

Neuronale Netze – Teil 1

Erarbeitet von
Dr. Katja Theune

Lernziele	1
Inhalt	2
Einstieg	2
Neuronale Netze: Aufbau	2
Funktionsweise des Output-Layers	3
Aktivierungsfunktionen: Sigmoid	4
Aktivierungsfunktionen: Softmax	6
Abschluss	7
Weiterführendes Material	7
Disclaimer	7

Lernziele

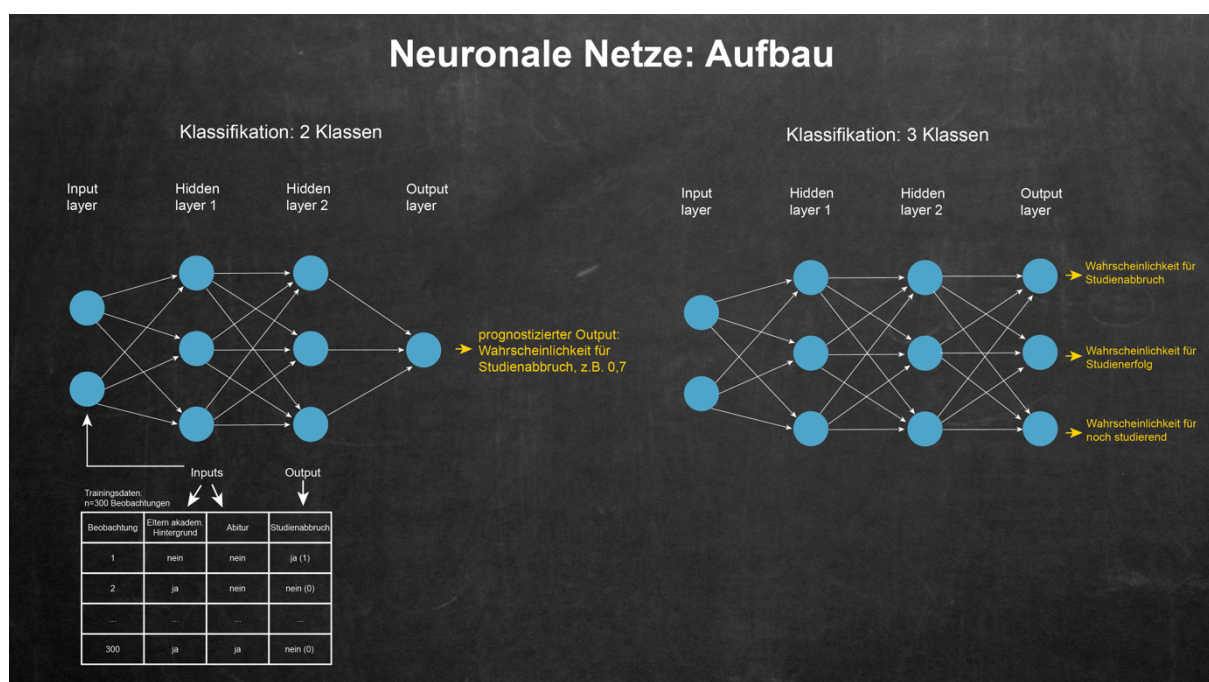
- Du kannst den Aufbau eines neuronalen Netzes, insbesondere des Output-Layers, erläutern
- Du kannst die allgemeine Vorgehensweise im Output-Layer erläutern
- Du kannst anhand eines einfachen Beispiels mittels der Sigmoid-Funktion den prognostizierten Output einer neuen Beobachtung bestimmen
- Du kannst anhand eines einfachen Beispiels mittels der Softmax-Funktion den prognostizierten Output einer neuen Beobachtung bestimmen

Inhalt

Einstieg

Künstliche neuronale Netze, oder englisch artificial neural networks, imitieren den Aufbau biologischer neuronaler Netze im menschlichen Gehirn. Allerdings nicht in seiner ganzen Komplexität. Aber ganz ähnlich wie im biologischen Gehirn, werden auch in den künstlichen neuronalen Netzen Informationen mit Hilfe der Neuronen weitergegeben. Aber wie genau funktioniert das?

Neuronale Netze: Aufbau



Hier sehen wir (siehe Grafik, linke Seite) den Aufbau eines neuronalen Netzes mit einem Input-Layer, zwei Hidden-Layern und einem Output-Layer. Wir beschäftigen uns hier nur mit sequenziellen feed-forward Modellen, d. h. Neuronen einer bestimmten Schicht erhalten Informationen von den Neuronen aus der vorangegangenen Schicht und geben sie an die Neuronen der nächsten Schicht weiter.

Wie viele Hidden-Layer wir verwenden, können wir selbst bestimmen. Auch die Anzahl an Neuronen in den Hidden-Layern können wir frei wählen. Im Input-Layer entspricht die Anzahl der Neuronen der Anzahl an Inputs. Wie der Output-Layer gestaltet ist, hängt vom Anwendungsfall ab.

Haben wir ein Klassifikationsproblem mit zwei sich ausschließenden Klassen, haben wir meist nur ein Neuron, da sich das Zutreffen der einen Klasse sozusagen aus dem Nicht-Zutreffen der anderen ergibt. Als konkretes Beispiel könnten wir uns vorstellen, dass wir prognostizieren wollen, ob ein Studienabbruch stattfindet oder nicht. Unsere Inputs im

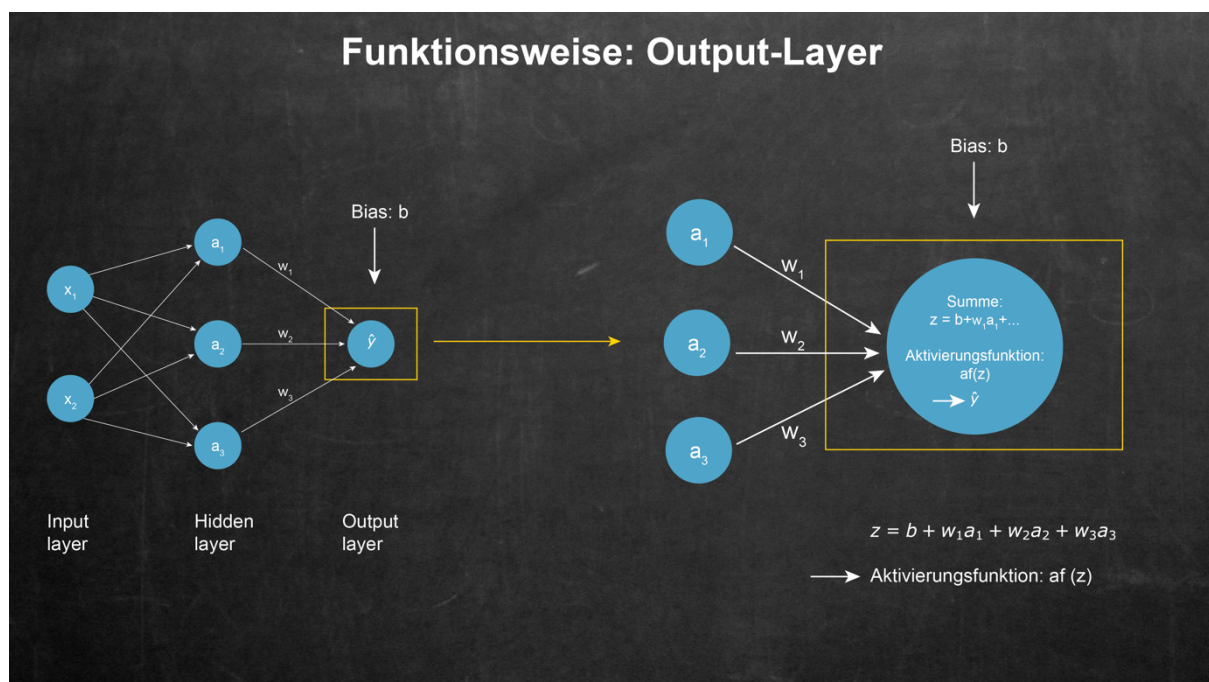
Input-Layer könnten hier z. B. der akademische Hintergrund der Eltern sein und ob man ein Abitur hat oder nicht. Sie werden durch die beiden Neuronen im Input-Layer repräsentiert. Unser Output in den Trainingsdaten sind die Klassen „Studienabbruch“ und „kein Studienabbruch“. Wir kodieren sie mit 1 und 0, was auch jeweils als die Wahrscheinlichkeit für einen Studienabbruch interpretiert werden kann. Der Output, der vom Netz im Output-Layer ausgegeben wird, wäre dann ebenfalls eine Wahrscheinlichkeit, zur Klasse „Studienabbruch“ zu gehören, z. B. 0,7. Die Wahrscheinlichkeit für die Klasse „kein Studienabbruch“ ergibt sich dann als 1 Minus dieser Wahrscheinlichkeit, hier also 0,3.

Bei einem Klassifikationsproblem mit mehreren Klassen, z. B. drei (siehe Grafik, rechte Seite), entspricht die Anzahl an Neuronen im Output-Layer üblicherweise der Anzahl an betrachteten Klassen. Wir könnten uns hier die drei sich gegenseitig ausschließenden Klassen „Studienabbruch“, „Studienerfolg“ und „noch studierend“ vorstellen. Wir haben dann also drei Neuronen im Output-Layer und für jede Klasse würde eine Wahrscheinlichkeit prognostiziert werden, die sich dann zu eins addieren.

Haben wir viele Hidden-Layer, sprechen wir von Deep Learning. Die Beziehung bzw. Vernetzung zwischen Inputs und Outputs ist hier sehr komplex, was das neuronale Netz sehr leistungsstark macht. Was in einem einzelnen Neuron passiert, ist aber eigentlich sehr einfach. Im Folgenden wollen wir uns das einmal speziell im Output-Layer ansehen.

Funktionsweise des Output-Layers

Wir schauen uns nun zunächst ein einfaches Netz für ein Klassifikationsproblem mit zwei Klassen an und betrachten die Funktionsweise des einzelnen Neurons in dem Output-Layer. Das allgemeine Prinzip lässt sich aber auf alle anderen Neuronen übertragen.



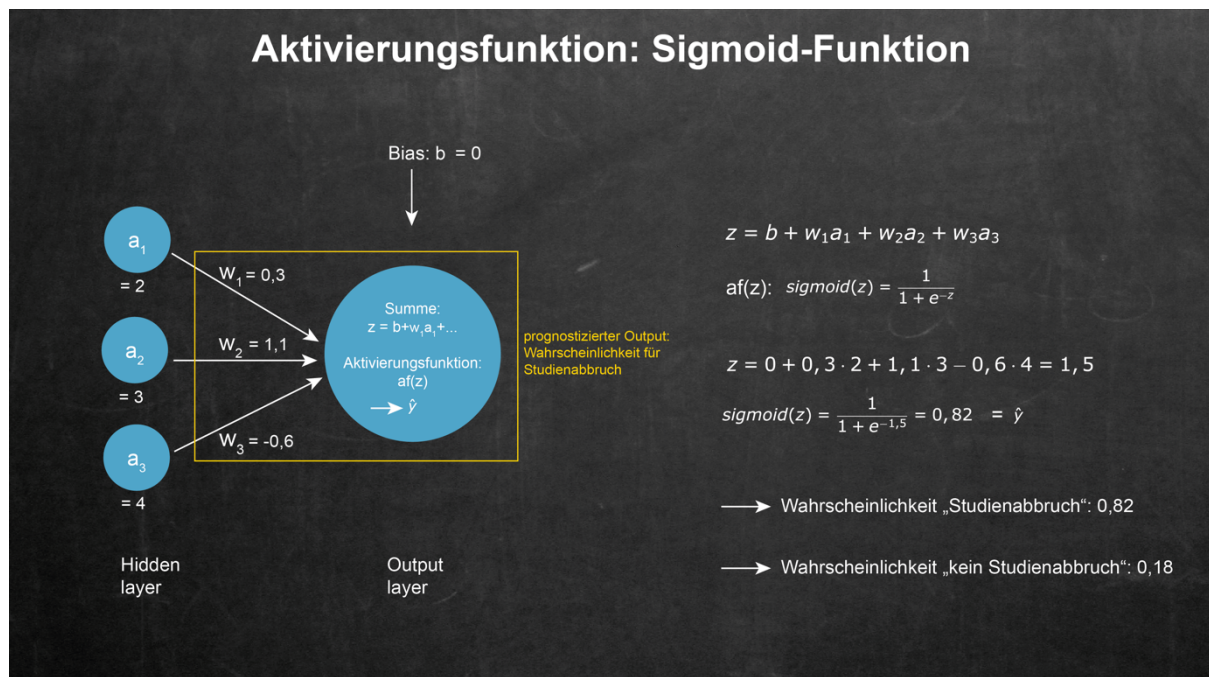
Einfach gesagt passiert hier nichts anderes, als das ankommende Informationen aus dem vorangegangenen Hidden-Layer verarbeitet werden. Zur Veranschaulichung ist die Notation hier vereinfacht dargestellt. Wir haben die drei Inputs aus dem Hidden-Layer a_1 , a_2 und a_3 und den prognostizierten Output $\hat{y}$ im Output-Layer. Zudem haben wir die Gewichte w_1 , w_2 und w_3 und einen Bias b (siehe Grafik).

Zunächst werden die ankommenden Informationen, hier unsere Inputs a , mit den Gewichten w multipliziert und aufsummiert. Die Summe nennen wir z .

Das entspricht jetzt einer multiplen linearen Regression: die Gewichte w nehmen die gleiche Funktion ein wie die Parameter β_1 usw. bei der Regression. Der Bias b entspricht dort dem vertikalen Achsenabschnitt β_0 . Ein Bias von ungleich 0 gewährleistet, dass auch wenn alle Inputs gleich 0 sind, das Neuron aktiviert werden kann.

Was das neuronale Netz aber jetzt so flexibel und leistungsstark macht, ist das Hinzufügen von nicht-Linearität durch die Transformation von z . Das passiert durch eine sogenannte Aktivierungsfunktion, hier mit af abgekürzt. Dadurch können komplexe Zusammenhänge in den Daten gelernt werden, die sich z. B. nicht mit einer einfachen Gerade beschreiben lassen. Der Name kommt wieder aus dem biologischen Kontext und beschreibt, ob ein Neuron aktiviert wird und Informationen weitergibt.

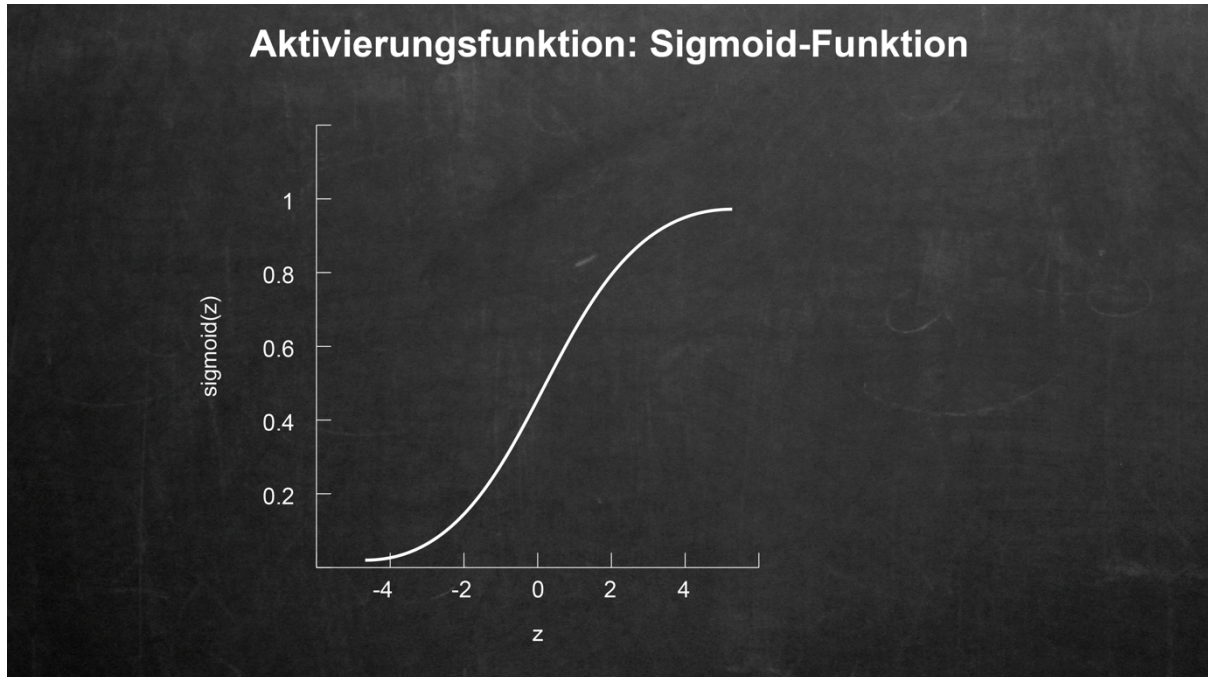
Aktivierungsfunktionen: Sigmoid



Es gibt verschiedene Aktivierungsfunktionen und die Wahl hängt von unserem konkreten Anwendungsfall ab. Wir bleiben zunächst bei unserem Zwei-Klassen-Problem und nehmen an, dass wir prognostizieren wollen, ob ein Studienabbruch stattfindet oder nicht. Dafür können wir die Sigmoid-Funktion als Aktivierungsfunktion verwenden. Sie ist nichts anderes

als die logistische Funktion. Unser Kürzel af für die Aktivierungsfunktion ersetzen wir hier also durch Sigmoid.

Hier sehen wir auch nochmal, dass sie eine S-Kurve darstellt.



Dadurch werden wie bei der logistischen Regression alle Werte von dem linearen Teil z in den Bereich von 0 bis 1 übertragen, was sich gut für Prognosen von Wahrscheinlichkeiten eignet und insbesondere für unseren Fall einer Klassifikation mit zwei Klassen.

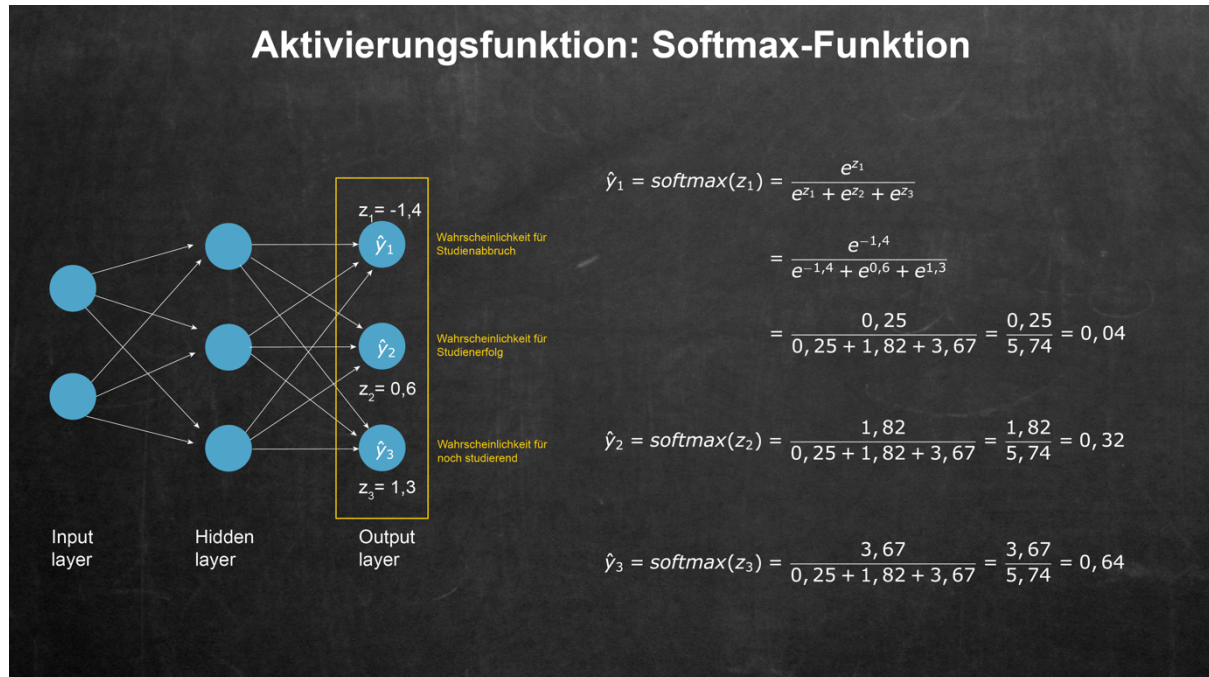
Die Werte der Inputs a für das Neuron im Output-Layer sollen hier für eine Beobachtung 2, 3 und 4 betragen. Die Gewichte w betragen 0.3, 1.1 und -0.6. Den Bias setzen wir der Einfachheit halber gleich 0. Wir erhalten dann eine Summe z von 1.5. Wenden wir darauf unsere Aktivierungsfunktion an, erhalten wir einen Wert von 0.82, was dann auch unsere Prognose $\hat{y}$ ist (siehe Grafik).

Mit einer Wahrscheinlichkeit von 0.82 wird also das Studium abgebrochen und mit einer Wahrscheinlichkeit von $1 - 0.82$, also 0.18 nicht. Jetzt könnten wir der Beobachtung die Klasse mit der höchsten Wahrscheinlichkeit zuordnen, hier dann der Klasse „Studienabbruch“.

Das einzelne Neuron verhält sich hier also wie bei einer logistischen Regression. Aber durch das Zusammenfügen vieler Neuronen in mehreren Layern kann ein neuronales Netz verschiedenste komplexe Zusammenhänge in den Daten erlernen.

Aktivierungsfunktionen: Softmax

Haben wir mehr als zwei sich gegenseitig ausschließende Klassen, eignet sich die Sigmoid-Funktion nicht als Aktivierungsfunktion im Output-Layer. Wir verwenden dafür hier die Softmax-Funktion.



Kommen wir zu unserem Beispiel mit den drei Klassen „Studienabbruch“, „Studienerfolg“ und „noch studierend“ zurück.

Sagen wir, wir hätten für eine Beobachtung die drei z-Werte für jede Klasse von $z_1 = -1.4$, $z_2 = 0.6$ und $z_3 = 1.3$. Dann berechnet sich mit der Softmax-Funktion die prognostizierte Wahrscheinlichkeit für Klasse 1 $\hat{y}_1$ so (siehe Grafik).

Im Zähler wird hier nur die gerade betrachtete Klasse berücksichtigt, wir benötigen also nur z_1 . Im Nenner wird die Summe über alle Klassen gebildet. Für Klasse 1, hier „Studienabbruch“, erhalten wir dann eine Wahrscheinlichkeit von 0.04.

Das machen wir jetzt für alle drei Klassen. Für Klasse 2, hier „Studienerfolg“ erhalten wir eine Wahrscheinlichkeit von 0.32. Und für Klasse 3 „noch studierend“ eine Wahrscheinlichkeit von 0.64.

Die Summe dieser drei Wahrscheinlichkeiten ergibt dann 1. Wir würden jetzt diese Beobachtung zu der Klasse mit der höchsten Wahrscheinlichkeit zuordnen, hier also „noch studierend“.

Abschluss

Wir kennen nun den Aufbau neuronaler Netze und die Funktionsweise im Output-Layer. Mit Hilfe von Aktivierungsfunktionen wie der Sigmoid- oder der Softmax-Funktion lässt sich ein neuronales Netz auf verschiedenste Problemstellungen anwenden. Durch das Verknüpfen mehrerer Neuronen in verschiedenen Layern ist das neuronale Netz in der Lage, sehr komplexe Zusammenhänge in Daten zu erkennen. Dadurch ist es sehr leistungsstark und ein sehr beliebtes Verfahren im maschinellen Lernen bzw. Deep Learning.

Weiterführendes Material

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining: Concepts and Techniques* (3. Auflage). Morgan Kaufmann.

James, G., Witten, D., Hastie, T., & Tibshirani, R., & Tylor, J. (2023). *An Introduction to Statistical Learning - with Applications in Python*. Springer.

Lantz, B. (2015). *Machine learning with R* (2. Auflage). Packt Publishing Ltd, Birmingham.

3Blue1Brown: Neural networks

https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi

Disclaimer

Transkript zu dem Video „Prognosemodelle (Klassifikation und Regression): Neuronale Netze – Teil 1“, Dr. Katja Theune.

Dieses Transkript wurde im Rahmen des Projekts ai4all des Heine Center for Artificial Intelligence and Data Science (HeiCAD) an der Heinrich-Heine-Universität Düsseldorf unter der Creative Commons Lizenz [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/) veröffentlicht. Ausgenommen von der Lizenz sind die verwendeten Logos, alle in den Quellen ausgewiesenen Fremdmaterialien sowie alle als Quellen gekennzeichneten Elemente.