Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
τομεας σηματων, ελεγχου και ρομποτικης

Evaluating the Effectiveness of Pretrained Audio Representations in Session-Based Music Recommender Systems

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Χαράλαμπου Σπυρίδωνα Μπότσα

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.

ΕΡΓΑΣΤΗΡΙΟ ΟΡΑΣΗΣ ΥΠΟΛΟΓΙΣΤΩΝ, ΕΠΙΚΟΙΝΩΝΙΑΣ ΛΟΓΟΥ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑΣ ΣΗΜΑΤΩΝ
Αθήνα, Οκτώβριος 2025

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου και Επεξεργασίας Σημάτων

Evaluating the Effectiveness of Pretrained Audio Representations in Session-Based Music Recommender Systems

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Χαράλαμπου Σπυρίδωνα Μπότσα

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13η Οκτωβρίου, 2025.

.....	.....	.....
Πέτρος Μαραγκός	Αλέξανδρος Ποταμιάνος	Γεράσιμος Ποταμιάνος
Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής Παν/μιου Θεσσαλίας

Αθήνα, Οκτώβριος 2025

.....

Χαραλαμπος Σπυριδων Μποτσας
Διπλωματούχος Ηλεκτρολόγος Μηχανικός
και Μηχανικός Υπολογιστών Ε.Μ.Π.

στην οικογένειά μου

Περίληψη

Οι πρόσφατες εξελίξεις στον τομέα της Ανάκτησης Μουσικής Πληροφορίας έχουν αναδείξει τη δυνατότητα των προεκπαιδευμένων ηχητικών αναπαραστάσεων να επιτυγχάνουν υψηλές επιδόσεις σε διάφορα προβλήματα, όπως η ταξινόμηση μουσικού είδους ή η αυτόματη επισήμανση μουσικής. Ωστόσο, οι δυνατότητές τους στο πεδίο των συστημάτων συστάσεων μουσικής—και ειδικότερα στις διαδοχικές συστάσεις—δεν έχει διερευνηθεί επαρκώς. Η παρούσα εργασία εξετάζει κατά πόσο οι ηχητικές αναπαραστάσεις που προέρχονται από προεκπαιδευμένα μοντέλα μπορούν να αποδειχθούν χρήσιμες σε ένα πρόβλημα παραγωγής συστάσεων με βάση συνεδρίες ακρόασης.

Χρησιμοποιώντας το Music4All, ένα δημόσια διαθέσιμο σύνολο δεδομένων που περιέχει ιστορικά ακρόασης μαζί με ηχητικά αποσπάσματα, δημιουργούμε συνεδρίες ακρόασης και εξάγουμε αναπαραστάσεις από τρία προεκπαιδευμένα μοντέλα: MusiCNN, MERT και ένα μοντέλο εκπαιδευμένο στην ομοιότητα μεταξύ καλλιτεχνών. Στη συνέχεια, πειραματιζόμαστε με διαφορετικές στρατηγικές ενσωμάτωσης των αναπαραστάσεων σε συστήματα συστάσεων βασισμένα στην αρχιτεκτονική Transformer, ακολουθώντας δύο υπάρχοντα πλαίσια διαδοχικών συστάσεων: τα Transformers4Rec και Represent-Then-Aggregate (RTA).

Τα αποτελέσματα δείχνουν ότι, παρότι οι ηχητικές αναπαραστάσεις επιταχύνουν τη σύγκλιση της εκπαίδευσης, δεν ξεπερνούν σταθερά τα μοντέλα αναφοράς με τυχαία αρχικοποίηση στην περίπτωση του Transformers4Rec. Αντίθετα, στο RTA οι προεκπαιδευμένες αναπαραστάσεις συνεισφέρουν μικρές αλλά συνεπείς βελτιώσεις, χωρίς ωστόσο να υπερτερούν έναντι μοντέλων αναφοράς που βασίζονται σε μεταδεδομένα. Οι αντίθετες αυτές τάσεις μεταξύ των δύο πλαισίων μπορούν να αποδοθούν στη διαφορετική διατύπωση του προβλήματος και στους διαφορετικούς στόχους εκμάθησης, καταδεικνύοντας ότι το όφελος των ηχητικών αναπαραστάσεων εξαρτάται σε σημαντικό βαθμό από το πόσο καλά ευθυγραμμίζεται το τελικό πρόβλημα με τη σημασιολογία που αυτές αποτυπώνουν. Το γεγονός αυτό επιβεβαιώνεται και από την παρατήρηση ότι η περαιτέρω προσαρμογή (fine-tuning) των αρχικών αναπαραστάσεων με δεδομένα αλληλεπιδράσεων βελτιώνει την αποτελεσματικότητά τους, αναδεικνύοντας έτσι την αξία της προσαρμογής πεδίου. Τέλος, μέσω μίας μελέτης αφαίρεσης αναδείχθηκε ο καθοριστικός ρόλος της σωστής επιλογής υπερπαραμέτρων στην πλήρη αξιοποίηση των προεκπαιδευμένων αναπαραστάσεων.

Λέξεις Κλειδιά — Συστήματα Συστάσεων Μουσικής, Ανάκτηση Μουσικής Πληροφορίας, Εκμάθηση Αναπαραστάσεων, Διαδοχικές Συστάσεις, Νευρωνικά Δίκτυα

Abstract

Recent advances in Music Information Retrieval (MIR) have showcased the ability of pretrained audio representations to achieve competitive performance in various downstream tasks. However, their potential in the area of music recommender systems, and particularly in sequential recommendation, has not been fully explored. The present work examines whether audio-based embeddings from pretrained models can prove useful in a session-based recommendation setting.

Using Music4All, an openly available dataset of listening histories along with audio segments, we generate listening sessions and we extract embeddings from three pretrained models: MusiCNN, MERT and a custom artist-based model and experiment with different strategies of integrating them into Transformer-based recommender architectures built upon two sequential recommendation frameworks: Transformers4Rec and Represent-Then-Aggregate (RTA).

Results show that while audio representations accelerate convergence, they do not consistently outperform randomly initialized baselines under the Transformers4Rec setup. In contrast, within the RTA framework, content-based embeddings yield modest but consistent improvements, but fail to surpass other metadata-driven baselines. These opposite trends across the two frameworks can be partly attributed to their different task formulation and training objective, highlighting that the benefit of pretrained audio representations depends on how well the downstream objective aligns with the semantics of the embedding space. This is also supported by the fact that fine-tuning the original embeddings on behavioral data improves their effectiveness, outperforming the out-of-the-box embeddings and demonstrating the value of lightweight domain adaptation. Finally, an ablation study revealed the crucial role of proper hyperparameter choice in leveraging the full potential of the pretrained representations.

Keywords — Music Recommender Systems, Music Information Retrieval, Representation Learning, Sequential Recommendation, Neural Networks

Ευχαριστίες

Ολοκληρώνοντας τις σπουδές μου σε αυτή τη σχολή, θα ήθελα να ευχαριστήσω θερμά τον καθηγητή κ. Πέτρο Μαραγκό για την ευκαιρία που μου έδωσε να εκπονήσω την παρούσα διπλωματική εργασία στο εργαστήριό του. Κατά την εκπόνηση της εργασίας, απέκτησα πολύτιμη εμπειρία και ανακάλυψα σε μεγαλύτερο βάθος τον τομέα της Μηχανικής Μάθησης και τις εφαρμογές της σε ένα σημαντικό για μένα τομέα, αυτόν της μουσικής.

Επιπλέον, θα ήθελα να ευχαριστήσω ιδιαίτερα τη Νάνσυ Ζλατίντση και τον Χρήστο Γαρούφη, συνεπιβλέποντες της διπλωματικής, οι οποίοι έπαιξαν καθοριστικό ρόλο στην ολοκλήρωσή της. Η καθοδήγηση και η βοήθειά τους ήταν πολύ σημαντικές στη διαμόρφωση του παρόντος αποτελέσματος.

Τέλος, θα ήθελα να ευχαριστήσω από καρδιάς την οικογένειά μου, για τη διαρκή στήριξη και εμπιστοσύνη που μου έδειξαν καθόλη τη διάρκεια των σπουδών μου, όπως επίσης και τους φίλους μου με τους οποίους περάσαμε μαζί αυτή την απαιτητική εμπειρία.

Χαράλαμπος Σπυρίδων Μπότσας
Οκτώβριος 2025

Contents

List of Figures

List of Tables

Εκτεταμένη Περίληψη στα Ελληνικά

Συστήματα Συστάσεων Μουσικής

Τα τελευταία χρόνια, οι υπηρεσίες αναπαραγωγής μουσικής έχουν επαναπροσδιορίσει τον τρόπο με τον οποίο απολαμβάνουμε τη μουσική. Οι υπηρεσίες αυτές έχουν σε μεγάλο βαθμό αντικαταστήσει τα φυσικά μέσα, όπως CDs και δίσκους βινυλίου, προσφέροντας ολοένα και διευρυνόμενους καταλόγους τραγουδιών σε εκατομμύρια χρήστες. Ωστόσο, καθώς δεν είναι εφικτό οι χρήστες να περιηγηθούν εξαντλητικά σε αυτούς τους απέραντους καταλόγους, ένα από τα βασικά ζητήματα που έχουν να αντιμετωπίσουν οι πλατφόρμες αυτές είναι το πώς θα φέρουν τους ακροατές πιο κοντά στη μουσική που τους αρέσει. Αυτόν ακριβώς τον ρόλο αναλαμβάνουν τα συστήματα συστάσεων μουσικής, τα οποία έχουν καταστεί θεμελιώδες στοιχείο των ψηφιακών πλατφορμών, καθορίζοντας τον τρόπο με τον οποίο εκατομμύρια ακροατές ανακαλύπτουν νέα μουσική και αλληλεπιδρούν με τους αγαπημένους τους καλλιτέχνες.

Συνεργατική Διήθηση (Collaborative Filtering)

Για πολλά χρόνια, τα συστήματα συστάσεων βασίζονταν σε τεχνικές που παράγουν προτάσεις αξιοποιώντας μοτίβα από τα δεδομένα αλληλεπιδράσεων χρήστη–αντικειμένου. Οι αλγόριθμοι αυτοί έχουν υιοθετηθεί ευρέως και έχουν παρουσιάσει σημαντική επιτυχία σε διάφορους τομείς, συμπεριλαμβανομένου αυτού της μουσικής. Ένα χαρακτηριστικό παράδειγμα είναι η μέθοδος της Συνεργατικής Διήθησης (Collaborative Filtering (CF)), η οποία εντοπίζει ομοιότητες μεταξύ χρηστών ή αντικειμένων βάσει μεγάλων πινάκων αλληλεπιδράσεων και προτείνει τραγούδια που έχουν ακούσει άλλοι χρήστες με παρόμοιο γούστο [92]. Τα δεδομένα αλληλεπιδράσεων μπορεί να περιλαμβάνουν άμεσες αξιολογήσεις, όπως βαθμολογίες 1–5 αστεριών, που εκφράζουν ξεκάθαρα την προτίμηση ενός χρήστη, ή έμμεσες μορφές ανατροφοδότησης, όπως αριθμός αναπαραγωγών, αγορές αντικειμένων ή άλλα σήματα που υποδηλώνουν προτίμηση με πιο έμμεσο τρόπο [48]. Μια άλλη οικογένεια αλγορίθμων CF βασίζεται σε τεχνικές παραγοντοποίησης πινάκων (matrix factorization) προκειμένου να αναπαραστήσει χρήστες και αντικείμενα σε έναν κοινό διανυσματικό χώρο χαμηλότερης διάστασης και στη συνέχεια να ανακαλύψει πρότυπα στις προτιμήσεις των χρηστών [36], [43], [54]. Αυτό όχι μόνο βελτιώνει την ακρίβεια των συστάσεων, αλλά αυξάνει και την κλιμακωσιμότητα και την υπολογιστική αποδοτικότητα των συστημάτων.

Περιορισμοί

Παρά την αποδεδειγμένη αποτελεσματικότητά τους, οι παραδοσιακές προσεγγίσεις παρουσιάζουν αρκετούς περιορισμούς. Ένα βασικό πρόβλημα είναι η αραιότητα δεδομένων, καθώς οι πίνακες αλληλεπιδράσεων χρήστη–αντικειμένου περιέχουν μόνο ένα μικρό ποσοστό όλων των πιθανών αλληλεπιδράσεων, με αποτέλεσμα να καθίσταται πιο δύσκολη η εξαγωγή ουσιαστικών προτύπων αποκλειστικά από τέτοιου είδους δεδομένα. Στην ακραία μορφή του, το πρόβλημα αυτό οδηγεί στο αποκαλούμενο πρόβλημα ψυχρής εκκίνησης (cold-start), το οποίο εμφανίζεται όταν νέοι χρήστες ή νέα αντικείμενα προστίθενται στο σύστημα και το οποίο παραμένει μία από τις μεγάλες προκλήσεις στον συγκεκριμένο τομέα [95]. Ουσιαστικά, λόγω της απουσίας επαρκών αλληλεπιδράσεων, καθίσταται δύσκολο να προταθεί κατάλληλο περιεχόμενο σε έναν νέο χρήστη ή να προταθεί ένα νέο τραγούδι στους υπάρχοντες χρήστες. Ως αποτέλεσμα, οι μέθοδοι αυτές δυσκολεύουν την ανάδειξη νέων καλλιτεχνών και εξειδικευμένων μουσικών ειδών, ενισχύοντας τη μεροληψία δημοφιλίας (popularity bias), όπου τα ήδη δημοφιλή και πολυακουσμένα τραγούδια προτείνονται συχνότερα, ενώ το λιγότερο γνωστό περιεχόμενο παραμένει αφανές. Το φαινόμενο αυτό όχι μόνο εγείρει ζητήματα δικαιοσύνης στα συστήματα συστάσεων, αλλά περιορίζει και την ποικιλία των προτάσεων, οδηγώντας σε μια πιο ομοιογενή εμπειρία ακρόασης [34], [56].

Μέθοδοι Βασισμένες στο Περιεχόμενο (Content-Based Methods)

Για την αντιμετώπιση των παραπάνω περιορισμών, η έρευνα έχει στραφεί σε μεθόδους βασισμένες στο περιεχόμενο (content-based), οι οποίες ορίζουν την ομοιότητα μεταξύ αντικειμένων με βάση χαρακτηριστικά που προέρχονται από το ίδιο το περιεχόμενο και όχι από τα δεδομένα συμπεριφοράς των χρηστών. Τα χαρακτηριστικά αυτά μπορεί να περιλαμβάνουν τόσο χαμηλού επιπέδου πληροφορίες που εξάγονται απευθείας από το ηχητικό σήμα, όσο και πιο υψηλού επιπέδου περιγραφικά στοιχεία, όπως μεταδεδομένα ή ετικέτες από χρήστες [27]. Επειδή τα χαρακτηριστικά αυτά είναι διαθέσιμα ακόμη και όταν δεν υπάρχουν δεδομένα αλληλεπιδράσεων, οι μέθοδοι αυτές είναι ιδιαίτερα χρήσιμες για καινούρια ή σπάνια αναπαραγμένα τραγούδια. Ενσωματώνοντας αυτόν τον τύπο πληροφορίας στη διαδικασία σύστασης, τα συστήματα μπορούν να προτείνουν τραγούδια ακόμη και χωρίς ιστορικό ακροάσεων, μετριάζοντας έτσι το πρόβλημα ψυχρής εκκίνησης. Στην πράξη, πολλά συστήματα συνδυάζουν συμπεριφορική πληροφορία και πληροφορία από το περιεχόμενο, ώστε να επωφεληθούν από τα πλεονεκτήματα και των δύο προσεγγίσεων.

Όταν πρόκειται για χαρακτηριστικά που εξάγονται απευθείας από το ηχητικό σήμα, μια απλή λύση είναι η χρήση προσχεδιασμένων χαρακτηριστικών [25], [120], που έχουν κατασκευαστεί ώστε να αποτυπώνουν τις ηχητικές και φασματικές ιδιότητες του σήματος. Παρότι αυτά τα χαρακτηριστικά είναι χρήσιμα, περιορίζονται εγγενώς ως προς την ικανότητά τους να αναπαριστούν τη σύνθετη, ιεραρχική δομή της μουσικής. Για τον λόγο αυτό, νεότερες εργασίες στράφηκαν στην αξιοποίηση βαθιών νευρωνικών δικτύων για την αυτόματη εκμάθηση αναπαραστάσεων που απεικονίζουν το ηχητικό σήμα σε έναν χώρο υψηλών διαστάσεων, αποτυπώνοντας ουσιαστικές μουσικές ιδιότητες. Πολλές από αυτές τις μεθόδους ακολουθούν το παράδειγμα της επιβλεπόμενης μάθησης (supervised learning) [49], [60], [67], [80] και βασίζονται σε μεγάλα σύνολα δεδομένων με ετικέτες για την καθοδήγηση της διαδικασίας εκμάθησης.

Ωστόσο, δεδομένου ότι η συλλογή τέτοιων ετικετών είναι χρονοβόρα και δαπανηρή, πρόσφατες εργασίες [14], [38], [62], [63], [76], [101], [115] επικεντρώθηκαν σε μεθόδους που αντλούν το εποπτικό σήμα απευθείας από τα ίδια τα δεδομένα, χωρίς την ανάγκη ανθρωπογενούς επισήμανσης—μια προσέγγιση γνωστή ως αυτοεπιβλεπόμενη μάθηση (self-supervised learning). Στο πεδίο της Εξαγωγής Μουσικής Πληροφορίας (Music Information Retrieval (MIR)), πολυάριθμα μοντέλα έχουν προεκπαιδευτεί σε μεγάλες συλλογές μη επισημασμένων ηχητικών δειγμάτων και μπορούν στη συνέχεια να χρησιμοποιηθούν για πλήθος προβλημάτων, όπως η αυτόματη επιλογή ετικετών και η ταξινόμηση είδους. Αυτό επιτυγχάνεται είτε με περαιτέρω προσαρμογή (fine-tuning) στο τελικό πρόβλημα είτε με χρήση του μοντέλου απλώς ως εξαγωγέα χαρακτηριστικών χωρίς επιπλέον εκπαίδευση. Τέτοιες προσεγγίσεις μεταφοράς μάθησης (transfer learning) συχνά επιτυγχάνουν ίση ή και ανώτερη απόδοση από εξειδικευμένα μοντέλα που εκπαιδεύονται αποκλειστικά για το συγκεκριμένο σκοπό και αποδεικνύονται ιδιαίτερα χρήσιμες όταν τα διαθέσιμα επισημασμένα δεδομένα είναι περιορισμένα.

Διαδοχικές Συστάσεις (Sequential Recommendation)

Παρά την εντυπωσιακή πρόοδο στην εκμάθηση ισχυρών ηχητικών αναπαραστάσεων, μια άλλη σημαντική διάσταση των συστάσεων αφορά τη μοντελοποίηση της χρονικής εξέλιξης των προτιμήσεων των χρηστών. Αν και είναι λογικό να υποθέσουμε ότι οι προτιμήσεις ενός χρήστη αλλάζουν με τον χρόνο, τα συμβατικά μοντέλα αντιμετωπίζουν το ιστορικό του ως στατικό και αγνοούν τη χρονική δυναμική της συμπεριφοράς. Αυτό είναι ιδιαίτερα κρίσιμο στον τομέα της μουσικής, καθώς η ακρόαση μουσικής είναι εγγενώς μια ακολουθιακή διαδικασία: οι χρήστες συνήθως ακούν μουσική από λίστες αναπαραγωγής, άλμπουμ ή σε συνεδρίες ακρόασης [95].

Για να αποτυπωθούν αυτές οι χρονικές εξαρτήσεις, οι μέθοδοι διαδοχικών συστάσεων (sequential recommendation) επεκτείνουν το παραδοσιακό πλαίσιο της συνεργατικής διήθησης, μοντελοποιώντας τη δραστηριότητα του χρήστη ως μια διατεταγμένη ακολουθία αλληλεπιδράσεων. Σε αντίθεση με τα παραδοσιακά συστήματα, τα διαδοχικά μοντέλα μαθαίνουν πώς οι πρόσφατες επιλογές επηρεάζουν τις επόμενες, επιδιώκοντας να προβλέψουν το επόμενο αντικείμενο με το οποίο είναι πιθανό να αλληλεπιδράσει ο χρήστης, βάσει του πρόσφατου ιστορικού του.

Η σύσταση βασισμένη σε συνεδρίες (session-based recommendation) αποτελεί ειδική περίπτωση της παραπάνω κατηγορίας, όπου οι προβλέψεις εξαρτώνται αποκλειστικά από την τρέχουσα συνεδρία του χρήστη, χωρίς να απαιτούνται μακροχρόνια ιστορικά δεδομένα. Αυτή η περίπτωση είναι ιδιαίτερα χρήσιμη για ανώνυμους ή νέους χρήστες, καθώς επιτρέπει προσωποποιημένες συστάσεις ακόμη και χωρίς την ύπαρξη ενός ολοκληρωμένου προφίλ χρήστη. Συνήθη παραδείγματα αποτελούν οι συνεδρίες ακρόασης μουσικής ή οι συνεδρίες περιήγησης σε ηλεκτρονικά καταστήματα, όπου το βραχυπρόθεσμο ιστορικό διαδραματίζει κυρίαρχο ρόλο. Επειδή οι παραπάνω

κατηγορίες συστάσεων συνιστούν ουσιαστικά προβλήματα μοντελοποίησης ακολουθιών, δεν προκαλεί έκπληξη ότι την τελευταία δεκαετία έχουν αξιοποιηθεί με μεγάλη επιτυχία οι ίδιες αρχιτεκτονικές που χρησιμοποιούνται στην Επεξεργασία Φυσικής Γλώσσας (NLP) για μοντελοποίηση γλώσσας, όπως οι GRU [20] και οι Transformers [109], καθώς και οι παραλλαγές τους που προσαρμόστηκαν ειδικά για συστάσεις [44], [45], [53], [103].

Στον χώρο της μουσικής, μια άμεσα σχετική εφαρμογή είναι η Αυτόματη Συνέχιση Λίστας Αναπαραγωγής (Automatic Playlist Continuation (APC)). Αναμφίβολα, οι λίστες αναπαραγωγής έχουν καταστεί ιδιαίτερα δημοφιλής τρόπος κατανάλωσης μουσικής τα τελευταία χρόνια, και οι χρήστες στις ψηφιακές πλατφόρμες συχνά δημιουργούν τις δικές τους λίστες βάσει αγαπημένων ειδών, συγκεκριμένης διάθεσης ή περίστασης ακρόασης. Επομένως, μία από τις προκλήσεις που καλούνται να αντιμετωπίσουν οι πλατφόρμες αυτές είναι η αυτόματη επέκταση τέτοιων λιστών με τρόπο που να διατηρεί τη μουσική συνοχή και να ταιριάζει με το ύφος των υπόλοιπων κομματιών.

Διατύπωση Προβλήματος και Ερευνητικά Ερωτήματα

Όπως αναφέρθηκε προηγουμένως, τα προεκπαιδευμένα μοντέλα έχουν αποκτήσει μεγάλη δημοφιλία στην κοινότητα του MIR, χάρη στην υψηλή τους απόδοση σε διάφορα προβλήματα εξαγωγής μουσικής πληροφορίας. Ωστόσο, η αποτελεσματικότητα τέτοιων μοντέλων στη σύσταση μουσικής με βάση συνεδρίες παραμένει σχετικά ανεξερεύνητη [104]. Το βασικό πρόβλημα που πραγματεύεται η παρούσα εργασία είναι κατά πόσο οι ηχητικές αναπαραστάσεις από προεκπαιδευμένα μοντέλα μπορούν να βελτιώσουν την απόδοση συστημάτων σύστασης μουσικής που βασίζονται σε συνεδρίες ακρόασης. Πιο συγκεκριμένα, η έρευνά μας στοχεύει στην εξαγωγή αναπαραστάσεων ήχου από πρόσφατα μοντέλα στο πεδίο του MIR και στη διερεύνηση διαφορετικών τρόπων ενσωμάτωσής τους στο σύστημα παραγωγής συστάσεων.

Για τον σκοπό αυτό, πειραματιζόμαστε με δύο διαφορετικά μοντέλα συστάσεων: τη βιβλιοθήκη Transformers4Rec (T4Rec) [100] της NVIDIA, και το μοντέλο Represent-Then-Aggregate (RTA) [7], που προτάθηκε πρόσφατα από τη Deezer Research. Οι αρχιτεκτονικές διαφορές μεταξύ των δύο μας επιτρέπουν να αξιολογήσουμε την αποτελεσματικότητα της προσέγγισής μας σε διαφορετικά σενάρια.

Συνολικά, η εργασία επιχειρεί να απαντήσει στα ακόλουθα ερευνητικά ερωτήματα:

- RQ1: Μπορούν οι προεκπαιδευμένες ηχητικές αναπαραστάσεις να βελτιώσουν την απόδοση συστημάτων συστάσεων που βασίζονται σε συνεδρίες;
- RQ2: Ποια τεχνική προεκπαίδευσης αποδίδει καλύτερα σε αυτό το πλαίσιο;
- RQ3: Ποιος είναι ο αποτελεσματικότερος τρόπος ενσωμάτωσης των προεκπαιδευμένων αναπαραστάσεων στο μοντέλο παραγωγής συστάσεων;

Συνεισφορές

Οι συνεισφορές της παρούσας διπλωματικής εργασίας στο πεδίο των διαδοχικών συστάσεων μουσικής συνοψίζονται ως εξής:

- Αξιολογούμε την απόδοση των προεκπαιδευμένων ηχητικών αναπαραστάσεων, οι οποίες χρησιμοποιούνται σε συνδυασμό με δύο συστήματα συστάσεων, τα T4Rec [100] και RTA [7], τα οποία προσαρμόζουμε στο πρόβλημα της σύστασης μουσικής με βάση συνεδρίες ακρόασης.
- Συγκρίνουμε τα ακόλουθα μοντέλα ως πηγές αναπαραστάσεων: MERT [63], MusiCNN [80] και ένα δικό μας μοντέλο βασισμένο στην ομοιότητα καλλιτεχνών. Εξερευνούμε επίσης μεθόδους προσαρμογής (fine-tuning) μέσω αντιθετικής μάθησης πάνω στα προεκπαιδευμένα embeddings ως τεχνική προσαρμογής πεδίου (domain adaptation).
- Επεξεργαζόμαστε το σύνολο δεδομένων Music4All [91], το οποίο περιλαμβάνει ιστορικά ακροάσεων περίπου 15.000 χρηστών της πλατφόρμας Last.fm[1], προκειμένου να χωρίσουμε τα ιστορικά σε συνεδρίες ακρόασης. Παράλληλα, χρησιμοποιούμε τα προεκπαιδευμένα μοντέλα για την εξαγωγή αναπαραστάσεων τραγουδιών απευθείας από το ηχητικό σήμα.

[1] https://www.last.fm/

- Διερευνούμε τόσο την αρχικοποίηση των αναπαραστάσεων κάθε τραγουδιού με τα προεκπαιδευμένα embeddings όσο και τη συνδυαστική χρήση τους πριν την τελική παραγωγή των μουσικών προτάσεων (late fusion).

Θεωρητικό Υπόβαθρο

Μηχανική Μάθηση

Η μηχανική μάθηση (Machine Learning) αποτελεί κλάδο της τεχνητής νοημοσύνης που αναπτύσσει μοντέλα ικανά να μαθαίνουν πρότυπα από δεδομένα, ώστε να πραγματοποιούν προβλέψεις για νέα δείγματα. Οι μέθοδοι μηχανικής μάθησης διακρίνονται σε τρεις βασικές κατηγορίες: (α) την επιβλεπόμενη μάθηση, όπου το μοντέλο εκπαιδεύεται με δεδομένα για τα οποία η επισήμανση κάθε δείγματος είναι γνωστή, και μαθαίνει τη συνάρτηση που συσχετίζει είσοδο και έξοδο, (β) τη μη επιβλεπόμενη μάθηση, όπου το σύστημα επιχειρεί να ανακαλύψει εσωτερικές δομές και συσχετίσεις χωρίς επισημασμένα δεδομένα, όπως σε προβλήματα συσταδοποίησης ή μείωσης διαστάσεων, και (γ) την ενισχυτική μάθηση, όπου ένας πράκτορας αλληλεπιδρά με το περιβάλλον, επιλέγοντας ενέργειες με στόχο τη μεγιστοποίηση μιας αθροιστικής ανταμοιβής. Μια ενδιάμεση και ιδιαίτερα σημαντική κατηγορία είναι η αυτο-επιβλεπόμενη μάθηση (Self-Supervised Learning), στην οποία το ίδιο το μοντέλο παράγει τα σήματα επίβλεψης—για παράδειγμα αποκρύπτοντας τμήματα της εισόδου και μαθαίνοντας την πρόβλεψή τους. Οι προσεγγίσεις αυτές έχουν καθιερωθεί τα τελευταία χρόνια, προσφέροντας αποτελεσματικούς τρόπους εκπαίδευσης χωρίς την ανάγκη μεγάλου όγκου επισημασμένων δεδομένων.

Εκμάθηση Αναπαραστάσεων (Representation Learning)

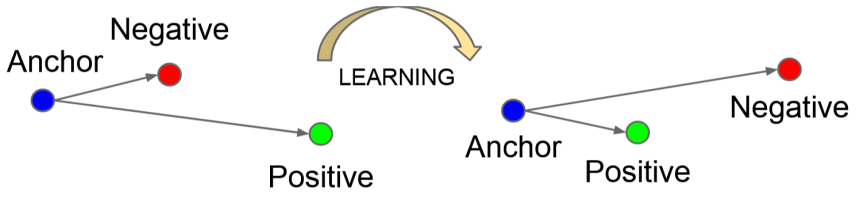
Η εκμάθηση αναπαραστάσεων αποσκοπεί στην αποτύπωση των δεδομένων σε έναν διανυσματικό χώρο που ενσωματώνει κάποια πρότερη γνώση που αφορά τα δεδομένα. Οι τεχνικές αυτές εκπαιδεύουν νευρωνικά δίκτυα να μαθαίνουν αναπαραστάσεις οι οποίες μπορούν ύστερα να αξιοποιηθούν σε διάφορα προβλήματα, όπως η ταξινόμηση ή η ανάκτηση.

Μετρική Μάθηση (Metric Learning)

Η μετρική μάθηση στοχεύει στην εκμάθηση μιας συνάρτησης απόστασης προσαρμοσμένης στο εκάστοτε πρόβλημα. Αντί για τη χρήση μιας τυπικής μετρικής όπως η Ευκλείδεια απόσταση, το μοντέλο χτίζει έναν διανυσματικό χώρο όπου παρόμοια δείγματα τοποθετούνται κοντά και ανόμοια μακριά, όπως απεικονίζεται στο Σχήμα 0.0.1. Η πιο κοινή διατύπωση είναι η συνάρτηση κόστους τριπλέτας (triplet loss), η οποία λαμβάνει ένα δείγμα αναφοράς (anchor) A, ένα θετικό δείγμα P (παρόμοιο με το anchor) και ένα αρνητικό δείγμα N (ανόμοιο). Ορίζεται ως:

$$\mathcal{L}_{\text{triplet}} = \max\left(\|z^A - z^P\|_2^2 - \|z^A - z^N\|_2^2 + \alpha, 0\right) \tag{0.0.1}$$

όπου z^A, z^P, z^N είναι τα embeddings των δειγμάτων και α είναι ένα περιθώριο που επιβάλλει ελάχιστη απόσταση ανάμεσα στα θετικά και τα αρνητικά ζεύγη. Η συνάρτηση κόστους ελαχιστοποιείται όταν το θετικό δείγμα βρίσκεται πιο κοντά στο anchor από το αρνητικό κατά τουλάχιστον α. Η επιλογή κατάλληλων τριάδων είναι κρίσιμη για την αποδοτικότητα της εκπαίδευσης, καθώς επιταχύνει τη σύγκλιση.



Σχήμα 0.0.1: Η συνάρτηση κόστους τριπλέτας φέρνει πιο κοντά τα παρόμοια δείγματα και απομακρύνει τα ανόμοια [97].

Αντιθετική Μάθηση (Contrastive Learning)

Η αντιθετική μάθηση (contrastive learning) μπορεί να θεωρηθεί ως ειδική περίπτωση της μετρικής μάθησης. Στόχος της είναι η αναγνώριση ζευγών δειγμάτων που σχετίζονται σημασιολογικά (θετικά ζεύγη) μεταξύ πολλών αρνητικών. Το θετικό ζεύγος μπορεί να προκύψει είτε από διαφορετικούς μετασχηματισμούς του ίδιου δείγματος [19], [101], είτε από αντιστοίχιση διαφορετικών μορφών δεδομένων, όπως ήχου και των αντίστοιχων περιγραφών σε φυσική γλώσσα [66]. Τα αρνητικά δείγματα προέρχονται συνήθως από τα υπόλοιπα στοιχεία της παρτίδας εκπαίδευσης (batch).

Μία ευρέως χρησιμοποιούμενη συνάρτηση κόστους είναι η InfoNCE [74], η οποία ενισχύει την ομοιότητα ανάμεσα στο δείγμα και το θετικό του, ενώ απομακρύνει τα αρνητικά:

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{\exp(\text{sim}(z, z^+)/\tau)}{\sum_i \exp(\text{sim}(z, z_i^-)/\tau) + \exp(\text{sim}(z, z^+)/\tau)} \tag{0.0.2}$$

όπου $\text{sim}(\cdot, \cdot)$ είναι μέτρο ομοιότητας (π.χ. ομοιότητα συνημιτόνου) και τ η παράμετρος θερμοκρασίας. Με τον τρόπο αυτό, ο χώρος αναπαραστάσεων οργανώνεται ώστε να αποτυπώνει τις σημασιολογικές σχέσεις των δεδομένων. Στην πράξη, τόσο η triplet loss όσο και η InfoNCE χρησιμοποιούνται ευρέως σε εφαρμογές όπως αναγνώριση προσώπων [97], ανάκτηση εικόνων [19] και εκμάθηση μουσικών αναπαραστάσεων [101].

Μετασχηματιστές (Transformers)

Η αρχιτεκτονική Transformer [109] είναι ένα είδος νευρωνικού δικτύου που εισήχθη το 2017 για το πρόβλημα της αυτόματης μετάφρασης και έκτοτε άλλαξε ριζικά τον τρόπο αντιμετώπισης ακολουθιακών δεδομένων. Αντί να επεξεργάζεται τις εισόδους σειριακά όπως τα αναδρομικά νευρωνικά δίκτυα (RNNs), χρησιμοποιεί έναν πλήρως παραλληλοποιήσιμο μηχανισμό αυτοπροσοχής (self-attention), που επιτρέπει την αποτελεσματική μοντελοποίηση εξαρτήσεων μεγάλης εμβέλειας. Συγκεκριμένα, κάθε στοιχείο της ακολουθίας μπορεί να εστιάζει σε όλα τα υπόλοιπα, βελτιώνοντας την αναπαράστασή του με τρόπο που λαμβάνει υπόψη τα συμφραζόμενα. Το αρχικό μοντέλο Transformer βασίζεται σε μια δομή κωδικοποιητή-αποκωδικοποιητή (encoder–decoder): ο κωδικοποιητής δημιουργεί αναπαραστάσεις της εισόδου και ο αποκωδικοποιητής τις χρησιμοποιεί για την παραγωγή της εξόδου. Το καθένα από αυτά τα τμήματα μπορεί να αποτελείται από πολλά πανομοιότυπα επίπεδα, όπου κάθε επίπεδο περιλαμβάνει αυτοπροσοχή πολλών κεφαλών (multi-head attention) και δίκτυα πρόσθιας τροφοδότησης που εφαρμόζονται ξεχωριστά σε κάθε στοιχείο, ενώ επιπλέον εφαρμόζονται υπολειμματικές συνδέσεις (residual connections), κανονικοποίηση και διανύσματα θέσης για κωδικοποίηση της διάταξης.

Μεταγενέστερες παραλλαγές της αρχιτεκτονικής Transformer οδήγησαν στην ανάπτυξη εξειδικευμένων μοντέλων που αποτελούνται αποκλειστικά από κωδικοποιητή ή αποκωδικοποιητή, ανάλογα με τον εκάστοτε στόχο. Τα πρώτα, όπως το BERT [28], είναι κατάλληλα για εργασίες όπως η ταξινόμηση, όπου απαιτείται κατανόηση του νοήματος της εισόδου. Αντίθετα, τα μοντέλα με αποκωδικοποιητή, όπως το GPT [84], υπερέχουν σε παραγωγικά προβλήματα, όπως η συμπλήρωση κειμένου ή η παραγωγή διαλόγου ανοιχτής μορφής. Τέλος, τα μοντέλα κωδικοποιητή-αποκωδικοποιητή, όπως ο αρχικός Transformer, χρησιμοποιούνται για σκοπούς μετασχηματισμού κειμένου, όπως η μηχανική μετάφραση, η περίληψη ή η παράφραση.

Μετρικές Κατάταξης (Ranking Metrics)

Στα συστήματα συστάσεων, οι μετρικές αξιολόγησης ποσοτικοποιούν το πόσο καλά οι προβλέψεις του μοντέλου αντιστοιχούν στην πραγματικότητα ή, με άλλα λόγια, το πόσο καλά οι προτεινόμενες επιλογές ταιριάζουν με εκείνες που ο χρήστης πράγματι προτιμά. Ορισμένες από αυτές τις μετρικές χρησιμοποιούνται κυρίως σε προβλήματα ταξινόμησης και αντιμετωπίζουν τη σύσταση ως ένα δυαδικό πρόβλημα πρόβλεψης (σχετικό ή μη σχετικό αντικείμενο). Μια άλλη κατηγορία, οι λεγόμενες μετρικές κατάταξης, αξιολογούν την ικανότητα του συστήματος να τοποθετεί τα σχετικά αντικείμενα ψηλότερα στη λίστα συστάσεων—καθώς εκεί θα θέλαμε να βρίσκονται οι πιο σωστές προτάσεις.

Recall@k

Η μετρική Recall@k μετρά το ποσοστό των συνολικών σχετικών αντικειμένων που εμφανίζονται στις πρώτες k θέσεις της ταξινομημένης λίστας:

$$\text{Recall@k} = \frac{\#\text{ σχετικών αντικειμένων στις πρώτες } k \text{ θέσεις}}{\#\text{ συνολικών σχετικών αντικειμένων στο ground truth}} \tag{0.0.3}$$

Αν και δεν λαμβάνει υπόψη τη σειρά εμφάνισης των σχετικών αντικειμένων εντός του top-k, αποτυπώνει πόσο καλά το σύστημα καλύπτει το επιθυμητό περιεχόμενο στο πιο ορατό τμήμα της λίστας.

MAP@k

Η μετρική MAP@k (Mean Average Precision at k) αξιολογεί επιπλέον τη σειρά κατάταξης. Για τον υπολογισμό της, αρχικά λαμβάνεται ο μέσος όρος της ακρίβειας στις θέσεις όπου εμφανίζονται τα σχετικά αντικείμενα, έως το k:

$$\text{AP@k} = \frac{1}{\min(k, R)} \sum_{i=1}^{k} P(i) \cdot \text{rel}(i) \tag{0.0.4}$$

όπου $P(i)$ είναι η ακρίβεια στη θέση i, $\text{rel}(i) \in \{0, 1\}$ δείχνει εάν το αντικείμενο είναι σχετικό, και R είναι ο αριθμός των σχετικών αντικειμένων. Ο μέσος όρος της AP@k για όλα τα δείγματα δίνει το MAP@k, το οποίο επιβραβεύει συστήματα που τοποθετούν τα σχετικά αντικείμενα νωρίτερα στη λίστα, σε αντίθεση με το Recall@k.

NDCG@k

Αντίστοιχα, η μετρική NDCG@k (Normalized Discounted Cumulative Gain) τιμωρεί τα σχετικά αντικείμενα που εμφανίζονται χαμηλότερα στην κατάταξη, διαιρώντας με τον λογάριθμο της θέσης i που εμφανίζονται:

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel}(i)} - 1}{\log_2(i+1)} \tag{0.0.5}$$

όπου $\text{rel}(i)$ είναι η βαθμολογία συνάφειας του αντικειμένου στη θέση i, η οποία μπορεί να είναι δυαδική ή αριθμητική (π.χ. 1–5). Η κανονικοποιημένη εκδοχή της μετρικής υπολογίζεται ως:

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}} \tag{0.0.6}$$

Σχετική βιβλιογραφία

Εκμάθηση Αναπαραστάσεων από Ηχητικά Σήματα

Σε αυτή την ενότητα εξετάζουμε μοντέλα που μαθαίνουν να αναπαριστούν τα ηχητικά σήματα σε έναν διανυσματικό χώρο, με τρόπο που να διευκολύνει την ανάκτηση παρόμοιων κομματιών και να αποτυπώνει τις δομικές ιδιότητες της μουσικής. Οι υπάρχουσες προσεγγίσεις ακολουθούν είτε επιβλεπόμενες μεθόδους, αξιοποιώντας κάποια μορφή προϋπάρχουσας γνώσης για τη μουσική ομοιότητα (π.χ. ετικέτες, στατιστικά συν-ακρόασης), είτε αυτο-επιβλεπόμενες τεχνικές που εκμεταλλεύονται συλλογές μη επισημασμένων δεδομένων και εκπαιδεύουν μοντέλα με ετικέτες που εξάγονται από τα ίδια τα δεδομένα.

Επιβλεπόμενες Μέθοδοι

Οι επιβλεπόμενες προσεγγίσεις εκμάθησης μουσικών αναπαραστάσεων χαρακτηρίζονται από τη χρήση εξωτερικά παρεχόμενων ετικετών ή σχολιασμών που καθοδηγούν την εκπαίδευση του μοντέλου. Τα σήματα εποπτείας μπορεί να περιλαμβάνουν δεδομένα αλληλεπίδρασης χρηστών, σημασιολογικές ετικέτες, πληροφορίες είδους ή διάθεσης, καθώς και άλλα επιμελημένα μεταδεδομένα. Με τη βελτιστοποίηση των αναπαραστάσεων ώστε να προβλέπουν ή να ευθυγραμμίζονται με τέτοιους ανθρωπογενείς στόχους, τα μοντέλα κατορθώνουν να συλλαμβάνουν μουσικά ουσιώδεις ιδιότητες που αντιστοιχούν σε καθιερωμένες κατηγορίες αντίληψης και χρήσης.

Ενδεικτικά, οι Pons και Serra πρότειναν το MusiCNN [80], μια συλλογή συνελικτικών δικτύων (CNNs) που εκπαιδεύονται με επιβλεπόμενο τρόπο για την πρόβλεψη ετικετών μουσικής. Το MusiCNN δέχεται ως είσοδο λογαριθμικά mel-φασματογραφήματα 3 δευτερολέπτων και εξάγει την πιθανότητα εμφάνισης κάθε μιας από τις 50 προκαθορισμένες ετικέτες. Συγκεκριμένα, το αρχικό συνελικτικό επίπεδο εφαρμόζει ομάδες κάθετων και οριζόντιων φίλτρων διαφορετικών μεγεθών ώστε να αποτυπώνει φασματικά και χρονικά χαρακτηριστικά, αντίστοιχα. Τα εξαγόμενα χαρακτηριστικά διοχετεύονται σε τρία συνελικτικά επίπεδα που καταγράφουν υψηλότερου επιπέδου πληροφορίες. Αυτά τα επίπεδα περιλαμβάνουν υπολειμματικές συνδέσεις που συμβάλλουν στη σταθεροποίηση της εκπαίδευσης. Τέλος, τα νέα χαρακτηριστικά υποβάλλονται σε max-pooling και mean-pooling στο πεδίο του χρόνου ώστε να παραχθούν δύο διανύσματα χαρακτηριστικών, τα οποία ενώνονται και προβάλλονται στη διάσταση του τελικού embedding (200 για το βασικό μοντέλο και 500 για το μεγαλύτερο), πριν περάσουν από ένα πλήρως συνδεδεμένο επίπεδο που παράγει τα logits για τις 50 κατηγορίες ετικετών. Οι συγγραφείς απέδειξαν μεταξύ άλλων την αποτελεσματικότητα του MusiCNN σε σενάρια μεταφοράς μάθησης, εξάγοντας τα χαρακτηριστικά MusiCNN και εκπαιδεύοντας έναν απλό ταξινομητή για ταξινόμηση είδους επιτυγχάνοντας ακρίβεια συγκρίσιμη με μοντέλα εκπαιδευμένα σε πολύ μεγαλύτερα σύνολα δεδομένων.

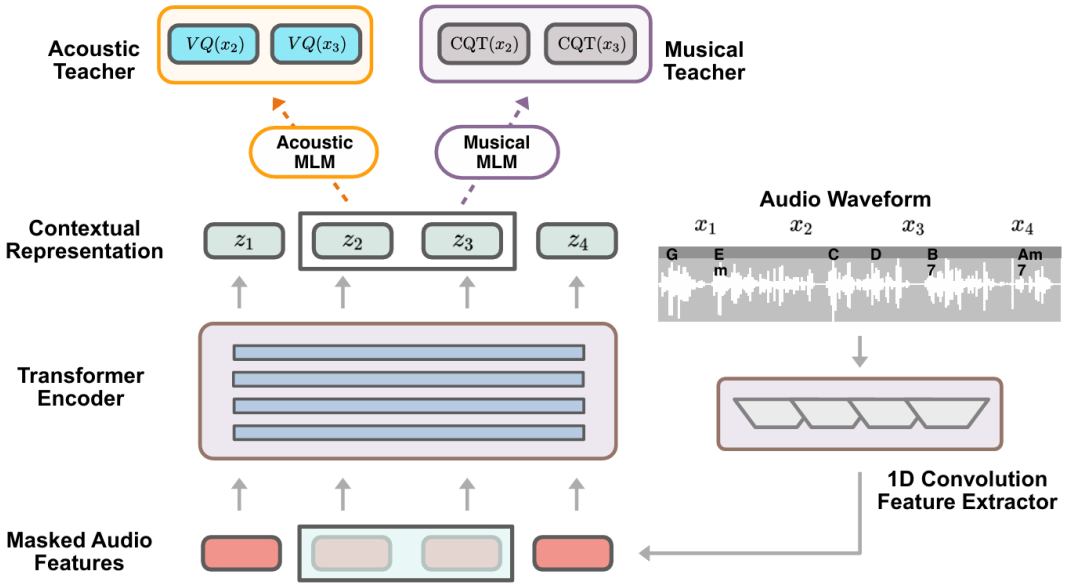
Αυτοεπιβλεπόμενες Μέθοδοι

Παρότι οι επιβλεπόμενες προσεγγίσεις έχουν επιτύχει υψηλή απόδοση σε πολλά προβλήματα, βασίζονται σε επιμελημένες ετικέτες όπως tags ή σχολιασμούς, των οποίων η συλλογή σε μεγάλη κλίμακα είναι δαπανηρή και συχνά θορυβώδης ή υποκειμενική. Ο περιορισμός αυτός οδήγησε στη στροφή προς την αυτοεπιβλεπόμενη μάθηση (Self-Supervised Learning, SSL), όπου τα μοντέλα εκπαιδεύονται σε μεγάλες συλλογές μη επισημασμένων ηχητικών δεδομένων, επιλύοντας προκαταρκτικά προβλήματα (pretext tasks) με ετικέτες που προκύπτουν από τα ίδια τα δεδομένα. Σε γενικές γραμμές, διακρίνονται δύο κύρια παραδείγματα: η αντιθετική μάθηση (contrastive learning), η οποία αντιμετωπίζει τροποποιημένες εκδοχές του ίδιου σήματος ως θετικά ζεύγη και άλλα δείγματα ως αρνητικά [19], [37], [101], και η προβλεπτική μοντελοποίηση (predictive/masked modeling), όπου το μοντέλο μαθαίνει να προβλέπει κρυμμένα ή ελλείποντα τμήματα της εισόδου [3], [4], [28], [47], [62], [63], με βάση τα διαθέσιμα συμφραζόμενα. Αξιοποιώντας τεράστιες ποσότητες μη επισημασμένων ηχητικών σημάτων, οι μέθοδοι αυτές επιτρέπουν την εκπαίδευση γενικών αναπαραστάσεων μουσικής που αποδίδουν αποτελεσματικά σε ποικίλα προβλήματα MIR, όπως πρόβλεψη ετικετών, ταξινόμηση και ανάκτηση.

Ένα επιτυχημένο παράδειγμα αυτής της κατηγορίας είναι το MERT [63], ένα γενικού σκοπού προεκπαιδευμένο μοντέλο για κατανόηση μουσικής, σημαντικά μικρότερο σε μέγεθος από άλλα αντίστοιχα μοντέλα όπως το JukeMIR [14]: 95M παράμετροι για το βασικό μοντέλο και 330M για το μεγάλο. Το MERT υιοθετεί την αρχιτεκτονική του HuBERT [47], με βασική διαφορά ότι αντί για ομαδοποιημένα MFCCs ως ground truth, χρησιμοποιεί χαρακτηριστικά σχεδιασμένα να συλλαμβάνουν την πολυπλοκότητα της μουσικής (Σχήμα 0.0.2). Συγκεκριμένα, όπως και στο HuBERT, το μοντέλο αρχικά κωδικοποιεί ηχητικά αποσπάσματα 5 δευτερολέπτων με μονοδιάστατες συνελίξεις και στη συνέχεια προωθεί μια μασκαρισμένη εκδοχή της ακολουθίας σε έναν Transformer που αποτυπώνει τις χρονικές εξαρτήσεις. Ιδιαίτερα σημαντικό είναι ότι, για να ενθαρρύνει την ακριβή διακριτή αναπαράσταση του μουσικού σήματος σε ακουστικό επίπεδο, οι στόχοι της συνάρτησης κόστους παρέχονται από έναν Residual Vector Quantisation–Variational AutoEncoder (RVQ-VAE) [26], έναν αυτοκωδικοποιητή που συμπιέζει τον ήχο σε διακριτές κωδικοποιημένες ακολουθίες και λειτουργεί ως «ακουστικός δάσκαλος». Εναλλακτικά, οι συγγραφείς διερεύνησαν και μοντελοποίηση του ήχου μέσω ομαδοποίησης (k-means) log-Mel φασματογραφημάτων και χαρακτηριστικών Chroma, η οποία ωστόσο παρουσιάζει ζητήματα κλιμάκωσης. Επιπλέον, εισάγεται ένας πρόσθετος όρος κόστους ανακατασκευής που λειτουργεί ως «μουσικός δάσκαλος», συγκρίνοντας την έξοδο του Transformer με το φασματογράφημα Constant-Q Transform (CQT) ώστε να αποτυπώνονται πληροφορίες τονικότητας και αρμονίας.

Διαδοχικές Συστάσεις (Sequential Recommendation)

Οι ακολουθίες αλληλεπιδράσεων χρηστών, όπως οι λίστες αναπαραγωγής ή τα ιστορικά ακροάσεων, μπορούν να θεωρηθούν φυσικά ως διατεταγμένες ακολουθίες αντικειμένων—παρόμοια με τις λέξεις σε μια πρόταση. Αυτή η αναλογία οδήγησε στην υιοθέτηση τεχνικών μοντελοποίησης ακολουθιών από το πεδίο της επεξεργασίας φυσικής γλώσσας (NLP) στον χώρο των συστάσεων. Πρώιμες εργασίες έδειξαν την αποτελεσματικότητα των αναδρομικών νευρωνικών δικτύων (RNNs) στην αποτύπωση βραχυπρόθεσμων και μεσοπρόθεσμων εξαρτήσεων στις συνεδρίες χρηστών [44], [45], ενώ μεταγενέστερα μοντέλα βασισμένα στην αυτοπροσοχή και την αρχιτεκτονική Transformer επέκτειναν αυτές τις ιδέες ώστε να χειρίζονται εξαρτήσεις μεγαλύτερης εμβέλειας [53], [100],



Σχήμα 0.0.2: Η διαδικασία προεκπαίδευσης που ακολουθήθηκε στο MERT [63]

[103]. Πιο πρόσφατα, η ραγδαία ανάπτυξη των μεγάλων προεκπαιδευμένων γλωσσικών μοντέλων (LLMs) έχει προκαλέσει ενδιαφέρον σχετικά με τον πιθανό τους ρόλο στις διαδοχικές συστάσεις [82].

Το Transformers4Rec των Moreira et al. [100] αποτελεί χαρακτηριστικό παράδειγμα της ενσωμάτωσης καινοτομιών του NLP στον τομέα των συστάσεων. Πρόκειται για μια βιβλιοθήκη ανοιχτού κώδικα, βασισμένη στη συλλογή Transformers της HuggingFace [113], που διευκολύνει τη χρήση σύγχρονων αρχιτεκτονικών Transformer για διαδοχικές και session-based συστάσεις. Από άποψη αρχιτεκτονικής, το Transformers4Rec υποστηρίζει πληθώρα μοντέλων Transformer (όπως GPT-2 [85], Transformer-XL [24], XLNet [116]) και εισάγει πρόσθετες λειτουργικότητες απαραίτητες για εφαρμογές συστάσεων, όπως κανονικοποίηση και συνάθροιση χαρακτηριστικών εισόδου, σταδιακή εκπαίδευση και αξιολόγηση και υπολογισμό δημοφιλών μετρικών κατάταξης. Πιο συγκεκριμένα, η βιβλιοθήκη παρέχει μηχανισμούς ενσωμάτωσης επιπρόσθετων πληροφοριών, όπως κατηγορικά χαρακτηριστικά που αφορούν το αντικείμενο (μουσικό είδος, καλλιτέχνη κ.ά.) ή τον χρήστη (περιοχή, τύπο συσκευής), τα οποία αναπαρίστανται μέσω ξεχωριστών πινάκων embeddings, όπως επίσης και συνεχών χαρακτηριστικών (π.χ. τιμή). Όλες αυτές οι πληροφορίες μπορούν να ενσωματωθούν ομαλά στην είσοδο του μοντέλου, για παράδειγμα με συνένωση με το embedding του αντικειμένου ή με πολλαπλασιασμό ανά στοιχείο.

Επιπλέον, το Transformers4Rec υποστηρίζει εκπαίδευση με διαφορετικούς μαθησιακούς στόχους:

- Causal Language Modeling (CLM): πρόβλεψη του επόμενου αντικειμένου με βάση τα προηγούμενα, όπως στο SASRec [53].
- Masked Language Modeling (MLM): τυχαία απόκρυψη αντικειμένων και ανακατασκευή τους, όπως στο BERT4Rec[103].
- Permutation Language Modeling (PLM): πρόβλεψη αντικειμένων με βάση τυχαίες αντιμεταθέσεις, όπως στο XLNet [116].
- Replacement Token Detection (RTD): διάκριση πραγματικών και αντικατεστημένων αντικειμένων, όπως στο ELECTRA [21].

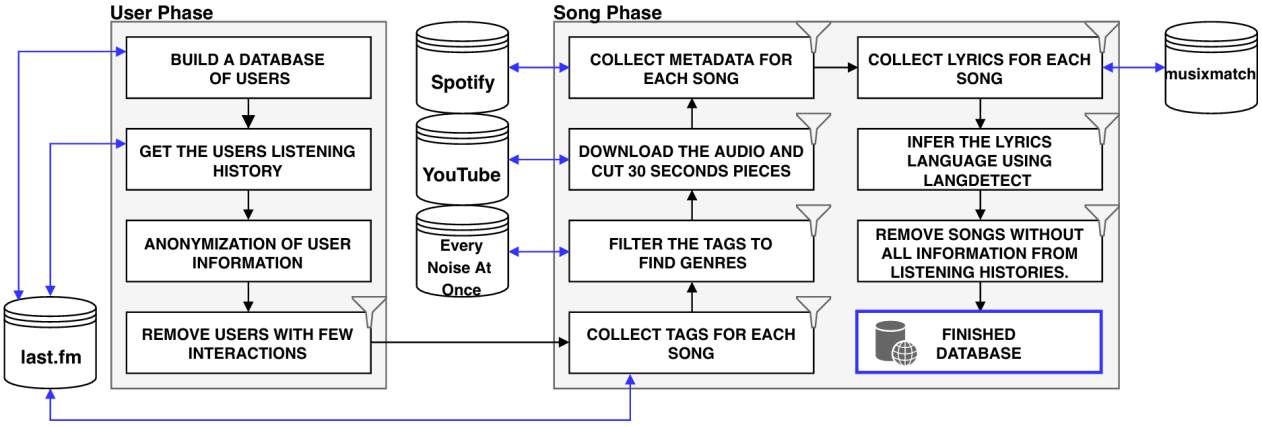
Κατά την εκπαίδευση, η ακολουθία εισόδου τροποποιείται ανάλογα με τον εκάστοτε στόχο και τροφοδοτείται σε μια διαμορφώσιμη στοίβα από επίπεδα Transformer, η οποία εμπλουτίζει τις αναπαραστάσεις των αντικειμένων με πληροφορία που αφορά τα συμφραζόμενα. Στη συνέχεια, μία κεφαλή πρόβλεψης (prediction head) εκτελεί τη ζητούμενη εργασία, είτε προβλέποντας το επόμενο αντικείμενο στη σειρά, είτε επιλύοντας ένα πρόβλημα ταξινόμησης ή παλινδρόμησης που αφορά ολόκληρη την ακολουθία, όπου οι αναπαραστάσεις των αντικειμένων συγχωνεύονται σε ένα ενιαίο διάνυσμα ακολουθίας. Πειραματικές μελέτες σε πολλαπλά σύνολα δεδομένων ηλεκ-

τρονικού εμπορίου και ειδήσεων κατέδειξαν ότι οι προτεινόμενες αρχιτεκτονικές Transformer υπερείχαν σταθερά έναντι των κλασικών RNN και μη νευρωνικών μεθόδων. Επιπλέον, το Transformers4Rec χρησιμοποιήθηκε επιτυχώς σε ρεαλιστικές εφαρμογές, κερδίζοντας δύο σχετικούς διαγωνισμούς [71], [96].

Διαδοχικές Συστάσεις Μουσικής (Sequential Music Recommendation)

Το Σύνολο Δεδομένων Music4All

Το σύνολο δεδομένων Music4All [91], που δημιουργήθηκε το 2020, περιλαμβάνει ιστορικά ακροάσεων χρηστών του Last.fm[2] και, το σημαντικότερο, συνοδεύεται από ηχητικά αποσπάσματα για κάθε τραγούδι. Δημιουργήθηκε σε δύο φάσεις, όπως φαίνεται στο Σχήμα 0.0.3: μία φάση συλλογής δεδομένων χρηστών (user phase), όπου καταγράφηκαν τα ιστορικά ακροάσεων, και μία φάση συλλογής δεδομένων τραγουδιών (song phase), όπου συγκεντρώθηκαν τα μεταδεδομένα και απορρίφθηκαν κομμάτια με ελλιπείς πληροφορίες. Πιο συγκεκριμένα, το σύνολο περιλαμβάνει χρονοσημασμένα ιστορικά ακροάσεων από 15.602 χρήστες του Last.fm, καταγεγραμμένα το διάστημα από 1 Ιανουαρίου έως 20 Μαρτίου 2019. Κάθε χρήστης έχει κατά μέσο όρο 361 γεγονότα ακρόασης, που αντιστοιχούν περίπου σε 200 διαφορετικά τραγούδια. Μετά την ολοκλήρωση της φάσης συλλογής τραγουδιών, το Music4All περιείχε 109.269 μουσικά κομμάτια, εμπλουτισμένα με πληθώρα χαρακτηριστικών: αποσπάσματα 30 δευτερολέπτων από το μέσο του τραγουδιού, ακουστικά χαρακτηριστικά από το Spotify API (όπως danceability, valence και tempo), καθώς και λεπτομερή μεταδεδομένα που περιλαμβάνουν τίτλο τραγουδιού, καλλιτέχνη, άλμπουμ, ετικέτες είδους, ετικέτες χρηστών από το Last.fm και στίχους. Για να αναδείξουν τη χρησιμότητά του, οι δημιουργοί του χρησιμοποίησαν το Music4All σε διάφορα προβλήματα, όπως σύσταση μουσικής, ταξινόμηση είδους και ταξινόμηση διάθεσης.



Σχήμα 0.0.3: Διαδικασία δημιουργίας του συνόλου δεδομένων Music4All [91]

Μέθοδοι Βασισμένες σε Δεδομένα Αλληλεπιδράσεων

Ομοίως με ό,τι περιγράψαμε για τις διαδοχικές συστάσεις, το πρόβλημα της Αυτόματης Συνέχισης Λίστας Αναπαραγωγής (APC) αποτελεί ουσιαστικά ένα πρόβλημα μοντελοποίησης ακολουθιών. Συνεπώς, δεν προκαλεί έκπληξη ότι πολλές πρόσφατες εργασίες αξιοποιούν καθιερωμένες αρχιτεκτονικές μοντελοποίησης ακολουθιών για την αντιμετώπισή του. Ωστόσο, πολλές από αυτές παραβλέπουν τον παράγοντα της κλιμακωσιμότητας, ο οποίος είναι κρίσιμος για εφαρμογές στον πραγματικό κόσμο. Με βάση αυτή την παρατήρηση, οι Bendada et al. [7] πρότειναν ένα γενικού σκοπού πλαίσιο για το APC, το οποίο επιτυγχάνει ισορροπία μεταξύ κλιμακωσιμότητας και ευελιξίας, στοχεύοντας να καταστήσει προχωρημένα μοντέλα ακολουθιών (όπως RNNs και Transformers) πρακτικά αξιοποιήσιμα σε πραγματικά συστήματα μετάδοσης μουσικής. Η προσέγγισή τους, με την ονομασία Represent-Then-Aggregate (RTA), διαχωρίζει τη διαδικασία σύστασης σε δύο βασικά δομικά στοιχεία: μία συνάρτηση αναπαράστασης τραγουδιού (representer) και μία συνάρτηση συνάθροισης σε επίπεδο λίστας αναπαραγωγής (aggregator). Οι αναπαραστάσεις τραγουδιών μπορούν να εξαχθούν με διάφορους τρόπους—απευθείας από δεδομένα αλληλεπιδράσεων μέσω WRMF [48] ή με εφαρμογή μηχανισμού προσοχής ή απλού μέσου όρου σε embeddings μεταδεδομένων—και συμβολίζονται ως $h_s = \phi(s)$. Αυτές οι αναπαραστάσεις

[2] https://www.last.fm/

στη συνέχεια συμπτύσσονται από μια συνάρτηση g σε μία αναπαράσταση ακολουθίας h_p, χρησιμοποιώντας απλές μεθόδους όπως τον μέσο όρο ή πιο σύνθετα μοντέλα ακολουθιών, όπως gated CNNs [112], GRUs [20] ή Transformer decoders [109]. Η τελική βαθμολογία σύστασης υπολογίζεται μέσω του εσωτερικού γινομένου μεταξύ των embeddings της ακολουθίας και του τραγουδιού. Κατά την εκπαίδευση, τα δύο βασικά δομικά στοιχεία—ο representer και ο aggregator—βελτιστοποιούνται από κοινού με μία αντιθετική συνάρτηση κόστους, η οποία διακρίνει το πραγματικό επόμενο κομμάτι κάθε υπο-λίστας p από αρνητικά δείγματα που προέρχονται από τραγούδια εκτός της p. Το άρθρο τονίζει ότι η κλιμακωσιμότητα διατηρείται με την ελαχιστοποίηση των online πράξεων: μόνο οι αναπαραστάσεις των λιστών υπολογίζονται κατά την εξαγωγή (inference) από τα embeddings, τα οποία είναι υπολογισμένα εκ των προτέρων, μειώνοντας την καθυστέρηση απόκρισης.

Μέθοδοι Βασισμένες στο Ηχητικό Περιεχόμενο (Audio-Driven Methods)

Ενώ η προηγούμενη προσέγγιση εστιάζει στην αξιοποίηση των δεδομένων αλληλεπίδρασης για τη συνέχιση λιστών αναπαραγωγής, πιο πρόσφατες προσπάθειες διερευνούν την ενσωμάτωση του ηχητικού περιεχομένου στη διαδικασία σύστασης. Σε μία πρόσφατη εργασία, οι Tamm και Aljanaki [104] μελετούν τη συμπεριφορά προεκπαιδευμένων ηχητικών αναπαραστάσεων όταν συνδυάζονται με τα ιστορικά ακροάσεων των χρηστών. Για την αξιολόγηση, οι συγγραφείς χρησιμοποιούν το σύνολο δεδομένων Music4All-Onion [72] με χρονικό διαχωρισμό (ο τελευταίος μήνας για validation και testing, το προηγούμενο έτος για εκπαίδευση) και συγκρίνουν έξι διαφορετικές ηχητικές αναπαραστάσεις (MusiCNN [80], MERT [63], EncodecMAE [76], Music2Vec [62], MusicFM [115], Jukebox [29]). Η αξιολόγηση πραγματοποιείται με τις μετρικές HitRate@50, Recall@50 και NDCG@50, κατατάσσοντας μόνο τραγούδια που δεν έχουν ακουστεί προηγουμένως, με μοντέλο αναφοράς (baseline) ένα μοντέλο με τυχαία αρχικοποιημένα embeddings αντικειμένων. Τα αποτελέσματα δείχνουν ότι σε κάποιες περιπτώσεις ο συνδυασμός περιεχομένου και πληροφοριών αλληλεπίδρασης βελτιώνει σημαντικά την απόδοση, ενώ ταυτόχρονα επισημαίνεται μία ασυμφωνία μεταξύ των κατατάξεων των μοντέλων σε προβλήματα MIR και το ειδικότερο πρόβλημα των συστάσεων.

Δεδομένα και Μεθοδολογία

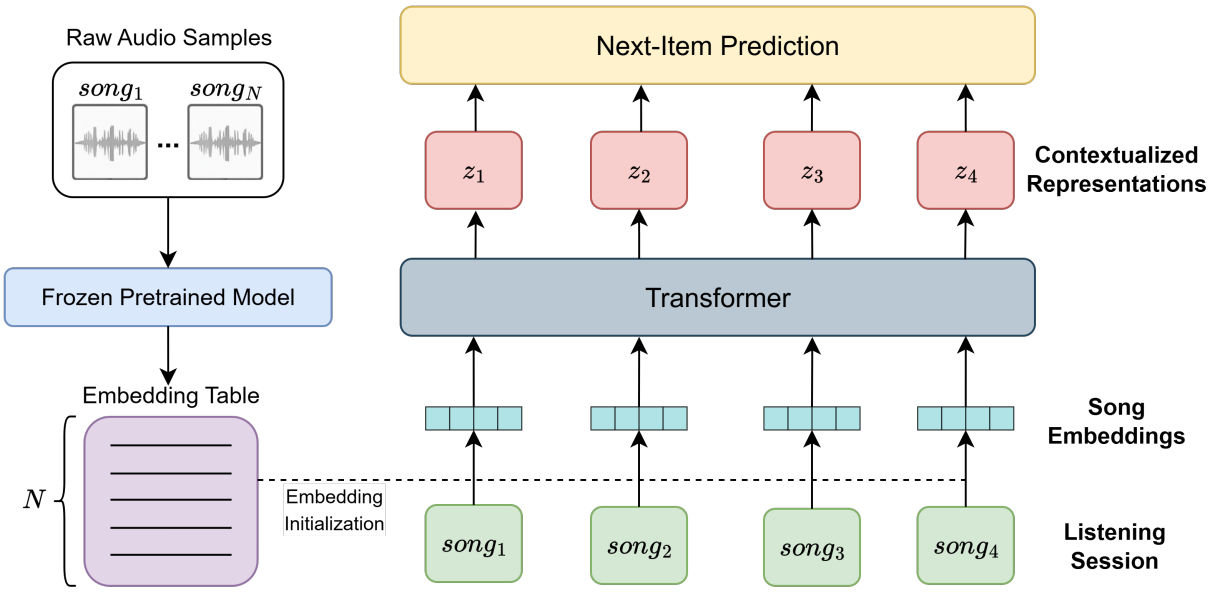
Ο βασικός στόχος της παρούσας εργασίας είναι η αξιοποίηση προεκπαιδευμένων μοντέλων για την εξαγωγή embeddings από σήματα μουσικής και η διερεύνηση διαφορετικών τρόπων ενσωμάτωσής τους σε ένα σύστημα συστάσεων βασισμένο σε συνεδρίες. Για τον σκοπό αυτό, βασιστήκαμε σε δύο υπάρχοντα πλαίσια που περιγράφηκαν παραπάνω: τη βιβλιοθήκη Transformers4Rec της NVIDIA [100] και το μοντέλο Represent-Then-Aggregate (RTA) [7] της Deezer Research. Και στις δύο περιπτώσεις, τα δεδομένα μας αποτελούνται από συνεδρίες ακρόασης, οι οποίες εξάγονται από ένα μουσικό σύνολο δεδομένων Music4All που περιέχει ιστορικά ακροάσεων περίπου 15.000 χρηστών [91].

Στο Σχήμα 0.0.4 παρουσιάζεται μια συνοπτική απεικόνιση της προτεινόμενης προσέγγισης. Σε γενικές γραμμές, τα εξαγόμενα ηχητικά embeddings χρησιμοποιούνται ως αρχικές αναπαραστάσεις για τα τραγούδια της κάθε συνεδρίας ακρόασης, που αποτελεί την ακολουθία εισόδου. Η ακολουθία αυτή εισάγεται σε μια αρχιτεκτονική τύπου Transformer, της οποίας τα επίπεδα αυτοπροσοχής παράγουν αναπαραστάσεις με έμφαση στα συμφραζόμενα, που στη συνέχεια χρησιμοποιούνται για την πρόβλεψη του επόμενου αντικειμένου. Στις επόμενες ενότητες περιγράφονται η προεπεξεργασία των δεδομένων και οι τεχνικές λεπτομέρειες για καθένα από τα δύο πλαίσια.

Προεπεξεργασία Δεδομένων

Επιλογή Συνόλου Δεδομένων

Κατά την επιλογή του συνόλου δεδομένων, κύριο κριτήριο ήταν η διαθεσιμότητα ηχητικών αποσπασμάτων, καθώς τα προεκπαιδευμένα μοντέλα που χρησιμοποιούμε λειτουργούν απευθείας πάνω στο σήμα ήχου. Αυτό το κριτήριο απέκλεισε αρκετά δημοφιλή σύνολα δεδομένων. Για παράδειγμα, το Melon Playlist Dataset [33], παρότι διαθέτει μεγάλο αριθμό επιμελημένων λιστών αναπαραγωγής, παρέχει μόνο mel-φασματογραφήματα χαμηλής ανάλυσης, καθιστώντας αδύνατη την ανακατασκευή ήχου υψηλής ποιότητας. Επιπλέον, περιέχει κυρίως κορεατική μουσική, κάτι που θα μπορούσε να εισάγει πολιτισμικές ή υφολογικές προκαταλήψεις. Παρομοίως, το Million Playlist Dataset (MPD) [15], αν και διαθέτει εκτενή δεδομένα συμπεριφοράς χρηστών, δεν περιλαμβάνει ηχητικά χαρακτηριστικά. Η ανάκτησή τους μέσω του Spotify API θα ήταν εξαιρετικά χρονοβόρα λόγω περιορισμών ταχύτητας και όγκου δεδομένων. Λαμβάνοντας υπόψη τους παραπάνω περιορισμούς, επιλέξαμε το Music4All [91], το οποίο



Σχήμα 0.0.4: Συνοπτική απεικόνιση της προτεινόμενης προσέγγισης. Οι αναπαραστάσεις των τραγουδιών εξάγονται από «παγωμένα» προεκπαιδευμένα μοντέλα και χρησιμοποιούνται για την αρχικοποίηση του αντίστοιχου πίνακα. Μια αρχιτεκτονική Transformer ενσωματώνει πληροφορίες από τα συμπεριφορικά δεδομένα, παράγοντας βελτιωμένες αναπαραστάσεις με σκοπό την πρόβλεψη του επόμενου τραγουδιού.

παρέχει ηχητικά αποσπάσματα 30 δευτερολέπτων μαζί με ιστορικά ακροάσεων. Αν και το πλήρες σύνολο περιλαμβάνει 109.269 τραγούδια, μόνο 99.596 εμφανίζονται στα ιστορικά χρηστών, αποτελώντας το αρχικό υποσύνολο μας.

Δημιουργία Συνεδριών

Το Music4All περιέχει το ιστορικό ακροάσεων κάθε χρήστη για διάστημα τριών μηνών. Για να το χρησιμοποιήσουμε σε εφαρμογή βασισμένη σε συνεδρίες, αναδιατάσσουμε τα γεγονότα ακρόασης σε συνεδρίες βάσει των χρονικών σημάνσεων. Συγκεκριμένα, ακολουθούμε τη μέθοδο των δημιουργών του συνόλου, δημιουργώντας νέα συνεδρία όταν δύο διαδοχικά γεγονότα απέχουν περισσότερο από 60 λεπτά μεταξύ τους.

Ως αποτέλεσμα αυτής της διαδικασίας περίπου 113.000 (18%) συνεδρίες περιλαμβάνουν μόνο ένα τραγούδι ($l = 1$). Αυτές δεν προσφέρουν πληροφορία σε σενάρια διαδοχικών συστάσεων, επομένως απορρίπτονται. Το τελικό σύνολο έχει μέσο μήκος συνεδρίας 9.70, σημαντικά μικρότερο από τα MPD (66.35) και Melon (41.46), γεγονός αναμενόμενο, αφού οι συνεδρίες είναι μικρότερες από ολόκληρα ιστορικά ή λίστες αναπαραγωγής.

Επιλογή Προεκπαιδευμένων Μοντέλων

Για την εξαγωγή ηχητικών αναπαραστάσεων από το Music4All χρησιμοποιήθηκαν τρία μοντέλα: MERT, MusiCNN και ένα προσαρμοσμένο CNN εκπαιδευμένο με αντιθετική εκμάθηση.

- Το MERT [63] είναι ένα γενικού σκοπού προεκπαιδευμένο μοντέλο μεγάλης κλίμακας, εκπαιδευμένο με αυτοεπιβλεπόμενη μάθηση. Έχει δείξει υψηλές επιδόσεις σε ποικίλα προβλήματα MIR, καθιστώντας το κατάλληλο για την εξαγωγή χρήσιμων αναπαραστάσεων ήχου.
- Το MusiCNN [80] είναι ένα CNN εκπαιδευμένο με επιβλεπόμενο τρόπο στο πρόβλημα της αυτόματης επισήμανσης ήχου. Επιλέχθηκε ως ενδεικτικό μοντέλο μίας διαφορετικής στρατηγικής προεκπαίδευσης από το MERT, αλλά και λόγω της αποδεδειγμένης χρησιμότητάς του σε εφαρμογές σύστασης [104].
- Το τρίτο μοντέλο βασίζεται στην αρχιτεκτονική CNN EfficientNet-B0 [105] και εκπαιδεύεται αντιθετικά ώστε να φέρνει τραγούδια του ίδιου καλλιτέχνη πιο κοντά στον χώρο αναπαραστάσεων. Στο εξής αναφερόμαστε σε αυτό ως artist model.

Συνδυάζοντας μοντέλα εκπαιδευμένα με διαφορετικές τεχνικές προεκπαίδευσης, διερευνούμε πώς αυτή η παράμετρος επηρεάζει την ποιότητα των ηχητικών αναπαραστάσεων και την τελική απόδοση στο πρόβλημα παραγωγής συστάσεων βασισμένων σε συνεδρίες.

Προεπεξεργασία Ηχητικών Σημάτων

Όλα τα αρχεία ήχου του Music4All παρέχονται σε στερεοφωνική μορφή με δειγματοληψία 48 kHz. Σε πρώτη φάση, μετατρέπουμε τα αρχεία σε μονοφωνικά υπολογίζοντας τον μέσο όρο των δύο καναλιών και όπου χρειάζεται αφαιρούμε ή συμπληρώνουμε δείγματα ώστε όλα τα σήματα να έχουν διάρκεια 30 δευτερόλεπτα. Επειδή κάθε μοντέλο απαιτεί διαφορετικό ρυθμό δειγματοληψίας (24 kHz το MERT, 16 kHz τα MusiCNN και artist model), επαναδειγματοληπτούμε ανάλογα τα σήματα κατά περίπτωση.

Εξαγωγή Αναπαραστάσεων

Για την εξαγωγή αναπαραστάσεων από το MERT χρησιμοποιούμε την ελαφρύτερη εκδοχή των 95M παραμέτρων[3], με 12 επίπεδα Transformer και διάσταση εξόδου 768. Στο τελευταίο επίπεδο λαμβάνουμε τον μέσο όρο στη χρονική διάσταση, καταλήγοντας με ένα διάνυσμα 768 διαστάσεων ανά τραγούδι. Για το MusiCNN χρησιμοποιούμε το μικρότερο μοντέλο, εκπαιδευμένο στο Million Song Dataset (MSD) [80], με 200-διάστατο τελικό διάνυσμα χαρακτηριστικών. Επειδή το MusiCNN εκπαιδεύτηκε σε αποσπάσματα 3 δευτερολέπτων, οι συγγραφείς προτείνουν τη χρήση του ίδιου χρονικού παραθύρου για εξαγωγή χαρακτηριστικών. Επομένως, χωρίζουμε κάθε απόσπασμα 30 δευτερολέπτων σε δέκα μέρη των 3 δευτερολέπτων και υπολογίζουμε τον μέσο όρο των δέκα embeddings. Για το μοντέλο καλλιτέχνη, πραγματοποιείται αντιθετική προεκπαίδευση στο Music4All [91], χρησιμοποιώντας την ταυτότητα καλλιτέχνη για τη δημιουργία θετικών ζευγών. Το μοντέλο δέχεται mel-φασματογραφήματα 1 δευτερολέπτου και εξάγει embeddings 1280 διαστάσεων. Ομοίως, για την παραγωγή του τελικού embedding κάθε κομματιού, λαμβάνουμε τον μέσο όρο όλων των τμημάτων διάρκειας 1 δευτερολέπτου.

Προσαρμογή των Αναπαραστάσεων του MusiCNN

Ορμώμενοι από τα ευρήματα του [82], πραγματοποιούμε περαιτέρω προσαρμογή των ηχητικών αναπαραστάσεων του MusiCNN με συμπεριφορικά δεδομένα, ώστε να ευθυγραμμιστούν με το τελικό πρόβλημα. Συγκεκριμένα, τροφοδοτούμε ένα απλό νευρωνικό δίκτυο με τα embeddings του MusiCNN και το εκπαιδεύουμε με αντιθετική συνάρτηση κόστους, θεωρώντας τραγούδια που ανήκουν στην ίδια συνεδρία ως θετικά δείγματα και τα υπόλοιπα της παρτίδας ως αρνητικά. Μετά την εκπαίδευση, πραγματοποιούμε ένα τελικό forward pass για την εξαγωγή των βελτιωμένων embeddings, στα οποία αναφερόμαστε ως MusiCNN-tuned.

Transformers4Rec (T4Rec)

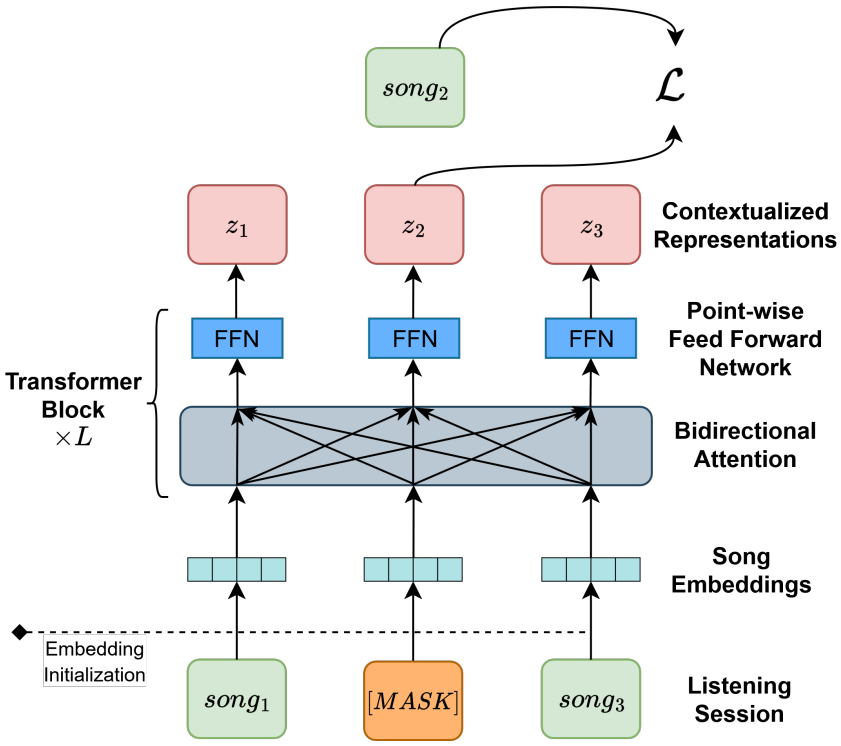
Για να εξετάσουμε τη χρησιμότητα των εξαγόμενων embeddings σε περιβάλλον συστάσεων, προτείνουμε αρχικά ένα μοντέλο βασισμένο στη βιβλιοθήκη T4Rec, το οποίο δανείζεται τεχνικές από δημοφιλή ακολουθιακά μοντέλα συστάσεων όπως το BERT4Rec [103].

Σύνολο Δεδομένων Για τα πειράματα με το T4Rec χρησιμοποιούμε τις προεπεξεργασμένες συνεδρίες της προηγούμενης ενότητας, στις οποίες έχουν αφαιρεθεί οι συνεδρίες που περιλαμβάνουν μόνο ένα τραγούδι. Επιπλέον, εφαρμόζουμε διαχωρισμό 70-15-15 στα δεδομένα, ταξινομημένα κατά αναγνωριστικό χρήστη, γεγονός που σημαίνει ότι στο σύνολο επικύρωσης και δοκιμής περιλαμβάνονται εντελώς νέοι χρήστες.

Αρχιτεκτονική

Επίπεδο Εισόδου Η είσοδος του μοντέλου είναι μια ακολουθία από song IDs που αντιστοιχούν σε μία συνεδρία ακρόασης. Η ακολουθία αυτή περνά από ένα επίπεδο αναπαραστάσεων (embedding layer), δηλαδή τον πίνακα embeddings $E \in R^{|V| \times d}$, όπου V είναι το σύνολο των αντικειμένων και d η διάσταση των embeddings. Ο πίνακας μπορεί είτε να αρχικοποιηθεί τυχαία και να εκπαιδευθεί από το μηδέν, είτε να αρχικοποιηθεί με τα προεκπαιδευμένα ηχητικά embeddings, παρέχοντας ένα σημείο εκκίνησης που ενσωματώνει μουσική πληροφορία. Σε περίπτωση ασυμφωνίας της διάστασης των αναπαραστάσεων και του μοντέλου (π.χ. εκπαιδεύουμε ένα μοντέλο με αναπαράστασεις MusiCNN με d=200, αλλά θέλουμε το μοντέλο να έχει διάσταση 64), προσθέτουμε

[3] https://huggingface.co/m-a-p/MERT-v1-95M



Σχήμα 0.0.5: Αρχιτεκτονική των μοντέλων που βασίζονται στο T4Rec. Κατά την εκπαίδευση, το μοντέλο αποκρύπτει τμήματα της εισόδου και χρησιμοποιεί αμφίδρομη προσοχή για την ανακατασκευή τους.

ένα γραμμικό επίπεδο προβολής ακολουθούμενο από συνάρτηση ReLU. Επειδή ο μηχανισμός αυτοπροσοχής δεν λαμβάνει υπόψη τη σειρά, προστίθεται μία κωδικοποίηση θέσης σε κάθε embedding για να παρακολουθείται η σειρά των τραγουδιών. Αντί της απόλυτης κωδικοποίησης θέσης που χρησιμοποιείται στον αρχικό μετασχηματιστή, χρησιμοποιούμε τη σχετική κωδικοποίηση του TransformerXL [24], εστιάζοντας στην κωδικοποίηση της απόστασης μεταξύ των στοιχείων και όχι στην απόλυτη θέση τους. Αξίζει να σημειωθεί ότι στα πειράματά μας τα προεκπαιδευμένα embeddings είναι εκπαιδεύσιμα και όχι «παγωμένα», καθώς αυτό θα οδηγούσε σε σημαντικά χαμηλότερη απόδοση όπως θα αναφέρουμε παρακάτω.

Επίπεδα Transformer Η ακολουθία των αλληλεπιδράσεων τροφοδοτείται στη συνέχεια στο δίκτυο Transformer, το οποίο βελτιώνει τις αναπαραστάσεις των στοιχείων, συλλέγοντας πολύτιμες πληροφορίες σχετικά με τις σχέσεις μεταξύ των τραγουδιών και τα μοτίβα συνύπαρξης. Όπως προαναφέραμε, το T4Rec επιτρέπει την εφαρμογή δημοφιλών αρχιτεκτονικών Transformer για παραγωγή συστάσεων. Σε αυτή την εργασία υλοποιούμε το δίκτυο ως στοίβα επιπέδων βασισμένων στην αρχιτεκτονική του XLNet [116], όπως προτείνεται στο αρχικό άρθρο [100]. Αυτό δεν σημαίνει ότι αντιγράφουμε πλήρως τη διαμόρφωση του XLNet (π.χ. αριθμό επιπέδων, εσωτερικές διαστάσεις, αριθμό κεφαλών), αλλά ότι υιοθετούμε τους εσωτερικούς μηχανισμούς του.

Επίπεδο Εξόδου Για την πρόβλεψη του επόμενου αντικειμένου στο χρονικό βήμα t, υπολογίζουμε τις βαθμολογίες όλων των αντικειμένων πολλαπλασιάζοντας την τελική τους αναπαράσταση με έναν πίνακα διαστάσεων $|V| \times d$. Ο πίνακας αυτός μπορεί να μοιράζεται τα ίδια βάρη με τον πίνακα αναπαραστάσεων E — μια τεχνική γνωστή ως weight tying, η οποία μειώνει σημαντικά τον αριθμό παραμέτρων, δεδομένου του μεγάλου πλήθους αντικειμένων σε εφαρμογές συστάσεων. Αν χρειάζεται, προβάλλουμε την αναπαράσταση πίσω στη διάσταση d μέσω γραμμικού επιπέδου. Εν συνεχεία, οι βαθμολογίες περνούν από συνάρτηση softmax, ώστε να παραχθεί μία κατανομή πιθανοτήτων για τα υποψήφια αντικείμενα, που εκφράζει την πιθανότητα κάθε αντικειμένου να είναι το αντικείμενο που λείπει.

Εκπαίδευση

Το μοντέλο εκπαιδεύεται με τη μέθοδο Masked Language Modeling (MLM) που χρησιμοποιείται στο δημοφιλές μοντέλο BERT4Rec [103]. Αυτό επιτρέπει τη χρήση αμφίδρομου μηχανισμού προσοχής κατά την πρόβλεψη ενός αντικειμένου, αποτρέποντας ταυτόχρονα τη διαρροή πληροφορίας παρόλο που είναι διαθέσιμα τα επόμενα κομμάτια. Σύμφωνα με το MLM, αντικαθιστούμε ένα ποσοστό ρ των αντικειμένων με ένα embedding [MASK] και τα οποία το μοντέλο μαθαίνει να ανακτά βάσει των υπόλοιπων κομματιών. Για κάθε θέση που αποκρύπτεται, η τελική αναπαράσταση χρησιμοποιείται για την παραγωγή κατανομής πιθανοτήτων επί του συνόλου αντικειμένων και η συνάρτηση κόστους υπολογίζεται ως ο αρνητικός λογάριθμος της πιθανότητας του σωστού αντικειμένου:

$$\mathcal{L} = \frac{1}{|\mathcal{S}_i^m|} \sum_{v_m \in \mathcal{S}_i^m} -\log P(v_m = v_m^* | \mathcal{S}_i') \tag{0.0.7}$$

όπου $\mathcal{S}_i^m$ είναι τα μασκαρισμένα αντικείμενα και v_m^* το πραγματικό αντικείμενο.

Αξιολόγηση

Κατά την αξιολόγηση στο σύνολο επικύρωσης ή δοκιμής, αξιολογούμε την ικανότητα του μοντέλου να προβλέψει το τελευταίο αντικείμενο κάθε συνεδρίας, δεδομένων όλων των προηγούμενων. Δηλαδή, μόνο το τελευταίο αντικείμενο αντικαθίσταται και προβλέπεται από το μοντέλο, το οποίο επιστρέφει μια ταξινομημένη λίστα συστάσεων. Η λίστα αξιολογείται με τις μετρικές NDCG@k, MAP@k και Recall@k, για $k = 10, 20$. Κατά το inference, η είσοδος επεκτείνεται με ένα επιπλέον [MASK] embedding και όλες οι προηγούμενες αλληλεπιδράσεις χρησιμοποιούνται για την πρόβλεψη του επόμενου αντικειμένου.

Ύστερη Συνένωση (Late Fusion)

Ένα από τα ερωτήματα που θέσαμε αφορά εναλλακτικούς τρόπους ενσωμάτωσης των προεκπαιδευμένων embeddings. Μία πιθανή ιδέα είναι η τυχαία αρχικοποίηση του πίνακα embeddings του Transformer, ώστε τα επίπεδα να αξιοποιούν αποκλειστικά τα συμπεριφορικά δεδομένα. Έπειτα, λαμβάνουμε την έξοδο του Transformer και τη συνενώνουμε με το μέσο embedding της συνεδρίας (εξαιρουμένων των μασκαρισμένων αντικειμένων). Μια δεύτερη επιλογή θα ήταν η συνένωση κάθε αναπαράστασης αντικειμένου με το αντίστοιχο embedding του ήχου. Ωστόσο, αυτό θα οδηγούσε σε διαρροή πληροφορίας, καθώς το μοντέλο θα μάθαινε να βασίζει τις προβλέψεις του στις ηχητικές αναπαραστάσεις, οι οποίες δεν μασκάρονται. Για τον ίδιο ακριβώς λόγο, δεν λαμβάνουμε υπόψη τα μασκαρισμένα αντικείμενα κατά τον υπολογισμό του μέσου embedding της συνεδρίας.

Represent-Then-Aggregate (RTA)

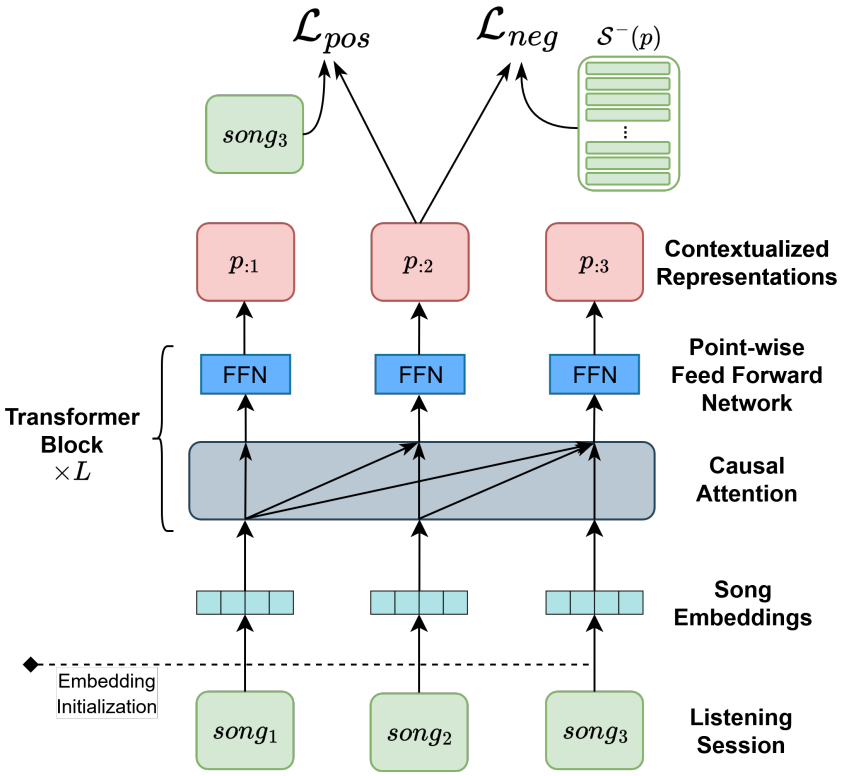
Το δεύτερο μέρος της εργασίας μας περιλαμβάνει την εκπαίδευση μοντέλων βασισμένων στο RTA χρησιμοποιώντας το Music4All, ώστε να εξεταστεί πώς συμπεριφέρονται τα ηχητικά embeddings σε μία διαφορετική διατύπωση του προβλήματος. Όπως θυμόμαστε, το πλαίσιο RTA χρησιμοποιεί μια συνάρτηση για να παράγει την αναπαράσταση κάθε τραγουδιού και στη συνέχεια εφαρμόζει μια συνάρτηση συνάθροισης ώστε να αναπαραστήσει μια συνεδρία p ως συνδυασμό των αναπαραστάσεων των τραγουδιών της. Σε σύγκριση με το πλαίσιο Transformers4Rec, εντοπίζουμε τις εξής βασικές διαφορές:

- Το RTA αφαιρεί τα διπλότυπα τραγούδια από κάθε συνεδρία, με αποτέλεσμα ένα ελαφρώς διαφορετικό σύνολο δεδομένων, ενώ στο T4Rec αφαιρούνται μόνο οι συνεδρίες μήκους ενός κομματιού.
- Το αρχικό άρθρο του RTA ακολουθεί διαφορετικό τρόπο διαχωρισμού των δεδομένων, δειγματοληπτώντας τυχαία σημαντικά μικρότερα σύνολα επικύρωσης και δοκιμής.
- Κάθε πλαίσιο χρησιμοποιεί διαφορετικό μαθησιακό στόχο: στο T4Rec ακολουθούμε προσέγγιση MLM (μερική απόκρυψη εισόδου και πρόβλεψη κρυμμένων αντικειμένων), ενώ στο RTA μεγιστοποιείται η ομοιότητα μεταξύ της αναπαράστασης της συνεδρίας και του embedding του πραγματικού επόμενου τραγουδιού.
- Η διαφορά στόχου εκπαίδευσης οδηγεί και σε διαφορετική αρχιτεκτονική: η αμφίδρομη αυτοπροσοχή του T4Rec αντικαθίσταται από αιτιώδη αυτοπροσοχή στο RTA.

- Τέλος, διαφέρει και το σενάριο αξιολόγησης: στο T4Rec χρησιμοποιούμε ολόκληρη τη συνεδρία για να προβλέψουμε το τελευταίο τραγούδι, ενώ στο RTA χρησιμοποιείται ένας μικρός αριθμός αρχικών τραγουδιών για την ανάκτηση όλων των υπολοίπων.

Σύνολο δεδομένων Όπως αναφέραμε, στο RTA εφαρμόζουμε ένα επιπλέον βήμα προεπεξεργασίας (ακολουθώντας την αρχική εργασία) για να εξασφαλίσουμε τη συμβατότητα με το υπόλοιπο πλαίσιο: πέρα από την αφαίρεση συνεδριών με ένα μόνο τραγούδι, αφαιρούμε και τα διπλότυπα μέσα σε κάθε συνεδρία, κρατώντας μόνο την πρώτη εμφάνιση ενός τραγουδιού. Το βήμα αυτό παράγει περίπου 20k επιπλέον συνεδρίες με ένα μόνο τραγούδι, τις οποίες και απορρίπτουμε στη συνέχεια.

Για τον διαχωρισμό δεδομένων επίσης δεν αποκλίνουμε από την αρχική δημοσίευση: δειγματοληπτούμε 20.000 συνεδρίες με μήκος $l \geq 20$ ώστε να δημιουργήσουμε σύνολο επικύρωσης και δοκιμής από 10.000 συνεδρίες το καθένα. Οι υπόλοιπες 473.297 συνεδρίες αποτελούν το σύνολο εκπαίδευσης.



Σχήμα 0.0.6: Αρχιτεκτονική των μοντέλων που βασίζονται στο RTA. Στο χρονικό σημείο i, ο transformer συνδυάζει τα προηγούμενα αντικείμενα σε μια αναπαράσταση συνεδρίας $p_{:i}$ εστιάζοντας μόνο σε προηγούμενα κομμάτια και χρησιμοποιεί το $p_{:i}$ για πρόβλεψη του επόμενου τραγουδιού. Η εκπαίδευση χρησιμοποιεί υβριδική συνάρτηση κόστους που ενθαρρύνει την ομοιότητα με το πραγματικό επόμενο τραγούδι ($\mathcal{L}_{pos}$) και αποθαρρύνει την ομοιότητα με ένα σύνολο αρνητικών δειγμάτων $S^-(p)$ ($\mathcal{L}_{neg}$).

Αρχιτεκτονική

Πέρα από τις συναρτήσεις αναπαράστασης του [7] (που θα χρησιμοποιήσουμε ως μοντέλα αναφοράς), στην εργασία μας αξιοποιούμε τα προεκπαιδευμένα μοντέλα για την εξαγωγή αναπαραστάσεων τραγουδιών. Όσον αφορά τη συνάρτηση συνάθροισης, τα αποτελέσματα της αρχικής δημοσίευσης έδειξαν ότι τα μοντέλα τύπου Transformer υπερέχουν έναντι εναλλακτικών όπως GRUs, Gated CNNs ή ο απλός μέσος όρος αναπαραστάσεων. Για τον λόγο αυτό εστιάζουμε αποκλειστικά σε αυτή την εκδοχή. Συγκεκριμένα, σε αυτή την περίπτωση ο μετασχηματιστής επεξεργάζεται την είσοδο p με μπλοκ αιτιώδους αυτοπροσοχής, δηλαδή το μοντέλο «βλέπει» μόνο θέσεις έως και το τρέχον στοιχείο i και όχι μελλοντικές. Η έξοδος που παράγεται μπορεί να θεωρηθεί

ως αναπαράσταση συνεδρίας που συναθροίζει τα $p_{s_1}, p_{s_2}, \ldots, p_{s_i}$ και να χρησιμοποιηθεί για πρόβλεψη του $p_{s_{i+1}}$ μέσω βαθμολόγησης με εσωτερικό γινόμενο όλων των αναπαραστάσεων αντικειμένων. Αυτό ισοδυναμεί με την τεχνική weight tying που αναφέρθηκε προηγουμένως. Όπως και στην περίπτωση του T4Rec, σε όλα μας τα πειράματα οι αναπαραστάσεις τραγουδιών παραμένουν εκπαιδεύσιμες.

Εκπαίδευση

Για την εκπαίδευση του μοντέλου, οι συγγραφείς προτείνουν τον εξής στόχο: δεδομένης μίας συνεδρίας p μήκους l και της συνάθροισης $p_{:i}$ των πρώτων i τραγουδιών της, με $i \in \{1, \ldots, l-1\}$, ενθαρρύνεται η υψηλότερη ομοιότητα μεταξύ της $p_{:i}$ και του πραγματικού επόμενου τραγουδιού $p_{s_{i+1}}$, και η χαμηλότερη ομοιότητα για ένα σύνολο δειγμάτων $S^-(p)$ που δεν ανήκουν στην p. Η πραγματική συνέχεια λειτουργεί ως θετικό ζεύγος, ενώ τα εκτός συνεδρίας τραγούδια ως αρνητικά, και η συνολική συνάρτηση κόστους γράφεται ως $\mathcal{L}(p) = \mathcal{L}_{\text{pos}}(p) + \mathcal{L}_{\text{neg}}(p)$, όπου:

$$\mathcal{L}_{\text{pos}}(p) = -\sum_{i=1}^{l-1} \log \sigma\left(f\left(p_{:i}, p_{s_{i+1}}\right)\right) \tag{0.0.8}$$

$$\mathcal{L}_{\text{neg}}(p) = -\sum_{i=1}^{l-1} \sum_{s^- \in \mathcal{S}^-(p)} \log \sigma\left(f\left(p_{:i}, s^-\right)\right) \tag{0.0.9}$$

όπου σ είναι η σιγμοειδής συνάρτηση και f το εσωτερικό γινόμενο.

Αξιολόγηση

Για την αξιολόγηση ακολουθούμε το [7] και χωρίζουμε τις 10k συνεδρίες του σετ επικύρωσης ή δοκιμής σε 10 υποσύνολα των 1.000 συνεδριών. Στο πρώτο υποσύνολο είναι ορατό μόνο το πρώτο τραγούδι, στο δεύτερο τα δύο πρώτα κ.ο.κ., μέχρι τα 10 ορατά τραγούδια. Το υπόλοιπο μέρος κάθε συνεδρίας πρέπει να προβλεφθεί από το μοντέλο, το οποίο βαθμολογεί όλα τα τραγούδια και παράγει μία ταξινομημένη λίστα υποψηφίων κομματιών για κάθε συνεδρία.

Μοντέλα Αναφοράς

Για να αξιολογήσουμε την αποτελεσματικότητα των προεκπαιδευμένων ηχητικών αναπαραστάσεων, τις συγκρίνουμε με τις τρεις συναρτήσεις που χρησιμοποιούνται στο RTA, οι οποίες λειτουργούν ως μοντέλα αναφοράς:

- Weighted Regularized Matrix Factorization (WRMF) [48]: Παραγοντοποίηση του πίνακα συνεμφάνισης συνεδρίας–τραγουδιού με Alternating Least Squares (ALS) για εξαγωγή αρχικών αναπαραστάσεων τραγουδιών με $d = 128$. Ανα φέρεται ως MF στα πειράματά μας.
- Factorization Machine (FM): Με βάση τις παραπάνω αναπαραστάσεις τραγουδιών, δημιουργούνται embeddings μεταδεδομένων (καλλιτέχνης, άλμπουμ, δημοτικότητα) ως μέσος όρος των τραγουδιών που μοιράζονται την ίδια τιμή μεταδεδομένων. Το embedding κάθε τραγουδιού είναι ο μέσος όρος των embeddings των μεταδεδομένων του.
- Neural Network (NN): Παρόμοιο με το FM, αλλά αντί για μέσο όρο χρησιμοποιεί μηχανισμό προσοχής πάνω στα embeddings των μεταδεδομένων.

Πειραματική Αξιολόγηση

Μοντέλα προς Αξιολόγηση

Στα πειράματά μας, ο βασικός τρόπος ενσωμάτωσης των προεκπαιδευμένων embeddings στα μοντέλα συστάσεων είναι ως αρχικοποίηση του πίνακα embeddings E των αντικειμένων, με στόχο να προσφέρουν στο μοντέλο ένα πιο χρήσιμο σημείο εκκίνησης. Ως βασικό σημείο αναφοράς, χρησιμοποιούμε την τυχαία αρχικοποίηση του πίνακα αναπαραστάσεων, αφήνοντας το μοντέλο να μάθει αποκλειστικά από τα δεδομένα συμπεριφοράς. Σε αυτή την περίπτωση, το E προφανώς παραμένει εκπαιδεύσιμο. Στην περίπτωση προεκπαιδευμένης αρχικοποίησης, προηγούμενες εργασίες [104] αναφέρουν ότι το αν το E παραμένει «παγωμένο» ή όχι δεν έχει σημαντική επίδραση στην απόδοση. Ωστόσο, τα προκαταρκτικά μας πειράματα έδειξαν ότι το πάγωμα του πίνακα embeddings

οδηγεί σε σημαντικά χειρότερα αποτελέσματα, επομένως το διατηρούμε εκπαιδεύσιμο καθ' όλη τη διάρκεια των πειραμάτων.

Όπως έχει ήδη αναφερθεί, χρησιμοποιούμε embeddings από τρία προεκπαιδευμένα μοντέλα: MusiCNN [80], MERT [63] και artist model, καθώς και μια τροποποιημένη εκδοχή των embeddings του MusiCNN που έχουν υποστεί περαιτέρω προσαρμογή (finetuning) στις συνεδρίες ακρόασης. Πέρα από τη χρήση διαφορετικής μεθόδου προεκπαίδευσης, κάθε μοντέλο διαθέτει διαφορετική διάσταση embedding d (200, 768 και 1280 αντίστοιχα). Για τον λόγο αυτό, η εκπαίδευση ενός μόνο baseline με π.χ. $d = 512$ δεν θα εξασφάλιζε δίκαιη σύγκριση, καθώς κάθε προεκπαιδευμένο μοντέλο θα είχε διαφορετικές δυνατότητες (capacity) από το μοντέλο αναφοράς. Για να απομονώσουμε την επίδραση της αρχικοποίησης, εκπαιδεύουμε τρία τυχαία αρχικοποιημένα μοντέλα, καθένα με διάσταση αναπαραστάσεων d ίση με εκείνη του αντίστοιχου προεκπαιδευμένου μοντέλου.

Επιπλέον, στα πειράματα με το Transformers4Rec, δοκιμάζουμε και την τεχνική της ύστερης συνένωσης (late fusion), όπως παρουσιάστηκε παραπάνω, τόσο για τα embeddings του MusiCNN όσο και του MERT. Σε αυτή την περίπτωση, θέτουμε τη διάσταση $d = 512$. Για παράδειγμα, αν η έξοδος του transformer έχει διάσταση 256, τότε μετά τη συνένωση με τα embeddings του MusiCNN ($d = 200$) προκύπτει διάνυσμα 456 διαστάσεων, το οποίο προβάλλεται στη διάσταση αναπαραστάσεων $d = 512$.

Όσον αφορά το RTA, εκτός από τα τυχαία αρχικοποιημένα μοντέλα που λειτουργούν ως μοντέλα αναφοράς, συμπεριλαμβάνουμε και τις τρεις συναρτήσεις της αρχικής δημοσίευσης (WRMF, FM και NN) που περιγράφηκαν προηγουμένως, καθώς και ένα επιπλέον τυχαίο μοντέλο ίδιας διάστασης embeddings με αυτά ($d = 128$) για άμεση σύγκριση.

Ποσοτικά Αποτελέσματα

Transformers4Rec

Τα αποτελέσματα όλων των μοντέλων T4Rec στο test set παρουσιάζονται στον Πίνακα 1 για λόγους σύγκρισης. Η δεύτερη στήλη δείχνει τη διάσταση των embeddings d κάθε μοντέλου, ενώ υπενθυμίζεται ότι η κρυφή διάσταση του transformer είναι 512 για όλα τα μοντέλα. Τα υπογραμμισμένα αποτελέσματα αντιστοιχούν στην καλύτερη επίδοση εντός μιας ομάδας μοντέλων (π.χ. MERT έναντι του τυχαίου μοντέλου με την ίδια διάσταση), ενώ τα έντονα γράμματα δηλώνουν τη συνολικά καλύτερη επίδοση στη συγκεκριμένη μετρική.

Πίνακας 1: Σύγκριση της απόδοσης των μοντέλων T4Rec σε όλες τις μετρικές αξιολόγησης.

Αρχικοποίηση	d	NDCG@10	MAP@10	Recall@10	NDCG@20	MAP@20	Recall@20
Random	200	0.2504	0.2013	0.4038	0.2629	0.2047	0.4530
MusiCNN	200	0.2225	0.1788	0.3603	0.2358	0.1824	0.4124
MusiCNN-tuned	200	0.2452	0.1953	0.4009	0.2583	0.1988	0.4525
Random	768	0.2618	0.2140	0.4105	0.2742	0.2174	0.4595
MERT	768	0.2582	0.2089	0.4089	0.2706	0.2122	0.4580
Random	512	0.2676	0.2216	**0.4119**	0.2805	0.2252	**0.4626**
MusiCNN late fusion	512	0.2655	0.2197	0.4088	0.2782	0.2231	0.4589
MERT late fusion	512	0.2628	0.2158	0.4096	0.2758	0.2193	0.4607
Random	1280	**0.2684**	**0.2232**	0.4095	**0.2808**	**0.2266**	0.4583
Artist model	1280	0.2309	0.1848	0.3739	0.2451	0.1886	0.4297

MusiCNN Ξεκινώντας από τα μοντέλα με διάσταση embedding $d = 200$, παρατηρούμε ότι η τυχαία αρχικοποίηση επιτυγχάνει την καλύτερη επίδοση σε όλες τις μετρικές αυτής της ομάδας. Το μοντέλο που αρχικοποιείται με embeddings του MusiCNN αποδίδει αισθητά χειρότερα, γεγονός που υποδηλώνει ότι οι αναπαραστάσεις που προέρχονται από εκπαίδευση για αυτόματη επισήμανση (auto-tagging) δεν ευθυγραμμίζονται καλά με τα μοτίβα συμπεριφοράς χρηστών στη συγκεκριμένη εφαρμογή συστάσεων. Ωστόσο, όταν οι αναπαραστάσεις του MusiCNN προσαρμόζονται με δεδομένα χρηστών, η απόδοσή τους βελτιώνεται σε όλες τις μετρικές. Αυτό δείχνει ότι, παρότι τα προεκπαιδευμένα ηχητικά χαρακτηριστικά μπορεί να μην ήταν ιδανικά στην αρχική τους μορφή, μπορούν να αποτελέσουν χρήσιμο σημείο εκκίνησης αν προσαρμοστούν κατάλληλα.

Παρ' όλα αυτά, ακόμη και η προσαρμοσμένη εκδοχή δεν φτάνει την επίδοση του τυχαίου μοντέλου, γεγονός που υπογραμμίζει τη σημασία της εκμάθησης embeddings απευθείας για το συγκεκριμένο πρόβλημα.

MERT Στην ομάδα με $d = 768$, συγκρίνουμε τα μοντέλα που αρχικοποιούνται με MERT με το τυχαίο μοντέλο αναφοράς. Και εδώ, το τυχαίο μοντέλο υπερέχει ελαφρώς, αν και η διαφορά απόδοσης είναι μικρότερη σε σχέση με την προηγούμενη ομάδα. Αυτό υποδεικνύει ότι το MERT, το οποίο έχει εκπαιδευτεί με αυτοεπιβλεπόμενη μάθηση, παρέχει πιο χρήσιμες αναπαραστάσεις για το πρόβλημα της παραγωγής συστάσεων σε σχέση με το MusiCNN. Η μικρότερη απόκλιση συνιστά μία ένδειξη ότι η μουσική γνώση που κωδικοποιεί η μέθοδος προεκπαίδευσης του MERT είναι πιο χρήσιμη στο συγκεκριμένο τελικό πρόβλημα. Ωστόσο, το γεγονός ότι η τυχαία αρχικοποίηση εξακολουθεί να υπερέχει, έστω και οριακά, επιβεβαιώνει πως τα μοντέλα ωφελούνται περισσότερο όταν τα embeddings μαθαίνονται εξ ολοκλήρου για το εκάστοτε πρόβλημα σύστασης.

Ύστερη Συνένωση (Late Fusion) Η ομάδα με $d = 512$ διερευνά την επίδραση της ύστερης συνένωσης, όπου τα προεκπαιδευμένα embeddings συνδυάζονται με τα εκπαιδεύσιμα embeddings των αντικειμένων αντί να τα αντικαθιστούν. Σε αυτή τη διαμόρφωση, το τυχαίο baseline παραμένει ελαφρώς ανώτερο, αν και οι παραλλαγές με ύστερη συνένωση των MusiCNN και MERT εμφανίζουν ανταγωνιστική απόδοση. Οι διαφορές είναι μικρές, με το MusiCNN να υπερέχει ελαφρώς του MERT. Τα αποτελέσματα δείχνουν ότι, παρότι οι ηχητικές αναπαραστάσεις από μόνες τους δεν αρκούν, μπορούν να προσφέρουν συμπληρωματική πληροφορία όταν συνδυάζονται με embeddings προσαρμοσμένα στο πρόβλημα. Η ύστερη συνένωση επιτρέπει στο μοντέλο να ενσωματώνει επιλεκτικά τα χαρακτηριστικά του ήχου χωρίς να περιορίζεται από αυτά, εξηγώντας έτσι τη σχετικά καλή απόδοση. Παρ' όλα αυτά, η συνεπής υπεροχή του πλήρως τυχαίου μοντέλου δείχνει ότι ακόμη και με συνένωση, η πληροφορία περιεχομένου δεν γεφυρώνει πλήρως το χάσμα με τα embeddings που βελτιστοποιούνται άμεσα για την παροχή συστάσεων. Επιπλέον, υπενθυμίζεται ότι η τεχνική που υιοθετήθηκε συνδυάζει τα embeddings αντικειμένων με μια μέση αναπαράσταση συνεδρίας, γεγονός που μπορεί να εξομαλύνει τη λεπτομέρεια των αναπαραστάσεων σε επίπεδο τραγουδιού.

Artist-based Μοντέλο Στην ομάδα με $d = 1280$, συγκρίνουμε την τυχαία αρχικοποίηση με embeddings που προέρχονται από το artist model. Το τυχαίο μοντέλο επιτυγχάνει την καλύτερη συνολική επίδοση στις περισσότερες μετρικές, δείχνοντας ότι η αυξημένη διάσταση αναπαραστάσεων μπορεί να ευνοεί την εκμάθηση από την αρχή. Αντίθετα, το artist model αποδίδει σημαντικά χειρότερα, υστερώντας σε σύγκριση με όλα τα υπόλοιπα ηχητικά μοντέλα. Αυτό πιθανότατα οφείλεται στη γενική φύση των συγκεκριμένων αναπαραστάσεων, οι οποίες δεν διακρίνουν αποτελεσματικά τα τραγούδια μέσα στο ρεπερτόριο του ίδιου καλλιτέχνη – μια κρίσιμη ιδιότητα για τη συγκεκριμένη εφαρμογή. Τα αποτελέσματα αυτά ενισχύουν την παρατήρηση ότι αναπαραστάσεις που βασίζονται αποκλειστικά σε μεταδεδομένα δεν επαρκούν για να αποτυπώσουν τη σύνθετη δυναμική της συμπεριφοράς των χρηστών.

Συνολικά, παρατηρούμε μια σταθερή τάση: τα μοντέλα με τυχαία αρχικοποίηση των embeddings τείνουν να υπερέχουν εκείνων που βασίζονται σε ηχητικά χαρακτηριστικά, ειδικά όταν το μέγεθος των embeddings είναι αρκετά μεγάλο ώστε να επιτρέπει την εκμάθηση επαρκών αναπαραστάσεων. Οι στρατηγικές προσαρμογής και συνένωσης μειώνουν τη διαφορά απόδοσης, αλλά δεν την εξαλείφουν. Τα ευρήματα αυτά δείχνουν ότι, παρότι οι ηχητικές αναπαραστάσεις παρέχουν χρήσιμη προϋπάρχουσα γνώση, η χρησιμότητά τους στις στο πρόβλημα συστάσεων του T4Rec είναι περιορισμένη, εκτός αν προσαρμοστούν ή συνδυαστούν προσεκτικά με εκπαιδεύσιμες αναπαραστάσεις που βελτιστοποιούνται άμεσα για το συγκεκριμένο πρόβλημα.

RTA

Όμοια με τον Πίνακα 1, ο Πίνακας 2 παρουσιάζει τις μετρικές του test set για όλα τα μοντέλα που βασίζονται στο RTA. Και εδώ, η δεύτερη στήλη αντιστοιχεί στη διάσταση d κάθε embedding, με τη διαφορά ότι στην περίπτωση του RTA, η διάσταση αυτή ταυτίζεται με την κρυφή διάσταση του transformer. Για παράδειγμα, αν $d = 200$, τότε και ο transformer έχει την ίδια κρυφή διάσταση.

Μοντέλα Αναφοράς RTA Στην ομάδα των μοντέλων με διάσταση $d = 128$, παρατηρούμε ότι τα μοντέλα που προτάθηκαν στην αρχική δημοσίευση του RTA βελτιώνονται προοδευτικά σε σχέση με το τυχαίο μοντέλο αναφοράς. Το MF εμφανίζει ήδη υψηλότερες επιδόσεις από το Random σε όλες τις μετρικές, ενώ το FM βελτιώνει περαιτέρω το MF, γεγονός που δείχνει ότι η αξιοποίηση μεταδεδομένων ή πιο πλούσιων αλληλεπιδράσεων βοηθά το μοντέλο να αποτυπώσει καλύτερα τη δομή των δεδομένων. Ανάμεσά στα μοντέλα αυτής

Πίνακας 2: Σύγκριση της απόδοσης των μοντέλων RTA σε όλες τις μετρικές αξιολόγησης.

Αρχικοποίηση	**d**	**NDCG@10**	**MAP@10**	**Recall@10**	**NDCG@20**	**MAP@20**	**Recall@20**
Random	128	0.1541	0.0919	0.0613	0.1331	0.0644	0.0940
MF	128	0.1708	0.1035	0.0674	0.1467	0.0725	0.1028
FM	128	0.2022	0.1350	0.0810	0.1751	0.0980	0.1243
NN	128	**0.2071**	**0.1371**	**0.0838**	**0.1806**	**0.1006**	**0.1298**
Random	200	0.1499	0.0900	0.0600	0.1302	0.0637	0.0926
MusiCNN	200	0.1668	0.1001	0.0660	0.1434	0.0702	0.1005
MusiCNN-tuned	200	0.1699	0.1042	0.0669	0.1442	0.0721	0.0999
Random	768	0.1588	0.0957	0.0618	0.1339	0.0656	0.0917
MERT	768	0.1507	0.0904	0.0604	0.1303	0.0635	0.0925
Random	1280	0.1522	0.0910	0.0600	0.1301	0.0630	0.0910
Artist model	1280	0.1819	0.1137	0.0722	0.1579	0.0814	0.1116

της ομάδας, το NN—το οποίο εφαρμόζει μηχανισμό προσοχής στα embeddings των μεταδεδομένων—επιτυγχάνει την υψηλότερη απόδοση, με NDCG@20 0.1806 έναντι 0.1331 για το Random. Αυτό δείχνει ότι ακόμη και μια απλή παραγοντοποίηση πινάκων, εμπλουτισμένη με μεταδεδομένα και μηχανισμό προσοχής, μπορεί να εξάγει περισσότερη πληροφορία από ένα απλό τυχαίο μοντέλο.

MusiCNN Περνώντας στην ομάδα με $d = 200$, παρατηρούμε ότι και οι δύο παραλλαγές του MusiCNN υπερέχουν του Random, σε αντίθεση με ό,τι παρατηρήθηκε στα πειράματα του T4Rec. Το NDCG@20 βελτιώνεται από 0.1302 (Random) σε 0.1434 (MusiCNN) και 0.1442 (MusiCNN-tuned), με ανάλογες βελτιώσεις σε MAP και Recall. Αυτό δείχνει ότι, στο πλαίσιο του RTA, οι ηχητικές αναπαραστάσεις μπορούν να προσφέρουν καλύτερο σημείο εκκίνησης από την τυχαία αρχικοποίηση. Επιπλέον, η περαιτέρω προσαρμογή οδηγεί ξανά σε σταθερή βελτίωση, αν και μικρότερη σε σχέση με το T4Rec, επιβεβαιώνοντας ότι τα προεκπαιδευμένα χαρακτηριστικά είναι πιο ωφέλιμα όταν ευθυγραμμίζονται με τον στόχο της σύστασης.

MERT Στη διάσταση $d = 768$, παρατηρούμε διαφορετική εικόνα για τα embeddings του MERT, με το τυχαίο baseline να υπερέχει ελαφρώς του MERT στις περισσότερες μετρικές (NDCG@20 = 0.1339 έναντι 0.1303). Αυτό υποδηλώνει ότι δεν λειτουργούν όλες οι προεκπαιδευμένες αναπαραστάσεις με την ίδια αποτελεσματικότητα στο πλαίσιο του RTA. Παρότι τα embeddings του MERT κωδικοποιούν αποδεδειγμένα χρήσιμη μουσική γνώση, φαίνεται να μην ευθυγραμμίζονται πλήρως με το τελικό πρόβλημα όπως αυτό διατυπώνεται στο RTA. Η μικρή διαφορά απόδοσης πάντως δείχνει ότι το MERT εξακολουθεί να παρέχει χρήσιμη πληροφορία, αλλά το πλαίσιο δεν φαίνεται να ωφελείται σημαντικά πέρα από ό,τι μαθαίνει από το μηδέν.

Artist-based Μοντέλο Τέλος, στη διάσταση $d = 1280$, τα embeddings σε επίπεδο καλλιτέχνη επιτυγχάνουν υψηλότερες επιδόσεις από το τυχαίο baseline σε όλες τις μετρικές, κάτι που επίσης διαφωνεί με τα ευρήματά μας στο T4Rec. Ενδεικτικά, το NDCG@20 βελτιώνεται από 0.1301 (Random) σε 0.1579 (Artist model). Αυτό δείχνει ότι, στο πλαίσιο του RTA, ακόμη και τα πιο γενικά χαρακτηριστικά, όπως η ταυτότητα καλλιτέχνη, μπορούν να παρέχουν χρήσιμη γνώση. Παρ' ότι βελτιωμένα σε σχέση με το Random, τα αποτελέσματα αυτά παραμένουν χαμηλότερα από τις επιδόσεις των μοντέλων NN και FM.

Συνολικά, τα αποτελέσματα του RTA παρουσιάζουν πιο διαφοροποιημένη εικόνα σε σχέση με τα πειράματα του T4Rec. Εδώ, οι προεκπαιδευμένες αναπαραστάσεις, όπως εκείνες του MusiCNN και του artist model, μπορούν να προσφέρουν αισθητές βελτιώσεις έναντι της τυχαίας αρχικοποίησης. Οι κλασικές προσεγγίσεις MF και FM λειτουργούν ως ισχυρά μοντέλα αναφοράς, ενώ το απλό νευρωνικό μοντέλο NN παρουσιάζει την καλύτερη απόδοση στην ομάδα $d = 128$. Τα ευρήματα υποδεικνύουν ότι το RTA επωφελείται περισσότερο από την ενσωμάτωση προεκπαιδευμένων αναπαραστάσεων, γεγονός που μπορεί να αποδοθεί στις σημαντικές διαφορές του σε σχέση με το T4Rec.

Οπτικοποίηση του Χώρου Αναπαραστάσεων

Για να κατανοήσουμε καλύτερα τι μαθαίνει πραγματικά κάθε μοντέλο, οπτικοποιούμε τους χώρους αναπαραστάσεων των μοντέλων T4Rec πριν και μετά την εκπαίδευση, χρησιμοποιώντας τη μέθοδο t-SNE [65] για 30.000 τυχαία δείγματα τραγουδιών. Κάθε σημείο χρωματίζεται ανάλογα με τον καλλιτέχνη του κομματιού, ενώ για λόγους ευκρίνειας παρουσιάζουμε μόνο τα τραγούδια των 20 πιο συχνά εμφανιζόμενων καλλιτεχνών του συνόλου δεδομένων. Τα γραφήματα που προκύπτουν (Σχήματα 0.0.7, 0.0.8) δείχνουν πώς οι διαφορετικές μέθοδοι αρχικοποίησης δομούν το μουσικό περιεχόμενο και πώς αυτή η δομή μεταβάλλεται μετά την εκπαίδευση. Το υπόμνημα κάτω από τα γραφήματα δείχνει την αντιστοίχιση μεταξύ των 20 πιο δημοφιλών καλλιτεχνών και των χρωμάτων που χρησιμοποιούνται στα γραφήματα.

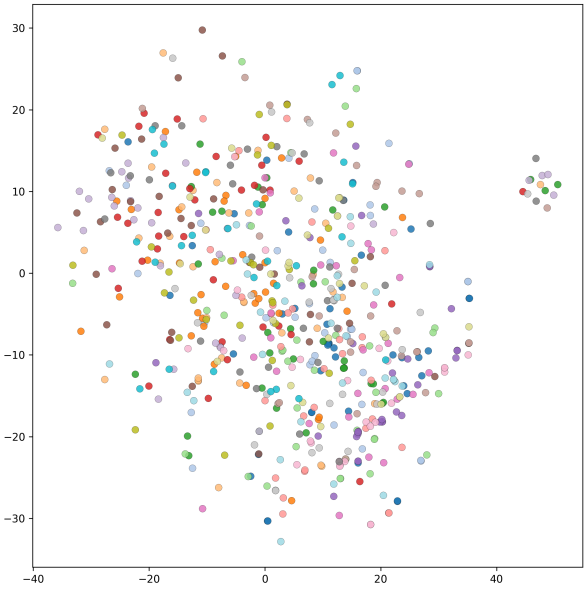
Πριν και μετά την εκπαίδευση Και στα δύο προεκπαιδευμένα μοντέλα παρατηρούμε ότι η συνολική γεωμετρία του διανυσματικού χώρου παραμένει σε μεγάλο βαθμό σταθερή μετά την εκπαίδευση. Αυτό υποδεικνύει ότι ο Transformer μαθαίνει κυρίως να μοντελοποιεί τις ακολουθίες συμπεριφοράς πάνω στις υπάρχουσες αναπαραστάσεις, χωρίς να τις αναδομεί πλήρως. Ωστόσο, τοπικές πυκνώσεις είναι ορατές, με ορισμένες συστάδες καλλιτεχνών να γίνονται ελαφρώς πιο συμπαγείς. Αυτό δείχνει ότι, παρότι η γενική τοπολογία διατηρείται, το μοντέλο πραγματοποιεί λεπτές προσαρμογές που δεν είναι ορατές στην κλίμακα του t-SNE αλλά επηρεάζουν την απόδοση.

MusiCNN έναντι MERT Τα embeddings του MusiCNN εμφανίζουν ήδη πριν από την εκπαίδευση μια σαφή οργάνωση σε επίπεδο καλλιτέχνη, ενώ αναπαριστά κοντά καλλιτέχνες που ανήκουν σε παραπλήσια μουσικά είδη. Για παράδειγμα, στην επάνω δεξιά περιοχή του Σχήματος 0.0.8c, παρατηρούμε ότι οι συστάδες των *Metallica*, *Iron Maiden*, *Motörhead* και *In Flames*, που ανήκουν σε συναφή είδη, εμφανίζονται κοντά στον χώρο αναπαραστάσεων. Μετά την εκπαίδευση, οι συστάδες αυτές γίνονται ελαφρώς πιο πυκνές, διατηρώντας όμως τη δομή τους—δείχνοντας ότι το μοντέλο ενισχύει υπάρχοντα μοτίβα χωρίς να μαθαίνει ριζικά νέες σχέσεις. Η συμπεριφορά αυτή συνάδει με την επιβλεπόμενη προεκπαίδευση του MusiCNN στο πρόβλημα της ταξινόμησης ετικετών, όπου το δίκτυο ενθαρρύνεται να σχηματίζει διακριτά όρια που αντικατοπτρίζουν κατηγορίες όπως είδος ή χροιά, αντί για συνεχείς έννοιες μουσικής ομοιότητας.

Αντίθετα, το MERT παράγει έναν πιο διάχυτο και ομαλό χώρο, με μεγαλύτερη επικάλυψη μεταξύ καλλιτεχνών. Αν και εξακολουθούν να παρατηρούνται τοπικές συστάδες, η συνολική γεωμετρία είναι πιο συνεχής και λιγότερο διακριτή σε επίπεδο καλλιτέχνη. Αυτό δικαιολογείται από την αυτοεπιβλεπόμενη προεκπαίδευση του MERT, που στοχεύει στην εκμάθηση λεπτής μουσικής ομοιότητας κατά μήκος φασματικών και αρμονικών διαστάσεων, αντί για το διαχωρισμό προκαθορισμένων ετικετών. Αυτές οι θεμελιώδεις διαφορές θα μπορούσαν να ερμηνεύσουν και τα ποσοτικά αποτελέσματα: η διακριτή, βασισμένη σε ετικέτες αναπαράσταση του MusiCNN μπορεί να διαχωρίζει ευρείες μουσικές κατηγορίες, αλλά δυσκολεύεται στη διάκριση σε επίπεδο τραγουδιού που απαιτείται από το T4Rec, ενώ ο συνεχής, βασισμένος στη μουσική ομοιότητα χώρος του MERT προσαρμόζεται ευκολότερα στη μοντελοποίηση ακολουθιών, αν και παραμένει λιγότερο αποτελεσματικός από το τυχαίο μοντέλο αναφοράς που βελτιστοποιείται απευθείας για πρότυπα συμπεριφοράς χρηστών.

Τυχαία Αρχικοποίηση Για την τυχαία αρχικοποίηση (Σχήμα 0.0.7), οπτικοποιούμε μόνο τον χώρο αναπαραστάσεων μετά την εκπαίδευση, καθώς τα αρχικά τυχαία embeddings των αντικειμένων προφανώς δεν περιέχουν κάποια ουσιαστική δομή. Όπως αναμενόταν, η προβολή t-SNE δείχνει έναν διάχυτο και ανοργάνωτο χώρο χωρίς σαφείς διακρίσεις σε επίπεδο καλλιτέχνη. Παρ' όλα αυτά, το τυχαίο μοντέλο επιτυγχάνει τη συνολικά καλύτερη απόδοση ανάμεσα σε όλα τα μοντέλα του T4Rec. Η παρατήρηση αυτή αναδεικνύει τη διαφορά μεταξύ της οπτικά ερμηνεύσιμης δομής και της χρησιμότητας στο τελικό πρόβλημα: παρότι τα τυχαία embeddings δεν παρουσιάζουν σημασιολογική οργάνωση στην οπτικοποίηση, είναι πλήρως βελτιστοποιημένα ώστε να αναπαριστούν τα πρότυπα συν-εμφάνισης που απορρέουν από τη συμπεριφορά των χρηστών κατά την εκπαίδευση. Με άλλα λόγια, ο Transformer μαθαίνει μια γεωμετρία αναπαραστάσεων που διαμορφώνεται αποκλειστικά από τη συμπεριφορά ακρόασης και όχι από ακουστική ομοιότητα, επιτρέποντάς του να αποτυπώνει τις λεπτές μεταβάσεις μεταξύ τραγουδιών που χαρακτηρίζουν τις συνεδρίες ακρόασης των χρηστών. Αυτό εξηγεί γιατί ένα μοντέλο εκπαιδευμένο από το μηδέν σε δεδομένα συμπεριφοράς μπορεί να ξεπεράσει εκείνα που αρχικοποιούνται με προεκπαιδευμένα ηχητικά embeddings, των οποίων η μουσικά εμπνευσμένη δομή ίσως να περιορίζει τη διαδικασία βελτιστοποίησης και δεν ευθυγραμμίζεται πλήρως με τη δυναμική των αλληλεπιδράσεων των χρηστών.

Συνολικά, οι οπτικοποιήσεις δείχνουν ότι τα προεκπαιδευμένα embeddings φέρουν ισχυρές επαγωγικές



Σχήμα 0.0.7: Οπτικοποίηση t-SNE του χώρου embeddings του τυχαία αρχικοποιημένου μοντέλου μετά την εκπαίδευση.

προκαταλήψεις (inductive biases)—το MusiCNN προς το διαχωρισμό κατηγοριών, το MERT προς την ακουστική ομοιότητα και το μοντέλο καλλιτέχνη προς τη διακριτή ταυτότητα— ενώ το τυχαίο μοντέλο αναφοράς παραμένει πλήρως προσανατολισμένο στα δεδομένα ακρόασης. Η εκπαίδευση βελτιστοποιεί αυτούς τους χώρους μόνο ελάχιστα, γεγονός που υποδηλώνει ότι η κύρια προσαρμογή των αναπαραστάσεων πραγματοποιείται στα στρώματα του Transformer και όχι στα ίδια τα embeddings. Τα ευρήματα αυτά συνάδουν με τα ποσοτικά αποτελέσματα: τα προεκπαιδευμένα embeddings αποτυπώνουν ουσιαστικές μουσικές δομές, αλλά η σχετική ακαμψία τους περιορίζει την απόδοση στο T4Rec, όπου η ευελιξία και η λεπτομερής διάκριση επιπέδου τραγουδιού είναι καθοριστικής σημασίας.

Μελέτη Αφαίρεσης

Από την ανάλυση που προηγήθηκε διαφαίνεται μια σύνδεση ανάμεσα στη δυνατότητα διάκρισης κατηγοριών υψηλού επιπέδου και την απόδοση του μοντέλου στην παραγωγή συστάσεων. Ωστόσο, θα ήταν βιαστικό να συναχθεί αιτιώδης σχέση μεταξύ αυτών των ιδιοτήτων, καθώς τα δύο εξεταζόμενα πλαίσια διαφέρουν σε πολλά επιπλέον σημεία πέρα από τη διατύπωση του προβλήματος. Για να κατανοήσουμε καλύτερα ποιοι από αυτούς τους παράγοντες ενδέχεται να συμβάλλουν στις αποκλίνουσες τάσεις απόδοσης που παρατηρούνται μεταξύ των T4Rec και RTA, πραγματοποιούμε μια μελέτη αφαίρεσης (ablation study), στην οποία τροποποιούμε συστηματικά συγκεκριμένα στοιχεία T4Rec ώστε να ευθυγραμμιστούν με εκείνα του RTA.

Προεπεξεργασία Δεδομένων

Στο πρώτο πείραμα, εφαρμόσαμε το βήμα προεπεξεργασίας του RTA, το οποίο αφαιρεί τα διπλότυπα τραγούδια από κάθε συνεδρία πριν από την εκπαίδευση. Όπως περιγράψαμε στην αντίστοιχη ενότητα, η τροποποίηση αυτή εξαλείφει τις επαναλαμβανόμενες εμφανίσεις τραγουδιών μέσα στην ίδια συνεδρία, κρατώντας μόνο την πρώτη αλληλεπίδραση με κάθε τραγούδι. Μετά την εφαρμογή αυτού του βήματος στο T4Rec, εκπαιδεύσαμε δύο μοντέλα με βάση αυτή τη ρύθμιση: ένα μοντέλο με αρχικοποίηση MusiCNN και ένα τυχαία αρχικοποιημένο μοντέλο αναφοράς με την ίδια διάσταση embeddings ($d = 200$), και τα δύο ακολουθώντας τις υπερπαραμέτρους του T4Rec. Οι αντίστοιχες μετρικές στο test set παρουσιάζονται στον Πίνακα 3.

Όπως φαίνεται από τα αποτελέσματα, τόσο το τυχαίο μοντέλο όσο και το μοντέλο με αρχικοποίηση MusiCNN εμφανίζουν αισθητή πτώση σε όλες τις μετρικές αξιολόγησης όταν εφαρμόζεται η απαλοιφή διπλοτύπων (0.2333 έναντι 0.2629 NDCG@20 για το Random, 0.1537 έναντι 0.2358 NDCG@20 για το MusiCNN), κάτι που συμφωνεί με τις χαμηλότερες τιμές που παρατηρούνται στα αποτελέσματα του RTA (Πίνακας 2). Η μείωση αυτή

στην απόδοση μπορεί να αποδοθεί στην απώλεια χρήσιμης πληροφορίας στα δεδομένα εισόδου. Ωστόσο, παρά την πτώση της απόδοσης, η σχετική κατάταξη μεταξύ των δύο μοντέλων παραμένει αμετάβλητη, με το τυχαίο μοντέλο να εξακολουθεί να υπερέχει του προεκπαιδευμένου. Αυτό δείχνει ότι οι διαφορές στην προεπεξεργασία μεταξύ των δύο πλαισίων δεν ευθύνονται για τα αντίθετα μοτίβα απόδοσης που παρατηρήθηκαν προηγουμένως.

Πίνακας 3: Σύγκριση της απόδοσης μοντέλων T4Rec που ακολουθούν τη διαδικασία προεπεξεργασίας δεδομένων του RTA

Αρχικοποίηση	**d**	**NDCG@10**	**MAP@10**	**Recall@10**	**NDCG@20**	**MAP@20**	**Recall@20**
Random	200	0.2214	0.1927	0.3112	0.2333	0.1959	0.3583
MusiCNN	200	0.1435	0.1227	0.2093	0.1537	0.1254	0.2498

Επιλογή Υπερπαραμέτρων

Στο δεύτερο πείραμα, στοχεύουμε να απομονώσουμε την επίδραση των υπερπαραμέτρων αρχιτεκτονικής και βελτιστοποίησης κάθε πλαισίου. Για τον σκοπό αυτό, εκπαιδεύσαμε τα μοντέλα T4Rec χρησιμοποιώντας τις υπερπαραμέτρους του RTA, διατηρώντας όμως τα βήματα προεπεξεργασίας δεδομένων, καθώς και τους στόχους εκπαίδευσης και αξιολόγησης του αρχικού T4Rec. Τα προκύπτοντα μοντέλα T4Rec διαθέτουν περισσότερες παραμέτρους και ακολουθούν ένα ουσιαστικά διαφορετικό σχέδιο βελτιστοποίησης (διαφορετικό βελτιστοποιητή, ρυθμιστή ρυθμού μάθησης και αρχικό ρυθμό μάθησης). Τα αποτελέσματα των μοντέλων αυτών στο test set παρουσιάζονται στον Πίνακα 4.

Από τα αποτελέσματα αυτά, παρατηρούμε μια σαφή μεταβολή στην κατάταξη των μοντέλων, καθώς όλα τα προεκπαιδευμένα μοντέλα ξεπερνούν πλέον τα αντίστοιχα μοντέλα αναφοράς. Πιο συγκεκριμένα, τα τυχαία μοντέλα εμφανίζουν μείωση απόδοσης σε σύγκριση το αρχικό πείραμά (Πίνακας 1), ενώ τα MusiCNN, MusiCNN-tuned και το μοντέλο καλλιτέχνη παρουσιάζουν αξιοσημείωτη βελτίωση με τις νέες υπερπαραμέτρους. Αντίθετα, το MERT είναι το μόνο προεκπαιδευμένο μοντέλο που δεν ωφελείται από αυτή την τροποποίηση. Ενδιαφέρον παρουσιάζει το γεγονός ότι σε πρόσθετα πειράματα, όπου απομονώσαμε ξεχωριστά τις υπερπαραμέτρους που αφορούν την αρχιτεκτονική ή τη βελτιστοποίηση, τα περισσότερα μοντέλα απέδωσαν χειρότερα, γεγονός που υποδηλώνει ότι τα παρατηρούμενα οφέλη προκύπτουν από τον συνδυασμό της αρχιτεκτονικής του μοντέλου με το σχέδιο βελτιστοποίησης, και όχι από κάποιον παράγοντα μεμονωμένα.

Με βάση τις παραπάνω παρατηρήσεις, μπορούμε να συμπεράνουμε ότι οι αρχιτεκτονικές και βελτιστοποιητικές επιλογές επηρεάζουν σημαντικά τόσο την ικανότητα του T4Rec να αξιοποιεί προεκπαιδευμένες ηχητικές αναπαραστάσεις όσο και τη δυνατότητά του να μαθαίνει αποτελεσματικά αναπαραστάσεις αντικειμένων από το μηδέν. Ακόμη πιο σημαντικό είναι ότι η κατάταξη των μοντέλων που παρατηρείται εδώ συμφωνεί με εκείνη του RTA, γεγονός που υποδεικνύει ότι η διαμόρφωση των υπερπαραμέτρων πιθανότατα συμβάλλει στα αντίθετα μοτίβα απόδοσης μεταξύ των δύο πλαισίων.

Πίνακας 4: Σύγκριση της απόδοσης μοντέλων T4Rec που εκπαιδεύτηκαν με τις υπερπαραμέτρους του RTA

Αρχικοποίηση	**d**	**NDCG@10**	**MAP@10**	**Recall@10**	**NDCG@20**	**MAP@20**	**Recall@20**
Random	200	0.2449	0.2056	0.3658	0.2579	0.2090	0.4167
MusiCNN	200	0.2511	0.2103	0.3769	0.2637	0.2137	0.4263
MusiCNN-tuned	200	**0.2657**	**0.2274**	**0.3844**	**0.2781**	**0.2307**	**0.4333**
Random	768	0.2458	0.2067	0.3672	0.2587	0.2101	0.4179
MERT	768	0.2470	0.2069	0.3712	0.2600	0.2103	0.4220
Random	1280	0.2008	0.1607	0.3237	0.2140	0.1642	0.3756
Artist model	1280	0.2601	0.2205	0.3830	0.2729	0.2239	0.4333

Συζήτηση

Σε αυτή την ενότητα συζητάμε τις γενικότερες διαφορές μεταξύ των Transformers4Rec και RTA, αναλύοντας πώς οι διαφορετικές διατυπώσεις των προβλημάτων και οι στόχοι εκμάθησης εξηγούν τις αντίθετες τάσεις που παρατηρήθηκαν στην απόδοσης μεταξύ των δύο πλαισίων. Συνολικά, η ανάλυση αυτή προσφέρει μια

πιο ολοκληρωμένη εικόνα της αλληλεπίδρασης μεταξύ των προεκπαιδευμένων ηχητικών αναπαραστάσεων, της αρχιτεκτονικής του μοντέλου και του στόχου εκπαίδευσης, βοηθώντας μας να κατανοήσουμε όχι μόνο ποιοι συνδυασμοί αποδίδουν καλύτερα, αλλά και γιατί παρουσιάζουν τη συγκεκριμένη συμπεριφορά.

Διαφορές στη Διατύπωση του Προβλήματος Παρά την εμφανή αρχιτεκτονική τους ομοιότητα, το πρόβλημα πρόβλεψης του επόμενου τραγουδιού, όπως ορίζεται στο πλαίσιο των Transformers4Rec [100] και RTA [7], εμπεριέχει ουσιώδεις διαφορές. Στην περίπτωση του Transformers4Rec, δεδομένης μιας λίστας αναπαραγωγής οποιουδήποτε μήκους, ο στόχος είναι η συμπλήρωση του ελλείποντος τραγουδιού σε μια κατά τα άλλα πλήρη λίστα. Αντίθετα, το RTA ορίζει ως Αυτόματη Συνέχιση Λίστας Αναπαραγωγής (APC) τη σαφώς δυσκολότερη διαδικασία εύρεσης κατάλληλων επεκτάσεων για μια λίστα επαρκούς μήκους, με επαναληπτικό τρόπο.

Από τους Πίνακες 1 και 2 παρατηρούμε ότι, ανεξαρτήτως της μεθόδου αρχικοποίησης των embeddings (τυχαία, βασισμένη σε μεταδεδομένα ή βασισμένη σε ηχητικό σήμα), το T4Rec αποδίδει υψηλότερα σε όλες τις εξεταζόμενες μετρικές. Λαμβάνοντας υπόψη τις παραπάνω διαφορές στη φύση του προβλήματος, καθώς και τα επιπλέον βήματα επεξεργασίας δεδομένων που πραγματοποιούνται στο RTA (όπως η αφαίρεση διπλοτύπων), τα οποία οδηγούν σε πιο απαιτητικές αρχικές λίστες, αλλά και το γεγονός ότι και στα δύο πλαίσια διατηρήσαμε όσο το δυνατόν πιο πιστά τις αρχικά προτεινόμενες υπερπαραμέτρους, αποδίδουμε τη διαφορά απόδοσης στη μεγαλύτερη δυσκολία του προβλήματος APC, όπως ορίζεται στο [7].

Συγκριτική Απόδοση Μοντέλων Οι διαφορετικές διατυπώσεις των προβλημάτων επιφέρουν επίσης διαφοροποιημένα αποτελέσματα για κάθε μέθοδο αρχικοποίησης αναπαραστάσεων. Το MERT [63], για παράδειγμα, είναι το προεκπαιδευμένο μοντέλο που προσεγγίζει περισσότερο την απόδοση της τυχαίας αρχικοποίησης στο πλαίσιο του Transformers4Rec, ενώ παρουσιάζει πτώση απόδοσης στην –πιο ευνοϊκή για τις ηχητικές αναπαραστάσεις –περίπτωση του RTA. Αντίστροφα, το μοντέλο που εκπαιδεύτηκε ώστε να φέρνει κοντά αποσπάσματα ήχου του ίδιου καλλιτέχνη εμφανίζει τη μεγαλύτερη βελτίωση έναντι της τυχαίας αρχικοποίησης στο RTA, αλλά παρουσιάζει τη χειρότερη επίδοση μεταξύ των προεκπαιδευμένων μοντέλων στο Transformers4Rec. Τέλος, η μόνη συμπεριφορά που εμφανίζει σταθερά θετική επίδραση στην απόδοση των αναπαραστάσεων και στα δύο πλαίσια είναι η προσαρμογή (tuning) των προεκπαιδευμένων αναπαραστάσεων στο προς επίλυση πρόβλημα, μέσω μιας απλής αντιθετικής διαδικασίας [82], όπως φαίνεται στην περίπτωση του MusiCNN [80]. Ωστόσο, η διαδικασία αυτή εισάγει ήδη πληροφορία σχετική με τον τελικό στόχο κατά την προσαρμογή των αναπαραστάσεων.

Συμπεράσματα

Η παρούσα εργασία διερεύνησε το κατά πόσο οι προεκπαιδευμένες ηχητικές αναπαραστάσεις μπορούν να ενισχύσουν την απόδοση ενός μοντέλου συστάσεων βασισμένου σε συνεδρίες ακρόασης. Το βασικό μας κίνητρο προέρχεται από το γεγονός ότι, παρά την αποδεδειγμένα υψηλή απόδοση των προεκπαιδευμένων μοντέλων σε διάφορα προβλήματα MIR [14], [63], [80], [115], η χρήση τους σε συστήματα συστάσεων παραμένει ανεξερεύνητη. Ειδικά στην περίπτωση των συνδεριών ακρόασης, όπου η διαθέσιμη πληροφορία είναι πιο περιορισμένη, η ιδέα ενσωμάτωσης αναπαραστάσεων περιεχομένου που κωδικοποιούν μουσικές ιδιότητες φαίνεται άξια διερεύνησης. Για να ελέγξουμε την προσέγγισή μας, εξαγάγαμε αναπαραστάσεις τραγουδιών από τρία προεκπαιδευμένα μοντέλα, συγκεκριμένα τα MusiCNN [80], MERT [63] και ένα μοντέλο βασισμένο σε ομοιότητα καλλιτεχνών, προκειμένου να μετρήσουμε την επίδραση διαφορετικών μεθόδων προεκπαίδευσης στην απόδοση του τελικού μοντέλου. Χρησιμοποιώντας συνεδρίες ακρόασης που εξαγάγαμε από το σύνολο δεδομένων Music4All [91], εκπαιδεύσαμε μοντέλα σύστασης βασισμένα σε αρχιτεκτονικές Transformers, ακολουθώντας δύο υπάρχουσες προσεγγίσεις από τη βιβλιογραφία: το Transformers4Rec [100] (επικεντρωμένο στην απλούστερη εργασία πρόβλεψης του επόμενου τραγουδιού) και το Represent-Then-Aggregate (RTA) [7] (που αντιμετωπίζει το δυσκολότερο πρόβλημα της συνέχισης λίστας αναπαραγωγής), τα οποία προσαρμόσαμε στην περίπτωση των συνεδριών ακρόασης αντί για λίστες αναπαραγωγής. Αντί να μάθουμε τις αναπαραστάσεις των αντικειμένων από το μηδέν, βασιζόμενοι αποκλειστικά σε δεδομένα αλληλεπιδράσεων, εξετάσαμε τρόπους ενσωμάτωσης των ηχητικών αναπαραστάσεων στη ροή της παροχής συστάσεων. Πιο συγκεκριμένα, τα χρησιμοποιήσαμε ως εκπαιδεύσιμη αρχικοποίηση του πίνακα αναπαραστάσεων, λειτουργώντας ως ένα σημείο εκκίνησης που ενσωματώνει πληροφορίες για το περιεχόμενο. Σε επιπλέον πειράματα, πραγματοποιήσαμε προσαρμογή των ηχητικών αναπαραστάσεων με ένα απλό μοντέλο, ως μια απλή τεχνική προσαρμογής πεδίου (domain adaptation), ενώ εξετάσαμε και μια στρατηγική ύστερης συνένωσης, όπου τα προεκπαιδευμένα embeddings ενσωματώθηκαν μετά

τα επίπεδα του Transformer μέσω συνένωσης σε επίπεδο συνεδρίας.

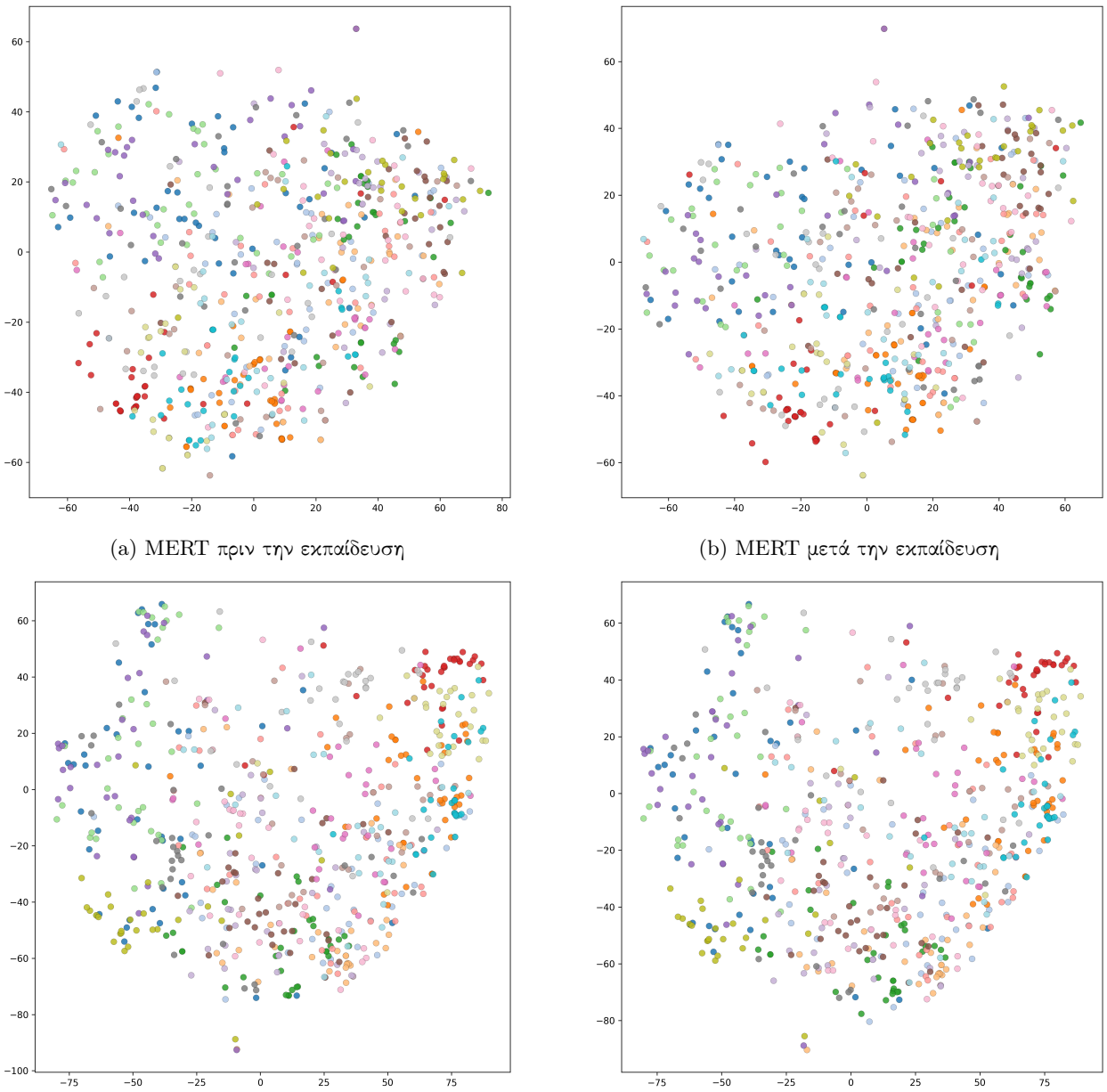
Τα πειράματά μας και στα δύο πλαίσια παρείχαν συμπληρωματικές παρατηρήσεις. Στην περίπτωση του Transformers4Rec, όπου η παραγωγή συστάσεων διατυπώνεται ως πρόβλημα πρόβλεψης ενός ελλείποντος τραγουδιού, καμία από τις εξεταζόμενες στρατηγικές αρχικοποίησης που βασίζονται σε προεκπαιδευμένες αναπαραστάσεις δεν φάνηκε να ξεπερνά την τυχαία αρχικοποίηση. Το εύρημα αυτό υποδηλώνει ότι, παρά το εμφανές πλεονέκτημα των μοντέλων που ενσωματώνουν ηχητική πληροφορία στις πρώτες εποχές της εκπαίδευσης, το μοντέλο μπορεί τελικά να μάθει την απαιτούμενη πληροφορία εξ ολοκλήρου από το μηδέν. Από την άλλη, παρά την (σύμφωνη με τη βιβλιογραφία [49], [55]) ανώτερη απόδοση της εμπλουτισμένης με μεταδεδομένα παραγοντοποίησης του πίνακα συν-εμφάνισης χρηστών–αντικειμένων για την αρχικοποίηση των embeddings, οι ηχητικές αναπαραστάσεις μπορούν να παρέχουν χρήσιμη πληροφορία για το πρόβλημα συνέχισης συνεδριών ακρόασης, στο πιο απαιτητικό πλαίσιο του RTA (χωρίς διπλότυπα τραγουδιών και με περιορισμένα συμφραζόμενα). Και στις δύο περιπτώσεις, τα πειράματά μας ανέδειξαν μια μικρή αλλά σταθερή θετική επίδραση της προσαρμογής των ηχητικών embeddings με χρήση δεδομένων συμπεριφοράς, ως μια μέθοδος προσαρμογής πεδίου (domain adaptation).

Σε μια προσπάθεια να κατανοήσουμε καλύτερα τις αντιφατικές επιδόσεις των προεκπαιδευμένων μοντέλων μεταξύ των δύο πλαισίων, αρχικά οπτικοποιήσαμε τους αντίστοιχους χώρους αναπαραστάσεων και παρατηρήσαμε μια πιθανή συσχέτιση ανάμεσα στην ικανότητα κάθε μοντέλου να σχηματίζει συστάδες σχετικές με μουσικά χαρακτηριστικά και στην απόδοσή του στον τομέα των συστάσεων. Στη συνέχεια πραγματοποιήσαμε μια επιπλέον μελέτη (ablation study), όπου απομονώσαμε την επίδραση διαφόρων παραγόντων που διαφοροποιούν τα δύο πλαίσια, αναδεικνύοντας την επίδραση της ρύθμισης των υπερπαραμέτρων στα πειραματικά αποτελέσματα.

Με βάση τα παραπάνω, θεωρούμε ότι η χρησιμότητα των προεκπαιδευμένων ηχητικών αναπαραστάσεων σε εφαρμογές συστάσεων ενδέχεται να εξαρτάται από τον βαθμό ευθυγράμμισης του στόχου προεκπαίδευσης με το τελικό πρόβλημα παραγωγής συστάσεων, ενώ οι κατάλληλες επιλογές αρχιτεκτονικής και υπερπαραμέτρων βελτιστοποίησης είναι απαραίτητες για την αξιοποίηση του πλήρους δυναμικού τους. Τέλος, δεν προέκυψε σαφές συμπέρασμα σχετικά με τη σχετική καταλληλότητα των διαφορετικών τεχνικών προεκπαίδευσης (αντιθετικών, αυτοεπιβλεπόμενων ή εποπτευμένων μέσω ετικετών).

Συνολικά, οι βασικές συνεισφορές της παρούσας εργασίας συνοψίζονται ως εξής:

- Πραγματοποιούμε συστηματική αξιολόγηση της απόδοσης προεκπαιδευμένων ηχητικών αναπαραστάσεων σε δύο υπάρχοντα πλαίσια συστάσεων: Transformers4Rec [100] και Represent-Then-Aggregate (RTA) [7].
- Εξάγουμε ηχητικές αναπαραστάσεις από τρία προεκπαιδευμένα μοντέλα: MERT [63], MusiCNN [80] και ένα μοντέλο που εστιάζει στην ταυτότητα καλλιτέχνη. Κάθε μοντέλο ακολουθεί διαφορετική στρατηγική προεκπαίδευσης (ακουστική μοντελοποίηση, επιβλεπόμενη πρόβλεψη ετικετών, αντιθετική εκμάθηση αντίστοιχα).
- Πειραματιζόμαστε με διάφορους τρόπους ενσωμάτωσης των embeddings στη ροή σύστασης, συμπεριλαμβανομένης μιας τεχνικής προσαρμογής πεδίου (domain adaptation), όπου τα προεκπαιδευμένα embeddings υφίστανται προσαρμόζονται σε δεδομένα ακρόασης μέσω ενός απλού αντιθετικού στόχου.
- Τα πειράματά μας ανέδειξαν σημαντικά διαφορετική κατάταξη μοντέλων ανάλογα με το πλαίσιο. Για να ερμηνεύσουμε αυτά τα αντιφατικά αποτελέσματα, χρησιμοποιήσαμε μία τεχνική οπτικοποίησης αναπαραστάσεων [65] και πραγματοποιήσαμε μία επιπλέον μελέτη (ablation study), απομονώνοντας επιμέρους διαφορές των δύο πλαισίων.
- Τα ευρήματά μας αναδεικνύουν την επίδραση τόσο της στρατηγικής προεκπαίδευσης όσο και της επιλογής υπερπαραμέτρων στη χρησιμότητα του κάθε ηχητικού μοντέλου στο πεδίο των συστάσεων.



(a) MERT πριν την εκπαίδευση

(b) MERT μετά την εκπαίδευση

(c) MusiCNN πριν την εκπαίδευση

(d) MusiCNN μετά την εκπαίδευση

Σχήμα 0.0.8: Οπτικοποιήσεις t-SNE των χώρων embeddings του MERT και του MusiCNN πριν και μετά την εκπαίδευση.

Chapter 1

Introduction

1.1 Music Recommender Systems

Over the past years, streaming platforms have redefined the way we enjoy music. These services have largely replaced physical media, such as CDs and vinyl records, and offer ever-growing song catalogs to their millions of users. However, since it is not feasible for users to exhaustively navigate these endless catalogs, one of the main challenges that arise is how to bring users closer to music they love. This is the purpose of music recommender systems, which have become a cornerstone of digital streaming platforms, shaping how millions of listeners discover new music and engage with their favorite artists.

1.1.1 Collaborative filtering

For many years, recommendation systems relied on techniques that produce recommendations by exploiting patterns in user-item interaction data. Such algorithms have been widely adopted and have demonstrated considerable success in many domains, including music. A typical example is a technique called Collaborative Filtering (CF), which essentially identifies similarities between users or items based on large interaction matrices and recommends tracks that users with similar taste have enjoyed [92]. Interaction data can range from 5-star ratings, explicitly expressing a user's like/dislike for an item, to other forms of feedback like play counts of songs, purchases of items, and similar implicit signals, which reflect user preferences in a more indirect way [48]. Alternatively, a family of CF algorithms employs matrix factorization techniques in order to represent users and items in a shared low-dimensional latent space and then discover patterns in user preferences [36], [43], [54]. This not only improves recommendation accuracy but also enhances scalability and computational efficiency.

Limitations

Despite their proven effectiveness, traditional approaches have several limitations. One major issue is data scarcity, since user–item interaction matrices are typically very sparse and contain only a small fraction of possible interactions. Consequently, it becomes more challenging to infer meaningful patterns based exclusively on interaction data. In its extreme form, data scarcity gives rise to the so-called "cold-start" problem, which occurs when new users or items are added to the system and remains one of the grand challenges in recommender systems [95]. Due to the lack of interaction data, it becomes difficult to recommend items to a new user or recommend a new item to existing users. As a result, these methods hinder emerging artists and niche genres from gaining visibility and tend to reinforce popularity bias, where popular, heavily-rated tracks are recommended more frequently, while lesser-known content remains overlooked. This not only raises a concern of unfairness in recommender systems, but also limits diversity in recommendations, leading to a more homogenized listening experience [34], [56].

1.1.2 Content-Based Methods

To tackle this, researchers have increasingly turned to content-based methods that define similar items based on features derived from the content, rather than user behavior data. Such features can range from low-level information extracted directly from the audio signal to more high-level textual descriptors, such as metadata or user-generated tags [27] and are available even in the absence of user–item interactions, which makes them especially useful for newly added or rarely played tracks. By incorporating this type of information into the recommendation process, systems can make suggestions even for tracks with little or no historical listening data, thus mitigating the cold-start problem. In practice, many real-world systems integrate both collaborative and content-based signals to combine their respective advantages.

When it comes to features extracted directly from raw audio, a simple solution is to use handcrafted descriptors [25], [120], designed to capture timbral, rhythmic, or spectral characteristics of the signal. While such features can be useful, they are inherently limited in their ability to represent the rich, hierarchical structure of music. For this reason, more recent works shifted toward leveraging deep neural networks to automatically learn representations that map the audio signal into a high-dimensional embedding space capturing meaningful musical properties. Many of these methods follow a supervised learning paradigm [49], [60], [67], [80], requiring large labeled datasets to guide the representation learning process. However, since collecting such annotations is both costly and time-consuming, recent works [14], [38], [62], [63], [76], [101], [115] have focused on methods that derive supervisory signals directly from the data itself, without the need for manually

labeled examples–an approach commonly referred to as self-supervised learning. In fact, in the field of Music Information Retrieval (MIR), numerous models have been pretrained on large collections of unlabeled audio and can then be used for several downstream tasks, such as auto-tagging and genre classification. This can be achieved by either fine-tuning the model on the final task or using it directly as a feature extractor with no additional training. Such transfer learning approaches often achieve comparable or even better performance than state-of-the-art task-specific models, and have proven to be particularly useful in cases where labeled data is scarce.

1.1.3 Sequential Recommendation

Despite the remarkable progress in learning powerful audio embeddings, another important dimension of recommendation lies in modeling how user preferences evolve over time. Although it's safe to argue that user preferences can shift over time, conventional models treat user history statically and ignore the temporal dynamics of user behavior. This is even more important in the music domain, since music consumption is an inherently sequential process – users typically listen to music within a playlist, an album or a listening session [95]. To capture these temporal dependencies, sequential recommendation methods extend the collaborative filtering paradigm by modeling user activity as an ordered sequence of interactions. Unlike traditional systems, sequential models learn how recent choices influence future ones, aiming to predict the next item a user is likely to engage with based on their recent history.

Session-based recommendation tasks represent a particular case of this paradigm, where predictions depend solely on the ongoing user session rather than long-term interaction histories. This setting is especially relevant for anonymous or newly registered users, as it allows personalized recommendations even in the absence of persistent user profiles. Common examples include music listening sessions and e-commerce browsing sessions, where short-term context plays a dominant role. Since sequential and session-based recommendations are essentially sequence modeling tasks, it shouldn't come as a surprise that over the past decade most of the prevalent architectures used in Natural Language Processing (NLP) for language modeling, like GRU [20] and Transformers [109], have been applied to those fields with great success [44], [45], [53], [103].

In the music domain, a closely related application is Automatic Playlist Continuation (APC). Undoubtedly, playlists have been an increasingly popular way of consuming music in recent years and users in streaming platforms often create their own playlists based on their favorite music genres, different moods or listening occasions. Therefore, one of the challenges that these platforms have to face is that of automatically extending such lists of music tracks in a way that preserves their musical coherence and fits well with the rest of the list.

1.2 Problem Statement and Research Questions

As mentioned earlier, pretrained models have become increasingly popular in the MIR community, thanks to their competitive performance in various downstream tasks. However, the effectiveness of such models for session-based music recommendation remains underexplored [104]. That being said, the core problem this thesis addresses lies in whether audio embeddings from pretrained models can enhance the performance of session-based music recommendation systems. More specifically, our research aims to extract audio representations from MIR foundation models recently presented in the literature and explore different ways of integrating these representations into the recommendation pipeline. To achieve this, we experiment with two different recommendation frameworks: an NVIDIA library called Transformers4Rec (T4Rec) [100], and a framework recently proposed by Deezer Research, referred to as Represent-Then-Aggregate (RTA) [7]. The architectural differences between these two help us assess the effectiveness of our approach under various setups.

Ultimately, our efforts try to address the following research questions:

- RQ1: Can pretrained audio embeddings improve recommendation performance in session-based scenarios?
- RQ2: Which pretraining technique performs best in this setting (supervised tagging, masked acoustic modeling, contrastive learning)?

- RQ3: How to effectively incorporate pretrained embeddings into the recommendation pipeline?

1.3 Contributions

The contributions of this thesis to the field of sequential music recommendation can be summarized as follows:

- We evaluate the performance of pretrained audio embeddings, utilized in tandem with two recommendation frameworks, T4Rec [100] and RTA [7], which we adapt to the music recommendation setting.
- We compare the following models as embedding sources: MERT [63], MusiCNN [80] and a custom artist-based model. We also explore embedding finetuning methods by employing contrastive learning on the pre-trained embeddings as a domain adaptation technique.
- We process the Music4All [91] dataset, which includes the listening histories of ∼15k Last.fm[1] users, in order to split the histories into listening sessions. Additionally, we use the pretrained models to extract song embeddings from raw audio.
- We investigate initializing the per-item embeddings with the audio embeddings, applying a simple domain adaptation technique, or using them in a late fusion setting.

1.4 Thesis Structure

The remainder of this work is organized into 5 chapters. Each chapter is outlined as follows:

- **Chapter 2: Theoretical Background**

  Familiarizes the reader with the necessary background knowledge in order to seamlessly follow the subsequent chapters. This includes the basic methods used in recommender systems, widely used machine learning models, as well as an overview of audio signal representation techniques and a brief description of evaluation metrics used in recommendation systems.

- **Chapter 3: Related Work**

  Provides a comprehensive analysis of previous works on the two fields that this thesis brings together: representation learning from audio signals, and sequential recommendation, including both general sequential models and more music-related approaches.

- **Chapter 4: Data and Methodology**

  Delves into the data preparation process we followed, together with an explanation of the internal details of each recommendation framework.

- **Chapter 5: Experimental Evaluation**

  Describes the experiments we conducted with the proposed models and analyzes the results.

- **Chapter 6: Conclusion**

  Sums up the results and contributions of our work and discusses potential future work based on our observations.

[1] https://www.last.fm/

Chapter 2

Theoretical Background

2.1 Machine Learning

Machine learning (ML) is a subfield of Artificial Intelligence (AI) that focuses on building models that learn from data in order to make predictions. ML algorithms detect patterns in training examples and exploit them to make informed decisions about unseen data.

2.1.1 Categories of Machine Learning Algorithms

Machine learning techniques can be broadly classified into three main categories depending on the availability of information about the correct prediction:

- **Supervised Learning**: In supervised learning algorithms, the model is presented with labeled data, which means that each input sample x is accompanied by the desired output y. The goal is to learn a function $f(x) \rightarrow y$ that generalizes well to new input data. The predicted output can be either a real value, therefore we have a regression task, or one of predefined, discrete categories, in which case we have a classification task.
- **Unsupervised Learning**: On the other hand, unsupervised methods operate on unlabeled data and try to identify underlying properties and structure of the data. Machine learning tasks that fall in this category include clustering and dimensionality reduction.

  A sub-field of unsupervised learning that has gained increasing attention in the past years is **Self-Supervised Learning (SSL)**, where the model is trained on supervisory signals generated from the data itself, e.g. by masking parts of the input and learning to predict these masked parts.
- **Reinforcement Learning**: In reinforcement learning, the computer program takes actions in an environment, trying to maximize its cumulative reward by deriving an optimal strategy. Typical application fields include autonomous driving cars, robotics and gaming.

2.1.2 Neural Networks

Artificial Neural Networks (ANN) constitute a family of machine learning models that are inspired by the way the human brain works. They consist of numerous units called neurons, which are connected to each other to form layers that process the input in order to make predictions. Before delving into the details of ANNs we will describe how these artificial neurons work.

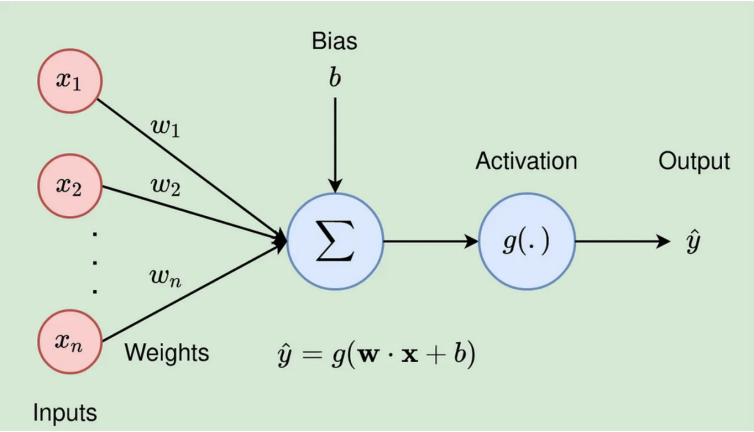


Figure 2.1.1: Strucure of an artificial neuron [30]

The Neuron

Each neuron takes a vector as input and multiplies it with learnable weights that control the contribution of each input value in the final result. To produce the output, the neuron sums the weighted inputs, adds a learnable bias and passes the outcome through an activation function that introduces non-linearities. These operations can be seen in Figure 2.1.1 and are mathematically expressed as:

$$\hat{y} = g\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1.1}$$

where g is the activation function, x is the input vector, of dimensionality n, w is the weight vector and b is the bias.

A well-known early algorithm that relied on a single neuron to perform binary classification is the Perceptron [89], which was introduced by Frank Rosenblatt in 1957. However, it was limited to solving only linearly separable problems, thus motivating the development of more complex multi-layer architectures.

Multi-Layer Networks

In pursuit of more expressive modeling capabilities, architectures that added additional neurons and organized them in layers were developed. Such networks are often called Multi-Layer Perceptrons (MLP) and consist of the following components:

- **Input layer**: Receives raw input data, with each neuron corresponding to an individual feature.
- **Hidden layers**: Intermediate layers containing multiple neurons that learn representations and transformations of the data.
- **Output layer**: Produces the final result, which depends on the nature of the task (e.g., classification or regression).

In the specific case where each neuron in every layer is connected to all the neurons of the previous layer, as in Figure 2.1.2, then we have a fully connected neural network. Additionally, when multiple hidden layers are added, the network becomes a deep neural network (DNN), which allows it to model highly complex relationships in data. Deep networks enable applications such as image recognition, speech processing, and natural language understanding by capturing hierarchical patterns in large datasets.

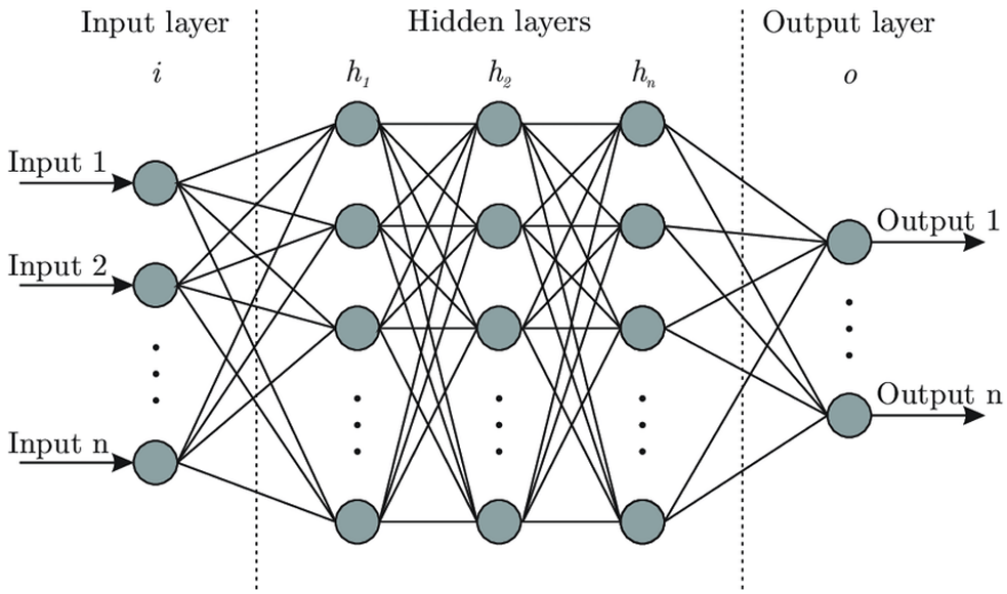


Figure 2.1.2: A Neural Network with 3 hidden layers [12]

Loss Functions

A neural network learns by minimizing the difference between its predicted outputs and the actual ground-truth values. This difference is quantified using a loss function, which provides a measure of error and guides the optimization process. Two widely used loss functions include:

- **Mean Squared Error (MSE)** – Used primarily for regression tasks, this function penalizes large deviations between predictions and true values:

$$L_{\text{MSE}}(\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{2.1.2}$$

  where N is the number of instances, y_i is the actual target value, and $\hat{y}_i$ is the predicted output.

- **Cross-Entropy (CE)** – Used for classification tasks, this function measures the divergence between predicted and actual probability distributions:

$$L_{CE}(\hat{y}_i, y_i) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{2.1.3}$$

  where y_i represents the ground-truth label (one-hot encoded), and $\hat{y}_i$ is the predicted probability of the correct class.

Gradient Descent

To optimize a neural network, we use gradient-based optimization algorithms that iteratively update parameters to minimize the loss function. The most fundamental approach is Gradient Descent (GD), where model parameters are updated iteratively based on computed gradients:

$$w^{(t+1)} = w^{(t)} - \eta\frac{\partial L}{\partial w} \tag{2.1.4}$$

where η is the learning rate, determining the step size, L is the loss value computed on all the available data points and $\frac{\partial L}{\partial w}$ is the gradient of the loss function with respect to the weight parameter, w.

A popular variant, Stochastic Gradient Descent (SGD), updates the weights after computing the gradient on a single randomly selected training sample instead of the entire dataset. This stochasticity introduces variability, often enhancing generalization and computational efficiency, at the cost however of training stability and convergence speed.

Another commonly used approach is Mini-Batch Gradient Descent, which balances the stability of full-batch GD and the efficiency of SGD by computing updates on small data subsets at each step. Furthermore, adaptive methods like Momentum-based GD, Adam, and RMSprop introduce learning rate adjustments to improve convergence speed and stability.

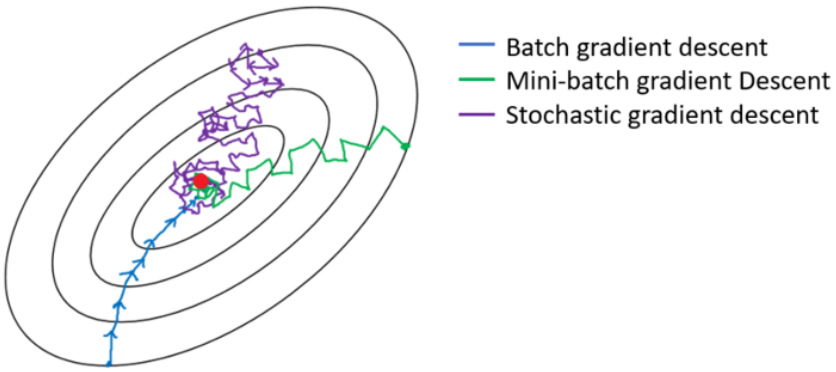


Figure 2.1.3: Optimization paths in a loss landscape [23]. Batch GD (blue): Stable but slow, Stochastic GD (purple): noisy but efficient, Mini-batch GD (green): balances efficiency and stability

Backpropagation

However, in neural networks, finding the relationship between the loss and the weights is not an easy task. To efficiently compute gradients in such cases, we use backpropagation [90], an algorithm that applies the

chain rule of differentiation to propagate errors backward through the network. This enables the systematic computation of the gradient of the loss function with respect to each parameter. The backpropagation process involves two main steps:

- Forward pass – Input data propagates through the network, generating activations and predictions at each layer.
- Backward pass – The error is propagated in reverse from the output layer through preceding layers, computing gradients using the chain rule.

These gradients are then used to update model parameters via gradient descent, progressively refining the network's predictive performance. The combination of backpropagation and gradient-based optimization forms the foundation of modern deep learning, enabling the effective training of large-scale neural architectures.

Activation Functions

As mentioned earlier, an activation function is used to produce the output of each artificial neuron. Its purpose is to apply a non-linear transformation on the input, thus allowing the network to capture complex patterns of the data. In fact, without activation functions, deep fully connected networks are equivalent to simple linear transformations.

- **Sigmoid**: The sigmoid (or logistic) function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.1.5}$$

  This function outputs values in the range (0,1), making it suitable for the output layer of binary classification problems. However, it suffers from the vanishing gradient problem, where gradients become too small in deep networks, slowing down learning.

- **Tanh**: The hyperbolic tangent (tanh) function is defined as:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.1.6}$$

  Unlike sigmoid, tanh outputs values in the range (-1,1), centering the data around zero, which often leads to faster convergence. However, it also suffers from the vanishing gradient problem. The tanh function is commonly used in the hidden layers of neural networks.

- **ReLU**: The Rectified Linear Unit (ReLU) activation function maps all negative values to zero and leaves positive values unaffected:

$$\text{ReLU}(x) = \max(0, x) \tag{2.1.7}$$

  It is also widely used in hidden layers due to its computational efficiency and ability to mitigate the vanishing gradient problem. However, it can suffer from the dying ReLU problem, where neurons output zero for all inputs, leading to a loss of learning capability.

- **Softmax**: The softmax function is used in multi-class classification problems and is defined as:

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.1.8}$$

  The raw scores produced by the final layer of a neural network before applying a transformation can be positive or negative and are not constrained within a specific range. This function converts these scores z_i, which are called logits, into probability distributions over multiple classes, ensuring that the sum of the outputs equals one. It is commonly applied to the output layer of classification networks. An example of using the softmax function for a 5-class classification problem is shown in Figure 2.1.5.

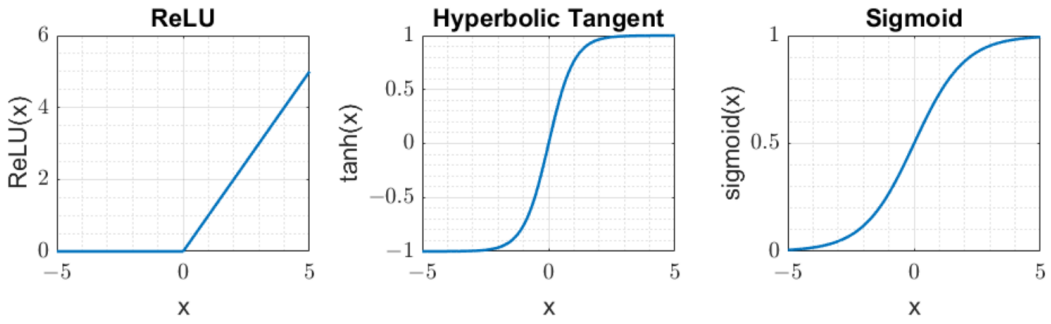


Figure 2.1.4: Activation functions: ReLU, tanh, and sigmoid [51]

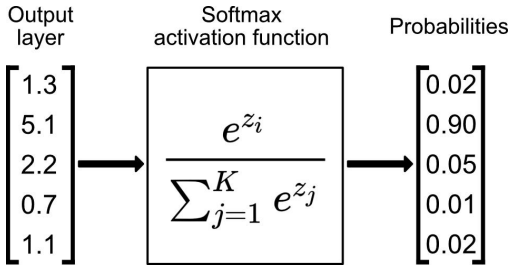


Figure 2.1.5: Example of applying Softmax to a 5-class classification problem [83]

Regularization

Regularization techniques are employed to mitigate overfitting in machine learning models by imposing constraints on the optimization process. Overfitting occurs when a model fits the training data too closely, capturing noise and random fluctuations rather than the true underlying pattern, thereby reducing its ability to generalize to unseen examples. This phenomenon typically arises when a model is excessively complex relative to the amount of available training data. Regularization improves generalization by controlling model complexity, discouraging excessive reliance on specific training instances, and introducing constraints that promote simpler, more robust solutions. This is typically accomplished by adding a regularization term to the loss function, which prevents the weights' magnitude from growing excessively.

L1 Regularization (Lasso) L1 regularization, also known as Lasso (Least Absolute Shrinkage and Selection Operator), adds a penalty term proportional to the absolute values of the model's parameters. It is formulated as:

$$L_{L1} = \lambda \sum_i |w_i| \tag{2.1.9}$$

where λ is a hyperparameter that controls the regularization strength, and w_i represents the model parameters. L1 regularization promotes sparsity by driving some weights to zero, effectively performing feature selection. This makes it particularly useful in scenarios where interpretability and dimensionality reduction are desired.

L2 Regularization (Ridge) L2 regularization, also known as Ridge regression, penalizes the sum of the squared values of the model's parameters. It is calculated as:

$$L_{L2} = \lambda \sum_i w_i^2 \tag{2.1.10}$$

Unlike L1 regularization, L2 does not enforce sparsity but instead shrinks all weights towards zero, preventing

excessively large values that could lead to overfitting. L2 regularization is widely employed in deep learning as it helps maintain smooth and stable weight distributions, improving convergence and generalization.

Dropout Another well-known technique used in neural networks to prevent overfitting is Dropout [102]. It works by randomly deactivating some neurons by setting their output to 0 with probability p ("dropping them out"). This prevents complex co-adaptations of hidden units on the training data and is usually applied to the fully-connected layers of Convolutional Neural Networks (CNNs), which will be described shortly.

Batch Normalization Another widely used technique to stabilize and accelerate training is Batch Normalization (BatchNorm) [50]. BatchNorm operates by normalizing the inputs of each layer using the batch's statistics. That is, for a given activation x in a batch, BatchNorm computes the normalized value as:

$$\hat{x} = \frac{x - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{2.1.11}$$

where μ_B and σ_B^2 are the mean and variance computed over the current batch, and ϵ is a small constant for numerical stability. The normalized output is then scaled and shifted using learnable parameters γ and β: f

$$y = \gamma\hat{x} + \beta \tag{2.1.12}$$

BatchNorm allows for faster convergence and more stable gradients by mitigating the problem of internal covariate shift, where the distribution of each layer's inputs keeps changing during training.

Layer Normalization Layer Normalization (LayerNorm) [2] is a similar normalization strategy, but instead of computing statistics across the batch dimension, it normalizes across the feature dimensions of a single data point. This makes it suitable for tasks where batch statistics are unreliable or inapplicable, such as in recurrent neural networks or transformer-based architectures, which we will describe shortly. Given a feature vector $x \in \mathbb{R}^d$ for a single input, LayerNorm computes:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{2.1.13}$$

where μ and σ^2 are the mean and variance computed across the d features of x. Like BatchNorm, it includes learnable parameters γ and β for scaling and shifting. LayerNorm has become particularly important in attention-based models, where consistent feature scaling is beneficial regardless of batch size.

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specialized class of deep learning models designed for processing structured grid-like data, such as images. Unlike fully connected networks, CNNs employ convolutional layers that leverage spatial hierarchies in data, making them highly effective in feature extraction and pattern recognition tasks. They have significantly advanced the field of computer vision, achieving breakthroughs in tasks such as image classification [42], [57], [59], object detection [40], and image segmentation [64].

Motivation and Structure Traditional neural networks struggle with high-dimensional inputs, as fully connecting all neurons leads to an explosion in the number of parameters. CNNs address this challenge by using convolutional layers, which apply shared weights across local receptive fields, drastically reducing parameter count while preserving spatial relationships.

A CNN typically consists of the following layers, each serving a distinct role in feature extraction and decision-making:

- **Convolutional Layers**: In each of these layers, small learnable matrices called filters (or kernels) are slid over the input and produce feature maps. In the initial convolutional layers, these filters extract low-level features such as edges and textures and as the network deepens, higher layers capture increasingly

complex patterns, including shapes and object structures. This operation is called convolution and is mathematically defined as:

$$(X * W)(i,j) = \sum_m \sum_n X(i-m, j-n)W(m,n) \tag{2.1.14}$$

where $*$ denotes convolution, X is the input, W is the filter (kernel), (i,j) represent the position of the filter on the input and (m,n) are the spatial indices of the filter. Key hyperparameters for convolutional layers include:

- **Kernel Size (F)**: Defines the size of the filter matrix that slides over the input.
- **Stride (S)**: Determines the step size of the filter as it moves across the input.
- **Padding (P)**: Controls whether the input is zero-padded to maintain spatial dimensions.

In this case, and supposing a square input with dimension D, the output dimension of the convolution is equal to:

$$\frac{D - F + 2P}{S} + 1 \tag{2.1.15}$$

An example of convolving a filter with an input image is presented in Figure 2.1.6. Note that each layer usually applies several filters on the input, each one learning a different feature. Non-linear activation functions such as ReLU are applied after convolutions to introduce non-linearities, enabling the network to learn intricate patterns beyond simple linear transformations.

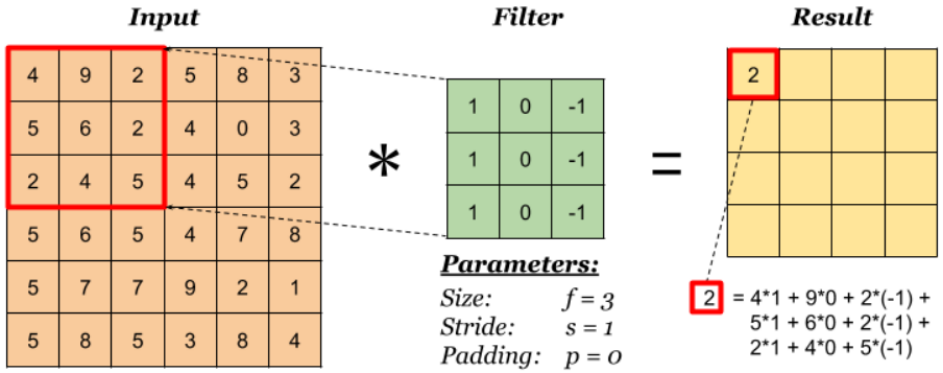


Figure 2.1.6: Convolution between an input image with $D = 6$ and a 3×3 filter [81]. According to Eq. 2.1.15, the output dimension will be equal to 4.

- **Pooling Layers**: These layers perform downsampling to reduce spatial dimensions by applying a specified aggregation function on regions of the feature maps. Pooling helps prevent overfitting by reducing the amount of information passed to subsequent layers, retaining only the most essential features. Two common operations are max pooling, which retains only the maximum value from each local region and average pooling, which computes the average value over the pooling region, as shown in Figure 2.1.7.
- **Fully Connected Layers**: Fully Connected (FC) layers are used to make predictions by combining the features extracted from the convolutional and pooling layers. FC layers transform learned representations into a final decision, such as class probabilities in an image classification task. This is normally done by applying the Softmax activation function (Figure 2.1.8).

Recurrent Neural Networks (RNNs)

A major shortcoming of traditional neural networks is their inability to efficiently handle sequential data of arbitrary length, such as text, audio and video. This challenge is addressed by Recurrent Neural Networks (RNNs) [90], a class of neural networks that process input data sequentially, while maintaining an internal memory of previous inputs by passing a hidden state across time (Figure 2.1.9). More specifically, if t is the

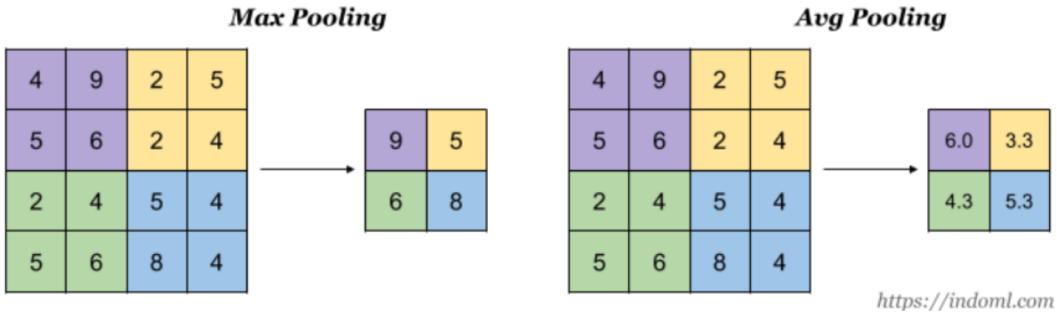


Figure 2.1.7: Max and average pooling operations with a 2×2 pool size [81].

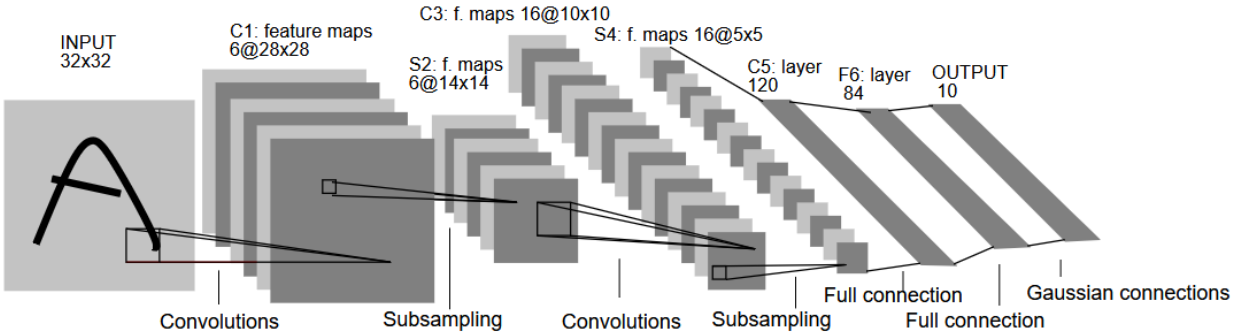


Figure 2.1.8: Architecture of LeNet-5, a CNN for handwritten character recognition [59]. Notice the alternation between convolutional and pooling layers, followed by 2 FC layers.

current timestep, the hidden state h_t is a combination of the input x_t with the hidden state h_{t-1} from the previous time step:

$$h_t = f(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \tag{2.1.16}$$

where W_{hx} is the input-to-hidden weight matrix, W_{hh} is the hidden-to-hidden weight matrix, b_h is the bias and f is an activation function, commonly tanh.

Correspondingly, the output y_t at timestep t is calculated as:

$$y_t = g\left(W_{yh}h_t + b_y\right) \tag{2.1.17}$$

where W_{yh} is the hidden-to-output weight matrix, b_y is the output bias and g is an activation function.

Vanishing and Exploding Gradient Problems A significant challenge in training RNNs is the vanishing and exploding gradient problems, which arise during backpropagation through time (BPTT). The gradients of the loss function with respect to earlier layers can diminish (vanish) or grow exponentially (explode), leading to ineffective learning. This issue is particularly problematic for long-range dependencies, as earlier inputs may lose influence over later predictions.

To mitigate these issues, more sophisticated RNN variants have been proposed, such as Long Short-Term Memory (LSTM) networks [46] and Gated Recurrent Units (GRU) [20]. To ensure that older information is preserved when sequences grow longer, LSTMs incorporate additional cell states, which get updated using gating mechanisms (input, forget, output gates) that regulate the flow of information by selectively retaining or discarding past context (Figure 2.1.10). The ability of these architectures to handle long-term dependencies has made them a preferable choice over vanilla RNN networks.

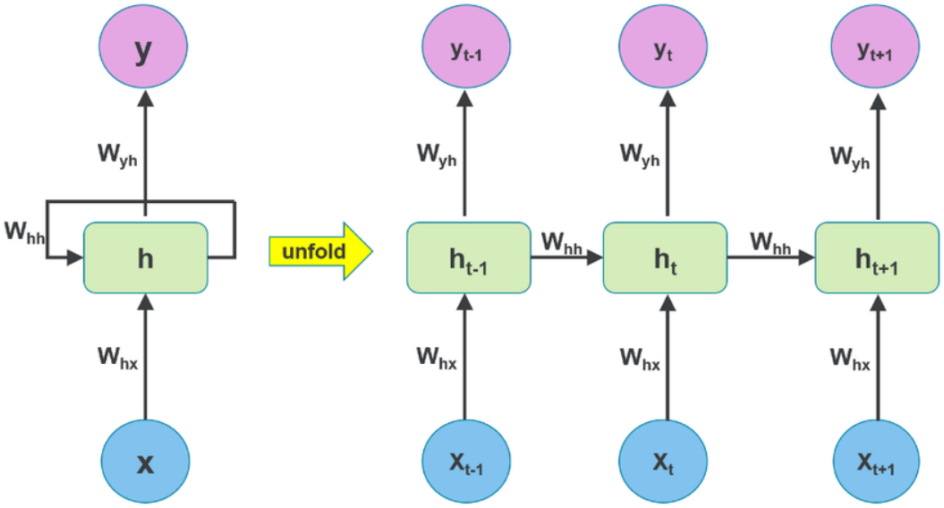


Figure 2.1.9: Structure of a simple RNN (left) and unfolded through-time RNN (right) [111]

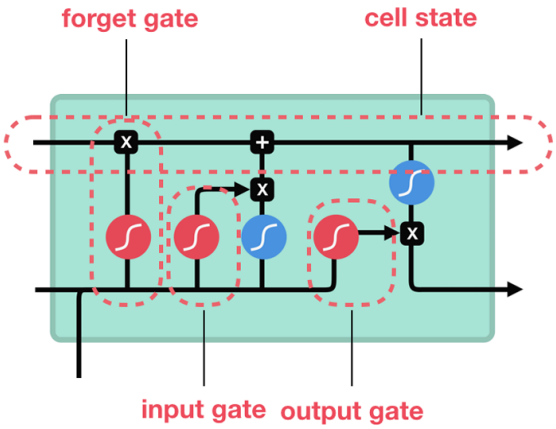


Figure 2.1.10: Structure of an LSTM cell, where Red: Sigmoid, Blue: Tanh [78]

Transformers

The Transformer, introduced in 2017 by Vaswani et al. [109], is a network architecture originally used for machine translation that has revolutionized sequence-based tasks and deep learning in general. Unlike traditional recurrent neural networks (RNNs), which process sequential data in an ordered manner, transformers rely on a highly parallelizable self-attention mechanism. This architectural innovation enables the model to capture long-range dependencies within sequences more efficiently than RNN-based architectures.

Architecture Overview The original transformer model follows an encoder-decoder structure. The encoder processes input sequences and generates context-aware representations, while the decoder utilizes these representations to generate output sequences. Both the encoder and the decoder consist of 6 stacked layers. In the encoder, each layer has two sub-layers: a multi-head self-attention module and a position-wise feed-forward (fully connected) neural network. The decoder layers follow the same structure, but also employ a third sub-layer between the other two that performs multi-head attention over the encoder's output.

Self-Attention Mechanism To implement self-attention, which allows each element in the input sequence to attend to all other elements, the transformer architecture proposed the scaled dot-product attention mechanism. Intuitively, given an element of the input sequence ("query"), this mechanism updates its representation by calculating a weighted sum over all the elements of the sequence ("values"). The weights are obtained by taking the dot product between the query and all the elements ("keys"), which acts as a compatibility

function, followed by a softmax function to turn it into a probability distribution. Mathematically, this is described as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1.18}$$

where:

- Q, K and V are the queries, keys and values matrices, respectively.
- d_k is the dimensionality of the key vectors, and the scaling factor $\sqrt{d_k}$ prevents extremely large values in the softmax function.

Multi-Head Attention To enhance the expressiveness of the attention mechanism, transformers employ multi-head attention, which runs the attention mechanism multiple times in parallel ($h = 8$ in the original architecture), with different learned weight matrices. The outputs from each attention head are concatenated and transformed through a linear projection. Mathematically, these operations are described as:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \\ \text{where} \quad \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.1.19}$$

In the above equations, W_i^Q, W_i^K and W_i^V are the weight matrices for the i^{th} attention head, and W^O is the output weight matrix that combines the outputs of all attention heads.

Position-Wise Feed-Forward Network As mentioned earlier, each encoder and decoder layer also includes a feed-forward network, which applies two linear transformations with a ReLU activation function in between:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.1.20}$$

where W_1, W_2 are weight matrices, and b_1, b_2 are bias terms. This component is applied to each position independently and identically and enables the model to introduce additional non-linearity and enhance feature learning.

Positional Encoding Since transformers do not have a built-in sense of sequence order, they introduce positional encoding to retain positional information. This is done by adding a position-dependent vector to each input embedding.

Layer Normalization and Residual Connections To stabilize training and improve gradient flow, transformers use layer normalization and residual connections around sublayers. Given an input x and a sublayer output $S(x)$, the residual connection is defined as:

$$y = \text{LayerNorm}(x + S(x)) \tag{2.1.21}$$

Masked Self-Attention Finally, the self-attention modules of the decoder layers are allowed to attend only to earlier positions in the output sequence, by masking all future positions.

All the aforementioned components are pieced together in Figure 2.1.11, which shows the overall structure of the transformer:

Later adaptations of the transformer architecture have led to specialized models featuring either encoder-only or decoder-only designs, depending on the task. Encoder-only architectures, such as BERT [28], are well-suited for tasks like text classification that require understanding the meaning of the input, while decoder-only architectures, like GPT [84], excel in generative tasks such as text completion and open-ended dialogue generation. Encoder-decoder models like the original transformer are suitable for tasks that transform the input text, such as machine translation, text summarization, paraphrasing etc..

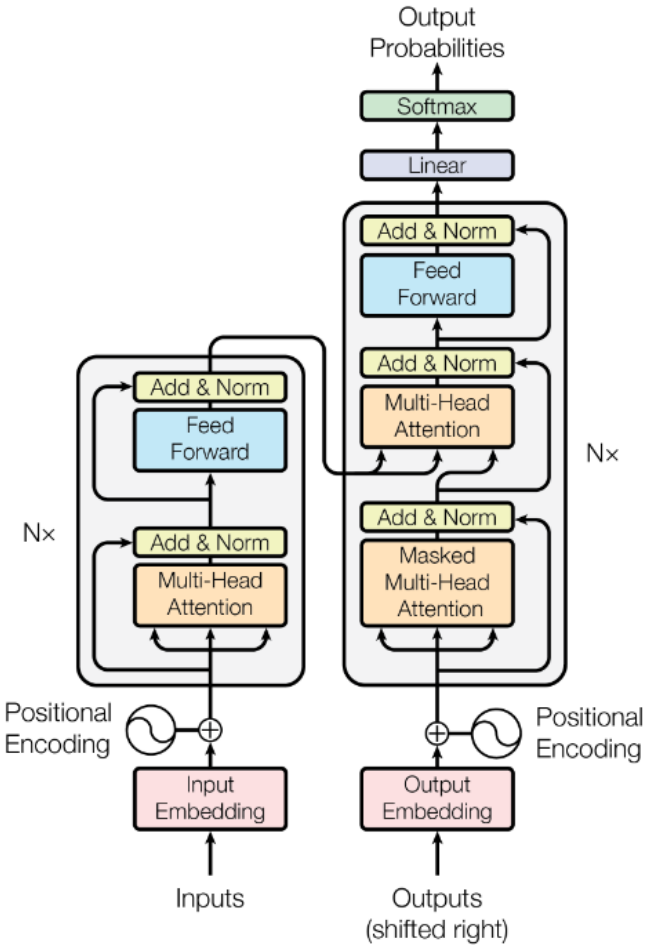


Figure 2.1.11: Architecture of the Transformer [109]

2.1.3 Representation Learning

Metric Learning

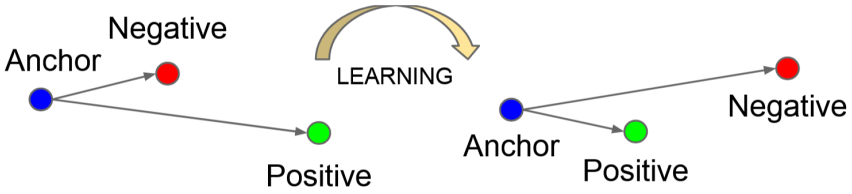


Figure 2.1.12: A triplet loss brings the anchor and positive samples (semantically similar items) closer in the embedding space, while pushing the negative sample (dissimilar item) farther away [97]

Metric learning refers to a machine learning technique that aims to learn a distance function optimized for a specific task. So instead of using a standard distance metric like Euclidean distance, metric learning learns to represent data in an embedding space such that similar items are close together and dissimilar items are far apart, as shown in Figure 2.1.12. This is achieved by using carefully designed loss functions and sampling strategies to create a discriminative embedding space, which can then be used with other techniques like k-Nearest Neighbors (k-NN) for tasks such as classification, clustering, and retrieval.

A common formulation is the triplet loss, which considers an anchor sample A, a positive sample P (similar

to the anchor), and a negative sample N (dissimilar to the anchor). The triplet loss for the triplet (A, P, N) is then defined as:

$$\mathcal{L}_{\text{triplet}} = \max\left(\left\|z^A - z^P\right\|_2^2 - \left\|z^A - z^N\right\|_2^2 + \alpha, 0\right) \tag{2.1.22}$$

where z^A, z^P, z^N are the embeddings of the anchor, positive, and negative, and α is a margin that enforces a minimum separation between positive and negative pairs. This loss is minimized when the positive is closer to the anchor than the negative by at least α. The above loss is calculated for a set of triplets selected from the training data. This set should be carefully constructed in order to include the most instructive triplets that would lead to faster convergence.

Contrastive Learning

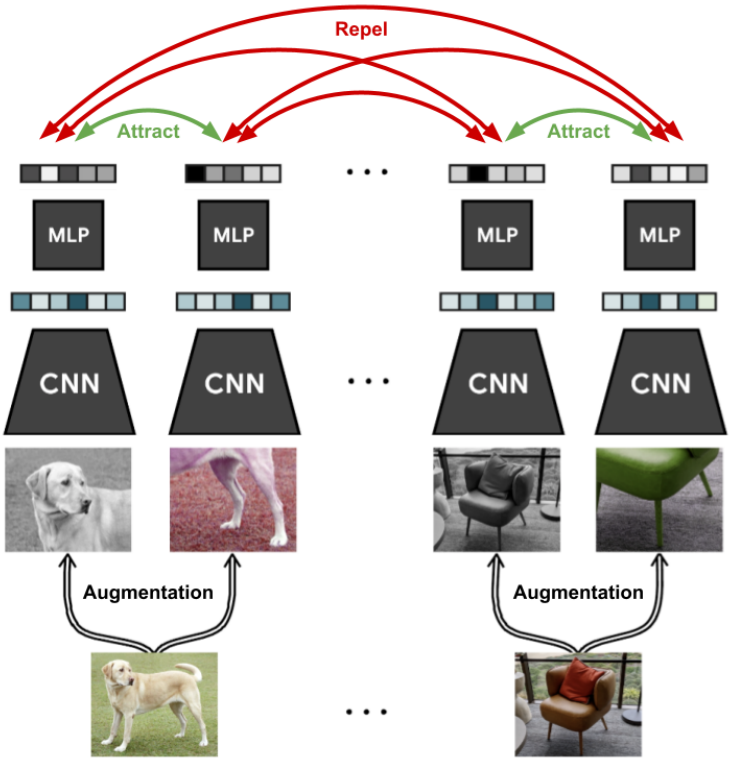


Figure 2.1.13: Overview of contrastive learning with data augmentations[2]. Different augmented views of the same image are treated as positive pairs and are encouraged to produce similar embeddings (attract), while views from different images form negative pairs that are pushed apart (repel). The CNN encoder extracts features, and an MLP projection head maps them into the embedding space where the contrastive objective is applied [19]

Contrastive learning can be seen as a specific type of metric learning that relies on identifying semantically correlated samples ("positive pair") among many negative samples. The creation of the positive pair can include applying different transformations to an input sample [19], [101], as shown in Figure 2.1.13 or aligning different modalities of a data point, e.g. an audio sample and its natural language description [66], while the negatives are commonly derived from the rest of the batch.

A widely used example of a contrastive objective is the InfoNCE loss introduced in [74]. Given a sample x, a positive sample x^+, and a set of $N-1$ negative samples $\{x_i^-\}_{i=1}^{N-1}$, the InfoNCE objective is defined as:

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{\exp\left(\text{sim}(z, z^+)/\tau\right)}{\sum_{i=1}^{N-1} \exp\left(\text{sim}(z, z_i^-)/\tau + \exp\left(\text{sim}(z, z^+)/\tau\right)\right)} \tag{2.1.23}$$

[2]https://simclr.github.io/

where z, z^+, z_i^- are the embeddings of x, x^+, x_i^-, $\text{sim}(\cdot,\cdot)$ is a similarity measure such as cosine similarity, and τ is a temperature parameter controlling the sharpness of the distribution. This formulation encourages x to be close to its positive example while being far from negatives, effectively shaping the embedding space to capture semantic relationships.

While both contrastive and metric learning aim to structure the embedding space based on similarity, contrastive learning typically relies on large batches with many negatives and uses a classification-like normalization (softmax over similarities), whereas metric learning with triplet loss directly enforces relative distance constraints at the level of individual triplets. In practice, both approaches are widely used in domains such as face recognition [97], image retrieval [19], and music representation learning [101].

2.2 Collaborative Filtering

Collaborative Filtering (CF) is a foundational technique in recommender systems that leverages past interactions between users and items to generate personalized recommendations. User feedback for items is usually represented in a user-item matrix and can be either explicit or implicit. Explicit feedback refers to data such as ratings on a 1-5 scale or likes/dislikes, with which users directly express whether they prefer an item or not. This type of feedback requires active participation from users and isn't always available, especially in domains where content is more disposable, as is the case with music [95]. Therefore, recommender systems often resort to implicit feedback, like play counts, purchases or clicks, to infer user preferences. The central task is to predict missing entries of this matrix, that is, the entries corresponding, for each user, to the items they have not interacted with, and recommend the unconsumed items with the highest scores. In this section, we will describe the two primary categories of CF: memory-based and model-based, followed by a discussion on their limitations.

2.2.1 Memory-Based

Memory-based methods operate directly on the user-item matrix and make recommendations using similarity measures. These can be further classified into user-based and item-based, depending on whether we are looking for similar users or similar items.

User-Based

User-based collaborative filtering (UBCF) identifies users with preferences similar to those of a target user and recommends items that these similar users have interacted with. The core of UBCF is the computation of similarity between users, commonly using metrics such as the Pearson correlation coefficient. Given a user-item matrix R, r_{ui} represents the rating of user u for item i. Further given a pair of users u, v, we can define I_{uv} as the set of items rated by both users and $\bar{r}_u$, $\bar{r}_v$ as the average ratings of users u and v, respectively, which we subtract from each rating, r_{ui} (or r_{vi}), to account for differences in the way users u and v tend to rate items. The exact formula for inter-user similarity computation is shown in Eq. 2.2.1:

$$\text{sim}(u,v) = \frac{\sum_{i\in I_{uv}}(r_{ui}-\bar{r}_u)(r_{vi}-\bar{r}_v)}{\sqrt{\sum_{i\in I_{uv}}(r_{ui}-\bar{r}_u)^2}\sqrt{\sum_{i\in I_{uv}}(r_{vi}-\bar{r}_v)^2}} \tag{2.2.1}$$

Once similarities are computed, the predicted rating for an item i for user u is estimated as:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v\in N_u}\text{sim}(u,v)(r_{vi}-\bar{r}_v)}{\sum_{v\in N_u}|\text{sim}(u,v)|} \tag{2.2.2}$$

where N_u is the neighborhood of user u, i.e. the set of users most similar to u.

Item-Based

Item-based collaborative filtering (IBCF) [92] instead focuses on the similarity between items rather than users. The key idea is that if two items are rated similarly by many users, they are likely to be related. The

similarity between two items i and j can be calculated using cosine similarity:

$$\text{sim}(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}} \tag{2.2.3}$$

where U_{ij} is the set of users who have rated both items i and j. Predictions are made by weighting the ratings user u has given to similar items:

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u(i)} \text{sim}(i,j) r_{uj}}{\sum_{j \in N_u(i)} |\text{sim}(i,j)|} \tag{2.2.4}$$

where $N_u(i)$ is the neighborhood of item i, i.e. the set of items most similar to i which user u has rated.

2.2.2 Model-Based

On the other hand, model-based methods use interaction data to learn a model that captures latent characteristics of users and items in a joint low-dimensional space. A widely adopted family of methods is Matrix Factorization (MF) [54], which decomposes the user-item matrix into lower-dimensional matrices containing user and item factors.

One of the most influential variants is Funk-SVD [36], which was introduced by Simon Funk in a famous blog post during the Netflix Prize challenge in 2006 [9]. As shown in Figure 2.2.1, in this model each item i is represented as a vector $q_i \in \mathbb{R}^f$ and each user as a vector $p_u \in \mathbb{R}^f$, where f is the dimensionality of the latent space. Intuitively, q_i gives the distribution of item i on the latent factors, while p_u measures the interest of user u for the corresponding factors. The dot product $q_i^T p_u$ between the item and user latent vectors quantifies the interaction between user u and item i, and can be used as an approximation for the rating that the user would give to the item:

$$\hat{r}_{ui} = q_i^T p_u \tag{2.2.5}$$

This technique resembles Singular Value Decomposition (SVD), a traditional linear algebra method to factorize a matrix, which aims to approximate the user-item rating matrix R as:

$$R \approx U \Sigma V^T \tag{2.2.6}$$

where U represents user factors, V represents item factors, and Σ is a diagonal matrix containing singular values.

However, SVD does not perform well when the matrix R is very sparse, which is usually the case in real-world recommendation applications, where users typically interact with only a small subset of the item catalog. Funk-SVD addresses this shortcoming by relying only on known ratings and ignoring missing values, while avoiding overfitting by applying regularization.

More specifically, to learn the latent vectors for each user and item, the Funk-SVD model minimizes the regularized squared error:

$$\min_{q^*, p^*} \sum_{(u,i) \in \kappa} \left(r_{ui} - q_i^T p_u\right)^2 + \lambda \left(\|q_i\|^2 + \|p_u\|^2\right) \tag{2.2.7}$$

where κ is the set of the (u,i) pairs for which r_{ui} is known and λ is a constant that controls the regularization.

To minimize this equation, we can use either Stochastic Gradient Descent (SGD), which was the method proposed in Funk-SVD or a method called Alternating Least Squares (ALS). The motivation for this method lies in the fact that the above equation is not convex, since both q_i and p_u are unknown. However, if we fix one of them, the objective becomes quadratic and can be solved optimally. Therefore, ALS alternates between fixing one of the matrices and solving for the other, until convergence.

At this point, we should mention that a drawback of classic MF methods is that they assume independence between latent factors, as well as their inability to model non-linear relationships between them (Eq. 2.2.5). This can limit the model's performance, considering the complexity of user-item interactions. A number of works in the literature attempted to utilize more complex mathematical structures, such as neural networks,

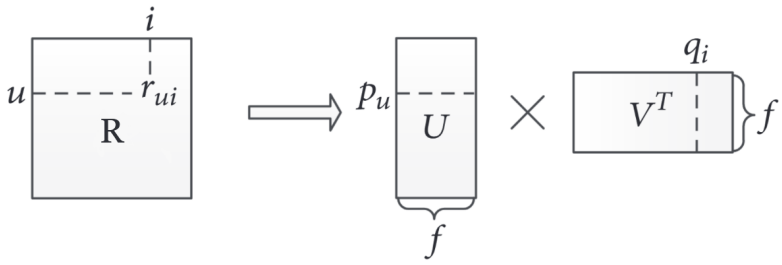


Figure 2.2.1: The FunkSVD model (image adapted from [117])

to overcome these drawbacks. For instance, He et al. [43] proposed Neural Matrix Factorization (NeuMF), a model that replaces the aforementioned dot product with a Multi-Layer Perceptron, aiming at enhancing the model's expressiveness by leveraging the non-linearities of neural networks.

2.3 Audio Signal Representations

Audio signals can be represented in different ways, each emphasizing specific characteristics of the sound. In music information retrieval and recommendation systems, the choice of representation influences which patterns can be learned and exploited. In the following subsections, we outline common representations, starting with the time-domain waveform, moving to frequency-based forms such as the spectrogram, and ending with feature sets such as Mel-Frequency Cepstral Coefficients (MFCCs) [25] and Chroma features.

2.3.1 Time-Domain Representations

The waveform is the most direct representation of an audio signal, depicting how amplitude varies over time. In the continuous domain, an audio signal can be expressed as a function $x(t)$, where t denotes time and $x(t)$ gives the instantaneous amplitude. To store and process this signal digitally, it is sampled at regular intervals determined by the sampling rate f_s (samples per second). According to the Nyquist–Shannon sampling theorem, accurate reconstruction of a band-limited signal is possible if f_s is at least twice the maximum frequency present in the signal. The resulting discrete-time signal is represented as a sequence $x[n] = x(n/f_s)$, where n is the sample index. Modern deep learning models can learn to extract relevant features directly from this representation without the need for handcrafted preprocessing.

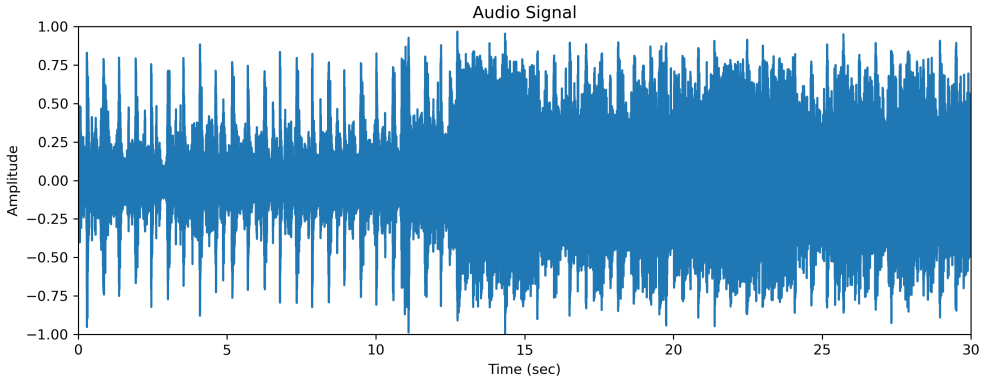


Figure 2.3.1: Waveform of an audio signal

2.3.2 Spectral Representations

While the waveform contains the complete time-domain information, it can be difficult to directly extract frequency-related characteristics such as pitch or timbre from it. The classical tool for frequency analysis is

the Fourier Transform (FT), which decomposes a signal into a collection of sinusoidal components at different frequencies. For a continuous-time signal $x(t)$, the Fourier Transform is defined as:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft}, dt \tag{2.3.1}$$

In the discrete domain, the Discrete Fourier Transform (DFT) is used. While the FT reveals the global frequency content of a signal, it does not provide any information about how the frequencies evolve over time, which is crucial in music where notes, chords, and timbral changes occur dynamically.

To address this limitation, the Short-Time Fourier Transform (STFT) is employed. The STFT assumes that the signal is approximately stationary within each short analysis frame and analyzes the signal over short, overlapping windows. Applying the Fourier Transform to each frame captures both time and frequency information. Mathematically, for a discrete-time signal $x[n]$, the STFT is given by:

$$X(m,k) = \sum_{n=0}^{N-1} x[n+mH], w[n], e^{-j2\pi kn/N} \tag{2.3.2}$$

where $w[n]$ is a window function of length N, m indexes the time frame, H is the hop size between successive frames, and k denotes the frequency bin.

Spectrogram

The spectrogram $S(m,k)$ is computed as the squared magnitude of the STFT:

$$S(m,k) = |X(m,k)|^2 \tag{2.3.3}$$

This produces a 2D representation showing how the signal's spectral content changes over time. The spectrogram illustrates the distribution of energy across time and frequency, yet it maps frequencies on a linear scale—contrasting with the human auditory system's logarithmic perception of pitch.

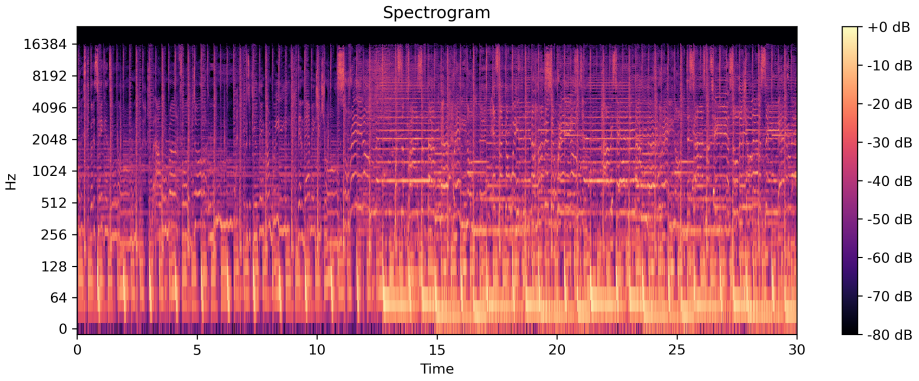


Figure 2.3.2: Spectrogram of the audio signal shown in Figure 2.3.1

Log-Spectrogram

A common modification that aligns the spectrogram with the log-like perception of amplitude is the *log-spectrogram*, where the magnitude values are transformed using a logarithmic scale to compress the dynamic range:

$$S_{\log}(m,k) = \log(1 + |X(m,k)|^2) \tag{2.3.4}$$

This transformation makes subtle spectral details more visible and more aligned with the human ear's logarithmic sensitivity to amplitude. Log-spectrograms are often preferred in machine learning pipelines, as they improve the representation of low-energy yet perceptually important features.

Log-Mel Spectrogram

Another widely used variant is the *log-mel spectrogram*, where the frequency axis is first converted to the mel scale, a perceptual scale that spaces frequencies according to the human ear's frequency resolution, before applying the logarithmic transform. This is achieved by projecting the STFT magnitude onto a mel filterbank $M(f, b)$:

$$S_{\text{mel}}(m, b) = \sum_k M(k, b) \cdot |X(m, k)|^2 \tag{2.3.5}$$

where m indexes the time frames, k indexes the STFT frequency bins on a linear scale, and b indexes the mel filterbank bands, followed by

$$S_{\log\text{-mel}}(m, b) = \log(1 + S_{\text{mel}}(m, b)) \tag{2.3.6}$$

The mel scale emphasizes lower frequencies, where human pitch resolution is finer, and down-samples higher frequencies, which are perceived more coarsely (see Figure 2.3.3). This representation is preferred in many MIR and speech applications because it balances perceptual relevance with compactness, and provides a more uniform distribution of information across the feature space compared to the basic spectrogram.

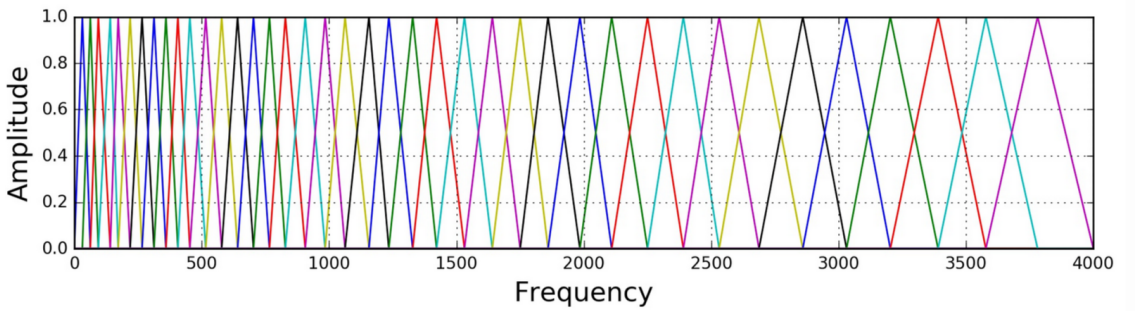


Figure 2.3.3: Mel-scale filter bank [31]

Constant-Q Transform and its Spectrogram

The Constant-Q Transform (CQT) [13] is an alternative time–frequency analysis method that uses logarithmically spaced frequency bins, making it well suited for music signals where pitch perception follows a logarithmic scale. Unlike the Short-Time Fourier Transform (STFT), which applies a fixed-length analysis window to all frequencies, the CQT adapts the window length depending on the frequency: longer windows for low frequencies provide finer frequency resolution, while shorter windows for high frequencies improve temporal resolution. This results in a constant ratio

$$Q = \frac{f_k}{\Delta f_k}$$

between the center frequency f_k of a bin and its bandwidth Δf_k, ensuring that each bin spans the same interval in musical terms (e.g. a semitone).

Similarly to the STFT, by taking the squared magnitude of X_{CQT}, one obtains the *CQT spectrogram*:

$$S_{\text{CQT}}(m, q) = |X_{\text{CQT}}(m, q)|^2$$

This representation is visually similar to an STFT-based spectrogram but differs in that its frequency axis is logarithmically spaced and its time resolution varies across frequencies. These properties make the CQT spectrogram particularly advantageous for tasks such as chord recognition [16], [98], melody extraction [22], [35], and other music information retrieval (MIR) applications where pitch relationships are central.

Mel-Frequency Cepstral Coefficients (MFCCs)

Mel-Frequency Cepstral Coefficients (MFCCs) [25] are a compact and perceptually motivated representation of the short-term power spectrum of a sound. They are designed to approximate the human auditory system's

frequency resolution by combining the mel frequency scale with cepstral analysis, making them particularly effective in speech and music classification tasks. MFCCs provide information about the spectral envelope, which is closely related to timbre, and are among the most widely used features in audio analysis.

- **Step 1: Pre-emphasis**

  The audio signal $x[n]$ is first passed through a pre-emphasis filter to amplify high frequencies, compensating for the natural spectral tilt in human speech and many musical instruments:

  $$y[n] = x[n] - \alpha\, x[n-1]$$

  where α is typically in the range 0.95–0.97.

- **Step 2: Framing and Windowing**

  The signal is divided into overlapping frames (e.g., 20–40 ms), each multiplied by a window function $w[n]$ (often Hamming) to reduce spectral leakage:

  $$x_m[n] = x[n + mH] \cdot w[n]$$

  where m is the frame index and H is the hop size.

- **Step 3: Short-Time Fourier Transform (STFT)**

  The magnitude spectrum of each frame is computed using the STFT:

  $$X(m,k) = \sum_{n=0}^{N-1} x_m[n]\, e^{-j2\pi kn/N}$$

  and the power spectrum is obtained as $|X(m,k)|^2$.

- **Step 4: Mel Filterbank Processing**

  The power spectrum is projected onto a mel-scaled filterbank $M(k,b)$, resulting in mel-band energies:

  $$S_{\text{mel}}(m,b) = \sum_{k} M(k,b) \cdot |X(m,k)|^2$$

  where b indexes the mel bands.

- **Step 5: Logarithmic Compression**

  The mel-band energies are converted to a logarithmic scale to match the human ear's sensitivity to loudness:

  $$S_{\log\text{-mel}}(m,b) = \log\left(S_{\text{mel}}(m,b) + \epsilon\right)$$

  where ϵ is a small constant to avoid $\log(0)$.

- **Step 6: Discrete Cosine Transform (DCT)**

  Finally, the Discrete Cosine Transform is applied along the mel-band axis to decorrelate the coefficients and concentrate the most important information in the first few terms:

  $$\text{MFCC}(m,c) = \sum_{b=0}^{B-1} S_{\log\text{-mel}}(m,b)\, \cos\left[\frac{\pi c(b+0.5)}{B}\right]$$

  where c is the cepstral coefficient index and B is the number of mel bands.

  Typically, the first coefficient ($c = 0$) represents the average log-energy and may be excluded depending on the application. The remaining coefficients capture the coarse shape of the spectral envelope.

Delta and Delta-Delta MFCC Coefficients In addition to the static MFCCs, temporal derivatives are often included to capture dynamics: Delta-MFCCs (Δ) are the first-order derivatives over time, representing the rate of change of the MFCCs. Delta-Delta MFCCs (Δ^2) are the second-order derivatives, representing the acceleration of change.

The complete feature vector for each frame is often the concatenation of static MFCCs, ΔMFCCs, and Δ^2MFCCs, providing both spectral shape and temporal dynamics for improved performance in classification and recognition tasks.

Chroma Features

Chroma features, or pitch class profiles, represent the energy distribution across the twelve semitone pitch classes (C, C#, D, ..., B) of the musical octave, independent of octave. This makes them especially suitable for tasks such as chord recognition [52] and key estimation [75], where harmonic content is more important than absolute pitch.

To compute chroma features, the audio signal is first transformed into a time–frequency representation, commonly using the Constant-Q Transform (CQT) by choosing a suitable logarithmic frequency spacing (i.e., 12 bins per octave) that aligns with the musical scale. The spectral energy is then mapped to the corresponding pitch classes by folding all octaves into a single octave range, summing the energies of frequency bins belonging to the same pitch class. Finally, the chroma vectors are normalized to reduce variations caused by dynamics or instrumentation.

By condensing the spectral representation into 12 dimensions per frame, chroma features provide a compact, musically meaningful descriptor that is robust to octave transpositions while preserving harmonic relationships.

2.4 Evaluation Metrics

In recommender systems, evaluation metrics quantify how well the model's predictions match the ground truth, or, equivalently, how well the produced recommendations match the items that the user actually likes. Some of these metrics, such as accuracy and recall are typical of classification problems and treat recommendation as a binary prediction task (relevant vs. non-relevant items). Another group of metrics, which we'll refer to as ranking metrics, quantify the system's ability to place relevant items higher on the recommendation list, since this is where we would like to find the most accurate recommendations.

2.4.1 Classification Metrics

These metrics assume that the model outputs a predicted relevance score or label for each item, and evaluate correctness without necessarily requiring an ordered list.

Accuracy

Accuracy is the proportion of correctly predicted instances, both relevant and non-relevant, among all predictions:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4.1}$$

where TP and TN are true positives and true negatives, and FP and FN are false positives and false negatives, respectively. Although easy to interpret, accuracy can be misleading in imbalanced datasets—common in recommendation—where relevant items form only a small fraction of all items.

Precision

Precision measures the fraction of recommended items that are actually relevant:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.4.2}$$

A system with high precision rarely suggests irrelevant items, which is important in scenarios where false positives are costly. However, focusing on precision may miss many relevant items, as it does not account for false negatives.

Recall

Recall measures the fraction of all relevant items that the system successfully recommends:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.4.3}$$

High recall indicates that most relevant items are retrieved, which is desirable in coverage-oriented applications. Precision and recall are often in tension, so their relative importance depends on application requirements.

ROC-AUC

The Area Under the Receiver Operating Characteristic curve (ROC-AUC) is the probability that a randomly chosen relevant item will be ranked higher than a randomly chosen non-relevant one. The ROC curve plots the true positive rate (recall) against the false positive rate (FPR) for various thresholds, where

$$\text{FPR} = \frac{FP}{FP + TN} \tag{2.4.4}$$

ROC-AUC is threshold-independent and reflects a model's general ability to discriminate between classes, but may be less informative in imbalanced settings.

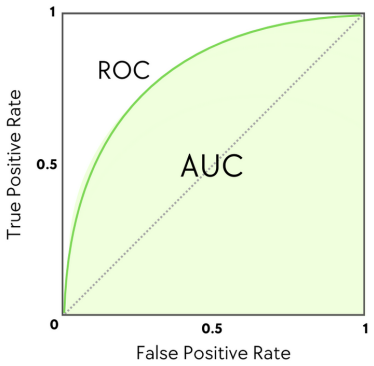


Figure 2.4.1: Example of a Receiver Operating Characteristic (ROC) curve, highlighting the Area Under the Curve (AUC). [99]

PR-AUC

The Area Under the Precision–Recall curve (PR-AUC) summarizes the trade-off between precision and recall across thresholds. It focuses solely on the positive class, making it more informative than ROC-AUC for imbalanced datasets, which are common in recommendation.

2.4.2 Ranking Metrics

These metrics evaluate the quality of a ranked list of recommendations, focusing more on the position of relevant items.

Recall@k

Recall@k is almost identical with the previously defined Recall, with the difference that it measures the proportion of relevant items appearing within the top k positions of the ranked list:

$$\text{Recall@k} = \frac{\#\text{ relevant items in top-}k}{\#\text{ relevant items in ground truth}} \tag{2.4.5}$$

Although it does not consider the order of relevant items within the top k, it reflects how well the system covers relevant content in the most visible part of the list.

HR@k

Hit Rate@k (HR@k) is a simplified variant of Recall@k that only checks whether at least one relevant item appears in the top-k recommendations:

$$\text{HR@k} = \begin{cases} 1, & \text{if at least one relevant item is within the top-}k \\ 0, & \text{otherwise} \end{cases} \tag{2.4.6}$$

In a session-based recommendation setting where each test instance has a single ground-truth next item, HR@k simply measures the proportion of cases where the correct item is retrieved within the top-k predictions. It provides an intuitive measure of success that is independent of ranking order.

MRR@k

Mean Reciprocal Rank at k (MRR@k) also focuses on whether the ground-truth item appears among the top-k positions, but takes into account where it appears. The metric computes the reciprocal of the rank at which the first relevant item occurs:

$$\text{MRR@k} = \frac{1}{|Q|} \sum_{q=1}^{|Q|} \frac{1}{\text{rank}_q} \tag{2.4.7}$$

where rank_q is the position of the first relevant item in the ranked list for query q, and $|Q|$ is the number of test queries. If the relevant item is not found within the top-k, its contribution is zero. Higher values of MRR@k indicate that relevant items tend to appear closer to the top of the recommendation list.

MAP@k

Other metrics, like MAP@k, not only focus on finding the first relevant item, but also evaluate the ranking order of the top k items. To calculate Mean Average Precision at k (MAP@k), we first average the precision values at the ranks where relevant items occur, up to k:

$$\text{AP@k} = \frac{1}{\min(k, R)} \sum_{i=1}^{k} P(i) \cdot \text{rel}(i) \tag{2.4.8}$$

where $P(i)$ is precision at rank i, $\text{rel}(i) \in \{0, 1\}$ indicates a binary relevance score (i.e. relevant/not-relevant), and R is the number of relevant items. By taking the mean AP@k over all queries, we then get MAP@k. This metric rewards systems that place relevant items earlier in the list, unlike Recall@k.

NDCG@k

Similarly, Normalized Discounted Cumulative Gain at k (NDCG@k) discounts (penalizes) relevant items that appear lower in the ranking:

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel}(i)} - 1}{\log_2(i+1)} \tag{2.4.9}$$

where $\text{rel}(i)$ is the relevance score of the item at position i, which can be either binary or numerical (e.g. 1-5). The metric is then calculated as:

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}} \tag{2.4.10}$$

where IDCG@k is the DCG@k of an ideal ranking, i.e., a ranking of items by descending relevance. It is apparent that, similar to MAP@k, NDCG@k prioritizes showing highly relevant items earlier in the ranking.

Chapter 3

Related Work

3.1 Representation Learning from Audio

In this section, we examine models that learn representations of audio signals in a latent space, in a way that facilitates the retrieval of similar tracks and captures structural properties of music. Existing works adopt either a supervised learning approach, exploiting some prior knowledge of music similarity (descriptive tags, co-listen statistics, etc.) or self-supervised techniques that take advantage of unlabeled data collections and train models using labels generated from the input data.

3.1.1 Supervised Methods

Supervised approaches to music representation learning are characterized by the use of externally provided labels or annotations to guide model training. These supervision signals can take the form of user interaction data, semantic tags, genre or mood annotations, or other editorial metadata. By optimizing representations to predict or align with such human-curated targets, models are able to capture musically meaningful properties that correspond to established categories of perception and use. The following works illustrate different ways in which supervised signals have been exploited to learn robust audio embeddings for music.

An early work is that of McFee et al. [67], who seek to advance content-based methods by leveraging interaction data in order to learn an audio similarity metric optimized to produce a ranked list of results similar to a query song (query-by-example). With this approach, they mitigate the long-standing cold-start problem of collaborative filtering systems and at the same time, promote novelty in recommendations. To begin with, song representations are obtained by clustering a large pool of delta-MFCC features to build a codebook V and then using vector quantization to represent each song X of the dataset as a codeword histogram (Figure 3.1.1). Authors then apply the Metric Learning to Rank (MLR) algorithm [68], which aims at optimizing a distance metric W based on a ranking loss, while using user preference data to extract a notion of correct ranking. More specifically, for each song, a set of relevant songs is derived based on the Jaccard similarity of their corresponding artists, as computed from a binarized users-artists matrix. Experiments on the CAL10K dataset [107] indicate that the distance metric learned from MLR surpasses native Euclidean distance, resulting in a ROC-AUC score of 0.808%.

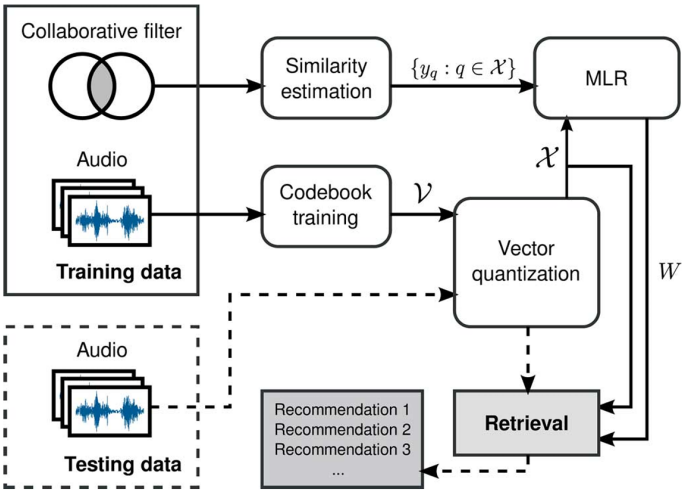


Figure 3.1.1: Schematic diagram of the training and retrieval process in [67]

Lee et al. [60] presented a work that also focuses on the query-by-example task and suggests a way of using deep metric learning to represent music in an embedding space that decouples the different aspects that comprise music similarity. The authors underline that relevance between music tracks relies on several co-existing measures of similarity (they consider genre, mood, instrumentation and tempo) and shouldn't be modeled in one, global-similarity dimension, as is the case with previous works. To build this disentangled multidimensional space, they extend the idea of Conditional Similarity Networks [110], which originally tackled the same task for images. More precisely, log mel spectrograms are used as input for a CNN model that maps audio into an embedding space, by using a triplet loss. To extract the positive and negative

samples of each triplet they exploit user annotations and algorithmic estimates. Decoupling of the similarity dimensions is achieved by the use of masks that help to create disjoint subspaces, each one encoding a specific similarity notion, thus allowing similarity along some dimensions and dissimilarity along others. Moreover, they introduce a track regularization technique to capture the idea of track similarity across all dimensions. Experimental evaluation of the similarity metric induced by this embedding space shows that the decoupled embeddings improve similarity performance and outperform a baseline following the codeword histogram approach presented in [67], but using Euclidean distance. In addition, evaluation on human-annotated triplets indicates the effectiveness of track regularization in modeling music similarity as perceived by humans. Overall, the authors argue that the idea of disentangling the embedding space could prove useful in modern applications, where users interested in music retrieval could specify which dimensions they consider important, potentially leading to better recommendations.

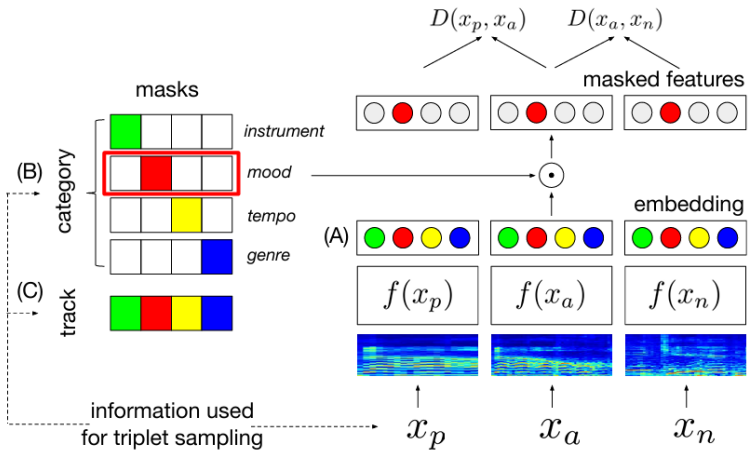


Figure 3.1.2: The proposed approach in [60]: (A) A standard triplet-based deep metric model maps audio input to the embedding space via a learned function f, (B) conditional similarity masking to compute distance metric D separately for each similarity dimension, and (C) track regularization

Another way of obtaining useful representations of audio content was presented by Huang et al. [49]. In their work, they face the challenge of learning audio embeddings from a large collection of more than 10 million music videos, in lack of strong supervisory signals. In fact, supervision is provided in the form of a fine-grained vocabulary of 100k text labels, extracted from unstructured metadata, as well as co-listen data. Their model consists of a CNN given log mel spectrograms of the audio as input, aiming to create a joint embedding space of the audio and the text labels. To incorporate both supervision types, they devise a hybrid optimization plan that follows a curriculum training [8] approach: at first, the network is optimized on a triplet loss to learn a similarity metric from the co-listen graph, followed by a cross-entropy loss on the labels to encapsulate more subtle semantic properties of the audio content. Evaluation on the tasks of predicting text labels and similarity of music video pairs shows that the hybrid objective benefits both tasks, although the contribution of the classification loss seems to dominate. Furthermore, their best-performing ResNet-18 model coupled with simple classifier architectures achieves SOTA performance (92% ROC-AUC) on the MagnaTagATune (MTT) [58] top-50 tag prediction task, even in a fixed-embedding setting, while it surpasses the baseline on the genre prediction task on the AudioSet dataset [39], achieving 91.6% ROC-AUC.

In a similar manner, Pons and Serra proposed MusiCNN [80], a collection of CNN models trained in a supervised way to predict audio tags. Overall, MusiCNN takes log mel spectrograms of 3-second frames as input and outputs the likelihood for each of the 50 predefined tags ("taggram"). More specifically, the first convolutional layer ("front-end") applies a group of vertical and horizontal filters with various shapes, to capture timbral features and temporal dependencies, respectively, as shown in Figure 3.1.3. The resulting features are then fed to three dense convolutional layers ("mid-end") that capture higher-level features (Figure 3.1.4). These layers employ residual connections, which are useful to stabilize training. Finally, the mid-end features are max-pooled and mean-pooled across time to produce two feature vectors. These two vectors are concatenated and projected into the final embedding dimension (200 for the base model and 500 for the larger one), before passing through a last fully-connected layer that produces the logits for the 50 tag classes

(Figure 3.1.5). The published models are trained on either the MTT dataset (19k songs) or the Million Song Dataset (MSD) [10] (on a training set of 200k songs) and have demonstrated comparable performance with the work of Huang et al. [49] (90.69% ROC-AUC / 38.44% PR-AUC on MTT), while on the MSD dataset they report 88.01% ROC-AUC / 28.90% PR-AUC for their base model and 88.41% ROC-AUC / 30.02% PR-AUC for their larger model. The authors also show the effectiveness of MusiCNN in transfer learning scenarios by extracting MusiCNN features and training a simple classifier to perform genre classification on the fault-filtered GTZAN dataset [108]. Their base MSD model achieved 77.24% accuracy, which is on par with other models trained on a much larger data collection of 2 million audio files. Interestingly, in a previous work, Pons et al. [79] found that exploiting domain knowledge in the filter design, as is the case with the aforementioned front-end layer (Figure 3.1.3), is preferable when training on limited data. On the other hand, when large-scale training data is available, generic architectures that rely directly on waveforms can outperform the domain-specific design.

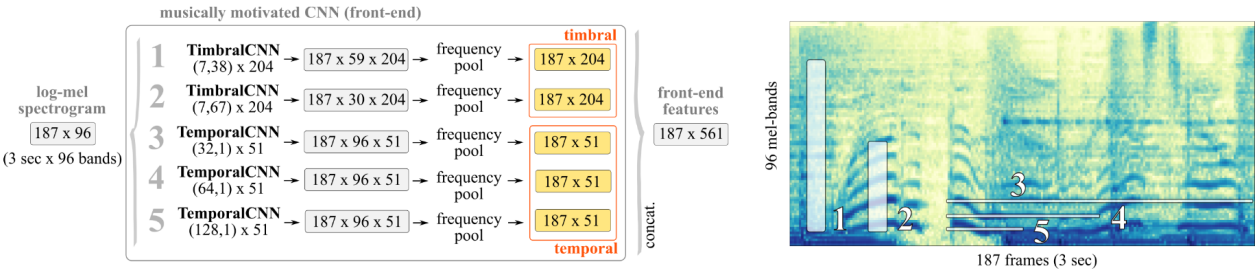


Figure 3.1.3: The timbral and temporal filters in the first layer of MusiCNN [80].

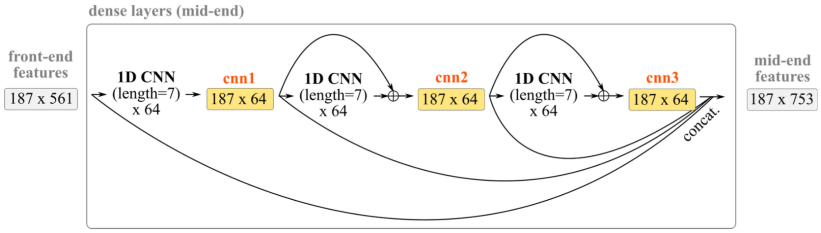


Figure 3.1.4: Dense convolutional layers capture higher-level features.

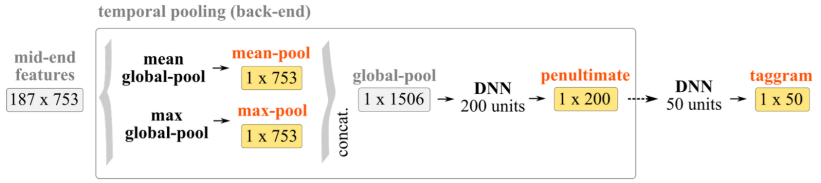


Figure 3.1.5: The backend produces the output, given the previously learned features.

3.1.2 Self-Supervised Methods

While supervised approaches have shown strong performance, they rely heavily on curated labels such as tags or annotations, which are costly to obtain at scale and often noisy or biased. This limitation has motivated a shift toward self-supervised learning (SSL), where models are trained on large collections of unlabeled audio by solving pretext tasks with labels derived from the data itself. Broadly, two paradigms can be distinguished: contrastive learning, which encourages invariance by treating augmented versions of the same signal as positives and other samples as negatives [19], [37], [101], and predictive or masked modeling, where models learn to infer missing or hidden parts of the input [3], [4], [28], [47], [62], [63], given the available context. By exploiting vast amounts of raw audio without the need for manual annotation, these methods enable the training of general-purpose music representations that transfer effectively across downstream MIR tasks such as tagging, classification, and retrieval.

Contrastive Learning

The effectiveness of contrastive learning has already been well demonstrated in extracting representations from various modalities. A prominent example from the image domain is SimCLR [19], which applies data augmentations such as cropping or rotating an image to generate two distinct views of the same instance (see Figure 2.1.13). These views are then treated as a positive pair within a contrastive objective, encouraging the model to learn invariant and discriminative visual representations.

Building on the success of SimCLR, Spijkervet and Burgoyne [101] adapt this paradigm to the music domain with their model CLMR (Contrastive Learning of Musical Representations). CLMR pre-trains a CNN directly on raw audio waveforms using the NT-Xent loss, where positive pairs are generated by applying diverse audio transformations like pitch shifting and frequency filtering to each input example, while all other samples in the minibatch serve as negatives (Figure 3.1.6). Frozen representations from this pretrained model are then used as input to a linear model for the downstream task of music tagging. Pretraining, as well as linear evaluation, are done on the MTT and the MSD datasets. The proposed self-supervised representations achieve a ROC-AUC of 88.7% and 35.6% PR-AUC on MTT, outperforming fully supervised baselines, but fall short on the MSD, with 85.7% ROC-AUC and 25% PR-AUC. Additional experiments show that training the linear classifier on 1% of the labeled MTT data gives comparable performance to a supervised model using all training data, thus proving the effectiveness of self-supervised pretraining. Finally, the authors try pretraining on different music datasets and evaluating on MTT and obtain competitive results even for smaller datasets, indicating the generalizability of the learned representations.

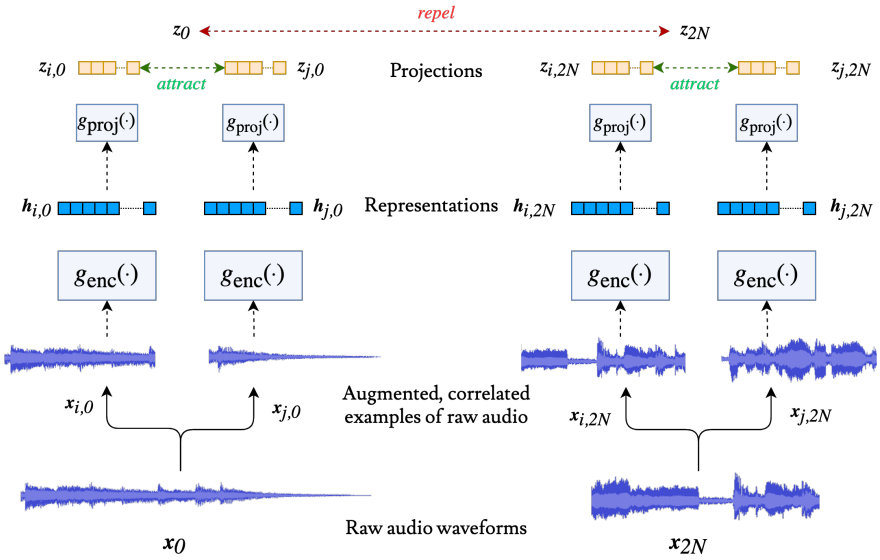


Figure 3.1.6: Overview of the CLMR framework [101]

Instead of relying solely on audio transformations to obtain positive pairs, another work of Manco et al. [66] explores multimodal contrastive learning to map music audio and corresponding text descriptions into a common latent space. The proposed framework uses separate encoders for each modality – a ResNet-50 [42] on mel-spectrograms for audio and a sentence-BERT [86] as text encoder – and tries to align audio and text representations by applying an InfoNCE loss, treating (audio, text) tuples as positives. The authors also incorporate a unimodal contrastive objective based on audio transformations, similarly to CLMR. The model is then evaluated on retrieval from one modality to the other (e.g., retrieving relevant audio based on a text query and vice versa) and is found to greatly surpass the baseline. Moreover, the system is also tested on the tasks of genre classification (GTZAN dataset) and auto-tagging (MTT dataset), by treating the target labels as input text, and demonstrates zero-shot transfer capabilities by hitting 62% accuracy on GTZAN and a 78% ROC-AUC and 29.3% PR-AUC on MTT.

The feasibility of learning meaningful representations from raw audio, without relying on large labeled corpora, had already been proven for speech recognition, by a highly influential model called wav2vec2.0 [4].

First, a CNN extracts latent speech representations from raw audio data. Parts of these are masked and then fed to a Transformer that contextualizes the learned representations. At the same time, the CNN output is used as input to a quantization module that learns discrete speech units, which are used as ground truth for a contrastive training objective where the model tries to distinguish the true quantized representation from a set of negative samples taken from the same audio excerpt (Figure 3.1.7). After pretraining, the model can be fine-tuned on labeled data to perform speech recognition by adding a linear layer after the Transformer and optimizing via a Connectionist Temporal Classification (CTC) loss [41], beating previous SOTA while using far less labeled data.

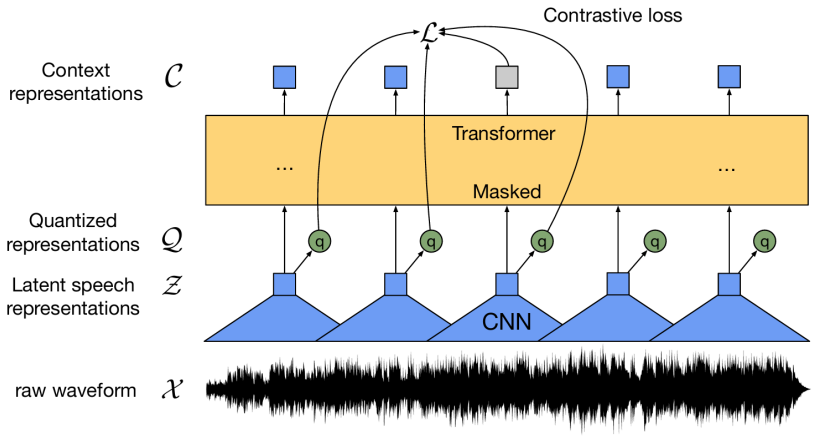


Figure 3.1.7: Illustration of wav2vec2.0 [4]

Other Self-Supervised Approaches

Building on the success of contrastive frameworks like wav2vec 2.0, HuBERT (Hidden-Unit BERT) [47] follows a similar architecture but adopts a different objective that relies on predicting discrete hidden units rather than distinguishing positives and negatives. Instead of relying on contrastive loss, HuBERT generates pseudo-labels via offline k-means clustering of acoustic features and trains a model that predicts which cluster each of the masked frames belongs to (Figure 3.1.8). This setup eliminates the need for large batches of negatives and mitigates instability associated with contrastive learning, leading to more efficient training. Crucially, HuBERT also adopts an iterative refinement strategy: after an initial pretraining phase with MFCC-based clusters, subsequent clusterings are performed on the model's own learned representations, gradually improving target quality. Empirically, HuBERT matches or surpasses wav2vec 2.0 across multiple fine-tuning settings with different volumes of labeled data.

By framing self-supervised learning as masked prediction of acoustic units, HuBERT paved the way for generalized predictive frameworks. One such work is data2vec (Baevski et al. [3]), which further generalizes self-supervised learning by eliminating discrete quantization entirely and predicting contextualized latent representations directly. Instead of classifying masked inputs into pre-defined clusters, data2vec uses a teacher-student architecture: the teacher produces contextualized representations of the full input, which are partially masked and given to the student model, which tries to predict the teacher's representations at the masked positions. A major contribution of this framework is that it can be applied uniformly across speech, vision, and language using this same training objective. By predicting continuous, context-rich embeddings rather than modality-specific tokens, data2vec simplifies SSL design while demonstrating performance gains against single-modality baselines.

This general self-supervised framework was later applied by Li et al. in music2vec [62], to pretrain representations on raw music data, which were then tested on several MIR tasks like music tagging and genre classification and achieved comparable performance to SOTA models [14], while having 50 times fewer parameters.

JukeMIR (Castellon et al. [14]) follows a different approach that leverages intermediate representations of a music generation model, OpenAI's Jukebox [29], and probes them by training shallow classifiers on down-

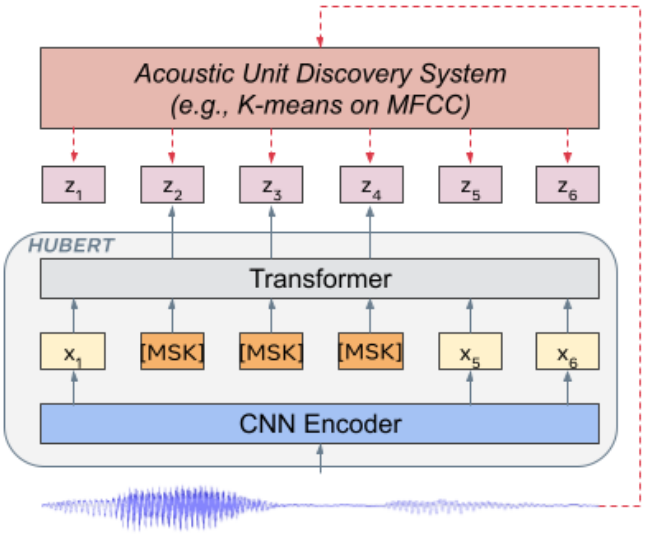


Figure 3.1.8: HuBERT predicts cluster assignments of the masked frames, which are derived from iterative k-means clustering [47]

stream MIR tasks. Jukebox's architecture first uses a hierarchical VQ-VAE [73], where raw audio is successively compressed into discrete codes at multiple temporal scales. A 72-layer, 5B-parameter Transformer is then trained autoregressively to model sequences of these codes. In JukeMIR, Castellon et al. freeze the pretrained Jukebox model and extract representations from the middle layer of the transformer, which are then mean pooled across time to obtain a final set of 4800 features for each input clip of 24 seconds. Remarkably, without any task-specific fine-tuning, the averaged JukeMIR results across four MIR tasks show a 30% better performance relative to the next best pretrained model and comparable performance to task-specific, SOTA methods (e.g. 91.5% ROC-AUC on the MTT tagging dataset compared to 92% from Huang et al. [49]), demonstrating that codified generative pretraining captures rich musical structure.

Despite JukeMIR's promising results, one could argue that its size (5B parameters) renders it prohibitively expensive for use in MIR applications. In search of another computationally affordable alternative, the authors of music2vec built MERT [63], a general-purpose large pretrained model for music understanding, with significantly smaller size compared to JukeMIR: 95M parameters for the base model and 330M for the large one. MERT adopts the architecture of HuBERT, with the main difference that instead of using clustered MFCCs as ground truth, they employ features suited to encapsulate the complex characteristics of music (Figure 3.1.9). In particular, similarly to HuBERT the model first encodes 5-second audio clips using 1D convolutions and then forwards a masked version of the sequence to a Transformer that captures context dependencies. Importantly, to encourage an accurate discretized acoustic-level representation of the music signal, the targets of the loss function are given by a Residual Vector Quantisation-Variational AutoEncoder (RVQ-VAE) [26], an autoencoder that compresses audio into discrete code sequences, which acts as an acoustic teacher. Alternatively, they also explore acoustic modeling by clustering log-Mel spectrograms and Chroma features with k-means, which however faces scaling issues. Moreover, an additional reconstruction loss is employed to serve as a musical teacher that compares the Transformer's output with the Constant-Q transform (CQT) spectrogram, in order to model pitch and harmonic information. It's worth mentioning that the authors observed severe training instabilities when trying to scale up to the large variant, which they circumvented by applying attention relaxation techniques [17] and pre-layer normalisation, admitting that further improvements can be made. MERT is thoroughly tested on 14 downstream MIR tasks by using the pretrained models as fixed feature extractors to train simple shallow architectures (linear or one-layer MLPs), similarly to JukeMIR. According to the results, MERT-330M demonstrates overall SOTA performance and outperforms the much larger JukeMIR on many tasks, while the lightweight variant also exhibits competitive performance, suggesting the effectiveness of Masked Language Modeling (MLM) pretraining for music understanding.

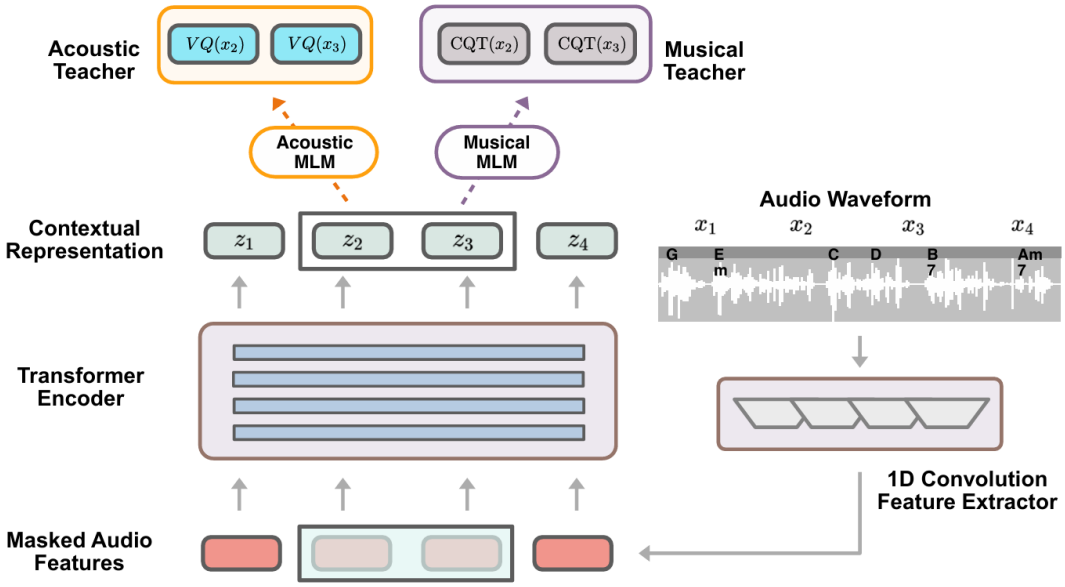


Figure 3.1.9: The pretraining procedure followed in MERT [63]

3.2 Sequential Recommendation

User interaction sequences, such as playlists or listening histories, can be naturally interpreted as ordered sequences of items, much like words in a sentence. This analogy has motivated the adaptation of sequence modeling techniques from natural language processing to the recommendation domain. Early works demonstrated the effectiveness of recurrent neural networks for capturing short- and medium-term dependencies in user sessions [44], [45], while later models based on self-attention and the Transformer architecture extended these ideas to handle long-range dependencies [53], [100], [103]. More recently, the rise of large pretrained language models has prompted research into their potential role in sequential recommendation [82].

GRU4Rec by Hidasi et al. [45] marked the first major application of deep sequence modeling to session-based recommendation. The architecture relies on Gated Recurrent Units (GRUs) [20] to process the sequence of user-item interactions in a session, predicting the next likely item. Unlike later models, GRU4Rec does not use an embedding layer since this didn't boost performance; instead, it directly inputs one-hot encodings of item IDs, processed by a single GRU layer. To address the scalability challenge of having to score all items at each training step, the authors introduced an efficient negative sampling strategy: apart from the expected output, they also score only the items within the same mini-batch, which are treated as negative examples. This doesn't require explicit negative sampling for every training example, thus significantly reducing computational load. Furthermore, they experimented with two pairwise ranking losses, namely BPR [88] and a custom TOP1 loss, encouraging higher ranking of the positive item compared to the negatives, rather than exact prediction of the positive. These performed better than typical classification losses such as cross-entropy, which proved numerically unstable. To handle variable-length sessions efficiently, they propose a session-parallel mini-batching mechanism, where multiple sessions are processed in parallel across time steps. Overall, GRU4Rec demonstrated that recurrent models, when carefully adapted for ranking and efficiency, could dramatically outperform simple item-similarity based methods in session-based recommendation settings.

SASRec by Kang and McAuley [53] extended the idea of sequence modeling by replacing recurrence with self-attention mechanisms (Figure 3.2.1). Architecturally, SASRec consists of a stack of Transformer decoder blocks, where each block contains a multi-head self-attention layer followed by a position-wise feed-forward network. To model the sequential nature of interactions, causal masking is applied to the attention layers, ensuring that each position can only attend to previous items and not future ones. Each item in the sequence is represented by a learnable item embedding summed with a learnable positional embedding, giving the model explicit access to order information. A key advantage of SASRec over RNNs like GRU4Rec is parallelization:

since self-attention layers operate on the entire sequence simultaneously, training becomes highly efficient. Moreover, SASRec can adaptively focus on relevant past items for each prediction, unlike RNNs, which tend to compress the entire past into a single hidden state. Extensive experiments showed that SASRec not only outperforms RNN-based models but also handles both short-term patterns (like Markov Chains) and long-term dependencies effectively, especially as sequence length grows.

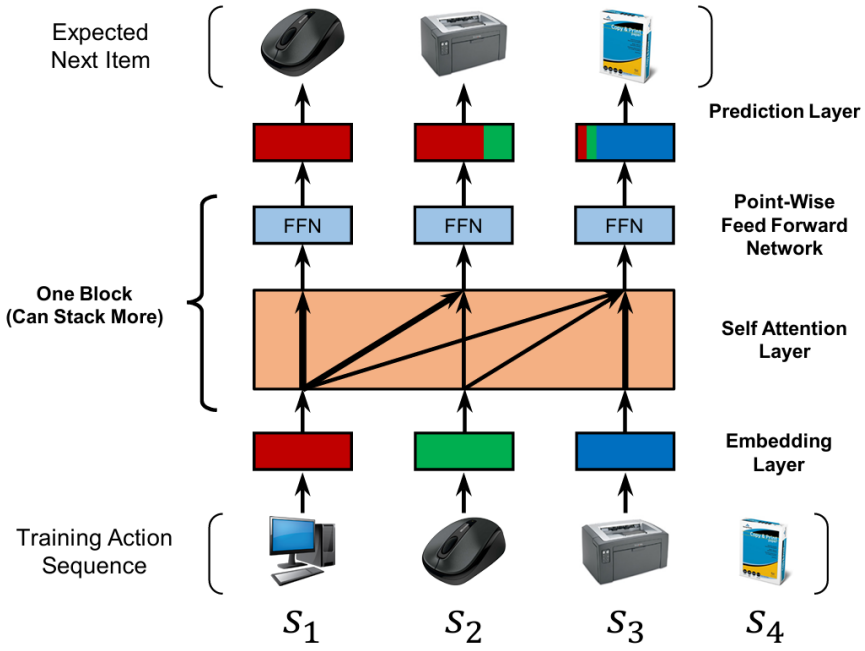


Figure 3.2.1: Architecture of SASRec [53]. At each time step, the model can attend only to previous items to predict the next item.

BERT4Rec by Sun et al. [103] pushed sequence modeling further by adopting bidirectional self-attention for sequential recommendation. Built entirely on Transformer encoder blocks, the model learns item representations by attending simultaneously to both past and future context in a sequence. To prevent information leakage during training, BERT4Rec introduces a Cloze-style objective [106] (also called MLM in BERT [28]): random items in the input sequence are masked, and the model must predict the masked items based on surrounding context, unlike unidirectional models that are trained on predicting the next item. During evaluation, a special [MASK] token is appended to the end of the sequence, and the model predicts the most probable next item. As in SASRec, items are represented using learnable item embeddings combined with learnable positional embeddings. Unlike causal self-attention models, however, BERT4Rec's bidirectional attention yields richer contextualized representations for each item. Empirical results on several datasets showed substantial gains over unidirectional models, highlighting the importance of full-context modeling in session-based recommendation.

Transformers4Rec by Moreira et al. [100] also follows the trend to apply NLP innovations to recommendation. Transformers4Rec is an open-source library built on HuggingFace's Transformers [113] that attempts to facilitate the use of the latest Transformer architectures for sequential and session-based recommendation. Architecturally, Transformers4Rec supports a variety of Transformer models (e.g., GPT-2 [85], TransformerXL [24], XLNet [116]), while introducing supplementary functionalities necessary for recommendation, such as input features normalization and aggregation, incremental training and evaluation, execution of recommendation tasks and evaluation on popular ranking metrics. More precisely, the library introduces mechanisms for handling side information: categorical features regarding the item (music genre, artist, etc.) or the user (region, device type, etc.), which are represented in their own embedding table or continuous features like price. All these types of side information can be seamlessly incorporated into the model input, e.g. by concatenating with the item embedding or by element-wise multiplication. Additionally, Transformers4Rec allows training with various objectives:

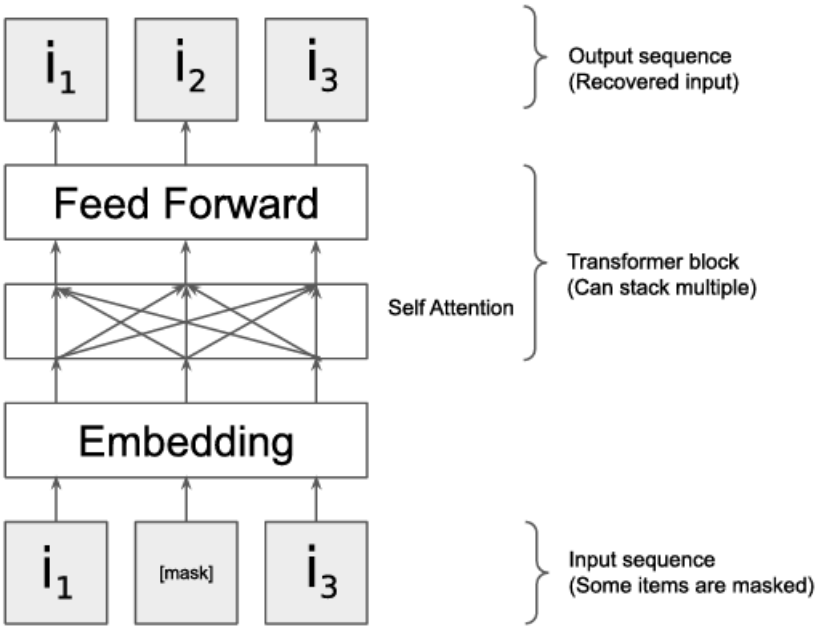


Figure 3.2.2: Architecture of BERT4Rec, which applies bidirectional attention and tries to recover the masked items [77]

- Causal Language Modeling (CLM): predict the next item based only on past items, as seen in SASRec.
- Masked Language Modeling (MLM): randomly mask items and recover them, as in BERT4Rec.
- Permutation Language Modeling (PLM): predict items based on random orderings (used in XLNet).
- Replacement Token Detection (RTD): distinguish real from replaced items (like ELECTRA [21]).

During training, the input sequence is masked according to the chosen objective and is fed to a configurable stack of Transformer blocks, which contextualize the item representations, followed by a prediction head to perform the task at hand. This can be either the prediction of the next item in the sequence, or a classification/regression task related to the whole sequence, in which case the item representations are merged into a sequence embedding. Empirical studies conducted on multiple e-commerce and news datasets demonstrated that Transformer-based architectures consistently outperformed classic RNN and non-neural baselines. Moreover, Transformers4Rec was successfully employed in industry applications, winning two session-based recommendation competitions [71], [96].

What's common between the aforementioned works is the use of transformer architectures for sequence modeling. However, with the advent of large Pretrained Language Models (PLMs), many recent works [5], [61], [87], [119] attempt to leverage these powerful tools to model user behavior. Qu et al. [82] challenge this approach by questioning whether the full potential of PLMs is exploited in this context. Through detailed analysis of Recformer [61], a PLM-based SR model built on Longformer [6], they reveal two key issues: first, PLMs tend to underutilize their deep architecture in this setting, leading to parameter redundancy; second, their sequence modeling behavior often mimics conventional ID-based models like SASRec [53], rather than leveraging their full linguistic reasoning power. Motivated by these findings, the authors explore a simpler and more efficient alternative: using behavior-tuned PLMs—i.e., PLMs fine-tuned on user interaction sequences by training with recommendation objectives like masked language modeling and item-item contrastive loss—only to initialize item embeddings, while replacing the PLM-based sequence modeling component with standard ID-based models such as SASRec [53] and BERT4Rec [103]. Their experiments on five Amazon datasets show that this hybrid design not only reduces training and inference cost but also consistently outperforms both classical ID-based models and full PLM-based ones, especially when embeddings are further fine-tuned. These findings suggest that the primary contribution of PLMs to SR may lie in content-aware item encoding, rather than in sequence modeling itself, and point toward a more efficient paradigm that decouples these two components.

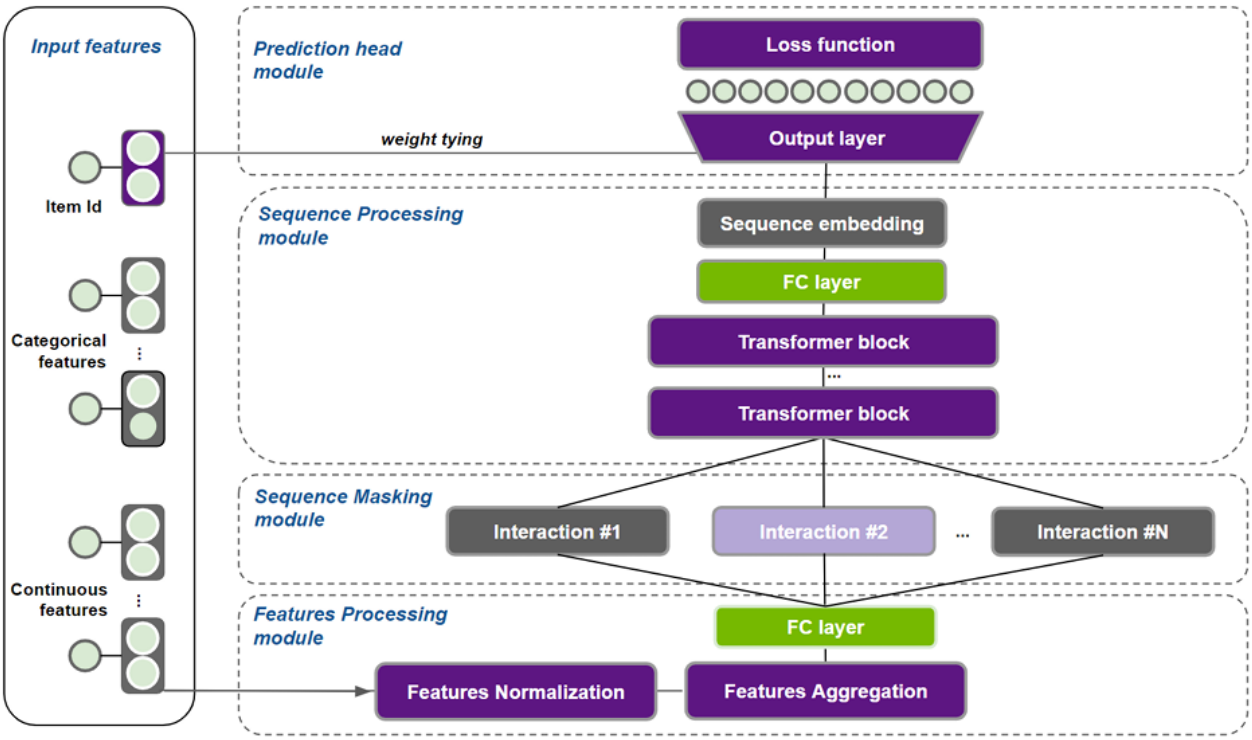


Figure 3.2.3: Schematic overview of Transformers4Rec [100]

3.3 Sequential Music Recommendation

3.3.1 Datasets

Million Playlist Dataset (MPD)

The Million Playlist Dataset (MPD) was released by Spotify as part of the RecSys Challenge 2018 [15], and has since become one of the most widely used benchmarks for Automatic Playlist Continuation (APC). It consists of 1,000,000 user-created playlists, containing over 66 million playlist-track pairs, with an average playlist length of 66.35 tracks, and includes 2.26 million unique tracks with metadata such as track titles, album and artist names, playlist titles, and Spotify URIs. The dataset does not include audio information directly, but for many tracks one may find 30-second previews using the Spotify API. In the context of the RecSys Challenge, MPD was used for the task of predicting missing tracks in incomplete playlists, across ten different scenarios that varied in available input (e.g., title only, no title and first 5 tracks, title and 100 random tracks). Participants were asked to recommend up to 500 tracks per playlist, and solutions were evaluated using NDCG, R-precision, and recommended songs clicks, with overall rankings determined by Borda rank aggregation across these metrics.

Melon Playlist Dataset

The Melon Playlist Dataset [33] was introduced in 2020 to address the need for a publicly available, real-world music playlist dataset that also contains audio information. Collected from the Melon music streaming service, one of the largest platforms in South Korea, the dataset includes 148,826 user-generated playlists and 649,091 unique tracks, with an average of 41.46 tracks per playlist and over 5.9 million playlist-track relations in total. The dataset primarily features Korean music and includes rich metadata: titles and tags for playlists, while songs are accompanied by their title, album and artist information, as well as various genre tags. Notably, the dataset includes mel-spectrograms (20–50 seconds) for every track, enabling research on audio-based recommendation and cold-start scenarios. Overall, thanks to its rich content and collaborative features, this dataset is well-suited for music tagging and APC, as well as for music representation learning approaches.

LFM-1b and LFM-2b Datasets

Unlike the previous datasets, the LFM-1b [93] and LFM-2b [94] datasets are large-scale collections containing music listening histories instead of playlists, sourced from Last.fm[3] users and designed to support research in music information retrieval and recommendation. LFM-1b, introduced in 2016, contains over 1 billion listening events from more than 120,000 users, with each event defined by a user, track, artist, album and timestamp tuple. In addition to basic user demographics (age, gender, country), it includes user descriptors such as mainstreaminess, novelty scores and temporal listening patterns, enabling detailed modeling of music consumption behavior.

Building on this, LFM-2b is a 2020 extension that expands the dataset to over 2 billion listening events, collected from the same Last.fm user base, but over a much longer time span of 15 years. LFM-2b enriches the available features with fine-grained genre labels, user-generated tags, and BERT-based lyric embeddings for over 1 million tracks, while it also maps many tracks to Spotify URIs, facilitating the retrieval of audio features. Thanks to its combined support for collaborative, content-based, and hybrid recommendation strategies, LFM-2b can serve as a powerful benchmark for both traditional and content-enriched music recommendation tasks.

Music4All Dataset

The Music4All dataset [91] introduced in 2020, also contains listening histories from Last.fm users, but, importantly, it comes with raw audio segments for each song. It was created in two phases, as shown in Figure 3.3.1: a user phase that collected the listening histories and a song phase to gather song data, while discarding songs with incomplete information. More specifically, the dataset includes timestamped listening histories from 15,602 Last.fm users, collected between January 1st and March 20th, 2019, with each user having listened to an average of 361 listening events, covering around 200 songs. By the end of the song phase, the dataset contained 109,269 music tracks, each enriched with a multitude of features including 30-second audio clips extracted from the middle of the track, audio features from the Spotify API (such as danceability, valence, and tempo), and detailed metadata including song title, artist, album, genre labels, user-generated tags from Last.fm, and lyrics. To demonstrate its range of applications, the creators used it in music recommendation, genre classification and mood classification tasks.

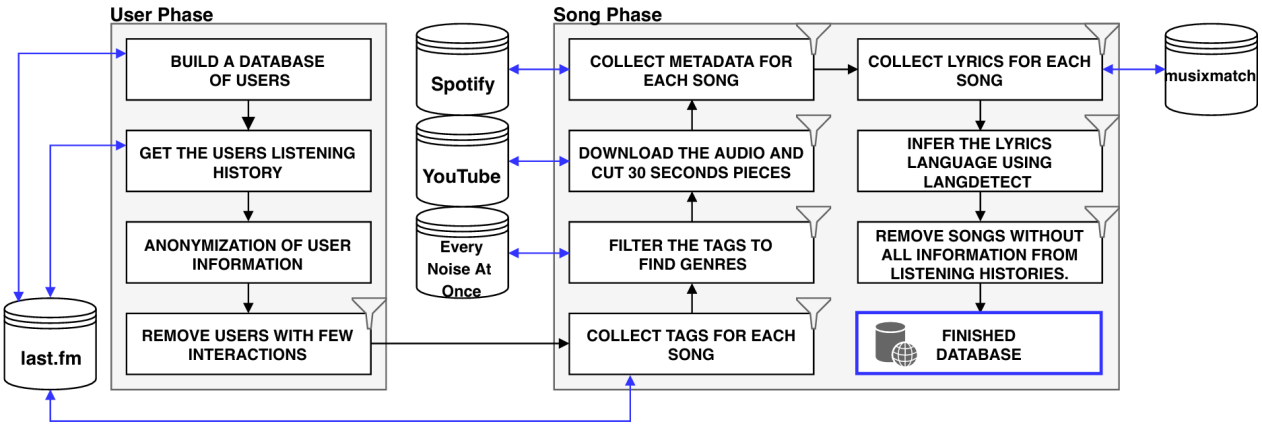


Figure 3.3.1: Development of the Music4All dataset [91]

Music4All-Onion Dataset

The Music4All-Onion dataset [72] was introduced two years later in 2022 and extends the original Music4All dataset with even richer multimodal and behavioral information, aimed at advancing content-centric music recommendation. In addition to the features in the original dataset, Music4All-Onion includes numerous other audio feature sets, pre-computed lyric embeddings, TF-IDF vectors for genres and tags, and image-based representations extracted from YouTube video frames. Furthermore, to provide an additional large

[3] https://www.last.fm/

set of user-item interactions, they filter the LFM-2b dataset by retaining only the events corresponding to Music4All tracks. From the 109,269 tracks, 56,512 were found in LFM-2b, resulting in 252,984,396 filtered listening events, dramatically increasing the scale and diversity of the available user behavior data. This expansion enables training of deep learning models on a realistic, large-scale recommendation setting while also supporting cross-modal learning.

3.3.2 Behavior-Based Methods

Similarly to non-sequential recommendation approaches, early works that tackled the APC task focused solely on collaborative data to discover the hidden factors that influence music consumption. The winning solution to the RecSys Challenge 2018, proposed by Volkovs et al. [112] introduced a two-stage architecture for large-scale APC. Their model consists of a first stage that aims to quickly select 20,000 candidate songs for each playlist, thus dramatically reducing the search space and facilitating the use of more complex and time-consuming techniques in the 2nd stage. The 2nd stage then refines these initial recommendations by re-ranking these songs using a number of features that attempt to capture more complex interactions between songs and playlists. Specifically, the Weighted Regularized Matrix Factorization (WRMF) [48] technique is first used to select the 20k candidate songs for each playlist. However, since this method does not consider the order of songs in a playlist, additional scores are calculated for each candidate song to take into account long-term and short-term dependencies at the playlist level. To this end, a gated CNN model is used to extract order-aware playlist embeddings, as well as classic user-user and item-item neighbor-based models. The four scores mentioned above, along with a linear combination of them, are included as input features in the second stage. Additionally, the second stage also considers various carefully selected song and playlist features, as well as pairwise playlist-song features, focusing on getting the most complete depiction of their similarity. All of the above are used as input to a tree-based gradient-boosting model (XGBoost [18]) with a pairwise ranking loss function, which re-ranks the retrieved songs and produces the final recommendations. With 22.41% R-Precision, 39.46% NDCG and 1.7839 clicks, the authors' team won the 1st place, showcasing the system's ability to provide satisfactory music recommendations on a large scale. At the same time, the results encourage the combination of collaborative filtering methods with more complex features that focus on content and sequential dynamics of the playlist continuation problem.

Building on this insight and the observation that more recent scientific works often neglect the important goal of scalability, Bendada et al. [7] introduced a general-purpose framework for APC that balances scalability and flexibility, aiming to make advanced sequence models (eg. RNNs, Transformers) usable in real-world music streaming systems. Their approach, called Represent-Then-Aggregate (RTA), decomposes the recommendation process into two key components: a song representation function and a playlist-level aggregation function. Song representations can be learned in multiple ways-directly from occurrence data using WRMF, applying an attention mechanism on learned metadata embeddings, or simply taking their average—and are denoted by $h_s = \phi(s)$. These are then aggregated by a function g into a playlist embedding h_p, as shown in Figure 3.3.2, using simple means (like averaging) or more complex sequence models, including gated CNNs (as in [112]), GRUs [20], or Transformer decoders [109]. The final recommendation score is computed using the inner product between the playlist and song embeddings. During training, the representer and the aggregator are jointly optimized with a contrastive loss that distinguishes between true continuations of possible subplaylists of a playlist p and negative samples drawn from the songs that don't belong in p. The paper emphasizes that scalability is maintained by minimizing online operations: playlist representations are computed from embeddings at inference time, and large offline precomputation is used to keep latency low. The task used for evaluation uses the MPD dataset and resembles that of the RecSys Challenge: given some of the first songs of each playlist, produce a ranked list of 500 recommended songs to predict the rest of the playlist. Experiments show that the Transformer aggregator, paired with any of the proposed representers, gives the best performance across all metrics. Notably, one such variant was successfully deployed in production at the music streaming service Deezer, improving performance in large-scale A/B testing and illustrating the framework's industrial readiness.

3.3.3 Audio-Driven Methods

While the previously discussed approaches focus primarily on leveraging interaction data for playlist continuation, more recent efforts explore the integration of audio content into the recommendation process.

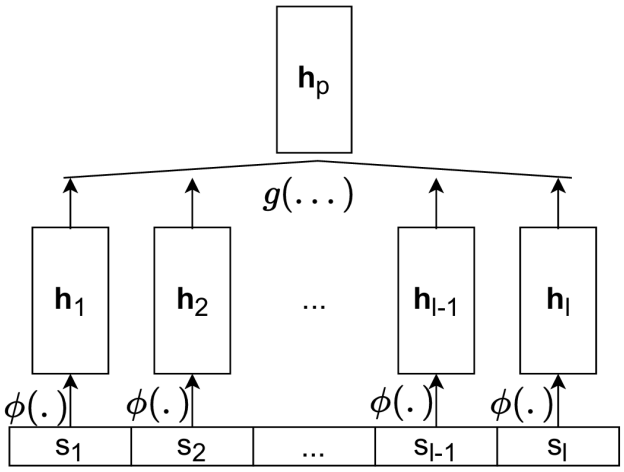


Figure 3.3.2: The Represent-Then-Aggregate (RTA) framework [7]

Meehan and Pauwels [69] examine this direction by conducting a systematic evaluation of contrastive learning methodologies that combine playlist-derived supervision with audio representations. Using the Melon Playlist Dataset [33], they unify several training paradigms from previous works—cross-modal [32] (aligning audio with metadata embeddings, obtained from Word2Vec [70] for genre tags and from matrix factorization for playlists), audio intra-modal [1](learning from co-occurrence of tracks within playlists, artists, or genres), and hybrid strategies that combine the two—into a common experimental framework. Two audio encoders, ResNet18 [42] and a short-chunk CNN [114] are pretrained under these regimes and subsequently assessed on downstream music tagging (MTG-Jamendo [11], MagnaTagATune [58], Melon-50), using a shallow MLP classification head and on automatic playlist continuation. Results indicate that weakly supervised approaches exploiting playlist or artist co-occurrence signals generally surpass self-supervised baselines, with artist-based co-occurrence yielding the best playlist continuation performance and playlist-based training excelling in music tagging. Their proposed hybrid strategy proves particularly robust, achieving consistently strong results across datasets. An additional contribution is the adaptation of mixup augmentation [118], previously confined to self-supervised setups, to weakly supervised contrastive training, where it significantly enhances transfer learning performance. Overall, this study highlights the value of incorporating audio features alongside behavioral signals, demonstrating that such multimodal supervision can yield richer and more versatile music representations for recommendation and tagging.

Building upon the same idea of bridging music representation learning and sequential recommendation, Tamm and Aljanaki [104] study how pretrained audio embeddings behave when fused with user listening histories. To evaluate this, the authors use the Music4All-Onion dataset [72] with a temporal split (last month for validation/test; previous year for training) and benchmark six backend audio representations (MusiCNN [80], MERT [63], EncodecMAE [76], Music2Vec [62], MusicFM [115], Jukebox [29]) by plugging their track-level embeddings into three recommenders of increasing complexity: a content-only K-Nearest Neighbours (KNN) model, a Shallow Net that passes user/item embeddings through a fully connected layer, and BERT4Rec [103] for sequence modeling. For both Shallow Net and BERT4Rec, pretrained item embeddings are kept frozen, while user embeddings are trainable, to learn from the interaction data. Evaluation is done on HitRate@50, Recall@50, and NDCG@50, ranking only previously unheard items and using a model with randomly initialized item embeddings as a baseline. Results show the expected hierarchy of model capacity (BERT4Rec $>$ Shallow Net $>$ KNN) and, crucially, that combining content with collaborative signals substantially boosts performance over content-only KNN. With BERT4Rec, MusiCNN delivers the strongest improvements over the pure collaborative baseline (0.044 vs. 0.038 NDCG@50), while MERT and EncodecMAE perform comparably to it; Music2Vec, MusicFM, and Jukebox (0.020, 0.016, 0.012 NDCG@50, respectively) fall behind the baseline, suggesting some representations require more elaborate integration and also showing the modeling power of BERT4Rec in this sequential setting. The study also notes a task mismatch between MIR and MRS leaderboards as MusiCNN fares best in recommendation despite middling MIR scores, while other models like MusicFM and Jukebox that lead in MIR, underperform in MRS.

Chapter 4

Data and Methodology

The main focus of this thesis is to leverage pretrained models to extract embeddings from raw audio and investigate various ways to incorporate them into a session-based recommendation pipeline. To achieve this, we build upon two existing frameworks we described briefly in the previous chapter. The first one is NVIDIA's Transformers4Rec library [100] and the other one is the Represent-Then-Aggregate (RTA) framework [7] introduced by Deezer Research. In both cases, our data comprises listening sessions, which we extract from a music dataset that provides listening histories from approximately 15k users [91].

In Figure 4.0.1, we present an overview of our approach. As a general idea, the extracted audio embeddings can serve as initial item representations for the songs of an input listening session. This sequence is fed into a Transformer architecture, whose self-attention layers yield sequence-aware representations, which can then be used to predict the next item. In the following sections w,e'll discuss the data setup, as well as the technical details for each of the two frameworks.

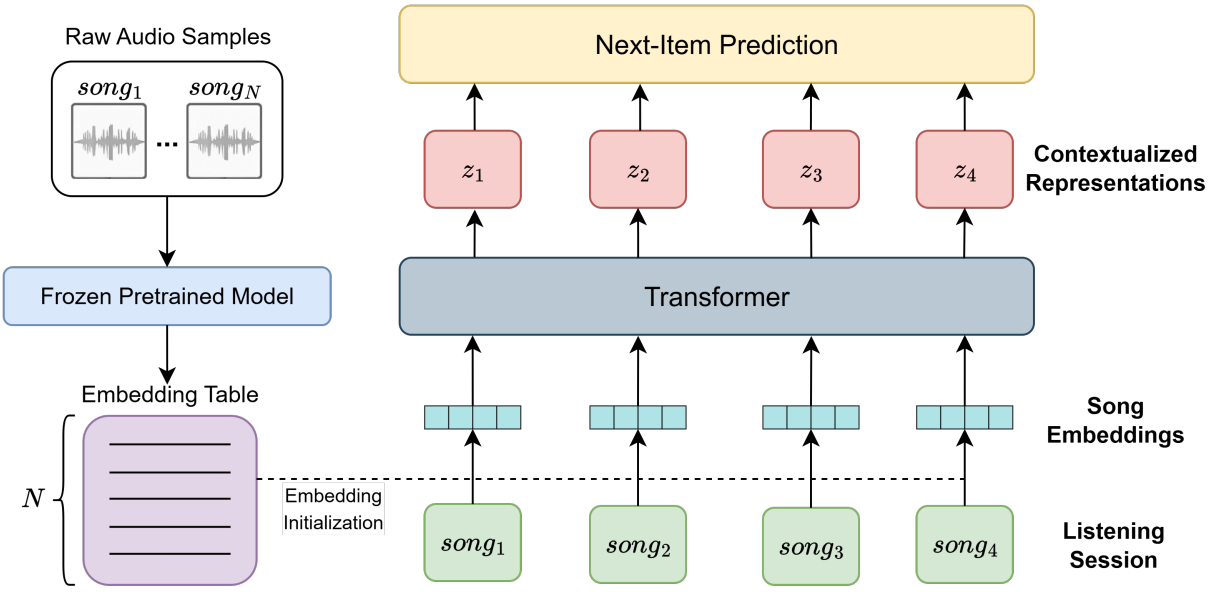


Figure 4.0.1: Generic overview of the proposed approach. Song embeddings are derived from frozen pretrained audio models and used to initialize the item embedding table. A transformer-based architecture then integrates behavioral context, producing contextualized representations for the next-item prediction task.

4.1 Data Preprocessing

Dataset Selection

In selecting the dataset for our experiments, our primary concern was ensuring the availability of raw audio segments, as the pretrained models we employ to extract embeddings are designed to operate directly on raw audio. This requirement immediately ruled out several otherwise popular datasets. For instance, the Melon Playlist Dataset [33], while offering a large number of curated playlists, only provides Mel spectrograms. An option would be to reconstruct the audio from the spectrograms, however, the Melon creators intentionally reduced the resolution to 48 mel bins to prevent copyright violations, making high-quality audio reconstruction impractical. Moreover, Melon is predominantly composed of Korean music, which could introduce cultural or genre-specific biases; our goal was instead to use a dataset that reflects broader and more diverse listening habits. Similarly, while the Million Playlist Dataset (MPD) [15] provides a great volume of user behavior data, it entirely lacks audio features. Attempting to retrieve them via the Spotify API would be prohibitively time-consuming due to API rate limits and the dataset's large size. Taking these limitations into account, we opted for the Music4All dataset [91], which provides 30-second raw audio segments alongside listening histories. Although the full dataset includes audio features for 109,269 tracks, only 99,596 of these appear in the user listening histories, which constitutes our initial track subset.

Session Creation

As mentioned in section 3.3.1, the Music4All dataset contains each user's listening history over a specified three-month window. In order to use this data for our session-based application, we take advantage of the timestamp of each listening event to rearrange the data into sessions. To do so, we follow the method proposed by the creators of the dataset and we create a new session when two consecutive events from the same user are more than 60 minutes apart from each other. In Figure 4.1.1 we present the resulting histogram of session lengths, grouping sessions longer than 50 interactions in one bin.

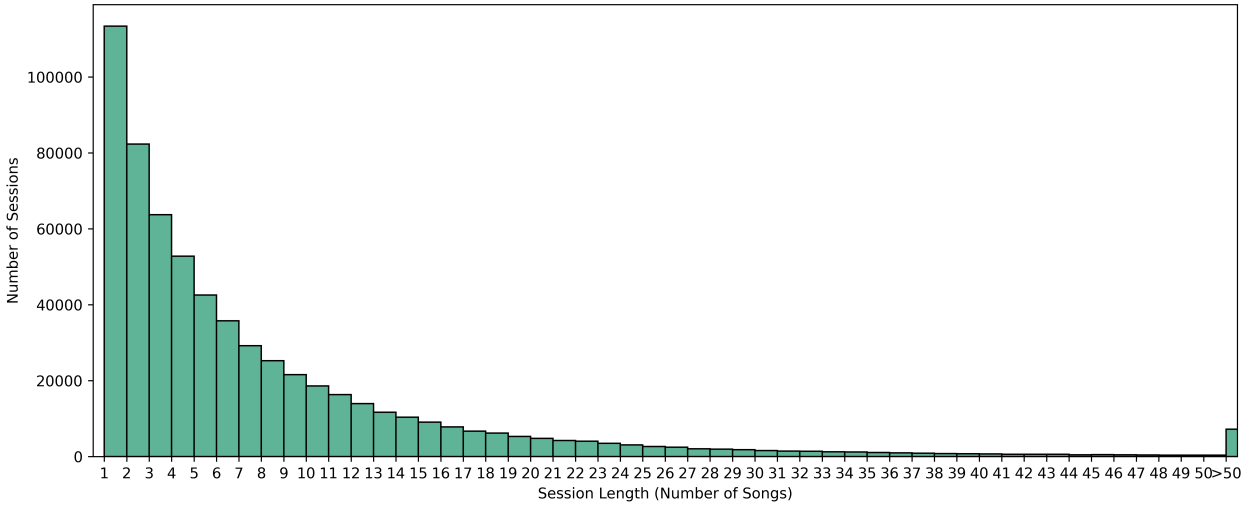


Figure 4.1.1: Histogram of session lengths of the processed Music4All dataset. Sessions with more than 50 interactions are grouped in one bin.

As seen in the histogram, around 113k (18%) of the listening sessions consist of only one event ($l = 1$). Clearly, such sessions are not informative in a sequential recommendation scenario, as they cannot be considered true sequences. Therefore, we make sure to filter them out.

It's worth commenting that the resulting dataset has an average session length of 9.70, which is significantly lower compared to the playlist datasets MPD (66.35) and Melon (41.46), as well as Music4All's original listening histories (361). This is expected for a session-based recommendation application, since listening sessions can be much shorter than whole listening histories or playlists.

Pretrained Model Selection

To extract audio embeddings from the Music4All dataset, we employed three distinct models: MERT and MusiCNN and a custom contrastively trained CNN.

- MERT [63] is a general-purpose, large-scale pretrained model trained using a self-supervised learning approach. It has demonstrated competitive performance across a variety of downstream music information retrieval (MIR) tasks, making it a strong candidate for capturing rich audio representations.
- In contrast, MusiCNN [80] is a CNN trained in a supervised fashion on the music auto-tagging task. We included MusiCNN not only to represent a different training paradigm—supervised learning versus MERT's self-supervised strategy—but also due to its demonstrated utility in music recommendation scenarios, as evidenced in [104].
- Our custom model is a convolutional network based on the EfficientNet-B0 architecture [105], trained on raw audio via a contrastive task that brings tracks from the same artist closer in the embedding space. From now on, we will refer to this model as "artist model".

By incorporating models trained under different paradigms, we aim to better understand how the nature of the training objective influences the quality of audio embeddings in the context of session-based music recommendation.

Audio Preprocessing

All audio files in the Music4All dataset are provided in stereo format with a sampling rate of 48 kHz. As a preprocessing step, we convert each file from stereo to mono by averaging the two channels. We then truncate or pad each audio sample to ensure a consistent length of 30 seconds. Since each model expects a different sampling rate (24 kHz for MERT and 16 kHz for MusiCNN and the artist model), we resample all audio to match this requirement.

Embeddings Extraction

To extract embeddings with MERT, we use the lighter 95-million parameter variant[4], which consists of 12 transformer layers and produces a 768-dimensional output at each layer with a temporal resolution of 75 Hz. In our setup, we use the representations from the final (12th) layer, resulting in a tensor of shape (num_seconds $\times$ 75, 768). We then apply mean pooling over the time axis to obtain a single 768-dimensional vector per audio track.

For MusiCNN, we use the smaller model variant trained on the Million Song Dataset. As our feature vector, we use the 200-dimensional projection of the temporal pooling layer, referred to as the "penultimate" feature set in Figure 3.1.5. The official MusiCNN implementation processes raw audio by computing a spectrogram over a fixed-length patch. Since the model was trained on 3-second segments, the authors recommend using the same duration at inference time. Therefore, in our case each 30-second audio clip is split into ten non-overlapping 3-second segments, each yielding a 200-dimensional embedding. We then average these ten vectors to produce a single 200-dimensional embedding per track.

For the artist-informed audio model, we perform contrastive pre-training using Music4All [91], using artist identity information to form the contrastive pairs until convergence. The model accepts 1-sec mel-spectrograms as input, and employs an EfficientNet-B0 encoder to produce embeddings of dimensionality equal to 1280. In order to produce a single embedding for each audio excerpt, all 1-sec excerpt embeddings were averaged across the length of each musical piece.

Finetuning of MusiCNN Embeddings

Motivated by the findings of [82], we finetune the audio embeddings with behavioral data, to align them with the final task. Specifically, we feed a shallow MLP model with the extracted MusiCNN embeddings and train it with a contrastive loss, treating songs from the same session as positives and the rest of each batch as negatives. After training, we perform a final forward pass to get the tuned audio embeddings. We refer to this approach as MusiCNN-tuned.

4.2 Transformers4Rec (T4Rec)

To test the usefulness of the extracted embeddings in session-based recommendation, we first propose a model built upon the T4Rec library, that borrows techniques from popular sequential recommendation models like BERT4Rec [103].

Dataset

For our experiments with T4Rec we use the preprocessed sessions from the previous section, in which we have excluded single-song sessions. In Table 4.1, we present some basic characteristics of the Music4All listening data, before and after removing the aforementioned sessions. As we said, around 18% of the sessions are removed, which results in a total of 514,786 sessions and a slightly higher average session length.

Additionally, we employ a 70-15-15 split, on data sorted by user IDs. This means that we have completely new users in the validation and test set.

[4] https://huggingface.co/m-a-p/MERT-v1-95M

Statistic	Original Music4All	T4Rec
Users	14,127	14,127
Songs	99,596	99,355
Number of sessions (l>1)	628,204	514,786
Filtered sessions with one song	-	113,418 (18.05%)
Average session length	8.13	9.70

Table 4.1: Statistics of the T4Rec listening sessions before and after the filtering steps.

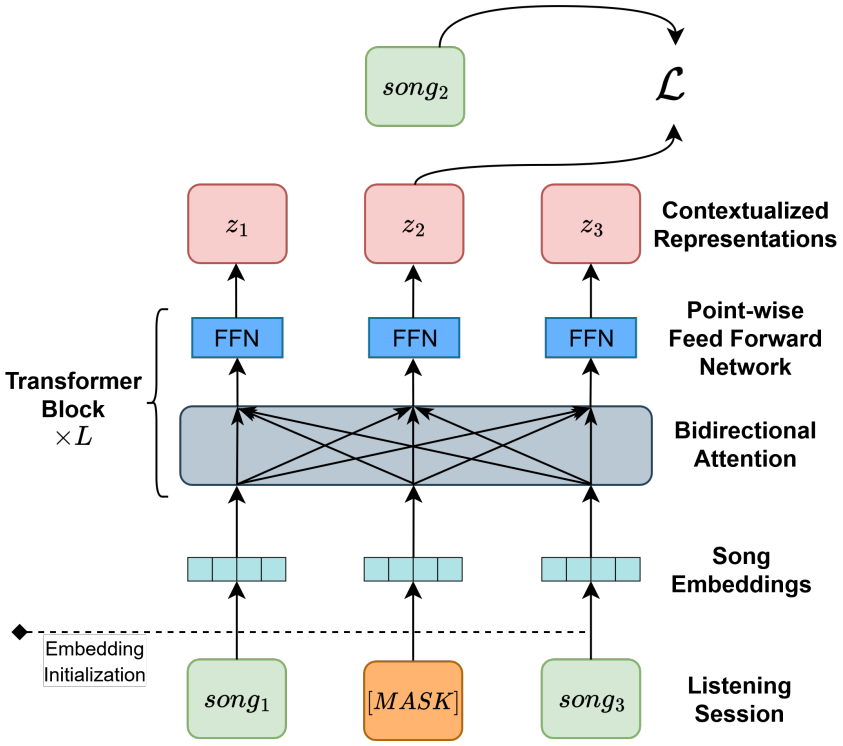


Figure 4.2.1: Architecture of our T4Rec-based models. During training, the model masks parts of the input session and uses bidirectional attention to recover them.

4.2.1 Architecture

Input Layer

The input of the model is a sequence of song IDs, which corresponds to a listening session. This sequence passes through an embedding layer, which is essentially the item embedding matrix $E \in R^{|V| \times d}$, where V is the item set and d is the embedding dimension. This matrix can be randomly initialized and learned from scratch during training, or can be initialized using the pretrained audio embeddings, providing a content-aware starting point. In the case of a mismatch between the embedding dimension and the model dimension (e.g. we train a model with MusiCNN embeddings with d=200, but we want the model to have dimension 64), we add a linear projection layer followed by a RELU activation function. Since the self-attention mechanism is order-agnostic, a positional encoding is added to each embedding (after the projection) to keep track of song order. Instead of the absolute positional encoding used in the original transformer, which assigns a fixed embedding to each position, we use the relative positional encoding introduced in TransformerXL [24], focusing on encoding the distance between items and not their absolute position. It is important to mention that in all our experiments the pretrained embeddings are not frozen, because this resulted in significantly worse performance, as we will discuss in the next chapter.

Transformer Layers

The interaction sequence is then fed to the transformer network, which refines the item representations, capturing valuable information about song relationships and co-occurrence patterns and produces a contextualized representation for each item of the sequence. As we described in the previous chapter, the Transformer4Rec framework allows the user to apply popular transformer architectures for session-based recommendation tasks. In this work, we chose to build our model as a stack of transformer layers based on the layer architecture of XLNet [116], following the original T4Rec paper [100]. We should clarify that this doesn't mean that we adopt the exact configuration of the XLNet model (number of layers, hidden dimension, number of attention heads etc.), but rather our transformer layers use the internal mechanisms found in XLNet layers.

Output Layer

Supposing that we want to predict a missing item at time step t, we score all items by multiplying the missing item's final representation with an output projection matrix of shape $|V| \times d$. Importantly, this matrix can share the same weights with the input embedding matrix E. This is a common technique called weight tying, which significantly reduces the model's parameters, since such an embedding table has a considerable size, given that the number of $|V|$ of possible items is quite large in RecSys applications. In order to be able to apply weight tying, we make sure to project the item representation back to the embedding dimension, if needed, using a linear layer. Finally, the scores of all items go through a softmax function, which produces a probability distribution over the item set, expressing how probable it is that each item is the missing item.

4.2.2 Training

Our model is trained using the Masked Language Modeling (MLM) training scheme used in BERT4Rec, as described in section 3.2. This enables the use of bidirectional context, i.e. considering both past and future interactions when predicting an item, while preventing information leakage due to the availability of future context. According to MLM, during training we replace a fraction ρ of items in the input with a [MASK] embedding, and the model learns to recover the masked items using the surrounding context. More precisely, for each masked position, we use its final representation to produce a probability distribution over the item set, as described above. Then, we take the probability the model has assigned to the true item and compute its negative log-likelihood. Minimizing this quantity is equivalent to maximizing the probability of the true item. Therefore, for a masked session S'_i the loss is formulated as:

$$\mathcal{L} = \frac{1}{|\mathcal{S}_i^m|} \sum_{v_m \in \mathcal{S}_i^m} -\log P(v_m = v_m^* | \mathcal{S}'_i) \tag{4.2.1}$$

where $\mathcal{S}_i^m$ are the masked items in S'_i and v_m^* is the true item for the masked item v_m.

4.2.3 Evaluation

When running the model on the validation or test set, we evaluate its ability to predict the last item of each session, given all previous interactions. That is, only the last item of each session is replaced with a [MASK] token and predicted by the model, which scores all items and returns a ranked list of recommendations. This list is evaluated using several ranking metrics, which we have defined in section 2.4, namely NDCG@k, MAP@k and Recall@k. We report these metrics for $k = 10, 20$. Finally, during inference, the input session is extended with an additional [MASK] embedding and all previous interactions are used to predict the next item.

4.2.4 Late Fusion

Among our goals is to experiment with alternative ways of integrating the pretrained audio embeddings into the model. One idea is to randomly initialize the item embedding layer of the transformer and let the transformer layers take full advantage of the behavioral data. Then, we take the output of the transformer and concatenate it with the average audio embedding of the session, excluding those of the masked items.

Another option would be to concatenate each item's hidden representation with its corresponding audio embedding, however this would lead to information leakage, as the model would simply learn to base its predictions on the audio embeddings, which remain unmasked. That is also the reason why we don't consider the masked items when computing the session average.

4.3 Represent-Then-Aggregate (RTA)

The second part of our work includes training an RTA-based model on the Music4All dataset to examine how the audio embeddings behave in a different task formulation. As we recall from section 3.3.2, the RTA framework uses a function to obtain a representation for each song and then applies an aggregation function to represent a session p as a combination of its songs' representations. In comparison to the Transformers4Rec framework, we can identify the following core differences:

- RTA removes duplicate songs from each session, which results in a slightly different dataset, since in T4Rec we only remove single-song sessions.
- The original RTA paper follows a different approach to split the dataset, randomly sampling a considerably smaller validation and test set.
- Each framework uses its own training objective. While in T4Rec we follow an MLM approach, partially masking the input and predicting the missing items, RTA maximizes the similarity between the session representation and the embedding of the true next song.
- This difference in training objective also motivates a different architecture: the bidirectional self-attention used in T4Rec is replaced with causal self-attention in RTA.
- Finally, the two frameworks differ in their evaluation task. In T4Rec, we use the whole session to predict the last song, whereas RTA uses a small number of first songs to retrieve all the remaining ones.

Dataset

As we just mentioned, in RTA, we apply an additional preprocessing step used in the original codebase, in order to ensure compatibility with the rest of the framework. Specifically, apart from removing sessions with only one song, we also remove duplicate songs from each session. That is, if a song is consumed numerous times within a session, we keep only the first interaction and reject subsequent ones. This step leaves around 20k more sessions with only one song, which we make sure to ignore later on. The characteristics of the Music4All subset used in our RTA experiments can be seen in Table 4.2, where we can also observe a decrease in the average session length, since all the duplicates are removed within each session.

Statistic	**Original Music4All**	**RTA**
Users	14,127	14,127
Songs	99,596	99,355
Number of sessions (l>1)	628,204	493,297
Filtered sessions with one song	-	134,907 (21.47%)
Average session length	8.13	8.06

Table 4.2: Statistics of the RTA listening sessions before and after the filtering steps. Apart from the steps applied in T4Rec, here we also filter out duplicates from each session, keeping only the first interaction.

Regarding the data split for this experiment group, we don't differentiate from the original work, so we sample 20,000 sessions with length $l \geq 20$ to create a validation and a test set of 10,000 sessions each. The rest 473,297 sessions form our training set.

4.3.1 Architecture

Apart from the methods used in [7] as representation functions, which we'll use as baselines, in our work we leverage the pretrained models MERT and MusiCNN to obtain song representations. Concerning the aggre-

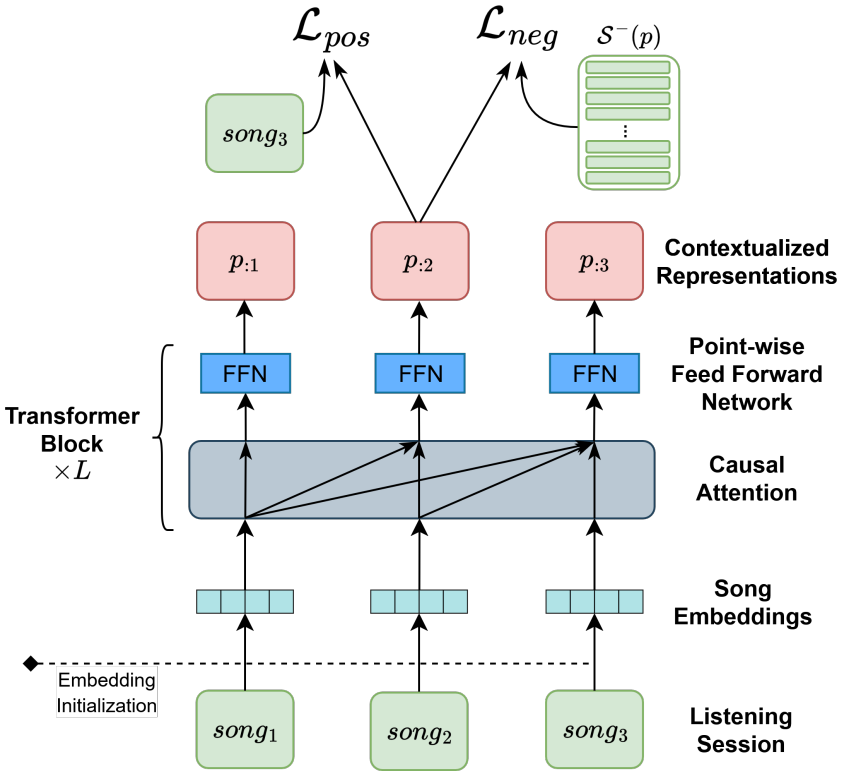


Figure 4.3.1: Architecture of our RTA-based models. At time step i, the transformer aggregates previous items into a session representation $p_{:i}$ with causal attention, and uses $p_{:i}$ for next-song prediction. Training uses a hybrid loss that encourages similarity with the true next song ($\mathcal{L}_{pos}$) and discourages similarity with a set of negative samples $S^-(p)$ ($\mathcal{L}_{neg}$).

gation function, the results of the original work showed that the models that used a transformer architecture dominated over other options, such as GRUs, Gated CNNs, or a simple average of the song representations. For this reason, we focus only on the transformer variant for our experiments. This processes the input sequence p using masked self-attention blocks, i.e. the model can attend only to positions up to the current item i and not to future ones, much like SASRec [53] does (see section 3.2). The resulting output can be seen as a session representation that aggregates songs $p_{s_1}, p_{s_2}, \ldots, p_{s_i}$ and can be used to predict the next song $p_{s_{i+1}}$, by scoring all item representations using a dot product. Notice that this essentially uses the weight tying technique we mentioned previously. Throughout our experiments, the song representations are kept trainable, as in the case of T4Rec.

4.3.2 Training

To train this model, the authors propose the following objective: given a session p with length l, and the aggregation $p_{:i}$ of its first i songs, with $i \in \{1, \ldots, l-1\}$, encourage a higher score between $p_{:i}$ and the true next song $p_{s_{i+1}}$, and a lower score for a sampled set of songs $S^-(p)$ that don't belong in p. The true continuation functions as a positive pair while the out-of-session songs as negatives and therefore the total loss can be expressed as $\mathcal{L}(p) = \mathcal{L}_{\text{pos}}(p) + \mathcal{L}_{\text{neg}}(p)$, with:

$$\mathcal{L}_{\text{pos}}(p) = -\sum_{i=1}^{l-1} \log \sigma\left(f\left(p_{:i}, p_{s_{i+1}}\right)\right) \tag{4.3.1}$$

$$\mathcal{L}_{\text{neg}}(p) = -\sum_{i=1}^{l-1} \sum_{s^- \in \mathcal{S}^-(p)} \log \sigma\left(f\left(p_{:i}, s^-\right)\right) \tag{4.3.2}$$

where σ denotes the sigmoid function and f the inner product.

4.3.3 Evaluation

To evaluate the model, we follow [7] and split the 10k sessions of the validation or test set into 10 configurations with 1,000 sessions each: for the first subset only the first song is visible, for the next subset only the first two songs and so on and so forth, up to 10 visible first songs. The remaining part of each session is to be predicted by the model, which scores all songs and gives a ranked list of candidate songs for each session.

4.3.4 Baselines

To check the effectiveness of the pretrained audio representations, we compare them with the three functions proposed in RTA, which act as our baselines:

- Weighted Regularized Matrix Factorization (WRMF) [48]: Factorizes the session-song occurrence matrix using the Alternating Least Squares (ALS) method (see section 2.2.2) to extract initial song representations with $d = 128$. Denoted as MF in our experiments.
- Factorization Machine (FM): Using the above song representations, we create embeddings for song metadata (artist, album, popularity) by taking the average of the songs that have the same metadata value. Each song's embedding is then the average of the embeddings of its metadata.
- Neural Network (NN): Similar to FM, but instead of averaging metadata embeddings, it applies an attention mechanism on them.

Chapter 5

Experimental Evaluation

As analysed in previous chapters, despite the effectiveness of pretrained audio representations in several MIR tasks, there have been few efforts that take advantage of them in the music recommendation domain, even though the need for content-driven methods to alleviate common problems of collaborative-only approaches remains topical.

The experiments described in this chapter aim to shed light on whether these representations can be useful in a music recommendation task and specifically in a session-based setting. Furthermore, since the models we used to extract audio representations are built on different training paradigms, our work is also expected to draw conclusions on how this choice affects the recommendation performance. At the same time, the way of integrating the pretrained embeddings into the recommendation pipeline is not trivial, and thus we try out various methods to bring them into play, hoping to gain insight into this aspect as well.

5.1 Experimental Setup

5.1.1 Evaluated Models

In our experiments, the basic way of integrating pretrained embeddings into the recommendation pipeline is by using them as initialization for the item embedding matrix E, hoping to provide the recommendation model with a helpful starting point. As a baseline, we use a random initialization of the embedding matrix and let the model learn solely from the behavioral data. Of course, in this case E is set to be trainable. In the case of pretrained initialization, prior works [104] suggest that whether E is frozen or not doesn't have a great impact on performance. However, our preliminary experiments showed that freezing the embedding table gave much worse results and thus we let it be modifiable throughout our experiments.

As stated previously, we will use embeddings from three pretrained models: MusiCNN [80], MERT [63] and our custom artist model, as well as a modification of the MusiCNN embeddings finetuned on the listening sessions. Apart from employing a different training approach, each model has its own embedding dimension d (200, 768 and 1280, respectively). Therefore, training a single baseline model with e.g. embedding dimension $d = 512$ would not guarantee a fair comparison, since each model would have a different capacity (higher or lower) from the baseline. In order to isolate the effect of the pretrained initialization and ensure that the compared models have the same capacity, we train three different randomly initialized models, each one matching the embedding dimension of a pretrained model.

Moreover, in our Transformers4Rec experiments, we also test the late fusion technique introduced in section 4.2.4, both for MusiCNN and MERT embeddings. In this case, we set the embedding dimension $d = 512$. So, for example, if the transformer's output dimension is 256, then the result after the fusion would be $256 + 200 = 456$ for MusiCNN, which is then projected to $d = 512$.

As for RTA, apart from the randomly initialized models that serve as baselines, we additionally include the three baselines described in section 4.3.4, along with another random model that has the same embedding dimension ($d = 128$) as these baselines.

5.1.2 Training Details

All our models were trained on a single GPU and we make sure to save the checkpoints with the best NDCG@20 on the validation set. We then run these checkpoints on the test set to obtain the reported results.

For the hyperparameters of the T4Rec transformer we follow BERT4Rec [103] and use 2 transformer layers and 2 attention heads in each layer. Our preliminary results agree with the findings of BERT4Rec that increasing the number of layers and heads can improve performance, however we stick to the above values to make our models more computationally efficient. What's different in our work is each layer's hidden dimension, which we set to be 512. Finally, for other training hyperparameters, like the optimizer and the learning rate, we keep the default training arguments from the HuggingFace Transformers library [113]. We train all our T4Rec models for 50 epochs, using the Adam optimizer and applying linear decay to the learning rate, starting from 0.00005.

Concerning our RTA experiments, we generally adopt the hyperparameter values suggested in the original

work. Models are trained for 30 epochs, using Stochastic Gradient Descent (SGD) as optimizer and halving the learning rate every 2-3 epochs.

All the aforementioned hyperparameters are presented in Table 5.1.

Table 5.1: Training and architecture details for Transformers4Rec and RTA models.

Hyperparameter	**Transformers4Rec**	**RTA**
Number of layers	2	1
Number of heads	2	8
Model dimension	512	Equal to embedding size
Batch size	128	128
Epochs	50	30
Optimizer	Adam	SGD
Learning rate scheduler	Linear	Step
Initial learning rate	0.00005	1.0 (0.13 for NN baseline)

5.2 Quantitative Results

5.2.1 Transformers4Rec

The results for all T4Rec models on the test set are presented in Table 5.2 to facilitate comparison. The second column notes each model's embedding dimension d, and we remind that the transformer's hidden dimension is 512 for all models. The underlined results correspond to the best performance among a certain group of models (e.g. MERT initialization and random initialization with the same embedding dimension), while the bold results signify the overall best performance on the specific metric.

Table 5.2: Performance comparison of T4Rec models across all evaluation metrics.

Initialization	**d**	**NDCG@10**	**MAP@10**	**Recall@10**	**NDCG@20**	**MAP@20**	**Recall@20**
Random	200	0.2504	0.2013	0.4038	0.2629	0.2047	0.4530
MusiCNN	200	0.2225	0.1788	0.3603	0.2358	0.1824	0.4124
MusiCNN-tuned	200	0.2452	0.1953	0.4009	0.2583	0.1988	0.4525
Random	768	0.2618	0.2140	0.4105	0.2742	0.2174	0.4595
MERT	768	0.2582	0.2089	0.4089	0.2706	0.2122	0.4580
Random	512	0.2676	0.2216	**0.4119**	0.2805	0.2252	**0.4626**
MusiCNN late fusion	512	0.2655	0.2197	0.4088	0.2782	0.2231	0.4589
MERT late fusion	512	0.2628	0.2158	0.4096	0.2758	0.2193	0.4607
Random	1280	**0.2684**	**0.2232**	0.4095	**0.2808**	**0.2266**	0.4583
Artist model	1280	0.2309	0.1848	0.3739	0.2451	0.1886	0.4297

MusiCNN

Starting with the models using embedding dimension $d = 200$, we observe that random initialization yields the best performance across all metrics within this group. The MusiCNN-initialized model performs substantially worse, suggesting that the representations it uses—originally trained for auto-tagging—may not align well with user behavior patterns in the session-based recommendation setting. However, when these MusiCNN embeddings are finetuned on the user data, their performance improves consistently across all metrics. This demonstrates that although the pretrained audio features may not be optimal for the recommendation task in their original form, they can still serve as a useful starting point when adapted through finetuning. Nevertheless, even the tuned version fails to match the performance of the randomly initialized model, highlighting the strength of learning embeddings directly for the task at hand.

MERT

In the next group with $d = 768$, we compare the performance of MERT-based initialization to the random baseline. Here, the random model again shows a slight edge, although the performance gap is narrower than in the $d = 200$ group. This indicates that the MERT model, which is pretrained via a self-supervised learning objective, provides more useful representations for the recommendation task than MusiCNN. These results suggest that the inductive bias introduced by MERT's pretraining may be better aligned with downstream tasks involving user preferences. However, the fact that random initialization still outperforms it (albeit marginally) reinforces the observation that models benefit from learning embeddings that are specifically tailored to the session-based recommendation objective, rather than relying entirely on features derived from audio content.

Late Fusion

The group with $d = 512$ explores the effect of late fusion, where pretrained audio embeddings are combined with the trainable item embeddings rather than replacing them. In this configuration, the random baseline remains slightly ahead, but the late fusion variants using MusiCNN and MERT are competitive. The performance differences among them are small, with MusiCNN late fusion slightly outperforming MERT in this setup. These results indicate that while audio-based representations alone may not be sufficient, they can provide complementary information when fused with task-specific embeddings. Late fusion allows the model to selectively incorporate audio features without being constrained by them, which helps explain their relatively strong performance. However, the consistent lead of the fully random model suggests that even with fusion, content-based information does not fully bridge the gap with embeddings optimized directly for recommendation. Moreover, we should recall that the adopted technique combines the item embeddings with a session-level audio representation, which might flatten out the song-level granularity of the original song representations.

Artist Model

Finally, in the $d = 1280$ group, we compare random initialization to embeddings derived from an artist-level model. Firstly, the random model with this higher-dimensional embedding space achieves the best overall performance across most metrics in the table, indicating that increased representational capacity can benefit the model when learning from scratch. In contrast, the artist-level model performs significantly worse, even underperforming all other audio-based initializations. This is likely due to the coarse nature of artist embeddings, which lack the detail needed to differentiate tracks within an artist's catalog—an essential property in a recommendation scenario that operates at the item (track) level. The poor results further support the conclusion that simple metadata-derived representations are insufficient in capturing the complex dynamics of user behavior.

Overall, we observe a consistent trend: models initialized with random embeddings tend to outperform those relying on audio-based features, especially when the embeddings are large enough to support expressive representations. Fine-tuning and fusion strategies help bridge the gap between pretrained content-based models and randomly initialized ones, but do not surpass them. These findings suggest that while audio-based representations offer valuable prior knowledge, their usefulness in session-based recommendation tasks is limited unless they are carefully adapted or combined with learnable representations that are directly optimized for the target task.

Impact of Model Dimensionality

The plots in Figure 5.2.1 offer additional insights into how varying the transformer's hidden dimension affects model performance across different initialization strategies. They show how NDCG@20 changes as the hidden dimension varies from 64 to 512, while keeping the rest of the hyperparameters unchanged. Each subplot focuses on a different group of models, grouped by their item embedding dimensionality. Across most settings, we observe that increasing the transformer's hidden size generally improves performance, although the magnitude and consistency of these gains differ between groups.

In the **MusiCNN group** (top left), all three variants—MusiCNN, MusiCNN-tuned, and Random 200—demonstrate clear performance improvements as the transformer's hidden size increases. The randomly

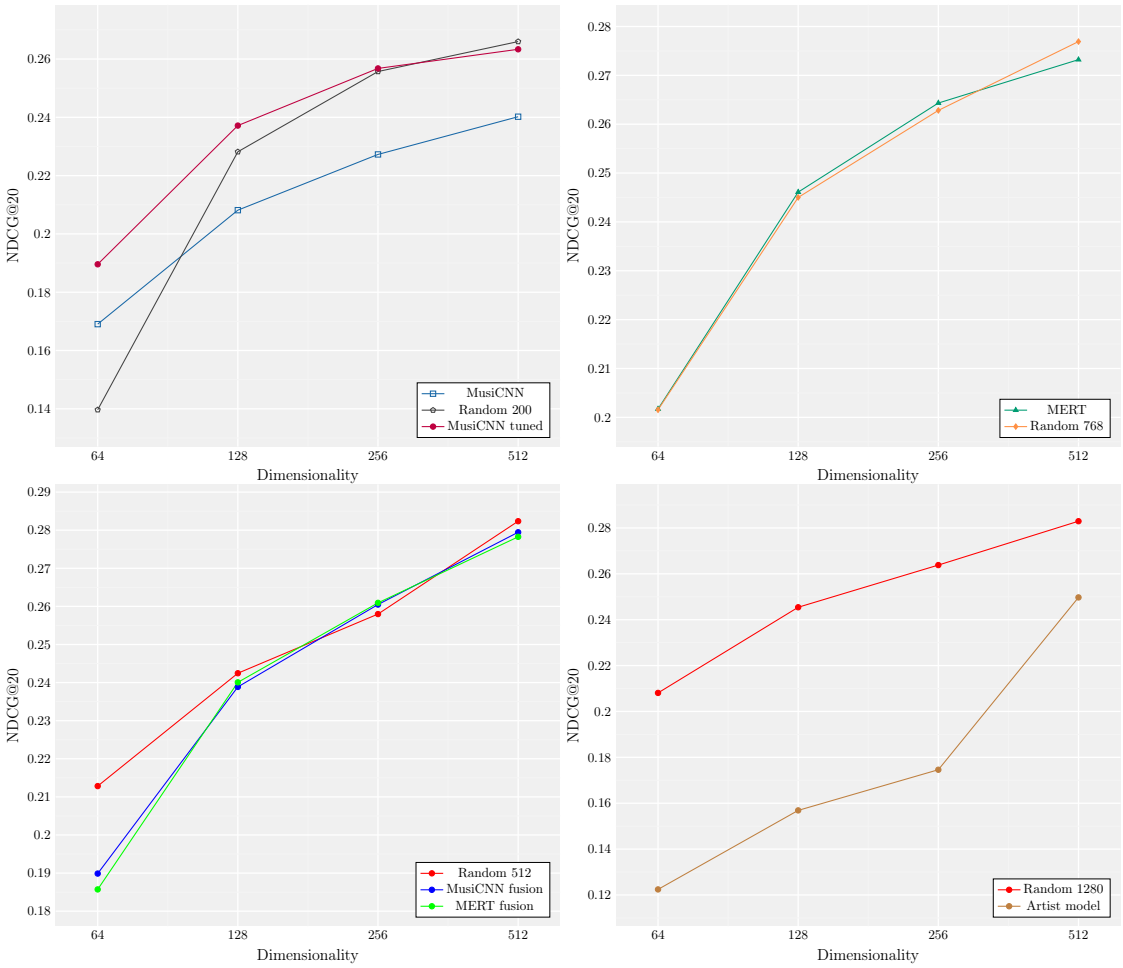


Figure 5.2.1: Effect of the Transformer hidden dimensionality on NDCG@20 for the different model groups.

initialized model starts at a lower baseline but benefits significantly from higher model capacity, eventually catching up with and slightly surpassing the tuned MusiCNN model. For all other hidden sizes, the fine-tuned MusiCNN variant consistently outperforms both other models, illustrating the value of adapting pretrained audio features to the recommendation task. These results suggest that while out-of-the-box audio embeddings are somewhat informative, tuning them beforehand is crucial, especially in lower-capacity models. However, as model capacity increases, randomly initialized models can eventually match or exceed the performance of models using pretrained embeddings.

In the **MERT group** (top right), both the MERT-initialized and Random 768 models show nearly identical performance trajectories across different transformer sizes. This similarity suggests that, for this task and dataset, MERT embeddings do not provide a significant advantage over random initialization when sufficient model capacity is available. The observed improvement with increasing hidden size in both cases indicates that the transformer layers play a crucial role in learning effective session representations, regardless of the initial item embedding.

The **late fusion group** (bottom left), which includes models using late fusion between pretrained audio embeddings and item embeddings learned from scratch, exhibits a consistent upward trend across all three variants. For $d = 64$, the model with random initialization performs considerably better than the two fusion-based variants, but at higher hidden sizes the three models converge toward similar performance. This behavior implies that incorporating audio information by fusing a session-level representation to each item embedding doesn't benefit performance, as the model can learn strong representations from the interaction patterns alone.

In contrast, the **artist model** initialization (bottom right) consistently underperforms relative to the Random 1280 baseline, despite improvements as the hidden size increases. It begins with a notably low score at hidden size 64 and improves steadily, yet always remains behind the random initialization counterpart. This result emphasizes the limited expressiveness of high-level categorical features like artist identity for item-level recommendation. Even as model capacity grows, the coarse-grained nature of these features cannot match the flexibility of randomly initialized embeddings that are trained end-to-end for the task.

In summary, increasing the transformer's hidden dimension generally leads to better performance across all initialization strategies. However, the degree of improvement and the final performance level depend on the quality and flexibility of the item representations. Pretrained features—especially when tunable—offer benefits in lower-capacity models, but their advantage diminishes as the model becomes more expressive. Meanwhile, random embeddings consistently benefit from increased hidden size, often outperforming audio representations when the model is sufficiently deep.

5.2.2 RTA

Table 5.3: Performance comparison of RTA models across all evaluation metrics.

Initialization	d	NDCG@10	MAP@10	Recall@10	NDCG@20	MAP@20	Recall@20
Random	128	0.1541	0.0919	0.0613	0.1331	0.0644	0.0940
MF	128	0.1708	0.1035	0.0674	0.1467	0.0725	0.1028
FM	128	0.2022	0.1350	0.0810	0.1751	0.0980	0.1243
NN	128	**0.2071**	**0.1371**	**0.0838**	**0.1806**	**0.1006**	**0.1298**
Random	200	0.1499	0.0900	0.0600	0.1302	0.0637	0.0926
MusiCNN	200	0.1668	0.1001	0.0660	0.1434	0.0702	0.1005
MusiCNN-tuned	200	0.1699	0.1042	0.0669	0.1442	0.0721	0.0999
Random	768	0.1588	0.0957	0.0618	0.1339	0.0656	0.0917
MERT	768	0.1507	0.0904	0.0604	0.1303	0.0635	0.0925
Random	1280	0.1522	0.0910	0.0600	0.1301	0.0630	0.0910
Artist model	1280	0.1819	0.1137	0.0722	0.1579	0.0814	0.1116

Similarly to Table 5.2, Table 5.3 shows the test set metrics for all our RTA-based models. Again, the second column corresponds to the dimension d of each item embedding, with the difference that in the RTA case, the embedding dimension matches the transformer's hidden dimension. So, for example, if $d = 200$, then the transformer will also have the same hidden dimension.

RTA Baselines

Within the group of models trained with an embedding dimension of d=128, we observe that those proposed in the RTA paper progressively improve upon the Random baseline. MF already yields better scores than Random across all metrics, and FM further improves on MF, suggesting that leveraging metadata or richer interactions helps the model capture more structure in the data. Among these, the NN model, which applies attention to the metadata embeddings, achieves the highest performance in this group, with NDCG@20 reaching 0.1806 compared to 0.1331 for Random. This demonstrates that even a simple Matrix Factorization initialization, paired with metadata information and an attention mechanism, can extract more predictive power from the same data compared to a simple random initialization.

MusiCNN

When moving to d=200, we compare MusiCNN embeddings and their tuned variant to the Random baseline. Here, both MusiCNN variants outperform the randomly initialized model, which contrasts with the trend seen in the T4Rec experiments. NDCG@20 improves from 0.1302 for Random to 0.1434 for MusiCNN and 0.1442 for MusiCNN-tuned, with similar gains observed across MAP and Recall. This indicates that, within the RTA framework, audio-based representations can provide a stronger starting point than random

initialization. Moreover, fine-tuning once again yields consistent improvements, confirming that pretrained features are most beneficial when adapted to the specific recommendation objective.

MERT

At d=768, we see a different pattern with the MERT initialization. Here, the Random baseline slightly outperforms MERT across most metrics, with NDCG@20 at 0.1339 for Random compared to 0.1303 for MERT. This suggests that not all pretrained representations transfer equally well in the RTA setting. While MERT embeddings are rich and learned through self-supervision, they may not align perfectly with the interaction-based signals that RTA leverages. The relatively small performance gap implies that MERT still provides meaningful information, but the framework does not appear to benefit from it beyond what it learns end-to-end.

Artist Model

Finally, at d=1280, the Artist-level embeddings achieve better performance than Random across all metrics, which is the opposite of what we observed in the T4Rec experiments. As an example, NDCG@20 improves from 0.1301 with Random to 0.1579 with the Artist model. This suggests that in the RTA framework, even coarse-grained features such as artist identity can provide useful priors when aggregated over sessions. The aggregation step may smooth out some of the granularity issues, allowing the model to capture high-level stylistic patterns that correlate with user behavior. Nevertheless, while improved over Random, these results remain below the best-performing NN and FM models.

Across all configurations, the RTA results present a more nuanced picture than the T4Rec experiments. Here, pretrained features such as MusiCNN and even coarse artist-level embeddings can offer tangible improvements over Random initialization, particularly when fine-tuned. Classical approaches like MF and FM also demonstrate strong baselines, while the simple NN model leads the performance within the d=128 group. These findings suggest that the RTA framework is more amenable to incorporating externally derived representations, possibly due to its explicit separation of representation learning and session aggregation. However, even in this setting, the gains from pretrained audio embeddings remain modest, and the best results still come from models that learn task-specific representations or blend content-based priors with interaction-driven learning.

5.3 Analysis of Model Behavior

While the previous section focused on the quantitative comparison of models across frameworks, here we take a closer look at how these models learn and what their internal representations reveal about the underlying dynamics of each approach. The goal of this analysis is to interpret the numerical trends observed in Section 5.2 — in particular, the contrasting behavior of the pretrained and randomly initialized models between the two frameworks — by examining their training curves and embedding spaces.

We begin by analyzing the convergence behavior of the different MusiCNN variants and the random baseline, using their validation performance throughout training to understand how pretraining, fine-tuning, and trainability affect learning speed and final performance. Also, we visualize the embedding spaces learned by the T4Rec models through t-SNE projections, which allow us to qualitatively assess how the various initialization strategies structure musical content and how this structure evolves after training.

5.3.1 Convergence Behavior

To investigate how different initialization strategies influence the learning process, we monitor the evolution of validation NDCG@20 across training epochs for the various MusiCNN configurations and the random baseline (Figure 5.3.1). This comparison includes a frozen variant, where the pretrained embedding table is kept fixed during training, as well as the trainable MusiCNN, fine-tuned MusiCNN, and randomly initialized models.

At the beginning of training, the MusiCNN-tuned model starts from the highest validation score, indicating that the lightweight contrastive fine-tuning step has already aligned the embeddings more closely with user-

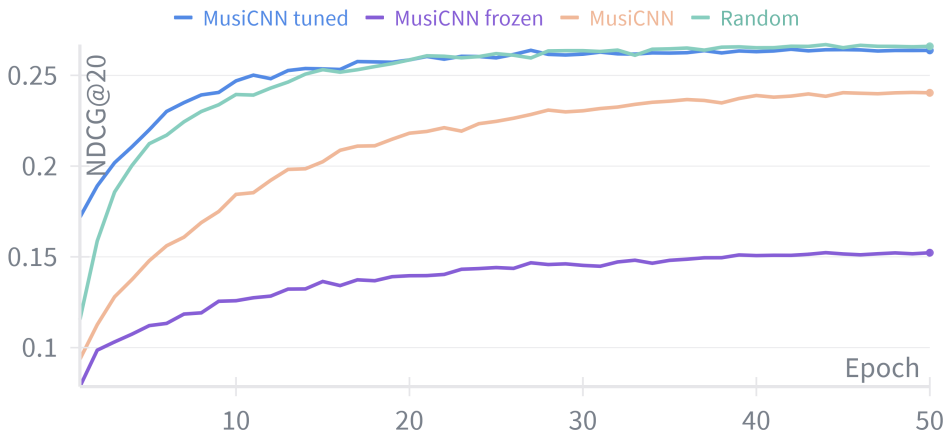


Figure 5.3.1: Learning curve of the different MusiCNN variants, showing NDCG@20 on the validation set

behavioral patterns. The random model, by contrast, begins from a lower baseline but improves steadily and eventually surpasses MusiCNN around the midpoint of training. Both curves plateau between epochs 10 and 15, reaching similar final performance ($\tilde{0}$.26–0.27 NDCG@20). The base MusiCNN model, without fine-tuning, converges more slowly and stabilizes at a clearly lower level, while the frozen MusiCNN shows little improvement, remaining well below all other variants throughout training.

These dynamics highlight two important trends. First, trainability is essential: the clear under-performance of the frozen model confirms that pretrained embeddings must adapt to the recommendation objective. Even if the global geometry of the embedding space changes little (as later shown by the t-SNE visualizations), small adjustments seem to be crucial for aligning the audio representation with behavioral data. Second, pretraining affects convergence speed rather than final accuracy. MusiCNN-tuned starts from a better initial point and reaches good performance faster, whereas the randomly initialized model shows inferior performance at the beginning of training, but ultimately exceeds the performance of the fine-tuned model.

5.3.2 Embedding Space Visualization

To better understand what each model learns internally, we visualize the item embedding spaces of the T4Rec models before and after training using t-SNE [65] projections of 30.000 randomly sampled songs, coloring each point according to its artist. To maintain clarity and focus on the most prominent patterns, we display only the songs belonging to the 20 most frequent artists in the dataset. The resulting plots (Figures 5.3.2, 5.3.3 and 5.3.4) illustrate how different initialization strategies structure musical content and how this structure evolves through training. The legend below each grid of plots shows the mapping between the 20 most frequent artists and the colors used in the plots.

Before vs. after training

Across all pretrained models, we can observe that the overall geometry of the embedding spaces remains remarkably stable after training. This indicates that the Transformer primarily learns to model behavioral sequences on top of the existing representations rather than reshaping them. Nevertheless, small local compactions can be observed, with some artists' clusters becoming slightly denser, showing that while the global topology is preserved, the model performs fine-grained adjustments that are invisible at the scale of t-SNE but important for performance. This observation is consistent with the convergence results of Section 5.3.1: the embeddings are updated only modestly, yet this flexibility is crucial—frozen embeddings fail to adapt and perform poorly.

MusiCNN vs. MERT

The MusiCNN embeddings already exhibit a clear artist-based organization before training, while also grouping artists that share similar acoustic characteristics.. For instance, at the top-right corner of Figure 5.3.3c,

we can observe that clusters of *Metallica*, *Iron Maiden*, *Motörhead* and *In Flames*, all belonging to related genres, appear relatively close in the embedding space. After training, these clusters become slightly tighter but retain their structure, suggesting that the model reinforces existing patterns rather than learning fundamentally new ones. This behavior is consistent with MusiCNN's supervised pretraining on tagging tasks, which encourages the network to form coarse categorical boundaries reflecting genre- or timbre-related tags rather than continuous notions of musical similarity.

In contrast, MERT produces a more diffuse manifold with greater overlap between artists. Although we can still witness artist clusters, the overall geometry is smoother and more cross-artist, which stems from MERT's self-supervised pretraining that learns to represent fine-grained musical similarity across timbral and harmonic dimensions instead of predefined label categories. These fundamental differences could explain the quantitative results: MusiCNN's discrete, tag-oriented representation can separate broad musical styles but struggles with track-level discrimination required by T4Rec, while MERT's continuous, similarity-based space adapts more readily to sequence modeling yet remains less effective than the randomly initialized baseline, which is optimized directly for user-behavioral patterns.

Effect of Fine-Tuning

The MusiCNN-tuned embeddings, obtained after a shallow contrastive fine-tuning stage prior to training the recommendation model, show a similar large-scale organization to the base MusiCNN representations, but the resulting model achieves clearly higher performance in the T4Rec experiments. This contrast suggests that the advantage of fine-tuning does not stem from visibly altering the global structure of the embedding space but from improving its compatibility with the downstream optimization process. After training, the overall geometry remains stable, indicating that the representational benefit arises mainly from the initialization itself rather than from major reshaping during sequence modeling.

Artist Model

The Artist-based embeddings exhibit perhaps the most discrete clusters among the pretrained models, which shouldn't come as a surprise since its training objective directly encourages artist-level similarity. Once again, after training, the artist clusters change only slightly. This categorical rigidity could interpret their poor performance in T4Rec—while useful for identifying artist identity, they cannot distinguish between tracks by the same artist.

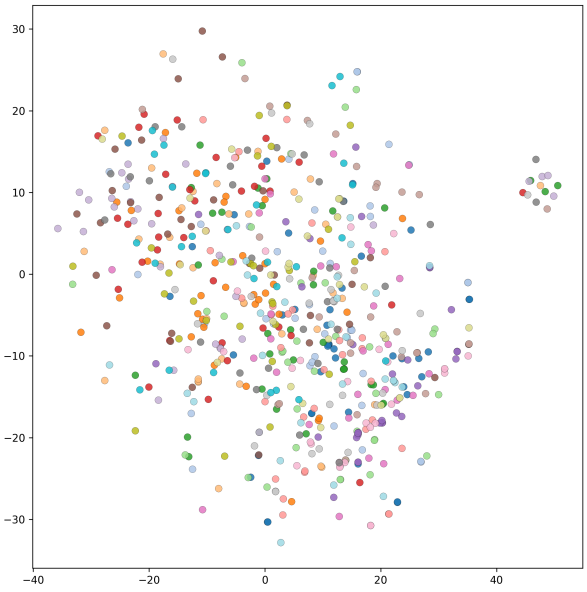


Figure 5.3.2: t-SNE visualization of the embedding space of the randomly initialized model after training.

Random Initialization

For the random initialization, we visualize only the post-training embedding space, since the initial item embeddings contain no meaningful structure. As expected, the resulting t-SNE projection shows a diffuse and unorganized distribution with no clear artist-level clustering or discernible patterns. Despite this apparent lack of structure, the random model achieves the best overall performance among all T4Rec variants. This observation highlights a crucial distinction between human-interpretable structure and task-specific utility: although the random embeddings do not exhibit semantic organization when visualized, they are fully optimized to represent behavioral co-occurrence patterns captured during training. In other words, the transformer learns an embedding geometry that is shaped purely by listening behavior rather than by acoustic similarity, allowing it to capture the fine-grained transitions between tracks that characterize user sessions. This explains why a model trained from scratch on behavioral data can outperform those initialized with pretrained audio embeddings, whose structure, while musically meaningful, constrains the optimization process and may not align with user-interaction dynamics.

Overall, the visualizations reveal that pretrained embeddings carry strong inductive biases—MusiCNN toward acoustic style, MERT toward semantic breadth, Artist toward discrete identity—while the random baseline remains fully behavior-driven. Training fine-tunes these spaces only minimally, implying that most representational adaptation occurs within the Transformer layers rather than in the item embeddings themselves. Interestingly, the relative ability of the pretrained models to separate artists in the embedding space—approximately following the order Artist > MusiCNN > MERT—broadly reflects their performance ranking in the RTA experiments, while appearing inverted compared to the results in T4Rec. This observation suggests that the capacity to encode high-level categorical distinctions, such as artist identity, may be beneficial in retrieval-style objectives that reward semantic or stylistic coherence, but less so in next-item prediction tasks that rely on fine-grained behavioral discrimination.

5.3.3 Ablation Study

The preceding analysis revealed a pattern that could relate high-level category discrimination with the model's performance on the downstream task. However, it would be premature to infer a causal relationship between these properties, as the two evaluated frameworks differ in multiple other aspects beyond their task formulation. To better understand which of these factors might contribute to the divergent performance trends observed between T4Rec and RTA, we conduct an ablation study that systematically adjusts specific components of the T4Rec setup to match those of RTA.

Data Preprocessing

In the first experiment, we replicated RTA's preprocessing step that removes duplicated songs from each session before training. As described in section 4.3, this modification eliminates repeated song occurrences within the same session, by keeping only the first interaction with each song. After applying this step to T4Rec, we ran two models based on this setup: a MusiCNN-initialized model and a randomly initialized baseline with the same embedding dimension ($d = 200$), both of them following the T4Rec hyperparameters shown in Table 5.1. The corresponding test set metrics are presented in Table 5.4.

As seen in the results, both the Random and MusiCNN-initialized models experience a noticeable drop across all evaluation metrics when deduplication is applied (0.2333 vs. 0.2629 NDCG@20 for Random, 0.1537 vs. 0.2358 NDCG@20 for MusiCNN), which agrees with the overall lower metrics observed in Table 5.3. This decreased performance can be attributed to the loss of behavioral redundancy in the input data. In T4Rec, repeated songs within a session act as strong contextual cues that reinforce user preferences and simplify the prediction of subsequent items. Removing these duplicates shortens the sequences and weakens the temporal structure on which the model relies, leading to less predictable next-item transitions. Nevertheless, despite this drop in performance, the relative ranking between the two models remains unchanged, with the random model still outperforming the pretrained one. This indicates that the preprocessing differences between the two frameworks do not account for the opposing performance patterns observed earlier.

Table 5.4: Performance comparison of T4Rec models following RTA's data preprocessing, which applies an additional deduplication step within each session.

Initialization	d	NDCG@10	MAP@10	Recall@10	NDCG@20	MAP@20	Recall@20
Random	200	0.2214	0.1927	0.3112	0.2333	0.1959	0.3583
MusiCNN	200	0.1435	0.1227	0.2093	0.1537	0.1254	0.2498

Training Hyperparameters

In the second ablation, we aim to isolate the effect of the architectural and optimization choices of each framework. To this end, we trained the T4Rec models using RTA's hyperparameters (rightmost column of Table 5.1), while keeping the original T4Rec data setup, as well as its training and evaluation objectives. The resulting T4Rec models have increased capacity, and follow a substantially different optimization plan (different optimizer, learning rate scheduler and initial learning rate). In Table 5.5 we present the test set results for these T4Rec models.

From these results, we can observe a clear shift in model ranking, as all three pretrained models now surpass their corresponding randomly initialized baselines. More specifically, the Random models exhibit a performance drop, compared to the ones from section 5.2.1, while MusiCNN, MusiCNN-tuned and the Artist model show notable improvements under this configuration. In contrast, MERT is the only pretrained model that does not benefit from this setting. Interestingly, in additional experiments where we further isolated either the architectural or the optimization components individually, most models performed poorly, suggesting that the observed gains arise from the combination of the model architecture with the optimization plan, rather than from either factor alone.

Based on the previous observations, we can assume that architectural and optimization choices considerably influence the ability of T4Rec to exploit pretrained audio priors and also to learn effective item embeddings from scratch. More importantly, the observed model ranking aligns with that observed in RTA, suggesting that the hyperparameter configuration likely contributes to the contrasting performance patterns between the two frameworks.

Table 5.5: Performance comparison of T4Rec models trained with the RTA hyperparameters.

Initialization	d	NDCG@10	MAP@10	Recall@10	NDCG@20	MAP@20	Recall@20
Random	200	0.2449	0.2056	0.3658	0.2579	0.2090	0.4167
MusiCNN	200	0.2511	0.2103	0.3769	0.2637	0.2137	0.4263
MusiCNN-tuned	200	**0.2657**	**0.2274**	**0.3844**	**0.2781**	**0.2307**	**0.4333**
Random	768	0.2458	0.2067	0.3672	0.2587	0.2101	0.4179
MERT	768	0.2470	0.2069	0.3712	0.2600	0.2103	0.4220
Random	1280	0.2008	0.1607	0.3237	0.2140	0.1642	0.3756
Artist model	1280	0.2601	0.2205	0.3830	0.2729	0.2239	0.4333

5.4 Discussion

Finally, in this section, we discuss the framework-level differences between Transformers4Rec and RTA, analyzing how their distinct task formulations and learning objectives account for the opposite performance trends observed across the two setups. Together, these analyses provide a more comprehensive view of the interaction between pretrained audio representations, model architecture, and training objective, helping us understand not only which configurations work best but also why they behave as they do.

Task Differences

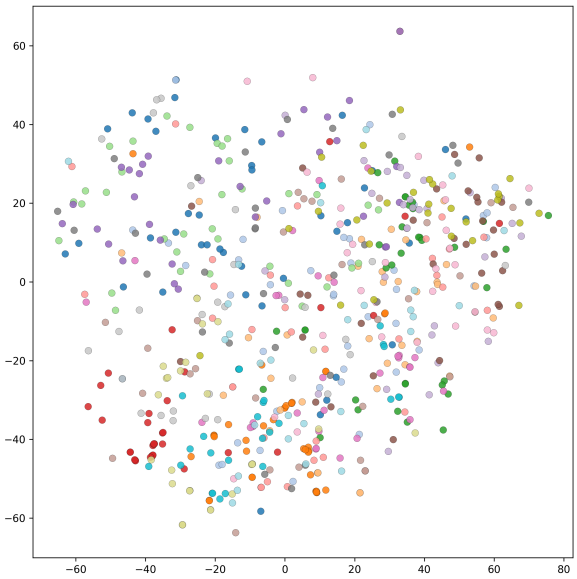
Despite their evidently similar architecture, the task of next-item prediction as defined within the Transformers4Rec [100] and RTA [7] encloses important differences. In the case of Transformers4Rec, given a playlist

of arbitrary length, the objective is to fill in the missing song of an otherwise complete playlist, while RTA defines as APC the significantly more difficult task of finding suitable successors to a playlist of sufficient length, in an iterative manner.

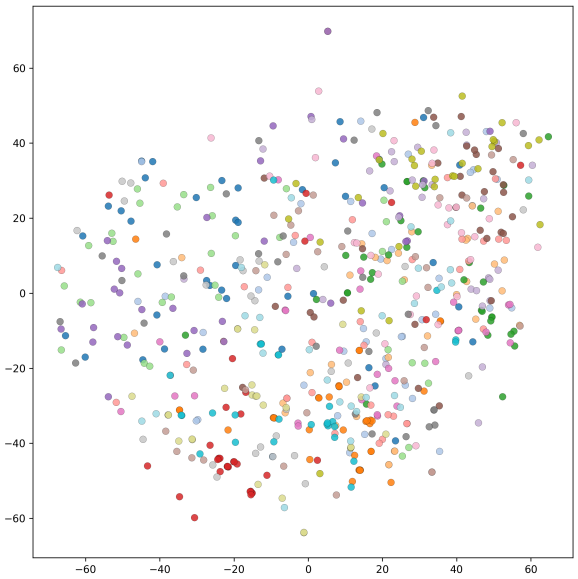
From the Tables 5.2, 5.3, one can observe that in spite of the approach (random, metadata-based or audio-based) used for initializing the trained embeddings, T4Rec yielded higher performance on all examined metrics. Taking into account the aforementioned task differences, the additional filtering steps undertaken in the RTA framework (such as duplicate removal), which leave a more challenging set of initial playlist seeds, as well as the fact that for both frameworks we stuck as much as possible to the originally proposed hyperparameters, we attribute the performance differential in the harder nature of the sequence continuation task as defined through [7].

Cross-Task Model Performance

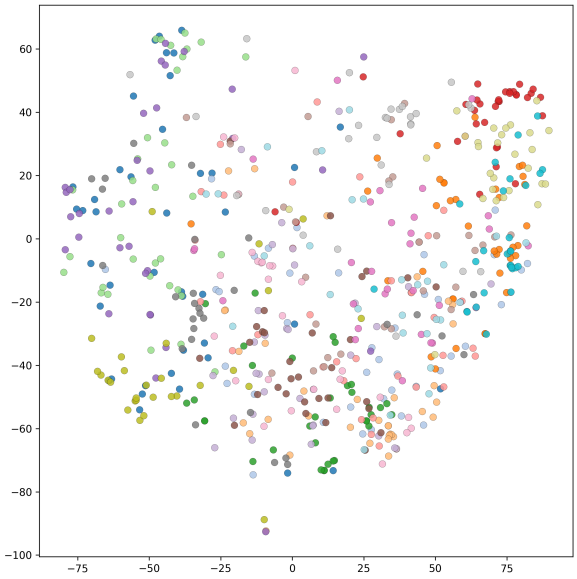
The different task formulations also cause a distinct effect on the performance of each embedding initialization method. MERT [63], for instance, is the pre-trained model that achieves the closest performance to random embedding initialization in the case of Transformers4Rec, while it displays a performance drop in the more favorable (for audio embeddings) case of RTA. Conversely, the custom model trained to bring audio excerpts from the same artist close to each other gains the most over random embedding initialization in the case of RTA, but is the worst-performing pre-trained model among the ones examined under the Transformers4Rec framework. Finally, the only behavior that consistently yields a positive effect on the embedding performance across both frameworks is tuning the pre-trained embeddings to the task at hand, through a simple contrastive procedure [82], as exhibited through the case of MusiCNN [80]; however, this procedure already introduces information about the downstream task-at-hand for embedding acquisition.



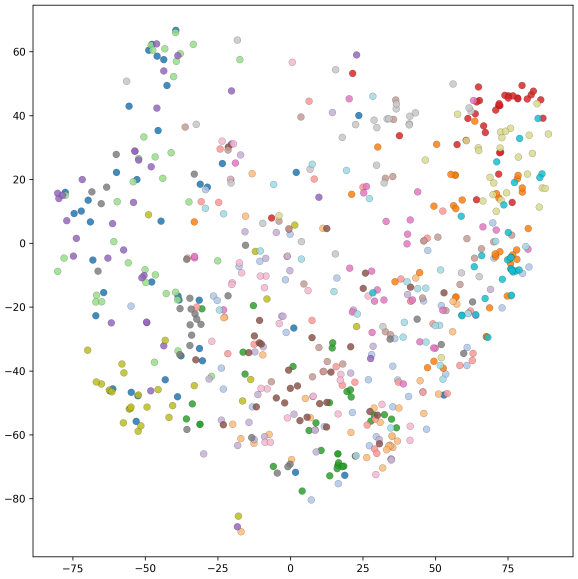
(a) MERT before training



(b) MERT after training

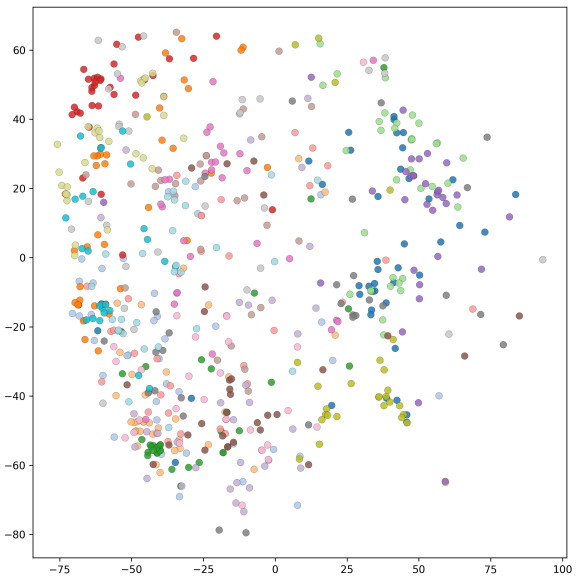


(c) MusiCNN before training

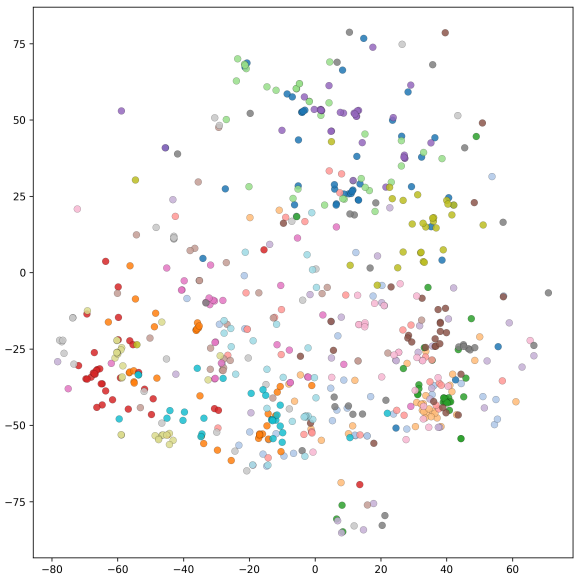


(d) MusiCNN after training

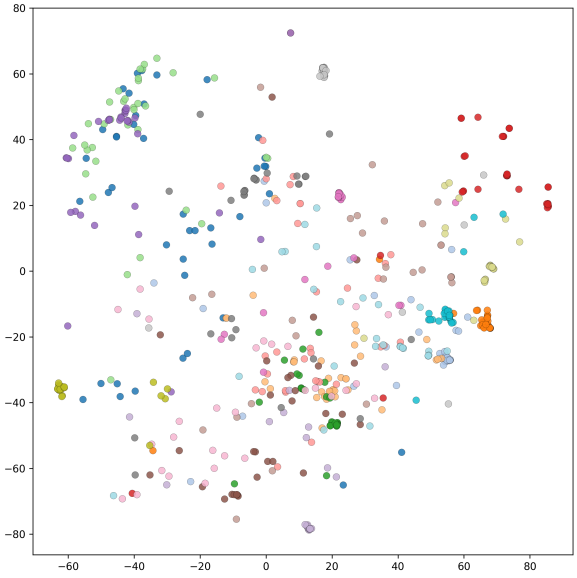
Figure 5.3.3: t-SNE visualizations of the MERT and MusiCNN embedding spaces before and after training.



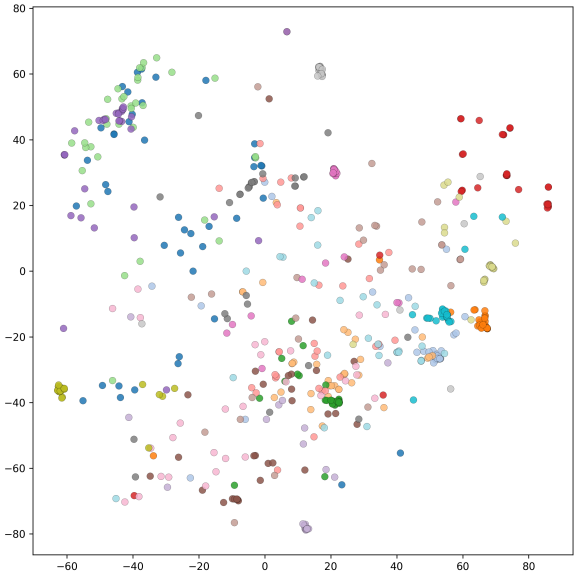
(a) MusiCNN-tuned before training



(b) MusiCNN-tuned after training



(c) Artist embeddings before training



(d) Artist embeddings after training

Figure 5.3.4: t-SNE visualizations of the MusiCNN-tuned and Artist embedding spaces before and after training.

Chapter 6

Conclusion

This work investigates whether pretrained audio embeddings can enhance the performance of a session-based recommendation model. Our main motivation lies in the fact that, despite the competitive performance of pretrained models in various MIR tasks [14], [63], [80], [115], their use in recommender systems has remained underexplored. Especially in a session-based setting, where the available context is limited, the idea of incorporating content embeddings that successfully encode musical properties appears promising. To test our approach, we extracted song embeddings using three pretrained models, namely MusiCNN [80], MERT [63], and a custom artist-based model, in order to measure the impact of diverse pretraining methods on recommendation performance. Using listening sessions derived from the Music4All dataset [91], we trained Transformer-based recommendation models, based on two previous works from the literature: Transformers4Rec [100] (focusing on the simpler case of next song prediction) and Represent-Then-Aggregate (RTA) [7] (tackling the harder task of playlist continuation), which we adapted to our session-based setting. Rather than learning item representations from scratch based only on behavioral data, we examined ways to incorporate the audio embeddings into this pipeline. Specifically, we used them as a trainable initialization of the item embeddings, much like a content-informed starting point. In additional experiments, we first fine-tuned the audio representations with a shallow model as a simple domain adaptation, while we also explored a late-fusion strategy, where pretrained embeddings were integrated after the Transformer layers using a session-level concatenation.

Our experimentation on both frameworks yielded complementary insights. In the case of Transformers4Rec, which formulates recommendation as a missing-song prediction task, none of the examined audio-based embedding initialization strategies seemed to surpass random initialization. This suggests that, despite the evident advantage of the audio-informed models during the early training epochs, the model can learn the required information solely from scratch. On the other hand, despite the (in-line with the literature [49], [55]) superior performance of metadata-based factorization of the user-item co-occurence matrix for embedding initialization, audio embeddings can provide helpful information for the session continuation task under the harder setting of the RTA framework (no song duplicates, limited context). In both cases, our experiments highlighted a slight but consistent positive effect of applying behavior-based fine-tuning to the audio embeddings, as a form of domain adaptation. In an attempt to better understand the contradictory performance of the pretrained models between the two frameworks, we first visualized the corresponding embedding spaces, and observed a potential link between each model's ability to form musically informative audio clusters and its performance in the recommendation setting. We then conducted an ablation study, in which we isolated the effect of several factors that differentiate the two frameworks and revealed the importance of hyperparameter configuration in the experimental results. That being said, we argue that that the usefulness of pretrained audio representations in recommendation applications could depend on the alignment of the pretraining objective with the downstream recommendation task, while appropriate architectural and optimization choices are essential for realizing their full potential. Finally, no clear conclusion could be drawn about the relative suitability of the various (metadata-based contrastive, self-supervised and tag-based supervised) training paradigms examined.

Overall, the key contributions of our work can be summarized as follows:

- We systematically evaluate the performance of pretrained audio representations under two existing recommendation frameworks: Transformers4Rec [100] and Represent-Then-Aggregate (RTA) [7].
- We extract audio representations from three pretrained models: MERT [63], MusiCNN [80] and a custom model that focuses on artist identity. Each model follows a different pretraining strategy (masked acoustic modeling, supervised tagging, contrastive learning, respectively).
- We experiment with multiple methods of integrating the embeddings to the recommendation pipeline, including a domain adaptation technique that fine-tunes pretrained embeddings on behavior data using a simple contrastive objective.
- Our experiments showed a significantly different model ranking depending on the framework. To explain these contradicting results we used an embedding visualization technique [65] and performed an ablation study focusing on specific differences of the two frameworks.
- Our findings highlight the impact of the pretraining strategy and the hyperparameter choice on how each audio model transfers to the recommendation domain.

Future work

To further enhance and extend our work, we outline several directions we plan to pursue as part of future research:

- Exploring a broader range of pretrained models could reveal additional insights into which representation learning approaches are more suitable for recommendation purposes. Beyond audio-based models, an interesting direction would be to extract song representations using Large Language Models (LLMs) on song metadata or lyrics, motivated by [82].
- Developing more sophisticated adaptation strategies for the audio embeddings could further improve performance. Since our results indicated that fine-tuning is, in general, beneficial compared to using embeddings out-of-the-box, experimenting with alternative tuning objectives and carefully adjusting critical hyperparameters (e.g., learning rate) may enable better exploitation of the pretrained representations.
- Fine-tuning the pretrained models themselves –rather than solely the extracted embeddings– could also be investigated, of course keeping in mind the tradeoff between the computational cost of such a process and the potential performance gains.
- Finally, assessing the benefits of content integration across different scenarios is another promising direction. Even if incorporating audio embeddings into collaborative methods does not consistently improve overall performance, such approaches might still provide advantages in specific settings, such as for less popular tracks, emerging artists, or shorter listening sessions.

Bibliography

[1] Alonso-Jiménez, P., Favory, X., Foroughmand, H., *et al.*, "Pre-training strategies using contrastive learning and playlist information for music classification and similarity," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2023)*, Rhodes Island, Greece, 2023.

[2] Ba, J. L., Kiros, J. R., and Hinton, G. E., "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016. [Online]. Available: https://arxiv.org/abs/1607.06450.

[3] Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., and Auli, M., "Data2vec: A general framework for self-supervised learning in speech, vision and language," in *Proceedings of the 39th International Conference on Machine Learning (ICML '22)*, Baltimore, Maryland, USA, 2022.

[4] Baevski, A., Zhou, Y., Mohamed, A., and Auli, M., "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2020.

[5] Bao, K., Zhang, J., Zhang, Y., Wang, W., Feng, F., and He, X., "TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation," in *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys '23)*, Singapore, Singapore, 2023.

[6] Beltagy, I., Peters, M. E., and Cohan, A., "Longformer: The Long-Document Transformer," *arXiv:2004.05150*, 2020. [Online]. Available: https://arxiv.org/abs/2004.05150.

[7] Bendada, W., Salha-Galvan, G., Bouabça, T., and Cazenave, T., "A Scalable Framework for Automatic Playlist Continuation on Music Streaming Services," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*, Taipei, Taiwan, 2023.

[8] Bengio, Y., Louradour, J., Collobert, R., and Weston, J., "Curriculum learning," in *Proceedings of the 26th International Conference on Machine Learning (ICML '09)*, Montreal, Quebec, Canada, 2009.

[9] Bennett, J. and Lanning, S., "The Netflix Prize," in *Proceedings of KDD Cup and Workshop 2007 (KDDCup'07)*, San Jose, California, USA, 2007.

[10] Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P., "The Million Song Dataset," in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, Miami, Florida, USA, 2011.

[11] Bogdanov, D., Won, M., Tovstogan, P., Porter, A., and Serra, X., "The MTG-Jamendo Dataset for Automatic Music Tagging," in *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML '19)*, Long Beach, California, USA, 2019.

[12] Bre, F., Gimenez, J. M., and Fachinotti, V. D., "Prediction of wind pressure coefficients on building surfaces using artificial neural networks," *Energy and Buildings*, vol. 158, pp. 1429–1441, 2018.

[13] Brown, J. C., "Calculation of a constant Q spectral transform," *The Journal of the Acoustical Society of America*, vol. 89, pp. 425–434, 1991.

[14] Castellon, R., Donahue, C., and Liang, P., "Codified audio language modeling learns useful representations for music information retrieval," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021.

[15] Chen, C.-W., Lamere, P., Schedl, M., and Zamani, H., "Recsys challenge 2018: Automatic music playlist continuation," in *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*, Vancouver, Canada, 2018.

[16] Chen, R., Shen, W., Srinivasamurthy, A., and Chordia, P., "Chord Recognition Using Duration-explicit Hidden Markov Models.," in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, 2012.

[17] Chen, S., Wang, C., Chen, Z., *et al.*, "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, pp. 1505–1518, 2022.

[18] Chen, T. and Guestrin, C., "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, San Francisco, California, USA, 2016.

[19] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G., "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning (ICML '20)*, Vienna, Austria, 2020.

[20] Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y., "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Doha, Qatar, 2014.

[21] Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D., "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators," in *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, Online, 2020.

[22] Cuesta, H., McFee, B., and Gómez, E., "Multiple f0 estimation in vocal ensembles using convolutional neural networks," in *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2020.

[23] Dabbura, I. "Gradient Descent Algorithm and Its Variants," medium.com. (2017), [Online]. Available: https://medium.com/data-science/gradient-descent-algorithm-and-its-variants-10f652806a3 (accessed 10/08/2025).

[24] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., and Salakhutdinov, R., "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, Florence, Italy, 2019.

[25] Davis, S. and Mermelstein, P., "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, pp. 357–366, 1980.

[26] Défossez, A., Copet, J., Synnaeve, G., and Adi, Y., "High Fidelity Neural Audio Compression," *arXiv preprint arXiv:2210.13438*, 2022. [Online]. Available: https://arxiv.org/abs/2210.13438.

[27] Deldjoo, Y., Schedl, M., and Knees, P., "Content-driven music recommendation: Evolution, state of the art, and challenges," *Computer Science Review*, vol. 51, p. 100 618, 2024.

[28] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, Minneapolis, Minnesota, USA, 2019.

[29] Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I., "Jukebox: A Generative Model for Music," *arXiv preprint arXiv:2005.00341*, 2020. [Online]. Available: https://arxiv.org/abs/2005.00341.

[30] Elijah, K. "The Basics of Neural Networks (Neural Network Series) - Part 1," medium.com. (2022), [Online]. Available: https://medium.com/data-science/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b (accessed 09/29/2025).

[31] Fayek, H. "Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between," haythamfayek.com. (2016), [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html (accessed 10/08/2025).

[32] Ferraro, A., Favory, X., Drossos, K., Kim, Y., and Bogdanov, D., "Enriched music representations with multiple cross-modal contrastive learning," *IEEE Signal Processing Letters*, vol. 28, pp. 733–737, 2021.

[33] Ferraro, A., Kim, Y., Lee, S., *et al.*, "Melon Playlist Dataset: A Public Dataset for Audio-Based Playlist Generation and Music Tagging," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2021)*, Toronto, Ontario, Canada, 2021.

[34] Ferraro, A., Serra, X., and Bauer, C., "What is fair? Exploring the artists' perspective on the fairness of music streaming platforms," in *18th IFIP International Conference on Human-Computer Interaction (INTERACT 2021)*, Bari, Italy, 2021.

[35] Fuentes, B., Liutkus, A., Badeau, R., and Richard, G., "Probabilistic model for main melody extraction using constant-Q transform," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, Kyoto, Japan, 2012.

[36] Funk, S. "Netflix Update: Try This at Home," sifter.org. (2006), [Online]. Available: https://sifter.org/simon/journal/20061211.html (accessed 10/08/2025).

[37] Garoufis, C., Zlatintsi, A., and Maragos, P., *Multi-Source Contrastive Learning from Musical Audio*, Stockholm, Sweden, 2023.

[38] Garoufis, C., Zlatintsi, A., and Maragos, P., "Pre-Training Music Classification Models via Music Source Separation," in *Proceedings of the 32nd European Signal Processing Conference (EUSIPCO 2024)*, Lyon, France, 2024.

[39] Gemmeke, J. F., Ellis, D. P., Freedman, D., *et al.*, "Audio set: An ontology and human-labeled dataset for audio events," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017)*, New Orleans, Louisiana, USA, 2017.

[40] Girshick, R., Donahue, J., Darrell, T., and Malik, J., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, Columbus, Ohio, USA, 2014.

[41] Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J., "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, Pittsburgh, Pennsylvania, USA, 2006.

[42] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16)*, Las Vegas, Nevada, USA, 2016.

[43] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S., "Neural Collaborative Filtering," in *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*, Perth, Australia, 2017.

[44] Hidasi, B. and Karatzoglou, A., "Recurrent Neural Networks with Top-k Gains for Session-based Recommendations," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*, Torino, Italy, 2018.

[45] Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D., "Session-based Recommendations with Recurrent Neural Networks," in *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, 2016.

[46] Hochreiter, S. and Schmidhuber, J., "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–1780, 1997.

[47] Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A., "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[48] Hu, Y., Koren, Y., and Volinsky, C., "Collaborative Filtering for Implicit Feedback Datasets," in *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM '08)*, Pisa, Italy, 2008.

[49] Huang, Q., Jansen, A., Zhang, L., Ellis, D. P. W., Saurous, R. A., and Anderson, J., "Large-Scale Weakly-Supervised Content Embeddings for Music Recommendation and Tagging," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, Online, 2020.

[50] Ioffe, S. and Szegedy, C., "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*, Lille, France, 2015.

[51] İşbitirici, A., Giarré, L., Xu, W., and Falcone, P., "LSTM-Based Virtual Load Sensor for Heavy-Duty Vehicles," *Sensors*, vol. 24, p. 226, 2024.

[52] Jiang, N., Grosche, P., Konz, V., and Müller, M., "Analyzing chroma feature types for automated chord recognition," in *Audio Engineering Society Conference: 42nd International Conference: Semantic Audio*, Ilmenau, Germany, 2011.

[53] Kang, W.-C. and McAuley, J., "Self-Attentive Sequential Recommendation," in *2018 IEEE International Conference on Data Mining (ICDM '18)*, Singapore, Singapore, 2018.

[54] Koren, Y., Bell, R., and Volinsky, C., "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, pp. 30–37, 2009.

[55] Korzeniowski, F., Nieto, O., McCallum, M., Won, M., Oramas, S., and Schmidt, E., "Mood Classification using Listening Data," in *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2020.

[56] Kowald, D., Schedl, M., and Lex, E., "The Unfairness of Popularity Bias in Music Recommendation: A Reproducibility Study," in *Advances in Information Retrieval: 42nd European Conference on IR Research (ECIR 2020)*, Lisbon, Portugal, 2020.

[57] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 26th International Conference on Neural Information Processing Systems (NeurIPS)*, Lake Tahoe, Nevada, USA, 2012.

[58] Law, E., West, K., Mandel, M. I., Bay, M., and Downie, J. S., "Evaluation of algorithms using games: The case of music tagging.," in *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, Kobe, Japan, 2009.

[59] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[60] Lee, J., Bryan, N. J., Salamon, J., Jin, Z., and Nam, J., "Disentangled multidimensional metric learning for music similarity," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, Online, 2020.

[61] Li, J., Wang, M., Li, J., *et al.*, "Text is all you need: Learning language representations for sequential recommendation," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*, Long Beach, California, USA, 2023.

[62] Li, Y., Yuan, R., Zhang, G., *et al.*, "MAP-Music2Vec: A simple and effective baseline for self-supervised music audio representation learning," in *Extended Abstracts for the Late-Breaking Demo Session of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, Bengaluru, India, 2022.

[63] Li, Y., Yuan, R., Zhang, G., *et al.*, "MERT: Acoustic Music Understanding Model with Large-Scale Self-supervised Training," in *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, Vienna, Austria, 2024.

[64] Long, J., Shelhamer, E., and Darrell, T., "Fully convolutional networks for semantic segmentation," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, Boston, Massachusetts, USA, 2015.

[65] Maaten, L. van der and Hinton, G., "Visualizing data using t-sne," *Journal of Machine Learning Research (JLMR)*, vol. 9, pp. 2579–2605, 2008.

[66] Manco, I., Benetos, E., Quinton, E., and Fazekas, G., "Contrastive Audio-Language Learning for Music," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, Bengaluru, India, 2022.

[67] McFee, B., Barrington, L., and Lanckriet, G., "Learning content similarity for music recommendation," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 20, pp. 2207–2218, 2012.

[68] McFee, B. and Lanckriet, G., "Metric learning to rank," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, Haifa, Israel, 2010.

[69] Meehan, G. and Pauwels, J., "Evaluating Contrastive Methodologies for Music Representation Learning Using Playlist Data," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2025)*, Hyderabad, India, 2025.

[70] Mikolov, T., Chen, K., Corrado, G., and Dean, J., "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013. [Online]. Available: https://arxiv.org/abs/1301.3781.

[71] Moreira, G. d. S. P., Rabhi, S., Ak, R., Kabir, M. Y., and Oldridge, E., "Transformers with multimodal features and post-fusion context for e-commerce session-based recommendation," in *Proceedings of the Fifth SIGIR eCommerce Workshop 2021 (SIGIR eCom '21)*, Online, 2021.

[72] Moscati, M., Parada-Cabaleiro, E., Deldjoo, Y., Zangerle, E., and Schedl, M., "Music4All-Onion – A Large-Scale Multi-faceted Content-Centric Music Recommendation Dataset," in *Proceedings of the*

31st ACM International Conference on Information & Knowledge Management (CIKM '22), Atlanta, GA, USA, 2022.

[73] Oord, A. van den, Vinyals, O., and Kavukcuoglu, K., "Neural discrete representation learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, Long Beach, California, USA, 2017.

[74] Oord, A. v. d., Li, Y., and Vinyals, O., "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2019. [Online]. Available: https://arxiv.org/abs/1807.03748.

[75] Peeters, G., "Chroma-based estimation of musical key from audio-signal analysis.," in *Proceedings of the 7th International Society for Music Information Retrieval Conference (ISMIR)*, Victoria, Canada, 2006.

[76] Pepino, L., Riera, P., and Ferrer, L., "EnCodecMAE: Leveraging neural codecs for universal audio representation learning," *arXiv preprint arXiv:2309.07391*, 2023. [Online]. Available: https://arxiv.org/abs/2309.07391.

[77] Petrov, A. and Macdonald, C., "A Systematic Review and Replicability Study of BERT4Rec for Sequential Recommendation," in *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys '22)*, Seattle, Washington, USA, 2022.

[78] Phi, M. "Illustrated Guide to LSTM's and GRU's: A step by step explanation," medium.com. (2018), [Online]. Available: https://medium.com/data-science/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21 (accessed 10/08/2025).

[79] Pons, J., Lidy, T., and Serra, X., "Experimenting with musically motivated convolutional neural networks," in *14th International Workshop on Content-Based Multimedia Indexing (CBMI '16)*, Bucharest, Romania, 2016.

[80] Pons, J. and Serra, X., "musicnn: pre-trained convolutional neural networks for music audio tagging," in *Late-breaking Demo Session of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, Delft, The Netherlands, 2019.

[81] Prijono, B. "Student Notes: Convolutional Neural Networks (CNN) Introduction," indoml.wordpress.com. (2018), [Online]. Available: https://indoml.wordpress.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/ (accessed 10/08/2025).

[82] Qu, Z., Xie, R., Xiao, C., Kang, Z., and Sun, X., "The Elephant in the Room: Rethinking the Usage of Pre-trained Language Model in Sequential Recommendation," in *Proceedings of the 18th ACM Conference on Recommender Systems (RecSys '24)*, Bari, Italy, 2024.

[83] Radečić, D. "Softmax Activation Function Explained," towardsdatascience.com. (2020), [Online]. Available: https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60/ (accessed 10/08/2025).

[84] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., *et al.* "Improving language understanding by generative pre-training," OpenAI Blog. (2018), [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (accessed 10/08/2025).

[85] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., *et al.* "Language models are unsupervised multitask learners," OpenAI Blog. (2019), [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf (accessed 10/08/2025).

[86] Reimers, N. and Gurevych, I., "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Hong Kong, China, 2019.

[87] Ren, X., Wei, W., Xia, L., *et al.*, "Representation Learning with Large Language Models for Recommendation," in *Proceedings of the ACM Web Conference 2024 (WWW '24)*, Singapore, Singapore, 2024.

[88] Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L., "BPR: Bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*, Montreal, Quebec, Canada, 2009.

[89] Rosenblatt, F., "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological Review*, vol. 65, pp. 386–408, 1958.

[90] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[91] Santana, I. A. P., Pinhelli, F., Donini, J., *et al.*, "Music4all: A new music database and its applications," in *27th International Conference on Systems, Signals and Image Processing (IWSSIP 2020)*, Niterói, Brazil, 2020.

[92] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J., "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*, Hong Kong, Hong Kong, 2001.

[93] Schedl, M., "The LFM-1b Dataset for Music Retrieval and Recommendation," in *Proceedings of the 2016 ACM International Conference on Multimedia Retrieval (ICMR '16)*, New York, New York, USA, 2016.

[94] Schedl, M., Brandl, S., Lesota, O., Parada-Cabaleiro, E., Penz, D., and Rekabsaz, N., "LFM-2b: A Dataset of Enriched Music Listening Events for Recommender Systems Research and Fairness Analysis," in *Proceedings of the 2022 Conference on Human Information Interaction and Retrieval (CHIIR '22)*, Regensburg, Germany, 2022.

[95] Schedl, M., Zamani, H., Chen, C.-W., Deldjoo, Y., and Elahi, M., "Current challenges and visions in music recommender systems research," *International Journal of Multimedia Information Retrieval*, vol. 7, pp. 95–116, 2018.

[96] Schifferer, B., Deotte, C., Puget, J.-F., *et al.*, "Using Deep Learning to Win the Booking.com WSDM WebTour21 Challenge on Sequential Recommendations.," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining Workshop on Web Tourism (WSDM WebTour '21)*, Online, 2021.

[97] Schroff, F., Kalenichenko, D., and Philbin, J., "FaceNet: A Unified Embedding for Face Recognition and Clustering," in *Proceedings of the 2015 IEEE conference on Computer Vision and Pattern Recognition (CVPR '15)*, Boston, Massachusetts, USA, 2015.

[98] Sigtia, S., Boulanger-Lewandowski, N., and Dixon, S., "Audio Chord Recognition with a Hybrid Recurrent Neural Network.," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015.

[99] Soni, P. "ROC-AUC Analysis – A Deep Dive," blog.trainindata.com. (2024), [Online]. Available: https://www.blog.trainindata.com/auc-roc-analysis/ (accessed 10/08/2025).

[100] Souza Pereira Moreira, G. de, Rabhi, S., Lee, J. M., Ak, R., and Oldridge, E., "Transformers4Rec: Bridging the Gap between NLP and Sequential / Session-Based Recommendation," in *Proceedings of the 15th ACM Conference on Recommender Systems (RecSys '21)*, Amsterdam, Netherlands, 2021.

[101] Spijkervet, J. and Burgoyne, J. A., "Contrastive Learning of Musical Representations," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021.

[102] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[103] Sun, F., Liu, J., Wu, J., *et al.*, "BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*, Beijing, China, 2019.

[104] Tamm, Y.-M. and Aljanaki, A., "Comparative Analysis of Pretrained Audio Representations in Music Recommender Systems," in *Proceedings of the 18th ACM Conference on Recommender Systems (RecSys '24)*, Bari, Italy, 2024.

[105] Tan, M. and Le, Q., "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning (ICML '19)*, Long Beach, California, USA, 2019.

[106] Taylor, W. L., ""Cloze Procedure": A New Tool For Measuring Readability," *Journalism Quarterly*, vol. 30, pp. 415–433, 1953.

[107] Tingle, D., Kim, Y. E., and Turnbull, D., "Exploring automatic music annotation with "acoustically-objective" tags," in *Proceedings of the International Conference on Multimedia Information Retrieval (MIR '10)*, Philadelphia, Pennsylvania, USA, 2010.

[108] Tzanetakis, G. and Cook, P., "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, pp. 293–302, 2002.

[109] Vaswani, A., Shazeer, N., Parmar, N., *et al.*, "Attention is All you Need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, Long Beach, California, USA, 2017.

[110] Veit, A., Belongie, S., and Karaletsos, T., "Conditional Similarity Networks," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '17)*, Honolulu, HI, USA, 2017.

[111] Venugopal, P. and T, V., "State-of-Health Estimation of Li-ion Batteries in Electric Vehicle Using IndRNN under Variable Load Condition," *Energies*, vol. 12, p. 4338, 2019.

[112] Volkovs, M., Rai, H., Cheng, Z., Wu, G., Lu, Y., and Sanner, S., "Two-stage model for automatic playlist continuation at scale," in *Proceedings of the ACM Recommender Systems Challenge 2018 (RecSys Challenge '18)*, Vancouver, Canada, 2018.

[113] Wolf, T., Debut, L., Sanh, V., *et al.*, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, Online, 2020.

[114] Won, M., Ferraro, A., Bogdanov, D., and Serra, X., "Evaluation of CNN-based automatic music tagging models," in *Proceedings of the 17th Sound and Music Computing Conference (SMC 2020)*, Torino, Italy, 2020.

[115] Won, M., Hung, Y.-N., and Le, D., "A foundation model for music informatics," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024)*, Seoul, Korea, 2024.

[116] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V., "XLNet: generalized autoregressive pretraining for language understanding," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019.

[117] Yu, X., Hu, Q., Li, H., Du, J., Gao, J., and Sun, L., "Cross-domain recommendation based on latent factor alignment," *Neural Computing and Applications*, vol. 34, pp. 3421–3432, 2022.

[118] Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D., "mixup: Beyond empirical risk minimization," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[119] Zhang, J., Xie, R., Hou, Y., Zhao, X., Lin, L., and Wen, J.-R., "Recommendation as instruction following: A large language model empowered recommendation approach," *ACM Transactions on Information Systems*, vol. 43, pp. 1–37, 2025.

[120] Zlatintsi, A. and Maragos, P., "Comparison of Different Representations Based on Nonlinear Features for Music Genre Classification," in *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO 2014)*, Lisbon, Portugal, 2014.