# Stochastic Optimization in Machine Learning

Fin Bauer, Stanislas Chambon, Roland Halbig,
Stefan Heidekrüger, Jakob Heuke

Technische Universität München

27. Mai 2015

# Outline

# Introduction (1)

## Challenges in Machine Learning

- massive amounts of training data
- construction of very large models
- how to handle the high memory/computational demands?

$$\Downarrow$$

**Stochastic Methods:** Update on small amounts of training data!

# Introduction (2)

## Optimization Problem

$$\min_{w \in \mathbb{R}^n} F(w) = \mathbb{E}[f(w; \xi)],$$

where $f(w; \xi) = f(w; x_i, z_i) = \mathcal{L}(h(w; x_i); z_i))$.

## Empirical Form of Objective Function

$$F(w) = \frac{1}{N} \sum_{i=1}^{N} f(w; x_i, z_i)$$

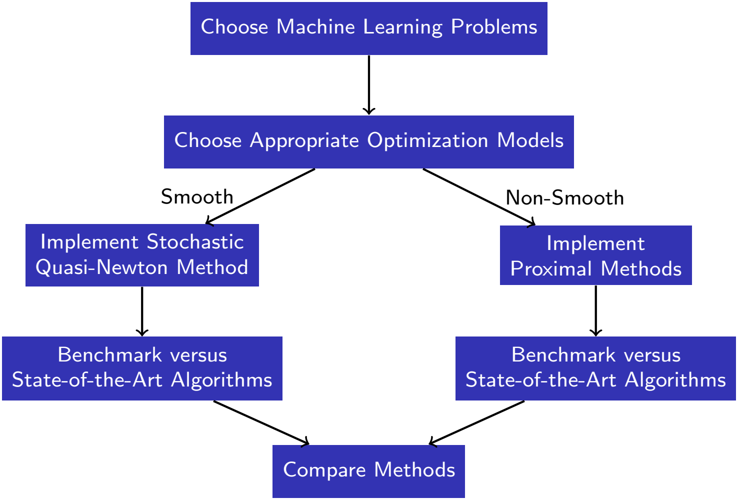
# Introduction (3)

### Mini-batch Stochastic Gradient

Consider small subset $\mathcal{S} \subset \{1, \ldots, N\}$, with $b := |S| \ll N$

Construct
$$\widehat{\nabla} F(w) = \frac{1}{b} \sum_{i \in \mathcal{S}} \nabla f(w; x_i, z_i)$$

# Structure of this Case Study



# Machine Learning Problems and Optimization Models

Possible Applications:

- Face recognition
- Text classification
- Speech recognition

# Machine Learning Problems and Optimization Models

Possible Applications:

- Face recognition
- Text classification
- Speech recognition

Possible Optimization Models:

- Linear Regression: $\min_w \frac{1}{N} \sum_{i=1}^{N} \|z_i - x_i w\|_2^2$
- Binary Classification:

$$f(w; x_i, z_i) = z_i \log(c(w; x_i)) + (1 - z_i) \log(1 - c(w; x_i))$$

  with $c(w; x_i) = \frac{1}{1+\exp(-x_i^T w)}$

- Neural Nets: Back propagation

# Stochastic Quasi-Newton Method (1)

Problem:

- Incorporating second-order information via full Hessian too expensive for large-scale problems
- Use of curvature information highly beneficial for algorithm performance

# Stochastic Quasi-Newton Method (1)

Problem:

- Incorporating second-order information via full Hessian too expensive for large-scale problems
- Use of curvature information highly beneficial for algorithm performance

Idea:

- Adapt BFGS method to stochastic framework
- Employ limited memory version of BFGS algorithm (L-BFGS)
- Compute gradient based on sample $\mathcal{S}$ of training set
- Compute Hessian update at regular intervals of length $L$ based on small subsample $\mathcal{S}_H$ of training set

# Stochastic Quasi-Newton Method (2)

## Iteration

$$w_{k+1} = w_k - \alpha_k H_t \widehat{\nabla} F(w_k)$$

## Hessian-Update

Choose

$$s_t = \bar{w}_t - \bar{w}_{t-1} \qquad y_t = \widehat{\nabla}^2 F(\bar{w}_t) s_t,$$

with $\bar{w}_t := \sum_{i=k-L}^{k} w_i$ and $\widehat{\nabla}^2 F(w) := \frac{1}{b_H} \sum_{i \in \mathcal{S}_H} \nabla^2 f(w; x_i, z_i)$.

Compute

$$H_{t+1} = (I - \rho_t s_t y_t^T) H_t (I - \rho_t y_t s_t^T) + \rho_t s_t s_t^T,$$

with $\rho_t = \frac{1}{y_t^T s_t}$.

# Stochastic Quasi-Newton Method (3)

**Stochastic L-BFGS Algorithm**

```
Initialize w_1, H_1, step-length sequence α_k > 0
for k = 1, ..., do
    Choose a sample S ⊂ {1, ..., N}
    Compute w_{k+1} = w_k − α^k H_t ∇̂F(w^k)
    if  mod (k, L) = 0 then
        Choose a sample S_H ⊂ {1, ..., N}
        Compute H_t
    end if
end for
```

Initialize $w_1$, $H_1$, step-length sequence $\alpha_k > 0$

**for** $k = 1, \ldots,$ **do**

  Choose a sample $\mathcal{S} \subset \{1, \ldots, N\}$

  Compute $w_{k+1} = w_k - \alpha^k H_t \widehat{\nabla} F(w^k)$

  **if** $\mod (k, L) = 0$ **then**

    Choose a sample $\mathcal{S}_H \subset \{1, \ldots, N\}$

    Compute $H_t$

  **end if**

**end for**

# Benchmarking of Stochastic Quasi-Newton Method

Challenge: Economical implementation of Algorithm is necessary for meaningful benchmarking

- Memory-efficient sparse coding
- Calculation of Hessian-Vector Product without storing the Hessian
- Computation of BFGS-Update via two-loop recursion

# Benchmarking of Stochastic Quasi-Newton Method

Challenge: Economical implementation of Algorithm is necessary for meaningful benchmarking

- Memory-efficient sparse coding
- Calculation of Hessian-Vector Product without storing the Hessian
- Computation of BFGS-Update via two-loop recursion

Benchmarking:

- Comparison to Stochastic Gradient Descent Method, Standard L-BFGS Method, (Stochastic) Conjugate Gradient Descent
- Comparison of run-time, accuracy, access-points etc. under different parameter regimes and objective functions

# Inducing Sparsity

## Dictionary Learning

$$\min_{D,\alpha} \frac{1}{N} \sum_{i=1}^{N} \|x_i - D\alpha_i\|_2^2 + \lambda \|\alpha_i\|_1$$

- control on $D$ and $\alpha$
- better convergence
- modifications of the algorithms

# Conclusion

## Situation

- Increasing amount of data in Machine Learning applications
- Need for robust and fast algorithms for smooth and non smooth optimization

## Stochastic Second-Order Methods

- Faster convergence through curvature information
- Moderate computational cost through mini-batches