چندین لایه برای تبدیل تصویر ورودی (در اینجا یک رقم) استفاده می‌کند تا رقم را تشخیص دهد. هر لایه تعدادی تبدیل را روی ورودی که از لایه‌های قبلی دریافت می‌کند، اعمال می‌کند. شبکه ژرف، تصویر رقم را به تصویرهایی تبدیل می‌کند که تمایل دارند سطح بالاتری از انتزاع را ثبت کنند. شبکه ژرف را می‌توان یک عملیات اطلاعاتی چند مرحله‌ای دانست که هر لایه پنهان تصویر ورودی را به تصویری تبدیل می‌کند که به طور فزاینده‌ای با تصویر اصلی متفاوت است و در نتیجه نهایی مورد استفاده قرار می‌گیرد.



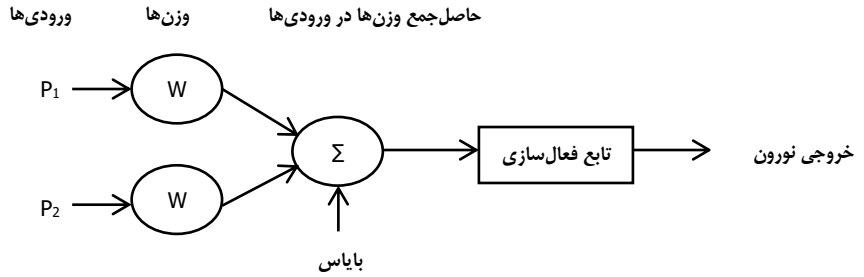
شکل۴- نمایش‌های آموخته شده در طول اولین گذردهی برای طبقه‌بندی رقم توسط یکی شبکه ژرف

به طور خلاصه، یک شبکه یادگیری ژرف ویژگی‌هایی در چندین سطح که ویژگی‌های بالاتر به عنوان توابعی از پایین‌ترها ساخته شده‌اند را ایجاد می‌کند. این یک عرصه به سرعت در حال رشد است که مسئله استخراج ویژگی را که به عنوان مقدمه روش‌های متداول یادگیری ماشین استفاده می‌شود، دور می‌زند. یادگیری ژرف به خودی خود قادر به یادگیری ویژگی‌های مناسب است که نیاز به هدایت کمی توسط کاربر دارد.

۱-۴- یادگیری ژرف چگونه کار می‌کند؟

شبکه‌های ژرف، ورودی را با دنباله‌ای از تغییرات لایه‌ای به هدف نگاشت می‌کنند و این تغییرات لایه‌ای را با وارد شدن مثال‌های آموزش (نمونه‌های آموزشی) آغاز می‌شود. تبدیلاتی که یک لایه در ورودی آن اعمال می‌کند توسط وزن لایه تعیین می‌شود، که اساساً یک دسته اعداد است. به عبارت دیگر، تغییرات اعمال شده توسط یک لایه با توجه به وزن آن، پارامتر و اندازه‌گیری می‌شود. در این زمینه، یادگیری می‌تواند به عنوان فرآیند یافتن مقادیر اوزان تمام لایه‌ها در شبکه به گونه‌ای تعریف شود که بتوان نمونه‌های ورودی را به درستی با اهداف مرتبط آن‌ها نگاشت کرد. یک شبکه یادگیری ژرف شامل هزاران پارامتر است و یافتن مقادیر مناسب و صحیح این پارامترها کار ساده‌ای نیست، به ویژه هنگامی که مقدار یک پارامتر بر مقدار پارامتر دیگر تأثیر بگذارد. برای درک موضوع می‌توان عملکرد یک شبکه عصبی تک لایه را مثال زد. یک شبکه عصبی از تعدادی نورون تشکیل شده است. یک شبکه عصبی مصنوعی، از سه لایه ورودی، خروجی و پردازش تشکیل می‌شود .هر لایه شامل گروهی از سلول‌های عصبی (نورون) است که عموماً با کلیه نورون‌های لایه‌های دیگر در ارتباط هستند، مگر این که کاربر ارتباط بین نورون‌ها را محدود کند؛ ولی نورون‌های هر لایه با سایر نورون‌های همان لایه، ارتباطی ندارند. نورون کوچک‌ترین واحد پردازشگر اطلاعات است که اساس عملکرد شبکه‌های عصبی را تشکیل می‌دهد. یک شبکه عصبی مجموعه‌ای از نورون‌هاست که با قرار گرفتن در لایه‌های مختلف، معماری خاصی را بر مبنای ارتباطات بین نورون‌ها در لایه‌های مختلف تشکیل می‌دهند. نورون می‌تواند یک تابع ریاضی غیرخطی باشد، در نتیجه یک شبکه عصبی که از اجتماع این نورون‌ها تشکیل می‌شود، نیز می‌تواند یک سامانه کاملاً پیچیده و غیرخطی باشد. در شبکه عصبی هر نورون به‌طور مستقل عمل می‌کند و رفتار کلی شبکه، برآیند رفتار نورون‌های متعدد است. به عبارت دیگر، نورون‌ها در یک روند همکاری، یکدیگر را تصحیح می‌کنند. یک شبکه عصبی تک لایه اولین داده ورودی را دریافت می‌کند. داده ورودی می‌تواند تعداد مشخصی متغیر (ویژگی) داشته باشد. به عنوان مثال ویژگی قد و وزن برای یک نمونه. واضح است که نمونه مورد نظر با یک بردار دوبعدی قابل نمایش است $X=[P_1,P_2]$ که P_1 نشان‌دهنده ویژگی قد و P_2 نشان دهنده ویژگی وزن می‌باشد که این دو ویژگی به عنوان ورودی به شبکه عصبی وارد می‌شود. لازم به ذکر است که هر لایه از تعدادی نورون تشکیل شده است. ورودی هر

نورون مجموع حاصل‌ضرب ورودی در وزن می‌باشد. مقدار حاصل با بایاس جمع می‌شود و بعد از عبور از یک تابع فعال‌سازی به عنوان خروجی نورون در نظر گرفته می‌شود. خروجی یک نورون می‌تواند برابر با خروجی واقعی نمونه یا به عبارتی برابر با برچسب آن باشد که در این صورت خطا وجود ندارد. اما اگر خروجی به دست‌آمده برابر با خروجی واقعی نباشد، آنگاه نیاز به محاسبه خطا می‌باشد. شکل ۵ مراحل گفته شده را نشان می‌دهد که به اصطلاح گام پیش‌خور[1] نام دارد. در این شکل از یک شبکه عصبی تک لایه به همراه یک نورون استفاده شده است.



شکل۵- عملکرد یک نورون در شبکه عصبی

می‌توان مطالب بیان شده را برای لایه‌های بیشتر و در یک شبکه ژرف بسط داد. برای آموزش یک شبکه ژرف باید فهمید که خروجی محاسبه شده شبکه از مقدار مطلوب چقدر فاصله دارد. این معیار با استفاده از یک تابع ضرر[2] بدست می‌آید که به آن تابع هدف[3] نیز گفته می‌شود. تابع هدف تفاوت بین خروجی پیش‌بینی شده از شبکه و مقدار واقعی هدف را برای یک مثال خاص محاسبه می‌کند. این یک معیار را نشان می‌دهد که شبکه یک مثال خاص را چقدر خوب یاد گرفته است. هدف از آموزش، یافتن مقادیر برای اوزانی است که تابع ضرر انتخابی را به حداقل می‌رساند. از اختلاف به دست آمده به عنوان یک سیگنال بازخورد برای تنظیم اوزان شبکه استفاده می‌شود، به گونه‌ای که امتیاز ضرر برای مثال فعلی کاهش بیابد. این تنظیم توسط بهینه ساز[4] یعنی الگوریتم پس‌انتشار[5]، انجام می‌شود. الگوریتم تکثیر به

[1]Feedforward
[2]Loss Function
[3]Objective Function
[4]Optimizer
[5]Backpropagation Algorithm

عقب یا پس‌انتشار در ابتدا شامل اختصاص مقادیر تصادفی به بردارهای وزنی است، به طوری که شبکه فقط یک سری تحولات تصادفی را پیاده‌سازی می‌کند. در ابتدا، خروجی به دست آمده از شبکه می‌تواند دور از مقدار واقعی باشد، باشد و بر این اساس امتیاز ضرر ممکن است بسیار زیاد شود. با هر مثالی که به شبکه داده می‌شود، وزن‌ها در جهتی تنظیم می‌شوند که باعث شود امتیاز ضرر کاهش یابد. این روند چندین بار تکرار می‌شود، تا زمانی که مقادیر وزنی که تابع ضرر را به حداقل می‌رساند، بدست آید. زمانی که می‌گویند یک شبکه بهتر یاد گرفته است که مقادیر خروجی به دست آمده از شبکه هرچقدر که می‌توانند به مقادیر هدف نزدیک‌تر باشند.

۱-۴-۱- مسیر عقب رو [1] (پس انتشار)

هنگام آموزش، یک ورودی به شبکه داده شده و خروجی محاسبه می‌شود. خروجی حاصل شده با خروجی واقعی، برای محاسبه خطا یا زیان، مقایسه می‌شود. سپس خطای محاسبه شده برای به روزرسانی وزن‌های شبکه استفاده می‌شود. این روند تا زمانی که خطا به حداقل برسد، تکرار می‌شود. تابعی که برای محاسبه خطا یا زیان استفاده می‌شود ،تابع هزینه / زیان[2] نام دارد. و تعداد معدودی توابع زیان در دسترس است. و دو تابع زیان رایج در آخرین لایه از شبکه عصبی (لایه‌ای که برای دسته‌بندی مورد استفاده قرار می‌گیرد)، تابع زیان میانگین مربع خطا[3] و آنتروپی متقاطع[4] هستند. میانگین مربع خطا مدل شبکه عصبی را می‌توان به صورت زیر نشان داد.

$$\text{loss} = E(Z.\text{target}) \qquad (۱-۱)$$

تابع زیان E به صورت زیر تعریف می شود:

$$E(Z.\text{target}) = \frac{1}{n}\sum_{k=1}^{n}(Z^k - \text{target}^k)^2 \qquad (۱-۲)$$

[1] Backward Pass
[2] Cost/Loss Function
[3] Mean Squared Error(MSE)
[4] cross-entropy

Z^k خروجی k امین ورودی است توسط شبکه عصبی و $target^k$ که برچسب واقعی داده ورودی است، تولید شده‌اند. همچنین n نشان‌دهنده تعداد کلاس‌ها (برچسب‌ها) می‌باشد و $E(Z, \text{target})$ خطا/زیانی است که نشان می‌دهد تا چه حد پیش بینی شبکه از هدف واقعی دور است. هدف از آموزش به حداقل رساندن تابع زیان است و برای انجام این کار وزن‌های شبکه تا زمانی که مقدار کمینه بدست آید، به طور مداوم به روز می‌شوند. پس از محاسبه خطا، گرادیان تابع زیان نسبت به هر وزن از شبکه عصبی محاسبه می‌شود. سپس وزن‌های شبکه با گرادیان مربوطه به روز می‌شوند. این روند تا زمانی‌که خطای کل به حداقل برسد ادامه می‌یابد. برای به حداقل رساندن تابع زیان E، مشتق E نسبت به وزن w^i محاسبه می‌شود. تابع زیان E به عنوان ترکیب توابع در حالت سری تعریف می‌شود بنابراین به صورت زیر است.

(۱-۳)

$$E = \text{loss}(\text{Activation Function}\left(F6\left(F5\left(F4\left(F3\left(F2\left(F1(input)\right)\right)\right)\right)\right)\right)$$

یعنی تابع زیان، تابع فعال‌سازی را می‌سازد، که تابعی از لایه تمام متصل $F6$ است که به نوبه خود تابعی از لایه $F5$ قبلی و غیره است. تابع زیان E مشتق پذیر است و می‌توان نسبت به هر وزنی در هر لایه‌ای، با استفاده از قاعده زنجیره‌ای، مشتق آن را گرفت. در پس انتشار ابتدا وزن‌های لایه آخر و سپس دومین لایه آخر و غیره به روز می‌شوند. برای شروع مشتق تابع هزینه نسبت به وزن w_i^k در لایه آخر $F6$ با استفاده از قاعده زنجیره ای محاسبه می‌شود.

$$\frac{\partial E}{\partial w_i^k} = \frac{\partial E}{\partial Z^k} * \frac{\partial Z^k}{\partial F6^k} * \frac{\partial F6^k}{\partial w_i^k} \qquad (۱-۴)$$

- E تابع هزینه است.
- Z^k نشان دهنده خروجی تابع فعال‌سازی است.
- $F6^k$ خروجی آخرین لایه است.
- k شاخص نورون لایه خروجی است.
- w_i^k ، i امین وزن از k امین نورون لایه آخر است.

پس از یافتن مشتق $\frac{\partial E}{\partial w_i^k}$ ، وزن w_i^k به صورت زیر به روز می شود:

$$w_i^k = w_i^k - \mu \frac{\partial E}{\partial w_i^k} \quad (۱-۵)$$

که در آن w_i^k وزن به روز شده و μ نرخ یادگیری است.

۱-۵- توابع زیان و طبقه‌بندی کننده بیشینه هموار

یک تابع زیان برای محاسبه‌ی خطا (تفاوت بین برچسب پیش بینی و داده مرجع) در حین فرایند آموزش یک شبکه ژرف استفاده می‌شود. بسته به کاربرد، توابع زیان زیادی وجود دارند که می‌توانند در شبکه‌های یادگیری ژرف مورد استفاده قرار گیرند. به‌عنوان مثال زیان میانگین مربع خطا[1] (L2)، زیان آنتروپی متقاطع، زیان هینج[2] معمولا در مسئله کلاس‌بندی استفاده می‌شوند. زیان انحراف مطلق خطا (L1) برای مسئله رگرسیون مناسب است. در زیر برخی از توابع زیان رایج مطرح شده‌اند.

۱-۵-۱- زیان میانگین مربع خطا

متداول‌ترین تابع زیان در یادگیری ماشین، تابع زیان میانگین مربعات خطا می‌باشد. تابع میانگین مربعات خطا که همچنین به‌عنوان تابع زیان L2 شناخته می‌شود، مربع میانگین خطای E تمام خطاهای تک را محاسبه می‌کند و از رابطه زیر بدست می‌آید:

$$E = \frac{1}{n}\sum_{i=1}^{n} e_i^2 \quad (۱-۶)$$

که در آن e_i نشان دهنده تک خطای نورون خارجی i ام است و از رابطه زیر بدست می‌آید:

$$e_i = \text{target}(i) - \text{output}(i) \quad (۱-۷)$$

[1] Mean Squared Error
[2] Hinge Loss

در طی فرایند آموزش، یک تابع زیان در لایه خروجی برای محاسبه خطا استفاده می‌شود و مشتق (گرادیان) آن در جهت عقب رو شبکه منتشر می‌شود. سپس وزن‌های شبکه نسبت به گرادیان مربوطه به روزرسانی می‌شوند.

۱-۵-۲- زیان آنتروپی متقاطع

زیان آنتروپی متقاطع یکی دیگر از توابع زیانی است که بیشتر در مسائل رگرسیون و کلاس‌بندی استفاده می‌شود و از رابطه زیر بدست می‌آید:

$$H(y) = -\sum_i y_i' \log(y_i) \quad (۱-۸)$$

که در آن y_i' برچسب هدف است و y_i خروجی کلاس‌بندی کننده است. تابع زیان آنتروپی متقاطع زمانی که خروجی، یک توزیع احتمالاتی[1] باشد استفاده می‌شود و بنابراین تابع زیانی است که برای طبقه‌بندی کننده (کلاس‌بندی کننده) بیشینه هموار ترجیح داده می‌شود.

۱-۵-۳- طبقه‌بندی کننده بیشینه هموار

طبقه‌بندی کننده بیشینه هموار یک تابع ریاضی است که یک بردار ورودی را گرفته و بردار خروجی را در محدوده (۱-۰) تولید می‌کند، تا جایی که عناصر بردار خروجی تا ۱ جمع شوند. یعنی مجموع تمام خروجی‌های تابع بیشینه هموار شود. تابع بیشینه هموار به صورت زیر می باشد:

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \quad (۱-۹)$$

از آنجا که تابع بیشینه هموار توزیع احتمالی تولید می‌کند، برای کلاس‌بندی چندکلاسه در لایه نهایی شبکه‌های عصبی ژرف مفید است. حین انتشار معکوس، مشتق این تابع زیان با استفاده از قاعده خارج قسمت محاسبه می‌شود.

[1] Probability Distribution

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = \frac{(e^{y_i}\cdot\sum_{j=1}^{n} e^{y_j}) - (e^{y_i}\cdot e^{y_i})}{\left(\sum_{j=1}^{n} e^{y_j}\right)^2} \qquad (۱-۱۰)$$

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = \frac{(e^{y_i}\cdot\sum_{i}^{n} e^{y_i})}{\left(\sum_{j=1}^{n} e^{y_j}\right)^2} - \frac{(e^{y_i}\cdot e^{y_i})}{\left(\sum_{j=1}^{n} e^{y_j}\right)^2}$$

$$=> \frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = \frac{(e^{y_i})}{\sum_{j=1}^{n} e^{y_j}} - \frac{(e^{y_i}\cdot e^{y_i})}{\left(\sum_{j=1}^{n} e^{y_j}\right)^2}$$

$$=> \frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = \frac{(e^{y_i})}{\sum_{j=1}^{n} e^{y_j}} - \left(\frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}}\right)^2$$

$$=> \frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = S(y_i) - \left(S(y_i)\right)^2$$

$$=> \frac{\mathrm{dS}(y_i)}{\mathrm{d}y_i} = S(y_i)\cdot(1 - S(y_i))$$

$$\frac{(e^{y_i})}{\sum_{j=1}^{n} e^{y_j}} = S(y_i)$$

به طور مشابه، مشتق $S(y_i)$ نسبت به y_k به صورت زیر است:

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_k} = \frac{(0\cdot\sum_{j=1}^{n} e^{y_j}) - (e^{y_i}\cdot e^{y_k})}{\left(\sum_{j=1}^{n} e^{y_j}\right)^2} \qquad (۱-۱۱)$$

از آنجا که $\frac{\mathrm{d}e^{y_i}}{\mathrm{d}y_k} = 0$ است، زیرا e^{y_i} در اینجا ثابت است، داریم:

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_k} = -\frac{(e^{y_i} \cdot e^{y_k})}{(\sum_{j=1}^{n} e^{y_j})^2} \quad (۱-۱۲)$$

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_k} = -\frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}} \cdot \frac{e^{y_k}}{\sum_{j=1}^{n} e^{y_j}}$$

$$\frac{\mathrm{dS}(y_i)}{\mathrm{d}y_k} = -S(y_i) \cdot S(y_k)$$

سپس مشتق $S(\mathrm{y_i})$ در معادله فوق برای به روزرسانی وزن‌ها استفاده می‌شود. محاسبات گرادیان برای به روزرسانی وزن‌ها به گونه‌ای استفاده می‌شود که خطای کل حداقل شود. ساده‌ترین راه برای به حداقل رساندن خطا استفاده از تکنیک‌های مختلف بهینه سازی مبتنی بر گرادیان است که به طور خلاصه در بخش بعدی مطرح می‌شود.

۱-۶- تکنیک‌های بهینه سازی مبتنی بر گرادیان کاهشی

گرادیان کاهشی یک تکنیک بهینه سازی است که برای به حداقل یا حداکثر رساندن تابع هزینه با محاسبه گرادیان‌های لازم برای به روزرسانی مقادیر پارامترهای شبکه، استفاده می‌شود. انواع مختلف این تکنیک بهینه سازی، نحوه محاسبه به روزرسانی پارامترها را با استفاده از این گرادیان‌ها تعریف می‌کند. سه نوع گرادیان کاهشی رایج وجود دارد. این انواع در تعداد نمونه‌های آموزشی مورد استفاده برای محاسبه گرادیان، تفاوت دارند. این سه نوع در زیر شرح داده شده‌اند.