Overfitting – Teil 4

Batch-Normalisierung

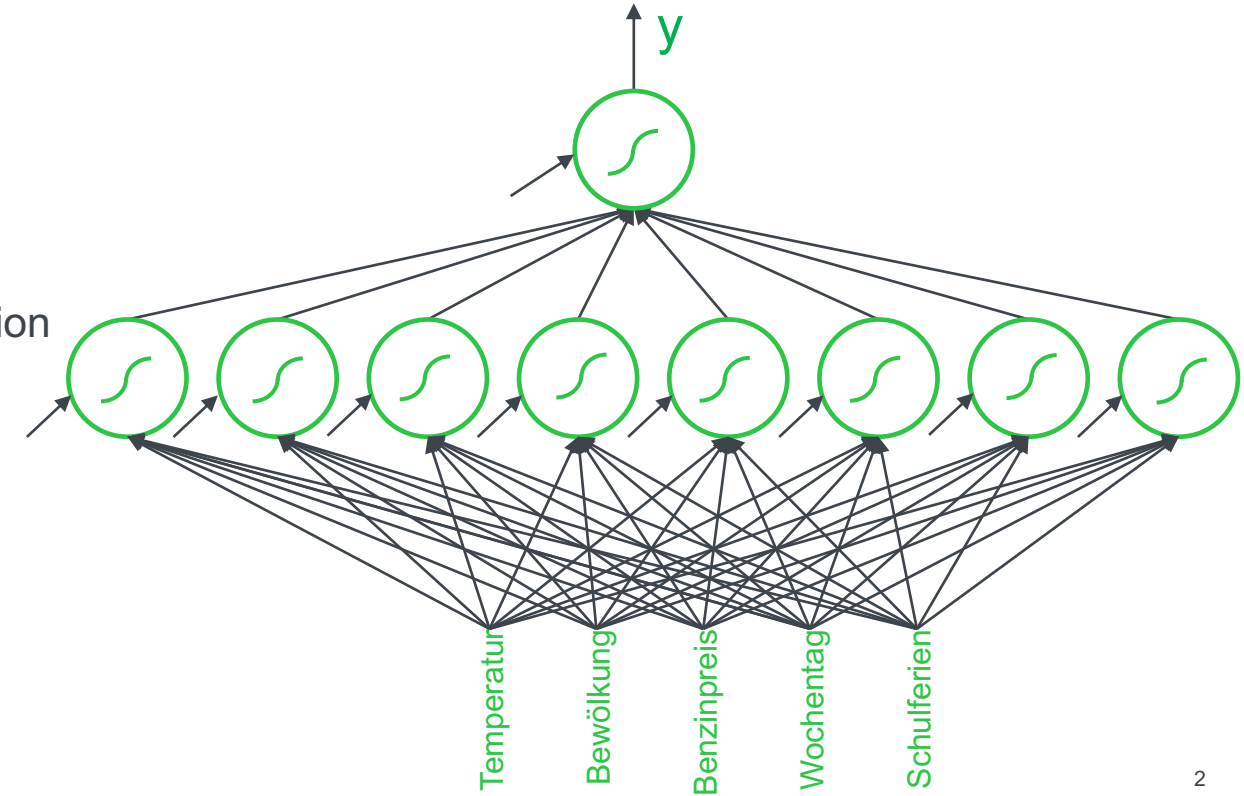
INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

INNOVET
GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

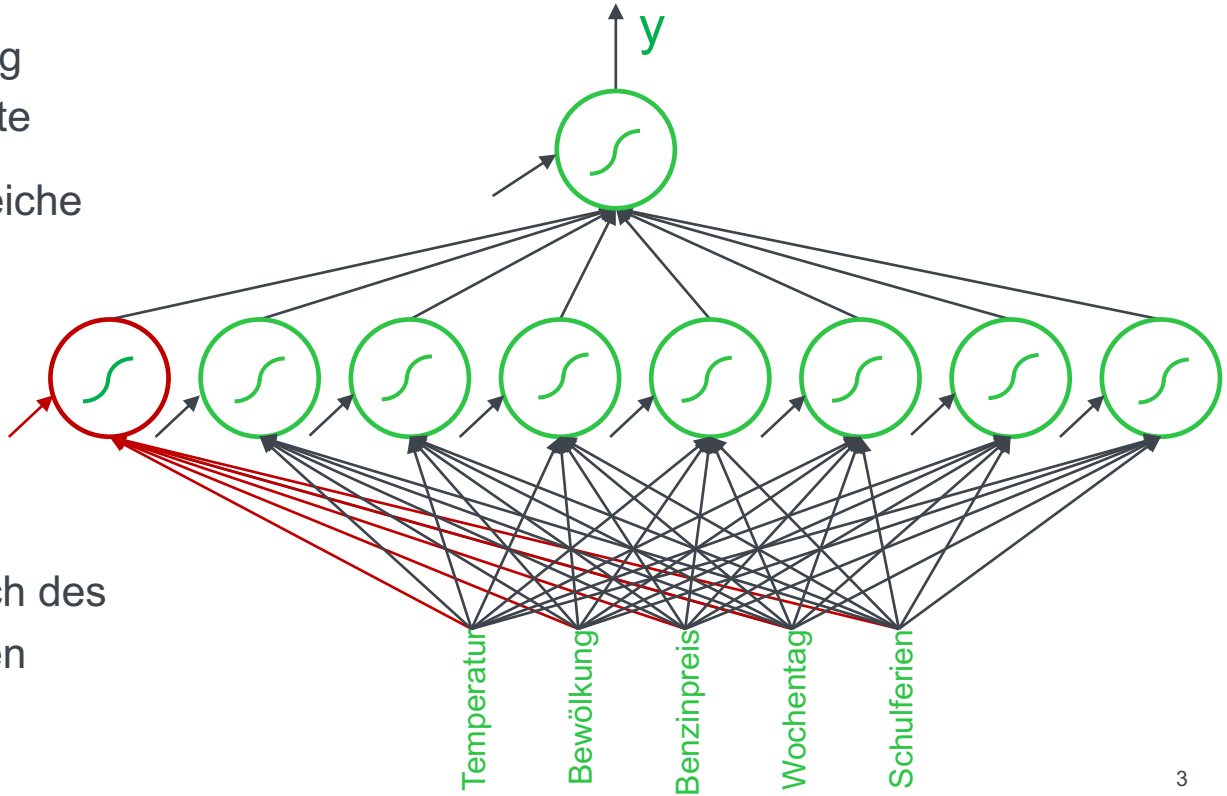
Das bekannte Beispiel

- Achtung, normalerweise dürfte ein so winziges Netzwerk kein Overfitting verursachen
- Trotzdem hier zur Illustration



Das Problem der „Internen Kovariatenverschiebung“

- In jeder Iteration im Training verändern sich die Gewichte
- Selbst wenn die Wertebereiche der Inputs gleich bleiben (normalisierte Daten!), verändert sich im Training der Wertebereich des Logits
- Und damit der Wertebereich des Inputs der darüberliegenden Schicht



Warum heißt das interne Kovariatenverschiebung?

- Kovariatenverschiebung normalerweise
  - Verteilung der (Input-)Daten während der Anwendung unterscheidet sich von der Verteilung der Trainingsdaten
  - Dazu später mehr
- Interne Kovariatenverschiebung*
  - Verteilung der Inputs zu Teilen des Modells (z.B. zu Schichten) verändert sich während des Trainings
  - Intuitiv: dadurch fangen höherliegende Schichten in jeder Iteration wieder „von vorne“ an, verlängert Training

* Sergey Ioffe und Christian Szegedy (2015), „Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift“

Während des Trainings verändern sich durch die Anpassung der Gewichte in den unteren Schichten die Verteilungen der Inputs in darüber liegenden Schichten. Dies wird oft als interne Kovariatenverschiebung bezeichnet.

Es wird angenommen, dass interne Kovariatenverschiebung den Trainingsprozess verlangsamt.

Idee der Batch Normalisierung

- Bringe beim Training die Inputs einer Schicht immer wieder in den selben Bereich
- Normalisierung wie bei der üblichen Datenvorbereitung: Standardisierung!
  - D.h. von den Inputs Mittelwert abziehen und durch Standardabweichung teilen
  - Dadurch Werte immer um Null, mit einer Standardabweichung von 1

Problem

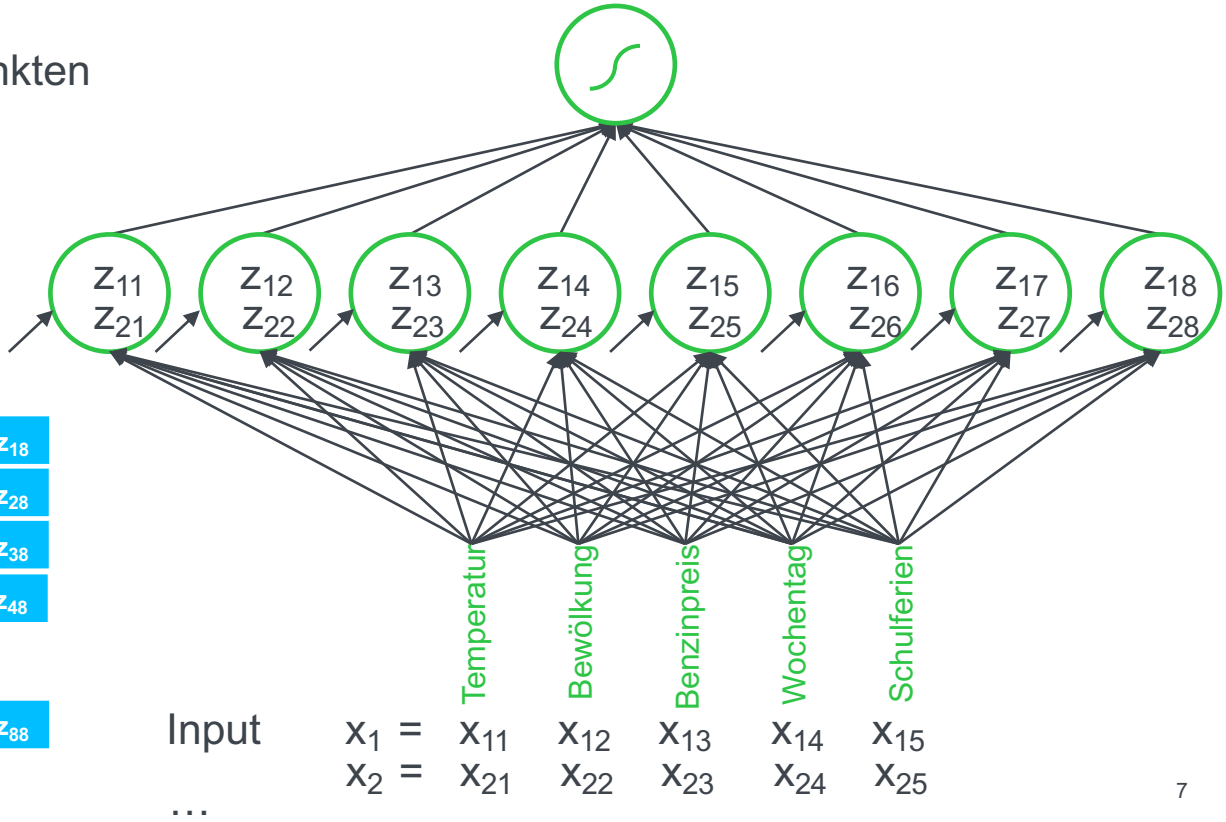
Woher kennen wir Mittelwerte und Standardabweichung der Inputs, die im Netzwerk entstehen?

Lösung

Trainingsdaten batchweise verarbeiten, Mittelwerte und Standardabweichung über den ganzen Batch berechnen

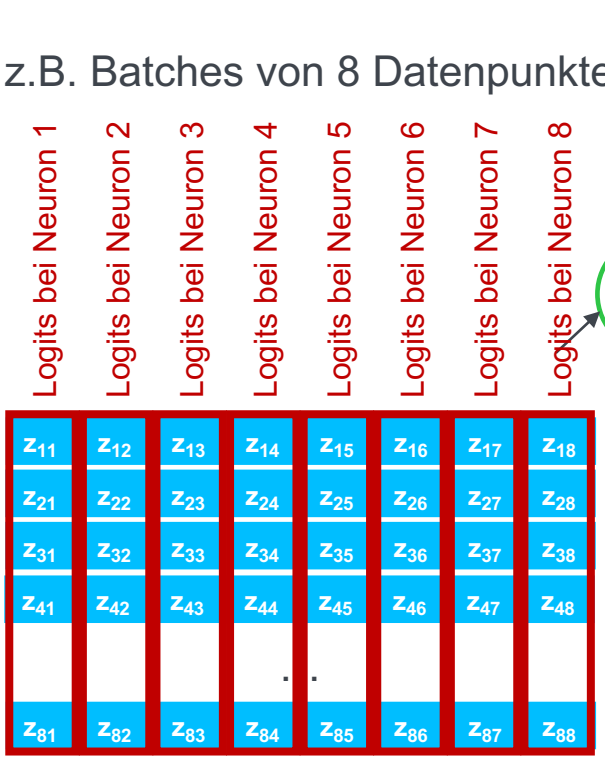
Logits eines Batches

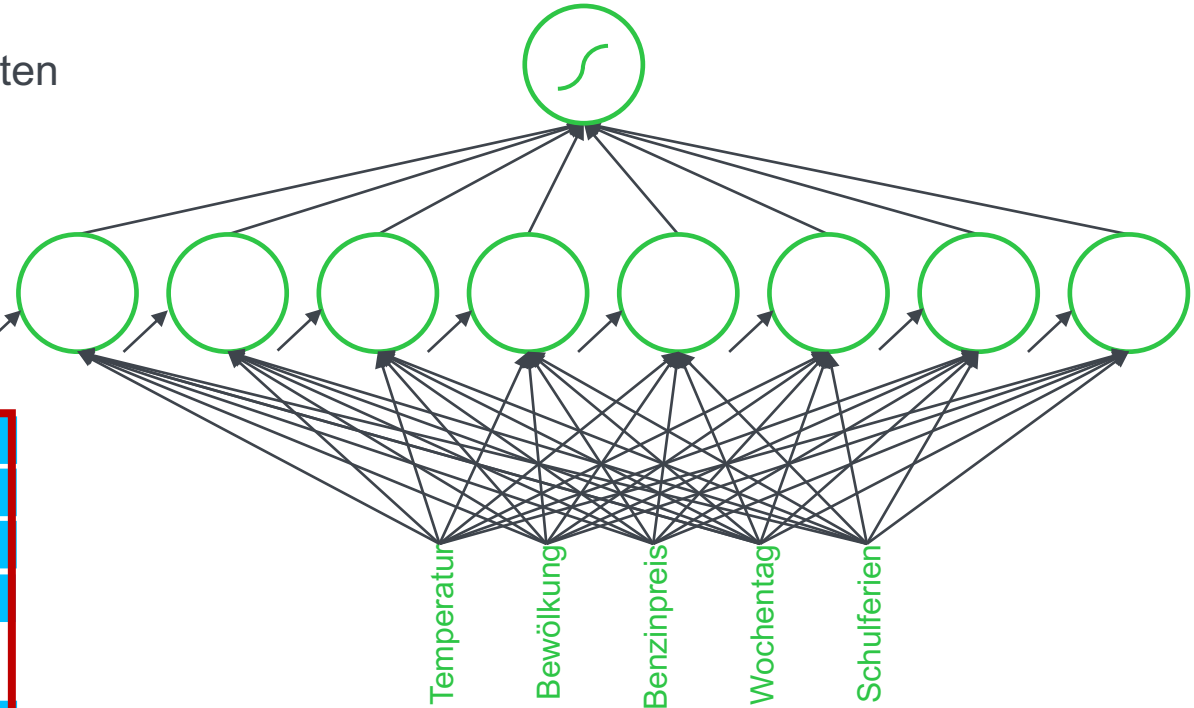
z.B. Batches von 8 Datenpunkten



Logits eines Batches

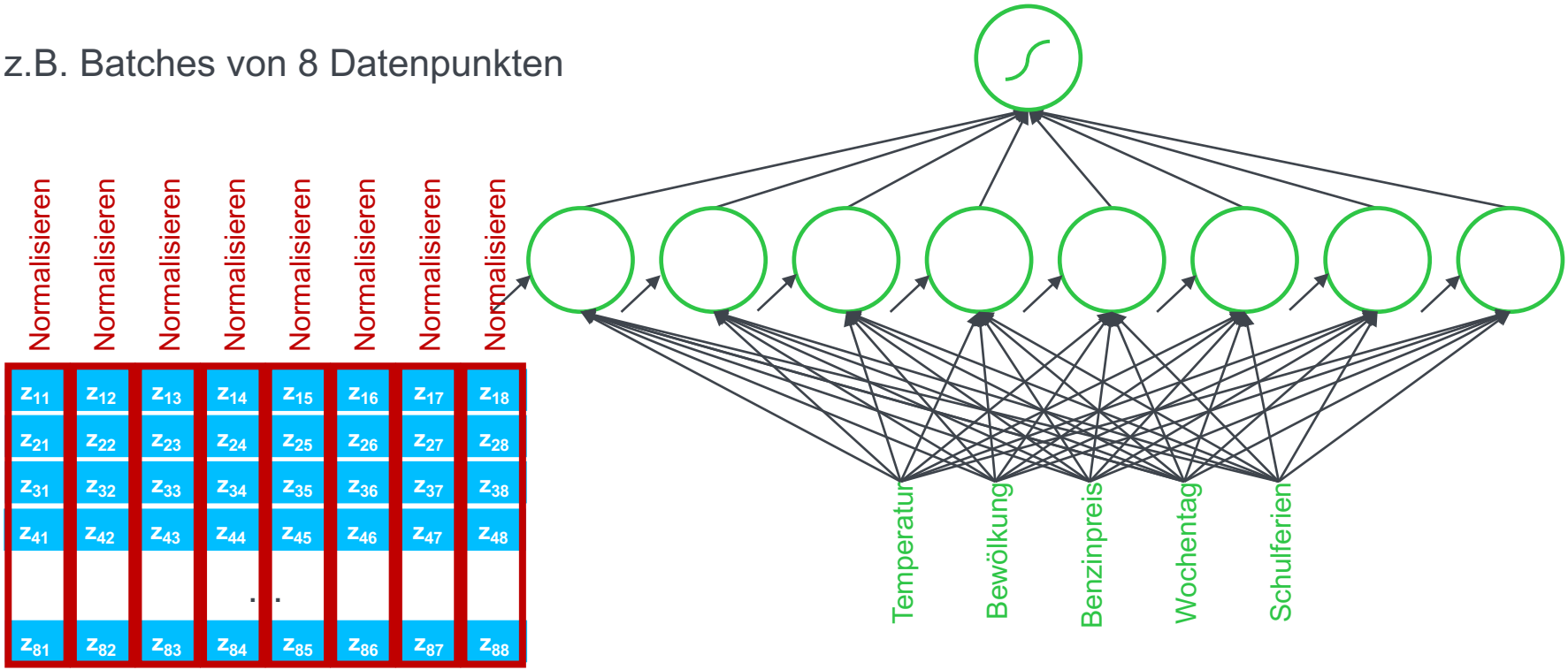
z.B. Batches von 8 Datenpunkten





Normalisierung der Logits

z.B. Batches von 8 Datenpunkten



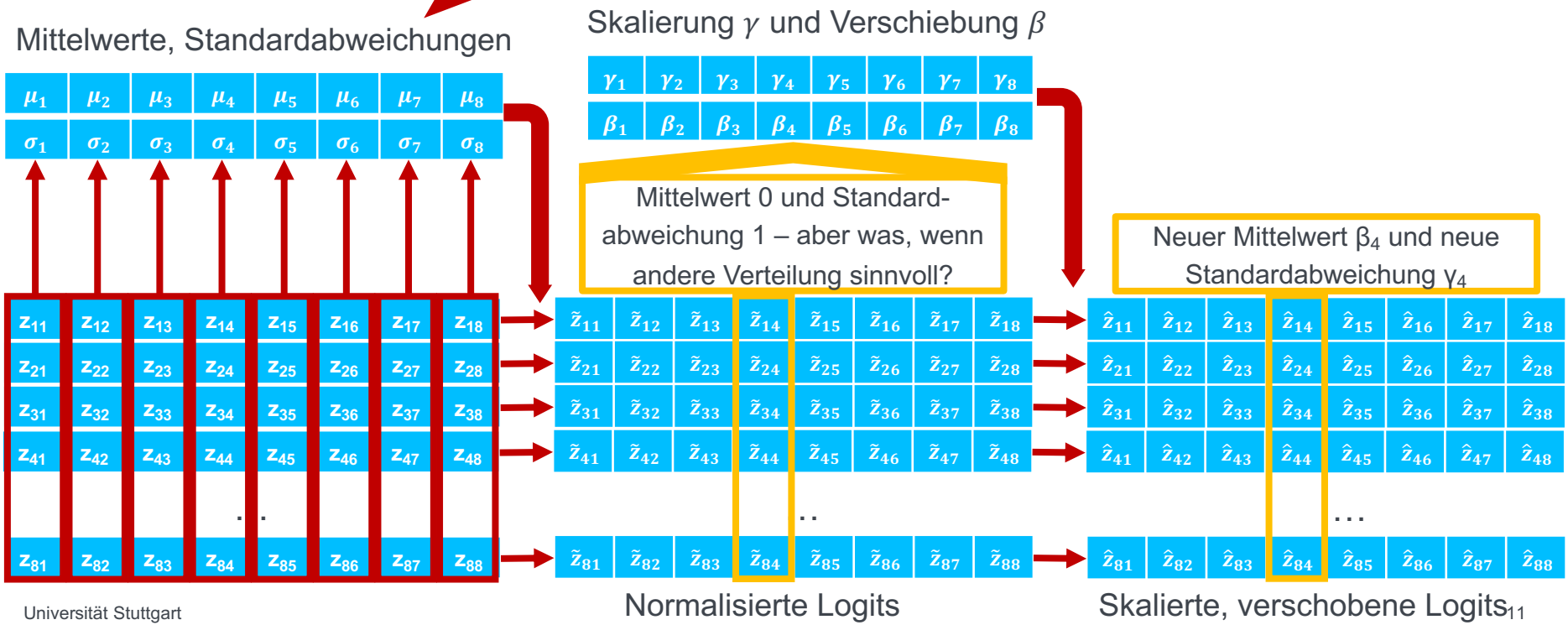
Ziel der Batchnormalisierung ist es, die Wertebereiche der Inputs in ein Neuron in jeder Trainingsiteration durch Normalisierung ähnlich zu halten.

Dadurch soll die interne Kovariatenverschiebung verhindert werden.

Batchnormalisierung

Berechnet durch Statistik über die Inputs

Wird direkt trainiert: lernbare Parameter



Bei der Batchnormalisierung werden die Logits für jedes Neuron der betreffenden Schicht batchweise standardisiert.

Dadurch ergeben sich über den Batch jeweils ein Mittelwert von 0 und eine Standardabweichung von 1.

Um zu erlauben, dass sich die Verteilungen trotzdem gezielt verändern können, werden die standardisierten Logits in jedem Neuron anschließend noch mit einem Parameter γ multipliziert und mithilfe eines weiteren Parameters β verschoben.

Dadurch liegt der neue Mittelwert bei β und die neue Standardabweichung bei γ.

Berechnungen bei der Batchnormalisierung

Mittelwerte, Standardabweichungen

μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8
σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8

Skalierung γ und Verschiebung β

γ_1	γ_2	γ_3	γ_4	γ_5	γ_6	γ_7	γ_8
β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8

$$\mu_1 = \frac{1}{8}\sum_i z_{i1}$$

$$\sigma_1 = \sqrt{\frac{1}{8}\sum_i (z_{i1}-\mu_1)^2}$$

$$\tilde{z}_{11} = \frac{z_{11}-\mu_1}{\sigma_1}$$

$$\hat{z}_{11} = \gamma_1 \cdot \tilde{z}_{11} + \beta_1$$

z_{11}	z_{12}	z_{13}	z_{14}	z_{15}	z_{16}	z_{17}	z_{18}
z_{21}	z_{22}	z_{23}	z_{24}	z_{25}	z_{26}	z_{27}	z_{28}
z_{31}	z_{32}	z_{33}	z_{34}	z_{35}	z_{36}	z_{37}	z_{38}
z_{41}	z_{42}	z_{43}	z_{44}	z_{45}	z_{46}	z_{47}	z_{48}
…							
z_{81}	z_{82}	z_{83}	z_{84}	z_{85}	z_{86}	z_{87}	z_{88}

$\tilde{z}_{11}$	$\tilde{z}_{12}$	$\tilde{z}_{13}$	$\tilde{z}_{14}$	$\tilde{z}_{15}$	$\tilde{z}_{16}$	$\tilde{z}_{17}$	$\tilde{z}_{18}$
$\tilde{z}_{21}$	$\tilde{z}_{22}$	$\tilde{z}_{23}$	$\tilde{z}_{24}$	$\tilde{z}_{25}$	$\tilde{z}_{26}$	$\tilde{z}_{27}$	$\tilde{z}_{28}$
$\tilde{z}_{31}$	$\tilde{z}_{32}$	$\tilde{z}_{33}$	$\tilde{z}_{34}$	$\tilde{z}_{35}$	$\tilde{z}_{36}$	$\tilde{z}_{37}$	$\tilde{z}_{38}$
$\tilde{z}_{41}$	$\tilde{z}_{42}$	$\tilde{z}_{43}$	$\tilde{z}_{44}$	$\tilde{z}_{45}$	$\tilde{z}_{46}$	$\tilde{z}_{47}$	$\tilde{z}_{48}$
…							
$\tilde{z}_{81}$	$\tilde{z}_{82}$	$\tilde{z}_{83}$	$\tilde{z}_{84}$	$\tilde{z}_{85}$	$\tilde{z}_{86}$	$\tilde{z}_{87}$	$\tilde{z}_{88}$

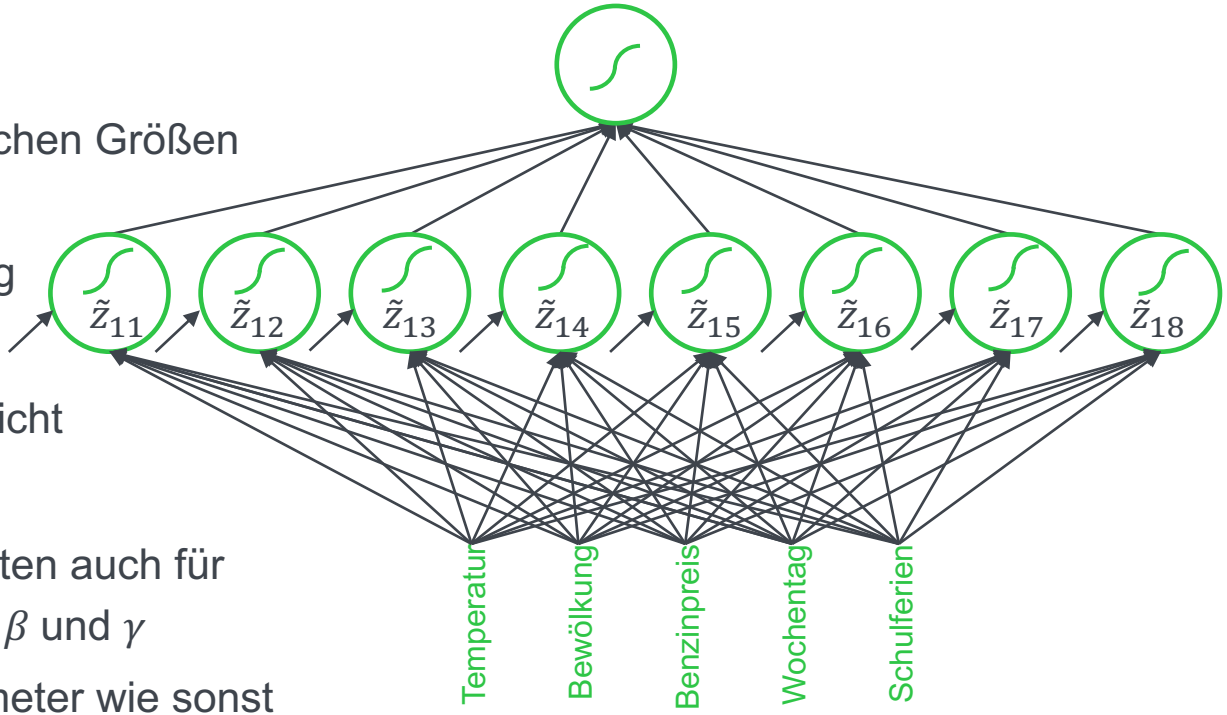
Normalisierte Logits

$\hat{z}_{11}$	$\hat{z}_{12}$	$\hat{z}_{13}$	$\hat{z}_{14}$	$\hat{z}_{15}$	$\hat{z}_{16}$	$\hat{z}_{17}$	$\hat{z}_{18}$
$\hat{z}_{21}$	$\hat{z}_{22}$	$\hat{z}_{23}$	$\hat{z}_{24}$	$\hat{z}_{25}$	$\hat{z}_{26}$	$\hat{z}_{27}$	$\hat{z}_{28}$
$\hat{z}_{31}$	$\hat{z}_{32}$	$\hat{z}_{33}$	$\hat{z}_{34}$	$\hat{z}_{35}$	$\hat{z}_{36}$	$\hat{z}_{37}$	$\hat{z}_{38}$
$\hat{z}_{41}$	$\hat{z}_{42}$	$\hat{z}_{43}$	$\hat{z}_{44}$	$\hat{z}_{45}$	$\hat{z}_{46}$	$\hat{z}_{47}$	$\hat{z}_{48}$
…							
$\hat{z}_{81}$	$\hat{z}_{82}$	$\hat{z}_{83}$	$\hat{z}_{84}$	$\hat{z}_{85}$	$\hat{z}_{86}$	$\hat{z}_{87}$	$\hat{z}_{88}$

Skalierte, verschobene Logits

Wie geht es nach der Normalisierung weiter?

- Beim Forward Pass
  - Berechnung der statistischen Größen für die Normalisierung
  - Skalierung, Verschiebung
  - Aktivierung
  - Weiter zur nächsten Schicht
- Bei der Back Propagation
  - Berechnung der Gradienten auch für die lernbaren Parameter β und γ
  - Anpassung dieser Parameter wie sonst



Die batch-normalisierten Logits werden anschließend wie üblich durch die Aktivierungsfunktion verändert, bevor sie als Input in die darüberliegende Schicht gehen.

Effekte von Batch Normalisierung

Beschleunigung des Trainings

- Eigentlicher Zweck
- Empfehlung daher:
  - Lernrate erhöhen
  - Bei Nutzung von Learning Rate Decay die Lernrate schneller verkleinern

Bei Nutzung von Learning Rate Decay – hierbei verkleinert man systematisch die Lernrate im Lauf des Trainings

Regularisierung

- Dropout nicht mehr nötig
- kleinere Werte für L2-Regularisierung

- Batch Normalisierung als Technik zur Vermeidung von Overfitting

INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

Dr. Antje Schweitzer

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung
Institut für Software Engineering

IHK Industrie- und Handelskammer Reutlingen
Reutlingen | Tübingen | Zollernalb

IHK Region Stuttgart

IHK Industrie- und Handelskammer Karlsruhe

LMU Ludwig-Maximilians-Universität München

GEFÖRDERT VOM
INNOVET
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

Lizenzbestimmungen

Quellenverzeichnis

Titelfoto: Hannah Troupe (https://unsplash.com/de/@htroupe) auf Unsplash (https://unsplash.com/de/fotos/flachfokusfotografie-des-audiomischers-wL_w5t_OKtI), Flachfokusfotografie des Audiomischers, lizenziert unter Unsplash-Lizenz (https://unsplash.com/license). Bildausschnitt verändert.