Physics-Aware
Deep Nonnegative Matrix Factorization
(PAD-NMF)

Erik Chinellato
Fabio Marcuzzi
February 22, 2024
Deep-NMF: Introduction

Let us consider a regularized NMF problem in the form

\[
\begin{align*}
\text{minimize} & & D_1(X, WH) + \mu \|H\|_1 \\
\text{subject to} & & W \in M_{M \times R}(\mathbb{R}^+) \quad , \quad H \in M_{R \times N}(\mathbb{R}^+)
\end{align*}
\]

which is tackled by alternate optimization of the factors

\[
H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})
\]
Deep-NMF: Introduction

Let us consider a regularized NMF problem in the form

\[
\text{minimize} \quad D_1(X, WH) + \mu \| H \|_1 \\
\text{subject to} \quad W \in M_{M \times R}(\mathbb{R}^+) , \quad H \in M_{R \times N}(\mathbb{R}^+) 
\]

which is tackled by alternate optimization of the factors

\[
H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)}) 
\]

By interpreting the iterative update scheme as a neural network, where \(H^{(k+1)}\) is the output of the \(k\)-th layer given the input \(H^{(k)}\) and activation function \(f\), Deep-NMF unfolds the iterations and unties the bases across layers: the result is a trainable neural network with parameters \(\{W^{(k)}\}_{k=0, \ldots, K}\).

\[
\begin{align*}
&H^{(0)} \xrightarrow{W^{(0)}} H^{(1)} \xrightarrow{W^{(1)}} \ldots \xrightarrow{W^{(K-2)}} H^{(K-1)} \xrightarrow{W^{(K-1)}} H^{(K)} \xrightarrow{-} E(W^{(K)}, H^{(K)}) \\
&X \xrightarrow{} \ldots \xrightarrow{} \ldots \xrightarrow{} \ldots \xrightarrow{} 
\end{align*}
\]
Deep-NMF: Introduction

Let us consider a regularized NMF problem in the form

\[
\begin{align*}
\text{minimize} & \quad D_1(X, WH) + \mu \|H\|_1 \\
\text{subject to} & \quad W \in \mathcal{M}_{M \times R}(\mathbb{R}^+), \quad H \in \mathcal{M}_{R \times N}(\mathbb{R}^+)
\end{align*}
\]

which is tackled by alternate optimization of the factors

\[
H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})
\]

By interpreting the iterative update scheme as a neural network, where \(H^{(k+1)}\) is the output of the \(k\)-th layer given the input \(H^{(k)}\) and activation function \(f\), Deep-NMF unfolds the iterations and unties the bases across layers: the result is a trainable neural network with parameters \(\{W^{(k)}\}_{k=0,\ldots,K}\).

\[
\begin{align*}
H^{(0)} & \xrightarrow{W^{(0)}} H^{(1)} \xrightarrow{W^{(1)}} \cdots \xrightarrow{W^{(K-1)}} H^{(K-1)} \xrightarrow{W^{(K-1)}} H^{(K)} \xrightarrow{\mathcal{E}(W^{(K)}, H^{(K)})}
\end{align*}
\]

Q: Why do we care about Deep-NMF? Why is it useful?
Deep-NMF: Introduction

Let us consider a regularized NMF problem in the form

\[
\text{minimize} \quad D_1(X, WH) + \mu \|H\|_1 \\
\text{subject to} \quad W \in \mathcal{M}_{M \times R}(\mathbb{R}^+) \quad , \quad H \in \mathcal{M}_{R \times N}(\mathbb{R}^+)
\]

which is tackled by alternate optimization of the factors

\[H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})\]

By interpreting the iterative update scheme as a neural network, where \(H^{(k+1)}\) is the output of the \(k\)-th layer given the input \(H^{(k)}\) and activation function \(f\), Deep-NMF unfolds the iterations and unties the bases across layers: the result is a trainable neural network with parameters \(\{W^{(k)}\}_{k=0, \ldots, K}\).

\[
\begin{align*}
H^{(0)} & \xrightarrow{W^{(0)}} H^{(1)} & \xrightarrow{W^{(1)}} \cdots & \xrightarrow{W^{(K-2)}} H^{(K-1)} & \xrightarrow{W^{(K-1)}} H^{(K)} & \xrightarrow{} \mathcal{E}(W^{(K)}, H^{(K)}) \\
X & \xrightarrow{} \cdots & \xrightarrow{} & \xrightarrow{} & \xrightarrow{} & \xrightarrow{}
\end{align*}
\]

Q: Why do we care about Deep-NMF? Why is it useful?

A: It provides a nonnegative, additive decomposition of \(X\)

\[X \approx W^{(K)} H^{(K)} = W_S^{(K)} H_S^{(K)} + W_N^{(K)} H_N^{(K)} = S + N\]

where

\[
W^{(K)} = \begin{bmatrix} W_S^{(K)} & W_N^{(K)} \end{bmatrix} \quad H^{(K)} = \begin{bmatrix} H_S^{(K)^\top} & H_N^{(K)^\top} \end{bmatrix}^\top
\]
Deep-NMF: A quick look at the backpropagation algorithm

Since the weights $W^{(k)}$ must remain nonnegative to retain interpretability, the backpropagation algorithm is non-conventional.

Indeed, the weights are updated with:

$$W^{(k)} \leftarrow W^{(k)} \odot \frac{[\nabla W^{(k)} E]_-}{[\nabla W^{(k)} E]_+}$$
Since the weights $W^{(k)}$ must remain nonnegative to retain interpretability, the backpropagation algorithm is non-conventional.

Indeed, the weights are updated with:

$$W^{(k)} \leftarrow W^{(k)} \odot \left( \frac{\nabla W^{(k)} E}{\nabla W^{(k)} E} \right)$$

Hence the need to split the gradients into positive and negative parts and back-propagate both quantities:

$$\frac{\partial E}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} = \sum_{m,r} \left( \frac{\partial E}{\partial H^{(k+1)}_{r,m}} \right)_+ \left( \frac{\partial H^{(k+1)}_{r,m}}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} \right)_+ + \left( \frac{\partial E}{\partial H^{(k+1)}_{r,m}} \right)_- \left( \frac{\partial H^{(k+1)}_{r,m}}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} \right)_-$$

$$\frac{\partial E}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} = \sum_{m,r} \left( \frac{\partial E}{\partial H^{(k+1)}_{r,m}} \right)_+ \left( \frac{\partial H^{(k+1)}_{r,m}}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} \right)_- + \left( \frac{\partial E}{\partial H^{(k+1)}_{r,m}} \right)_- \left( \frac{\partial H^{(k+1)}_{r,m}}{\partial W^{(k)}_{\tilde{n},\tilde{r}}} \right)_+$$

$$\frac{\partial E}{\partial H^{(k)}_{r,\tilde{m}}} = \sum_{r} \left( \frac{\partial E}{\partial H^{(k+1)}_{r,\tilde{m}}} \right)_+ \left( \frac{\partial H^{(k+1)}_{r,\tilde{m}}}{\partial H^{(k)}_{r,\tilde{m}}} \right)_+ + \left( \frac{\partial E}{\partial H^{(k+1)}_{r,\tilde{m}}} \right)_- \left( \frac{\partial H^{(k+1)}_{r,\tilde{m}}}{\partial H^{(k)}_{r,\tilde{m}}} \right)_-$$

$$\frac{\partial E}{\partial H^{(k)}_{r,\tilde{m}}} = \sum_{r} \left( \frac{\partial E}{\partial H^{(k+1)}_{r,\tilde{m}}} \right)_+ \left( \frac{\partial H^{(k+1)}_{r,\tilde{m}}}{\partial H^{(k)}_{r,\tilde{m}}} \right)_- + \left( \frac{\partial E}{\partial H^{(k+1)}_{r,\tilde{m}}} \right)_- \left( \frac{\partial H^{(k+1)}_{r,\tilde{m}}}{\partial H^{(k)}_{r,\tilde{m}}} \right)_+$$
The Deep-NMF algorithm and architecture was originally designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean components $S$ have strong **intrinsic structure** that should be preserved by the network.

We proposed several **physics-aware enhancements**: 

- Creation of an optimal discriminative dictionary $\hat{W}(K)$ (not modified by back-propagation) which enables the recognition of physically-characterized clean components;
- Embedding of the (time-)correlation between consecutive columns of $X$ by employing block-Hankel weights $W(k)$;
- Preservation of the block-Hankel structure by projection, thus modifying both the forward- and back-propagation;
- Construction of a suitable loss function for the training process, enforcing an accurate reconstruction of the clean component.
The Deep-NMF algorithm and architecture was originally designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean components $S$ have strong *intrinsic structure* that should be preserved by the network.

We proposed several *physics-aware enhancements*:

- Creation of an **optimal discriminative dictionary** $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized clean components;
The Deep-NMF algorithm and architecture was originally designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean components $S$ have strong *intrinsic structure* that should be preserved by the network.

We proposed several *physics-aware enhancements*:

- Creation of an optimal discriminative dictionary $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized clean components;

- Embedding of the (time-)correlation between consecutive columns of $X$ by employing *block-Hankel* weights $W^{(k)}$;
The Deep-NMF algorithm and architecture was originally designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean components $S$ have strong *intrinsic structure* that should be preserved by the network.

We proposed several *physics-aware enhancements*:

- Creation of an optimal discriminative dictionary $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized clean components;

- Embedding of the (time-)correlation between consecutive columns of $X$ by employing *block-Hankel* weights $W^{(k)}$;

- Preservation of the block-Hankel structure by *projection*, thus modifying both the forward- and back-propagation;
The Deep-NMF algorithm and architecture was originally designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean components $S$ have strong *intrinsic structure* that should be preserved by the network.

We proposed several *physics-aware enhancements*:

- Creation of an *optimal discriminative dictionary* $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized clean components;

- Embedding of the (time-)correlation between consecutive columns of $X$ by employing *block-Hankel* weights $W^{(k)}$;

- Preservation of the block-Hankel structure by *projection*, thus modifying both the forward- and back-propagation;

- Construction of a *suitable loss function* for the training process, enforcing an accurate reconstruction of the clean component.

$$\mathcal{E} = \|S - F \circ X\|_2^2 + \gamma \|S - W^{(K)}_S H^{(K)}_S\|_2^2$$

- **Penalty hyperparameter**
- **Penalty term**
- **Problem-specific term (Wiener filter)**
Example 1: Hits detection on synthetic data
Example 2: Hits detection on real data