Es soll eine Methode (z.B. ein Optimierer) des Maschinellen Lernens näher beschrieben werden, mit

anschaulichen Beispielen erläutert werden und die Effekte der Methode, sowie der Einsatzbereich und die

Limitierungen dargelegt werden. Hinweise auf Alternativen und eine kurze Abgrenzung gehören ebenfalls

dazu.

## Was ist ein Optimierer?

//Was ist ein Modell? <br />

//Was ist eine Verlustfunktion? <br />

Mit Hilfe der Verlustfunktion können Modelle anhand ihrer Gewichte optimiert werden. Dieser Prozess kann manuell durchgeführt werden. Dabei werden die Gewichte so angepasst, dass die Verlustfunktion minimiert wird. Ein Optimierer automatisiert diesen Prozess. Er setzt die Ausgabe der Verlustfunktion und die Eingabe der Gewichte in Relation und optimiert die Gewichte so, dass die Verlustfunktion minimiert wird.

Ein beispielhafter Optimierer ist Gradient Descent. Gradient Descent optimiert Modelle in zwei Schritten:

1. Berechne für jedes Gewicht welchen Einfluss kleine Änderungen auf die Verlustfunktion haben. Dieser Einfluss auf die Verlustfunktion wird Gradient genannt. Ein Gradient ist ein Spaltenvektor, der die partiellen Ableitungen von f nach den Spaltenvektoren (also den Gewichten) enthält (Q:https://studyflix.de/mathematik/gradient-berechnen-1350).

2. Modifiziere die Gewichte anhand der Gradienten, sodass die Verlustfunktion kleiner wird.

Die Schritte eins und zwei werden so lange wiederholt, bis die Verlustfunktionen einen festgelegten minmalen Wert erreicht hat. Die Geschwindigkeit mit der Gradient Descent operiert, kann über die Lernrate angepasst werden. Die Lernrate bestimmt die Größe der Änderungen, die Gradient Descent an den Gewichten vornimmt. Diese wird in Kommazahlen angegeben und hat meistens Werte von 0.0001 bis 0.001. Die Gradienten werden mit der Lernrate multipliziert. Eine zu kleine Lernrate kann dazu führen, das der Gradient Descent Optimierer Probleme mit lokalen Minima hat. In diesem Fall geht der Optimierer davon aus, dass er den minimalsten Punkt erreicht hat, jedoch handelt es sich bei dem Punkt nur um ein lokales Minimum.

(Q: https://www.datarobot.com/blog/introduction-to-optimizers/)

Ein weiteres Problem für Optimierer kann Overfitting darstellen. Beim Overfitting wird das Model zu sehr auf die Daten mit denen es trainiert wurde angepasst. Dadurch kann das Modell mit den Trainingsdaten nahezu perfekt arbeiten, hat jedoch Probleme mit realen Daten. Overfitting tritt auf, wenn einzelne Gewichte einen zu hohen Einfluss auf das Model haben und dadurch das Ergebnis zu stark modifizieren. Optimierer können Overfitting durch das Hinzufügen von einem zusätzlichen Strafparameter zur Verlustfunktion vermeiden. Der Strafparameter bestraft Optimierer, wenn sie zu große Werte für Gewichte nehmen, auch wenn dadurch die Verlustfunktion kleiner wird. Dadurch verändern Optimierer mehrere Gewichte in kleinen Schritten, anstatt einzelne Gewichte in großen Schritten.

# Adam Optimizer

Der Adam Optimizer ist ein Algorithmus für Stochastische gradientenbasierte Optimierung erster Ordnung von stochastischen Zielfunktionen [siehe Abstract](https://arxiv.org/abs/1412.6980). 2014 wurde dieser auf der [ICLR 2015](https://arxiv.org/abs/1412.6980) Konferenz für Deep-Learning-Forscher von Jimmy Ba und diederik Kingma vorgestellt.

Dieser zeichnet sich durch folgende Punkte aus:

- einfache Implementierung

- rechnerische Effizienz

- geringer Speicherbedarf

- invariant zu diagonalen Neuskalierungen der Steigungen

- gut geeignet für Probleme in Bezug auf Daten oder Parameter

Dieser Optimierer ist weit verbreitet und wird beim Training von neuronalen Netzen eingesetzt. [Introduction to Optimizers, Adam](https://www.datarobot.com/blog/introduction-to-optimizers)

Mit diesem Optimizer beziehungsweise dieser Methode können individuelle adaptive Lernraten für unterschiedliche Parameter aus Schätzungen der ersten und zweiten Momente der Gradienten berechnet werden. Abgeleitet ist der Name Adam von 'adaptive moment estimation'. Diese Methode soll die Vorteile der beiden Methodiken AdaGrad (Duchi et al., 2011) sowie RMSProp (Tielemann & Hinton, 2012) kombinieren.

So gehören zu den Vorteilen von ADAM, dass

- die Größenordnungen der Parameteraktualisierungen invariant gegenüber der Skalierung der Gradienten sind,

- die Schrittweiten näherungsweise durch den Hyperparameter der SChrittweite begrenzt ist,

- kein stationäres Ziel erforderlich ist,

- mit spärlichen Gradienten arbeiten kann und

- es natürlich eine Form von Schrittgrößen-Annealing durchfüghrt.

![image.png](attachment:image.png)

In disem Ausschnitt sind wichtige Teile des Algorithmus als Pseudocode abgebildet. Dieser wird in der Folge näher beschrieben.