Algebraic expressions

Mick Bos en Youri de Vor

Uitleg code

Door de code te runnen, trainen wij het neurale netwerk met de testset, en testen het neurale netwerk met een validatieset

Voor alle functionaliteit wordt Keras gebruikt. Het model wat wordt gebruikt is een Keras **Sequential model** Met als layers een Flatten (input), Dense (Hidden layer), Dropout (anti overfitting) en een laatste Dense (output layer).

Hierna woden de prediction en de loss function gedefinieerd.

Vervolgens wordt het model gefit en getest. Via **.numpy()** Wordt een output gegenereerd.

Resultaten

Structuur

Wij hebben een aantal beslissingen gemaakt als het gaat om ons neurale netwerk. Zo hebben wij gekozen voor een hidden layer. Na meerdere keren testen zijn we tot de conclusie gekomen dat 1 hidden layer de beste resultaten gaf binnen acceptabele tijd. Dit was voor ons de beste oplossing uit meerdere testen die we gedaan hebben met verschillende nummers en aantal layers. Het gebruik van meerdere hidden layers leverde nauwelijks hogere slagingspercentages op, maar had wel significante invloed op de runtime. De accuraatheid schommelde continu rond de 0.97. En fluctueerde meer tussen herhalingen van dezelfde opstelling dan tussen onderlinge opstellingen.

De output layer is 10 omdat dit het aantal verschillende outputs zijn.

Na testen met verschillende breakout nummers, hebben we gekozen voor een breakout van 0.1. Dit doen we zodat het neurale netwerk niet alleen werkt met de testdata, maar ook met meerdere datapunten. Voor de input maken we gebruik van 784 neuronen, gezien de MNIST dataset uit afbeeldingen van 784 pixels bestaat.

Hidden Layer

Na een korte for loop van 1 tot en met 512 neuronen in de hidden layer kwamen we tot de conclusie dat er boven de 128 neuronen geen verbeteringen meer te zien waren. Hier zijn we dus gestopt met kijken naar de vorm van het netwerk en de activatiefunctie gaan testen.

Activatie

De activatie functie onderzoeken leverde eigenlijk de enige interessante resultaten op. Softsign leverde duidelijk hogere accuraatheid op dan de andere activatiefuncties met een accuraatheid van 0.9886 bij 15 epoch. Dit is het enige tot dusver gevonden verschil wat significant is over meer dan 5 runs.

Uit interesse hebben we de activatiefuncties met de grotere steigingen tussen 5 en 15 epoch nog een keer "met grote epoch" (dus 100) gedraaid. Dit leverde echter geen significant betere resultaten op.

activatie functie	**resultaat 5 epoch**	**resultaat 10 epoch**	**resultaat 15 epoch**	**resultaat 100 epoch**
elu	0.9761	0.9789	0.9790	
exponential	0.9743	0.9751	0.9778	
hard sigmoid	0.9705	0.9765	0.9785	0.9799
linear	0.9782	0.9774	0.9779	
relu	0.9790	0.9799	0.9807	
selu	0.9717	0.9785	0.9781	0.9816
sigmoid	0.9697	0.9765	0.9803	0.9785
softplus	0.9764	0.9794	0.9813	
softsign	0.9698	0.9764	0.9886	0.9800
swish	0.9791	0.9805	0.9802	
tanh	0.9757	0.9780	0.9797	