به نام خدا

گزارش تمرین 6

ائلدار صمزاده طریقت – 40033044

A. در دو حالت الف و ب چه تفاوت‌هایی در سرعت همگرایی و خطای نهایی مشاهده می‌کنید؟

مقایسه شبکه‌ها با ۴ و ۶۴ نورون در لایه مخفی نشان داد که شبکه ۴ نورونی خیلی زود در حوالی یک نقطه بهینه محلی گیر می‌کند و پس از چند Iteration ابتدایی دیگر قادر به کاهش ملموس خطای آموزش نیست، در نتیجه پس از ۱۰۰ Iteration روی داده‌های تست به MSE حدود ۱۳۳.۰۵۵۱ رسید. اما شبکه ۶۴ نورونی، ضمن اینکه کاهش هزینه را سریع‌تر و پیوسته‌تر ادامه داد، توانست الگوی داده را بسیار بهتر بیاموزد و در نهایت خطای تست را به حدود ۲۴.۳۰۳۶ کاهش دهد. بنابراین، افزایش تعداد نورون از ۴ به ۶۴ باعث تسریع همگرایی و دستیابی به خطای نهایی بسیار کمتر شده است؛ دلیل اصلی این بهبود، ظرفیت بالاتر مدل ۶۴ نورونی برای یادگیری پیچیدگی‌ها و نویزهای موجود در داده می‌باشد.

در آزمایش‌های انجام‌شده برای شبکه با یک لایه مخفی شامل ۱۶ نورون و تغییر تابع فعال‌سازی بین«tanh» ، «ReLU»و «sigmoid»مشاهده شد که سرعت کاهش تابع هزینه تا حد زیادی به نوع تابع فعال‌سازی وابسته است. وقتی از ReLU استفاده شد، مدل پس از حدود ۳۷ تکرار (iteration) به یک وضعیت پایدار رسید و دیگر مقدار خطا را چندان کاهش نداد؛ به‌عبارت دیگر، ReLUخیلی سریع مسیریابی را آغاز کرد اما خیلی زود در یک نقطه بهینه محلی متوقف شد. در مقایسه، با tanhروند کاهش هزینه کندتر ولی پیوسته‌تر بود و مدل تا حدود ۱۶۲ تکرار همچنان به کاهش خطا ادامه می‌داد. عملکرد sigmoid (logistic)از هر دو کندتر بود و حتی پس از رسیدن به بیشینه تعداد تکرار ۲۰۰ iteration هنوز به همگرایی کامل نرسیده بود؛ به این ترتیب، sigmoidبیشترین تعداد iteration را برای کاهش اولیه خطا نیاز داشت.

باtanh ، مدل توانست خطای تست را تا تقریباً ۲۳.۴۸ کاهش دهد که کمترین مقدار میان گزینه‌ها بود.

باsigmoid ، خطای نهایی تقریباً ۲۳.۸۲ باقی ماند که تفاوت قابل‌توجهی با tanh نداشت اما با توجه به اینکه نیاز به ۲۰۰ iterationداشت، مقرون‌به‌صرفه‌تر نیست.

در مقابل، ReLU پس از توقف زودهنگام در حوالی نقطه بهینه محلی، خطای تست را حدود ۵۸.۹۳ بر جای گذاشت که قابل‌مقایسه با حالت‌های tanh یا sigmoid نبود و نشان می‌داد علی‌رغم سرعت همگرایی بالاتر، کیفیت نهایی یادگیری برای این داده ساده با ReLU به‌شدت کاهش یافت.

B. آیا مدل دچار overfitting یا underfitting می‌شود؟

در آزمایش‌های صورت‌گرفته، مدل با ۴ نورون در لایهٔ مخفی به‌دلیل ظرفیت کم، نتوانست الگوی داده‌ها را به‌خوبی یاد بگیرد و هم در دادهٔ آموزش و هم در ولیدیشن خطای بالا داشت (یعنی تحت‌آموزش یا underfitting). با افزایش تعداد نورون به ۶۴، ظرفیت مدل افزایش یافت و منحنی خطا در آموزش و ولیدیشن تا انتها به‌صورت پیوسته کاهش یافت بدون اینکه خطای ولیدیشن پس از یک نقطه شروع به افزایش کند، بنابراین نشانه‌ای از بیش‌آموزش (overfitting) دیده نشد. همچنین در مورد تغییر تابع فعال‌سازی، وقتی از ReLU استفاده شد، مدل حتی با ۱۶ نورون هم دچار underfitting شد و نتوانست خطای

خود را به مقدار پایین‌تری برساند، اما با tanh و sigmoid مدل تا انتهای آموزش همچنان به کاهش خطا ادامه داد و نشانه‌های overfitting یا underfitting جدی از خود نشان نداد.

2- چرا در مدل‌های رگرسیون از تابع فعال‌سازی خطی در لایه‌ی خروجی استفاده می‌شود ولی در مدل‌های طبقه‌بندی از sigmoid یا softmax ؟

در مدل‌های رگرسیون هدف پیش‌بینی مقادیری عددی و پیوسته است (مانند قیمت، دما یا هر کمیت کمی دیگر)؛ بنابراین خروجی شبکه باید بتواند هر مقدار حقیقی (چه مثبت، چه منفی و چه بزرگ‌تر از یک) را تولید کند. تابع فعال‌سازی خطی در لایه‌ی خروجی دقیقاً همین امکان را فراهم می‌کند، چرا که به‌جای محدود کردن خروجی به بازه خاصی، اجازه انتقال مستقیم مقدار خروجی لایه قبل را می‌دهد. از سوی دیگر، وقتی از تابع خطای میانگین مربعات (MSE) استفاده می‌کنیم، مشتق خطی خروجی مستقیم و محاسبه‌پذیر است که یادگیری مدل را ساده‌تر و پایدارتر می‌کند.

در مقابل، در مسائل طبقه‌بندی ما نیاز داریم تا شبکه احتمال تعلق یک نمونه به هر یک از کلاس‌ها را ارائه دهد؛ به این معنی که خروجی هر نورون باید مقداری بین صفر و یک داشته باشد و برای طبقه‌بندی چندکلاسه نیز معمولاً مجموع مقادیر خروجی در لایه خروجی برابر با یک باشد. تابع سیگموید این ویژگی را برای طبقه‌بندی دودویی فراهم می‌کند و مقادیر را به بازه [۰, ۱] محدود می‌سازد، در حالی که تابع سافت‌مکس برای طبقه‌بندی چندکلاسه بکار می‌رود و علاوه بر محدود کردن هر مقدار به بازه [۰, ۱]، جمع کل خروجی‌ها را برابر با یک می‌کند. با این ساختار خروجی، می‌توان از تابع خطای آنتروپی متقابل (Cross-Entropy) بهره برد که برای مسائل دسته‌بندی بسیار مناسب است و مدل را برای پیش‌بینی توزیع احتمالاتی هدایت می‌کند.

3- اگر تابع فعال‌سازی در لایه‌های میانی از نوع sigmoid باشد، چرا ممکن است مدل در یادگیری داده‌های پیچیده دچار مشکل شود؟

استفاده از تابع سیگموید در لایه‌های میانی باعث می‌شود گرادیان‌ها در هنگام پس‌انتشار عبور کنند و به‌سرعت ناچیز (vanishing gradients) شوند، زیرا سیگموید برای ورودی‌های بزرگ و کوچک به سطوح اشباع نزدیک می‌شود و مشتق آن تقریباً صفر است. در نتیجه وزن‌ها عملاً به‌روزرسانی نمی‌شوند و شبکه نمی‌تواند روابط پیچیده را یاد بگیرد. علاوه‌براین، خروجی سیگموید همواره در بازه (۰,۱) قرار می‌گیرد و صفرمرکز نیست، به این معنا که میانگین فعال‌سازی‌ها از صفر فاصله دارد و باعث نوسان یا تأخیر در همگرایی گرادیانی می‌شود.

4- چگونه می‌توان با تغییر معماری شبکه (تعداد لایه‌ها یا نورون‌ها) مسئله‌ی بیش‌برازش یا زیر برازش را بهبود داد؟

برای کاهش زیربرازش می‌توان ظرفیت مدل را افزایش داد؛ به‌عنوان مثال با افزودن لایه‌های بیشتر یا افزایش تعداد نورون‌ها در هر لایه، شبکه توانایی یادگیری ویژگی‌های پیچیده‌تر را پیدا می‌کند و عملکرد بهتری روی داده‌های آموزشی و اعتبارسنجی خواهد داشت. اما اگر مدل دچار بیش‌برازش شده باشد، ظرفیت آن بیش از حد بزرگ است و الگوهای نویزی یا تصادفی داده‌های آموزشی را هم یاد می‌گیرد؛ در این حالت می‌توان با کاهش تعداد لایه‌ها یا نورون‌ها پیچیدگی شبکه را کم کرد تا جانشینی بهتری برای تقریب تابع هدف واقعی به دست آورد و تعمیم‌پذیری روی داده‌های جدید را بهبود بخشد. در عمل، معماری بهینه معمولاً با آزمایش و اعتبارسنجی متقابل (Cross-Validation) مشخص می‌شود؛ ابتدا یک ساختار کوچک‌تر (چند لایه و نورون کم) امتحان می‌کنیم تا از زیربرازش جلوگیری شود، سپس کم‌کم به‌اندازه مناسب نورون‌ها یا لایه‌ها اضافه می‌کنیم تا عملکرد روی داده‌های اعتبارسنجی بهبود یابد، بدون آنکه بیش‌برازش رخ دهد.