Investigating Nonnegative Autoencoders for Efficient Audio Decomposition

Yigitcan Özer, Jonathan Hansen, Tim Zunner, Meinard Müller
International Audio Laboratories Erlangen, Erlangen, Germany
yigitcan.oezer@audiolabs-erlangen.de, meinard.mueller@audiolabs-erlangen.de

***Abstract*—Nonnegative Matrix Factorization (NMF) is a powerful technique for decomposing a music recording's magnitude spectrogram into musically meaningful spectral and activation patterns. In recent years, musically informed NMF-based audio decomposition has been simulated using neural networks, which opens up new paths of exploiting recent deep learning frameworks, including libraries for efficient gradient computations. In this article, we continue this strand of research by considering Nonnegative Autoencoders (NAE) in combination with gradient projection and structured dropout techniques. Conducting experiments based on piano recordings, we compare the decomposition results of NAE-based approaches with those obtained from a score-informed NMF variant. In this context, we examine various gradient descent methods using fixed and adaptive learning rates for deriving the NAE encoder and decoder parameters. Among others, we show how the famous multiplicative update rules for NMF can be transferred to the case of NAEs. The overall goal of our contribution is to illustrate the benefits and limitations of the various techniques concerning implementation issues (CPU, GPU), convergence speed, and overall runtime.**

***Index Terms*—Nonnegative Autoencoders, Adaptive Gradient Methods, Nonnegative Matrix Factorization, Audio Decomposition, Audio Source Separation**

I. Introduction

Nonnegative Matrix Factorization (NMF) is a prominent low-rank factorization method that imposes nonnegativity constraints in all matrices involved. Notably, its effectiveness and ability to yield *interpretable* results have attracted great attention in various research fields [1], [2]. In the context of music processing, NMF has been widely applied for the decomposition of complex sound mixtures, using the magnitude spectrogram of music signals as input representation [3]–[10]. As a result of the decomposition, NMF approximates the magnitude spectrogram by the product of two nonnegative matrices, where the columns of the first matrix encode spectral prototype patterns (called *templates*) and the rows of the second matrix encode their occurrences in time (called *activations*).

Motivated by recent advances in designing and training neural networks, Smaragdis and Venkataramani [11] introduced a Nonnegative Autoencoder (NAE) architecture as a neural network alternative for NMF-based audio decomposition. Fig. 1 gives an overview of the simulation of NMF through a shallow NAE architecture, which comprises a single-layer encoder and a single-layer decoder. The NAE decoder directly corresponds to the NMF template matrix. However, rather than learning an activation matrix as in NMF, the NAE learns an encoder which yields an activation matrix as output (also called *code*). To ensure the nonnegativity constraints of templates and activations, one can combine NAEs with gradient projection and structured dropout techniques [12]. The simulation of NMF through NAE makes it possible to exploit recent deep learning frameworks including libraries for automatic and GPU-accelerated gradient computations. This may also open up new paths for tackling the audio decomposition problem with deeper and more complex models.

This work was supported by the German Research Foundation (DFG MU 2686/10-2). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institut für Integrierte Schaltungen IIS.

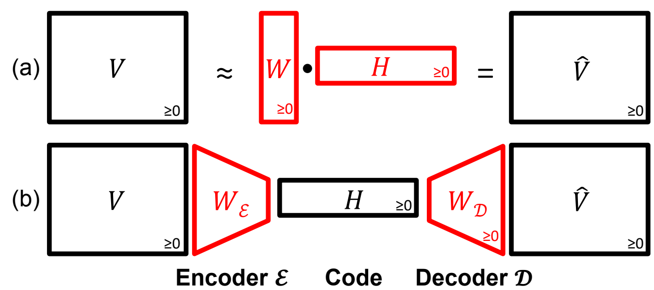

Fig. 1. **(a)** NMF used for decomposing a nonnegative matrix V into the product of a nonnegative template matrix W and nonnegative activation matrix H. **(b)** Simulation of the decomposition using NAE (see the text for details). The learned components are shown in red.

As starting point of this paper, we consider the work by Ewert and Müller [13], which uses a score-informed NMF variant for decomposing the magnitude spectrograms of piano recordings. As the main contribution of this paper, we simulate this original approach by considering different NAE variants (inspired by [11], [12]) and conduct systematic experiments to compare the resulting decompositions with the NMF-based approach used as a reference. In particular, we show how one can adapt the famous multiplicative update rules of NMF [1] to the case of NAEs. Furthermore, we investigate projected versions of additive gradient descent methods such as Stochastic Gradient Descent (SGD), Root Mean Square Propagation (RMSprop) [14], and Adaptive Moment Optimization (ADAM) [15]. Our systematic experiments highlight the benefits and limitations of different techniques in terms of implementation issues, convergence speed, and overall

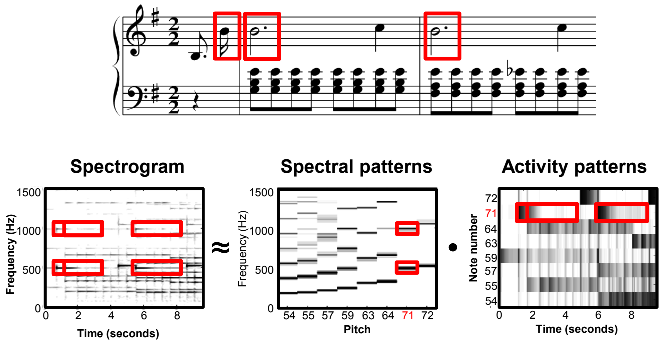

Fig. 2. Decomposing the magnitude spectrogram of an audio excerpt of Chopin's Prelude Op.28 No.4 into template and activation matrices. The information related to the note number $p = 71$ (B4) is indicated by the red rectangular frames.

runtime.

The remainder of the paper is organized as follows. In Section II, we provide an overview of the score-informed NMF approach that serves as our reference. In Section III, we investigate the simulation of NMF through NAE and introduce the multiplicative update rules for NAE. In Section IV, we report on our systematic experiments and conclude in Section V with prospects on future work.

II. Score-Informed NMF for Audio Decomposition

NMF is a nonnegative factorization algorithm that accounts for an additive, part-based representation of a nonnegative input matrix. Nonnegative matrix entries prevent undesired effects such as destructive inferences, where a positive component might be canceled out by adding a kind of inverse (negative) component.

Given the magnitude spectrogram of a music recording $V \in \mathbb{R}_{\geq 0}^{K\times N}$ and a *target rank* $R \in \mathbb{N}$ that is much smaller than both $K \in \mathbb{N}$ and $N \in \mathbb{N}$, NMF seeks an optimal approximation $V \approx WH$ enforcing both learned matrices $W \in \mathbb{R}^{K\times R}$ and $H \in \mathbb{R}^{R\times N}$ to be nonnegative. As shown in Fig. 2, W indicates the *template matrix* and H the *activation matrix*, where K and N, respectively, denote the number of frequency and time bins in the input spectrogram. In this example, the target rank R corresponds to the number of distinct pitches played in the input music recording. The loss function of the least-square optimization problem (with additional nonnegativity constraints for W and H) can be written as

$$\varphi(W, H) = ||V - WH||_F^2, \tag{1}$$

where $||\cdot||_F$ is the Frobenius norm.

Alternating Least Squares (ALS) defines the optimization procedure, where the first matrix W is updated with fixed H, and then H is updated with fixed W, and so on. In particular, the iterative multiplicative update rules in [1] for NMF have proven to be easy and efficient. The crucial idea is to use an adaptive learning rate, which transforms the additive update rules of the usual gradient descent to multiplicative ones, resulting in

$$\begin{aligned} H &\leftarrow H \odot (W^\top V) \oslash (W^\top WH + \varepsilon), \\ W &\leftarrow W \odot (VH^\top) \oslash (WHH^\top + \varepsilon), \end{aligned} \tag{2}$$

for the case of the Euclidean loss. Here, $\odot$ and $\oslash$ denote pointwise multiplication and division, respectively. The parameter ε denotes the machine epsilon, which is used to avoid division by 0.

Besides nonnegativity constraints, prior musical knowledge, e.g., coming from a musical score, can also be easily integrated into the learning process of NMF to guide the decomposition [8], [16], [17]. Multiplicative update rules in Eq. 2 ensure that the zero-valued matrix entries in the template and activation matrices remain zero during the entire learning process. Therefore, one can avoid undesired template and activation values by initiating the corresponding positions in the matrices with zero. In [13], the templates are initialized using a sparse, binary matrix $W^{\mathrm{C}} \in \{0,1\}^{K\times R}$ to constrain frequencies and enforce an overtone model. Similarly, using the score information, the activation matrix can be constrained through a sparse, binary matrix $H^{\mathrm{C}} \in \{0,1\}^{R\times N}$. As an example, the red boxes in Fig. 2 indicate spectral and activation constraints (initialized with one-values inside and with zero-values outside the red boxes) corresponding to the note number $p = 71$ (B4).

This work uses the multiplicative NMF with Euclidean loss as the reference model. We apply the same score-informed initialization procedure as described in [13].

III. Simulation via Constrained NAEs

Following [11], [12], we now show how one can simulate constrained NMF via an NAE model in combination with projected gradient descent methods and rectifier activation functions.

The NMF model can be reformulated through a simple linear autoencoder [11], [18] as

$$\begin{aligned} H &= W_{\mathcal{E}} V, \\ \hat{V} &= W_{\mathcal{D}} H. \end{aligned}$$

The matrix $W_{\mathcal{E}} \in \mathbb{R}^{R\times K}$ denotes the *encoder*, which yields the activation matrix H as output. The decoder $W_{\mathcal{D}} \in \mathbb{R}^{K\times R}$ can be thought of as the equivalent to the template matrix W in the NMF decomposition. To ensure the nonnegativity of the activation output matrix H and the template weight matrix $W_{\mathcal{D}}$ in NAE, one has to introduce further constraints.

Our proposed NAE model applies a rectified linear unit (ReLU) after the encoder layer as in [19] to ensure the nonnegativity of the activation matrix H. For the nonnegativity of the decoder matrix $W_{\mathcal{D}}$, we use a projected gradient descent method as in [20], setting the negative values in $W_{\mathcal{D}}$ to zero during training. Chorowski and Zurada [21] state that constraining the weight matrices to be nonnegative improves the interpretability of an autoencoder's operation, whereas it does not lower the network's capability. In contrast, our experiments showed that applying a simple ReLU after the encoder layer resulted in a better convergence, rather than using projected gradients for the encoder layer as well.

As in the NMF case, prior music knowledge can also be integrated into the NAE model to guide the learning process. Ewert and Sandler introduced the *structured dropout* for activation constraints in [12]. Dropout layers typically regularize networks to avoid overfitting by randomly setting neurons to zero during the training process [22]. In contrast, structured dropout imposes prior musical knowledge and selectively removes undesired activations by setting

$$H' = H^{\mathrm{C}} \odot H.$$

To enforce structured dropout, one can adapt the loss function in Eq. 1 to the constrained NAE case as follows:

$$\begin{aligned}\varphi(W_{\mathcal{E}}, W_{\mathcal{D}}) &= \|V - W_{\mathcal{D}} H'\|_F^2 \\ &= \|V - W_{\mathcal{D}}(\sigma(W_{\mathcal{E}} V) \odot H^{\mathrm{C}})\|_F^2,\end{aligned}$$

where σ denotes the ReLU activation function. Computing the gradients with respect to the encoder and decoder matrices, one can derive multiplicative update rules for this NAE model similar to the NMF case:

$$\begin{aligned}W_{\mathcal{E}} \leftarrow W_{\mathcal{E}} \odot \Big(\big(((W_{\mathcal{D}}^\top V) \odot H^{\mathrm{C}}) V^\top \big) \oslash \\ \big(((W_{\mathcal{D}}^\top W_{\mathcal{D}} H') \odot H^{\mathrm{C}}) V^\top + \varepsilon \big) \Big), \\ W_{\mathcal{D}} \leftarrow W_{\mathcal{D}} \odot \big((V {H'}^\top) \oslash (W_{\mathcal{D}} H' {H'}^\top + \varepsilon) \big).\end{aligned} \tag{3}$$

For a derivation of the multiplicative update rules for NAE, we refer to [23]. We call this model as *multiplicative NAE*.

To train an NAE, additive methods like stochastic gradient descent (SGD), in which the learning rate remains constant during training, can also be used. Another alternative is the integration of other adaptive strategies for optimizers such as Root Mean Square Propagation (RMSprop) [14] and Adaptive Moment Optimization (ADAM) [15], which adjust the learning rate during training.

As for multiplicative NMF, the multiplicative NAE has the property that zero-valued entries remain zero. To enforce this property for template weights $W_{\mathcal{D}}$ also in the case of using additive update rules, we add further projection by applying binary masking on $W_{\mathcal{D}}$ using the constrain matrix W^{C}:

$$W_{\mathcal{D}} \leftarrow W_{\mathcal{D}} \odot W^{\mathrm{C}}.$$

IV. Experiments

This section reports on our experiments where we compare various NAE-based approaches with the score-informed NMF model used as reference. To this end, we decompose the magnitude spectrograms of piano recordings into musically meaningful spectral vectors and their activations. In our experiments, we use eight publicly-available, nonsynthetic piano recordings using the same experimental setting as in [13].[1]

[1] http://resources.mpi-inf.mpg.de/MIR/ICASSP2012-ScoreInformedNMF/

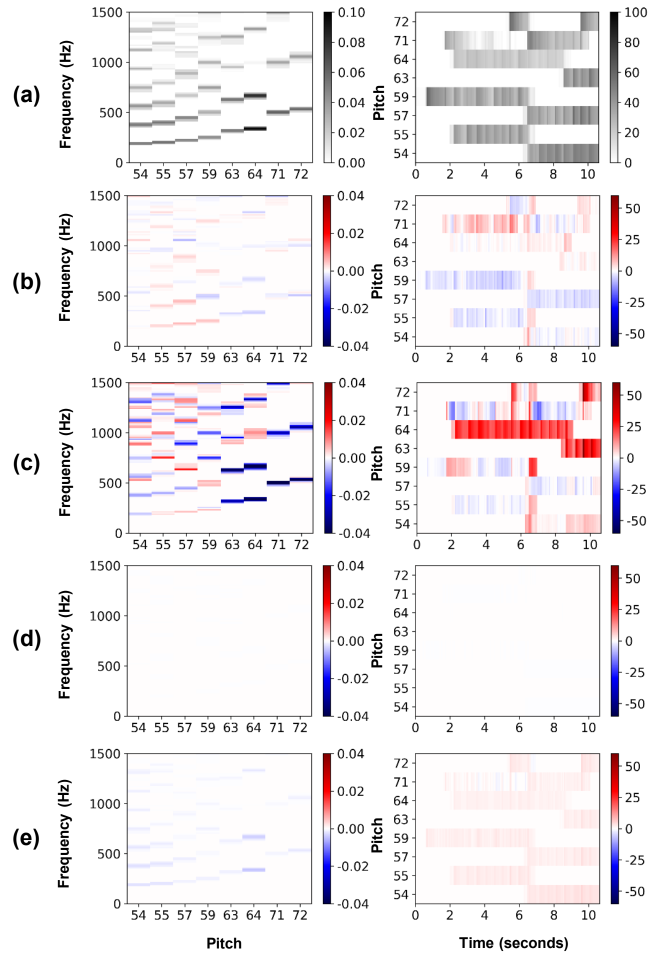

Fig. 3. Continuation of our Chopin example from Fig. 2. **(a)** Template matrix W (left) and activation matrix H (right) learned by the score-informed NMF model. **(b)-(e)** Difference between template (left) and activation (right) matrices obtained from NMF (used as reference) and NAE-based approaches. The columns of W and $W_{\mathcal{D}}$ are ℓ^1-normalized. **(b)** NAE trained with multiplicative update rules. **(c)** NAE trained with SGD with a fixed learning rate $\gamma = 0.1$. **(d)** NAE trained with ADAM. **(e)** NAE trained with RMSprop.

The pieces are listed in the Table I. The music recordings are mono and sampled at 22.05 kHz. Their durations vary between around 100 seconds and 9 minutes.

In the preprocessing phase, we compute the magnitude spectrograms of each recording using a Hann window of size 2048 and a hop size of 1024. For the reference NMF model, we use the same initialization procedure as in [13]. Similarly, we initialize the decoder matrix $W_{\mathcal{D}}$ of NAEs using the binary constrained matrix W^C, while we initialize and the encoder matrix $W_{\mathcal{E}}$ randomly. At the end of the training of each model, we ℓ^1-normalize the columns of the learned matrices W and $W_{\mathcal{D}}$, and accordingly scale the columns of the activation matrix H. This normalization and rescaling accounts for the scale ambiguity in the NMF decomposition and makes the decomposition results better comparable.

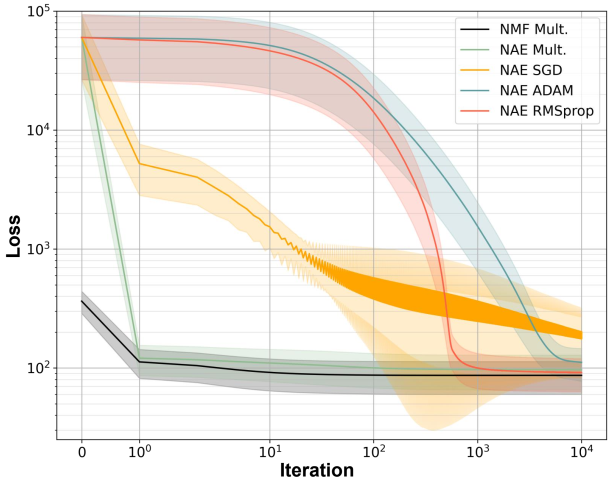

Fig. 4. Average column-wise absolute approximation loss between $\hat{V}$ and V per iteration evaluated on the entire dataset. All the NAE variants use the same weight initialization procedure.

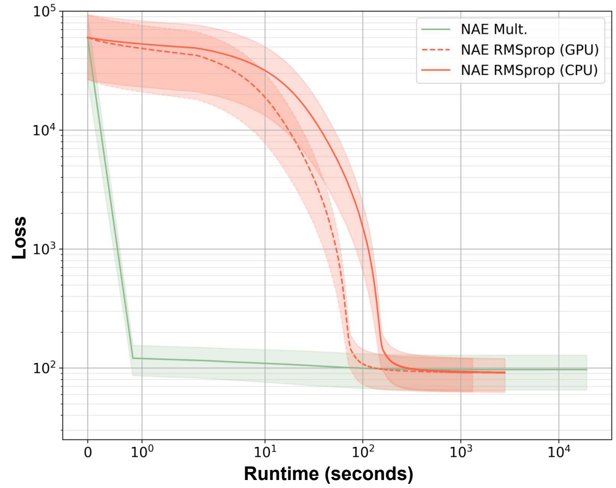

Fig. 5. Runtime comparison of the multiplicative NAE and NAE variants trained with RMSProp. The NAE variant trained on GPU with RMSprop is shown with dashed lines.

TABLE I
APPROXIMATION ERROR BETWEEN V AND $\hat{V}$ (COLUMNWISE AVERAGE) OF NMF AND NAE-BASED APPROACHES

Model / File ID	NMF Mult.	NAE Mult.	NAE SGD	NAE ADAM	NAE RMSprop
Chopin_Op028-04_SMD	46.4	49.0	62.4	57.6	48.1
Chopin_Op028-15_SMD	48.5	53.2	67.3	66.2	50.9
Chopin_Op066_SMD	79.5	87.2	139.0	101.1	85.7
Beethoven_Op031No2-01_SMD	90.7	99.2	105.1	104.4	94.9
Chopin_Op028-01_SMD	94.8	103.6	299.9	122.8	97.4
Bach_BWV875-01_SMD	97.5	107.3	219.9	129.2	104.3
Beethoven_Op111-01_EA	103.7	129.4	328.5	148.4	113.0
Chopin_Op064No1_EA	131.9	145.9	383.6	161.6	137.2

In the following, we regard an iteration to be the update of both the matrices W and H in the NMF case, and similarly $W_{\mathcal{E}}$ and $W_{\mathcal{D}}$ in the NAE case. In our experiments, we performed $10,000$ iterations in the training phase. During training, we used a learning rate of $\gamma = 0.1$ for the SGD, and the recommended default values for the RMSprop [14] and ADAM [15] optimizers.

Implementing the multiplicative NMF and NAE is straightforward using the derived update rules in Eq. 2 and Eq. 3 respectively. We implemented the multiplicative models with NumPy using matrix operations. Furthermore, we used the Tensorflow library to exploit the automatic gradient computation and GPU acceleration to train the NAE models that use additive gradient descent techniques. For the GPU-based computations we used a single Nvidia GTX 1080 Ti GPU.

To get a first impression of the approximation behavior of the various decomposition approaches, Fig. 3 shows a comparison of learned template and activation matrices learned by the reference NMF model and ones learned by the various NAE-based approaches. First, note that Multiplicative NMF and NAE yield similar template and activation matrices. Furthermore, among the additive NAE-based approaches, the NAE variant trained with SGD leads to the worst results compared to the NMF reference. Our comparison also indicates that NAEs trained with adaptive gradient descent methods lead to template and activation matrices close to the NMF case when using a huge number of iterations (up to $10,000$ in our experiments).

Next, we compare the approximation quality of the various decompositions in a quantitative fashion. Table I shows a comparison of approximation errors between V and $\hat{V}$ yielded by the NMF reference and NAE-based approaches based on the entire dataset. Here, each entry indicates the average columnwise ℓ^1-error between the approximation matrix $\hat{V}$ and the input spectrogram V. For example, the first row shows the results obtained from the spectrogram decomposition of the entire recording of Chopin's Prelude Op.28 No.4. The multiplicative NMF results in the approximation error of 46.4, and the multiplicative NAE in a similar value of 49.0. Among the NAE variants trained with additive gradient descent techniques, RMSprop reaches the smallest approximation error of 48.1, whereas SGD results in the highest approximation error of 62.4. Moreover, we can infer that the reference NMF model and NAE variants perform similarly over the entire dataset: the multiplicative NMF leads to the best approximation, while NAE with RMSprop results in the smallest approximation error among the NAE variants.

In our next experiment, we analyze the convergence behavior of all approaches over the number of iterations. Fig. 4 illustrates the mean and standard deviations per iteration over the columnwise Euclidean error, evaluated using all eight recordings in the dataset. The rapid decay in error after the first iteration of the multiplicative models is remarkable, whereas additive NAE variants need more iterations until they reach a steeper decline in the error. NAE with SGD shows a slow and unstable convergence behavior, resulting in a poor approximation even after $10,000$ iterations. We also have tried using other learning rates for the SGD case; however, it is unclear how to choose an optimal learning rate that guarantees convergence. $\gamma = 0.1$ has shown the best performance among various learning rates. In contrast, NAE with ADAM converges after $10,000$ iterations to a similar decomposition as NMF. Similarly, NAE with RMSprop converges to this result after only $1,000$ iterations. It is also worthwhile to note that the both adaptive NAE variants reach a decomposition result as NMF, although NAEs learn fewer parameters than the NMF reference. (The encoder $W_{\mathcal{E}} \in \mathbb{R}^{R\times K}$ has usually much fewer parameters than the activation matrix $H \in \mathbb{R}^{R\times N}$.)

Finally, we compare in Fig. 5 the training runtime of the multiplicative NAE and NAE with RMSprop. Although the multiplicative NAE shows a very steep decay within the first second, NAE with RMSprop trained on CPU and GPU both outperform the multiplicative model after around 100 seconds. Additionally, the gradient computation of the multiplicative update rules for NAEs becomes challenging for deeper networks. The implementation of NAE with RMSprop, on the other hand, exploit the automatic gradient computation. We also see that the GPU-accelerated model converges twice as fast as the NAE with RMSprop trained on CPU. The hardware acceleration becomes more evident in the case of deeper and more complex networks, which involve more matrix multiplications.

V. Conclusions

In this paper, we investigated different NAE-based approaches for decomposing piano recordings into musically meaningful spectral vectors. We simulated the reference score-informed NMF model with various NAE-based methods. We showed that NAEs acquire higher efficiency through hardware-accelerated frameworks while yielding similar results as the reference NMF model. We also explored different adaptive gradient technique methods, including multiplicative rules for NAEs. We showed that the GPU-accelerated, adaptive RMSprop method outperformed other NAE variants in terms of the approximation quality and efficiency, while the learned templates and activations remain similar to those of the NMF reference. In the future, we aim to develop deeper and more complex models, which result in faster and better convergence and preserve interpretability. This will enable the design of explainable deep learning models, as our constrained NAE, while improving the performance of the network.

References

[1] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proceedings of the Neural Information Processing Systems (NIPS)*, Denver, Colorado, USA, November 2000, pp. 556–562.

[2] N. Gillis, *Nonnegative Matrix Factorization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2020. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611976410

[3] P. Smaragdis and J. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proceedings of the IEEE Workshop on Applications of Signal Processing (WASPAA)*, 2003, pp. 177–180.

[4] N. Bertin, R. Badeau, and E. Vincent, "Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 538–549, 2010.

[5] R. Hennequin, R. Badeau, and B. David, "NMF with time–frequency activations to model nonstationary audio events," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 744–753, 2011.

[6] A. Lefevre, F. Bach, and C. Févotte, "Semi-supervised NMF with time–frequency annotations for single-channel source separation," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, 2012, pp. 115–120.

[7] J. Fritsch and M. D. Plumbley, "Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013, pp. 888–891.

[8] S. Ewert, B. Pardo, M. Müller, and M. Plumbley, "Score-informed source separation for musical audio recordings: An overview," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 116–124, April 2014.

[9] P. Smaragdis, C. Févotte, G. J. Mysore, N. Mohammadiha, and M. D. Hoffman, "Static and dynamic source separation using nonnegative factorizations: A unified view," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 66–75, 2014.

[10] C. Dittmar and M. Müller, "Reverse engineering the Amen break – score-informed separation and restoration applied to drum recordings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1531–1543, 2016.

[11] P. Smaragdis and S. Venkataramani, "A neural network alternative to non-negative audio models," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2017, pp. 86–90.

[12] S. Ewert and M. B. Sandler, "Structured dropout for weak label and multi-instance learning and its application to score-informed source separation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, Louisiana, USA, 2017, pp. 2277–2281.

[13] S. Ewert and M. Müller, "Using score-informed constraints for NMF-based source separation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, 2012, pp. 129–132.

[14] T. Tieleman and G. Hinton, "RmsProp: Divide the gradient by a running average of its recent magnitude," October 2012.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference for Learning Representations (ICLR)*, San Diego, California, USA, 2015.

[16] S. Raczynski, N. Ono, and S. Sagayama, "Multipitch analysis with harmonic nonnegative matrix approximation." in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Vienna, AT, 2007, pp. 381–386.

[17] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 982–994, 2007.

[18] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, 2006.

[19] P. Smaragdis and S. Venkataramani, "A neural network alternative to non-negative audio models," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, Louisiana, USA, 2017, pp. 86–90.

[20] C. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007. [Online]. Available: https://doi.org/10.1162/neco.2007.19.10.2756

[21] J. Chorowski and J. M. Zurada, "Learning understandable neural networks with nonnegative weight constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 62–69, 2015. [Online]. Available: https://doi.org/10.1109/TNNLS.2014.2310059

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, June 2014.

[23] T. Zunner, "Neural networks with nonnegativity constraints for decomposing music recordings," Master's thesis, Friedrich-Alexander-University of Erlangen-Nuremberg, 2021.