

Unsupervised Deep Embedding for Clustering Analysis

Bachelorseminar Data Mining

Lukas Mahr

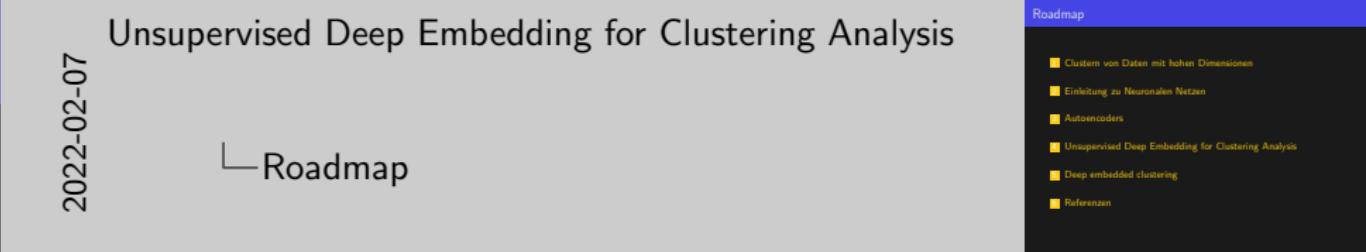
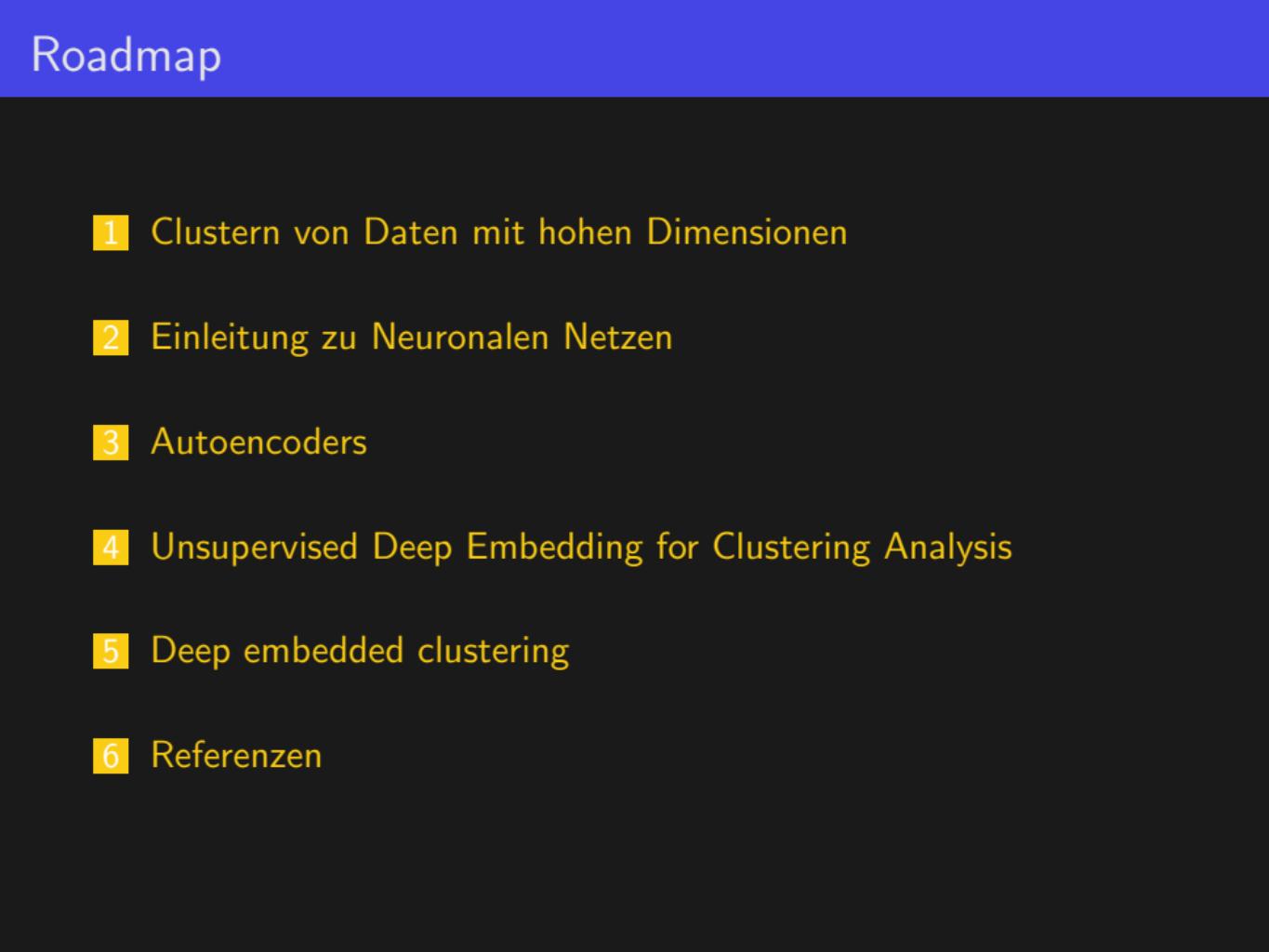
Ludwig-Maximilians-Universität München

Unsupervised Deep Embedding for Clustering Analysis

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis
Bachelorseminar Data Mining

Lukas Mahr
Ludwig-Maximilians-Universität München



■ Problem

■ Gaussian Mixture Models, KMeans

- Abstandsmetriken sind beschränkt auf den ursprünglichen Datendimensionen
- unwirksam wenn Datendimension hoch sind[1]

■ Variationen von KMeans für Daten mit hohen Dimensionen

- limitiert zu linearen Embeddings[2]

■ Spectral clustering

- Quadratische oder Super-quadratische Komplexität

■ Idee

■ Neuronalesnetzwerk zum reduzieren der Dimensionen

- nicht lineares mapping

■ Clustern der reduzierten Daten

- einfaches Clustering möglich, da Dimensionen reduziert

■ Verbessern des NN und der Cluster durch Backpropagation

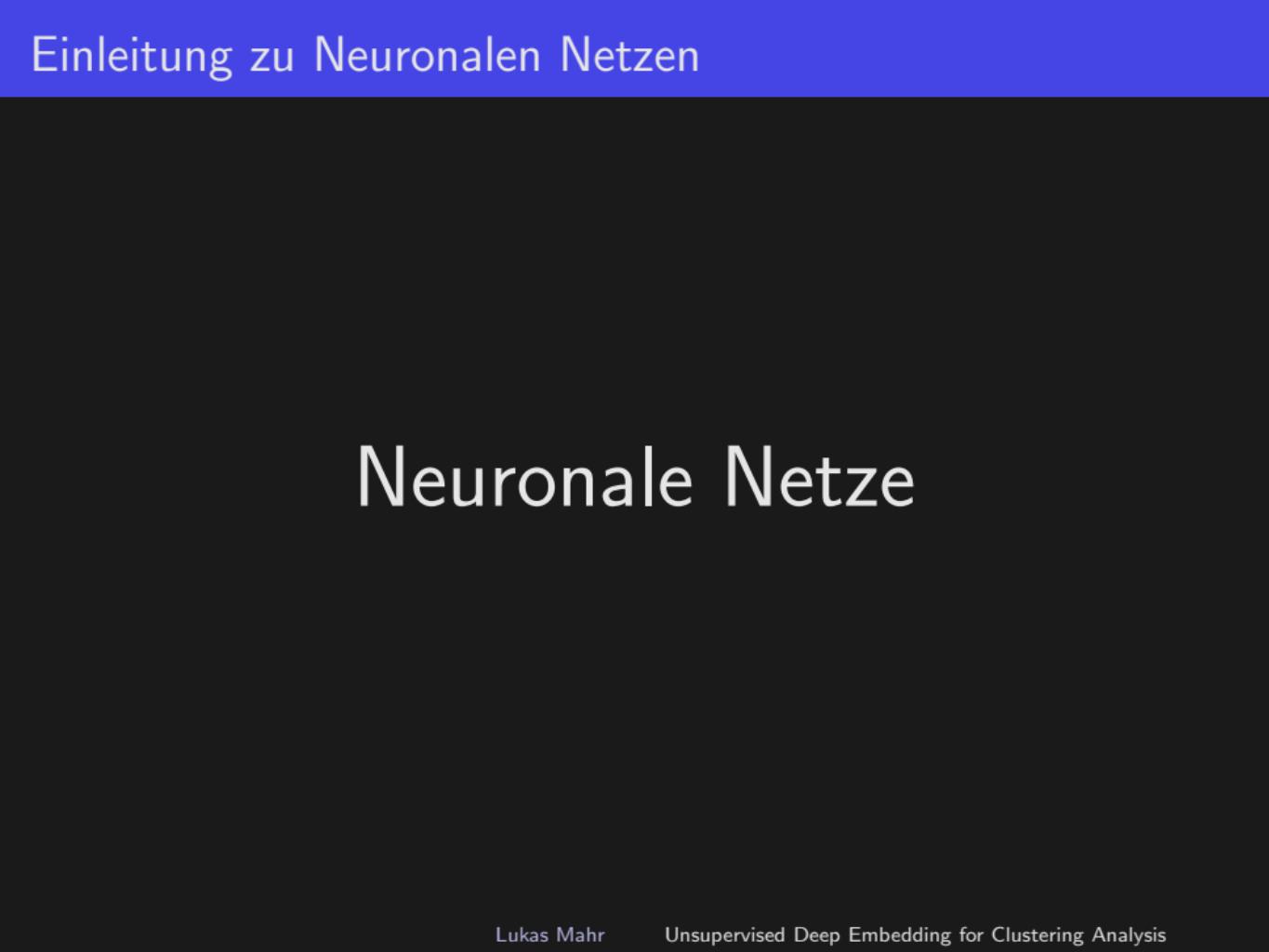
└ Clustern von Daten mit hohen Dimensionen

└ Clustern von Daten mit hohen Dimensionen

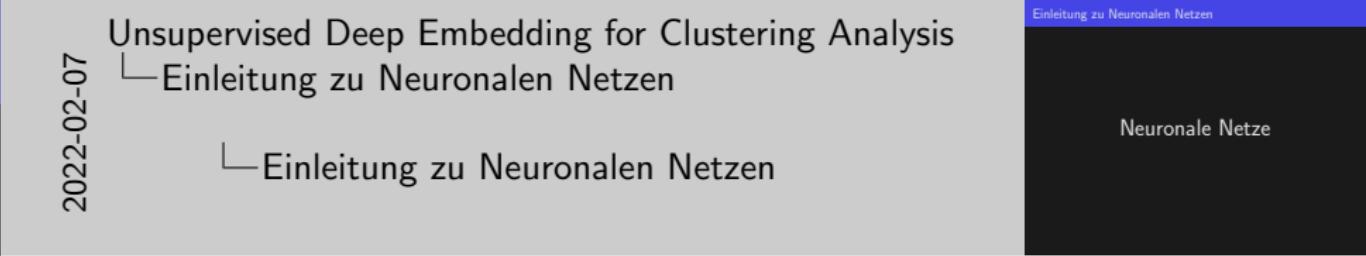
2022-02-07

viele Daten Punkte viele Distanzen zu berechnen schwierig zu visualisieren
ohne die Dimensionen zu reduzieren Komplexität von Kmeans die exponentiell ansteigt

- Problem
 - Gaussian Mixture Models, KMeans
 - Abstandsmetriken sind beschränkt auf den ursprünglichen Dimensionen
 - unwirksam wenn Dimensionen hoch sind[1]
 - Variationen von KMeans für Daten mit hohen Dimensionen
 - limitiert zu linearen Embeddings[2]
 - Spectral clustering
 - Quadratische oder Super-quadratische Komplexität
- Idee
 - Neuronalesnetzwerk zum reduzieren der Dimensionen
 - nicht lineares mapping
 - Clustern der reduzierten Daten
 - einfaches Clustering möglich, da Dimensionen reduziert
 - Verbessern des NN und der Cluster durch Backpropagation



Neuronale Netze



2022-02-07

Unsupervised Deep Embedding for Clustering Analysis

└ Einleitung zu Neuronalen Netzen

└ Einleitung zu Neuronalen Netzen

Einleitung zu Neuronalen Netzen

Neuronale Netze

Einleitung zu Neuronalen Netzen

Nicht-lineare statistische Modelle zur
Informationsverarbeitung

Informationsverarbeitung umfasst hierbei unter anderem

Klassifikation

Prognosenerstellung

Units der Neuronalen Netze angelehnt an Neuronen

Inputs zusammenfassen

Mit Schwellenwert vergleichen bzw. aktivieren

Verbindungen zwischen Units angelehnt an Synapsen

Gewichtung mit verstärkender oder schwächender Wirkung

Unsupervised Deep Embedding for Clustering Analysis

└ Einleitung zu Neuronalen Netzen

└ Idee

└ Einleitung zu Neuronalen Netzen

2022-02-07

Einleitung zu Neuronalen Netzen

Nicht-lineare statistische Modelle zur
Informationsverarbeitung
Informationsverarbeitung umfasst hierbei unter anderem
Klassifikation
Prognosenerstellung
Units der Neuronalen Netze angelehnt an Neuronen
Inputs zusammenfassen
Mit Schwellenwert vergleichen bzw. aktivieren
Verbindungen zwischen Units angelehnt an Synapsen
Gewichtung mit verstärkender oder schwächender Wirkung

wofür braucht man neuronale Netze

- Klassifizierung von Daten
- Prognose eines bestimmten Wertes
- Neuronen an Neuronen in unserem Gehirn angelehnt
- Verbindungen zwischen Künstlichen Neuronen an Synapsen angelehnt
- supervised learning = überwachtes lernen
- man besitzt Daten mit Beschriftungen (labels)
- unsupervised learning nicht überwachtes lernen
- unbeschriftete Daten, also es sind keine labels vorhanden

Einleitung zu Neuronalen Netzen

Künstlichen Neurons

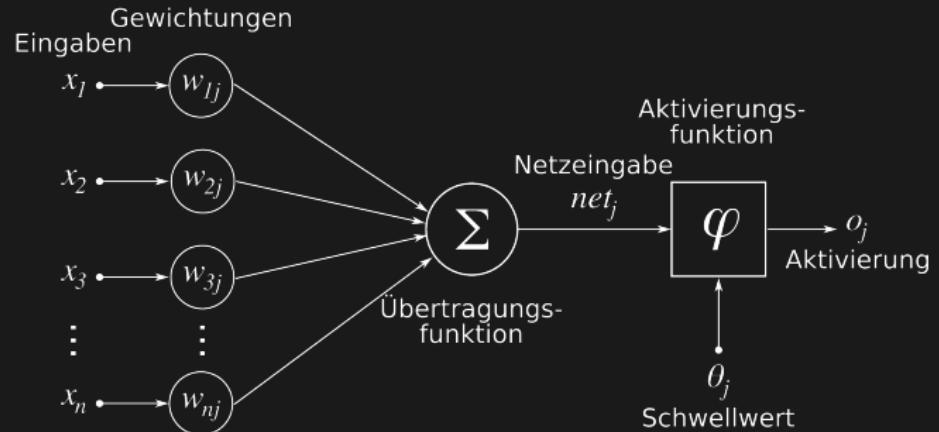


Figure: Darstellung eines künstlichen Neurons mit seinen Elementen

https://de.wikipedia.org/wiki/Datei:ArtificialNeuronModel_deutsch.png

Unsupervised Deep Embedding for Clustering Analysis

- └ Einleitung zu Neuronalen Netzen
 - └ Künstliches Neuron
 - └ Einleitung zu Neuronalen Netzen

2022-02-07



x_1, \dots, x_n sind die input variablen, jede der Eingabe variablen besitzt ein Gewicht, w_{1j}, \dots, w_{nj} . Diese werden Multipliziert und davon dann die summe berechnet. Hier die Übertragungsfunktion. Dazu wird ein Bias, in dem Fall der Schwellenwert gerechnet. Als letztes gibt es noch die Aktivierungsfunktion die meistens einen Wert zwischen 0 und 1 zurückgibt. Das ist dann der input für das nächste Neuron.

Einleitung zu Neuronalen Netzen

Layer/Schichten

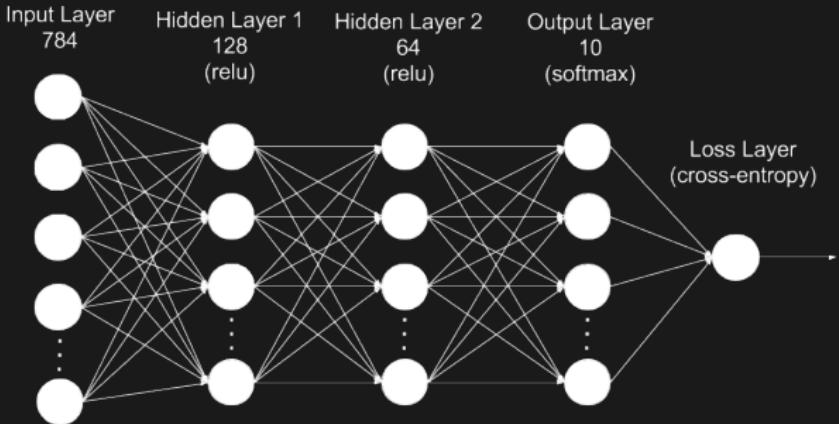
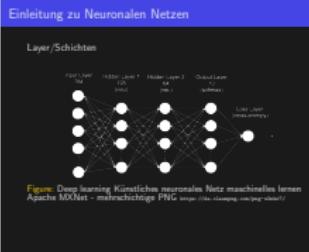


Figure: Deep learning Künstliches neuronales Netz maschinelles lernen
Apache MXNet - mehrschichtige PNG <https://de.cleanpng.com/png-x3zkr7/>

Unsupervised Deep Embedding for Clustering Analysis

- └ Einleitung zu Neuronalen Netzen
 - └ Layer/Schicht
 - └ Einleitung zu Neuronalen Netzen

2022-02-07



Layer/Schicht sind mehrere Neuronen die mit allen Neuronen des nächsten Layer/Schicht verbunden sind. Alle Neuronen in einem Layer haben die gleiche Aktivierungsfunktion. Hidden Layer haben meistens die Aktivierungsfunktion rectified linear, da diese recht einfach und schnell zu berechnen ist. Das outputlayer hat meistens eine etwas kompliziertere Funktion wie softmax oder sigmoid. Abhängig von der Aufgabe des Netzwerkes. Letztes Layer hier direkt mit dem Loss

Aktivierungsfunktionen

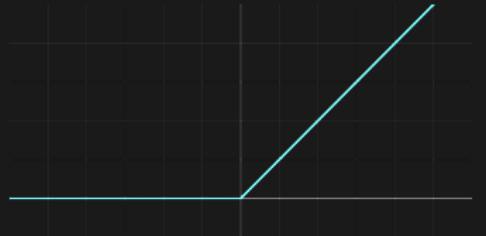


Figure: Rectifier-Aktivierungsfunktion

https://de.wikipedia.org/wiki/Datei:Activation_rectified_linear.svg

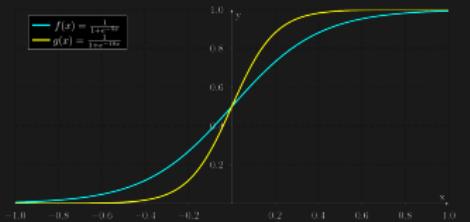
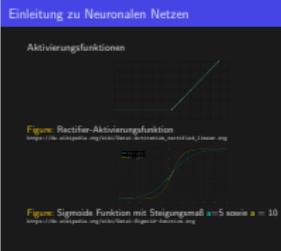


Figure: Sigmoid Funktion mit Steigungsmaß $a=5$ sowie $a = 10$

<https://de.wikipedia.org/wiki/Datei:Sigmoid-function.svg>

└ Einleitung zu Neuronalen Netzen
 └ Aktivierungsfunktion
 └ Einleitung zu Neuronalen Netzen

2022-02-07



alles negativ ist wird bei relu zu 0 während bei sigmoid, abhängig von der Steigung Werte zwischen -1 und 1 möglich sind

Loss/Kostenfunktion

Mean Squared Error

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

Mean absolute error

$$MAE = \frac{\sum_{i=1}^n |\hat{y}_i - y_i|}{n}$$

Binary Cross-Entropy

$$H(y, \hat{y}) = -\frac{1}{n} \sum_{i=1}^n y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

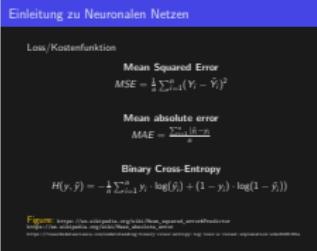
Figure: https://en.wikipedia.org/wiki/Mean_squared_error#Predictor

https://en.wikipedia.org/wiki/Mean_absolute_error

<https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a>

2022-02-07

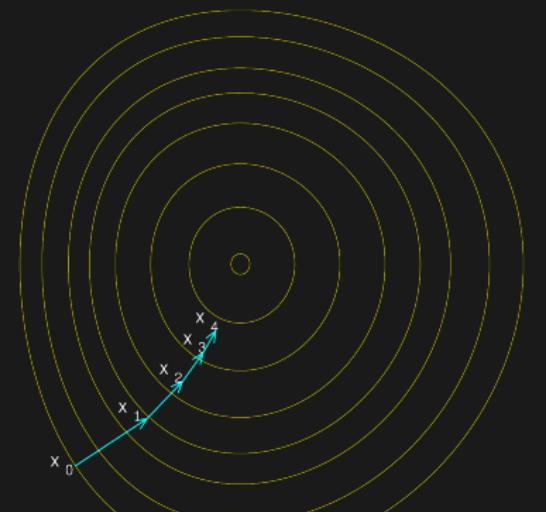
- └ Einleitung zu Neuronalen Netzen
 - └ Loss/Kostenfunktion
 - └ Einleitung zu Neuronalen Netzen



Man berechnet immer den Unterschied zwischen den wahren Labels und den predicteden Labels um zu erkennen wie weit diese auseinander liegen. Es wird immer versucht den Loss zu minimieren. Also ein Minimum der Kostenfunktion zu finden. Die Parameter der Funktion, welche angepasst werden müssen sind alle weights und biases der einzelnen Neuronen und den Layern.

Einleitung zu Neuronalen Netzen

Backpropagation mit Gradient descent



The figure shows a series of concentric circles on a black background, representing level sets of a function. A small white circle at the center represents the starting point of the optimization. Four blue arrows labeled x_0 , x_1 , x_2 , and x_3 indicate the path of the gradient descent algorithm as it moves from the outer level sets towards the minimum at the center. The arrows point inwards, showing the iterative steps of the algorithm.

Figure: Illustration of gradient descent on a series of level sets
https://en.wikipedia.org/wiki/File:Gradient_descent.svg

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis

- └ Einleitung zu Neuronalen Netzen
- └ Backpropagation mit Gradient descent
- └ Einleitung zu Neuronalen Netzen

Einleitung zu Neuronalen Netzen

Backpropagation mit Gradient descent

Figure: Illustration of gradient descent on a series of level sets

Autoencoder

2022-02-07

Ein Autoencoder ist ein feedforward Neural network, was versucht den input zu Kopieren. Das hört sich im ersten Moment nutzlos an, hat aber doch ein paar Anwendungsmöglichkeiten. Dazu gehört z.B Denosing oder Reduktion des Features Spaces



Autoencoders

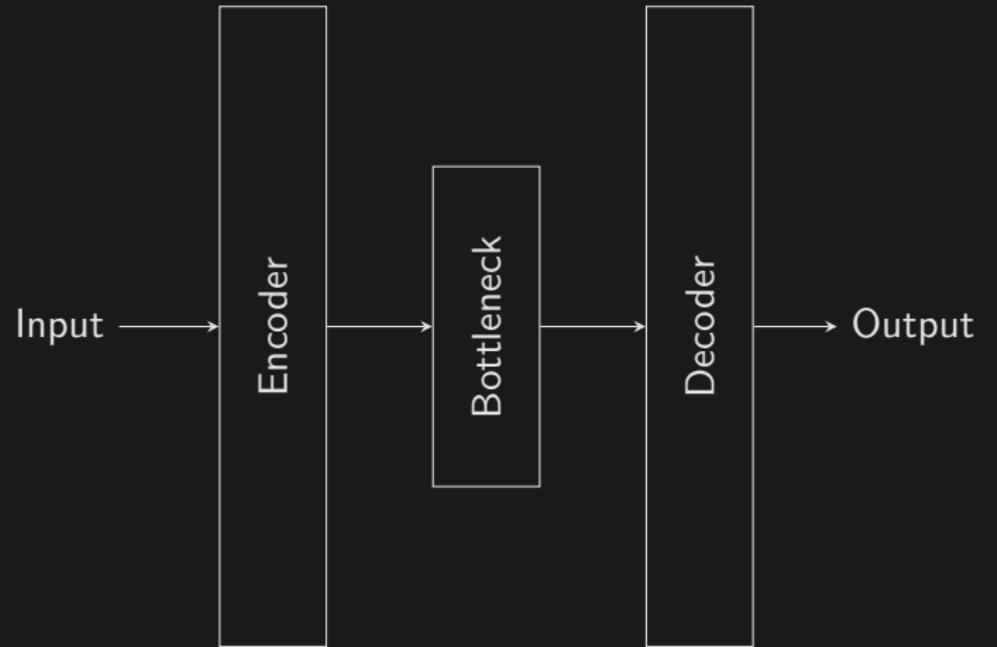


Figure: Einfaches Autoencoder Model

Unsupervised Deep Embedding for Clustering Analysis

2022-02-07

Autoencoders
└ Autoencoders
 └ Aufbau
 └ Autoencoders

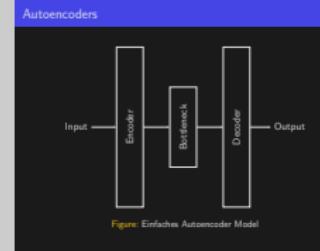


Figure: Einfaches Autoencoder Model

- Aufbau, Das Netzwerk besteht im Grunde aus 2 Teilen.
- Encoder und Decoder
- Encoder Transformiert die Input Daten in eine kleinere gewünschte Dimension
- Decoder Transformiert die Daten aus der Kleinen Dimension zurück in die Orginal Dimensionen.
- Hoffnung das der Encoder die Daten auf die Wichtigsten Features reduziert.

Autoencoders

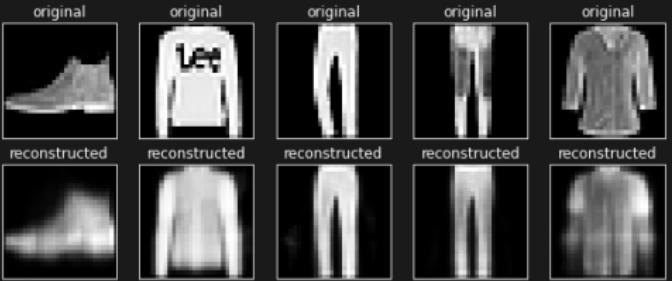


Figure: Orginal und Decoded Bilder

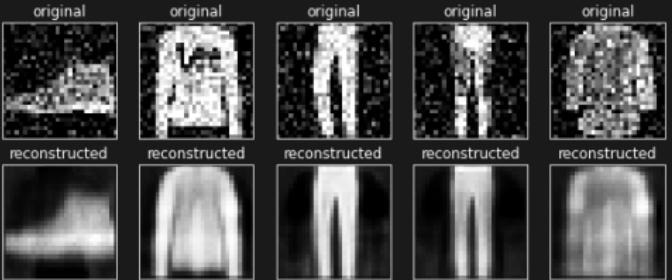


Figure: Noisy und Decoded Bilder

<https://github.com/Plutokekz/dec/blob/main/Autoencoders.ipynb>

Unsupervised Deep Embedding for Clustering Analysis

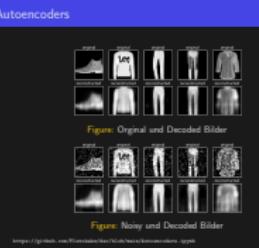
Autoencoders

Aufbau

Autoencoders

2022-02-07

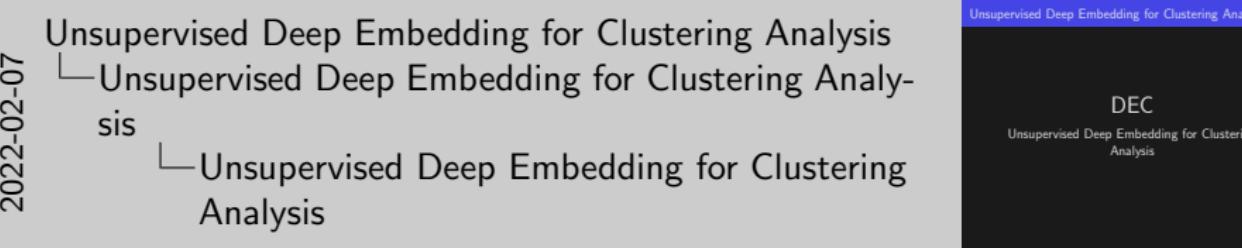
- 784 input zu 10 zu 784 (28*28)
- Bottleneck size = 10 oder feature reduktion auf 10
- Random noise



<https://github.com/Plutokekz/dec/blob/main/Autoencoders.ipynb>

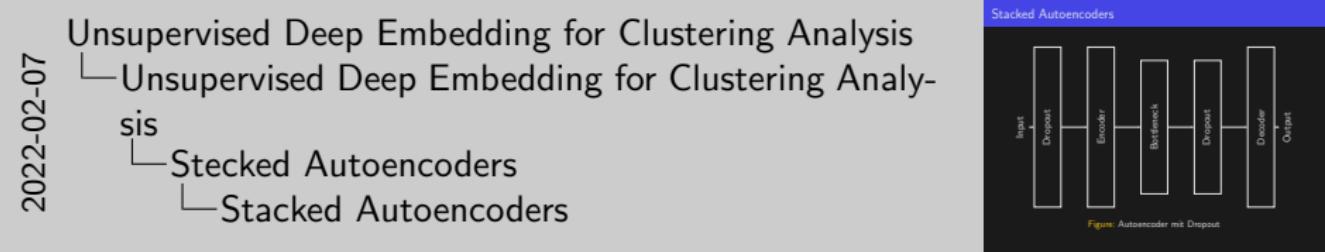
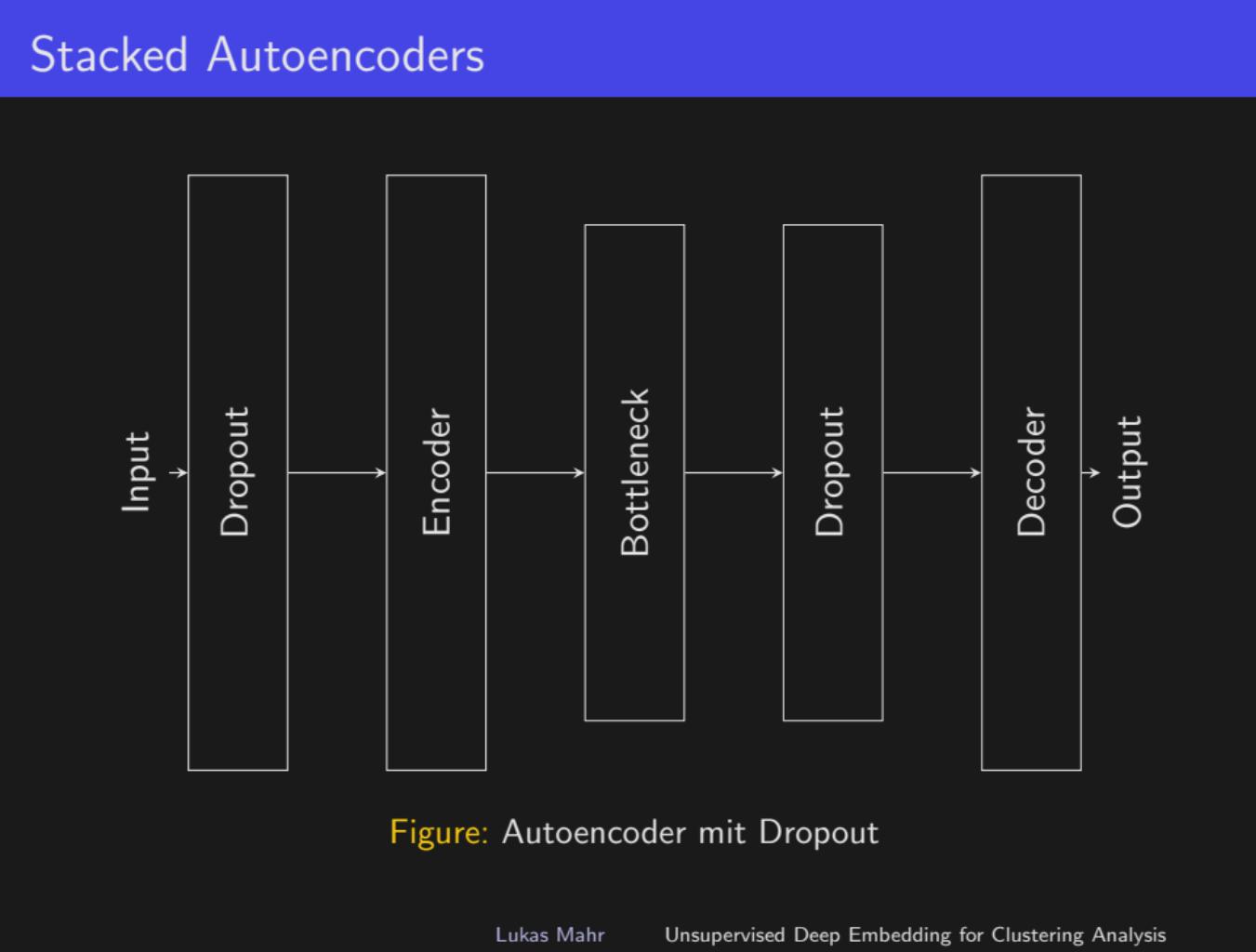
DEC

Unsupervised Deep Embedding for Clustering Analysis



DEC has two phases: (1) parameter initialization with a deep autoencode parameter optimization (i.e., clustering), where we iterate between computing an auxiliary target distribution and minimizing the Kullback–Leibler (KL) divergence to it.

- Besteht aus 2 teilen
- embedding mapping in den kleineren raum Z
- Stacked Autoencoders
- clustering
- Clustern der embddedten Daten — Embeddings verbessern so wie die Cluster anpassen



Stacked Autoencoders

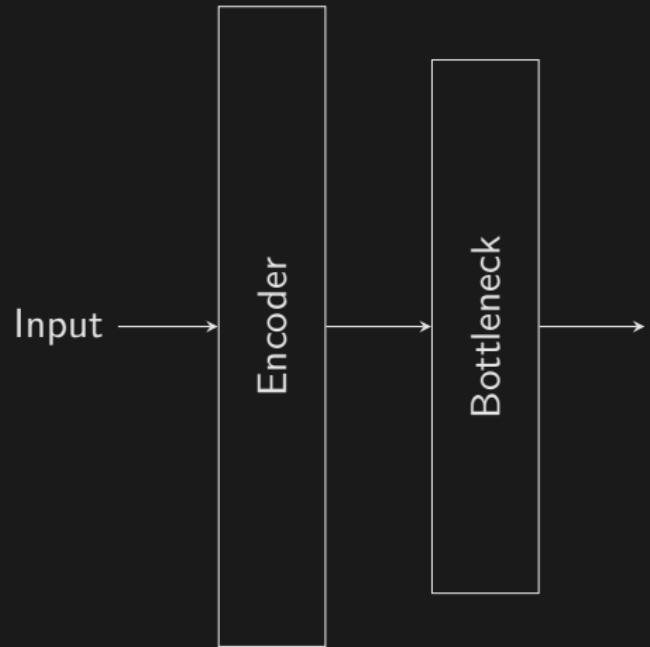
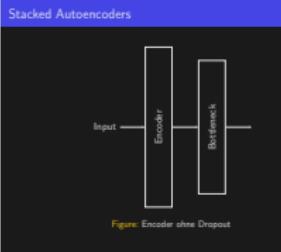


Figure: Encoder ohne Dropout

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis
└ Unsupervised Deep Embedding for Clustering Analysis
 └ Stecked Autoencoders
 └ Stacked Autoencoders

Encoder teil ohne noise/dropout und ohne dem decoder



Stacked Autoencoders

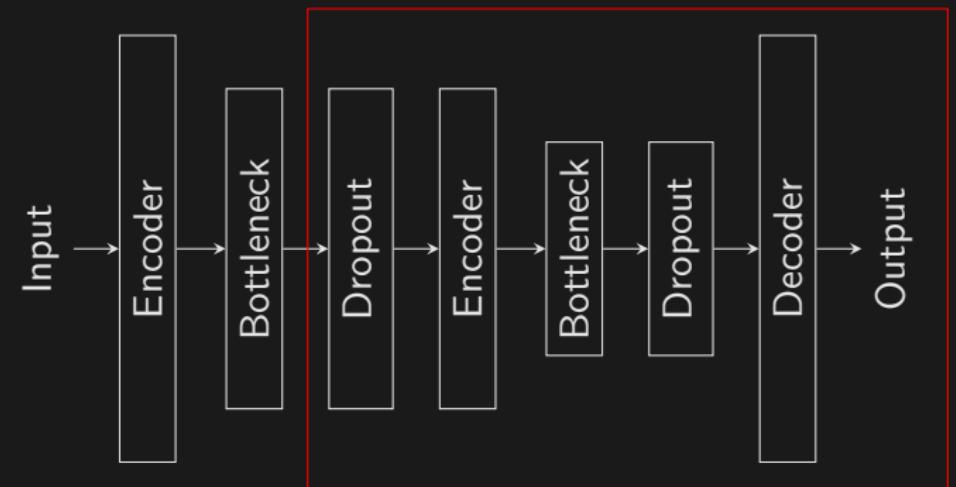
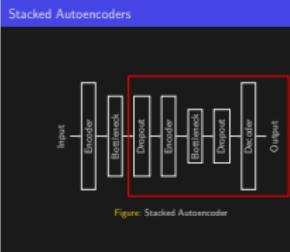


Figure: Stacked Autoencoder

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis
└ Unsupervised Deep Embedding for Clustering Analysis
 └ Stecked Autoencoders
 └ Stacked Autoencoders



Aktivierungsfunktion relu außer im letzten Decoder und Encoder Layer
sigmoid

für die zero-mean images

Zero-mean images - \bar{i} durchschnitt pro pixel über alle bilder im Datensatz
das bild wird dann von jedem bild abgezogen und damit liegt der durschnitt
bei den Bilder bei null deswegen zero mean images.

loss = least-square

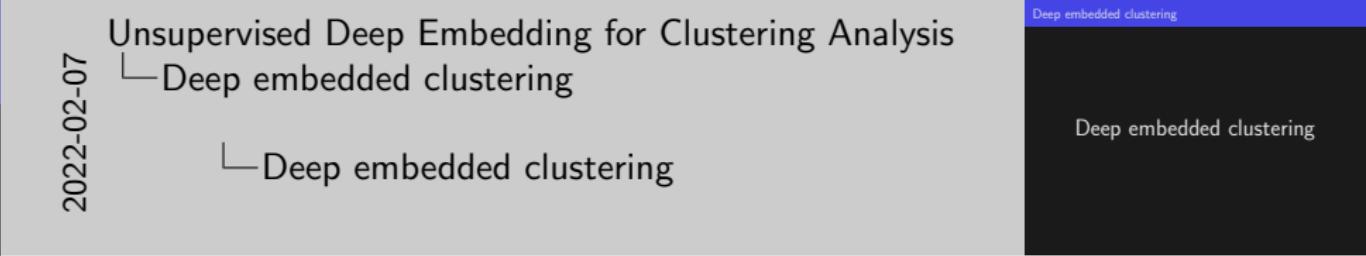
nach dem Training des Autoencoders wird der Encoder als nicht lineares
mapping für die daten genutzt die man Clustern möchte

Netzwerk Aufbau input-500-500-2000-10

50000 epochs pro layer

100000 ganzes Netz ohne dropout epochs fine tuning

256 batchsize



Deep embedded clustering

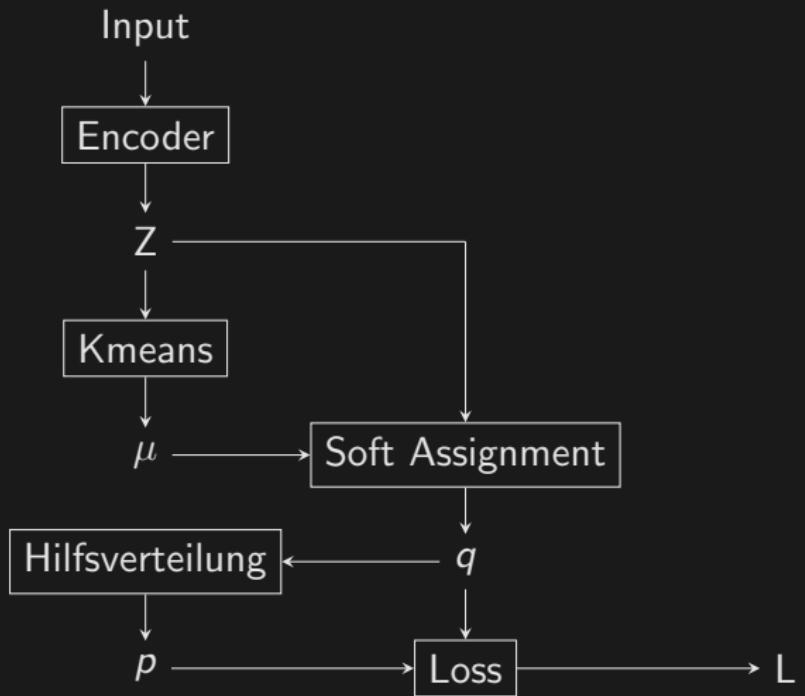


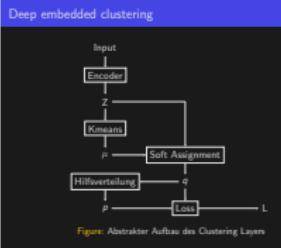
Figure: Abstrakter Aufbau des Clustering Layers

Unsupervised Deep Embedding for Clustering Analysis

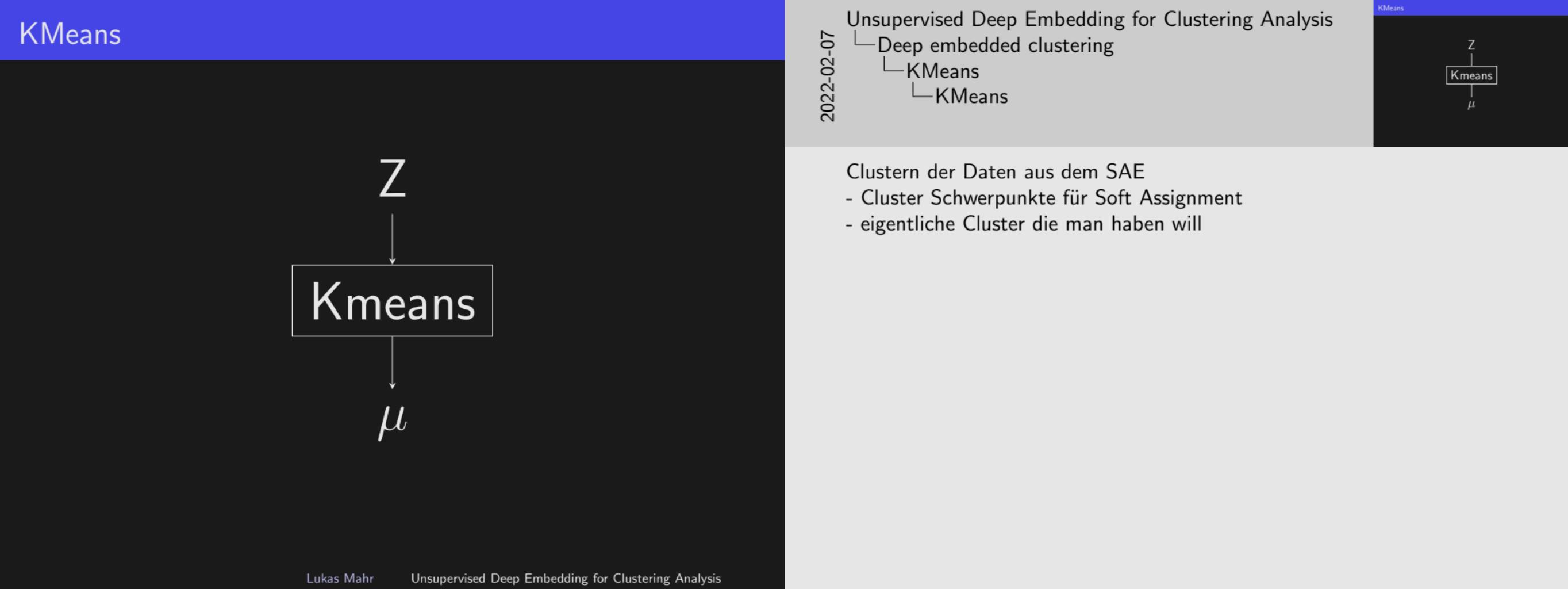
Deep embedded clustering

Deep embedded clustering

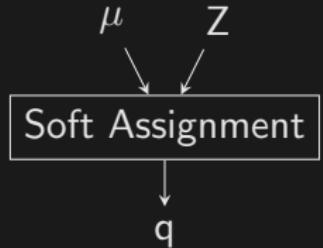
2022-02-07



- input daten durch sae werden auf kleinen feature raum Z abgebildet
- clustern der Daten Z es entstehen die Cluster Schwerpunkte μ
- soft assignments zwischen μ und z, wird zu q
- Hilfsverteilung aus q , wird zu p
- Loss zwischen q und p



Soft Assignments

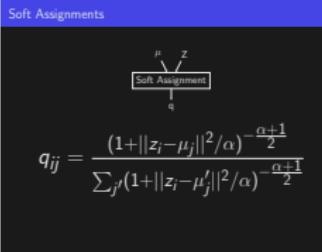


$$q_{ij} = \frac{(1+||z_i - \mu_j||^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1+||z_i - \mu'_{j'}||^2/\alpha)^{-\frac{\alpha+1}{2}}}$$

Unsupervised Deep Embedding for Clustering Analysis

2022-02-07

- └ Deep embedded clustering
- └ Soft Assignments
- └ Soft Assignments



Student t-Verteilung um die Ähnlichkeit zwischen z_i und μ_j zu messen
Freiheitsgrad ist 1[3]

q_{ij} ist die Wahrscheinlichkeit die z_i dem Cluster q_j zuzuordnen

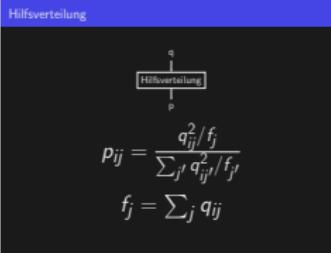
$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}}$$

$$f_j = \sum_j q_{ij}$$



2022-02-07

└ Deep embedded clustering
 └ Hilfsverteilung
 └ Hilfsverteilung



Loss

Kullback-Leibler-Divergenz


$$L = KL(P||Q) \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis

- Deep embedded clustering
- Loss
- Loss

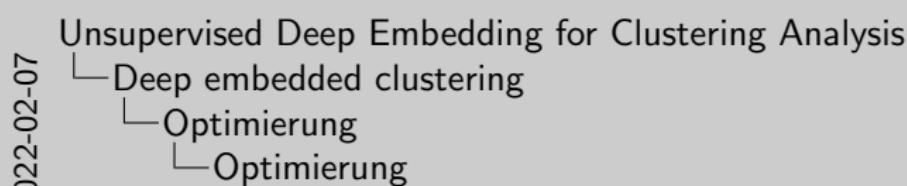
Kullback-Leibler-Divergenz


$$L = KL(P||Q) \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

$$\frac{\partial L}{\partial z_i} = \frac{\alpha+1}{\alpha} \sum_j \left(1 + \frac{\|z_i - \mu_j\|^2}{\alpha}\right)^{-1} \times (p_{ij} - q_{ij})(z_i - \mu_j)$$

$$\frac{\partial L}{\partial \mu_j} = -\frac{\alpha+1}{\alpha} \sum_i \left(1 + \frac{\|z_i - \mu_j\|^2}{\alpha}\right)^{-1} \times (p_{ij} - q_{ij})(z_i - \mu_j)$$

Stochastic Gradient Descent



Optimierung

Stochastic Gradient Descent

$$\frac{\partial L}{\partial z_i} = \frac{\alpha+1}{\alpha} \sum_j \left(1 + \frac{\|z_i - \mu_j\|^2}{\alpha}\right)^{-1} \times (p_{ij} - q_{ij})(z_i - \mu_j)$$

$$\frac{\partial L}{\partial \mu_j} = -\frac{\alpha+1}{\alpha} \sum_i \left(1 + \frac{\|z_i - \mu_j\|^2}{\alpha}\right)^{-1} \times (p_{ij} - q_{ij})(z_i - \mu_j)$$

Gradient von L wird berechnet in Abhängigkeit von q_i und μ_j
 $\frac{\partial L}{\partial z_i}$ wird dann für backpropagation genutzt.
wird Optimiert bis eine bestimmter Prozentsatz an Cluster punkten nicht mehr das Cluster wechselt, zwischen zwei fortlaufenden Iterationen.

MNIST



mnist

- 70k Handgeschriebene Zahlen
- von 0 - 9
- 28*28 pixel

STL-10

STL-10

- 10 Klassen
- 1300 Bilder pro Klasse

zusätzlich:

- 100000 Unbeschriftete Bilder
- haben beide benutzt
- aber gepreprocessed
- HOG features mit einem 8*8 filter

HOG:

- Histogram of Oriented Gradients
- Ausrichtung der Kanten und des Gradienten in 8*8 pixel blöcken

REUTERS

2022-02-07
Reuters:

- 810000 English Nachrichtenberichte
- in Kategorie Bäume eingeteilt
- 4 Haupt Kategorien (corporate/industrial, government/social, markets, economics)
- alle die mehr Haupt Kategorien hatten sind raus geflogen
- 685071 übrig
- tf-idf von den 2000 häufigsten Wort Stämmen
- Term Frequency-Inverse Document Frequency
- wie wichtig ein Wort in einem Satz von einem Text ist.

$$ACC = \max_m \frac{\sum_{i=1}^n 1\{l_i=m(c_i)\}}{n}$$

2022-02-07

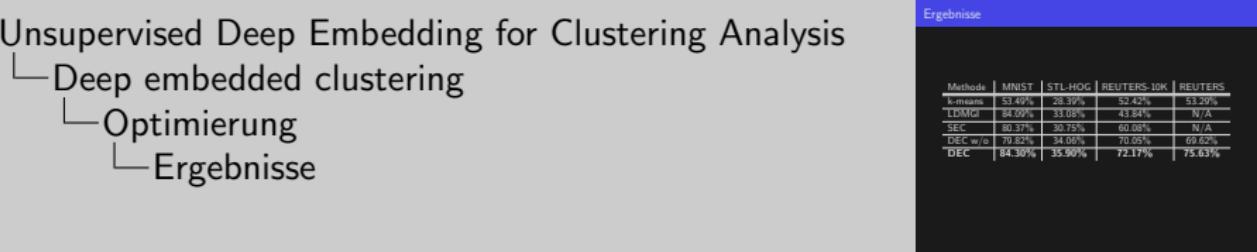
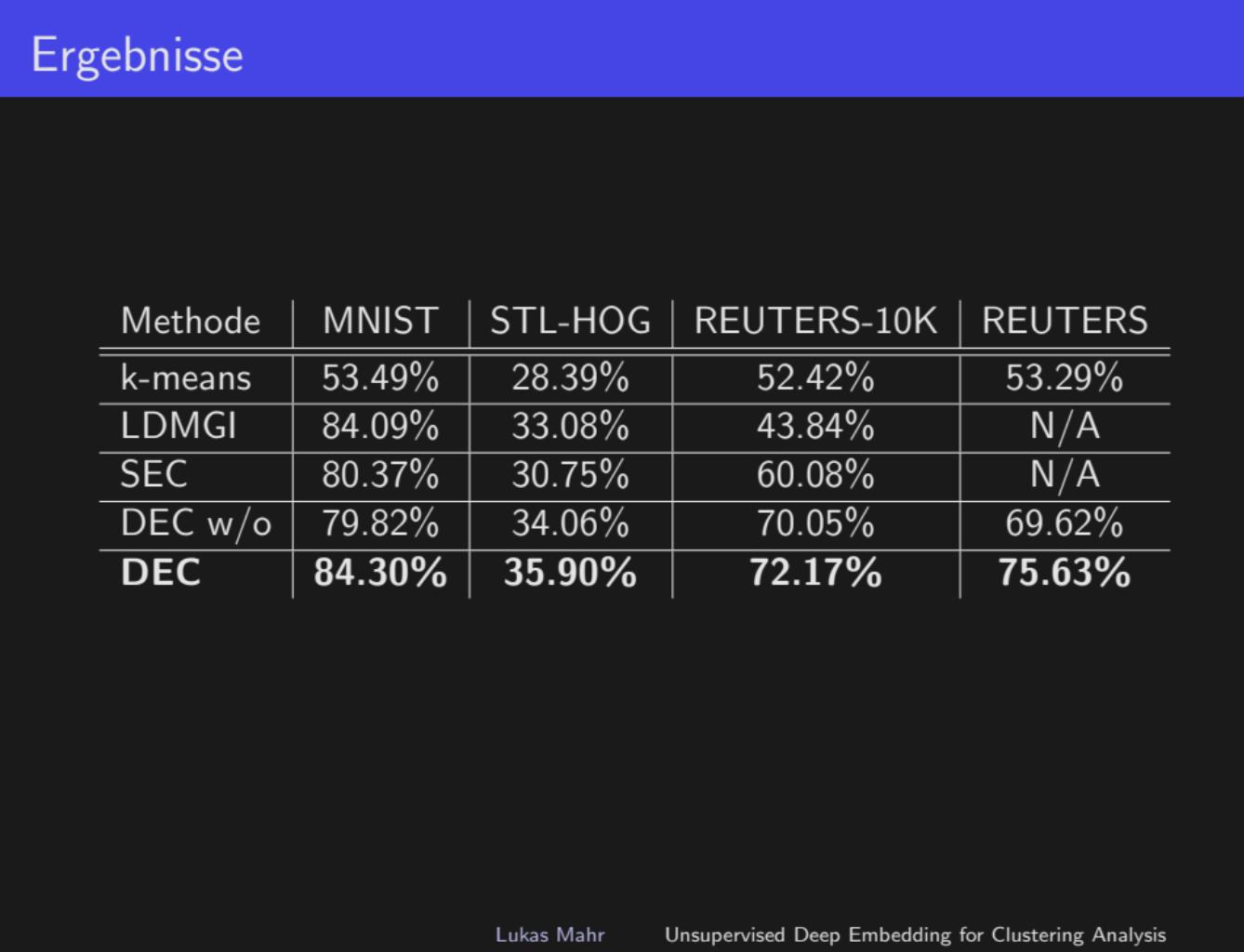
Unsupervised Deep Embedding for Clustering Analysis

- Deep embedded clustering
- Optimierung
- Bewertungsmetrik

Bewertungsmetrik

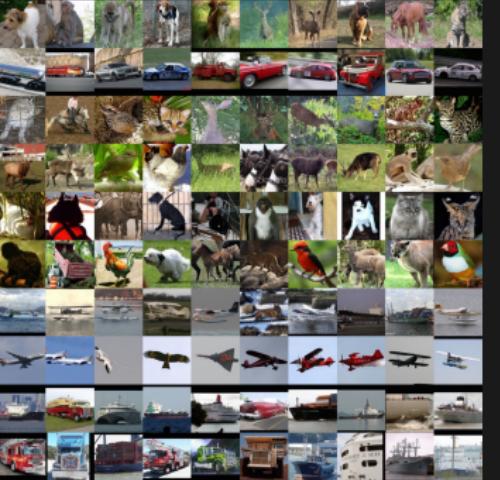
$$ACC = \max_m \frac{\sum_{i=1}^n 1\{l_i=m(c_i)\}}{n}$$

- l_i das wahre label
 - c_i Cluster Zuordnung
 - m alle one to one Zuordnungen zwischen Clustern und Labels
- Findet das beste mapping zwischen Cluster und Label



erklären der anderen Algorithmen und deren Start Bedingungen
 DEC can process the entire REUTERS dataset in half an hour with GPU acceleration while the second best algorithms, LDGMI and SEC, would need months of computation time and terabytes of memory. We, indeed, could not run these methods on the full REUTERS dataset and report N/A in Table 2

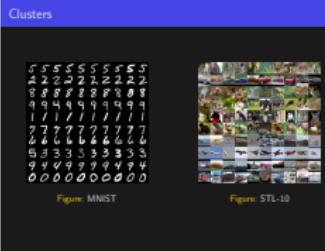
Clusters



Unsupervised Deep Embedding for Clustering Analysis

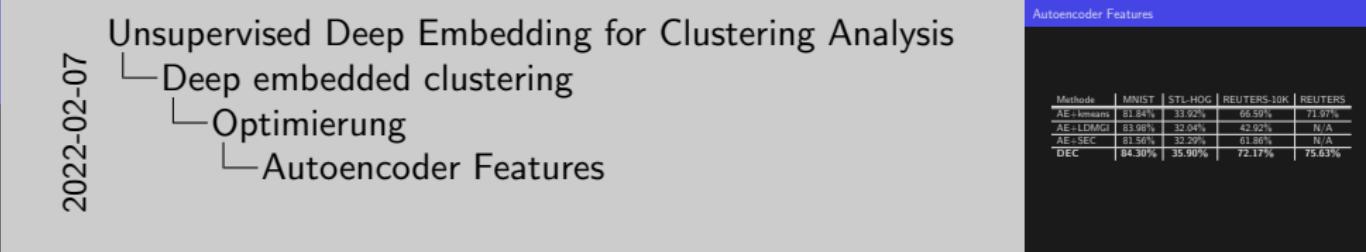
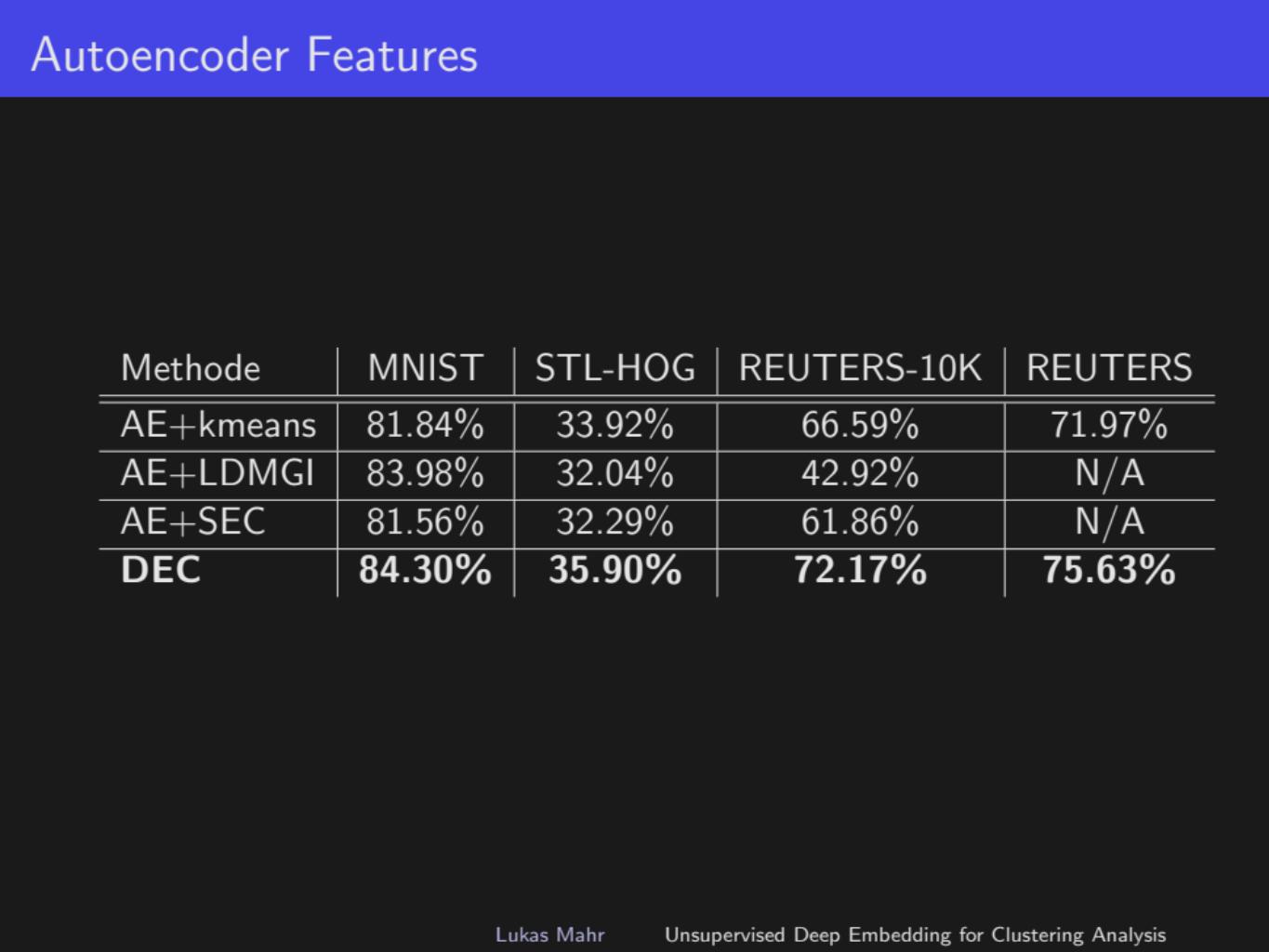
- Deep embedded clustering
- Optimierung
- Clusters

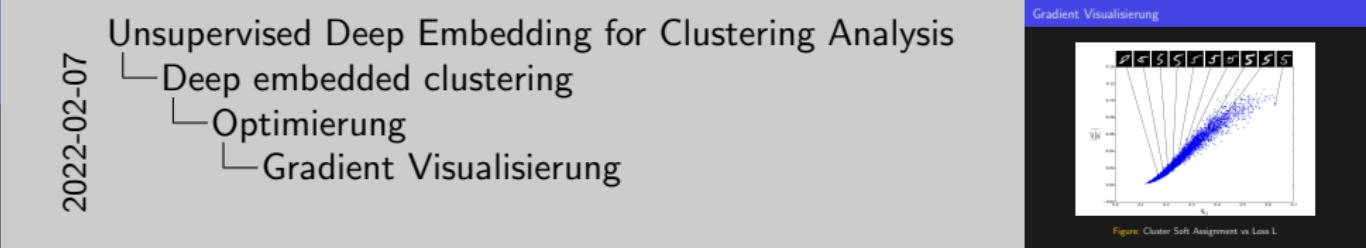
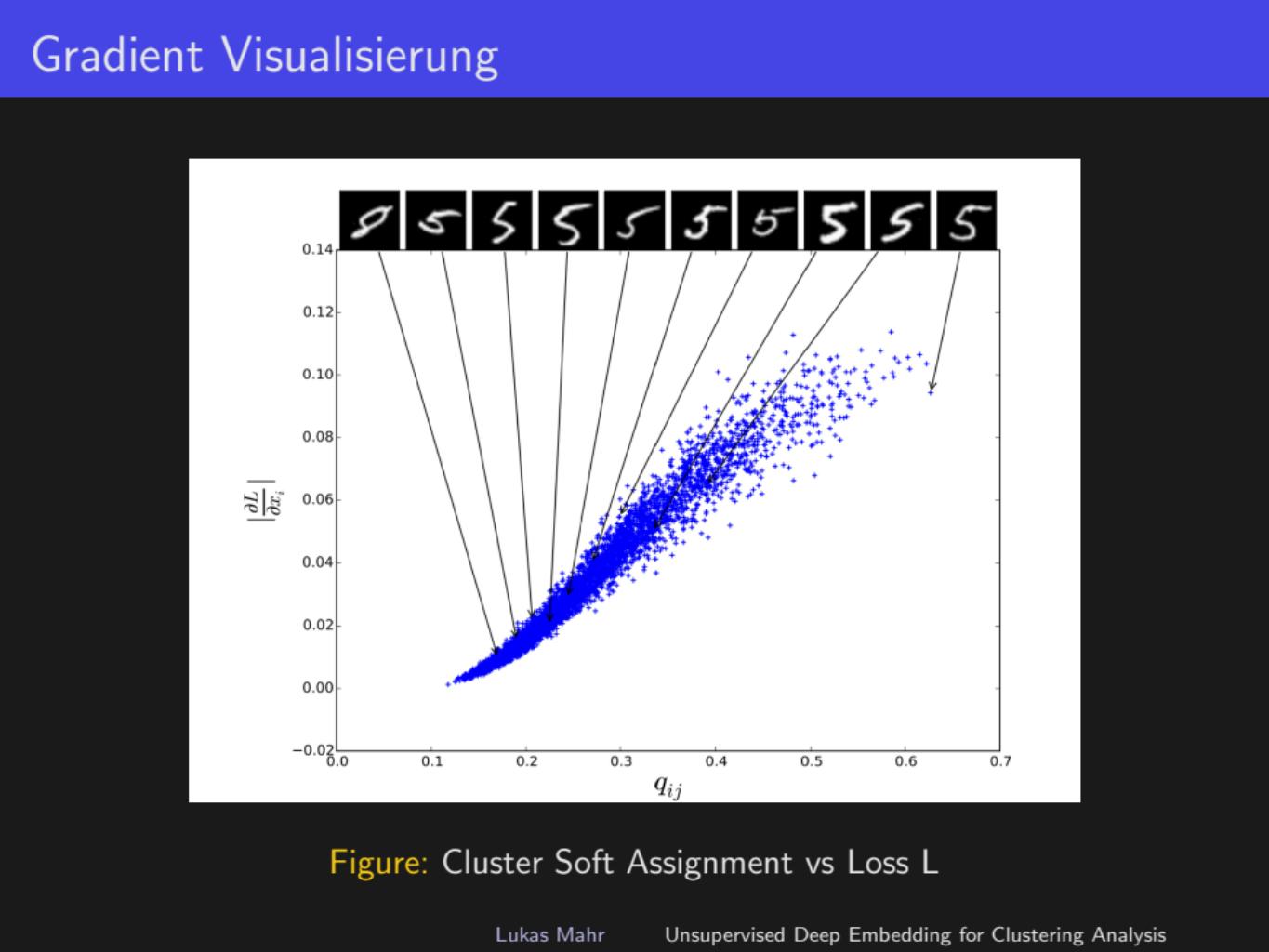
2022-02-07



Each row contains the top 10 scoring elements from one cluster.

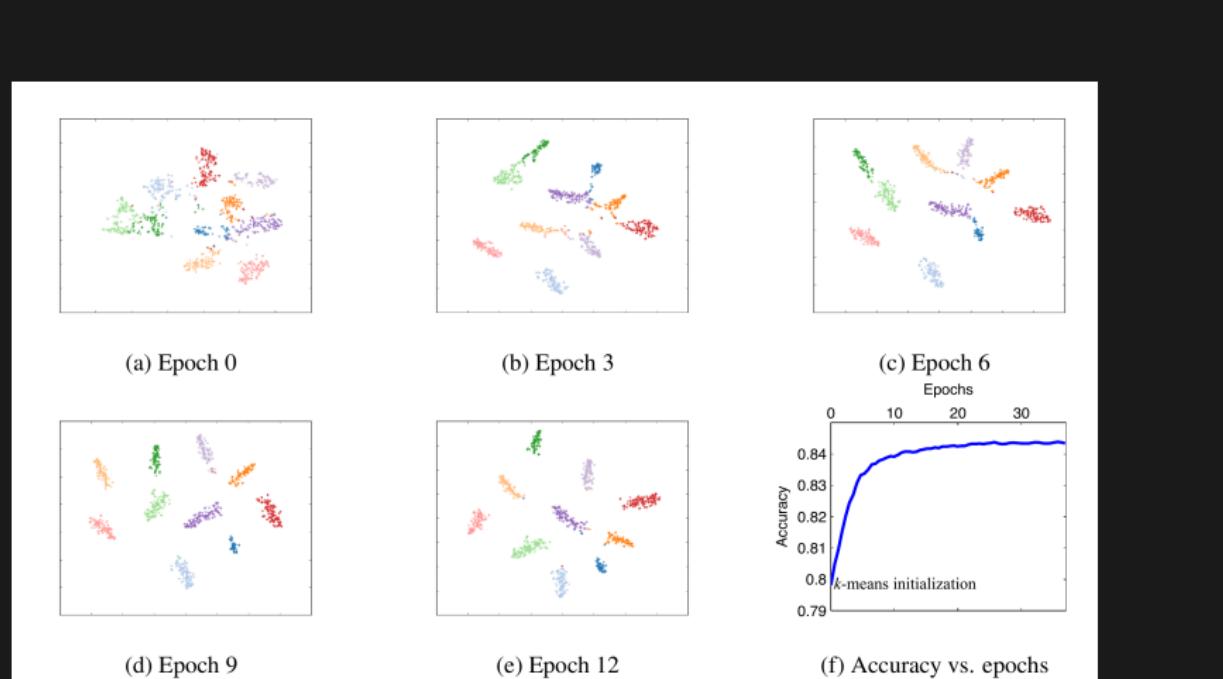
In Fig. 3 we show 10 top scoring images from each cluster in MNIST and STL. Each row corresponds to a cluster and images are sorted from left to right based on their distance to the cluster center. We observe that for MNIST, DEC's cluster assignment corresponds to natural clusters very well, with the exception of confusing 4 and 9, while for STL, DEC is mostly correct with airplanes, trucks and cars, but spends part of its attention on poses instead of categories when it comes to animal classes.





Richtigkeit von Hilfsverteilung P und Punkte die nähere am Cluster Mittelpunkt sind haben mehr Einfluss auf den Gradienten.

Cluster Überzeit



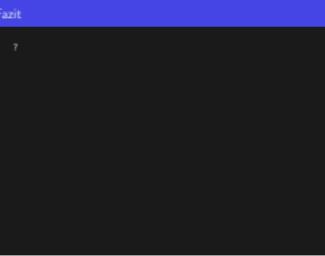
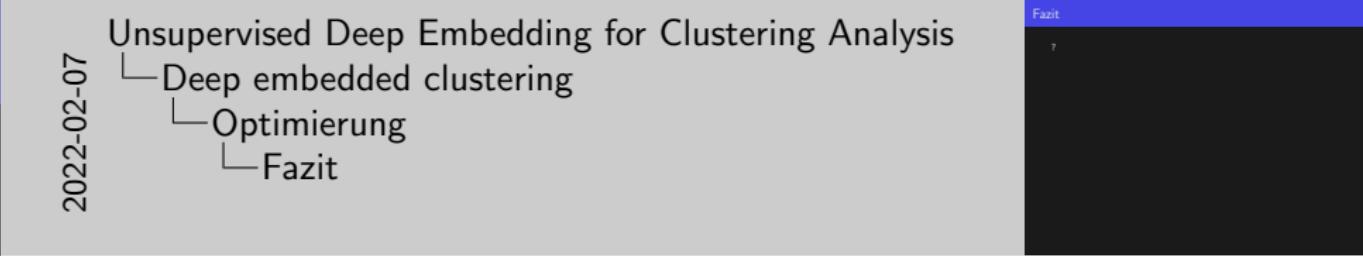
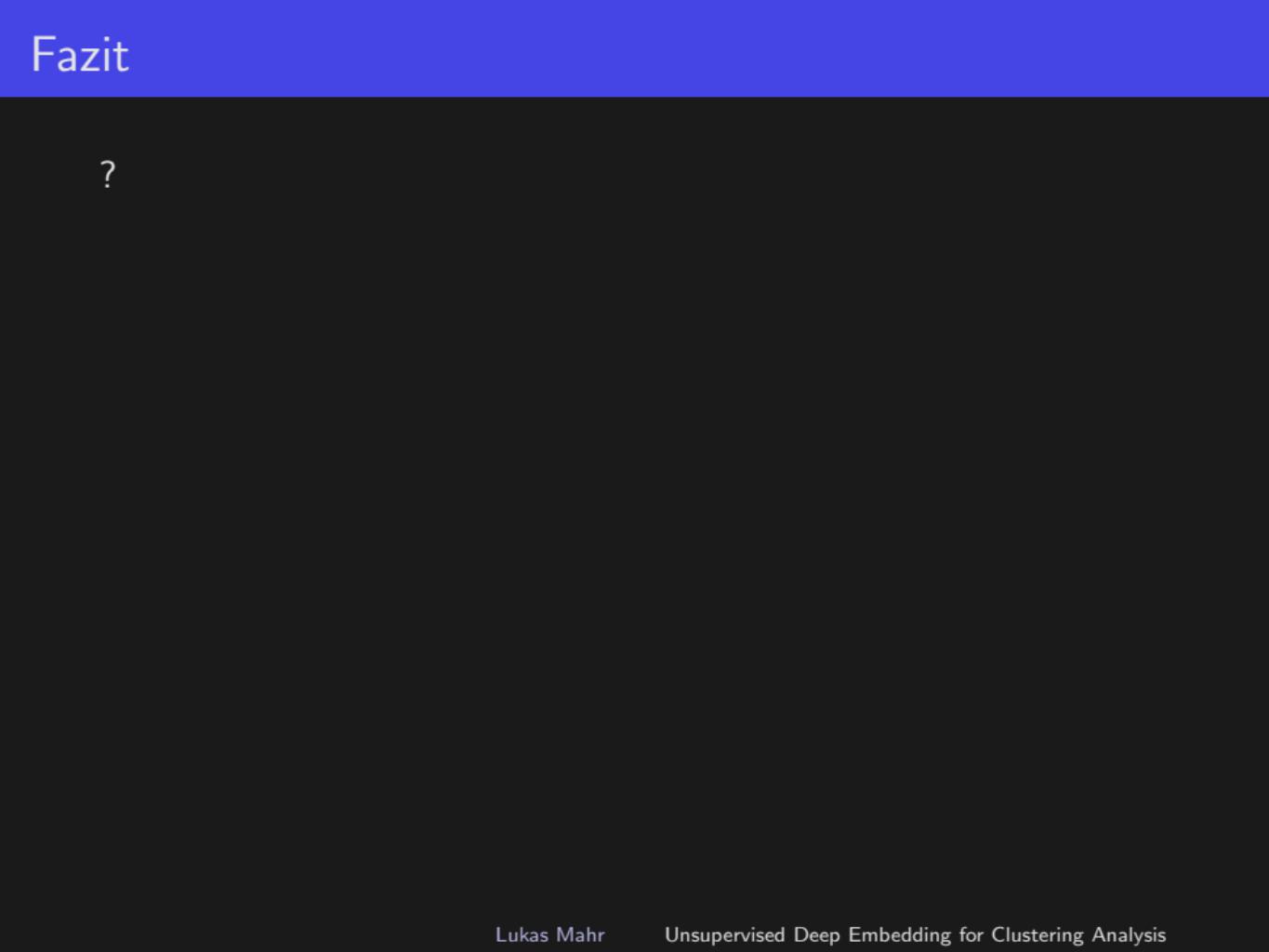
2022-02-07

Unsupervised Deep Embedding for Clustering Analysis

- Deep embedded clustering
- Optimierung
- Cluster Überzeit

verbesserung der Cluster über Zeit ???







Von wem ist das Paper

Referenzen

Steinbach, Michael, Ertöz, Levent, and Kumar, Vipin. The challenges of clustering high dimensional data. In *New Directions in Statistical Physics*, pp. 273–309. Springer, 2004.

Ye, Jieping, Zhao, Zheng, and Wu, Mingrui. Discriminative k-means for clustering. In *NIPS*, 2008.

van der Maaten, Laurens. Learning a parametric embedding by preserving local structure. In *International Conference on Artificial Intelligence and Statistics*, 2009.

2022-02-07

Unsupervised Deep Embedding for Clustering Analysis

Referenzen

Steinbach, Michael, Ertöz, Levent, and Kumar, Vipin. The challenges of clustering high dimensional data. In *New Directions in Statistical Physics*, pp. 273–309. Springer, 2004.

Ye, Jieping, Zhao, Zheng, and Wu, Mingrui. Discriminative k-means for clustering. In *NIPS*, 2008.

van der Maaten, Laurens. Learning a parametric embedding by preserving local structure. In *International Conference on Artificial Intelligence and Statistics*, 2009.