مبانی بینایی کامپیوتر

تمرین دوازدهم

نیکی نزاکتی
98522094

.1

Underfitting: هر گاه مدل نتواند دیتا را به خوبی آموزش ببیند به طوری که هم در دیتای آموزش و هم در دیتای آزمایش عملکرد خوبی نداشته باشد، می گوییم مدل دچار underfitting شده است. علت آن می تواند ساده بودن مدل و آموزش نیافتن به اندازه کافی باشد. این موارد می توانند با پیچیده و عمیق تر کردن مدل و افزایش زمان آموزش رفع شوند.

Overfitting: هر گاه مدل دیتا را به خوبی آموزش ببیند به طوری که در دیتای آموزش عملکرد خوبی داشته باشد ولی توانایی تعمیم نداشته باشد و در دیتای آزمایش عملکرد بدی داشته باشد، می گوییم مدل دچار overfitting شده است. این اتفاق به دلایلی مانند دیتاست کوچک، پیچیدگی زیاد مدل، و نویزی و غیر دقیق بودن داده ها اتفاق می افتد. برای رفع این مشکل می توان از بیشتر کردن دیتا، ساده تر کردن مدل، استفاده کردن از early stopping و regularization کمک گرفت.

.4

۱- اگر مقدار padding برابر با same باشد بعد از عمل کانولوشن ابعاد خروجی برابر با ابعاد ورودی خواهد بود اما اگر مقدار آن برابر با valid باشد یعنی هیچ padding ای اعمال نشده است.

۲- تابع فعال سازی کمک می کند تا مدل بتواند روابط غیر خطی را یاد بگیرد. اگر این تابع را نداشتیم، عمیق تر کردن شبکه هیچ کمکی به ما نمی کند زیرا روابط همچنان خطی باقی می ماند.

۳- پارامتر kernel_initializer به ما این امکان را می دهد تا با الگوریتم دلخواه بتوانیم وزن های اولیه را تعریف کنیم.

۴- در واقع این دو عمل برعکس هم عمل می کنند. Conv2D ابعاد تصویر را کم می کند ولی Conv2Dtranspose ابعاد تصویر را بزرگتر می کند که به این عمل upsampling نیز گفته می شود.

۵- در تابع double_conv_block یک ورودی x و تعداد فیلتر را به عنوان پارامتر می گیرد و دو لایه Conv2D با ابعاد 3x3 با padding با مقدار same و تابع فعال سازی relu و همچنین وزن های این دو لایه با الگوریتم he_normal تعریف می شوند. در تابع downsample_block فقط از تابع double_conv_block استفاده کرده و در ادامه با لایه maxpooling2D ابعاد تصویر را نصف می کند. در تابع upsample_block، ابتدا ابعاد ورودی x را با استفاده از Conv2Dtranspose بیشتر می کنیم و در

ادامه خروجی را با فیچر های کانولوشنی مجاور concat می کنیم و در نهایت خروجی نهایی را دوباره به تابع double_conv_block می دهیم.

۶- برای update کردن وزن های مدل نیاز به یک optimizer داریم تا با توجه به خروجی loss function آن ها را تغییر دهد.

۷- برای مشخص کردن optimizer و loss function از این تابع استفاده می کنیم.

۸- برای اینکه مسئله یک سوال classification است.

۹- برای اینکه از overfitting جلو گیری کنیم و همچنین در وقت صرفه جویی کنیم، از این تابع استفاده می کنیم. در این تابع اگر مدل در تعداد epoch مشخصی بهبود نیابد، آموزش متوقف می شود.

۱۰- در تابع کامپایل optimizer و loss function را مشخص می کنیم اما در تابع فیت، دیتای آموزش و تست و تعداد epoch و همچنین batch size و ... را مشخص می کنیم.

۱۱- برای اینکه نمی توانیم کل دیتاست را هم زمان آموزش دهیم، آن را به چند قسمت تقسیم می کنیم و به هر کدام از این قسمت ها یک batch گفته می شود. به یک بار دیدن تمام batch ها توسط مدل یک epoch می گوییم.