

MusicContainerNet: Learning Representations of Musical Artists Using Aggregation-Based Networks

Timothy Greer¹, Wo Jae Lee^{2,†}, Aileen Mi¹, Emanuele Coviello¹ and Amina Shabbeer¹

¹Amazon Music

²Google

Abstract

Music recommendations rely on robust artist embeddings that capture stylistic and behavioral similarity. A common approach is to learn track-level embeddings from customer co-listening data and aggregate them per artist, but simple heuristic aggregation (e.g., averaging) treats all tracks equally and may fail to capture which tracks best define an artist’s identity. We introduce MusicContainerNet (MCN), a lightweight self-attention model that learns to aggregate an artist’s track embeddings—capped at 20 tracks—into a single artist representation. Inspired by the CLS token mechanism in BERT, MCN attends over the full track set and produces a fixed-length output that accounts for inter-track relationships. The architecture is intentionally constrained for production deployment: with at most 20 input tracks, MCN processes approximately 5,000 artists per second on a single GPU and meets Amazon Music’s strict latency budget ($p_{99} < 10$ ms). Offline, MCN variants substantially outperform averaging and graph-based baselines on retrieval and tagging tasks, achieving up to 15.2% relative mAP improvement on cross-entity evaluations. In a two-week live A/B test at Amazon Music, MCN drove +4 basis points in Play Days, +8 basis points in Listening Hours, and +11 basis points in Long-Term Engagement Retention relative to the production baseline, validating that improvements in embedding quality translate to meaningful real-world impact.

Keywords

aggregation networks, attention models, representation learning

1. Introduction

Digital music platforms have transformed how listeners engage with music, and embeddings have become a core tool for capturing musical preferences compactly. Representing artists in an embedding space enables recommendations of similar artists based on customer listening history. A typical approach learns track-level embeddings from co-listening graphs and aggregates them per artist. However, simple aggregation (e.g., averaging) treats all tracks equally and may fail to capture which tracks best represent an artist’s style [1, 2]. Alternatively, learning artist embeddings directly from co-listening data may not align well with track-level signals. Prior work has explored heuristic aggregation for cross-entity similarities [3, 4, 5, 6], but these rules are typically determined manually and refined heuristically [7, 8].

We propose MusicContainerNet (MCN), an aggregation model that learns to combine an artist’s track embeddings into a single vector via a self-attention mechanism with a CLS token. MCN is designed for production at Amazon Music: inputs are capped at 20 tracks, keeping the quadratic attention cost negligible ($\sim 5,000$ artists/s on one GPU, $p_{99} < 10$ ms). These design choices ensure the model meets strict latency constraints required by Amazon Music’s online systems while still substantially outperforming simple baselines.

We validate MCN both offline—with up to 15.2% relative mAP improvement on cross-entity retrieval tasks—and in a live A/B test (+4 bps Play Days, +8 bps Listening Hours, +11 bps Long-Term Engagement Retention).

MuRS: Music Recommender Systems Workshop 2026, September 28, 2026, Minneapolis, MN

[†]Work was done while at Amazon Music.

✉ greertim@amazon.com (T. Greer)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

2. Related Work

Prior work on music embeddings spans several approaches. Early methods relied on audio content or metadata features for artist similarity [9, 10], while more recent methods use graph neural network approaches on user consumption data [11]. Track embeddings trained on co-listening graphs have been used as building blocks in music recommendation systems. Aggregating track embeddings to represent higher-level containers such as artists, albums, or playlists has been studied via simple heuristics or graph neural networks, but there is relatively little prior work on *learning* artist-level embeddings from tracks.

Attention mechanisms have proven effective in music analysis tasks. Won et al. [12] introduced attention-based neural networks for music tagging, demonstrating improved performance over non-attention models. Liu et al. [13] employed self-attention for tail-entity classification in genre tasks. The bag-of-tracks concept has been applied with notable success: McFee et al. [5] represented playlist containers with features from constituent artists, and Ma et al. [14] used multi-instance learning and film cues to classify genres. To our knowledge, no prior work uses learned attention-based aggregation of an artist’s own tracks to produce artist embeddings—the novel contribution of this work.

3. Proposed Method

MusicContainerNet (MCN) learns artist embeddings by aggregating track-level embeddings via self-attention with a CLS token, trained on co-listening data using a contrastive objective.

3.1. Track-Level Music Embedding

Before building artist embeddings, we obtain track embeddings from customer consumption data. We use one year of Amazon Music listening logs to construct co-listening graphs. Specifically, if two tracks are frequently played in the same customer session, an edge is added between them. We compute a similarity score for each track pair based on co-listen frequency [15], retaining up to the top-1000 edges per track. To mitigate popularity biases and hub nodes [16], we apply hub-reduction via mutual k-nearest neighbor filtering, followed by multi-hop edge augmentation for low-degree nodes to ensure connectivity.

To learn track-level embeddings v_i , we define the affinity between two tracks as the dot product of their embeddings. We optimize a weighted cross-entropy of the form $\sum_{i,j} s_{i,j} \log(\text{softmax}_j(\exp(v_i'v_j)))$ using negative sampling [17], where $s_{i,j}$ is an increasing function of the similarity strength of the pair (i, j) . The output is a 128-dimensional vector where two track embeddings are close in space if they have behavioral similarity: they are likely to co-occur in a listening session. This follows standard item2vec-style approaches [18].

Track embeddings are held fixed (frozen) during MCN training. This ensures that MCN learns to aggregate within the existing track embedding space without distorting it, preserving compatibility with other downstream systems that consume the same track representations in production.

3.2. MusicContainerNet (MCN) Architecture

We present a high-level overview of the proposed model in Fig. 1. A self-attention-based aggregation model combines variable-length sets of track embeddings into a single-vector representation. A CLS token—similar to BERT [19]—allows the entire input sequence to be encoded into a fixed-size representation by aggregating information across all tracks, capturing global context and relationships among individual tracks from an artist’s catalog.

Variable-length track sets are handled by random subsampling (if $|\text{set}| > k$) or zero-padding with attention masking (if $|\text{set}| < k$), ensuring uniform batch dimensions. After applying sampling and padding, we prepend a learnable parameter $\mathbf{h}_{\text{CLS}} \in \mathbb{R}^d$ to a Bag-of-Tracks (BoT). Given padded

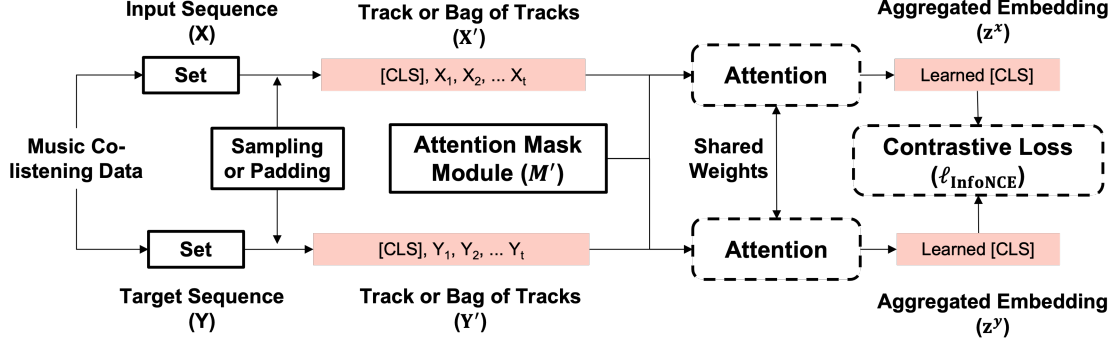


Figure 1: An overview of our proposed contrastive learning approach which uses attention-based aggregation with a CLS token to generate container embeddings. These embeddings are optimized via a contrastive loss informed by music co-listening data, where songs frequently listened to together are treated as positive pairs—an input sequence and its corresponding target sequence—and in-batch negative sampling is used to collect negative examples.

embeddings $\tilde{\mathbf{X}} \in \mathbb{R}^{B \times T \times d}$ and an attention mask $\mathbf{M}_X \in \{0, 1\}^{B \times T}$, we prepend $\mathbf{h}_{\text{CLS}}$ to each sequence as follows:

$$\mathbf{X}' = [\mathbf{h}_{\text{CLS}}; \tilde{\mathbf{X}}] \in \mathbb{R}^{B \times (T+1) \times d}, \quad \mathbf{M}'_X = [1; \mathbf{M}_X] \in \{0, 1\}^{B \times (T+1)}.$$

Then, the core operations for a given $\mathbf{X}'$ are:

$$\mathbf{Q} = \text{MLP}_Q(\mathbf{X}'), \quad \mathbf{K} = \text{MLP}_K(\mathbf{X}'), \quad \mathbf{S} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}},$$

$$\mathbf{A} = \text{softmax}(\mathbf{S} \odot \mathbf{M}'_X), \quad \mathbf{H} = \mathbf{A}\mathbf{X}', \quad \mathbf{z}^{x_i} = \mathbf{H}_{i,0}$$

where MLP_Q and MLP_K are feed-forward networks with two hidden layers to obtain queries $\mathbf{Q}$ and keys $\mathbf{K}$. $\mathbf{S}$ denotes the scaled dot-product attention scores. $\mathbf{M}'_X$ is the attention mask used to prevent attending to padding tokens. $\mathbf{A}$ is the normalized attention weights. $\mathbf{H}$ refers to the updated representations obtained by multiplying the attention weight matrix $\mathbf{A}$ by the original embeddings $\mathbf{X}'$. Finally, $\mathbf{z}^{x_i}$ is the CLS embedding from position $t = 0$ for each sequence i (we apply the same procedure to target sequences $\tilde{\mathbf{Y}}$ to compute $\mathbf{z}^{y_i}$).

Rather than learning a value function (i.e., $V = \text{MLP}_V$), we define the value to be directly equivalent to the input embeddings $\mathbf{X}'$. This ensures that the learned representation is a convex combination of the input embeddings, increasing the likelihood that cross-entity queries are well-aligned. This formulation also facilitates interpretability: the attention weight matrix $\mathbf{A}$ can be directly inspected to identify which tracks contribute most to the artist representation, since the output is an explicit weighted combination of the original track vectors rather than learned transformations of them.

The model processes an input sequence and a target sequence: each is independently passed through the self-attention model to obtain paired embeddings $(\mathbf{z}^{(x_i)}, \mathbf{z}^{(y_i)})_{i=1}^B$ for contrastive learning. During training, this enables the model to learn how to weigh each track in the set, forming a meaningful aggregate representation. During inference, the CLS output serves as the artist embedding. With $n=20$ tracks, MCN processes $\sim 5,000$ artists/s on a single V100 GPU, and attention weights are cached for real-time serving.

3.3. InfoNCE Contrastive Loss with Bag-of-Tracks (BoT) Pairs

The model is trained using InfoNCE loss [20], a contrastive loss that encourages similar sequences to have closer embeddings while pushing apart dissimilar sequences. For each positive pair $(\mathbf{z}^{(x_i)}, \mathbf{z}^{(y_i)})$, the goal is to encourage $\mathbf{z}^{(x_i)} \cdot \mathbf{z}^{(y_i)}$ to be larger than similarity with any negative pair $(\mathbf{z}^{(x_i)}, \mathbf{z}^{(y_j)})$ for $j \neq i$. We discuss positive pair construction in Section 4.1.

Given a batch of B paired embeddings, the InfoNCE loss is:

$$\ell_{\text{InfoNCE}}(\mathbf{z}^{(x_i)}, \mathbf{z}^{(y_i)}) = -\log \frac{\exp(\mathbf{z}^{(x_i)} \cdot \mathbf{z}^{(y_i)} / \tau)}{\sum_{j=1}^B \exp(\mathbf{z}^{(x_i)} \cdot \mathbf{z}^{(y_j)} / \tau)},$$

where $\tau > 0$ is the temperature parameter and j indexes all batch examples.

4. Experiments

4.1. Music Consumption Dataset and Track Embedding Preprocessing

We collected music consumption data in the form of track-to-track and artist-to-artist co-listening pairs from 365 days of listening history. Each pair is defined by two entities that were co-listened to in the same session. We represent these relationships as a weighted, undirected graph, where nodes are items and higher edge weights indicate more frequent co-listening.

Using this data, we apply the graph processing steps described in Section 3.1 with entity-specific adjustments. An edge is formed between two artists if they have any co-listening signal within the trailing 365 days (rather than only within a single session). We rank pairwise similarities and retain only mutual top-100 neighbors for each artist. For tracks, we follow the same procedure but retain mutual top-1,000 neighbors, reflecting the denser nature of track-level relationships.

To avoid sparsity for long-tail entities, we apply a densification step. In the artist graph, if an artist has fewer than 50 retained connections after pruning, we allow additional edges to be formed with artists within three hops. Similarly, in the track graph, tracks with fewer than 100 retained neighbors are allowed to connect to tracks within up to five hops. This improves connectivity while preserving locality in the similarity structure.

After constructing the final similarity graphs, we derive two training datasets. First, from the artist-artist graph, we form set-set pairs where each artist is represented as a set of its associated tracks. If two artists are connected in the artist graph, their corresponding track sets form a positive pair. We denote this dataset as $\mathcal{D}_{AtoA}$, which contains approximately 11 million positive pairs. Second, from the track-track graph, we construct set-item pairs by selecting a target track and sampling a “bag” of its neighboring tracks from the similarity graph. This produces approximately 7.5 million positive pairs, denoted $\mathcal{D}_{TtoT}$.

The artist population encompasses all artists in the Amazon Music catalog with at least one track having meaningful listening activity, yielding approximately 600,000 artists without additional filtering.

4.2. Training Setup

We train the proposed MCN model and its variants after an extensive hyperparameter search, tuning batch size, number of training epochs, optimizer settings, temperature in the contrastive loss, and network architecture. For this search, 20% of the data from both $\mathcal{D}_{TtoT}$ and $\mathcal{D}_{AtoA}$ is held out as a validation set. The split is performed at the edge level (i.e., by sampling positive pairs): all entities may appear in both splits, but specific co-listening relationships are partitioned. We choose edge-level splitting because our deployment processes all catalog artists. The goal is to learn generalizable aggregation weights rather than generalize to entirely unseen entities.

Based on this validation process, all models are trained with the following configuration: number of epochs = 50, batch size $B = 1024$, embedding dimension $d = 128$, and uniform parameter initialization with magnitude 0.1. We use the Adam optimizer [21] with a learning rate of 1×10^{-4} and temperature $\tau = 0.1$ in the contrastive objective. To reduce overfitting, we apply L_2 regularization with a weight decay of 0.001. For the multilayer perceptrons (MLP_Q and MLP_K), we use two hidden layers with 1024 units each. To handle variable-sized track sets, we evaluate two maximum set sizes $k \in \{10, 20\}$, determining whether to apply random subsampling or zero-padding for a given artist. All experiments are conducted on a single NVIDIA Tesla V100 GPU. Training completes within 8 hours per model.

4.3. Offline Evaluation Tasks

We evaluate artist embeddings on two downstream tasks: music-entity retrieval and manually-annotated artist-artist similarities.

4.3.1. Music-Entity (Container) Retrieval

We evaluate the effectiveness of MCN through an array of retrieval tasks measuring how well artist embeddings align with other entity types including tracks, albums, customer profiles, and other artists.

For each task, we compute ranking-based metrics at cutoffs $k \in \{1, 3, 10, 50, 100\}$. The evaluation metrics include Precision@k, Recall@k, F1@k, nDCG@k, and mAP. We primarily use mAP as the main summary metric, as it is widely adopted in retrieval settings. In our experiments, F1 and mAP are highly correlated ($\rho > 0.95$).

When artists are used as queries, we evaluate over the full set of approximately 600,000 artists. For artist-album and artist-track retrieval, ground truth is constructed by ranking each artist’s associated albums or tracks according to play count. For artist-artist retrieval, the ground truth is derived from ranked similarities in the co-listening graph. For the profile-artist retrieval task, we select the 500,000 most active listeners and build ground truth by ranking artists per profile according to historical play counts. Predictions are generated using KNN search in the embedding space, and profile embeddings are constructed following the method described in [22].

To prevent information leakage, evaluation ground truths use signals distinct from training data. Artist-track and artist-album tasks use catalog association metadata ranked by play counts, which is orthogonal to session co-listening. Profile-artist evaluation aggregates listening at the profile level, structurally distinct from pairwise session signals. For artist-artist retrieval, the ground truth is derived from the same co-listening graph, so we treat the manually curated evaluation (Section 5.1.2) as the most reliable indicator of performance for this task.

4.3.2. Baseline Models

We compare MCN against several baseline approaches for constructing artist embeddings:

- *top10_unweighted*: The unweighted average of an artist’s top 10 tracks, ranked by age-normalized counts of plays longer than 30 seconds.
- *top10_weighted*: A weighted average where weights are proportional to the age-normalized >30s play counts.
- *top10_log_norm*: Similar to weighted, but using the logarithm of play counts to reduce dominance of popular tracks.
- *sim_based*: An embedding model trained directly on the artist similarity graph (Section 4.1), mirroring the track embedding method but using artist-artist co-listening signals.

In early experiments, we also explored alternative neural aggregation strategies, including MCN-style models without a CLS token (using mean/max pooling over attended representations) and architectures with learned value projections (MLP_V). Models without the CLS token lacked a dedicated aggregation point and produced less stable training. Models with learned value projections performed comparably on same-entity tasks but degraded cross-entity retrieval, likely because transformed embeddings diverged from the shared track embedding space. These approaches are not included in the main comparison.

We focus on MCN variants differing in the number of input tracks and training data. We consider $k \in \{10, 20\}$ tracks per artist because this range provides a practical balance between representation quality and inference efficiency. Larger values of k increase latency and memory cost while yielding diminishing performance gains. The variants are:

- *MCN_top10_all*: Top 10 tracks ($k = 10$), trained on both $\mathcal{D}_{TtoT}$ and $\mathcal{D}_{AtoA}$.
- *MCN_top20_all*: Top 20 tracks ($k = 20$), trained on both datasets.

Table 1

Evaluation of the artist retrieval task. Averaged metrics across retrieval types (artist-to-track, artist-to-artist, track-to-artist, profile-to-artist, artist-to-album). We report absolute metrics and Δ vs. the best-performing baseline.

Model	mAP (Δ)	F1 (Δ)	nDCG (Δ)	Prec. (Δ)
<i>Baselines</i>				
top10_unweighted	.4231 (-.0024)	.1931 (-.0011)	.5013 (-.0027)	.3062 (-.0023)
top10_weighted	.3925 (-.0330)	.1799 (-.0143)	.4729 (-.0311)	.2911 (-.0174)
top10_log_norm	.4255 (+.0000)	.1942 (+.0000)	.5040 (+.0000)	.3085 (+.0000)
sim_based	.3587 (-.0668)	.1629 (-.0313)	.4347 (-.0693)	.2790 (-.0295)
<i>Proposed (MCN)</i>				
MCN_top10_all	.4518 (+.0263)	.2002 (+.0060)	.5328 (+.0288)	.3338 (+.0253)
MCN_top20_all	.4381 (+.0126)	.1977 (+.0035)	.5177 (+.0137)	.3195 (+.0110)
MCN_top10_AtoA	.4529 (+.0274)	.1996 (+.0054)	.5344 (+.0304)	.3358 (+.0273)
MCN_top20_AtoA	.4439 (+.0184)	.1987 (+.0045)	.5245 (+.0205)	.3258 (+.0173)
MCN_top10_TtoT	.5515 (+.1260)	.2279 (+.0337)	.6267 (+.1227)	.3946 (+.0861)
MCN_top20_TtoT	.5646 (+.1391)	.2324 (+.0382)	.6384 (+.1344)	.4006 (+.0921)

- *MCN_top10_TtoT*: Top 10 tracks, trained on $\mathcal{D}_{TtoT}$ only.
- *MCN_top20_TtoT*: Top 20 tracks, trained on $\mathcal{D}_{TtoT}$ only.
- *MCN_top10_AtoA*: Top 10 tracks, trained on $\mathcal{D}_{AtoA}$ only.
- *MCN_top20_AtoA*: Top 20 tracks, trained on $\mathcal{D}_{AtoA}$ only.

We note that heuristic baselines use 10 tracks while the best MCN variant uses 20. To control for this: (1) *MCN_top10_TtoT* still outperforms all baselines across most tasks (Table 1), and (2) preliminary experiments with top-20 unweighted/weighted averages showed $<0.5\%$ relative mAP change over their top-10 counterparts, indicating that simply increasing the number of averaged tracks does not explain MCN’s gains.

4.4. Online A/B Evaluation

We conducted a controlled online A/B experiment to measure the real-world impact of MCN within the Amazon Music production recommendation system. The experiment was implemented as a customer-level randomized online experiment across global territories with equal traffic allocation (33.3% each): Control (C) = the production model (*top10_unweighted*), Treatment 1 (T1) = *top10_log_norm*, and Treatment 2 (T2) = the proposed MCN model.

The experiment ran for two weeks and impacted artist-driven recommendation surfaces including artist-to-artist, artist-to-track, artist-to-album, artist-to-station, and artist-to-playlist. In the production system, artist embeddings serve as the index for approximate nearest-neighbor (ANN) retrieval: when a customer interacts with an artist, the system retrieves neighbors of that artist’s embedding to populate recommendation candidates, which are subsequently re-ranked before presentation. We evaluated Play Days, Listening Hours, and Long-Term Engagement Retention, which capture both short-term consumption and sustained engagement.

5. Results

5.1. Offline Results

5.1.1. Retrieval Tasks (Quantitative Analysis)

Table 1 summarizes average performance across all retrieval tasks. *MCN_top20_TtoT* achieves the best overall performance, with statistically significant gains over all baselines (paired t -test, $p < 0.001$). The

Table 2Evaluation of three models on four retrieval tasks (mAP) $\pm$ Standard Error

Retrieval Type	Rank	top10_unweighted	MCN_top10_all	MCN_top20_TtoT
artist-to-track	@1	0.8722 $\pm$ 0.006	0.9209 $\pm$ 0.004	0.9330 $\pm$ 0.004
	@10	0.7817 $\pm$ 0.006	0.7427 $\pm$ 0.007	0.8199 $\pm$ 0.006
	@100	0.7523 $\pm$ 0.007	0.7022 $\pm$ 0.007	0.7932 $\pm$ 0.006
artist-to-album	@1	0.7824 $\pm$ 0.006	0.7902 $\pm$ 0.006	0.8204 $\pm$ 0.006
	@10	0.4378 $\pm$ 0.008	0.4071 $\pm$ 0.008	0.4590 $\pm$ 0.008
	@100	0.3551 $\pm$ 0.008	0.3310 $\pm$ 0.008	0.3699 $\pm$ 0.008
profile-to-artist	@1	0.4468 $\pm$ 0.008	0.4938 $\pm$ 0.008	0.7951 $\pm$ 0.006
	@10	0.2574 $\pm$ 0.007	0.3000 $\pm$ 0.007	0.5265 $\pm$ 0.008
	@100	0.1154 $\pm$ 0.005	0.1324 $\pm$ 0.005	0.2016 $\pm$ 0.006
artist-to-artist	@1	0.5165 $\pm$ 0.008	0.7220 $\pm$ 0.007	0.6992 $\pm$ 0.007
	@10	0.3260 $\pm$ 0.007	0.5018 $\pm$ 0.008	0.4676 $\pm$ 0.008
	@100	0.2512 $\pm$ 0.007	0.3807 $\pm$ 0.007	0.3152 $\pm$ 0.007

Table 3

Evaluation on the manually-annotated artist similarity dataset (Wilson Score 95% CI) under Fixed k and Adaptive $k = s_a$. We report the % of retrieved artists that are Relevant/Highly Relevant (Rel./HR) and Highly Relevant only (HR). Δ is vs. the *best-performing baseline* (percentage points).

Cat.	Model	Fixed k				Adaptive $k = s_a$			
		Rel./HR	Δ	HR	Δ	Rel./HR	Δ	HR	Δ
Base	<i>top10_unweighted</i>	46.2%	−3.3	34.9%	−1.5	56.0%	−9.0	39.7%	+0.0
	<i>top10_weighted</i>	35.1%	−14.4	23.2%	−13.2	38.5%	−26.5	22.6%	−17.1
	<i>top10_log_norm</i>	45.9%	−3.6	34.1%	−2.3	56.0%	−9.0	39.3%	−0.4
	<i>sim_based</i>	49.5%	+0.0	36.4%	+0.0	65.0%	+0.0	38.0%	−1.7
Prop.	<i>MCN_top10_all</i>	53.2%	+3.7	36.1%	−0.3	56.4%	−8.6	35.0%	−4.7
	<i>MCN_top20_all</i>	50.5%	+1.0	36.4%	+0.0	57.7%	−7.3	37.2%	−2.5
	<i>MCN_top10_AtoA</i>	50.6%	+1.1	37.1%	+0.7	59.4%	−5.6	38.5%	−1.2
	<i>MCN_top20_AtoA</i>	50.5%	+1.0	36.4%	+0.0	57.3%	−7.7	37.6%	−2.1
	<i>MCN_top10_TtoT</i>	59.7%	+10.2	48.4%	+12.0	70.5%	+5.5	46.2%	+6.5
	MCN_top20_TtoT	60.7%	+11.2	53.0%	+16.6	74.8%	+9.8	48.7%	+9.0

largest gains appear in cross-entity tasks (e.g., track→artist, up to 15.2% relative mAP improvement). The one exception is the graph-derived artist–artist task, where *sim_based* performs best due to label leakage from its training graph, making the manually curated evaluation (Section 5.1.2) a more reliable indicator. Table 2 provides per-task breakdowns. Detailed results appear in Appendix 1.

We attribute the TtoT variant’s superiority to its richer supervision: the track graph provides ~ 7.5 M pairs with up to 1,000 neighbors per node, offering denser local structure than the artist graph (~ 11 M pairs, 100 neighbors). The combined “all” variants underperform TtoT alone, suggesting that mixing supervision at different granularities introduces conflicting gradients that hinder optimization.

5.1.2. Performance on Manually-Annotated Artist Similarity Dataset

We evaluate with a more semantically grounded dataset where music experts manually annotated artist similarity relationships. For each of 26 seed artists—selected to span genres (pop, hip-hop, rock, country, electronic, Latin) and popularity tiers—three experienced curators labeled candidate artists using relevance scores on a scale where lower values indicate higher similarity: 1 indicates *highly relevant* and 2 indicates *relevant*. Annotation considered sonic similarity, shared audience, and genre overlap, with periodic calibration sessions to ensure consistency.

For each seed artist, we define s_a as the number of artists labeled with scores 1 or 2. Using learned artist embeddings, we retrieve the nearest neighbors for each seed artist and measure overlap with

the curated relevant sets. We report two evaluation settings: (1) fixed $k = 9$, corresponding to the average number of candidate relevant artists per seed, and (2) adaptive $k = s_a$, allowing variable neighborhood size. Results in Table 3 show that *MCN_top20_TtoT* achieves substantially higher recall of curator-identified relevant artists than all baselines under both criteria.

Table 4

Online A/B results showing relative lift over the production Control (*top10_unweighted*). “Prob>C” denotes the probability that the treatment outperforms Control.

Metric	<i>top10_log_norm</i>		<i>MCN_top20_TtoT</i>	
	Lift	Prob>C	Lift	Prob>C
Play Days	3 bps	86.87%	4 bps	92.86%
Listen Hours	7 bps	86.09%	8 bps	87.14%
Long-Term Engagement	8 bps	93.58%	11 bps	98.54%

5.2. Online Experiment Result

Table 4 summarizes the key results comparing both treatments against the production baseline. MCN delivered consistent improvements across primary engagement metrics and met launch criteria. In particular, MCN increased long-term engagement retention by +11 bps, alongside gains in overall Play Days (+4 bps) and total Listening Hours (+8 bps). Positive trends were observed across both visual and voice-driven experiences, indicating that improvements generalized across interaction surfaces rather than being confined to a single product entry point.

6. Conclusion and Lessons Learned

We introduced MusicContainerNet (MCN), a self-attention neural network for learning artist embeddings by aggregating track-level representations. MCN consistently outperforms simple aggregation baselines on multiple offline tasks, and a live A/B test confirmed substantial customer engagement gains (+4 bps in Play Days, +8 bps in Listening Hours, +11 bps in Long-Term Engagement compared to Control).

MCN’s success reflects a deliberate balance: one attention layer over at most 20 tracks satisfies strict latency and memory budgets while outperforming naive averaging. Key lessons for practitioners: (1) build production constraints into the model design from the start; (2) validate with a live A/B test, since offline metrics do not fully predict customer impact; and (3) prefer a simpler, well-tuned model over a complex one that cannot be served at scale. Future work will extend MCN to albums and playlists and incorporate richer signals beyond co-listening.

Limitations. Our evaluation relies on proprietary data that cannot be released due to business constraints. The 20-track cap may under-represent the stylistic breadth of prolific artists, and future work could explore adaptive set sizes. The golden evaluation covers 26 seed artists without formal inter-annotator agreement metrics. We have not evaluated performance stratified by artist popularity tier, which we leave to future work.

Declaration on Generative AI

The authors used generative AI tools solely for improving the language, formatting, and readability of this manuscript. All research, data analysis, and conclusions were conducted independently by the authors.

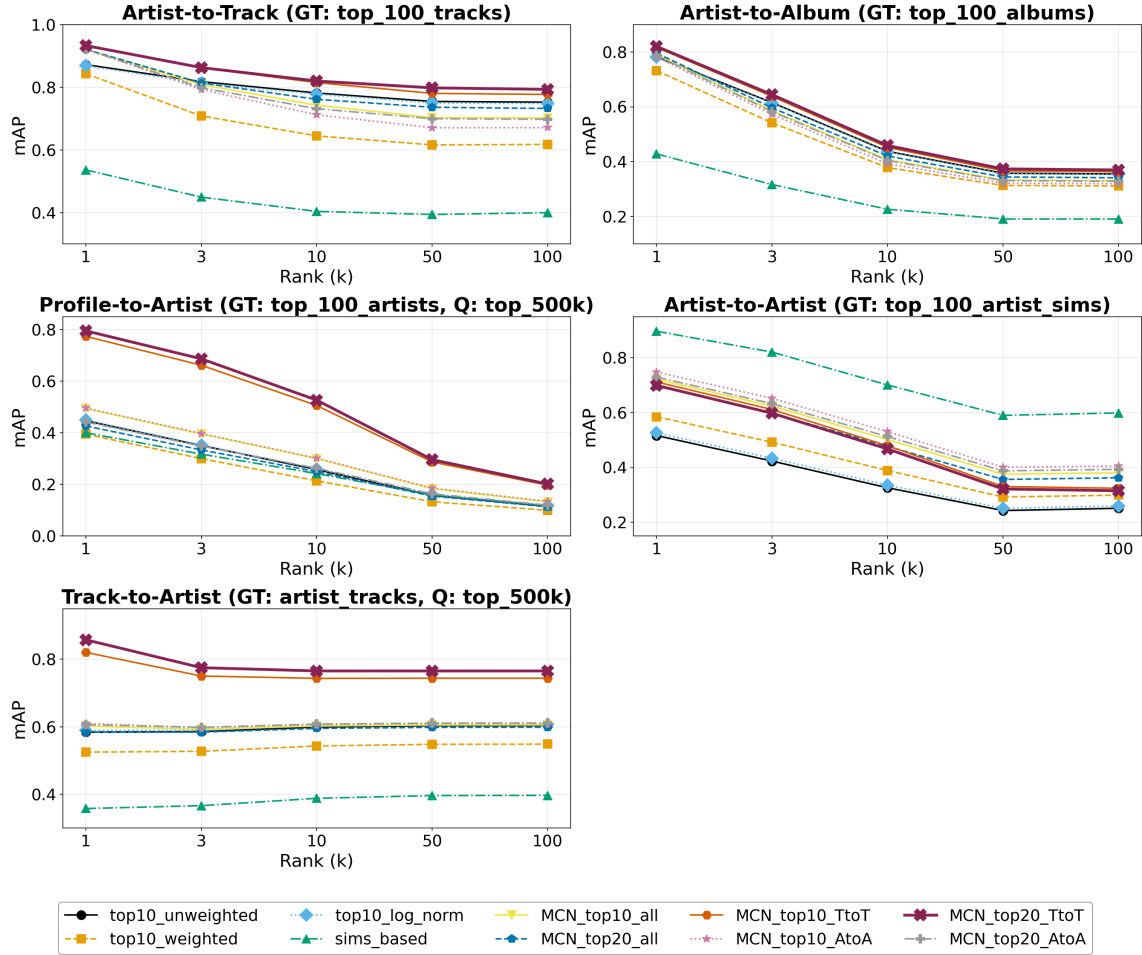
References

- [1] M. Schedl, H. Zamani, C.-W. Chen, Y. Deldjoo, M. Elahi, Current challenges and visions in music recommender systems research, *International Journal of Multimedia Information Retrieval* 7 (2018) 95–116.
- [2] S. Oramas, O. Nieto, F. Barbieri, X. Serra, Multi-label music genre classification from audio, text, and images using deep features, *arXiv preprint arXiv:1707.04916* (2017).
- [3] G. Vigliensoni, I. Fujinaga, Automatic music recommendation systems: Do demographic, profiling, and contextual features improve their performance?, in: *ISMIR*, 2016, pp. 94–100.
- [4] J. Park, J. Lee, J. Park, J.-W. Ha, J. Nam, Representation learning of music using artist labels, *arXiv preprint arXiv:1710.06648* (2017).
- [5] B. McFee, G. R. Lanckriet, The natural language of playlists., in: *ISMIR*, volume 11, 2011, pp. 537–541.
- [6] D. Liang, M. Zhan, D. P. Ellis, Content-aware collaborative music recommendation using pre-trained neural networks., in: *ISMIR*, 2015, pp. 295–301.
- [7] P. Knees, M. Schedl, A survey of music similarity and recommendation from music context data, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 10 (2013) 1–21.
- [8] D. Bogdanov, H. Boyer, et al., How much metadata do we need in music recommendation?: A subjective evaluation using preference sets, in: Klapuri A, Leider C, editors. *12th International Society for Music Information Retrieval Conference (ISMIR 2011)*; 2011 Oct 24-28; Miami, USA. Montréal: ISMIR; 2011. p. 97-102, *International Society for Music Information Retrieval (ISMIR)*, 2011.
- [9] Z. Nasrullah, Y. Zhao, Music artist classification with convolutional recurrent neural networks, in: *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2019, pp. 1–8.
- [10] A. Van den Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, *Advances in neural information processing systems* 26 (2013).
- [11] F. Korzeniowski, S. Oramas, F. Gouyon, Artist similarity for everyone: A graph neural network approach., *Trans. Int. Soc. Music. Inf. Retr.* 5 (2022) 129–140.
- [12] M. Won, S. Chun, X. Serra, Toward interpretable music tagging with self-attention, *arXiv preprint arXiv:1906.04972* (2019).
- [13] X. Liu, S. Song, M. Zhang, Y. Huang, Matt. a multiple-instance attention mechanism for long-tail music genre classification, in: *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2022, pp. 782–787.
- [14] B. Ma, T. Greer, D. Knox, S. Narayanan, A computational lens into how music characterizes genre in film, *PloS one* 16 (2021) e0249957.
- [15] G. Linden, B. Smith, J. York, Amazon. com recommendations: Item-to-item collaborative filtering, *IEEE Internet computing* 7 (2003) 76–80.
- [16] M. Radovanovic, A. Nanopoulos, M. Ivanovic, Hubs in space: Popular nearest neighbors in high-dimensional data, *Journal of Machine Learning Research* 11 (2010) 2487–2531.
- [17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, *Advances in neural information processing systems* 26 (2013).
- [18] O. Barkan, N. Koenigstein, Item2vec: neural item embedding for collaborative filtering, in: *2016 IEEE 26th international workshop on machine learning for signal processing (MLSP)*, IEEE, 2016, pp. 1–6.
- [19] J. D. M.-W. C. Kenton, L. K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: *Proceedings of naacL-HLT*, volume 1, Minneapolis, Minnesota, 2019, p. 2.
- [20] A. v. d. Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, *arXiv preprint arXiv:1807.03748* (2018).
- [21] K. Diederik, B. Jimmy, Adam: A method for stochastic optimization. *arxiv prepr. int*, *arXiv preprint arXiv:1412.6980* (2014).

- [22] M. Gao, L. Chen, X. He, A. Zhou, Bine: Bipartite network embedding, in: The 41st international ACM SIGIR conference on research & development in information retrieval, 2018, pp. 715–724.

A. Appendix

Mean Average Precision (mAP) Across Retrieval Tasks



Appendix 1: Mean Average Precision (mAP) performance across five retrieval tasks at rank cutoffs $k \in \{1, 3, 10, 50, 100\}$. Ground truth is defined as the most-played items belonging to each query artist, except in the artist-to-artist case where it is derived from co-listening similarity scores. The profile-to-artist and track-to-artist evaluations use the top 500K queries (most active profiles and most popular tracks, respectively). Artist-to-track, artist-to-album, and artist-to-artist evaluations use all available queries ($\sim 600K$ artists). The *sims_based* model performs best on artist-to-artist because its training graph overlaps with the ground truth (see Section 5.1.1). Line styles and a colorblind-friendly palette are used to ensure accessibility.