به نام خدا

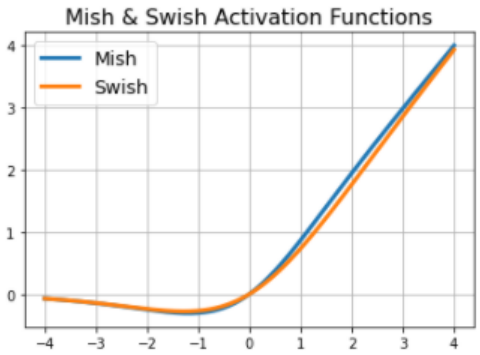
ملیکا احمدی رنجبر 97521036

1) توابع Swish، Mish

a. رابطه هر کدام به شکل زیر است:

Swish: $f(x) = x * \sigma(x), \sigma(x) = (1 + \exp(-x))^{-1}$

Mish: $f(x) = x * \tanh(\ln(1 + \exp(x)))$

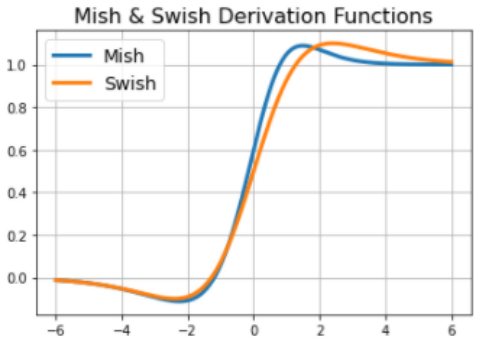


b. مشتق اول هر دو تابع:

Swish: $f'(x) = \sigma(x) + x \cdot \sigma(x)(1 - \sigma(x)) = \sigma(x) + x \cdot \sigma(x) - x \cdot \sigma(x)^2$

$= x \cdot \sigma(x) + \sigma(x)(1 - x \cdot \sigma(x)) => f(x) + \sigma(x)(1 - f(x))$

Mish: $f'(x) = \text{sech}^2(\ln(1 + \exp(x))) * x * \text{sigmoid}(x) + f(x)/x$



c. یک نکته، هزینه محاسباتی تابع RELU است، به دلیل عدم وجود exp. در تابع Sigmoid در اکثر مقادیر اشباع شده به دلیل حد بالای 1 (برای ورودی مثبت) اما در ReLU گرادیان تغییرات بهتری خواهد داشت (Gradient is able to flow)، همچنین Optimization را کند و کمی دشوار می‌کند(Sigmoid و Tanh).

برتری دو تابع Swish و Mish نسبت به ReLU مقادیر آنها برای ورودی منفی است، که اگرچه تغییرات بسیار کوچک است، **اما به طور مطلق صفر نیست** و تغییرات گرادیان را بهتر می‌کند. در واقع Smooth

و non-monotonic هستند. علاوه برا این، جریان بهتر و بیشتر گرادیان باعث افزایش Robustness نسبت به مقادیر اولیه و نرخ آموزش می‌شود. (با توجه به توضیحات قسمت Highlight شده مقاله)

d. این Parameter می‌تواند کمک کند فرم خروجی تابع کنترل شود، برای مثال اگر مقادیر بزرگ به Beta داده شود، Swish به سمت ReLU میل می‌کند (در عمل نیز تست شد). به دلیل راحتی تغییر دادن Activation Function برای برخی مسائل بسیار مفید است.

e. با مقایسه تصاویر و مطالعه مقاله متوجه می‌شویم که تابع Mish گردایان بسیار Smoothتری از Swish دارد. و در نتیجه Optimization بهتر دارد.

2) در این سوال داریم:

a. با توجه به نوع تابع ضرر، مقادیر Loss بسیار متفاوت است، با در نظر گرفتن این نکته که MSE مقادیر را به توان دو می‌رساند، Loss مقدار بسیار کوچک‌تری خواهد داشت (به همین دلیل معیار خوب و درستی برای مقایسه عملکرد نیست). این دلیل کوچک‌تر بودن مقدار Loss برای MSE است. همچنین دلیل این مقادیر آن است که طبق فرمول آن‌ها، و با توجه به اینکه در ابتدای آموزش احتمال پیشبینی صحیح 0.5 است، مقادیر 0.25 و 0.7 بدست آمده‌است.

b. دلیل اختلاف مقادیر Loss برای داده‌های آموزشی و تست در دو نمودار، فرمول هر کدام است (دلیل اولیه مطرح شده در قسمت قبل)، با توجه به اینکه مقادیر برای MSE بسیار کوچکتر هستند اختلاف کمتری نیز دارند، در حالی که برای Binary Cross Entropy به این شکل نیست.

c. زمانی که تا حد خوبی شبکه به Convergence رسیده باشد و تعدل بین Under fit و Over fit رعایت شود. برای مثال در حالت MSE با توجه به اینکه مقادیر Loss همچنان رو به کاهش است، در همان Epoch = 100 مناسب است (اگر همگرا شود)، و برای Binary Cross Entropy زمان Epoch = 80 مناسب است، چرا که مدل بعد از آن دچار افزایش Loss شده و احتمال Over fit شدن دارد، و پیش از آن احتمال آنکه به طور کامل آموزش ندیده باشد است. (در Epoch = 60 هم ممکن است اما مقدار Loss برای داده‌های آموزش زیاد است و در کل بین 60 تا 80 توقف آموزش خوب است).

3) بدترین عملکرد برای Alpha = 1 است، با توجه به اینکه نمودار خطی y = x خواهد شد و بنابراین هیچ حدی برای پایین نداریم. در حالات دیگر اختلاف بین Accuracyها بسیار کم بود، اما در نهایت Alpha = -0.5 بهترین عملکرد را برای test Data با اختلاف بسیار کمی با Alpha = -1, 0 داشت.

```
loss: 0.0523 - accuracy: 0.9836 - val_loss: 0.0512 - val_accuracy: 0.9840
loss: 2.3036 - accuracy: 0.1131 - val_loss: 2.3067 - val_accuracy: 0.1078
```

4) منابع:

a. https://numpy.org/doc/stable/reference/generated/numpy.divide.html