Machbarkeitsstudie des L-BFGS Verfahrens für das Training von Deep Learning Problemen

WAW-ML3 19.11.-20.11.2018

Ilona Shonia

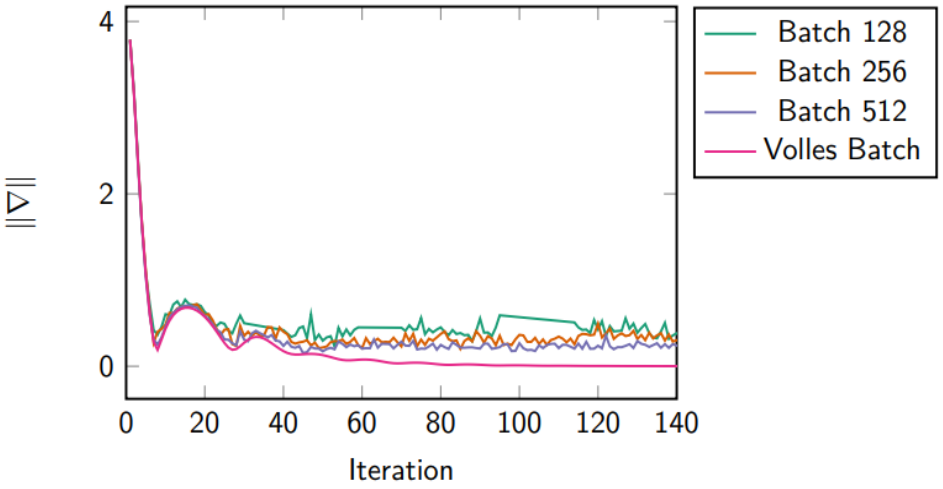
DLR

Motivation

Das Training von großen Modellen erfordert
- hohe Rechenkapazität
- großen Speicherplatz

an die (gradientenbasierten) Optimierungsalgorithmen, die zum Lernen verwendet werden.

Um diesen Aufwand zu reduzieren, werden Gradienten von einem zufällig gewählten Teil des Datensatzes (Batch) gebildet. Dies gibt dem berechneten Gradienten einen stochastischen Charakter.

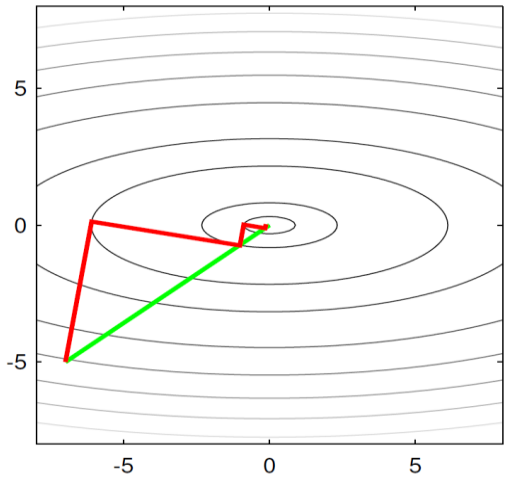


Motivation

- Die stochastische Abwandlung des gradientenbasierten Optimierungsalgorithmus Gradient Descent wird in ML erfolgreich verwendet und ist als Stochastic Gradient Descent (SGD) bekannt.

- Können jedoch anspruchsvollere Optimierungsalgorithmen unter Verwendung von rauschhaltigen Gradienten noch bessere Ergebnisse liefern?

- Problemstellung: die Anwendbarkeit der L-BFGS Methode mit stochastischen Gradienten für das Training von Deep Learning Problemen.



Anwendungsproblem

Klassifizierung von Bildern auf Basis des MNIST-Datensatzes - ein Problem des überwachten Lernens.

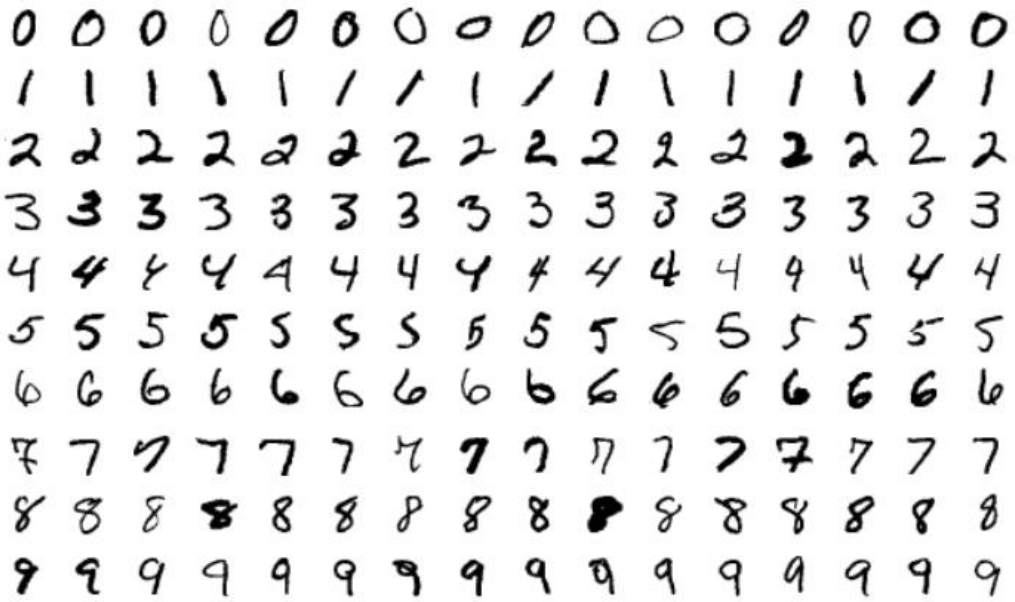
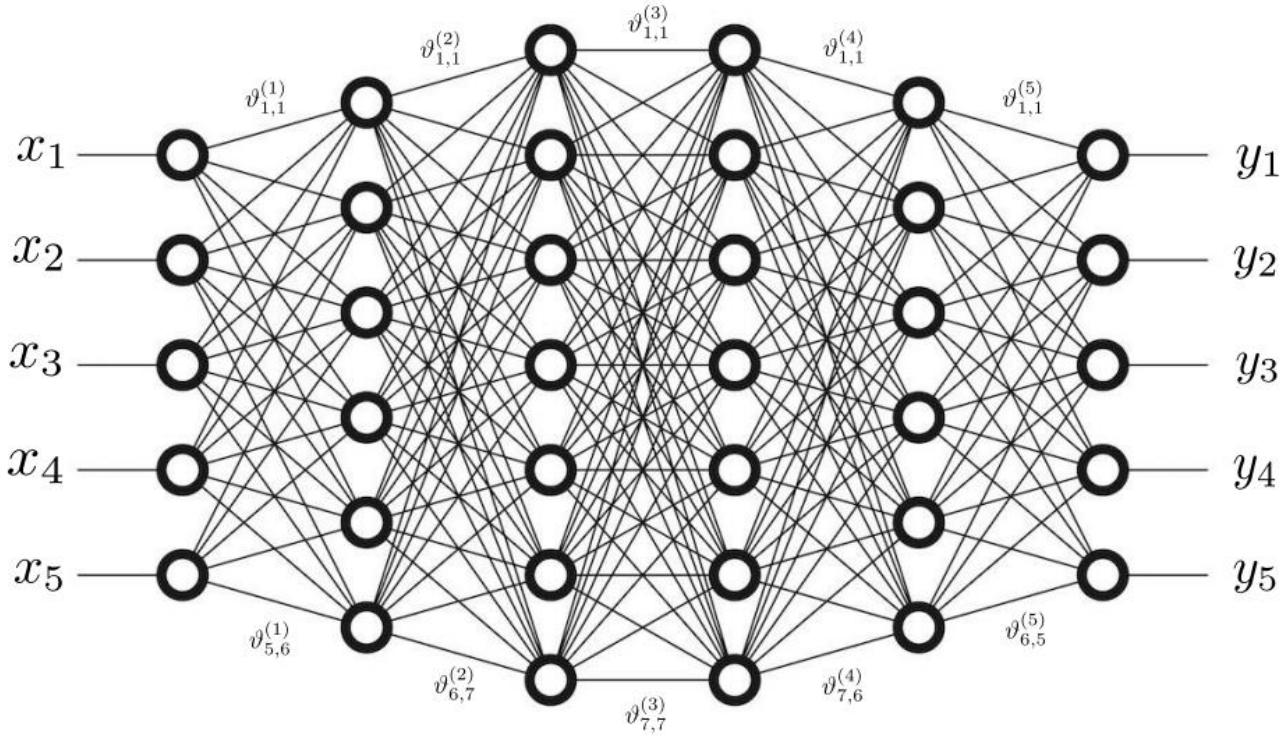
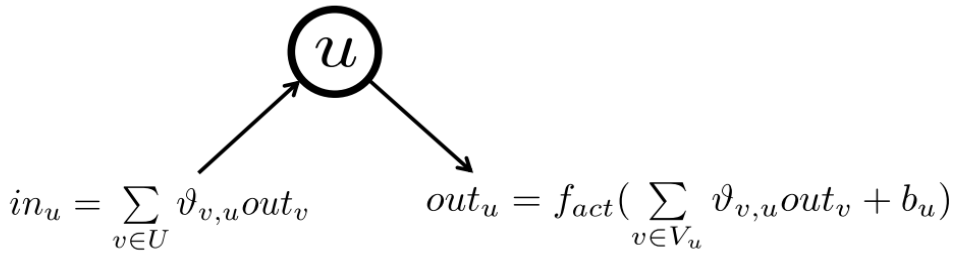


Abbildung: Der MNIST Datensatz
https://en.wikipedia.org/wiki/MNIST_database

Lernalgorithmus





Training

- Bewertung der Modelqualität:

Fehlerfunktion, die den Unterschied zwischen der Grundwahrheit und der von dem Modell gemachten Vorhersage misst.

- Training des Netzes:

$$min_{\vartheta \in \mathbb{R}^n} J(\vartheta)$$

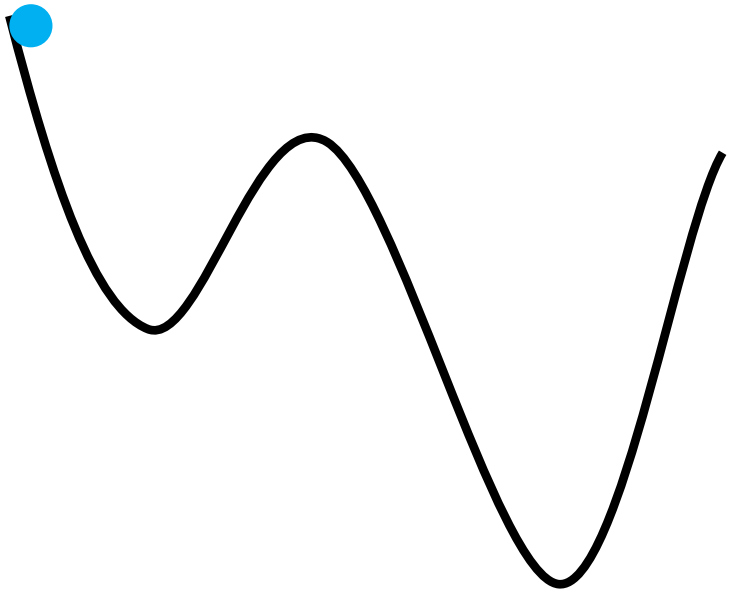
➡ ein Optimierungsproblem

Optimierungsalgorithmen

- Stochastic Gradient Descent:

Stochastischer Gradient

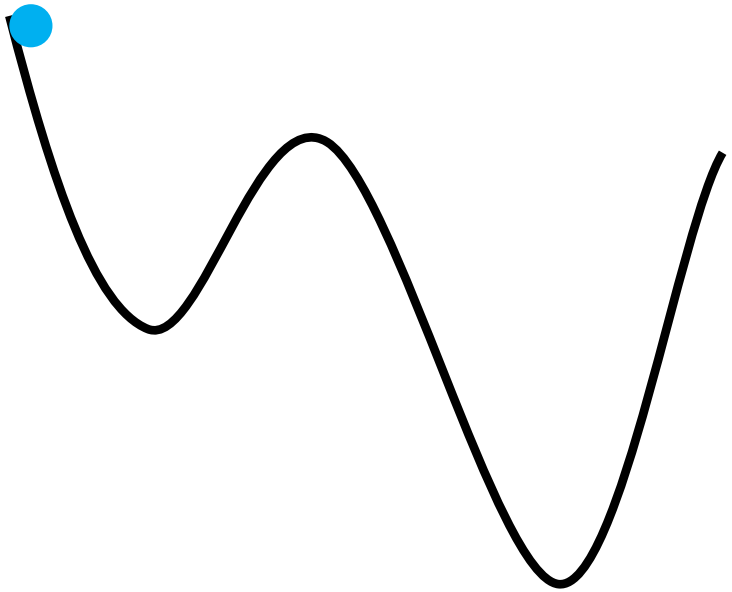
$$\vartheta_{k+1} = \vartheta_k - \alpha \hat{\nabla} J(\vartheta_k), \text{ wobei } \hat{\nabla} J(\vartheta_k) = \frac{1}{|S_k|} \sum_{i \in S_k} \nabla J_i(\vartheta_k)$$



Optimierungsalgorithmen

- Adam: stochastischer Gradient + Momentum

$$\vartheta_k \leftarrow \vartheta_{k-1} - \alpha \hat{m}_k / \left(\sqrt{\hat{v}_k} + \varepsilon\right)$$



Methoden zweiter Ordnung

- Das einfachste Gradientenverfahren besitzt nur eine lineare Konvergenzgeschwindigkeit.

- Eine quadratische Konvergenzgeschwindigkeit kann mit dem Newton-Verfahren erreicht werden.

- Iterationsvorschrift für das Newton-Verfahren:

$$\vartheta_{k+1} = \vartheta_k - H_k^{-1} \nabla J_k$$

- Nachteil des Newton-Verfahrens: Verwendung der vollen inversen Hesse Matrix ➡ für große Probleme sehr rechen- und speicheraufwändig.

Quasi-Newton Verfahren
Approximation der inversen Hesse-Matrix

- Iterationsvorschrift für allgemeine Quasi-Newton Verfahren:

$$\vartheta_{k+1} = \vartheta_k - \alpha_k B_k \nabla J_k, \text{ wobei } B_k \approx H_k^{-1}$$

- Das BFGS Verfahren. Approximation der inversen Hesse Matrix durch

$$B_{k+1} = \left(I - \rho_k s_k y_k^T\right) B_k \left(I - \rho_k y_k s_k^T\right) + \rho_k s_k s_k^T,$$

$$\text{wobei } \rho_k = 1/y_k^T s_k,\ s_k = \vartheta_{k+1} - \vartheta_k \text{ und } y_k = \nabla J_{k+1} - \nabla J_k.$$

Das L-BFGS Verfahren

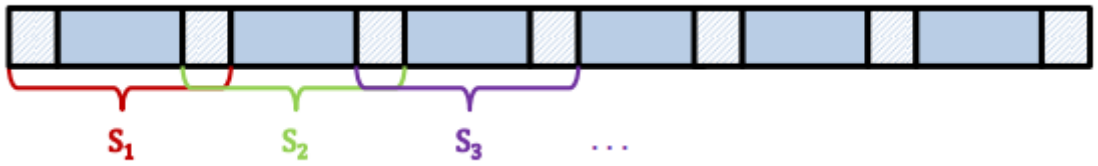
➢ Das Limited-Memory BFGS (L-BFGS) Verfahren:

Berücksichtigung der Krümmungsinformation nur aus den letzten m Iterationen zur Berechnung der Approximation der inversen Hesse Matrix
➡ gesparter Speicherplatz.

➢ Modifizierung der Methode für stochastische Fälle: das MB-LBFGS und das PB-LBFGS Verfahren.

➢ Das MB-LBFGS Verfahren:

Erzwingen von Überlappungen zwischen aufeinanderfolgenden Batches und Auswertung von stochastischen Gradienten auf den Überlappungen für Berechnung der Änderung im Gradienten.



PB-LBFGS

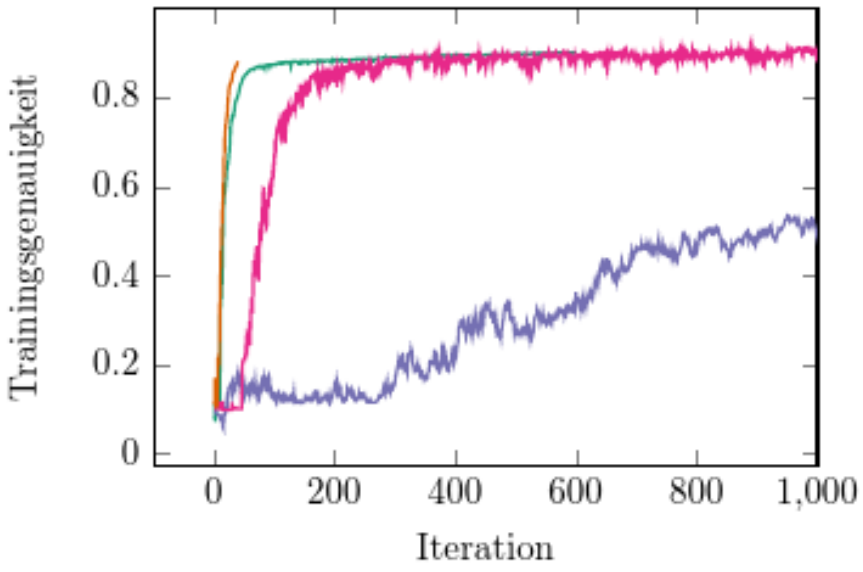
- Anpassung der Batchgröße in jeder Iteration mit dem Inneres-Produkt-Quasi-Newton-Test:

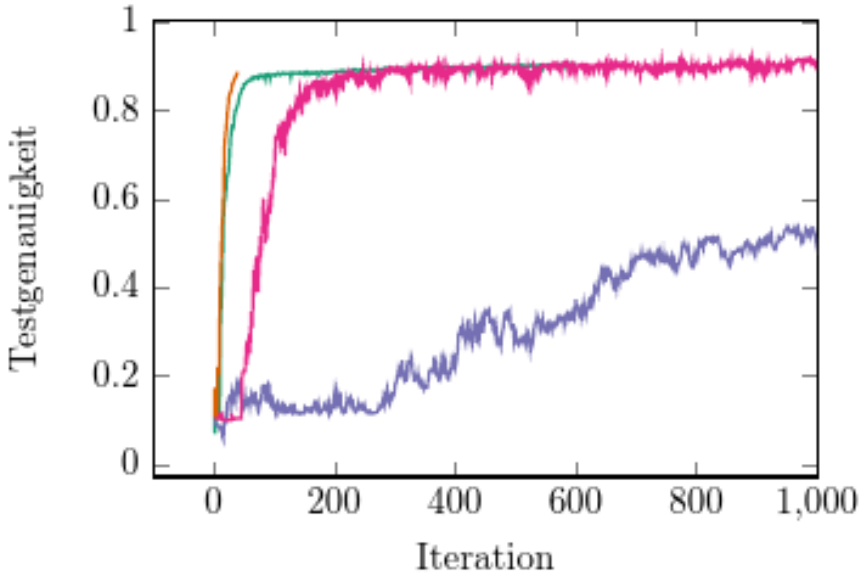
$$\frac{Var_{i \in S_k^v}\left((g_k^i)^T B_k^2 g_k^{S_k}\right)}{|S_k|} \leq \theta^2 \|B_k g_k^{S_k}\|^4$$

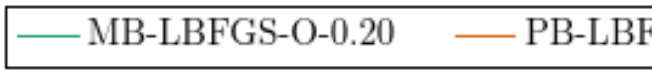
- Modifikation der Lernrate in jeder Iteration mit dem Startwert:

$$\alpha_k = \left(1 + \frac{Var_{i \in S_k^v}\{g_k^i\}}{|S_k| \|g_k^{S_k}\|^2}\right)^{-1}$$

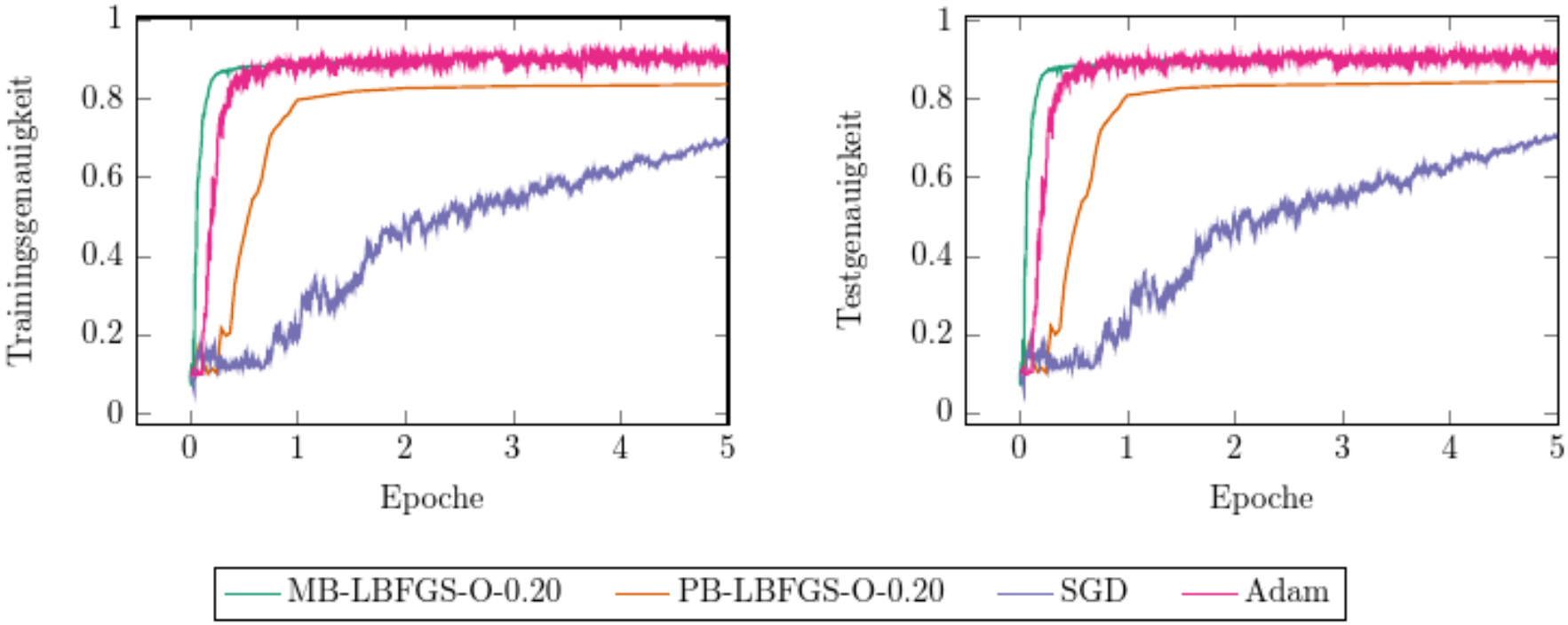
Numerische Ergebnisse



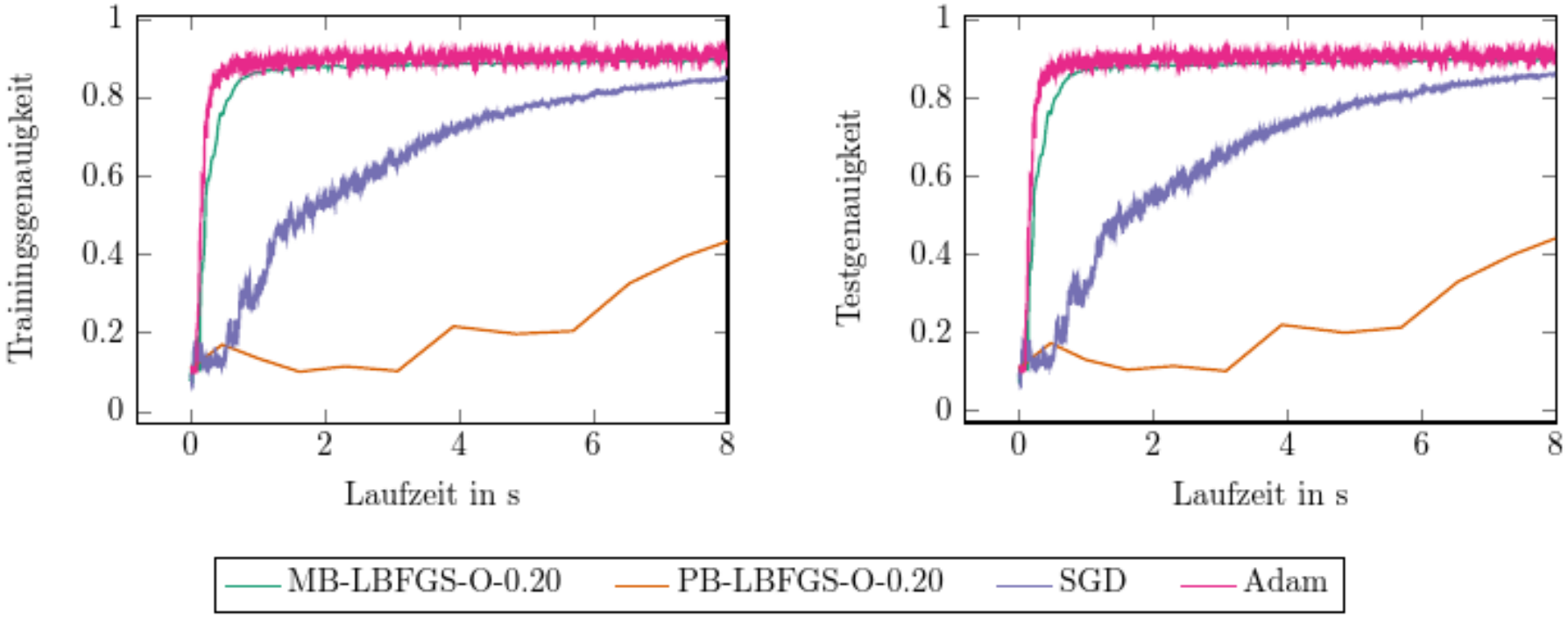




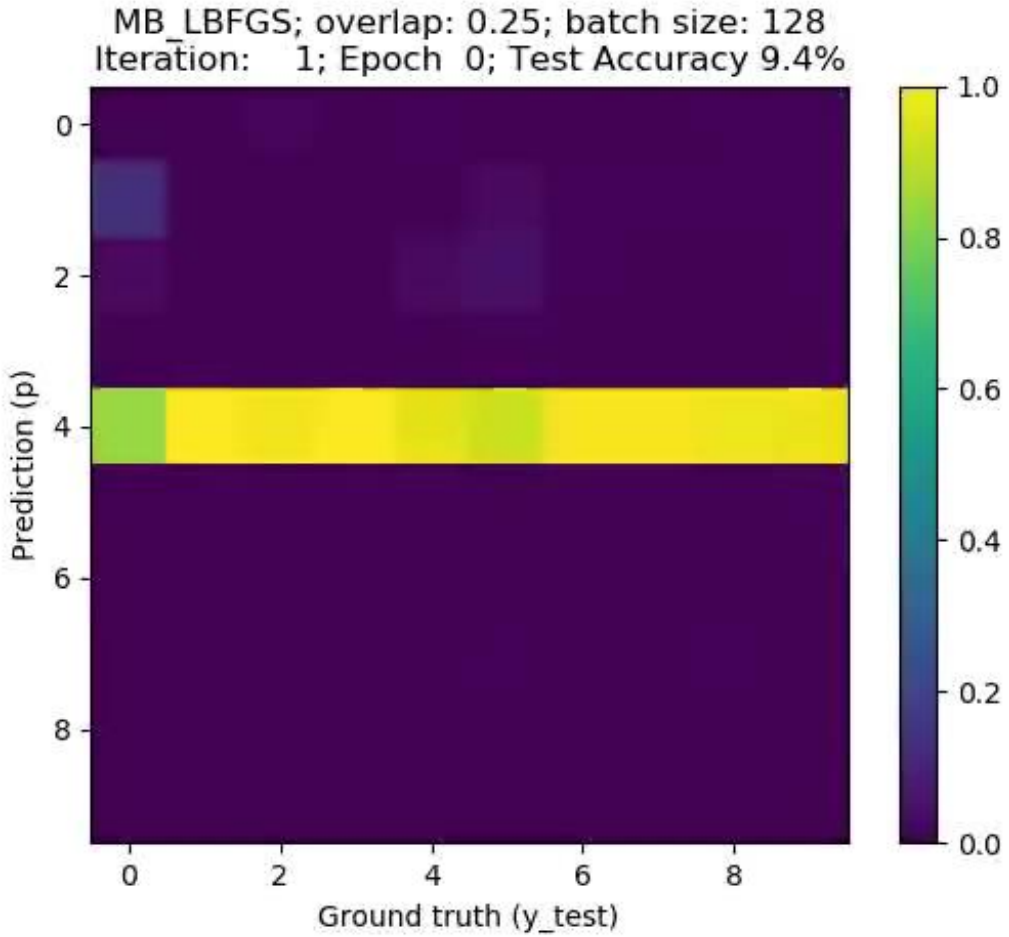
Numerische Ergebnisse



Numerische Ergebnisse



MB-LBFGS am Training



Fazit & Ausblick

- Das Einbeziehen der approximierten Informationen zweiter Ordnung hilft dabei, bessere Schritte in die Richtung des Optimums zu machen.

- Dadurch benötigen die Methoden zweiter Ordnung deutlich weniger Iterationen, als die von der ersten Ordnung (SGD, Adam,…).

- Jedoch wächst dabei der Rechenaufwand so sehr, dass es nicht möglich ist die Performanz des Adam-Algorithmus in Laufzeit zu übertreffen.

- Für die Zukunft lässt sich hoffen, dass durch Zulassen von mehr Rauschen in Varianz bessere Varianten der PB-LBFGS Methode erscheinen.

Danke für Ihre Aufmerksamkeit!