

QUANTIZE & FACTORIZE: A FAST YET EFFECTIVE UNSUPERVISED AUDIO REPRESENTATION WITHOUT DEEP LEARNING

Jaehun Kim

Matthew C. McCallum

Andreas F. Ehmann

SiriusXM+Pandora, USA

firstname.lastname@siriusxm.com

ABSTRACT

Foundation models have become increasingly prevalent in tackling Music Information Retrieval (MIR) tasks. Although they can be a powerful tool for understanding music, the computation required for the training and inference of these models continues to grow as they become more complex. Specialized acceleration, such as Graphical Processing Units (GPUs), has become necessary for operating these models, as they are mostly based on large Deep Learning (DL) architectures. Furthermore, it is difficult for users to interpret them due to their black-box nature. In this work, we propose Quantizers and Factorizers for Music embeddings (QFM), a fast, unsupervised audio representation for music understanding backed by a wide range of rich MIR features and efficient feature learners. Experimental results show that QFM models perform within the range of results achieved by recent previous open source DL models on all evaluated tasks, with competitive results on a subset. This is surprising given the significantly smaller computational requirements of QFM models for training and inference.

1. INTRODUCTION

Computing data representations through pre-trained foundation models is a common Machine Learning (ML) practice which is widely applied to the field of MIR [1]. These models essentially replace the role of conventional music audio features [2, 3]. Typically, DL models have been the core of the recent evolution of foundation models in MIR. Various works explored new architectures and learning objectives, leading to substantial improvement of foundation models recently [4, 5, 6, 7, 8]. However, non-DL representation learning in MIR is notably understudied, leaving only a handful of works [9, 10, 11] after DL methods became almost ubiquitous in ML practices. However, given increasing concerns about some weaknesses, such as lack of interpretability [12] or extensive energy consumption [13], it seems worthwhile to revisit non-DL alternatives for the foundation models, especially with regard to

computational efficiency for training and embedding computation, and possibly for better interpretability.

This work proposes a framework for learning unsupervised music representation as an efficient yet effective alternative to DL models. The primary goal of the framework is to address the computational efficiency of training and inference while maintaining comparable effectiveness. Specifically, we achieve these goals by employing established MIR features and encoding their information through efficient parallel dictionary learners, namely Quantization – Factorization (QF) modules. We conduct experiments similar to recent literature on music foundation models [8, 7, 6] to show the effectiveness of QFM embeddings. The results suggest that the method can achieve comparable effectiveness to some of recent DL models while being significantly faster in training and inference.

Our contributions in this work are as follows: 1) we introduce QFM, a low-resource foundation model useful for various MIR downstream tasks. 2) we provide QFM as a generic architecture in which feature engineering and machine learning practices are almost equally important, encouraging collaborations between wider audiences within the community. We stress that this work does *not* aim to show the state-of-the-art performance. Instead, we aim to provide an alternative, MIR feature-oriented ML model comparable to some DL-based models for effectiveness, while achieving notable computational efficiency.

2. RELATED WORK

Early works in unsupervised music representation learning include probabilistic component models such as the Gaussian Mixture Model (GMM) or its extension to infinite mixtures [14, 9]. Other popular methods are dictionary learning approaches such as sparse coding or vector quantization [15, 10, 16, 17]. Essentially, both approaches try to represent the music as a sparse response vector to the learned latent components.

Later, the field evolved to seek more high-capacity, nonlinear models. Deep Belief Networks (DBN) were explored as an unsupervised, nonlinear representation learner for spectral feature frames from short chunks of audio [18, 19]. Later, various types of Convolutional Neural Network (CNN) architectures were explored for representation learning in the transfer learning context [2, 20, 21, 22, 23, 24]. Another breakthrough was contrastive learning [25], which is one of the most popular and successful



© J. Kim, M. C. McCallum, A. F. Ehmann. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** J. Kim, M. C. McCallum, A. F. Ehmann, “Quantize & Factorize: A fast yet effective unsupervised audio representation without deep learning”, in *Proc. of the 26th Int. Society for Music Information Retrieval Conf.*, Daejeon, South Korea, 2025.

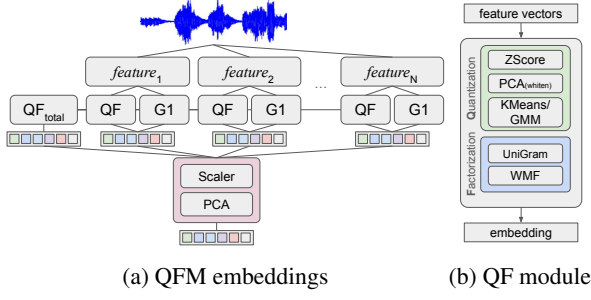


Figure 1: The overall architecture of QFM (a) and detailed illustration of submodules of each QF module (b).

learning objectives today for unsupervised representation learning [6, 7]. Recent developments in natural language modeling inspired new architectural approaches, such as transformers, and have also been reported to be effective in learning music representation [8, 5]. Other notable developments include semi-supervised approaches [26, 27], multi-modal learning approaches [28, 29, 4] and the use of music generative models as pre-trained foundation models [30, 31].

However, these high capacity models often require extensive data and computation for training and inference, raising concerns about excessive energy consumption [13]. In addition, the lack of interpretability of such models is considered one of the main drawbacks and extensively tackled within the MIR field [32, 33] and beyond [12].

3. METHODOLOGY

To address some of the aforementioned shortcomings of DL-based foundation models, we propose an unsupervised music representation learning framework that is highly efficient, yet effective. At the framework’s core are Quantization–Factorization (QF) modules and multi-aspect feature fusion. QF modules serve as dictionary learners for each *feature set*, comprising groups of MIR features. Figure 1 illustrates the diagrams of the general architecture of QFM (a) and the QF module (b). In this section, we describe each component of QFM in detail.

3.1 MIR Feature Sets

The features are the most critical building block of QFM, as we employ shallow feature learners. However, the rationale of QFM is that shallow learners provide sufficient representation if enough feature sets are provided. We use 5 sets of features in total, which are illustrated in Table 1.

The selected features are grouped into 5 categories according to the categorization following that of librosa [40]¹. From an input audio chunk, all features are computed and grouped, and then passed to the corresponding group-specific QF modules. For audio features, we set the sampling rate, hop size as 22kHz and 512, respectively. Other parameters vary for features, where we use the defaults specified in the librosa package. 2-D patches are

Group	Sub Features	Dim
MFCC	96 MFCC coefficients excluding the first	95
CQT	84-bin log Constant-Q power spectrum	84
Rhythmic*	Multi-band Onset Strength [†] (4), Tempogram Ratio [34, 35] (13), normalized BPM estimation (1)	18
	Spectral { Centroid [36] (1), Bandwidth [36] (1), Contrast [37] (7), Flatness [38] (1), Rolloff (1) }, Zero-crossing Rate (1), Tonnetz [39] (6)	
Patches	(9 × 9) patches from log mel spectrogram	81
Total		296

Table 1: List of feature groups and features belong to each group. The numbers in parenthesis mean the dimensionality of the feature. * We apply the log transform on Rhythmic feature group [†] Each 4 bands are divided by {16, 32, 64}-th bins among 96 mel bins

randomly sampled from nonoverlapping 9×9 tile grid on 96-bin log melspectrogram.

3.2 QF module

A QF module is composed of cascading submodules of quantization and factorization. The role of the quantization module is to map each input feature vector within the input audio chunk to a sequence of discrete codes. For instance, a set of feature vectors is computed from audio frames and then a quantizer, such as K-Means clustering, can assign each vector to the closest cluster centroid index. Such quantization has been a common method in dictionary learning [17, 10] and also recently in language-model-inspired audio models [8].

The factorization submodule is to encode the quantized code sequence into latent vectors using factorization models. In the aforementioned transformer models [8], transformer layers are chosen for this step. Instead, we take a simplified assumption of code exchangeability within an audio chunk [41], with which we wish to capture a sufficient amount of information while enjoying significant computational efficiency. We can then formulate a sparse unigram matrix whose row and column correspond to each audio chunk within the dataset and the codes (i.e., cluster index), respectively. Thus, each entry indicates how frequently each code is present for a given audio chunk after quantization. With such simplification, an efficient latent model such as Weighted Matrix Factorization (WMF) [42] can be applied to encode the resulting factors.

While the QF module is a generic framework suits many submodule options, we chose a specific configuration: 1) For quantization, we use a pipeline combining an optional standard scaler, then whitening Principal Component Analysis (PCA), followed by K-Means clustering or Gaussian Mixture Models (GMMs), which has been proven effective [17]. 2) The factorization includes the unigram representation [43] and WMF. The training of QF modules is conducted module-by-module sequentially, from the first submodule of the quantization to the last submodule of factorization. Once trained, the new embeddings can be efficiently inferred from the unigram code

¹ <https://librosa.org/doc/0.10.2/index.html>

frequency vectors of unseen feature vectors [42].

3.3 G1 and QF_{total}

Other feature encoders within QFM are G1 and QF_{total}. G1 refers to a single multivariate Gaussian, where we compute the mean and standard deviation of each feature set from an audio chunk. The concatenation of these Gaussian parameters is often used as a baseline feature in the literature [2, 9, 31]. However, we include it as a submodule of QFM to preserve the original raw characteristic of the features [17], which we lose during the QF embedding process as factorizers only observe quantized codes, not the raw features directly. Also, since QFs do not consider sequential dependency, the standard deviations of G1 provides a valuable summary of temporal variability within features. We compute the G1 and QF factors for each feature set we consider.

The QF_{total} is a separate factorization module that factorizes the total count matrix where per feature-set unigram count matrices are stacked horizontally. The process thus captures the interaction between feature sets, whereas independent QF modules are isolated to each feature set.

3.4 QFM Inference

QFM computes an embedding for a given audio chunk as follows; 1) For a given audio chunk, N feature sets are computed. 2) These values are passed to G1 modules, feature specific QF, and QF_{total} modules. Finally, 3) scaling and dimensionality reduction is performed using a “robust” scaling² and PCA on the concatenated total embedding.

4. EXPERIMENT

To verify the effectiveness of QFM, we conducted an experiment inspired by previous work. The experiment is designed to test a learned music audio representation on a range of downstream tasks, employing several publicly available datasets. We generally follow the high-level design and details from [7] yet simplify the dataset selection, adding a few additional analyses such as an ablation study.

4.1 QFMs

We compare the usefulness of representations: 1) within different configurations of QFMs, and 2) between selected open source foundation models established recently.

The details of the QFM configurations can be found in Table 2. We consider 5 different QFM configurations to examine the factors affecting performance and efficiency. To that end, the model ranges from small to large in terms of parametrization and volume of the training dataset. For example, the number of components K is set to 512 for the nano model, and 8192 for the large model, while we generally apply the same factor dimensionality D for factorizers per feature set. The final dimensionality of PCA

² the robust scaling method is non-parametric alternative to the standard scaling where we use the median and interquartile range instead of mean and standard deviation, which leads to a scaling more robust to outliers [44].

QFM/Models	Trainset	Quantizer	$W(s)$	K	D	D'
nano	FMA	GMM	9	512	128	762
micro	FMA	GMM	9	1024	256	1350
small	FMA	K-Means	9	4096	256	1266
medium	FMA	K-Means	9	8192	256	1219
large	MSD	K-Means	9	8192	256	1167
G1				9		592
CLMR	MTT		2.6			512
MusicFM	FMA	Random	29.1			1024
MULE	Musicset		3			1728

Table 2: Configurations of QFM models and comparing other foundation models. W , K , D , D' refer to the input audio length in seconds, the number of component/cluster for quantizers, dimensionality of the WMF factors, and the final dimensionality of resulting embeddings.

D' is automatically determined by the desired explained variance ratio of the PCA, set to 97%. We use diagonal covariance GMMs for the smaller models (micro, nano) instead of K-Means since they are designed to test faster inference by setting the number of components K small. As GMMs tend to better fit the data due to increased flexibility, we use them to compensate for the performance trade-off due to the small model size.

For each feature set, we apply slightly different configurations for QF modules for better performance; 1) we generally apply the standard scaling as an initial step of quantization, except the CQT, where we assume that the average magnitude of specific notes or octaves can be useful information. 2) We set a different hyperparameter for each PCA within each quantizer where we set 95%, 94%, and 99% of the explained variance ratio for MFCC, CQT, and Patches feature groups, respectively. In contrast, we use the automatic method [45] to find the optimal dimensionality for the Rhythmic and Spectral features where the number of features is smaller. Finally, while we generally apply the same dimensionality reported in Table 2 for WMF, we halve the number for the Rhythmic features. This is due to the observation that the number of unique codes with nonzero frequency in this feature group is much less for each audio chunk, meaning the code count matrix is sparser. This is primarily because the tempogram ratio feature is generally stable over the given audio chunk. Another exception is QF_{total}, where we double the dimensionality D as it factorizes concatenated, wide matrices that include all features.

Finally, there are some hyperparameters that are applied globally: we set 5 and 0.5 for the weighting and regularization coefficients of WMF, and we filter components (columns) in a unigram matrix if they present less than 0.1% of songs within the training set. Also, the length of the audio chunk during training time is set to 9 seconds due to the default setup of the tempogram ratio features. The embeddings can still be computed with shorter audio chunks at inference time, with a potential loss of certain information in the rhythmic features.

4.2 Baselines and Other Foundation Models

We compare QFMs to audio foundation models from the literature, specifically within the scope of unsupervised

unimodal audio representations.

G1 is the concatenation of the G1 features described in Section 3.3, followed by the robust scaling. We include all 5 feature sets in Table 1, which result in a total dimensionality of 592. This serves as the baseline for all comparisons. We take 9 second audio chunks with a 3 second overlap as the standard input following QFM models.

CLMR [6] is a 1D CNN model, which shows the effectiveness of the contrastive loss [25] on the unsupervised music representation when combined with data augmentation. We use non-overlapping 2.6 second audio chunks as input following the original work [6].

MULE [7] is another unsupervised audio foundation model trained with a similar objective. The main differences between CLMR are the training dataset and the architecture. MULE employs 2D CNN based on [46] instead of 1D CNN and is trained on Musicset, which is a proprietary dataset of only music audio unlike datasets like Audioset [47, 7]. It also applies advanced sampling and data augmentation strategies. The work took the global average pooling on timeline embeddings produced from overlapping 3 second audio chunks, used later for downstream tasks. As an exception, we reprint the results of MULE from its original work instead of reproducing them.

MusicFM [8] is a transformer-based foundation model employing masked token modeling, inspired by language models. We chose a version trained on the FMA-large dataset, which is the same dataset used for training QFM models and is publicly available. We use non-overlapping 29.1s window lengths for input audio chunking ³.

4.3 Datasets

4.3.1 Training Set

For training QFM models, we employ the audio data from the FMA dataset [48] and the Million Song Dataset (MSD) [49]. In particular, we use the ‘FMA-large’ version, which contains 106,574 30-second song segments for all QFM variants introduced in Table 2, except the `large` model, where we employ the entire one million song previews from MSD. For efficient training, we sample 3 9-second audio chunks from each audio file and use these samples as observations. The per-feature vector models within each quantizer module (i.e., PCA, K-Means) are trained on further subsampled vectors from these chunks, where we sample 60 vectors per chunk, except the `large` model, where we sample 200 vectors per chunk instead.

4.3.2 Downstream Datasets

For downstream datasets, we generally follow [7], selecting a subset. The overall description of the downstream datasets can be found in Table 3.

GS_{key} dataset is the 24-way key-scale classification dataset introduced in [50]. Following previous work [31, 7], we use the second version of the dataset ⁴ for training

Dataset	Task	Probe	#Items	#Labels	Avg. Secs	Hrs
MTT	tagging	MLP	26k	50	29	209
NSynth _p	pitch	LR	306k	112	4	340
NSynth _i	instrument	LR	306k	11	4	340
GTZAN	genre	LR	930	10	30	7.8
Emo	emotion	MLP	744	N/A	45	9.3
Jam-MT	mood/theme	MLP	18k	56	219	1.1k

Table 3: Details of downstream task datasets. The probe setup is following previous works [7,31]. MLPs have a single hidden layer with 512 units except on the Emo dataset which has 1024 units. LR refers to Logistic Regression.

and validation, and the first version is used for testing. We apply the weighted accuracy to measure performance ⁵.

MTT is a popular music tagging dataset [51]. Following [7] we use the 50 most frequent tags with the commonly used data split from [52]. The approach leaves a small subset of songs not annotated with any tags, which we still keep to better compare with the previous works. The effectiveness of this task is measured by AUC-ROC (AUC) [53] and mean Average Precision (mAP).

NSynth contains audio recordings of various instrumental sounds generated from commercial audio sample libraries [54]. Two main tasks are pitch detection and instrument classification. We take both tasks and use the official split from the dataset. For both datasets, we measure effectiveness by classification accuracy.

GTZAN is a dataset for genre classification [55]. We apply the fault-filtered split developed in [56,57] ⁶. Similarly to NSynth tasks, we measure accuracy.

Emo Music dataset provides time-continuous annotation of coordinates in the Arousal-Valence (AV) space, collected by multiple annotators for total 744 songs [58]. We simplify the problem to a per-song multiregression where we take the average value of each arousal and valence dimension, across annotators and over the timeline, as the global coordinate estimates of the song in the AV space. We use the artist-based split in [31], and the coefficient of determination (R^2) is measured separately on arousal (R_a^2) and valence (R_v^2) values, to assess performance.

Jam-MT is a part of the MTG-Jamendo dataset specialized in music mood tagging [59]. We employ the official split provided, using full 56 tags. Similarly to MTT, we use AUC and mAP metrics for this dataset.

4.4 Other Experimental Details

We employed a simplified model selection protocol for the main downstream task experiments while generally following [7, 31]. We choose the best model type for each task found by [7] and run a hyperparameter tuning similar to [31] within each model setup per dataset. For example, we use a Multi-Layer Perceptron (MLP) classifier for the MTT dataset with a single hidden layer of dimensionality 512. At the same time, the optimal hyperparameter for other factors such as learning rate, regularization, and

³ <https://github.com/minzwon/musicfm>

⁴ <https://github.com/GiantSteps/giantsteps-key-dataset>

⁵ https://www.music-ir.org/mirex/wiki/2021:Audio_Key_Detection

⁶ https://github.com/jongpillee/music_dataset_split

standardization, are determined according to best performance on the validation set, for QFM and each of the other foundation models with which we compare.

For all QFM models and other foundation models, we apply a multiple-instance learning approach [60] for downstream tasks when the audio clip length exceeds their recommended input window length [46], except when the audio input is shorter than that (i.e., NSynth dataset). The probability estimates⁷ of trained probe models are averaged over the timeline to determine the final prediction. We apply this score-pooling approach to all downstream tasks. All QFMs use a 9s window with 3s overlap, and other models use the window and overlap size described in Section 4.2. Further, we run 5 runs considering the randomness of MLP training and report the distribution through boxplot visualization. In contrast, linear models report no variability in the final result.

We implement QFM through ML modules provided by scikit-learn [61] and employ librosa [40] for feature computation. Finally, we use `implicit`⁸ for WMF implementation with a simple custom scikit-learn wrapper. As for hardware, for all computations, including training, embedding extraction, and experiments, we employ a Virtual Machine (VM) with 32 virtual CPU (vCPU) cores and 117 GB memory without any accelerators such as GPUs, where each vCPU core runs a single thread. We run the experiment on Ubuntu24.04 Linux system, with Python 3.10.

5. RESULT AND DISCUSSION

5.1 Effectiveness

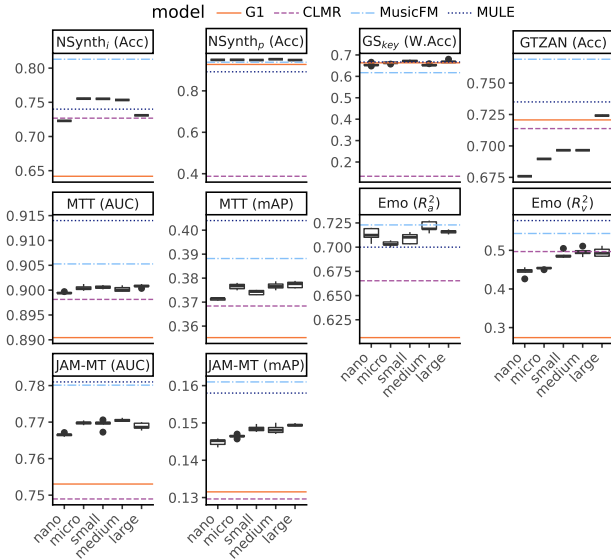


Figure 2: Performance on test datasets. Boxplots in each pane indicate the performance distribution of QFM variants, while horizontal lines refer the mean performance of baseline foundation models. We indicate the metrics used for each dataset in the box on top of each pane.

⁷ logits are used for regression problems.

⁸ <https://github.com/benfred/implicit>

QFM/Models	Avg. Perf.	H_{MTT} (Hrs)
G1	0.5964	0.3416 (± 0.0024)
nano	0.6343	0.4561 (± 0.0050)
large	0.6480	0.7008 (± 0.0081)
CLMR	0.5269	2.7503 (± 0.0010)
MusicFM	0.6638	8.8634 (± 0.1744)

Table 4: Overall performance and efficiency of selected foundation models measured on MTT dataset. ‘Avg. Perf.’ refers the averaged performance measure across all tasks considered.

We first examine the overall effectiveness of QFM in downstream tasks. As depicted in Figure 2 and Table 4, the QFM embeddings are more effective than the CLMR and G1 embeddings, although generally worse than the more recent DL models MULE and MusicFM. In particular, there are a few tasks in which some QFMs are on par with or outperform MULE and MusicFM, such as GS_{key} , $NSynth_p$, and Emo (Arousal). On the other hand, QFM shows weakness especially in tagging tasks such as MTT and Jam-MT, while they might still outperform some DL models like CLMR. The average result suggests that QFMs are generally comparable to DL representations, substantially better than CLMR and a few points behind MULE and MusicFM, as shown in Table 4 and Figure 2.

Within QFM, the overall effectiveness roughly positively correlates with the ‘size’ of the QFM configuration. Some exceptions are the cases where all QFM models show similarly good performance, such as GS_{key} and Emo (Arousal) or $NSynth_i$ where *large* perform relatively worse than smaller models.

5.2 Efficiency

We evaluate the efficiency of QFM by measuring the embedding computation time and compare with other foundation models⁹. We measure inference time over MTT dataset consisting of 25,860 29.1 seconds preview audio clips, equivalent to approximately 209 audio hours. We measure H_{MTT} , the time spent in hours computing the embeddings of entire audio clips within MTT, using each model in the computing environment described in Section 4.4. We report the average and standard deviation after running 5 runs.

The result suggests that the QFMs are significantly faster than both DL foundation models. In particular, QFM *large* and *nano* outperform MusicFM by more than 12 and nearly 20 times. Compared to the relatively lightweight CLMR, they are faster by approximately 4 and 6 times each. Compared to G1, QFMs are slightly more expensive, suggesting that the majority of computation time is feature computation. Although we do not formally compare training time, the training cost of QFMs is also substantially lower than DL models; the training time of the medium model is about 3 hours on a 32-core virtual machine without GPU, which is significantly faster than those of DL foundation models¹⁰.

⁹ We exclude MULE as we do not reproduce the result, and we only choose the largest and smallest QFM configurations for simplicity.

¹⁰ *large* model’s training time is about 30 hours on the same 32-core

The efficient computation of QFM provides several benefits: 1) It can lead to a faster development cycle. Due to the modular nature of QF modules for each feature set, it allows one to experiment with various features and model configurations thereof, which can translate into better performance. 2) one can compute QFMs at scale with less computational resources. Given that QFMs offer reasonable music understanding and sometimes even outperform the most significant DL models in some downstream tasks, the ease of computation can be appealing for employing QFMs as either a complimentary to DL models, or as a main embedding when computation requirements dictate.

5.3 Ablation Study

5.3.1 Feature Ablation

We conducted two ablation studies to understand the contribution of some components to QFM model performance. In the first study, we focus on the effect of feature groups by comparing QFMs with different subsets of feature groups. We chose *large* variant as the reference QFM model, with a full set of features. We compare 5 QFM variants in which only a subset of feature sets is present. For simplicity, these models include an increasing number of feature sets in the order of feature sets presented in Table 1. We refer to them based on the initial of included feature sets. (i.e., the ‘M’ model only includes MFCC, and ‘MCR’ includes MFCC, CQT, and Rhythmic feature sets). For these feature-subset models, we exclude final PCA and QF_{total} . Hence, the dimensionality of subset embeddings can be larger or smaller than the *large* model. Experimental results can be found in Fig 3 (a).

In general, the addition of each feature set significantly improves the overall effectiveness of QFMs in all downstream tasks. Especially the first three feature sets – MFCC, CQT, and Rhythmic – bring notable improvements compared to Spectral Feature and Patches. It may indicate that the Spectral and Patches features set do not provide much additional information than the first three feature sets. In some tasks, such as GTZAN and GS_{key} , the MCR variant outperforms the version with more feature sets. However, other tasks such as Emo and tagging datasets benefit from complete sets of features.

5.3.2 Model Configuration

We compare three variants to examine the effect of model sub-components on model performance: G1 model, and *medium* model, and finally *medium* model with randomized SVD [62] instead of WMF with the same final dimensionality. As shown in Fig 3 (b), the *medium* model outperforms both the G1 and SVD variants, suggesting: 1) QF modules add substantially more information on top of G1, furthermore, 2) given that the use of SVD substantially underperforms when compared to the WMF (and even worse than the G1 model) WMF is crucial for the effectiveness of QFMs.

VM we described earlier, as we employed MSD which is about 10 times larger than the FMA-large dataset.

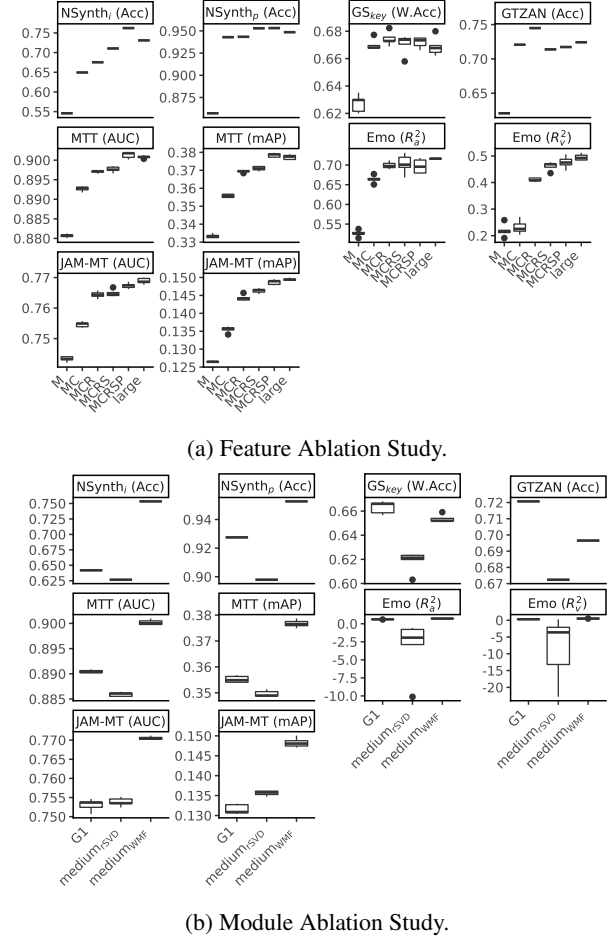


Figure 3: Ablation study result.

6. CONCLUSION AND FUTURE WORKS

We propose QFM as a non-DL foundation model empowered by the ensemble of rich MIR features utilizing efficient shallow feature learners. The experimental result shows that it performs comparable to some recent DL models while being significantly more efficient, implying that QFM models can be complimentary to DL audio representations in industrial scale applications. Furthermore, QFMs provide an alternative where the cost of DL model inference is prohibited.

Although with these benefits, we observe weaknesses, especially in some downstream tasks compared to the recent DL-based foundation models. Research towards additional features or their manipulations with QF modules may further improve the overall effectiveness, as shown in the results. Given that there are many music/audio features that have not been explored and further uncharted new features that may be introduced in the future, QFM can be an alternative platform that the MIR community can contribute to, to improve music representations.

Another potential benefit is that the interpretable linear and clustering operators in QFM models open up alternative avenues to interpreting music audio representations, which can be studied further in a scope beyond that of the current work.

7. REFERENCES

- [1] Y. Ma, A. Øland, A. Ragni, B. M. D. Sette, C. Saitis, C. Donahue, C. Lin, C. Plachouras, E. Benetos, E. Quinton, E. Shatri, F. Morreale, G. Zhang, G. Fazekas, G. Xia, H. Zhang, I. Manco, J. Huang, J. Guinot, L. Lin, L. Marinelli, M. W. Y. Lam, M. Sharma, Q. Kong, R. B. Dannenberg, R. Yuan, S. Wu, S. Wu, S. Dai, S. Lei, S. Kang, S. Dixon, W. Chen, W. Huang, X. Du, X. Qu, X. Tan, Y. Li, Z. Tian, Z. Wu, Z. Wu, Z. Ma, and Z. Wang, “Foundation models for music: A survey,” *CoRR*, vol. abs/2408.14340, 2024.
- [2] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, “Transfer learning for music classification and regression tasks,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull, Eds., 2017, pp. 141–149.
- [3] E. J. Humphrey, J. P. Bello, and Y. LeCun, “Moving beyond feature design: Deep architectures and automatic feature learning in music informatics,” in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, F. Gouyon, P. Herrera, L. G. Martins, and M. Müller, Eds. FEUP Edições, 2012, pp. 403–408.
- [4] Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, and S. Dubnov, “Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*. IEEE, 2023, pp. 1–5.
- [5] Y. Li, R. Yuan, G. Zhang, Y. Ma, X. Chen, H. Yin, C. Xiao, C. Lin, A. Ragni, E. Benetos, N. Gyenge, R. B. Dannenberg, R. Liu, W. Chen, G. Xia, Y. Shi, W. Huang, Z. Wang, Y. Guo, and J. Fu, “MERT: acoustic music understanding model with large-scale self-supervised training,” in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.
- [6] J. Spijkervet and J. A. Burgoyne, “Contrastive learning of musical representations,” in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, J. H. Lee, A. Lerch, Z. Duan, J. Nam, P. Rao, P. van Kranenburg, and A. Srinivasamurthy, Eds., 2021, pp. 673–681.
- [7] M. C. McCallum, F. Korzeniowski, S. Oramas, F. Gouyon, and A. F. Ehmann, “Supervised and unsupervised learning of audio representations for music understanding,” in *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022, Bengaluru, India, December 4-8, 2022*, P. Rao, H. A. Murthy, A. Srinivasamurthy, R. M. Bittner, R. C. Repetto, M. Goto, X. Serra, and M. Miron, Eds., 2022, pp. 256–263.
- [8] M. Won, Y. Hung, and D. Le, “A foundation model for music informatics,” in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*. IEEE, 2024, pp. 1226–1230.
- [9] J. Kim and C. C. S. Liem, “The power of deep without going deep? A study of HDPGMM music representation learning,” in *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022, Bengaluru, India, December 4-8, 2022*, P. Rao, H. A. Murthy, A. Srinivasamurthy, R. M. Bittner, R. C. Repetto, M. Goto, X. Serra, and M. Miron, Eds., 2022, pp. 116–124.
- [10] Y. Vaizman, B. McFee, and G. R. G. Lanckriet, “Codebook-based audio feature representation for music information retrieval,” *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 22, no. 10, pp. 1483–1493, 2014.
- [11] H. Eghbal-zadeh, B. Lehner, M. Schedl, and G. Widmer, “I-vectors for timbre-based music similarity and music artist classification,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, M. Müller and F. Wiering, Eds., 2015, pp. 554–560.
- [12] C. Molnar, *Interpretable machine learning*. Lulu.com, 2020.
- [13] A. Holzapfel, A. Kaila, and P. Jäskeläinen, “Green mir? investigating computational cost of recent music-ai research in ISMIR,” in *Proceedings of the 25th International Society for Music Information Retrieval Conference, ISMIR 2024, San Francisco, California, USA and Online, November 10-14, 2024*, B. Kaneshiro, G. J. Mysore, O. Nieto, C. Donahue, C. A. Huang, J. H. Lee, B. McFee, and M. C. McCallum, Eds., 2024, pp. 371–380.
- [14] M. D. Hoffman, D. M. Blei, and P. R. Cook, “Content-based musical similarity computation using the hierarchical dirichlet process,” in *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, J. P. Bello, E. Chew, and D. Turnbull, Eds., 2008, pp. 349–354.
- [15] B. McFee, L. Barrington, and G. R. G. Lanckriet, “Learning content similarity for music recommendation,” *IEEE Trans. Speech Audio Process.*, vol. 20, no. 8, pp. 2207–2218, 2012.

- [16] J. Nam, J. Herrera, M. Slaney, and J. O. S. III, "Learning sparse feature representations for music annotation and retrieval," in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S. Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, F. Gouyon, P. Herrera, L. G. Martins, and M. Müller, Eds., FEUP Edições, 2012, pp. 565–570.
- [17] S. Dieleman and B. Schrauwen, "Multiscale approaches to music audio feature learning," in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, A. de Souza Britto Jr., F. Gouyon, and S. Dixon, Eds., 2013, pp. 3–8.
- [18] J. Nam, J. Herrera, and K. Lee, "A deep bag-of-features model for music auto-tagging," *CoRR*, vol. abs/1508.04999, 2015.
- [19] H. Lee, P. T. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds., Curran Associates, Inc., 2009, pp. 1096–1104.
- [20] J. Kim, J. Urbano, C. C. S. Liem, and A. Hanjalic, "One deep music representation to rule them all? A comparative analysis of different representation learning strategies," *Neural Comput. Appl.*, vol. 32, no. 4, pp. 1067–1093, 2020.
- [21] J. Lee, J. Park, K. L. Kim, and J. Nam, "Samplecnn: End-to-end deep convolutional neural networks using very small filters for music classification," *Applied Sciences*, vol. 8, no. 1, 2018.
- [22] J. Pons, T. Lidy, and X. Serra, "Experimenting with musically motivated convolutional neural networks," in *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2016, pp. 1–6.
- [23] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. W. Wilson, "CNN architectures for large-scale audio classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*. IEEE, 2017, pp. 131–135.
- [24] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 609–617.
- [25] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. *Proceedings of Machine Learning Research*, vol. 119. PMLR, 2020, pp. 1597–1607.
- [26] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, J. H. Lee, A. Lerch, Z. Duan, J. Nam, P. Rao, P. van Kranenburg, and A. Srinivasamurthy, Eds., 2021, pp. 769–776.
- [27] J. Guinot, E. Quinton, and G. Fazekas, "Semi-supervised contrastive learning of musical representations," in *Proceedings of the 25th International Society for Music Information Retrieval Conference, ISMIR 2024, San Francisco, California, USA and Online, November 10-14, 2024*, B. Kaneshiro, G. J. Mysore, O. Nieto, C. Donahue, C. A. Huang, J. H. Lee, B. McFee, and M. C. McCallum, Eds., 2024, pp. 571–579.
- [28] A. Ferraro, J. Kim, S. Oramas, A. F. Ehmann, and F. Gouyon, "Contrastive learning for cross-modal artist retrieval," in *Proceedings of the 24th International Society for Music Information Retrieval Conference, ISMIR 2023, Milan, Italy, November 5-9, 2023*, A. Sarti, F. Antonacci, M. Sandler, P. Bestagini, S. Dixon, B. Liang, G. Richard, and J. Pauwels, Eds., 2023, pp. 375–382.
- [29] J. Cramer, H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*. IEEE, 2019, pp. 3852–3856.
- [30] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *CoRR*, vol. abs/2005.00341, 2020.
- [31] R. Castellon, C. Donahue, and P. Liang, "Codified audio language modeling learns useful representations for music information retrieval," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, J. H. Lee, A. Lerch, Z. Duan, J. Nam, P. Rao, P. van Kranenburg, and A. Srinivasamurthy, Eds., 2021, pp. 88–96.
- [32] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis," in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull, Eds., 2017, pp. 537–543.

- [33] —, “Understanding a deep machine listening model through feature inversion,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, E. Gómez, X. Hu, E. Humphrey, and E. Benetos, Eds., 2018, pp. 755–762.
- [34] M. Prockup, A. F. Ehmann, F. Gouyon, E. M. Schmidt, and Y. E. Kim, “Modeling musical rhythmat-scale with the music genome project,” in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2015, pp. 1–5.
- [35] G. Peeters, “Rhythm classification using spectral rhythm patterns,” in *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11-15 September 2005, Proceedings*, 2005, pp. 644–647.
- [36] D. FitzGerald and J. Paulus, *Unpitched Percussion Transcription*. Boston, MA: Springer US, 2006, pp. 131–162.
- [37] D. Jiang, L. Lu, H. Zhang, J. Tao, and L. Cai, “Music type classification by spectral contrast feature,” in *Proceedings of the 2002 IEEE International Conference on Multimedia and Expo, ICME 2002, Lausanne, Switzerland, August 26-29, 2002. Volume I*. IEEE Computer Society, 2002, pp. 113–116.
- [38] S. Dubnov, “Generalization of spectral flatness measure for non-gaussian linear processes,” *IEEE Signal Process. Lett.*, vol. 11, no. 8, pp. 698–701, 2004.
- [39] C. Harte, M. Sandler, and M. Gasser, “Detecting harmonic change in musical audio,” in *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, ser. AMCMM ’06. New York, NY, USA: Association for Computing Machinery, 2006, p. 21–26.
- [40] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in python,” in *Proceedings of the 14th Python in Science Conference 2015 (SciPy 2015), Austin, Texas, July 6 - 12, 2015*, K. Huff and J. Bergstra, Eds. scipy.org, 2015, pp. 18–24.
- [41] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” in *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2001, pp. 601–608.
- [42] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 2008, pp. 263–272.
- [43] Z. S. Harris, “Distributional structure,” *WORD*, vol. 10, no. 2-3, pp. 146–162, 1954.
- [44] H. Kaltenbach, *A Concise Guide to Statistics*, ser. SpringerBriefs in Statistics. Springer Berlin Heidelberg, 2011.
- [45] T. P. Minka, “Automatic choice of dimensionality for PCA,” in *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds. MIT Press, 2000, pp. 598–604.
- [46] L. Wang, P. Luc, Y. Wu, A. Recasens, L. Smaira, A. Brock, A. Jaegle, J. Alayrac, S. Dieleman, J. Carreira, and A. van den Oord, “Towards learning universal audio representations,” in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*. IEEE, 2022, pp. 4593–4597.
- [47] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*. IEEE, 2017, pp. 776–780.
- [48] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, “FMA: A dataset for music analysis,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull, Eds., 2017, pp. 316–323.
- [49] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, A. Klapuri and C. Leider, Eds. University of Miami, 2011, pp. 591–596.
- [50] P. Knees, Á. Faraldo, P. Herrera, R. Vogl, S. Böck, F. Hörschläger, and M. L. Goff, “Two data sets for tempo estimation and key detection in electronic dance music annotated from user corrections,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, M. Müller and F. Wiering, Eds., 2015, pp. 364–370.
- [51] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, “Evaluation of algorithms using games: The case of music tagging,” in *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*, K. Hirata, G. Tzanetakis, and K. Yoshii, Eds. International Society for Music Information Retrieval, 2009, pp. 387–392.

- [52] J. Pons and X. Serra, “musicnn: Pre-trained convolutional neural networks for music audio tagging,” *CoRR*, vol. abs/1909.06654, 2019.
- [53] A. P. Bradley, “The use of the area under the roc curve in the evaluation of machine learning algorithms,” *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.
- [54] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi, “Neural audio synthesis of musical notes with wavenet autoencoders,” 2017.
- [55] G. Tzanetakis, “Automatic musical genre classification of audio signals,” in *ISMIR 2001, 2nd International Symposium on Music Information Retrieval, Indiana University, Bloomington, Indiana, USA, October 15-17, 2001, Proceedings*, 2001.
- [56] B. L. Sturm, “The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use,” *CoRR*, vol. abs/1306.1461, 2013.
- [57] C. Kereliuk, B. L. Sturm, and J. Larsen, “Deep learning and music adversaries,” *IEEE Trans. Multim.*, vol. 17, no. 11, pp. 2059–2071, 2015.
- [58] M. Soleymani, M. N. Caro, E. M. Schmidt, C. Sha, and Y. Yang, “1000 songs for emotional analysis of music,” in *Proceedings of the 2nd ACM international workshop on Crowdsourcing for multimedia, CrowdMM@ACM Multimedia 2013, Barcelona, Spain, October 22, 2013*, K. Chen, W. Chu, and M. A. Larson, Eds. ACM, 2013, pp. 1–6.
- [59] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, and X. Serra, “The mtg-jamendo dataset for automatic music tagging,” in *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States, 2019.
- [60] J. D. Keeler, D. E. Rumelhart, and W. K. Leow, “Integrated segmentation and recognition of hand-printed numerals,” in *Advances in Neural Information Processing Systems 3, [NIPS Conference, Denver, Colorado, USA, November 26-29, 1990]*, R. Lippmann, J. E. Moody, and D. S. Touretzky, Eds. Morgan Kaufmann, 1990, pp. 557–563.
- [61] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [62] N. Halko, P. Martinsson, and J. A. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions,” *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.