Kontinuierliche Filter

9. Februar 2017

1 Motivation

Für den Übergang von einer Schicht in die darauffolgende Schicht in einem neuronalen Netz gibt es eine entsprechende *Weight-Matrix*, die diese verbindet. Wenn wir eine Schicht mit x Neuronen mit einer Schicht mit y Neuronen *fully-connecten*, dann ergibt sich eine Weight-Matrix der Größe $x \times y$. Bei einer Faltung eines Bildes in den sogenannten *Convolutional Layern* verhält sich dies anders. Convolutions zeichnen sich durch zwei Besonderheiten aus:

1. lokale *Receptive Fields*
2. gemeinsame Gewichte und Biase für unterschiedliche Receptive Fields

In Convolution Neural Networks wird nicht jedes Neuron mit jedem Neuron der darauffolgenden Schicht verbunden, sondern ein kleines Fenster (z.B. 5×5) wird zu einem Neuron der darauffolgenden Schicht gemappt. Jedes Neuron der nächsten Schicht hat dann 5×5 Gewichte und einen Bias. Es ist wichtig anzumerken, dass jedes Fenster nicht unterschiedliche Gewichte generiert, sondern die gleichen. Das ermöglicht es, die gleichen Merkmale zu lernen, unabhängig der Position des Fensters im Bild. Die Gewichte werden deshalb auch oft *Filter* genannt. Jeder Filter W_{ij} hat damit genau $i \times j$ diskrete Werte, die ihn beschreiben.

Die Ausgabe der Aktivierungsfunktion lässt sich dann über

$$\sigma\left(b + \sum_{m=0}^{i} \sum_{n=0}^{j} w_{m,n} \cdot a_{x+m,y+n}\right) \tag{1}$$

beschreiben.

2 Kontinuierliche Filter

Anstatt dass ein Filter aus $i \times j$ diskreten Werten besteht, gibt es die Idee, diese Werte in einen kontinuierlichen Rahmen zu bringen. Das bedeutet, wir suchen eine (zweidimensionale) Filterfunktion, die diese Werte interpoliert. Mathematisch gesprochen suchen wir eine Filterfunktion W_l, sodass $W_l(i, j) = W_{ij}$

3 Anwendung auf Graphen

Wir lernen eine Weight-Matrix, die als eine kontinuierliche Gewichtsfunktion aufgefasst wird und falten über die direkte Nachbarschaft eines Knotens. Wählen wir zum Beispiel eine Weight-Matrix mit Maßen 3×3, so haben wir 9 Gewichte, über die die Filterfunktion interpoliert werden kann. Der Knoten und die Nachbarschaft müssen nun in den Kontext dieser Funktion gebracht werden. Dafür die Position der Knoten bzw. die Lage des Subgraphen (quasi ein Stern) in die Matrix projiziert, sodass der Wurzelknoten in der Mitte liegt und die Nachbarschaft den maximalen Raum der Matrix ausfüllen, aber skalierungs- und rotationsvariant bleiben.

Für diese Punkte ermitteln wir über bilineare Interpolation eine Position und bestimmen dessen Werte w_i in der Gewichtsfunktion.

Die Ausgabe der Aktivierungsfunktion lässt sich dann über

$$\sigma\left(b + \sum_n w_n \cdot a_n\right) \tag{2}$$

bestimmen.

Da ein Knoten unterschiedliche viele Nachbarn haben kann, kann man auch folgende Ausgabe evaluieren:

$$\sigma\left(b + \frac{\sum_n w_n \cdot a_n}{N}\right) \tag{3}$$