Polynomial Regression

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} w_{jj'} \, x_{ij} \, x_{ij'}$$

- $\mathbf{x}_i \in \mathbb{R}^D$ is an example from the dataset $\mathbf{X} \in \mathbb{R}^{N \times D}$
- $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{W} \in \mathbb{R}^{D \times D}$ are parameters of the model

Polynomial Regression

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} w_{jj'} \, x_{ij} \, x_{ij'}$$

- Dense parameterization is not suited for sparse data
- Computationally expensive - $\mathcal{O}\left(N \times D^2\right)$

Factorization Machines

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j\, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle\, x_{ij}\, x_{ij'}$$

Factorization Machines

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j\, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle x_{ij}\, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$

Factorization Machines

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \langle \mathbf{v}_j, \mathbf{v}_{j'} \rangle \, x_{ij} \, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$

Factorization Machines

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j\, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle\, x_{ij}\, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$
- $\mathbf{v}_j \in \mathbb{R}^K$ denotes latent embedding for the j-th feature

Factorization Machines

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \, x_{ij} \, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$
- $\mathbf{v}_j \in \mathbb{R}^K$ denotes latent embedding for the j-th feature
- Computationally much cheaper - $\mathcal{O}\left(N \times D \times K\right)$

Factorization Machines

$$
\begin{aligned}
f\left(\mathbf{x}_i\right) &= w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \, x_{ij} \, x_{ij'} \\
&= w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \frac{1}{2} \sum_{k=1}^{K} \left\{ \left(\sum_{d=1}^{D} v_{dk} x_{id} \right)^2 - \sum_{j=1}^{D} v_{jk}^2 x_{ij}^2 \right\}
\end{aligned}
$$

Factorization Machines

$$\mathcal{L}(\mathbf{w}, \mathbf{V}) = \frac{1}{N}\sum_{i=1}^{N} l\left(f\left(\mathbf{x}_i\right), y_i\right) + \frac{\lambda_w}{2}\left(\|\mathbf{w}\|_2^2\right) + \frac{\lambda_v}{2}\left(\|\mathbf{V}\|_2^2\right)$$

- λ_w and λ_v are regularizers for **w** and **V**
- $l(\cdot)$ is loss function depending on the task

Factorization Machines

Gradient Descent updates (First-Order Features)

$$
\begin{aligned}
w_j^{t+1} &\leftarrow w_j^t - \eta \sum_{i=1}^{N} \nabla_{w_j} l_i \left(\mathbf{w}, \mathbf{V}\right) + \lambda_w \; w_j^t \\
&= w_j^t - \eta \sum_{i=1}^{N} G_i^t \cdot \nabla_{w_j} f\left(\mathbf{x}_i\right) + \lambda_w \; w_j^t \\
&= w_j^t - \eta \sum_{i=1}^{N} G_i^t \cdot x_{ij} + \lambda_w \; w_j^t
\end{aligned} \tag{1}
$$

where, multiplier G_i^t is given by,

$$
G_i^t = \begin{cases} f\left(x_i\right) - y_i, & \text{if squared loss (regression)} \\ \frac{-y_i}{1+\exp(y_i \cdot f_i(x_i))}, & \text{if logistic loss (classification)} \end{cases} \tag{2}
$$

Factorization Machines

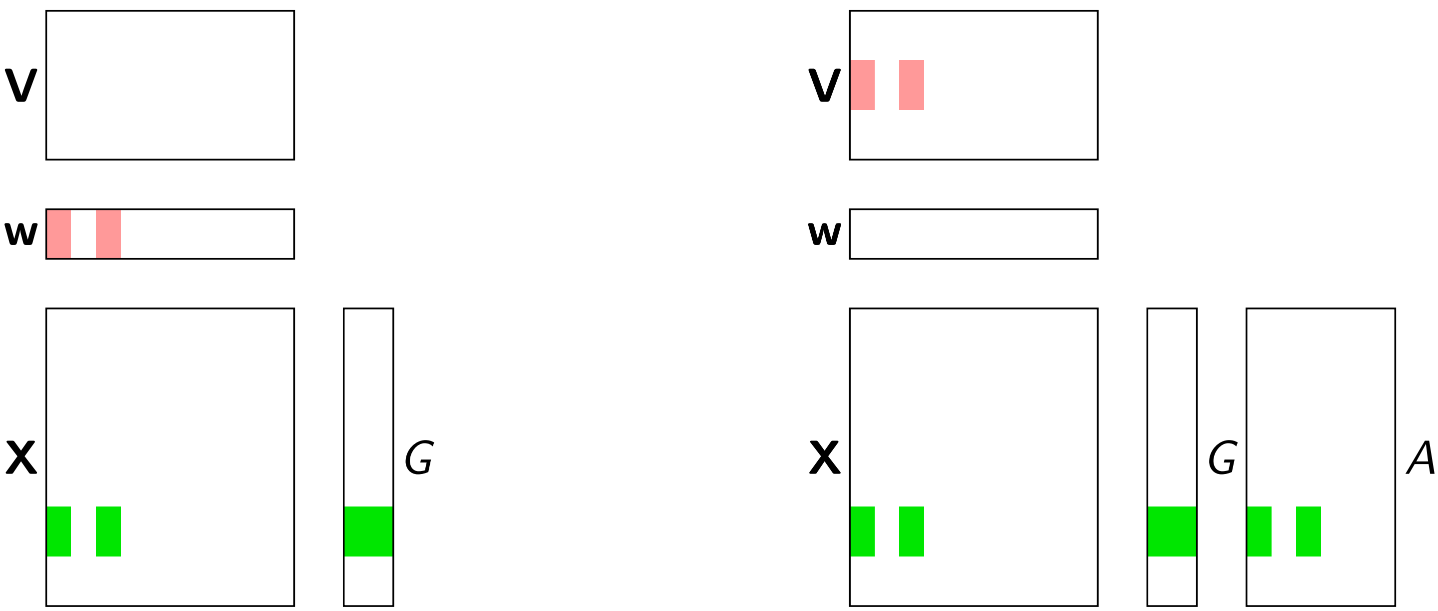
Gradient Descent updates (Second-Order Features)

$$
\begin{aligned}
v_{jk}^{t+1} &\leftarrow v_{jk}^{t} - \eta \sum_{i=1}^{N} \nabla_{v_{jk}} l_i\left(\mathbf{w}, \mathbf{V}\right) + \lambda_v \, v_{jk}^{t} \\
&= v_{jk}^{t} - \eta \sum_{i=1}^{N} G_i^t \cdot \nabla_{v_{jk}} f\left(\mathbf{x}_i\right) + \lambda_v \, v_{jk}^{t} \\
&= v_{jk}^{t} - \eta \sum_{i=1}^{N} G_i^t \cdot \left\{ x_{ij} \left(\sum_{d=1}^{D} v_{dk}^{t} \cdot x_{id} \right) - v_{jk}^{t} \, x_{ij}^2 \right\} + \lambda_v \, v_{jk}^{t} \qquad (3)
\end{aligned}
$$

where, multiplier G_i^t is same as before, synchronization term $a_{ik} = \sum_{d=1}^{D} v_{dk}^{t} \, x_{id}$.

Factorization Machines

Access pattern of parameter updates for w_j and v_{jk}

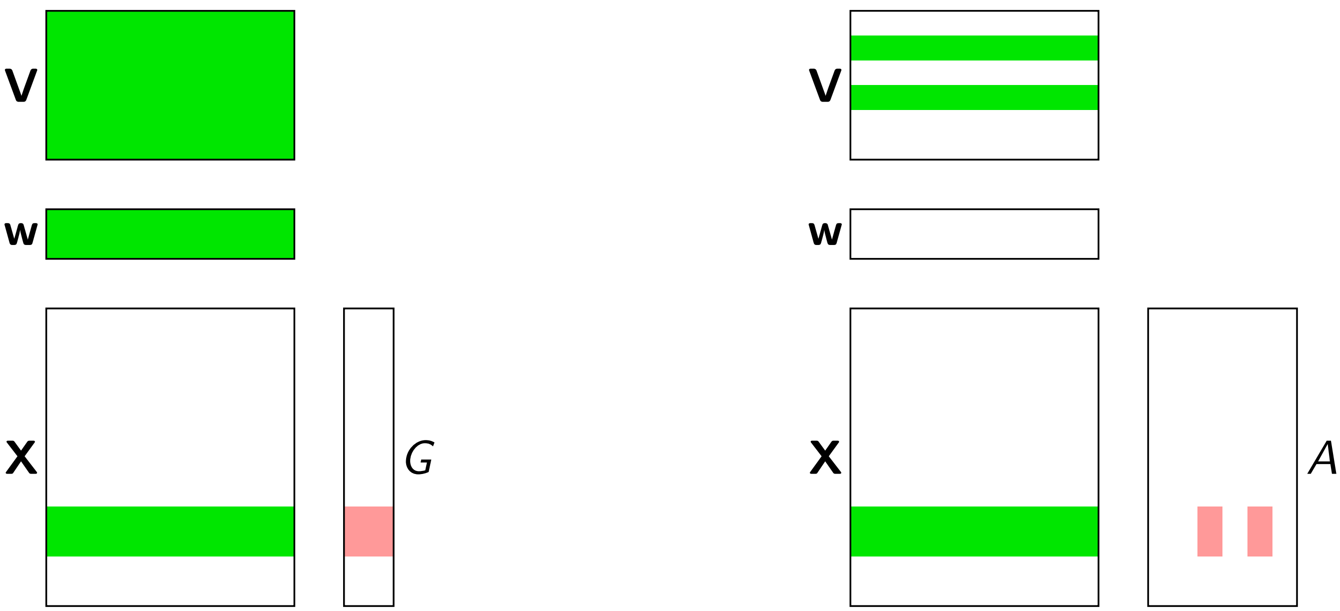


(a) w_j update (b) v_{jk} update

Figure: Updating w_j requires computing G_i and likewise updating v_{jk} requires computing a_{ik}.

Factorization Machines

Access pattern of parameter updates for w_j and v_{jk}



(a) computing G_i (b) computing a_{ik}

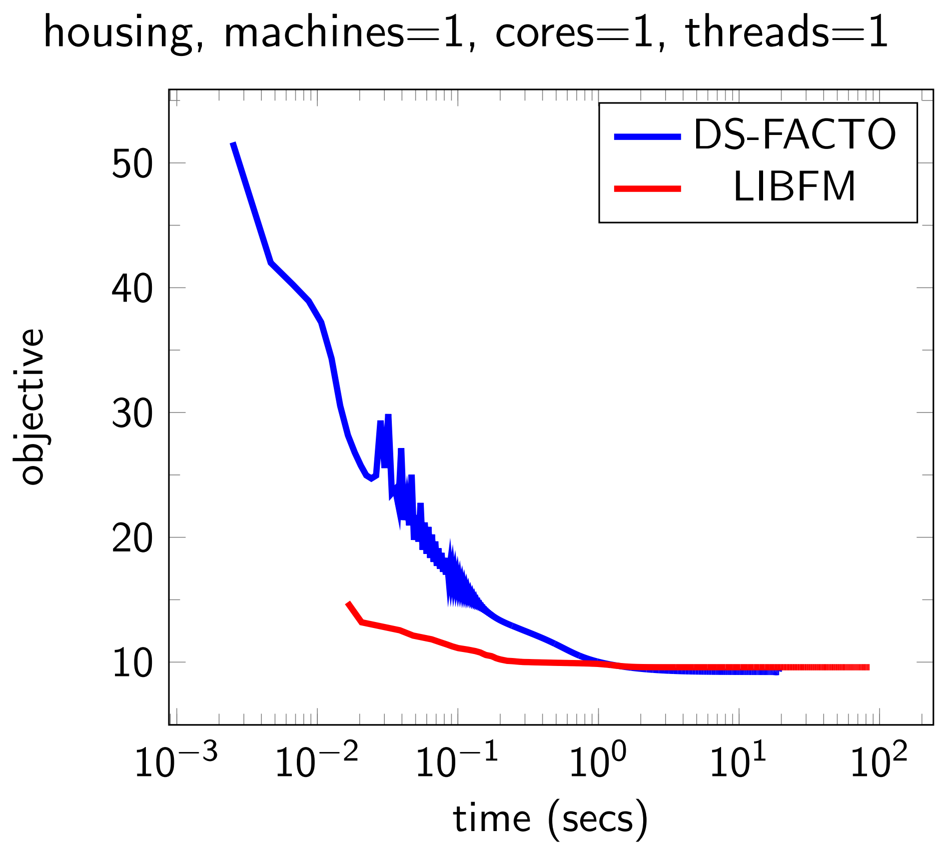
Figure: Computing both G and A requires accessing all the dimensions $j = 1, \ldots, D$. This is the main synchronization bottleneck.

Doubly-Separable Factorization Machines (DS-FACTO)

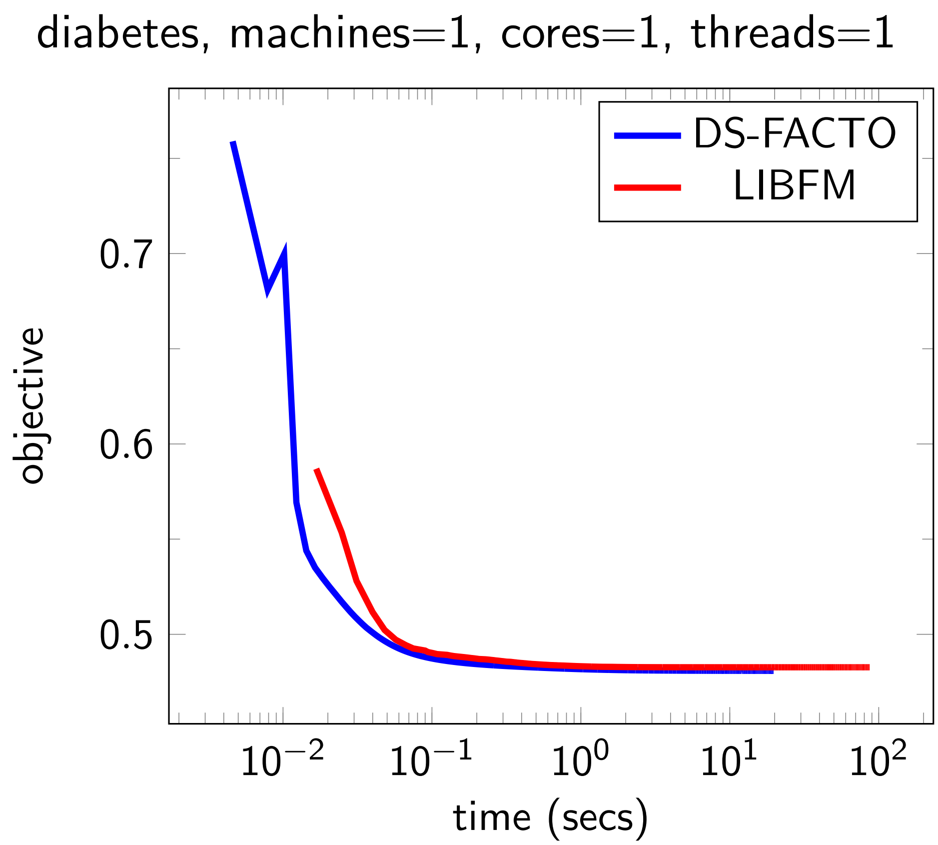
Avoiding bulk-synchronization

- Perform a round of Incremental synchronization after all D updates

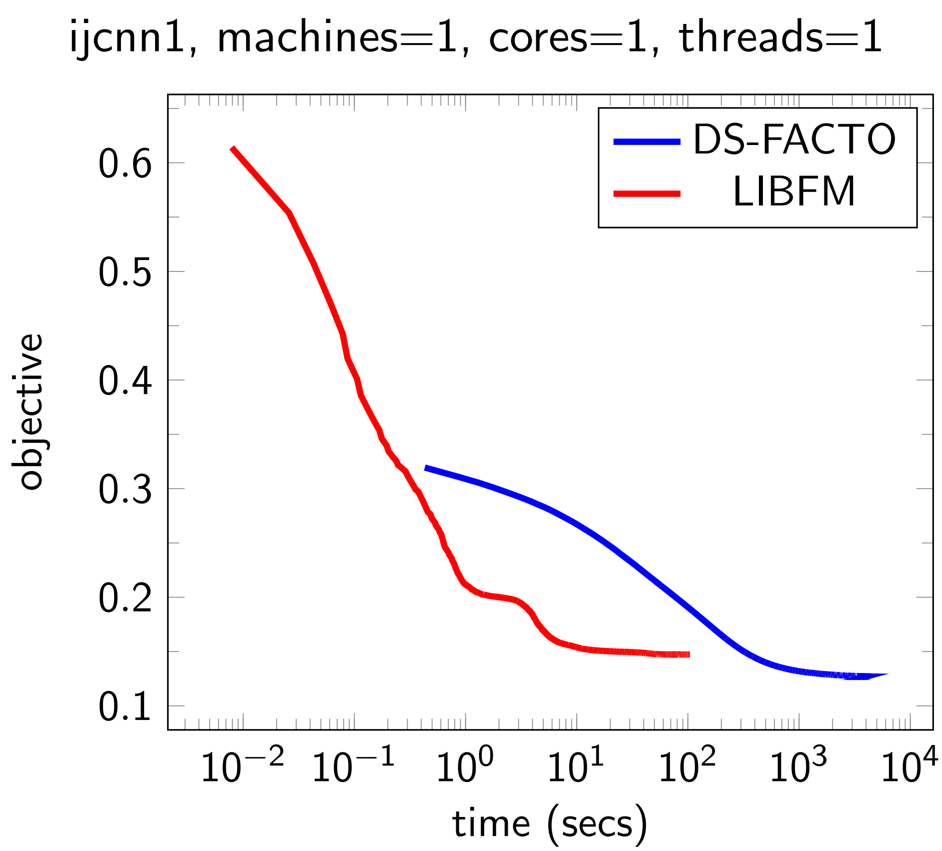
Experiments: Single Machine



Experiments: Single Machine

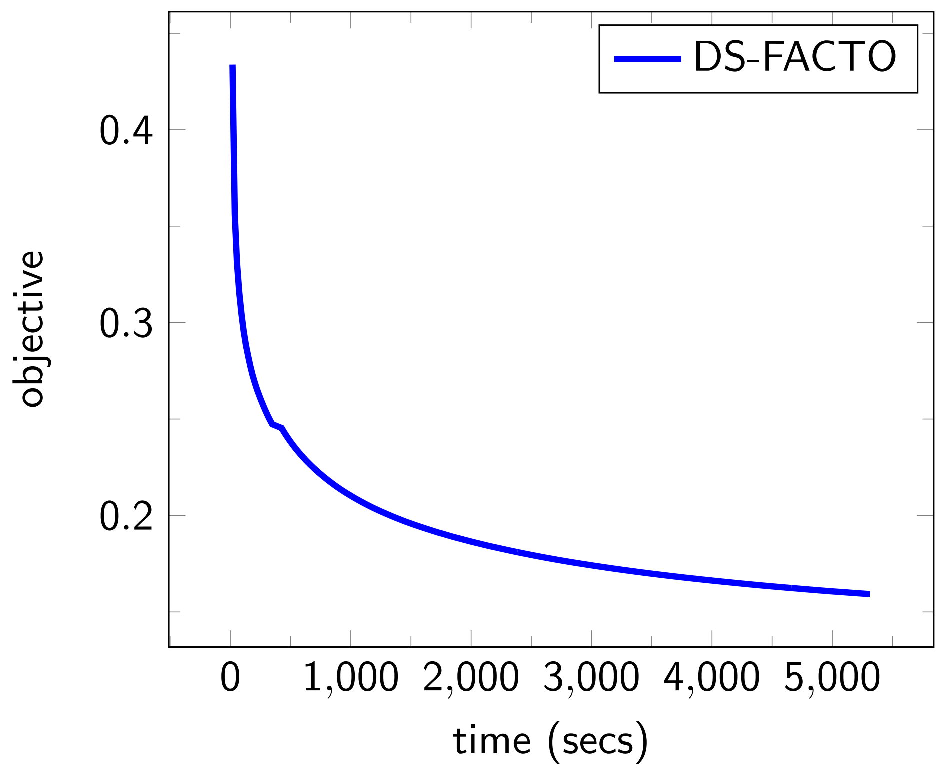


Experiments: Single Machine



Experiments: Multi Machine

realsim, machines=8, cores=40, threads=1



Experiments: Scaling in terms of threads

realsim, Varying cores and threads as 1, 2, 4, 8, 16, 32