ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μηχανικής Μάθησης

Αυτόματη Σύσταση Ταινιών με Βάση την Περίληψή τους

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Εμμανουήλ Βλάσση

Επιβλέπων: Γεώργιος Στάμου
Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2024

Εθνικο Μετσοβιο Πολυτεχνειο
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μηχανικής Μάθησης

Αυτόματη Σύσταση Ταινιών με Βάση την Περίληψή τους

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Εμμανουήλ Βλάσση

Επιβλέπων: Γεώργιος Στάμου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26η Μαρτίου 2024.

.....	.....	.....
Γεώργιος Στάμου Καθηγητής Ε.Μ.Π.	Στέφανος Κόλλιας Καθηγητής Ε.Μ.Π.	Αθανάσιος Βουλόδημος Επ. Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2024

.....
Εμμανουήλ Βλάσσης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Το αντικείμενο της σύστασης ταινιών έχει συγκεντρώσει πολλή προσοχή από την επιστημονική κοινότητα, ιδιαίτερα στη σύγχρονη ψηφιακή εποχή όπου βασιζόμαστε όλο και περισσότερο σε εργαλεία για τη διαδικασία λήψης αποφάσεων. Τα σύγχρονα συστήματα συστάσεων χρησιμοποιούν κυρίως μεθόδους συνεργατικού φιλτραρίσματος, χωρίς να αξιοποιούν την πλούσια πληροφορία που είναι διαθέσιμη στις ταινίες. Η συνεχής βελτίωση των Μεγάλων Μοντέλων Γλώσσας (LLMs) έχει επιτρέψει την πιο εμπεριστατωμένη ανάλυση κειμένου, δίνοντας μας νέες δυνατότητες. Αυτή η διπλωματική εργασία διερευνά τη χρήση συγχρόνων τεχνικών Επεξεργασίας Φυσικής Γλώσσας (NLP) για την εξαγωγή χρήσιμων πληροφοριών από περιλήψεις ταινιών, με στόχο τη βελτίωση των συστημάτων συστάσεων. Για τον λόγο αυτό, εξετάζουμε τις διάφορες μεθόδους στη σύσταση αντικειμένων, και σχεδιάζουμε συστήματα που μπορούν να χρησιμοποιήσουν την πληροφορία που εμπεριέχεται στις συνόψεις. Επιπλέον, κατασκευάσαμε ένα εξειδικευμένο σύνολο δεδομένων, προκειμένου να διευκολυνθεί η διεξαγωγή πειραμάτων που αξιολογούν την επίδραση των περιλήψεων ταινιών στη διαδικασία της σύστασης. Παρατηρήσαμε μια μικρή αλλά σημαντική βελτίωση στην απόδοση τόσο των συστημάτων βασισμένων στο περιεχόμενο όσο και των υβριδικών συστημάτων όταν ενσωματώνονται περιλήψεις. Αυτά τα αποτελέσματα τονίζουν τις προοπτικές που έχουν οι σύγχρονες τεχνικές ανάλυσης κειμένου, ιδιαίτερα σε μοντέλα μάθησης με λίγα δείγματα (few-shot learning), όπου κλιμάκωση των μοντέλων θα μπορούσε να βελτιώσει περαιτέρω τις μετρικές. Η εργασία δημιουργεί μια βάση για μελλοντικές έρευνες στην αξιοποίηση κειμενικής πληροφορίας για τη βελτίωση των συστημάτων σύστασης ταινιών.

Λέξεις-κλειδιά

Συστήματα Συστάσεων, Επεξεργασία Φυσικής Γλώσσας, Συνεργατικό Φιλτράρισμα, Μεγάλα Γλωσσικά Μοντέλα, Μάθηση με Λίγα Δείγματα

Abstract

The domain of movie recommendation has garnered significant attention from the scientific community, especially in the digital age where we increasingly rely on tools in the decision-making processes. Modern recommendation systems predominantly use collaborative filtering methods, which do not utilize the rich textual information available in movies. The development of ever advancing Large Language Models (LLMs) has enabled us to analyze text more comprehensively. This thesis investigates the integration of modern Natural Language Processing (NLP) techniques to extract valuable insights from movie synopses, aiming to improve the capabilities of existing recommendation frameworks. By examining the various recommendation methods, we design systems that can use the nuanced information within synopses. A specialized dataset is constructed to facilitate comprehensive experiments that assess the impact of movie plot information on the recommendation process. The empirical findings indicate a modest yet significant enhancement in the performance of both content-based and hybrid recommendation systems when synopses are incorporated. These results underscore the potential of employing advanced text analysis techniques, particularly in few-shot learning models where further scaling could amplify the improvements we observed. Our work lays a foundational groundwork for future research into the utilization of textual data for refined movie recommendation, making a step forward in the advancement of personalized recommendation systems.

Keywords

Recommender Systems, Natural Language Processing, Collaborative Filtering, Large Language Models, Few-shot Learning

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον κ. Γιώργο Στάμου για την επίβλεψη της παρούσιας διπλωματικής εργασίας, και την ευκαιρία που μου έδωσε να ασχοληθώ με ένα θέμα που με μαγνήτισε από την πρώτη στιγμή. Επίσης, ευχαριστώ πολύ τη Μαρία Λυμπεραίου, τον Γιώργο Φιλανδριανό, και τον Κωνσταντίνο Θωμά, για την καθοδήγηση που μου έδωσαν σε όλη τη διάρκεια εκπόνησης της εργασίας. Η βοήθεια τους ήταν πολύτιμη και μου έδωσε τη δυνατότητα να εντρυφήσω πραγματικά στο θέμα της σύστασης ταινιών και να παραγάγω ένα έργο για το οποίο είμαι περήφανος.

Είμαι βαθιά ευγνώμων στους γονείς μου για τη συνεχή στήριξή τους σε κάθε βήμα και την αμέριστη αγάπη που μου δίνουν. Ευχαριστώ επίσης βαθιά τον αδελφό μου, Γιώργο, για το πάθος που μου έχει μεταφέρει για τους υπολογιστές, και για την ατέρμονη πίστη σε εμένα και τις ικανότητές μου. Ευχαριστώ θερμά τον θείο μου, για την αγάπη που μετέφερε για τα μαθηματικά, η οποία εξακολουθεί να με καθοδηγεί μέχρι και σήμερα.

Κλείνοντας, θέλω να ευχαριστήσω όλους μου τις φίλους για τις υπέροχες στιγμές που μου έχουν χαρίσει. Ιδιαίτερα, ευχαριστώ τον Θοδωρή, τον Πάνο, τον Λαυρέντη, την Τίνα, τον Κρις, και τον Παναγιώτη, με τους οποίους μοιράστηκα το ακαδημαϊκό μου ταξίδι και μου χάρισαν αξέχαστα φοιτητικά χρόνια.

Contents

List of Figures

List of Tables

Chapter 1

Εκτεταμένη Περίληψη στα Ελληνικά

1.1 Εισαγωγή

Στη σύγχρονη ψηφιακή εποχή, συχνά βρισκόμαστε αντιμέτωποι με αυτό που ονομάζεται παράδοξο της επιλογής. Σήμερα έχουμε περισσότερες επιλογές από ποτέ, αλλά αυτή η αφθονία περιπλέκει τη διαδικασία λήψης αποφάσεών μας. Αποτελεσματικά, τα εργαλεία που μας βοηθούν να κάνουμε ενημερωμένες επιλογές γίνονται όλο και πιο σημαντικά στις ζωές μας. Η σύσταση αντικειμένων ως ερευνητικό θέμα έχει συγκεντρώσει πολλή προσοχή από την ακαδημαϊκή κοινότητα, κυρίως λόγω του τεράστιου εμπορικού ενδιαφέροντος. Ίσως το πιο μελετημένο αντικείμενο σύστασης είναι η σύσταση ταινιών, η οποία με την άνοδο του πλατφορμών streaming έχει γίνει κομβική για την εμπειρία των χρηστών.

Τα σύγχρονα συστήματα σύστασης συνήθως βασίζονται σε τεχνικές συνεργατικού φιλτραρίσματος που χρησιμοποιούν δεδομένα από τις αλληλεπιδράσεις χρηστών και αντικειμένων, ή μεταδεδομένα για να κάνουν συστάσεις. Ωστόσο, αυτή η προσέγγιση δεν αξιοποιεί την πληροφορία που εμπεριέχεται σε δεδομένα κειμένου, όπως είναι οι περιλήψεις των ταινιών. Ταυτόχρονα, η ανάπτυξη προηγμένων τεχνικών στην επεξεργασία φυσικής γλώσσας και η δημιουργία Μεγάλων Γλωσσικών Μοντέλων (LLMs) παρουσιάζει μια ενδιαφέρουσα ευκαιρία για τη βελτίωση των συστημάτων σύστασης. Η παρούσα εργασία εξερευνά τις προοπτικές της χρήσης μοντέρνων τεχνικών μηχανικής μάθησης για την εξαγωγή χρήσιμης πληροφορίας από περιλήψεις ταινιών, καθώς και τη χρήση τους σε συστήματα σύστασης.

Για αυτήν την εργασία, θέτουμε δύο ερευνητικά ερωτήματα. Το κύριο ερώτημα εξετάζει το αν οι περιλήψεις ταινιών μπορούν να βελτιώσουν την απόδοση των συστημάτων σύστασης ταινιών. Σε περίπτωση που πράγματι μπορεί να βελτιωθεί η διαδικασία σύστασης, τότε θα πρέπει να εξετάσουμε τους καλύτερους τρόπους αξιοποίησης της επιπλέον πληροφορίας, καθώς και τον βαθμό στον οποίο βελτιώνονται οι μετρικές. Το δεύτερο ερώτημα σχετίζεται με τη δυνατότητα αντικατάστασης των τυπικών μεταδεδομένων από τις περιλήψεις ταινιών στα συστήματα σύστασης. Τα μεταδεδομένα χρησιμοποιούνται συχνά στα μοντέρνα συστήματα συστάσεων, οπότε η εύρεση περιττών δεδομένων θα μπορούσε να βελτιώσει την

αποδοτικότητα του συστήματος. Επίσης, οι παρατηρήσεις μας είναι ιδιαίτερα σημαντικές σε τεχνικές που χρησιμοποιούν ελάχιστα δεδομένα, όπως η μάθηση με λίγα δείγματα.

Για να απαντήσουμε στα παραπάνω ερωτήματα, κάναμε τις ακόλουθες συνεισφορές:

- Δημιουργία ενός εξειδικευμένου συνόλου δεδομένων που συνδυάζει περιλήψεις ταινιών, μεταδεδομένα, και κριτικές χρηστών.
- Ανάπτυξη μοντέλων που ανήκουν σε διαφορετικές κατηγορίες συστημάτων σύστασης.
- Σχεδιασμός και η εκτέλεση πειραμάτων των μοντέλων χρησιμοποιώντας συγκεκριμένες μετρικές.
- Ανάλυση και συζήτηση των αποτελεσμάτων, εστιάζοντας στην αποτελεσματικότητα των περιλήψεων ταινιών στη βελτίωση των συστημάτων σύστασης.

Η εκτεταμένη περίληψη αποτελείται από τις εξής ενότητες:

- Ενότητα 1 - Εισαγωγή: Περιλαμβάνονται τα κίνητρα πίσω από την εργασία, καθώς και τα ερευνητικά ερωτήματα.
- Ενότητα 2 - Θεωρητικό Υπόβαθρο: Εξηγούνται τις θεμελιώδεις έννοιες που σχετίζονται με το πεδίο της σύστασης αντικειμένων.
- Ενότητα 3 - Βιβλιογραφική Ανασκόπηση: Εξετάζονται τις υπάρχει τις σύγχρονες μεθοδολογίες στη σύσταση αντικειμένων, και τα εργαλεία τεχνητής νοημοσύνης που θα χρησιμοποιήσουμε.
- Ενότητα 4 - Μέθοδος: Περιγράφονται το σύνολο δεδομένων που δημιουργήθηκε, και την αρχιτεκτονική των μοντέλων που αναπτύχθηκαν. Εξηγούνται τα πειράματα που πραγματοποιήθηκαν.
- Ενότητα 5 - Αποτελέσματα: Παρουσιάζονται τα αποτελέσματα των πειραμάτων, μαζί με την ανάλυση και την πιθανή ερμηνεία τους.
- Ενότητα 6 - Συμπέρασμα: Συνοψίζονται τα ευρήματα της εργασίας, απαντώνται στα ερωτήματα που τέθηκαν, και δίνονται προτάσεις για μελλοντικές εργασίες.

1.2 Θεωρητικό Υπόβαθρο

Τα συστήματα συστάσεων ορίζονται ως εργαλεία και τεχνικές που προσφέρουν προτάσεις αντικειμένων που δίνουν αξία σε κάποιον χρήστη [6, 36]. Αυτές οι προτάσεις μας βοηθούν στη διαδικασία λήψης αποφάσεων όπως το ποια προϊόντα να αγοράσουμε, ποια μουσική να ακούσουμε, ή ποια ταινία να δούμε. Διαφέρουν από άλλα συστήματα ανάκτησης πληροφορίας όπως οι μηχανές αναζήτησης επειδή περιλαμβάνουν τα κριτήρια της «χρήσιμης» και «προσωποποιημένης» σύστασης.

Προκειμένου να προτείνει αντικείμενα στον χρήστη, ένα σύστημα συστάσεων πρέπει να μπορεί να προσδιορίσει αν ένα αντικείμενο αξίζει να προταθεί. Συνεπώς, πρέπει να έχει την ικανότητα να εκτιμήσει τη «χρησιμότητα» κάθε αντικειμένου, ή να συγκρίνει τη σχετική «χρησιμότητα» δύο αντικειμένων. Για να εκφράσουμε τη χρησιμότητα ενός αντικειμένου i για έναν χρήστη u, χρησιμοποιούμε τον συμβολισμό $R(u, i)$ [1]. Ένα σύστημα συστάσεων προσπαθεί να προσεγγίσει αυτήν τη συνάρτηση με το ελάχιστο δυνατό σφάλμα.

Ανάλογα με το πώς παράγουν τις προτάσεις αντικειμένων στους χρήστης, τα συστήματα συστάσεων χωρίζονται στις εξής κατηγορίες [2, 10]:

- Συστήματα βασισμένα στο περιεχόμενο: Το σύστημα προτείνει αντικείμενα παρόμοια με τα αντικείμενα που άρεσαν στον χρήστη στο παρελθόν. Η ομοιότητα μεταξύ αντικειμένων υπολογίζεται με βάση τα χαρακτηριστικά τους.

- Συστήματα συνεργατικού φιλτραρίσματος: Το σύστημα πρώτα αναγνωρίζει χρήστες με παρόμοιες προτιμήσεις με έναν συγκεκριμένο χρήστη, και ύστερα προτείνει αντικείμενα που τους άρεσαν. Η ομοιότητα μεταξύ χρηστών υπολογίζεται με βάση τα χαρακτηριστικά των αντικειμένων που έχουν ήδη δοκιμάσει.

- Υβριδικά συστήματα: Το σύστημα συνδυάζει τεχνικές βασισμένες στο περιεχόμενο και τεχνικές συνεργατικού φιλτραρίσματος.

Για να μπορέσει να υπολογισθεί η χρησιμότητα ενός αντικειμένου, πρέπει να διαθέτουμε πληροφορίες σχετικά με αυτό. Κάθε αντικείμενο έχει ένα σύνολο από γνωρίσματα ή χαρακτηριστικά που το περιγράφουν και το διαχωρίζουν από τα υπόλοιπα αντικείμενα. Χαρακτηριστικά όπως η διάρκεια ταινίας, το είδος, ή το έτος παραγωγής, λέγονται *δομημένα* επειδή έχουν καθορισμένο μέγεθος και εύρος τιμών [31]. Τα δομημένα χαρακτηριστικά είναι πολύ εύκολο να χρησιμοποιηθούν από αλγορίθμους της τεχνητής νοημοσύνης. Για αυτόν λόγο, συνηθίζουμε να μετατρέπουμε τα μη δομημένα χαρακτηριστικά, που συνήθως έχουν τη μορφή ελεύθερου κειμένου, σε δομημένα. Αυτό μπορεί να επιτευχθεί με τεχνικές όπως το Term Frequency-Inverse Document Frequency (TF-IDF), ή την παραγωγή διανυσμάτων (embeddings).

Η επίδοση ενός συστήματος συστάσεων μπορεί να αξιολογηθεί με τη χρήση μετρικών. Στη βιβλιογραφία έχουν προταθεί πολλές διαφορετικές μετρικές, και δεν υπάρχει μια αντικειμενικά καλύτερη επιλογή. Οι περισσότερες μελέτες επιλέγουν μετρικές ορθότητας (accuracy metrics), κυρίως επειδή είναι απλές, εύκολες στην κατανόηση, και διευκολύνουν τη σύγκριση μεταξύ συστημάτων [7]. Μερικές από τις πιο σημαντικές μετρικές ορθότητας, και αυτές που θα χρησιμοποιήσουμε, είναι μέσο απόλυτο σφάλμα (mean average error ή MAE), το μέσο τετραγωνισμένο σφάλμα (mean squared error ή MSE), η ορθότητα (accuracy), η ανάκληση (recall), και το F_1 score.

Βέβαια, οι μετρικές ορθότητας δεν αρκούν για να προσδιοριστεί πλήρως η επίδοση ενός συστήματος. Για αυτόν τον λόγο έχουν προταθεί και εναλλακτικές μετρικές για πτυχές του συστήματος που τυπικά είναι δύσκολο να ποσοτικοποιηθούν [16]. Μια από αυτές είναι η καινοτομία (novelty), που περιγράφει την ικανότητα του συστήματος να προτείνει αντικείμενα που οι χρήστες δεν γνώριζαν προηγουμένως. Φυσικά, υπάρχουν και άλλα που συχνά αγνο-

ούνται από τη μελέτη των συστημάτων συστάσεων, όπως η κλιμακωσιμότητα (scalability) του αλγορίθμου και η ασφάλεια των προσωπικών δεδομένων των χρηστών.

1.3 Βιβλιογραφική Ανασκόπηση

1.3.1 Μέθοδοι Βασισμένες στο Περιερχόμενο

Η μέθοδος συστάσεων βασισμένες στο περιεχόμενο που θα επιλέξουμε πρέπει να ικανοποιεί δύο βασικά κριτήρια. Πρώτον, στη σύσταση βασισμένη στο περιεχόμενο, το σύστημα δεν μπορεί να βασίζεται σε συστάσεις ή πληροφορίες από άλλους χρήστες, οπότε η ποσότητα των δεδομένων που μπορεί να χρησιμοποιηθεί για κάθε χρήστη είναι περιορισμένη. Επομένως, η μέθοδος που θα διαλέξουμε πρέπει να λειτουργεί με περιορισμένα δεδομένα, πράγμα που αποκλείει περίπλοκες μεθόδους όπως τα νευρωνικά δίκτυα. Δεύτερον, η μέθοδος θα πρέπει να μπορεί να ενσωματώσει εύκολα επιπρόσθετες πληροφορίες, όπως οι συνόψεις ταινιών, χωρίς να αλλάξει η αρχιτεκτονική του μοντέλου. Μια μέθοδος που ικανοποιεί και τις δύο αυτές συνθήκες, ανάμεσα σε άλλα πλεονεκτήματα, ονομάζεται Κ-Πλησιέστεροι Γείτονες (K-Nearest Neighbors ή KNN).

Οι μέθοδοι KNN κατηγοριοποιούν ένα αντικείμενο y βρίσκοντας τα k κοντινότερα αντικείμενα σε ένα σύνολο δεδομένων, και αναθέτοντας στην κλάση που έχουν τα περισσότερα από αυτά τα k αντικείμενα [13].. Συνήθως χρησιμοποιούνται για κατηγοριοποίηση δεδομένων, αλλά μπορούν και εύκολα να επεκταθούν στην πρόβλεψη τιμής παίρνοντας σταθμισμένο μέσο όρο των k κοντινότερων γειτόνων.

Η εύρεση των k κοντινότερων γειτόνων γίνεται συγκρίνοντας κάθε ζεύγος στοιχείων με βάση με συνάρτηση ομοιότητας (similarity function). Για δομημένα δεδομένα όπως αυτά που θα χρησιμοποιήσουμε, προτιμάται η ομοιότητα συνημιτόνου (cosine similarity):

$$cosine\ similarity = S_C(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

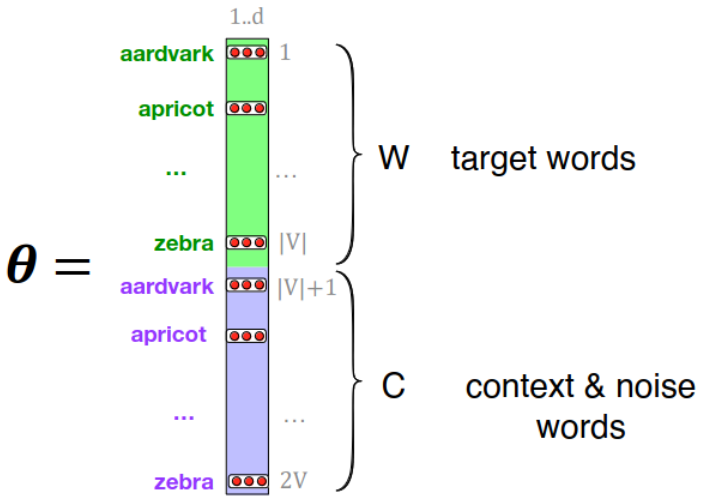
Για να μπορέσουμε να ενσωματώσουμε τις συνόψεις ταινιών στη μέθοδο KNN, πρέπει πρώτα η πληροφορία να μετατραπεί σε δομημένη. Για να περιγράψουμε μια λέξη ή ένα κείμενο ως διάνυσμα, μπορούμε να χρησιμοποιήσουμε μια *αραιή* ή μια *πυκνή* αναπαράσταση. Μια γνωστή αραιή αναπαράσταση ονομάζεται πίνακας όρου-όρου (term-term matrix). Σε ένα σύνολο κειμένων (corpus), κάθε κελί του πίνακα δείχνει πόσες φορές μια επιλεγμένη λέξη εμφανίζεται κοντά σε μια άλλη λέξη. Η έννοια του «κοντά» συνήθως σημαίνει ένα παράθυρο λέξεων γύρω από τη λέξη, για παράδειγμα 4 λέξεις αριστερά και 4 λέξεις δεξιά. Έτσι σχηματίζεται ένας δισδιάστατος πίνακας, όπου λέξεις που μοιάζουν νοηματικά τείνουν να έχουν παρόμοια διανύσματα. Η εικόνα 1.1 δείχνει ένα παράδειγμα πίνακα όρου-όρου. για να ποσοτικοποιήσουμε την ομοιότητα δύο λέξεων, μπορούμε να χρησιμοποιήσουμε μια συνάρτηση όπως η ομοιότητα συνημιτόνου.

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Σχήμα 1.1: Παράδειγμα ενός πίνακα όρου-όρου από ένα σύνολο κειμένων του Wikipedia. Ένα κελί με μεγάλες τιμές σημαίνει ότι οι δύο αντίστοιχες λέξεις εμφανίζεται συχνά κοντά μεταξύ τους. Πηγή: [20].

Μια πιο σύγχρονη διανυσματική αναπαράσταση λέξης ή κειμένου ονομάζεται embeddings. Τα embeddings είναι μια πυκνή αναπαράσταση, με διαστάσεις στο εύρος 50-100, και σε αντίθεση με τον πίνακα όρου-όρου, οι επιτρεπτές τιμές είναι θετικοί και αρνητικοί πραγματικοί αριθμοί, ενώ οι περισσότεροι όροι είναι μη μηδενικοί. Τα πυκνά διανυσματικά τείνουν να έχουν καλύτερες επιδόσεις σε tasks επεξεργασίας φυσικής γλώσσας, για λόγους που δεν είναι ακόμα πλήρως κατανοητοί [20]. Ένας πιθανός λόγος είναι ότι η χαμηλή διαστατικότητα μειώνει την υπερπροσαρμογή (overfitting), ενώ έχει προταθεί και ότι η χαμηλή διαστατικότητα διευκολύνει το να βρίσκονται παρόμοιες νοηματικά λέξεις κοντά μεταξύ τους.

Το 2013 προτάθηκε μια νέα αρχιτεκτονική με την ονομασία «word2vec», η οποία επιτάχυνε σημαντικά τη δημιουργία λεκτικών embeddings από ένα οποιοδήποτε σύνολο κειμένων [26]. Ο αλγόριθμος με την καλύτερη επίδοση λέγεται «skip-gram», και λειτουργεί εκπαιδεύοντας έναν ταξινομητή λογιστικής παλινδρόμησης (logistic regression classifier) που προβλέπει την πιθανότητα μια λέξη συγκειμένου «c» (context word) να βρίσκεται κοντά σε μια λέξη-στόχο «w» (target word). Κατά τη διάρκεια της εκπαίδευσης, το μοντέλο αποθηκεύει δύο embeddings για κάθε λέξη, ένα για όταν είναι συγκείμενο, και ένα για όταν είναι λέξη-στόχος. Αν όλο το λεξιλόγιο έχει μήκος $|V|$ και η κρυφή διάσταση των embeddings είναι d, τότε συνολικά έχουμε $2|V| \cdot d$ βάρη.

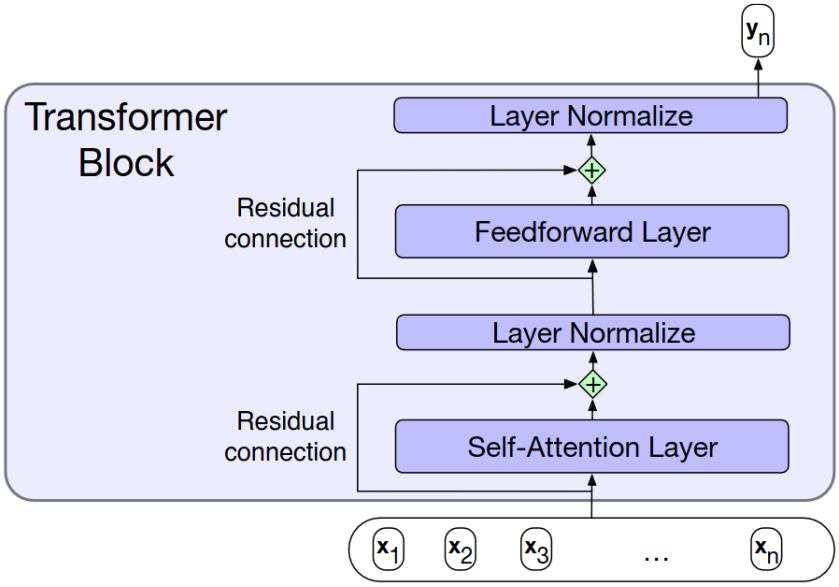


Σχήμα 1.2: Το μοντέλο skip-gram αποθηκεύει δύο embeddings για κάθε λέξη. Ο αλγόριθμος μαθαίνει την παράμετρο θ, που είναι πίνακας $2|V|$ διανυσμάτων, με διάσταση διανύσματος d. Πηγή: [20].

Η αρχιτεκτονική του Word2Vec επεκτάθηκε το 2014 ώστε να μπορεί να παράξει και embeddings για ολόκληρα κείμενα. Έτσι, προτάθηκε ο αλγόριθμος Doc2Vec, ο οποίος εκτός από τα embeddings για κάθε λέξη, αποθηκεύει και ένα διάνυσμα παραγράφου D που περιγράφει το νόημα της παραγράφου συνολικά [23]. Το Doc2Vec μπορεί να εφαρμοστεί σε κείμενα οποιουδήποτε μήκους, και έχει σταθερά καλύτερες επιδόσεις σε tasks τεχνητής νοημοσύνης από τον μέσο όρο των embeddings όλων των λέξεων.

Οι παραπάνω τεχνικές που περιγράψαμε παράγουν *στατικά embeddings*, όπου μια λέξη έχει πάντα την ίδια αναπαράσταση ανεξάρτητα από την πρόταση στην οποία βρίσκεται. Έχουν αναπτυχθεί πιο σύγχρονες τεχνικές, οι οποίες παράγουν *δυναμικά embeddings*. Στα δυναμικά embeddings, η ίδια λέξη μπορεί να έχει διαφορετικά διανύσματα αν έχει διαφορετικό νόημα ανάλογα με το συγκείμενο (context). Τα δυναμικά embeddings μπορούν να περιγράψουν καλύτερα το νόημα των λέξεων, και πλέον προτιμώνται στα περισσότερα tasks επεξεργασίας φυσικής γλώσσας.

Ένα από τα πολύ γνωστά μοντέλα που παράγει δυναμικά embeddings από κείμενο ονομάζεται BERT (Bidirectional Encoder Representations from Transformers) [8]. Η αρχιτεκτονική του BERT βασίζεται σε δομές που ονομάζονται *μετασχηματιστές* (transformers). Οι μετασχηματιστές είναι μη-αναδρομικά δίκτυα που αξιοποιούν την έννοια της *αυτοπροσοχής* (self-attention). Η αυτοπροσοχή επιτρέπει σε ένα δίκτυο να περνάει άμεσα πληροφορία από το ένα επίπεδο στο άλλο, χωρίς τη χρήση ενδιάμεσων συνδέσεων όπως γίνεται σε άλλες αρχιτεκτονικές. Ένα block μετασχηματιστή αποτελείται από ένα επίπεδο αυτοπροσοχής, και ένα επίπεδο πρόσθιας τροφοδότησης (feed-forward layer), με επίπεδα κανονικοποίησης ανάμεσά τους.



Σχήμα 1.3: Απεικόνιση ενός block μετασχηματιστή και των επιπέδων του. Πηγή: [20].

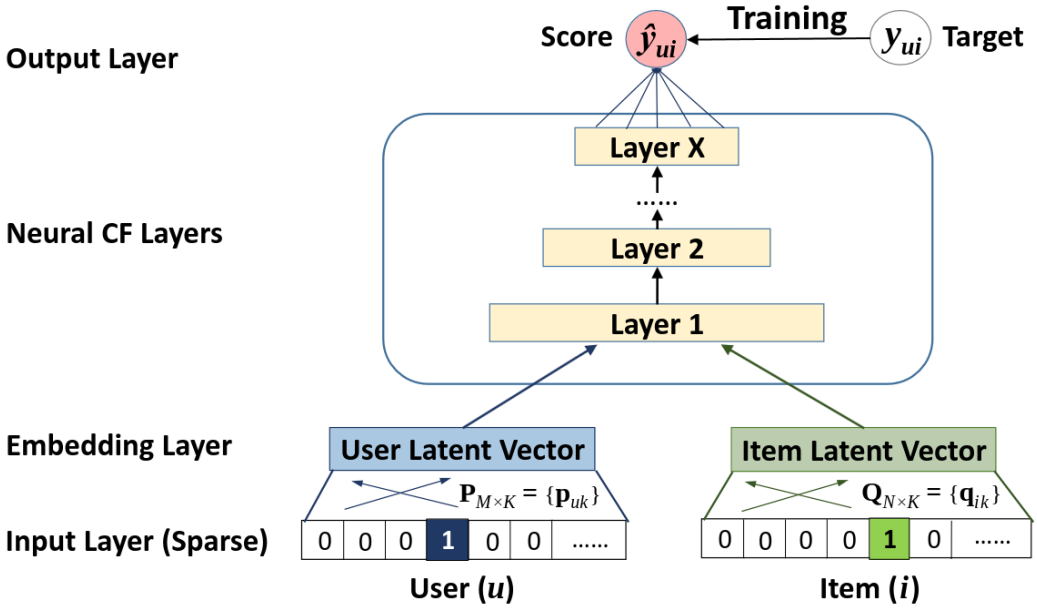
Η έξοδος του BERT είναι μια ακολουθία από διανύσματα, με κάθε διάνυσμα να αντιστοιχεί σε μια λεξικογραφική μονάδα (token). Το πρώτο στοιχείο της ακολουθίας αντιστοιχεί σε ένα ειδικό σύμβολο *[CLS]*, που λειτουργεί ως αναπαράσταση όλου του κειμένου. Στα μοντέλα μας θα χρησιμοποιούμε αυτό το διάνυσμα για να περιγράφουμε τις περιλήψεις ταινιών.

Μια πολύ σημαντική αρχιτεκτονική στο πεδίο της επεξεργασίας φυσικής γλώσσας είναι τα Generative Pre-trained Transformers (GPTs). Τα πιο γνωστά μοντέλα GPT έχουν αναπτυχθεί από την OpenAI, με το πρώτο να δημοσιεύεται το 2018 [33]. Τα GPTs έχουν παρόμοια αρχιτεκτονική με τα μοντέλα BERT, με τη διαφορά ότι χρησιμοποιούν άλλο είδος επιπέδων αυτοπροσοχής, που τα κάνει πιο κατάλληλα για παραγωγή κειμένου. Βέβαια, μπορούν να χρησιμοποιηθούν και για την παραγωγή δυναμικών embeddings, κάτι που θα κάνουμε στα μοντέλα που θα εξετάσουμε.

Η σύσταση αντικειμένων χρησιμοποιώντας embeddings λέξεων έχει εξεταστεί ερευνητικά [30]. Οι συγγραφείς πραγματοποίησαν πειράματα χρησιμοποιώντας στατικά embeddings που δημιουργήθηκαν από το Word2Vec για να προτείνουν ταινίες, σημειώνοντας βελτιωμένη επίδοση σε όλες τις μετρικές συγκριτικά με συστήματα που χρησιμοποιούσαν παλαιότερα μοντέλα όπως το bag-of-words και το TF-IDF. Για τα δικά μας μοντέλα, θα ακολουθήσουμε παρόμοια μέθοδο, δοκιμάζοντας περισσότερη πληροφορία με τη μορφή συνόψεων ταινιών, και χρησιμοποιώντας δυναμικά embeddings που παράγονται από διάφορα μοντέλα BERT και GPT.

1.3.2 Νευρωνικό Συνεργατικό Φιλτράρισμα

Μια ιδιαίτερα επιτυχημένη μέθοδος στο πεδίο της σύστασης αντικειμένων λέγεται Νευρωνικό Συνεργατικό Φιλτράρισμα (Neural Collaborative Filtering ή NCF) [15]. Το νευρωνικό συνεργατικό φιλτράρισμα είναι ένα γενικό πλαίσιο για συνεργατικό φιλτράρισμα που συνδυάζει την παραγοντοποίηση πινάκων με βαθιά νευρωνικά δίκτυα. Στην παραγοντοποίηση πινάκων, ο πίνακας αλληλεπιδράσεων μεταξύ χρηστών και αντικειμένων αποσυντίθεται σε δύο ορθογώνιους πίνακες μικρότερης διαστατικότητας, έτσι ώστε το γινόμενό τους να είναι καλή προσέγγιση του αρχικού πίνακα [22]. Στο νευρωνικό συνεργατικό φιλτράρισμα, ο πίνακας-γινόμενο περνάει από πολλά πλήρως συνδεδεμένα επίπεδα ώστε να προκύψει η τελική πρόβλεψη της βαθμολογίας ενός χρήστη. Η εικόνα 1.4 δείχνει το γενικό πλαίσιο που προτάθηκε.



Σχήμα 1.4: Το πλαίσιο του νευρωνικού συνεργατικού φιλτραρίσματος. Πηγή: [15].

Στην αρχική εργασία, η είσοδος του μοντέλου είναι μόνο ο χρήστης και το αντικείμενο για πρόταση. Επεκτείνουμε το πλαίσιο μεταβάλλοντας την είσοδο του μοντέλου ώστε να δέχεται στατικά ή δυναμικά embeddings ως επιπρόσθετη πληροφορία. Στη δική μας περίπτωση, θα προκύπτουν από την επεξεργασία συνόψεων ταινιών. Έτσι, θα δημιουργήσουμε διαφορετικά υβριδικά μοντέλα που αξιοποιούν πολλαπλά είδη δεδομένων.

1.3.3 Μάθηση με λίγα δείγματα

Η μάθηση με λίγα δείγματα (few-shot learning) είναι μια μέθοδος μηχανικής μάθησης που ασχολείται με μοντέλα που μπορούν να γενικεύσουν από πολύ λίγα δεδομένα. Η μέθοδος γίνεται δίνοντας στο μοντέλο μερικά παραδείγματα του συγκεκριμένου task, χωρίς τη δυνατότητα να αλλάξουν τα βάρη. Το μοντέλο πρέπει να μάθει από αυτά τα παραδείγματα τον σκοπό του task, και να γενικεύσει κατάλληλα. Μια εργασία το 2020 έδειξε ότι αρκετά μεγάλα μοντέλα μπορούν να σημειώσουν εντυπωσιακά αποτελέσματα χωρίς fine-tuning των παραμέτρων [5]. Μια σημαντική παρατήρηση είναι ότι η επίδοση φαίνεται να αυξάνεται με το μέγεθος του μοντέλου.

Στην παρούσα εργασία, θα εξετάσουμε τη μάθηση με λίγα δείγματα στο πεδίο της πρότασης ταινιών χρησιμοποιώντας μοντέλα διαφορετικού αριθμού παραμέτρων. Επιπλέον, θα ενσωματώσουμε συνόψεις ταινιών στα παραδείγματα που θα δίνουμε στο μοντέλο, προκειμένου να εξετάσουμε αν αρκούν ώστε να παραχθούν καλές συστάσεις. Μέχρι τη στιγμή της δημοσίευσης αυτού του άρθρου, αυτή η προσέγγιση δεν έχει προταθεί ή δοκιμαστεί σε άλλες έρευνες.

1.4 Μέθοδος

Για την εκπαίδευση και τη δοκιμή όλων των μοντέλων που κατασκευάσαμε, δημιουργήσαμε ένα νέο σύνολο δεδομένων συνδυάζοντας δύο σύνολα δεδομένων που ήδη υπάρχουν, το «MovieLens 20M» και το «Wikimedia Movie Plots» [29, 40]. Κρατήσαμε μόνο τις ταινίες που εμφανίζονται και στα δύο σύνολα δεδομένων, και συγχωνεύσαμε όλες τις πληροφορίες σε ένα καινούριο dataframe που ονομάσαμε «df_movies». Από τα ratings κρατήσαμε τις πρώτες ένα εκατομμύριο κριτικές λόγω περιορισμών στη διαθέσιμη μνήμη. Όλη η προεπεξεργασία έγινε στη γλώσσα Python με τη βιβλιοθήκη Pandas.

Για τη διεξαγωγή πειραμάτων, σχεδιάσαμε διαφορετικά μοντέλα που χωρίζεται στις εξής κατηγορίες:

- Μοντέλα βασισμένα στο περιεχόμενο: Αυτά τα μοντέλα προτείνουν ταινίες με βάση τις προτιμήσεις του κάθε χρήστη ξεχωριστά. Χρησιμοποιούν μεθόδους K-Κοντινότερου Γείτονα (KNN) και δεν χρησιμοποιούν δεδομένα από άλλους χρήστες.

- Υβριδικά μοντέλα: Αυτά τα μοντέλα συνδυάζουν τα χαρακτηριστικά των ταινιών καθώς και όλες αλληλεπιδράσεις μεταξύ χρηστών και ταινιών. Βασίζονται στο γενικό πλαίσιο που περιγράφεται στο [15].

- Μοντέλα μάθησης με λίγα δείγματα: Αυτά τα μοντέλα αξιοποιούν μεγάλα γλωσσικά μοντέλα για να κάνουν συστάσεις με ελάχιστα δεδομένα. Οι προτάσεις των μοντέλων βασίζονται μόνο σε δεδομένα που σχετίζονται με κάθε χρήστη ξεχωριστά.

Για τα μοντέλα βασισμένα στο περιεχόμενο, χωρίζουμε το σύνολο δεδομένων σε σύνολο για εκπαίδευση και σύνολο δοκιμής. Στόχος είναι να εκτιμηθεί σωστά η βαθμολογία που βάζουν οι χρήστες σε ταινίες από το σύνολο δοκιμής. Τα μοντέλα λειτουργούν σε κάθε

ταινία βρίσκοντας τους k κοντινότερους γείτονες με βάση μια συνάρτηση ομοιότητας, και υπολογίζουν τον σταθμισμένο μέσο όρο όλων των γειτόνων. Κάθε μοντέλο αξιοποιεί έναν διαφορετικό πίνακα ομοιότητας, που προκύπτει συγκρίνοντας διαφορετικά χαρακτηριστικά. Ο πίνακας «genre_sim» δείχνει την ομοιότητα με βάση τα genres, ο πίνακας «director_sim» την ομοιότητα με βάση τους σκηνοθέτες, ενώ οι υπόλοιποι πίνακες συγκρίνουν τις ταινίες με βάση τα διανύσματα που παράγονται από μοντέλα όπως το BERT ή το GPT. Ο πίνακας 1.1 συνοψίζει όλα τα μοντέλα που δοκιμάσαμε. Σημειώνουμε ότι οι μετρικές που χρησιμοποιήθηκαν είναι το μέσο απόλυτο σφάλμα και το μέσο τετραγωνισμένο σφάλμα.

Πίνακας 1.1: Μοντέλα βασισμένα στο περιεχόμενο

Όνομα Μοντέλου	Πίνακας Ομοιότητας
Content-genre	genre_sim
Content-genre-director	genre_sim + director_sim
Content-bert-base	bert_base_sim
Content-bert-imdb	bert_imdb_sim
Content-distilbert-imdb	distilbert_imdb_sim
Content-gpt2s	gpt2_s_sim
Content-gpt2m	gpt2_m_sim
Content-gpt2l	gpt2_l_sim
Content-gpt2xl	gpt2_xl_sim
Content-bert-base-g-d	bert_base_sim + genre_sim + director_sim
Content-bert-imdb-g-d	bert_imdb_sim + genre_sim + director_sim
Content-distilbert-imdb-g-d	distilbert_imdb_sim + genre_sim + director_sim
Content-gpt2s-g-d	gpt2_s_sim + genre_sim + director_sim
Content-gpt2m-g-d	gpt2_m_sim + genre_sim + director_sim
Content-gpt2l-g-d	gpt2_l_sim + genre_sim + director_sim
Content-gpt2xl-g-d	gpt2_xl_sim + genre_sim + director_sim

Για τα υβριδικά μοντέλα, ακολουθούμε την αρχιτεκτονική που παρουσιάστηκε στην αρχική εργασία του νευρωνικού συνεργατικού φιλτραρίσματος. Τα μοντέλα αποτελούνται από επίπεδα embeddings για τους χρήστες και τα αντικείμενα, και τρία πλήρως συνδεδεμένα επίπεδα, με συνάρτηση ενεργοποίησης ReLU. Η έξοδος περνάει από σιγμοειδή συνάρτηση ενεργοποίησης και κλιμακώνεται για να βρίσκεται στο σωστό εύρος τιμών. Ως μετρικές χρησιμοποιούνται το μέσο απόλυτο σφάλμα και το μέσο τετραγωνισμένο σφάλμα.

Η είσοδος του μοντέλου είναι ένα *user_id* και ένα *item_id* που αντιστοιχούν σε συγκεκριμένο χρήστη και συγκεκριμένη ταινία αντίστοιχα. Συμπληρωματικά, κάποια μοντέλα μπορεί να λαμβάνει επιπρόσθετη πληροφορία από τη σύνοψη του κειμένου με τη μορφή embeddings. Αρχικά κατασκευάσαμε το μοντέλο *Hybrid-baseline* που προτάθηκε στην αρχική εργασία και θα χρησιμοποιηθεί ως baseline. Στον πίνακα 1.2 συνοψίζονται τα μοντέλα που εξετάσαμε.

Πίνακας 1.2: Υβριδικά μοντέλα

Όνομα Μοντέλου	Επιπρόσθετη Πληροφορία
Hybrid-baseline	-
Hybrid-doc2vec	Doc2Vec embeddings
Hybrid-bert-base	$\text{BERT}_{\text{BASE}}$ embeddings
Hybrid-bert-imdb	$\text{BERT}_{\text{IMDB}}$ embeddings
Hybrid-gpt2-s	GPT2-small embeddings
Hybrid-gpt2-m	GPT2-medium embeddings
Hybrid-gpt2-l	GPT2-large embeddings
Hybrid-gpt2-xl	GPT2-xl embeddings

Για τα μοντέλα μάθησης με λίγα δείγματα, το task είναι δυαδική ταξινόμηση. Οι μετρικές που χρησιμοποιήθηκαν είναι η ορθότητα (accuracy), η ανάκληση (recall), και το F_1 score. Χαρακτηρίσαμε όλες τις αξιολογήσεις κάτω από 3.5 ως «Δεν Προτείνεται» ή «Not Recommend», και όλες τις αξιολογήσεις από 3.5 και πάνω ως «Προτείνεται» ή «Recommend», με βάση τη σύμβαση που έχει χρησιμοποιηθεί σε προηγούμενες μελέτες [16].

Όλα τα ερωτήματα (queries) προς το μοντέλο έχουν την ακόλουθη δομή:

```
Recommend movies with Yes or No.
[Ένα πλήθος από παραδείγματα]
plot: [Πλοκή ταινίας που εξετάζουμε]
recommend:
```

Τα μοντέλα που εξετάσαμε και οι παράμετροί τους φαίνονται στον πίνακα 1.3.

Πίνακας 1.3: Μοντέλα μάθησης με λίγα δείγματα

Όνομα Μοντέλου	Προσαρμογή	Παραδείγματα	Σειρά	Γλωσσικό Μοντέλο
Fewshot-trunc-2ex-s-125M	Truncation	2	P-N	GPT-Neo 125M
Fewshot-trunc-2ex-r-125M	Truncation	2	N-P	GPT-Neo 125M
Fewshot-2ex-s-125M	Summarization	2	P-N	GPT-Neo 125M
Fewshot-2ex-r-125M	Summarization	2	P-N	GPT-Neo 125M
Fewshot-3ex-s-125M	Summarization	3	P-N-P	GPT-Neo 125M
Fewshot-3ex-r-125M	Summarization	3	N-P-N	GPT-Neo 125M
Fewshot-4ex-s-125M	Summarization	4	P-N-P-N	GPT-Neo 125M
Fewshot-4ex-r-125M	Summarization	4	N-P-N-P	GPT-Neo 125M
Fewshot-2ex-s-1.3B	Summarization	2	P-N	GPT-Neo 1.3B
Fewshot-2ex-r-1.3B	Summarization	2	N-P	GPT-Neo 1.3B
Fewshot-2ex-s-2.7B*	Summarization	2	P-N	GPT-Neo 2.7B
Fewshot-2ex-r-2.7B*	Summarization	2	N-P	GPT-Neo 2.7B
Fewshot-3ex-s-2.7B*	Summarization	3	P-N-P	GPT-Neo 2.7B
Fewshot-3ex-r-2.7B*	Summarization	3	N-P-N	GPT-Neo 2.7B
Fewshot-2ex-s-6B*	Summarization	2	P-N	GPT-Neo 6B
Fewshot-2ex-r-6B*	Summarization	2	N-P	GPT-Neo 6B
Fewshot-2ex-s-20B**	Summarization	2	P-N	GPT-NeoX 20B
Fewshot-2ex-r-20B**	Summarization	2	N-P	GPT-NeoX 20B

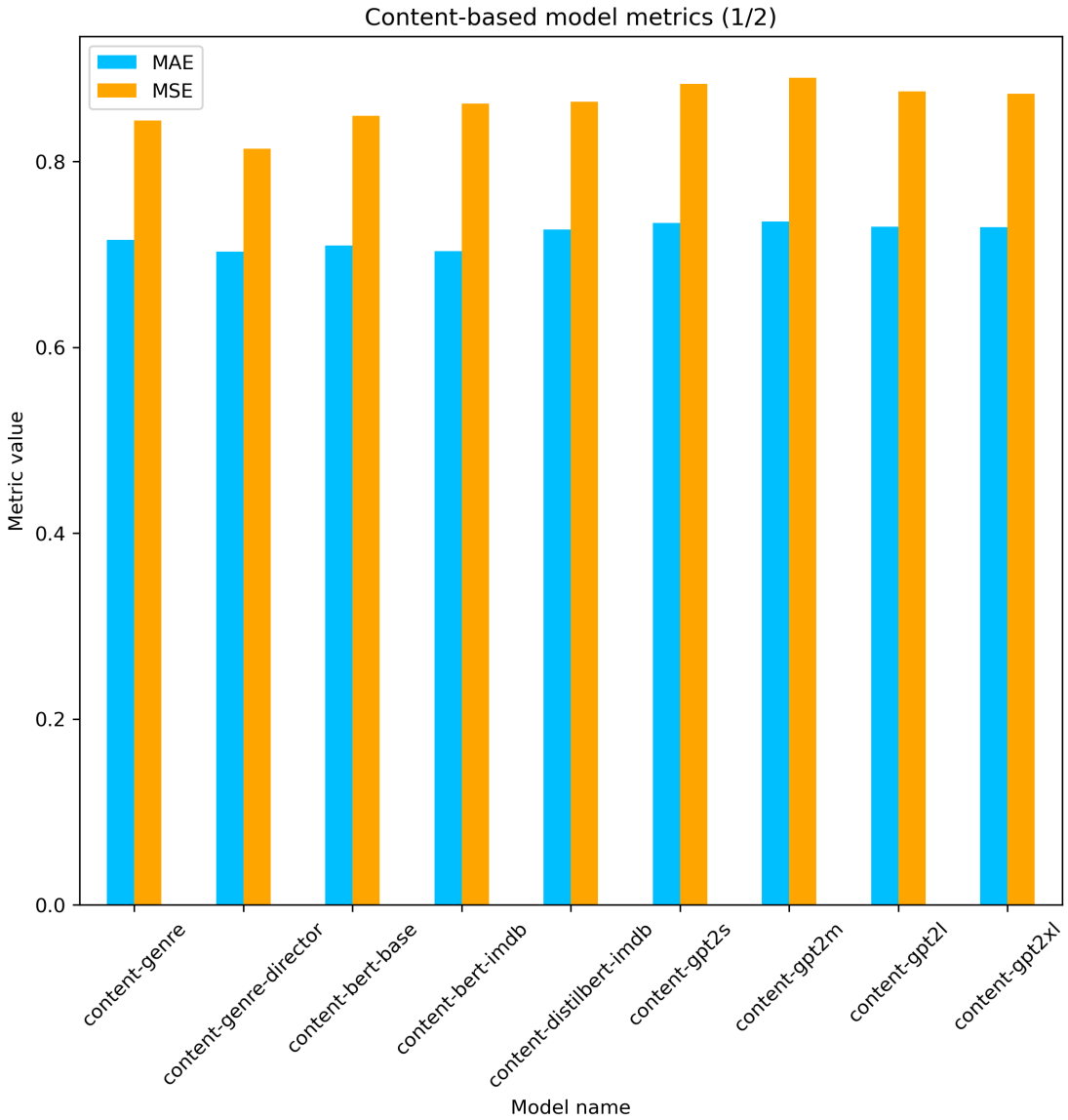
* Τα πειράματα έγιναν χρησιμοποιώντας ακρίβεια τεσσάρων bit λόγω υψηλών απαιτήσεων σε μνήμη. Η χβαντοποίηση των μοντέλων έγινε χρησιμοποιώντας τη βιβλιοθήκη *bitsandbytes* [24].
** Τα πειράματα έγιναν στην πλατφόρμα Google Colab λόγω υψηλών απαιτήσεων σε μνήμη [12].

Επιπλέον, φτιάξαμε τα παρακάτω μοντέλα για να χρησιμοποιηθούν ως baseline:

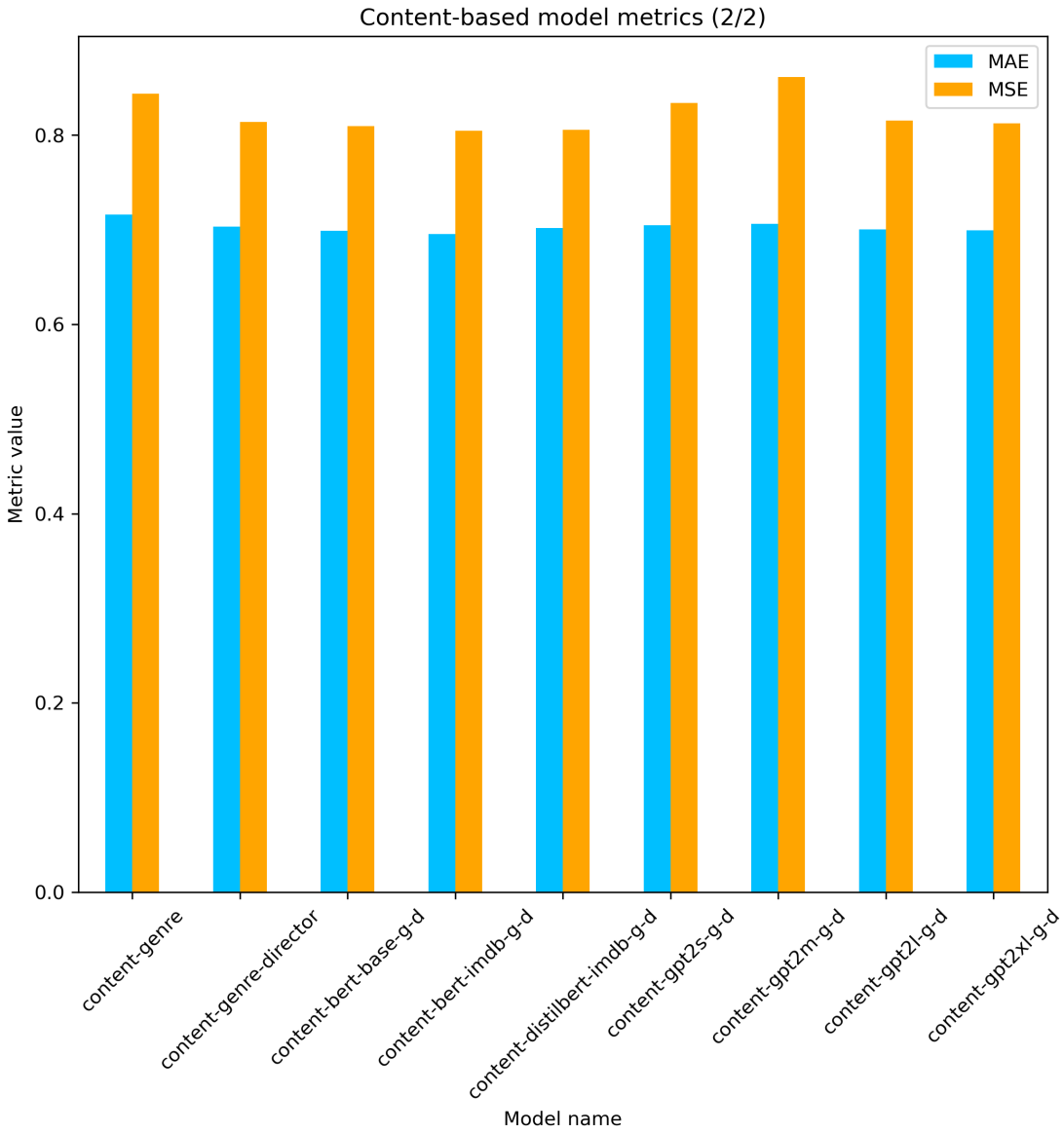
- Fewshot-baseline-dummy: Απλός ταξινομητής που επιλέγει την πιο συνηθισμένη κλάση για κάθε χρήστη.
- Fewshot-baseline-genre: Ταξινομητής KNN που χρησιμοποιεί ομοιότητα με βάση τα genres ($k = 20$).

1.5 Αποτελέσματα

Τα αποτελέσματα για τα μοντέλα βασισμένα στο περιεχόμενο παρουσιάζονται στις εικόνες 1.5 και 1.6. Σημειώνουμε ότι τα baseline μοντέλα εμφανίζονται και στις δύο εικόνες για ευκολία στη σύγκριση.



Σχήμα 1.5: Αποτελέσματα για τα μοντέλα βασισμένα στο περιεχόμενο (1/2).



Σχήμα 1.6: Αποτελέσματα για τα μοντέλα βασισμένα στο περιεχόμενο (2/2).

Από τα αποτελέσματα παρατηρούμε τα εξής:

- Συγκρίνοντας τις δύο εικόνες μεταξύ τους, βλέπουμε ότι όλα τα μοντέλα έχουν βελτιωμένη επίδοση όταν χρησιμοποιούν περισσότερη πληροφορία με τη μορφή μεταδεδομένων όπως τα genres ή οι σκηνοθέτες.
- Από όλα τα μοντέλα που εξετάσαμε, το μοντέλο «content-bert-imdb-g-d» ξεχωρίζει ως το καλύτερο σε επίδοση τόσο στο μέσο απόλυτο σφάλμα όσο και στο μέσο

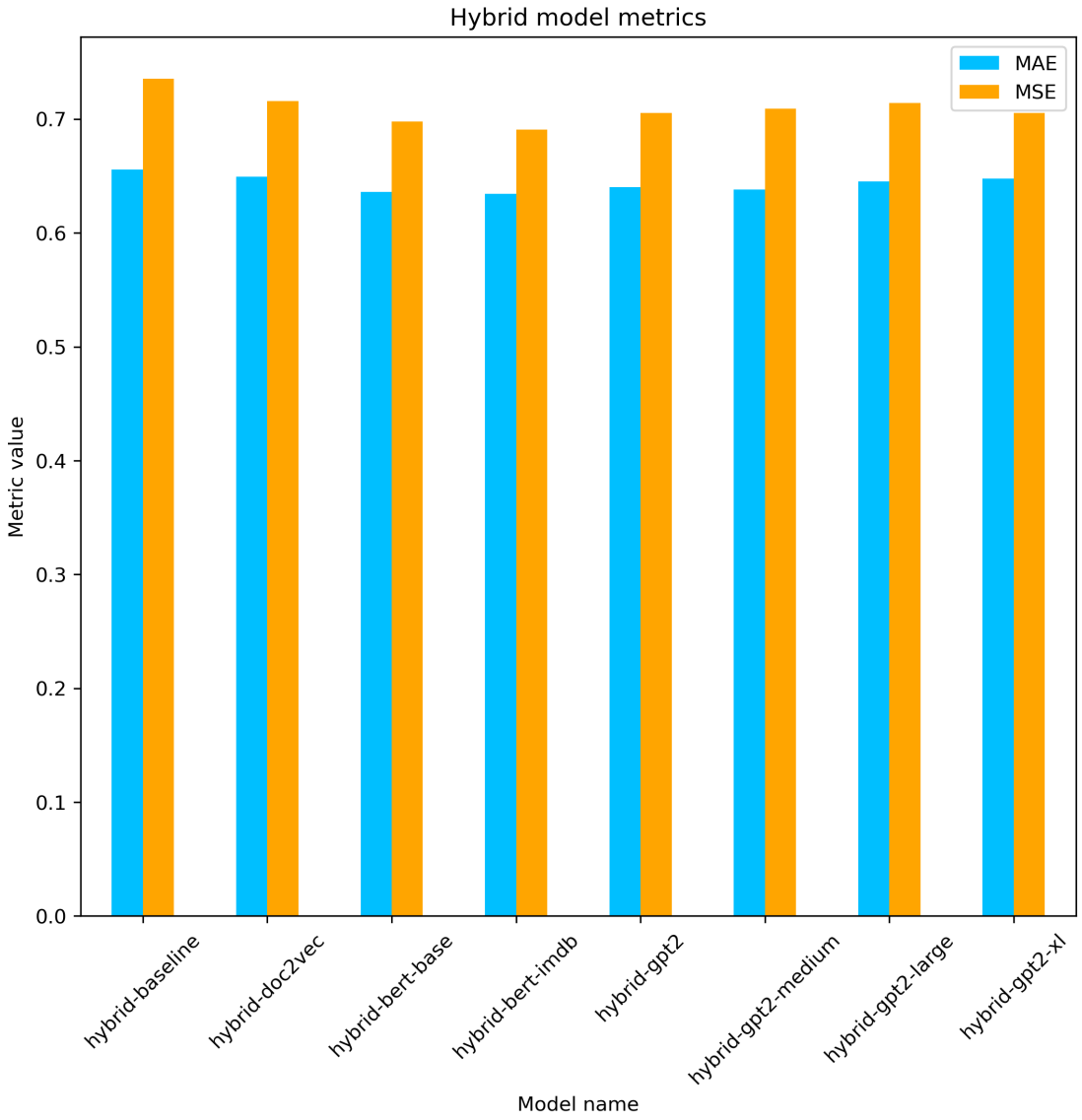
τετραγωνισμένο σφάλμα. Το συγκεκριμένο μοντέλο έχει μια βελτίωση από 1.11% έως και 4.62% σε σχέση με τα baselines.

- Το μοντέλο «content-bert-imdb» έχει καλύτερη επίδοση από το μοντέλο «content-bert-base», γεγονός που αποδίδουμε στην εξειδίκευση που έχει το fine-tuned μοντέλο BERT στην κατανόηση περιεχομένου σχετικό με ταινίες.

- Το μοντέλο «content-distilbert-imdb» έχει παρόμοιες τιμές μετρικών με το «content-bert-imdb» στο μέσο τετραγωνισμένο σφάλμα, αλλά διαφορετικές στο μέσο απόλυτο σφάλμα. Αυτό είναι ένα σημαντικό αποτέλεσμα και δείχνει πως μπορούν να προτιμηθούν distilled μοντέλα που είναι πιο μικρά και πιο γρήγορα, ανάλογα με τη μετρική.

- Όσον αφορά τα μοντέλα GPT, τα μεγαλύτερα μοντέλα δεν είναι καλύτερα από τα πιο μικρά. Αυτό μπορεί να αιτιολογηθεί από την φθίνουσα απόδοση που έχει η αύξηση του μεγέθους των μοντέλων, ή την πιθανή υπερπροσαρμογή (overfitting) των μεγαλύτερων μοντέλων στα δεδομένα.

Τα αποτελέσματα για τα υβριδικά μοντέλα παρουσιάζονται στην εικόνα 1.7.



Σχήμα 1.7: Αποτελέσματα για τα υβριδικά μοντέλα.

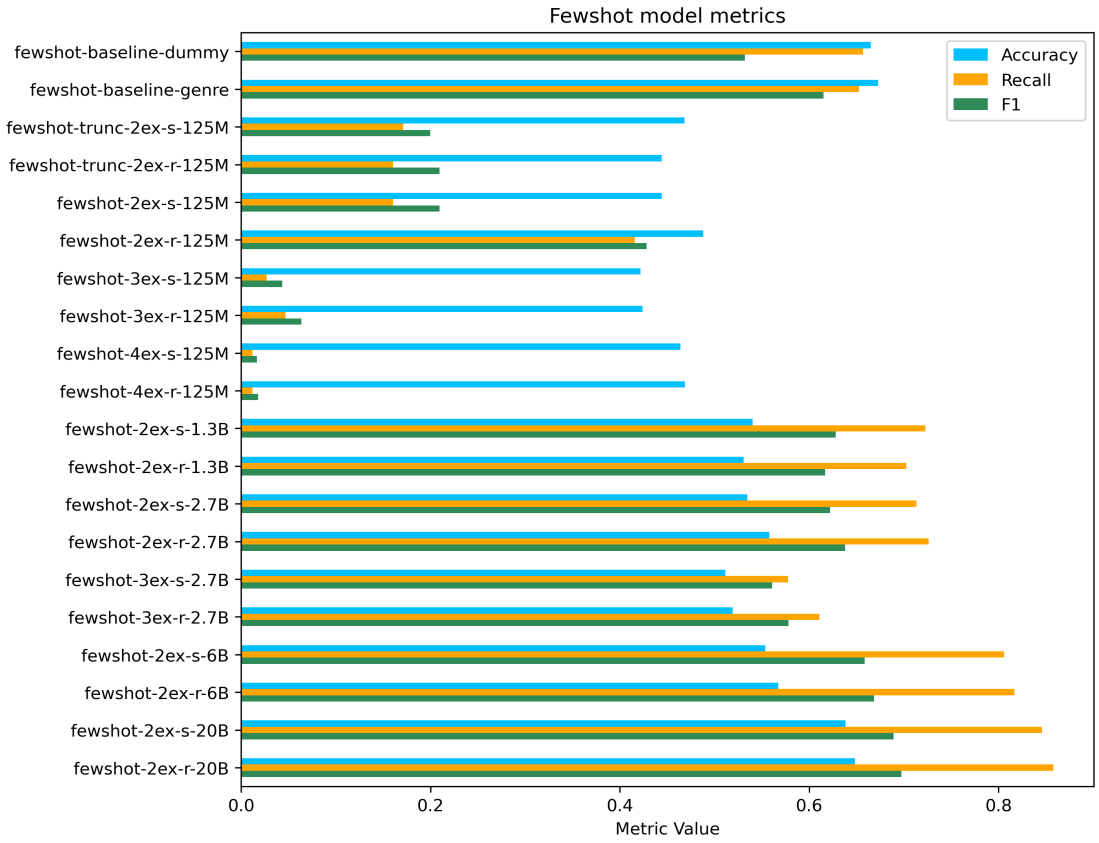
Από τα αποτελέσματα σημειώνουμε τα εξής:

- Όλα τα μοντέλα που σχεδιάσαμε ξεπερνούν τη βασική υλοποίηση του νευρωνικού συνεργατικού φιλτραρίσματος που χρησιμοποιήθηκε ως σημείο αναφοράς.
- Το καλύτερο μοντέλο από αυτά που αξιολογήθηκαν είναι το «hybrid-bert-imdb», καταγράφοντας βελτιώσεις 2.16% και 3.76% στις μετρικές έναντι του «hybrid-baseline».
- Η ενσωμάτωση embeddings από τον αλγόριθμο Doc2Vec φαίνεται να βελτιώνει τις μετρικές, αλλά σε σχετικά μικρό βαθμό. Η παρατήρηση αυτή επιβεβαιώνει τους περιο-

ρισμούς των στατικών embeddings στο να μπορούν να αποτυπώσουν με πληρότητα το περιεχόμενο των συνόψεων για τον σκοπό της σύστασης.

- Το μοντέλο «hybrid-bert-imdb», που βασίζεται σε γλωσσικό μοντέλο που έχει προσαρμοστεί στο πεδίο των ταινιών, υπερέχει του «hybrid-bert-base». Το γεγονός αυτό επιβεβαιώνει την αξία του fine-tuning των μοντέλων στο περιεχόμενο του συγκεκριμένου task που εξετάζεται.
- Όπως και στα μοντέλα βασισμένα στο περιεχόμενο, η αύξηση των παραμέτρων στα μοντέλα GPT δεν εξασφαλίζει βελτιωμένη απόδοση. Αυτό το αποτέλεσμα αμφισβητεί την κοινή υπόθεση ότι τα μεγαλύτερα μοντέλα έχουν πάντα καλύτερα αποτελέσματα.

Τα αποτελέσματα για τα μοντέλα μάθησης με λίγα δείγματα παρουσιάζονται στην εικόνα 1.8.



Σχήμα 1.8: Αποτελέσματα για τα μοντέλα μάθησης με λίγα δείγματα.

Κάνουμε τις εξής παρατηρήσεις:

- Υπάρχει συσχέτιση ανάμεσα στο μέγεθος του μοντέλου και την απόδοση στις μετρικές. Μπορούμε να συμπεράνουμε ότι τα μεγαλύτερα μοντέλα μπορούν να κατανοήσουν καλύτερα το κείμενο που δίνεται και να γενικεύσουν από τα παραδείγματα.

Βλέπουμε επίσης ότι δεν έχουμε φτάσει ένα ανώτατο όριο απόδοσης, οπότε περαιτέρω αύξηση του μεγέθους του μοντέλου μπορεί να συνεχίζει να βελτιώσει τις μετρικές.

- Ο βέλτιστος αριθμός παραδειγμάτων σε κάθε ερώτημα φαίνεται να είναι δύο. Τα μοντέλα που εκπαιδεύτηκαν με τρία και τέσσερα παραδείγματα έχουν μειωμένη απόδοση, με τα τέσσερα παραδείγματα να δίνουν τα χειρότερα αποτελέσματα. Μια πιθανή εξήγηση είναι ότι τα επιπλέον παραδείγματα εισάγουν θόρυβο στο μοντέλο και ωθούν το μοντέλο στο να προτιμά μια συγκεκριμένη κλάση.

- Μια περίεργη παρατήρηση είναι ότι τα μικρότερα μοντέλα που δοκιμάστηκαν, ειδικά αυτά με 125M παραμέτρους, έχουν χειρότερη απόδοση ακόμα και από τα baseline μοντέλα που χρησιμοποιούν πολύ απλές τεχνικές. Αυτό δείχνει ότι τα μικρά γλωσσικά μοντέλα δεν διαθέτουν την απαραίτητη πολυπλοκότητα για να μάθουν από τα λίγα παραδείγματα που δίνονται.

- Όσον αφορά τη σειρά των παραδειγμάτων, η σειρά «αρνητικό-θετικό» έχει καλύτερα αποτελέσματα από τη σειρά «θετικό-αρνητικό», δείχνοντας ότι η σειρά επεξεργασίας από τα γλωσσικά μοντέλα μπορεί να αλλάξει την αποτελεσματικότητά τους. Η διαφορά μπορεί επίσης να αποδοθεί στην ανισορροπία του συνόλου δεδομένων, καθώς ταινίες έχουν περισσότερες θετικές από αρνητικές κριτικές.

- Τέλος, η ανακεφαλαίωση (summarization) των κειμένων βελτιώνει τις μετρικές έναντι της μεθόδου της αποκοπής (truncation). Το γεγονός αυτό δείχνει ότι η ανακεφαλαίωση είναι ικανή να περιγράψει το νόημα των συνόψεων, τουλάχιστον στο συγκεκριμένο task.

Συνοψίζουμε όλες μας τις παρατηρήσεις στα εξής:

- Συνολικά, η ενσωμάτωση συνόψεων ταινιών φαίνεται να βελτιώνει τις επιδόσεις των συστημάτων σύστασης ταινιών, τονίζοντας την αξία της επιπρόσθετης πληροφορίας στη διαδικασία της σύστασης.

- Στη σύσταση βασισμένη στο περιεχόμενο, τα καλύτερα αποτελέσματα προκύπτουν από τον συνδυασμό των συνόψεων με τυπικά μεταδεδομένα, όπως τα genres, το cast, ή την παραγωγή.

- Βλέπουμε ότι υπάρχει ένα όριο στην αποτελεσματικότητα κάθε τεχνικής, το οποίο δεν μπορεί να ξεπεραστεί με την προσθήκη συμπληρωματικών πληροφοριών. Σε περιπτώσεις που είμαστε κοντά σε αυτό το όριο, αλλαγή στην αρχιτεκτονική του μοντέλου μπορεί να φέρει καλύτερα αποτελέσματα.

- Οι τεχνικές σύστασης με λίγα δείγματα δείχνουν πολλές προοπτικές, αφού περιορίζονται μόνο από τον αριθμό παραμέτρων του μοντέλου ή το μέγεθος του context window.

1.6 Συμπέρασμα

Στην παρούσα εργασία, εξετάσαμε αν μπορούν οι περιλήψεις ταινιών να βελτιώσουν την ακρίβεια των συστημάτων σύστασης ταινιών. Αναλύσαμε τους διαφορετικούς τύπους συστημάτων σύστασης και προτείναμε αρχιτεκτονικές στις οποίες μπορούν να ενσωματωθούν πληροφορίες της πλοκής. Στη συνέχεια, δημιουργήσαμε διάφορα μοντέλα βασισμένα στο περιεχόμενο και υβριδικά μοντέλα, που χρησιμοποιούν σύγχρονες τεχνικές μηχανικής μάθησης, καθώς και ένα σύνολο δεδομένων για να βοηθήσει στη διαδικασία σύστασης. Επίσης, δοκιμάσαμε τη χρήση των περιλήψεων ταινιών στη μάθηση με λίγα δείγματα. Συγκρίνοντας τα baseline μοντέλα με αυτά που σχεδιάσαμε, η έρευνά μας δείχνει μια μικρή αλλά μετρήσιμη αύξηση στην επίδοση μετά από την προσθήκη των περιλήψεων. Αυτά τα αποτελέσματα τονίζουν τη σημασία του περιεχομένου στη βελτίωση των ήδη υπαρχόντων συστημάτων σύστασης.

Απαντώντας στην κύρια ερώτηση της εργασίας, συμπεραίνουμε ότι οι περιλήψεις ταινιών μπορούν όντως να βελτιώσουν την επίδοση των συστημάτων σύστασης. Αποδίδουμε τη βελτίωση στην ικανότητα των γλωσσικών μοντέλων να εξάγουν περισσότερη χρήσιμη πληροφορία που δεν βρίσκεται στα τυπικά μεταδεδομένα. Βέβαια, η βελτίωση είναι μικρή, και κάθε τεχνική φαίνεται να έχει ένα δικό της άνω όριο επίδοσης. Όσον αφορά τη δεύτερη ερώτηση σχετικά με την αντικατάσταση των μεταδεδομένων, δεν μπορούμε να βγάλουμε ένα ξεκάθαρο συμπέρασμα. Στα μοντέλα βασισμένα στο περιεχόμενο, η αντικατάσταση των μεταδεδομένων με τα embeddings περιλήψεων δεν έχει πάντα τα καλύτερα αποτελέσματα. Συνεπώς, θεωρούμε ότι τα μεταδεδομένα ακόμα διαδραματίζουν σημαντικό ρόλο σε αυτόν τον τύπο συστημάτων. Από την άλλη, τα συστήματα μάθησης με λίγα δείγματα είχαν εντυπωσιακά αποτελέσματα χωρίς τη χρήση μεταδεδομένων. Σε αυτήν την κατηγορία συστημάτων, οι συνόψεις ταινιών αρκούν για να παραχθούν καλές συστάσεις.

Φυσικά, το συγκεκριμένο θέμα που εξετάσαμε είναι εκτενές και πολύπλευρο, οπότε δεν μπορεί να καλυφθεί πλήρως από μια διπλωματική εργασία. Για αυτό, προτείνουμε τις ακόλουθες κατευθύνσεις για μελλοντική έρευνα:

- **Ενσωμάτωση διαφορετικών ειδών κειμένου και μεταδεδομένων**: Θα μπορούσαν να δοκιμαστούν διαφορετικά κείμενα που περιγράφουν μια ταινία, όπως κριτικές ή σενάρια από trailers. Η ανάλυση διαφορετικών συνδυασμών μεταδεδομένων μπορεί να δώσει μια πληρέστερη εικόνα για τον βέλτιστο συνδυασμό.

- **Δοκιμή διαφορετικών μεθόδων ή αλγορίθμων**: Το πεδίο των συστάσεων αντικειμένων έχει μια πληθώρα από τεχνικές που χρησιμοποιούνται και έχουν καλά αποτελέσματα. Προτείνουμε μεθόδους όπως τα δέντρα αποφάσεων ή τους ταξινομητές δικτύου Bayes που εμφανίζονται συχνά στη βιβλιογραφία.

- **Αξιοποίηση σεναρίων ταινιών**: Το σενάριο μιας ταινίας προσφέρει λεπτομέρειες στην πλοκή που δεν μπορούν να περιγραφούν από μια σύνοψη. Βέβαια, το μεγάλο μήκος των σεναρίων επιβάλλει τη χρήση τεχνικών για να χωρέσουν στο context window των γλωσσικών μοντέλων. Επίσης, η ανάπτυξη νέων μεθοδολογιών για την ανακεφαλαίωση κειμένων θα διευκολύνει σημαντικά την αξιοποίηση δεδομένων στα συστήματα σύστασης.

- **Χρήση μεγαλύτερων γλωσσικών μοντέλων για τη μάθηση από λίγα δείγματα**: Η εργασία μας έδειξε τις προοπτικές της μάθησης με λίγα δείγματα στο task σύστασης ταινιών. Μελλοντική έρευνα θα μπορούσε να εστιάσει στη δοκιμή ακόμα μεγαλύτερων μοντέλων και την αξιολόγηση της κλιμακωσιμότητας της συγκεκριμένης τεχνικής. Ταυτόχρονα, θα μπορούσε να μελετηθεί η επίδραση του context window στην αποτελεσματικότητα της μεθόδου.

- **Δοκιμή σε άλλα αντικείμενα**: Η επέκταση των μεθόδων που αναπτύξαμε σε άλλα αντικείμενα είναι μια ενδιαφέρουσα πρόταση για μελλοντική έρευνα. Για παράδειγμα, η ανάλυση των στίχων τραγουδιών θα μπορούσε να βελτιώσει τα συστήματα σύστασης μουσικής λαμβάνοντας υπόψη το θεματικό περιεχόμενο της μουσικής. Η εξερεύνηση διαφορετικών αντικειμένων μπορεί να επιβεβαιώσει την ευελιξία της προσέγγισής μας, και θα συνέβαλε στην ανάπτυξη πιο εξελιγμένων τεχνικών σε άλλους τομείς.

Chapter 2

Introduction

2.1 Overview and Motivations

In today's digital era, we are often faced with what is called the paradox of choice. While we now have more choices than ever before, granting us unprecedented agency and autonomy, this abundance also complicates our decision-making process. This phenomenon becomes increasingly prevalent in the realm of entertainment. Modern streaming platforms have transformed how we access our entertainment, offering an unprecedented selection of movies, series, and other content at our fingertips. Navigating these countless choices of daily life requires us to make decisions without sufficient knowledge of the alternatives. Consequently, tools or systems that help us make decisions are becoming more and more important.

Movie recommendation, in particular, has attracted significant attention as a research topic, not only because of its direct impact on consumer satisfaction but also due to the immense commercial interest it attracts. Its significance has grown with the evolution of streaming platforms, where personalized recommendations have become central to the user experience. Combined with the abundance of quality data available for analysis, this task presents vast potential for exploring diverse recommendation techniques. Moreover, there are other often neglected aspects such as designing for scalability and ensuring user privacy, which add a layer of complexity and make the task an area for innovation. Given these unique dynamics, movie recommendation systems have emerged as a central focus for both academic and industry experts.

Recommendation systems have become an essential tool for navigating digital context, guiding users towards content tailored to their preferences. These systems usually rely on collaborative filtering techniques such as matrix factorization, which primarily utilizes user interaction data and content metadata for generating recommendations. However, this approach overlooks the rich information that is embedded within textual data such as movie synopses.

The emergence of advanced Natural Language Processing (NLP) techniques and Large Language Models (LLMs) presents a novel and interesting opportunity to enhance

recommender systems. These advancements have enabled us to analyze textual content more comprehensively, unlocking new possibilities in data analysis and enhancing personalized recommendations. This thesis explores the potential of using these modern tools to extract useful information from movie synopses.

2.2 Problem Statement

This study is driven by two research questions. The primary question investigates the impact of movie synopsis data on the performance of movie recommendation systems. It seeks to determine whether incorporating synopses into the recommendation process can improve the accuracy and relevance of recommendations provided to users. If movie synopsis data can indeed enhance the recommendation process, then we should examine the best ways to utilize that additional information, and the extent to which the metrics improve.

The secondary question examines the possibility of utilizing movie synopsis data as a substitute for traditional metadata in recommendation systems. This is important for two reasons. Firstly, modern recommendation systems often rely on traditional metadata to provide meaningful recommendations, so identifying potentially redundant data can improve the system's overall efficiency. Secondly, as state-of-the-art techniques such as few-shot learning operate with minimal data, the question of how much data is truly necessary to make good recommendations becomes intriguing.

We made the following contributions in order to address these two questions:

- The creation of a specialized dataset that combines movie synopsis, metadata, and user reviews.
- The development and evaluation of various models across different classes of recommendation systems.
- The design and execution of experiments to assess the models using selected performance metrics.
- The analysis and discussion of the experimental results, focusing on the efficacy of movie synopses in improving recommender systems.

2.3 Thesis Structure

This thesis is organized into the following chapters:

- Chapter 1 - Introduction: Outlines the motivations behind the study, and the research questions and objectives.
- Chapter 2 - Theoretical Background: Provides the foundational concepts and theories that are relevant to the task of item recommendation.

- Chapter 3 - Literature Review: Reviews the existing methodologies in item recommendation, the current state-of-the-art, and the possible gaps in knowledge.
- Chapter 4 - Method: Describes the dataset that was created, and the architecture of the models that were developed. The experiments that were carried out are are also explained.
- Chapter 5 - Results: Presents the findings of the experiments, including analysis and interpretations of the results.
- Chapter 6 - Conclusion: Summarizes the study's findings, implications, and suggestions for future work.

Chapter 3

Theoretical Background

3.1 Recommender Systems

3.1.1 Basic Concepts

Recommender systems are tools and techniques that provide suggestions for items that are valuable to a user [6, 36]. These recommendations assist in various decision-making processes, such as what products to purchase, what music to listen to, or what movies to watch. Recommender systems become especially useful in situations where an individual must select an item from a potentially overwhelming set of options a service may offer [34].

The distinguishing factors that set recommender systems apart from information retrieval systems or search engines are the criteria of ”interesting and useful” and ”individualized” [6]. While non-personalized recommendations exist and may be useful in specific contexts, they are generally not the focus of academic research.

3.1.2 Knowledge Sources

In order to build their recommendations, recommender systems gather and process various kinds of data. As a general classification, data used by recommender systems can refer to three kinds of objects: items, users, and the interactions between users and items, which are named transactions [36].

Items are the objects that are recommended, and can be characterized by their complexity and their utility to the users. Items have a range of properties and features with variable complexity that recommender systems must take into account. Items with greater complexity might contain more information than is necessary for decision-making. This makes an effective information filtering method crucial [27]. Users of a recommender system usually have varied goals and characteristics. Information about users is utilized in order to fulfill the ”personalized” criterion of recommender systems. The selection of what information to consider depends on the recommendation technique.

Transactions are defined as recorded interactions between users and the recommender systems [36]. They are log-like data that store information exchanged between users and the system and are useful to the recommendation algorithm. The most common forms of transaction data are ratings and item tags [39].

3.1.3 Recommendation Techniques

To fulfill its core function of identifying items for the user, a recommender system must predict that an item merits a recommendation. To accomplish this, the system must be able to evaluate the usefulness of certain items, or compare their relative utility, and then make decisions regarding which items to suggest. A model for the prediction step used by all recommendation algorithms has been suggested in [1]. The utility of the user u for the item i is modeled as a real valued function $R(u, i)$. The recommender system creates predictions for the value of R over pairs of users and items. We denote the estimation of the utility of the pair u, i with $\hat{R}(u, i)$. After calculating the estimation $\hat{R}$ for a specific user on a set of items, the system recommends a small number of them with the highest predicted utility.

In recommender systems, the utility function is usually represented by a rating that indicates how much a particular user likes an item. Utility can be any arbitrary function that depends on the application. For instance, the function may be Boolean, i.e., an indication of whether the user likes the item or not, or a profit function.

Depending on how recommendations are generated, recommender systems are classified into the following categories [2, 10]:

- Content-based recommenders: The system recommends items that are similar to the ones that the user liked in the past. The similarity of items is calculated based on the features of the compared items.
- Collaborative Filtering recommenders: The system identifies users with tastes that are similar to a specific user and then suggests item they like. The similarity in taste is calculated based on the items they have already rated.
- Hybrid recommenders: The system uses methods that combine both content-based and collaborative approaches.

3.2 Item Representation

3.2.1 Structured and Unstructured Data

Content-based and hybrid recommender systems use the features of a dataset's items to predict the utility for a user based on his profile. Candidate items for recommendation are described by a set of features, also called *attributes*, *properties*, or *values*. When each item is represented by the same set of attributes, and there is a known set of values for each attribute, we say that the item is represented with structured data [31]. In the

case of movie recommendation, examples of structured data are duration, genres, or the release year. Such data can be easily incorporated to a machine learning algorithm that creates a user profile.

Unstructured data are defined as data with no attribute names with well-defined values. Examples of unstructured data are unrestricted texts such as news articles or movie reviews. This type of data introduces various challenges when constructing a user profile, primarily stemming from the inherent ambiguity of natural language [36]. As a result, recommendation systems utilize different techniques that convert unstructured data to a structured representation, usually a vector of restricted values.

3.2.2 Representation of Textual Data

Most recommender systems use the Vector Space Model (VSM) to represent text documents. In VSM, each document is represented by a n-dimensional vector, where each dimension represents a term from the total vocabulary of the document set. Each element of the vector has a weight that indicates how much the corresponding term is associated to that specific document. Before the weighting method is applied, the document set or corpus undergoes a series of processing operations such as stop-word removal and stemming.

The most commonly used method for weighting is Term Frequency-Inverse Document Frequency (TF-IDF) weighting. TF-IDF satisfies both the precision and recall functions of information retrieval, since it assigns higher scores to terms that occur frequently in individual documents but infrequently in the remainder of the document corpus [38]. The term frequency (TF) of term t_k in the document d_j is defined as:

$$TF(t_k, d_j) = \frac{f_{k,j}}{max_z f_{z,j}}$$

where $f_{k,j}$ is the frequency of term t_k in document d_j, and $max_z f_{z,j}$ is the maximum computer over the frequencies $f_{z,j}$ of all terms t_z in the document. Inverse Document Frequency is defined as:

$$IDF(t_k, d_j) = \log \frac{N}{n_k}$$

where N is the number of documents in the corpus, and n_k is the number of documents in the corpus where t_k occurs at least once. TF-IDF is then defined as:

$$\textit{TF-IDF}(t_k, d_j) = TF(t_k, d_j) \cdot IDF(t_k, d_j)$$

Additionally, the resulting weights are usually normalized so as to fall in the $[0, 1]$ interval:

$$w_{k,j} = \frac{\textit{TF-IDF}(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} \textit{TF-IDF}(t_s, d_j)^2}}$$

After the weights for all documents have been calculated, the semantic proximity between two documents is calculated using a similarity measure. When the vector space

model is used, the cosine similarity measure is a popular option [38]:

$$sim(d_i, d_j) = \frac{\sum_k w_{k,i} \cdot w_{k,j}}{\sqrt{\sum_k w_{k,i}^2} \cdot \sqrt{\sum_k w_{k,j}^2}}$$

Despite its simplicity, it is still widely used and performs competitively with newer and more complex algorithms [31]. More advanced methods of producing weights that utilize machine learning models will be discussed in the next chapter.

3.3 Metrics

The quality of a recommender system can be evaluated by comparing the produced recommendations to a set of known user ratings. Several different metrics have been proposed to evaluate a system's performance [16]. The majority of published evaluations of recommender systems have adopted a system's *accuracy* as a measure of its effectiveness. An accuracy metric quantifies the difference between the system's predicted item ranking and the user's actual preference order. Recommendation accuracy metrics can be broadly classified into three classes: *predictive accuracy* metrics, *classification accuracy* metrics, and *rank accuracy* metrics.

3.3.1 Predictive Accuracy Metrics

Predictive accuracy metrics evaluate the proximity between the predicted ratings and the actual ratings given by users. This type of measure is most appropriate for scenarios in which an accurate prediction of the ratings for all items is of high importance. Predictive accuracy has been used to evaluate some of the earliest recommendation systems in literature [35].

The simplest such metric is the *mean absolute error* (often referred to as MAE), which measures the average absolute deviation between the predicted and user ratings:

$$MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N}$$

where N is the number of items, p_i is the predicted rating of an item, and r_i is its actual rating. Besides its relative simplicity, mean absolute error has thoroughly examined statistical properties that facilitate comparison between systems.

Two important measures related to mean absolute error are *mean squared error* (MSE) and *root mean squared error* (RMSE), which square each error before summing them:

$$MSE = \frac{\sum_{i=1}^{N} |p_i - r_i|^2}{N}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} |p_i - r_i|^2}{N}}$$

Both of these metrics put more emphasis on large errors, which makes them more frequently used in the context of recommender systems [36].

3.3.2 Classification Accuracy Metrics

Classification metrics measure the frequency with which a system correctly or incorrectly determines whether an item is favorable. Therefore, they are suitable for tasks where users have definite binary preferences. These types of metrics do not directly assess an algorithm's ability to accurately predict ratings. Variations from the user ratings are acceptable, provided they do not result in classification errors.

Precision and recall were introduced the 1966 as metrics for evaluating information retrieval systems and have remained popular ever since [7]. They are also commonly used in item recommendation [16]. In order to calculate precision and recall, the item set must be separated into two classes — relevant (N_r) and not relevant (N_{nr}). The item set must also be divided into the items that were selected for recommendation (N_s), and the items that were not (N_{ns}). Precision is the proportion of relevant items chosen (N_{rs}) out of the total items selected:

$$P = \frac{N_{rs}}{N_s}$$

Conversely, recall is the proportion of relevant items chosen out of all available relevant items. It indicates the probability that a relevant item will be selected:

$$R = \frac{N_{rs}}{N_r}$$

Recall is often impractical to measure in a recommender system, since it requires knowing the relevance of each item for all users. A common way to approximate the true values of recall and precision is splitting the dataset of user ratings into a training set and test set, and predict the top N items *for which ratings exist.*

A useful metric that is often used in binary item recommendation studies is accuracy, which describes the ratio of "correct" predictions out of all predictions. The true positives are the relevant items that were selected N_{ns}, and the true negatives are the not relevant items that were not selected N_{nrns}. Accuracy is defined as::

$$Acc = \frac{N_{rs} + N_{nrns}}{N}$$

It is important to note that the definition of "relevance" is nontrivial and a subject of discourse in the field of information retrieval, especially in the context of recommender systems [14]. If a dataset has a rating scale that is not already binary, it must be converted into a binary one, and the threshold for item relevance may vary significantly between users.

Measuring precision alone may be suitable if the user does not need a comprehensive list of all potentially relevant items. However, if the objective is to identify all relevant

items, Precision and recall both must be considered. These two metrics are inversely related, so an approach to combine them into a single metric is usually needed. A popular metric that attempts to summarize the precision and recall into a single value is the F_1 metric:

$$F_1 = \frac{2PR}{P+R}$$

Metrics based on the Receiver Operating Characteristic (ROC) curve are an alternative to precision and recall. The ROC model seeks to evaluate how well an information system can differentiate between signal (relevance) and noise. It operates under the assumption that the system will assign a predicted level of relevance to every potential item. ROC-based metrics work best on datasets where there is a clear binary relevance relationship, so they are not ideal for the task of movie recommendation.

3.3.3 Rank Accuracy Metrics

Rank accuracy metrics assess how well a recommendation system can generate an item ranking that aligns with how the user would have ordered the same items. These metrics are more appropriate than classification metrics in systems that present ranked recommendation lists, especially when user preferences are not binary [16]. Rank accuracy metrics also distinguish between the "best" and "good" choices and are best suited to domains where that distinction is important. However, if a recommender system will be displaying the projected rating values, then it should be evaluated using predictive accuracy metrics like those examined in the previous sections.

An important factor to consider when choosing an appropriate rank accuracy metric is that of partial ordering. A system's reference ranking (i.e. the ranking based on the user's ratings) may have items with the same ratings. This is more common in contexts like movie recommendation where the user's ratings are usually in the form of a five-star system. In those cases, rank correlation metrics like Spearman's ρ or Kendall's τ might not be the best choice, since they are designed to handle complete orderings. The Normalized Distance-based Performance Measure (NDPM) is suitable for such cases [41]:

$$NDPM = \frac{2C^- + C^u}{2C^i}$$

In the above equation, C^- is the number of pairs that the system ranking asserts incorrectly. C^u is the number of pairs where the system ranking ties but the reference ranking does not tie, and C^i is the number of pairs of items for which the reference ranking has a preference. NDPM is not affected for different orderings when the user rankings are tied. However, it shares the same issue as other rank correlation metrics, where ranking changes at the top of the ranking are weighted the same as those at the bottom.

3.3.4 Metrics Beyond Accuracy

While in this section we have focused on accuracy-based metrics, accuracy alone does not ensure an effective and satisfying user experience in recommender systems. For example,

a system might attain high accuracy by only recommending popular items similarly to a simple popularity list, or make predictions only for items that are more easy to predict. In both these cases, the criterion of "interesting and useful" that we have used for the definition of a recommender system is not satisfied. Therefore, we must also consider other measures that enhance a system's ability to recommend in less obvious ways.

Coverage is a measure of the proportion of items that the system can recommend. If a recommender system has low coverage, then it might be less helpful to some users as it inherently limits their options [36]. The simplest way to measure coverage is by calculating the percentage of all items for which predictions can be made. Another measure named *catalog coverage* expresses the percentage of items that are recommended to at least one user during a specific period.

Novelty describes a recommender system's ability to introduce users to items they were previously unaware of [16]. It has been suggested that when it comes to providing "value" to users, novelty was as important as accuracy [21]. If a user study can be conducted, a system's novelty can be measured by asking users if they already familiar with an item that was recommended. When doing an online experiment, the dataset could be split based on time, so that the training dataset only has user ratings before a specific point in time. Then, the system loses novelty whenever it recommends items that were rated before that point, and gains novelty whenever it recommends items that were rated after that point.

A similar but distinct metric is serendipity. A recommendation is considered serendipitous when it helps a user find a surprisingly interesting item that might not have been discovered otherwise [16]. To distinguish between novelty and serendipity, consider a user that likes movies of a certain director. If a system recommends the director's latest movie, the recommendation is novel, but not surprising. Providing serendipitous recommendations is nontrivial, since the best and most accurate recommendations are usually not surprising.

Of course, metrics apart from those explained above do exist. A recommender system should be designed for scalability when dealing with large collections of items, adhere to contemporary privacy standards, and foster confidence and trust it its users.

A common aspect that most non-accuracy metrics share is that they are generally harder to measure accurately [16]. Additionally, a standard way to measure them has not been developed. This makes it more challenging to rely on them when evaluating recommender systems in academic research.

Chapter 4

Literature Review

4.1 Movie recommendation in literature

The domain of movies is frequently used in the research field of recommendation systems [32]. A systematic review of 26 recommender systems in literature has found that the movie domain is the most common domain with 10 occurrences [32]. This is partly because movies as items come with a wealth of data, much of which is stored in text. This textual data can be easily processed and manipulated, thanks to the development of numerous algorithms designed to extract useful information from such sources.

Another important reason for the popularity of movies in case studies is that quality data in the movie domain is plentiful and easily available. GroupLens Research, a human-computer interaction research lab at the University of Minnesota, has created a dataset named MovieLens with several movie ratings that can be used in recommendation research. The latest update of the dataset contains 25 million ratings and 1 million tag applied to 62000 movies by 162000 users, which can be used for both content-based and collaborative filtering recommendations [28]. The MovieLens research lab created a web-based recommender system in 1997 also named MovieLens, which garnered significant media and research attention. IMDb, an online database of movie information, also produces a dataset that contains movie titles along with ratings that have been submitted by users of the site [19].

In Fall 2009, the recommender systems field reached a significant milestone with the completion of the $ 1,000,000 Netflix Prize competition. The competition began in 2006 and annually awarded the algorithm that most accurately predicted user ratings for movies, using the root mean squared error as a metric [3].

4.2 History of the State of the Art

Item recommendation has been studied in literature since the 1990s with the introduction of Tapestry, an automatic filtering system designed to recommend documents drawn from newsgroups [11]. The first recommender systems relied on basic correlation

statistics and predictive modeling, without fully utilizing the broader technique in statistics and machine learning [4]. The task of collaborative filtering was then treated as a classification problem, which allows the application of various algorithms from machine learning literature.

As discussed above, the availability of public datasets and their significance to e-commerce fueled additional research. The Netflix Prize in 2006 initiated a competition for the best collaborative filtering algorithm, and Matrix Factorization, based on numerical linear algebra and statistical matrix analysis, became the state-of-the-art in the recommendation domain.

Modern recommendation systems rely on various techniques adopted from the field of Artificial Intelligence (AI) such as deep learning, natural language processing, and reinforcement learning [42].

4.3 Content-based Methods

This section presents a range of machine learning techniques that serve as the foundation for the content-based models discussed in this paper.

The content-based recommendation method we choose should satisfy two key criteria. First, in content-based recommendation, the system cannot rely on recommendations or information from other users, so the amount of data that can be used for each user is limited. Therefore, the chosen method must be efficient even with limited data, which excludes complex methods like neural networks due to their high data requirements. Second, the method needs the flexibility to incorporate additional information, such as movie synopses, which we aim to test. A method that satisfies both of these conditions and has several other advantages is called K-Nearest Neighbors.

4.3.1 K-Nearest Neighbors Algorithm

The K-Nearest Neighbors (KNN) method is a simple and effective non-parametric method for classification [13]. It classifies a new object with input vector y by looking at the k nearest points in the training dataset and assigning it to the majority class among these k points. The KNN method can also be used for estimating continuous variables by calculating a weighted average of the k nearest neighbors.

Nearest neighbor methods require choosing a value k, as well as defining the proximity or similarity of each pair of vectors. The simplest form of KNN sets $k = 1$, which creates a classifier with high variance and sensitivity to data. Increasing k makes predictions more consistent but increases the bias of the method since more averaging is introduced [13]. Techniques for finding the best k without testing do exist, but they are not consistent and heavily depend on the data at hand. The best k can be determined by testing various values of k in the training dataset, and selecting the one with the lowest error or highest accuracy.

The nearest neighbor algorithm's similarity function between two data points A and B varies with data type. For structured data, the Euclidean distance is common, while

in vectors space models the cosine similarity is favored:

$$cosine\ similarity = S_C(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

In the Euclidean distance function, two data points with small values in a specific feature are considered equivalent to two data points with large values. In contrast, the cosine similarity function yields a lower value when the data points both have small values. This makes cosine similarity better suited for text, ensuring that documents are considered similar when they are about the same topic, and not when they are both not about the same topic [31].

The nearest neighbor method boasts several appealing features. It's simple to understand and to implement, requires no training or optimization, and often rivals more complex methods like neural networks in accuracy [31]. Additionally, it requires a relatively small amount of data, and it can be easily adapted to multiple classes. We will use KNN techniques to build the content-based models that are presented in the next chapter.

It should be noted that KNN has a non-negligible computational cost during the classification of new instances. This is primarily because the majority of the calculations occur during classification and not when the training dataset is initially processed. Some techniques have been developed that minimize the computation at query time, but they are out of scope for this paper.

4.3.2 Word and Document Embeddings

Sparse Representation

In the previous chapter we discussed that documents can be represented as a vector of term counts. A similar idea can be applied to words, where each word is represented as a vector of term counts. In the *term-term matrix*, each cell shows how many times the row (target) word occurs in the same context as the column (context) word [20]. The context is a window around the word, for example 4 words to the left and 4 words to the right, but can extended to the whole document.

Figure 4.1 shows a subset of the term-term co-occurence matrix in a specific document corpus. Similar words have similar vectors since they tend to appear in similar documents. Thus, we can use determine how similar two words are by comparing their vectors. The most common similarity metric used to compare vectors is the cosine of the angle between the vectors.

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Figure 4.1: An example of a term-term matrix from a corpus of Wikipedia documents. A cell with high counts means that the two corresponding words often appear together in a context. Retrieved from [20].

In a corpus with vocabulary size $|V|$, the resulting matrix has dimensions $|V| \times |V|$. $|V|$ is usually between 10000 and 50000 words, after stopwords and very infrequent words are discarded. It should be noted that this word representation is sparse, since most of the cells are zero.

Word2Vec

A more advanced word representation that uses short dense vectors is called *embeddings*. Embeddings represent words in a short format, with a number of dimensions in the range 50-1000. In contrast to the term-term matrix, embeddings have both positive and negative real-valued numbers, and most of the values are non-zero.

Dense vectors generally outperform sparse ones in NLP tasks. The reasons behind this are not fully understood [20]. The low-dimensionality of embeddings may help with generalization and reduce overfitting when training classifiers or other models. Another possible explanation is that dense vectors might better capture the concept of synonymy, as they can better represent similar words like boat and ship more closely than disjoint, unrelated dimensions in sparse representations.

In 2013, a new architecture named *word2vec* was introduced that greatly increased the speed of creating word embeddings for a corpus [26]. The best performing algorithm of word2vec is named *skip-gram*. Skip-gram trains a logistic regression classifier that predicts the probability that a word c is a real context word for a target word w. This probability is calculated by getting the dot product of the embeddings of the target word and the context word. The sigmoid function $\sigma(x)$ is then applied:

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

While training, two embeddings are stored for every word, a matrix W that is used when that word is a target, and a matrix C for when it is used as a context. Each element in the embeddings has a dimension of d that is selected beforehand. Thus, the algorithm has to learn $2|V| \cdot d$ weights.

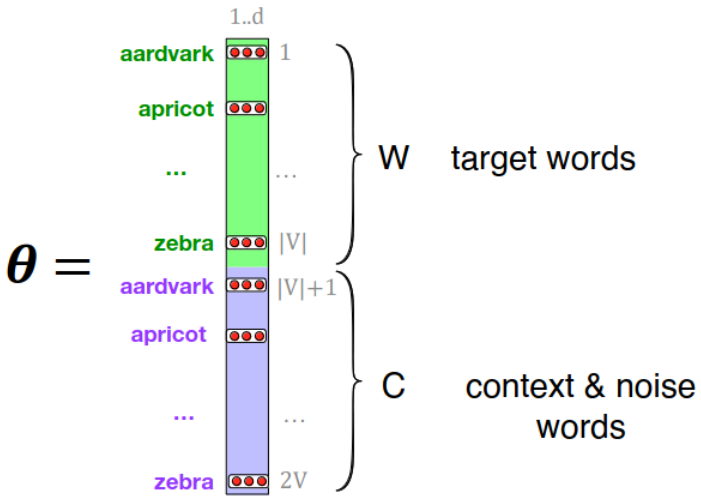


Figure 4.2: The skip-gram model stores two embeddings for each word, the target embedding and the context embedding. The algorithm learns a parameter θ, which is matrix of 2|V| vectors, each of dimension d. Retrieved from [20].

Before the training has begun, the corpus of documents is processed in order to produce positive and negative examples. Each example consists of a word w and a context word c. Positive examples are created by taking a context word that exists in the corpus, while negative examples are created by choosing a random word from the total vocabulary. The algorithm starts with randomly initialized W and C matrices, and adjust those embeddings to [20]:

- Maximize the similarity of pairs (w, c_{pos}) drawn from the positive examples
- Minimize the similarity of pairs (w, c_{neg}) drawn from the negative examples.

For a target word w, correct context word c_pos, and incorrect context word c_neg, the loss function is calculated as:

$$L_{CE} = -(log\ \sigma(c_{pos} \cdot w) + log\ \sigma(-c_{neg} \cdot w))$$

The final representations for each word is created by adding the target embedding w and the context embedding c. This creates one vector of size d for each word in the vocabulary.

Doc2Vec

In 2014, an architecture that applies the principles of Word2Vec to create fixed-length feature vectors for texts was proposed [23]. The paper introduces the Doc2Vec algorithm, which can be applied to text of any length and outperform simple-average of Word2Vec

vectors. For each document, Doc2Vec stores word embeddings for every word called W, as well as a unique paragraph vector D that represents the document's context.

There are two implementations presented in the paper: Paragraph Vector - Distributed Memory (PV-DM), and Paragraph Vector - Distributed Bag of Words (PV-DBOW). In PV-DM, the paragraph vector D and the word embeddings W are concatenated, and then used in order to predict the next word. The prediction task is done via a multiclass classifier, such as softmax:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

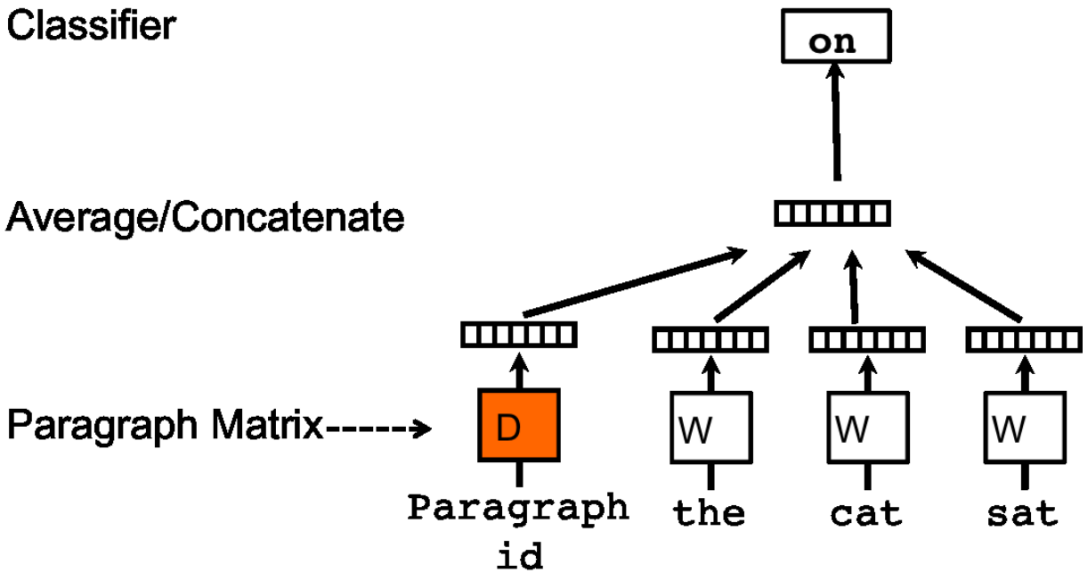


Figure 4.3: A representation of Doc2vec using the Distributed Memory implementation. In order to predict the next word, the paragraph's matrix with a context of three words is used. Retrieved from [23].

Given a sequence of words $w_1, w_2, w_3, \ldots, w_T$, and a document d, the probability p for a word w_t to exist in a given context is:

$$p(w_t|w_{t-k}, \ldots, w_{t+k}) = \frac{e^{w_t}}{\sum_i e^{y_i}}$$

The variable y_i is the log-probability for each output word, computed as:

$$y = b + Uh(w_{t-k}, \ldots, w_{t+k}; W)$$

where U, b are the softmax parameters, and h is the concatenation of the paragraph vector D and the word embeddings W as described above.

In PV-DBOW, the only input is the paragraph vector, the models tries to predict words randomly sampled from the paragraph. This implementation is simpler and less performant than PV-DM. However, it requires less data as it only uses the softmax weights as input, and not the word vectors that were used by PD-DM. The Doc2Vec architecture produces paragraph vectors that is a combination of the 2 implementation described above.

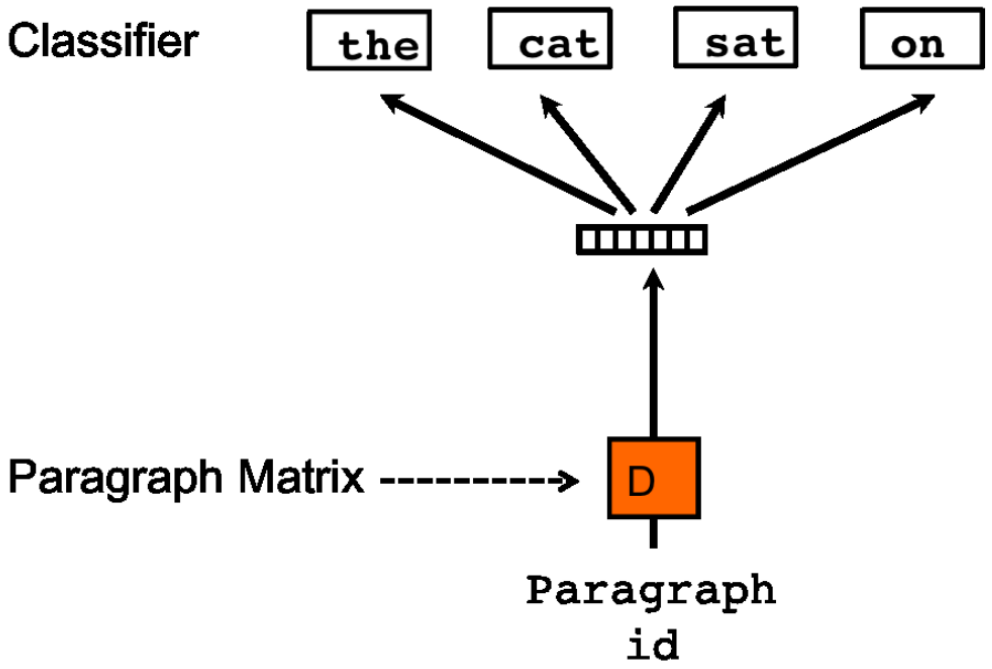


Figure 4.4: A representation of doc2vec using the Distributed Bag of Words implementation. Here, the paragraph matrix is trained to predict all words in the context window. Retrieved from [23].

Both Word2Vec and Doc2Vec embeddings are *static embeddings*, meaning that the algorithm learns one fixed embedding for each word in the vocabulary [20]. Static embeddings are not able to capture the different contexts in which a specific word might be used.

Transformers and BERT

In contrast to static embeddings that essentially represent vocabulary entries, contextual embeddings represent instances of words in a particular context. For example, the word

"left" in the sentence "I left my phone in the left side of the table" will have a single vector representation if static embeddings are used, but contextual embeddings produce two distinct vectors that better capture the word's context. Contextual embeddings can be used for assessing the semantic similarity of words within their specific context, and are preferred in linguistic tasks that necessitate understanding of word semantics.

The most well-known model that creates contextual embeddings for textual data is called Bidirectional Encoder Representations from Transformers, or most commonly BERT [8]. BERT's model architecture is a multi-layer bidirectional transformer encoder and is based on the transformer block.

Transformers are non-recurrent networks that utilize the concept of *self-attention*. Self-attention enables a network to directly access and utilize information from arbitrarily large contexts, bypassing the need to pass it through intermediate connections as other architectures like Recurrent Neural Networks do [20]. A self-attention layer maps an input sequence $(x_1, \ldots, x_n)$ to an output sequence of the same length $(y_1, \ldots, y_n)$.

In self-attention, the outputs y_i are based on a set of comparison operations between elements of the sequence. Depending on the flow of information, self-attentions layers are classified into the following categories:

- Causal self-attention layers, where in each element of the sequence, the model takes into account all inputs up to, and including, the current one.
- Bidirectional self-attention layers, where in each element of the sequence, the model takes into account all inputs, both before and after the current one.

Figures 4.5 and 4.6 illustrate the structure and the information flow in causal and bidirectional self-attention layers respectively.

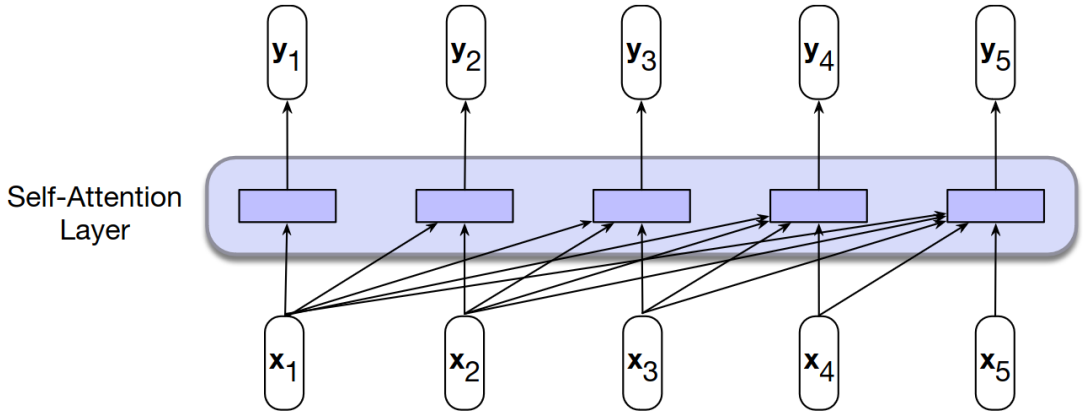


Figure 4.5: A visual representation of a causal self-attention layer. Retrieved from [20].

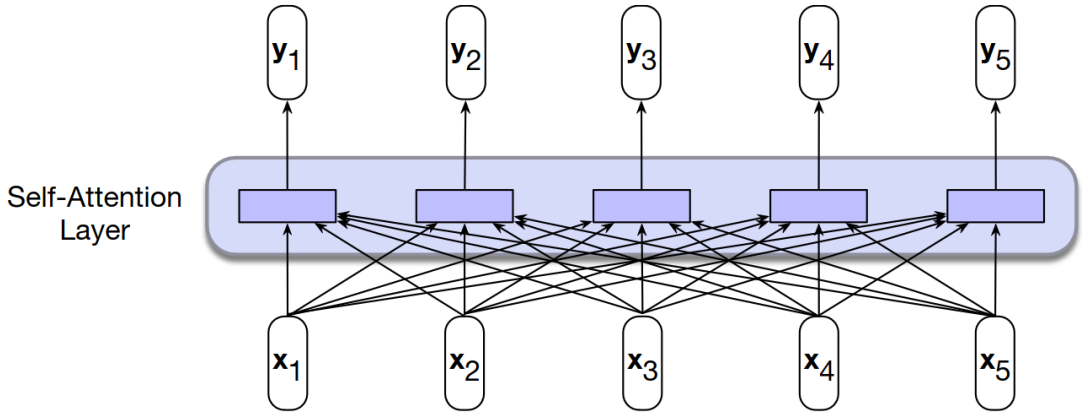


Figure 4.6: A visual representation of a bidirectional self-attention layer. Retrieved from [20].

The transformer block consists of a self-attention layer followed by a feed-forward layer, with residual connections and layer normalizations following each. The calculations for each items can be conducted independently from all other computations, meaning that training and inference can be parallelized.

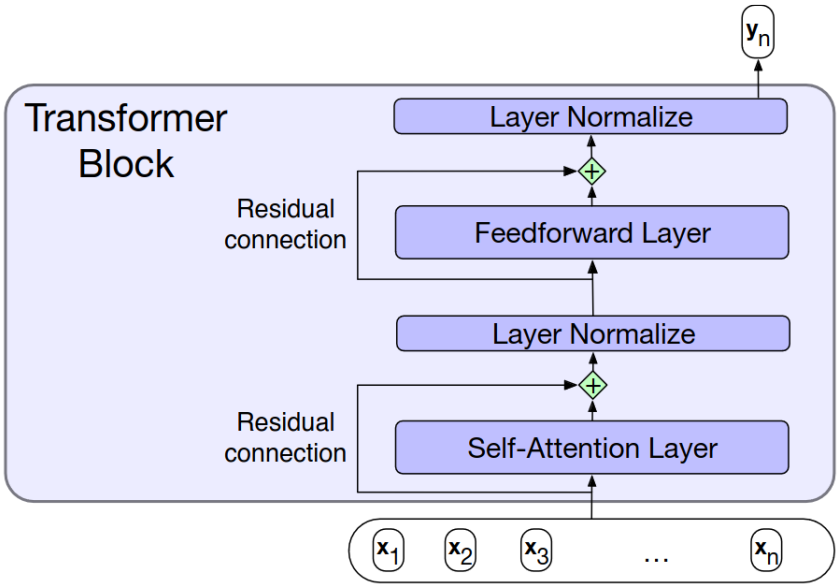


Figure 4.7: A visual representation of a transformer block and its layers. Retrieved from [20].

The original paper that proposed BERT described the following architectures [8]:

- $BERT_{BASE}$: Consists of 12 transformer blocks, each of size 768, with a total *110M parameters*
- $BERT_{LARGE}$: Consists of 24 transformer blocks, each of size 1024, with a total *340M parameters*

The BERT model's output is a sequence of vectors, with each element in the sequence corresponding to one token of the input. The first token of every sequence is a special classification token, denoted as "[CLS]". The model can be fine-tuned to a specific task by selecting the appropriate tokens. For the purpose of item recommendation, we can use the final hidden state corresponding to the "[CLS]" token as a vector representation of the entire sequence.

BERT can be efficiently compressed with little impact on accuracy, a valuable feature for practical applications [37]. The main techniques that have been used are knowledge distillation, quantization, and pruning. For example, DistilBERT, a compression of BERT based on knowledge distillation, retains 97% of BERT performance, while reducing its size by 40% and operating 60% faster.

Generative pre-trained transformers

Generative Pre-trained Transformers (GPTs) have become a widely popular architecture in the field of artificial intelligence for their advanced capabilities in language understanding and generation. GPTs are language models (LM) that are based on the transformer architecture.

The most influential GPT models have been developed by OpenAI, with its first model GPT-1 releasing in 2018. GPT-1 and its successors have an architecture that is similar to BERT. However, they differ in the type of self-attention that they utilize. GPT models use causal self-attention layers, where every token can only attend to previous tokens [33]. This makes them more suitable for tasks like text generation. However, GPT models are also able to produce sentence-level or paragraph-level contextual embeddings, used in tasks like item recommendation.

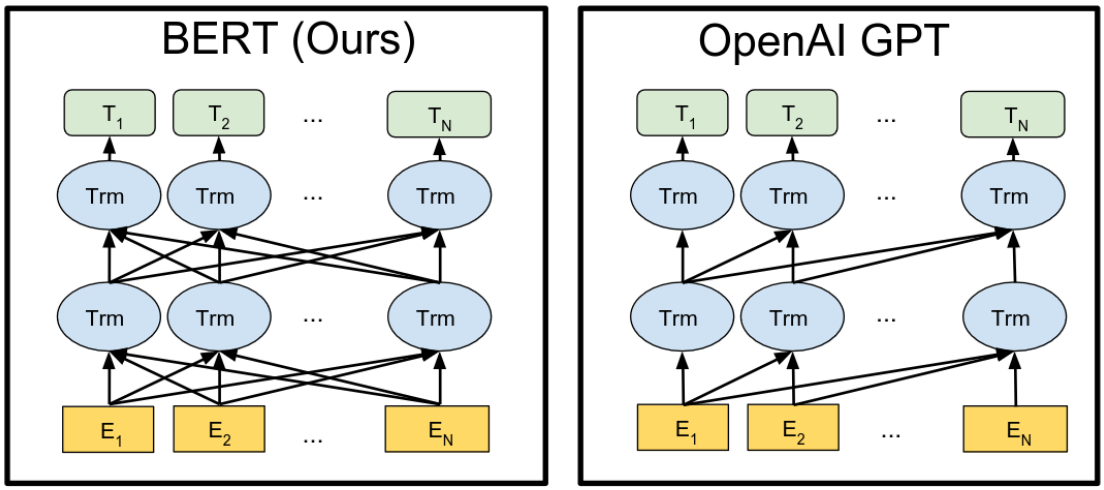


Figure 4.8: A comparison of the architectures of BERT and GPT models. BERT uses bidirectional transformer blocks, while GPT uses left-to-right transformer blocks. Retrieved from [8].

Research has been done to test content-based item recommendation using word embeddings [30]. The authors carried out experiments with using static embeddings generated from Word2Vec to recommend movies based on similarity, demonstrating improved performance across all metrics compared to baseline systems utilizing traditional models like bag-of-words and TF-IDF. For content-based recommendation, we adopt this model as a baseline and enhance the method by incorporating movie plots as additional textual information, and using more advanced contextual embeddings from various BERT and GPT models.

4.4 Neural Collaborative Filtering

The most prominent class of collaborative filtering algorithms used in item recommendation is *matrix factorization*. Matrix factorization algorithms work by approximating the user-item interaction matrix through decomposition [22]. Users and items are represented in a lower dimensional latent space, with weights such that the dot product of these matrices provides good approximation of the original user-item interaction matrix. Gaining widespread recognition through the Netflix Prize, matrix factorization has become the standard in model-based recommendation systems. Since then, various techniques directed at improving matrix factorization have been introduced.

Figure 4.9: A visualization of matrix factorization. The model learns a user embedding matrix and an item embedding matrix such that their product is a good approximation of the user-item interaction matrix. Retrieved from [25].

A method that has been highly successful and improved on the state-of-the-art results in item recommendation achieved is *neural collaborative filtering*. Neural collaborative filtering (NCF) is a general framework for collaborative filtering based on the combination of matrix factorization with deep neural networks [15]. The original paper proposes a multi-layer model, as shown in the figure below. The model's input is two feature vectors v_u and $v_i i$ which describe a specific user and a specific item. The next layer is an embedding layer, where the sparse representation of the input vectors are mapped to a dense vector. Then, the model consists of several layers named *neural collaborative filtering layers*. The original paper used a standard multilayer perceptron (MLP) to build the neural collaborative filtering layers. The final layer simply output a predicted rating $\hat{y}_{u,i}$ that user u will give to item i. Training is done by trying to minimize the loss between the predictions $\hat{y}_{u,i}$ and the actual ratings $y_{u,i}$ with a method lie stochastic gradient descent. Figure 4.10 shows the architecture of the neural collaborative filtering framework.

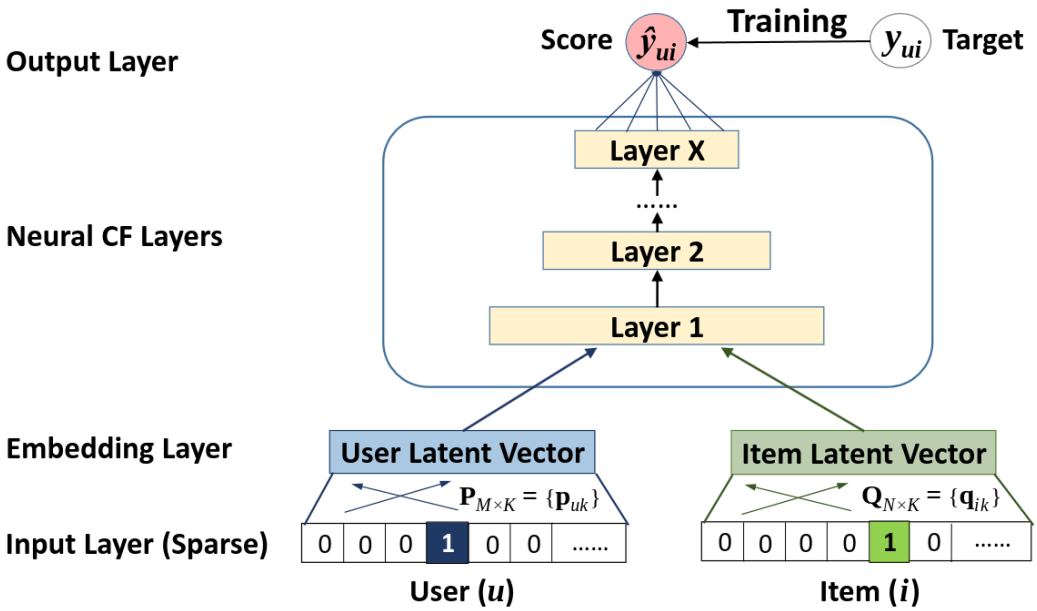


Figure 4.10: The neural collaborative filtering framework. Retrieved from [15].

The original paper focused on pure collaborative filtering, meaning only user and item identities were utilized as input. We build upon this model by modifying the model's input to accept static or contextual embeddings as additional data. In our case, these embeddings will be created by processing movie plots. We thus create a hybrid recommender system that uses both item information and information from other users.

4.5 Few-Shot Learning

Few-shot learning is a machine learning method that focuses on models that can understand or generalize from a very limited amount of data. While traditional machine learning models often require large datasets to achieve high accuracy, few-shot learning models are designed to learn to perform tasks with only a few examples. In few-shot learning, the model is given a few demonstrations of the task at inference time, but without the ability to update any weights. In cases where the number of examples is one or zero, the method is called one-shot learning and zero-shot learning respectively.

A paper in 2020 demonstrated that relatively large language models can achieve impressive performance in few-shot learning without fine-tuning [5]. The researchers used pretrained language models like GPT-3 to perform common NLP tasks like question answering or translation. The number of examples for each query depends on how many examples can fit in the model's context window.

The bigger models like GPT-3 showed competitive results with state-of-the-art techniques that used fine-tuned models. An important observation is that the difference in

performance in zero-shot, one-shot, and few-shot learning typically increases with the size of the model. This suggests that larger models may be more effective at meta-learning.

In the original work, several common NLP tasks such as question answering, translation, and a superset of tasks referred to as SuperGlue, were examined [5]. In our study, we evaluate few-shot learning for the task of movie recommendation using language models of various sizes. Additionally, we incorporate movie synopses in each query to test whether the models can capture the movies' plots the and provide good recommendations. As of the time of release of this paper, this approach has not been proposed or tested in other research.

Chapter 5

Method

This paper's main objective is to test whether additional information in the form of a movie's synopsis can provide more accurate recommendations. For the purposes of this task, various models have been created. They can be classified into the following categories:

- Content-based models: These models recommend movies based only on the user's preferences. They use KNN methods and do not rely on data from other users.
- Hybrid models: These models leverage both the content features of the movies and the user-item interactions. They are based on the framework as described in [15].
- Few-shot learning models: These models use large language models to make accurate recommendations with minimal user data. The model's recommendations are based solely on the data specific to an individual user.

5.1 Dataset

To train and test the models described above, we developed a new dataset by integrating two distinct datasets named MovieLens 20M and Wikimedia Movie Plots [29, 40].

5.1.1 MovieLens 20M

MovieLens 20M is a dataset of ratings and tags from MovieLens, a movie recommendation service. It contains 20,000,263 ratings and 465,564 tags across 27,278 movies. The dataset was created on October 17, 2016 and includes data that were created by 138,493 users between January 09, 1995 and March 31, 2015.

The dataset is split into six files, of which the following two were used for our new dataset:

- `movie.csv`: Contains movie information, 27,278 items.

- `rating.csv`: Contains ratings of movies by users. In total, there are 20,0000,263 ratings that range from 0.5 to 5, with increments of 0.5.

Figure 5.1 shows the number of each rating value that exists in the MovieLens 20M dataset. Note that it is clear that users tend to give favorable review score more often than note, with an average rating of 3.49. The median and mode score are 3.5 and 4.0 respectively.

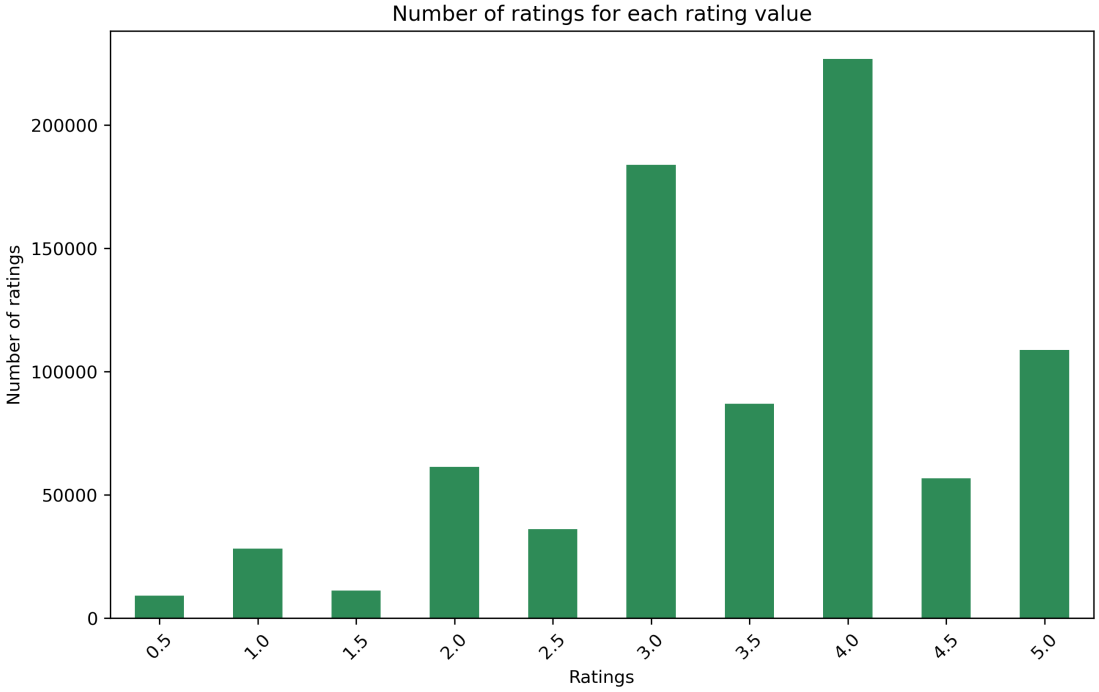


Figure 5.1: Counts of each rating value from 0.5 to 5, based on user reviews from the dataset.

5.1.2 Wikipedia Movie Plots

Wikipedia Movie Plots contains plot summary descriptions that were scraped from Wikipedia. It contains metadata and descriptions on 34,886 movies. It includes only one file with information on the release year, country of origin, director, actors, genres, and a long form description on the movie plot.

5.1.3 Preprocessing

The two datasets described above are preprocessed in order to create the dataset that will be used as the training data for our models. The preprocessing part consists of the following steps:

- The files `filesmovie.csv` and `rating.csv` from MovieLens, and `wiki\_movie\_plots\_deduped.csv` from Wikimedia Movies Plot, are parsed into dataframes using the Python library Pandas.
- The two dataframes are merged into a new dataframe named "df_movies", only keeping the rows with entries in both dataframes.
- All ratings about movies that are not present in df_movies are deleted, and the first one million ratings are kept in order to fit storage constraints.
- New indexes named "user_id" and "movie_id" are created.

The few-shot learning models create prompts that include movie plots and them feed them to a large language model. In order to have prompts that always fit the language model's context window, we create a modified version of the "df_movies" was created. The dataframe is named "df_movies2" and contains movies where the size of the movie plot after tokenization is applied is less than 500 tokens.

5.2 Content-based Models

For all content-based models, the task is to perform regression on the movie ratings. The dataset is split into a training and a test dataset. All models utilize the training dataset to try and predict the rating a specific user will give to a specific movie. The metrics used are mean absolute error (MAE) and mean squared error (MSE).

We have chosen to use KNN method for all content-based models. While other machine learning methods like neural networks require large datasets to learn and make accurate predictions, KNN can work effectively with the data of a single user. All models work by finding the k closest neighbors from the training dataset based on a similarity metric, and then calculating the weighted average from those neighbors.

Each model's differentiating factor is the similarity matrix they are using. In order to test whether movie plot information can improve the models' accuracy, we create various similarity matrices that use different movie metadata. The similarity matrices created are:

- genre_sim: The similarity based on each movie's genres.
- director_sim: The similarity based on each movie's directors.
- bert_base_sim: The similarity based on the vectors that were created by passing the movie's plot to $\text{BERT}_{\text{BASE}}$
- bert_imdb_sim: The similarity based on the vectors that were created by passing the movie's plot to a fine-tuned version of BERT on the IMDB dataset.
- distilbert_imdb_sim: The similarity based on the vectors that were created by passing the movie's plot to a fine-tuned version of DistilBERT on the IMDB dataset.

- gpt2_s_sim: The similarity based on the vectors that were extracted after passing the movie's plot to GPT-2-small.
- gpt2_m_sim: The similarity based on the vectors that were extracted after passing the movie's plot to GPT-2-medium.
- gpt2_l_sim: The similarity based on the vectors that were extracted after passing the movie's plot to GPT-2-large.
- gpt2_xl_sim: The similarity based on the vectors that were extracted after passing the movie's plot to GPT-2-xl.

The fine-tuned versions of BERT and DistilBERT were imported to our workspace using the transformers Python library [18, 17]. From the combination of these similarity matrices, the following models were created:

Table 5.1: Content-based models and their configuration

Model Name	Similarity Matrix
Content-genre	genre_sim
Content-genre-director	genre_sim + director_sim
Content-bert-base	bert_base_sim
Content-bert-imdb	bert_imdb_sim
Content-distilbert-imdb	distilbert_imdb_sim
Content-gpt2s	gpt2_s_sim
Content-gpt2m	gpt2_m_sim
Content-gpt2l	gpt2_l_sim
Content-gpt2xl	gpt2_xl_sim
Content-bert-base-g-d	bert_base_sim + genre_sim + director_sim
Content-bert-imdb-g-d	bert_imdb_sim + genre_sim + director_sim
Content-distilbert-imdb-g-d	distilbert_imdb_sim + genre_sim + director_sim
Content-gpt2s-g-d	gpt2_s_sim + genre_sim + director_sim
Content-gpt2m-g-d	gpt2_m_sim + genre_sim + director_sim
Content-gpt2l-g-d	gpt2_l_sim + genre_sim + director_sim
Content-gpt2xl-g-d	gpt2_xl_sim + genre_sim + director_sim

For each model, grid search was performed in order to find the best value of the hyperparameter k in a set of 12 possible values from 1 to 100.

5.3 Hybrid Models

For the hybrid models, the task is to perform regression on the movie ratings. All models follow the architecture that was suggested in the original neural collaborative filtering paper [15]. The models are comprised of embedding layer for users and items, and three multi-layer perceptron (MLP) layers, with the ReLU activation function. The model's output is passed through a sigmoid function, and scaled in order to fit the range of possible ratings. The metrics used are mean absolute error (MAE) and mean squared error (MSE).

The model's input is an $item\_id$ and a $user\_id$. Additionally, each model may receive additional input in the form of document embeddings of the movie's plot. The document embeddings have a predetermined size denoted as $embeddings_{size}$.

In more detail, each model consists of:

- An embedding layer for the users and the items, with a hidden dimension of 768
- A layer that concatenates the two embeddings, as well as the plot embeddings that may be passed as additional input
- An MLP layer with input size $2 * 768 + embeddings_{size}$ and output size 64
- An MLP layer with input size 64 and output size 32
- An MLP layer with input size 32 and output size 1
- A sigmoid activation function that scales the output to the range $[0, 5]$

The following three models were initially created:

- Hybrid-baseline: The model does not receive any additional embeddings as input, and thus is used as baseline.
- Hybrid-doc2vec: The model receives Doc2Vec embeddings as input. The Doc2Vec embeddings were created using the Gensim library with parameters $vector\_size = 100, window = 5, epochs = 10$ [9].
- Hybrid-bert: The model receives BERT embeddings as input.

In order to determine the best hyperpameters, grid search on the learning rate and batch size was performed for all three models. After the best hyperpameters for *Hybrid-bert* were found, additional models that use embeddings from pre-trained large language models were created. In total, the following configurations were tested:

Table 5.2: Hybrid models and their configuration

Model Name	Supplemental information used
Hybrid-baseline	-
Hybrid-doc2vec	Doc2Vec embeddings
Hybrid-bert-base	$\text{BERT}_{\text{BASE}}$ embeddings
Hybrid-bert-imdb	$\text{BERT}_{\text{IMDB}}$ embeddings
Hybrid-gpt2-s	GPT2-small embeddings
Hybrid-gpt2-m	GPT2-medium embeddings
Hybrid-gpt2-l	GPT2-large embeddings
Hybrid-gpt2-xl	GPT2-xl embeddings

5.4 Few-shot Learning Models

For this section, the task is changed from regression to binary classification. As a result, the metrics used here are accuracy, recall, and F_1 score. All ratings below 3.5 were converted to ”Not Recommend”, and all ratings from 3.5 and above were converted to ”Recommend”. This follows the convention that has been used in previous studies that use the MovieLens dataset [16]. Figure 5.2 shows the number of ”Recommend” and ”Not Recommend” labels in the dataset. Note that there is an imbalance in the dataset in favor of the ”Recommend” label.

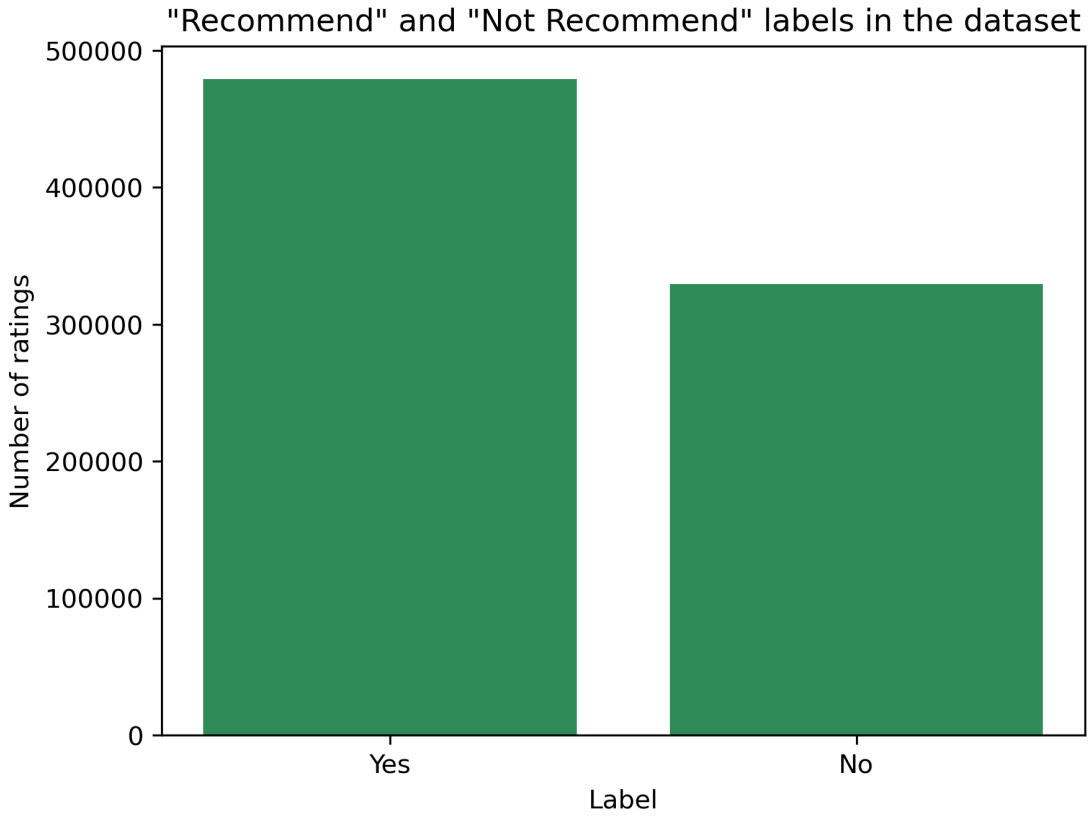


Figure 5.2: The number of ”Recommend” and ”Not Recommend” labels in the dataset used for this section.

An example has the following structure, with ### signifying the EOS token:

```
plot: [Movie plot]
recommend: [Yes/No]
###
```

All models use the following prompt in queries:

```
Recommend movies with Yes or No.
[Any number of examples]
plot: [Movie plot of query]
recommend:
```

The configurations tested and their properties are presented in the following table:

Table 5.3: Few-shot learning models and their configuration

Model Name	Fitting method	No. of examples	Order	LLM used
Fewshot-trunc-2ex-s-125M	Truncation	2	P-N	GPT-Neo 125M
Fewshot-trunc-2ex-r-125M	Truncation	2	N-P	GPT-Neo 125M
Fewshot-2ex-s-125M	Summarization	2	P-N	GPT-Neo 125M
Fewshot-2ex-r-125M	Summarization	2	P-N	GPT-Neo 125M
Fewshot-3ex-s-125M	Summarization	3	P-N-P	GPT-Neo 125M
Fewshot-3ex-r-125M	Summarization	3	N-P-N	GPT-Neo 125M
Fewshot-4ex-s-125M	Summarization	4	P-N-P-N	GPT-Neo 125M
Fewshot-4ex-r-125M	Summarization	4	N-P-N-P	GPT-Neo 125M
Fewshot-2ex-s-1.3B	Summarization	2	P-N	GPT-Neo 1.3B
Fewshot-2ex-r-1.3B	Summarization	2	N-P	GPT-Neo 1.3B
Fewshot-2ex-s-2.7B*	Summarization	2	P-N	GPT-Neo 2.7B
Fewshot-2ex-r-2.7B*	Summarization	2	N-P	GPT-Neo 2.7B
Fewshot-3ex-s-2.7B*	Summarization	3	P-N-P	GPT-Neo 2.7B
Fewshot-3ex-r-2.7B*	Summarization	3	N-P-N	GPT-Neo 2.7B
Fewshot-2ex-s-6B*	Summarization	2	P-N	GPT-Neo 6B
Fewshot-2ex-r-6B*	Summarization	2	N-P	GPT-Neo 6B
Fewshot-2ex-s-20B**	Summarization	2	P-N	GPT-NeoX 20B
Fewshot-2ex-r-20B**	Summarization	2	N-P	GPT-NeoX 20B

* These models were loaded and run using 4-bit precision in order to accommodate the available memory constraints. This quantization was achieved using the *bitsandbytes* library [24].

** These models were loaded and run on the Google Colab platform due to memory constraints [12].

Additionally, the following baseline models were created:

- Fewshot-baseline-dummy: Dummy classifier that selects the most frequent class for each user.
- Fewshot-baseline-genre: KNN classifier based on genre similarity ($k = 20$).

Chapter 6

Results

In this chapter we present the results of our experiments as mentioned in the previous chapter.

6.1 Results of Content-based Models

The following two graphs show the results of all content-based models. Figure 6.1 shows models that use only one similarity matrix, while figure 6.2 show models that use a combination of multiple similarity matrices. Note that the two baseline models ”content-genre” and ”content-genre-director” appear on both graphs for comparison purposes.

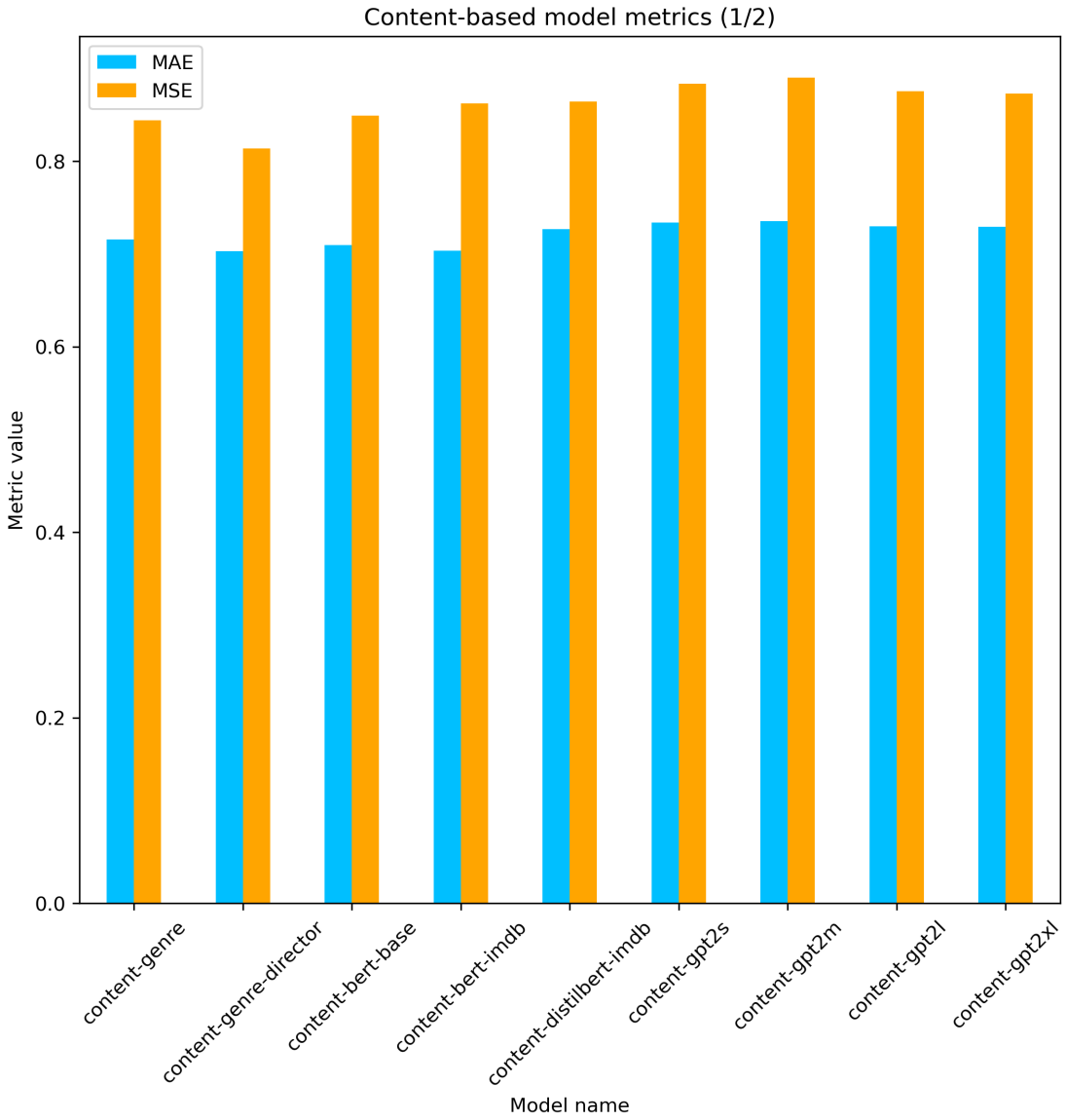


Figure 6.1: The results of the content-based models (1/2).

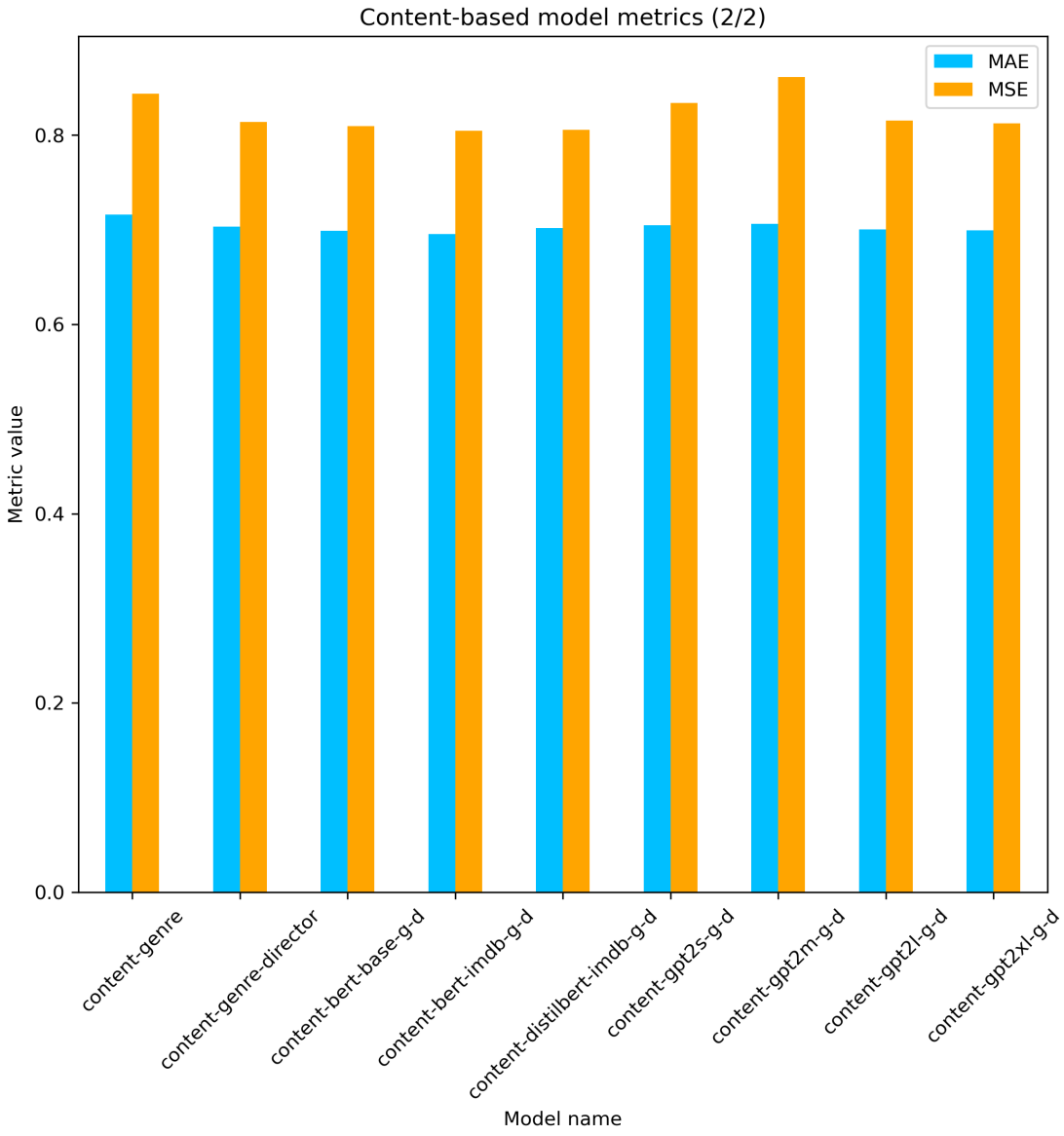


Figure 6.2: The results of the content-based models (2/2).

The baselines set by the first model ”content-genre” are 0.716 for MAE and 0.844 for MSE. The second baseline model ”content-genre-directors,” which uses additional information about the movie’s directors, achieves a MAE value of 0.703 and an MSE value of 0.814. Among all the models we evaluated, the ”content-bert-imdb-g-d” model stands out as the best performer in terms of both MAE and MSE, with values of 0.695 and 0.805 respectively. When compared to the two baseline models, this represents an improvement of 2.93% and 1.14% in MAE, and an improvement of 4.62% and 1.11% in MSE.

The better performance of "content-bert-imdb" over the standard "content-bert-base" model can be attributed to its specialized adaptation to the nuances of movie-related content. This demonstrates the value of fine-tuning pre-trained models on specialized datasets to enhance their effectiveness in domain-specific applications.

While the model "content-distilbert-imdb" has performance metrics that are comparable in terms of MSE to those of "content-bert-imdb", it achieves a distinct MAE. This divergence suggests that while DistilBERT models are generally consistent in minimizing squared errors, they might not be as effective in reducing the absolute differences across all predictions. Given that DistilBERT is a lighter, distilled version of BERT, this could reflect a trade-off between model complexity and precision in capturing the nuances of the data.

An intriguing observation emerges when considering the performance of models based on the GPT architecture. Contrary to what might be expected, increasing the size of GPT models does not necessarily translate into improved performance. This phenomenon could be attributed to several factors, including the diminishing returns of model complexity on certain tasks or the potential for larger models to overfit to the training data, thereby decreasing their generalization capabilities.

When comparing the first and the second graphs, all models show improved performance when they integrate document embeddings with additional metadata, such as a movie's director and genres. This finding indicates that a movie's synopsis should not be considered a substitute for standard movie metadata, but rather as supplementary information that contributes to a more comprehensive description of a movie.

6.2 Results of Hybrid Models

Tables 6.1, 6.2, and 6.3 show the results of the grid search done on the models "hybrid-baseline", "hybrid-doc2vec", and "hybrid-bert-base".

Table 6.1: MAE and MSE of grid search on "hybrid-baseline". Values in **bold** show the best result for each metric.

	Learning Rate		
	0.01	0.001	0.0001
Batch Size	MAE | MSE	MAE | MSE	MAE | MSE
16	0.679 | 0.778	0.649 | **0.718**	0.669 | 0.755
32	0.647 | 0.735	0.656 | 0.735	0.671 | 0.760
64	**0.645** | 0.732	0.654 | 0.736	0.674 | 0.766
128	0.654 | 0.734	0.661 | 0.744	0.678 | 0.774

Table 6.2: MAE and MSE of grid search on "hybrid-doc2vec". Values in **bold** show the best result for each metric.

	Learning Rate		
	0.01	0.001	0.0001
Batch Size	MAE | MSE	MAE | MSE	MAE | MSE
16	0.678 | 0.751	0.645 | **0.711**	0.663 | 0.750
32	0.645 | 0.720	0.649 | 0.716	0.670 | 0.758
64	0.658 | 0.718	0.653 | 0.727	0.672 | 0.763
128	**0.647** | 0.710	0.652 | 0.743	0.676 | 0.772

Table 6.3: MAE and MSE of grid search on "hybrid-bert-base". Values in **bold** show the best result for each metric.

	Learning Rate		
	0.01	0.001	0.0001
Batch Size	MAE | MSE	MAE | MSE	MAE | MSE
16	0.659 | 0.747	**0.635** | 0.701	0.656 | 0.735
32	0.673 | 0.752	0.636 | 0.698	0.662 | 0.742
64	0.662 | 0.742	0.639 | **0.696**	0.664 | 0.748
128	0.658 | 0.723	0.639 | 0.700	0.671 | 0.760

Figure 6.3 shows the results of the experiments we carried on all hybrid models.

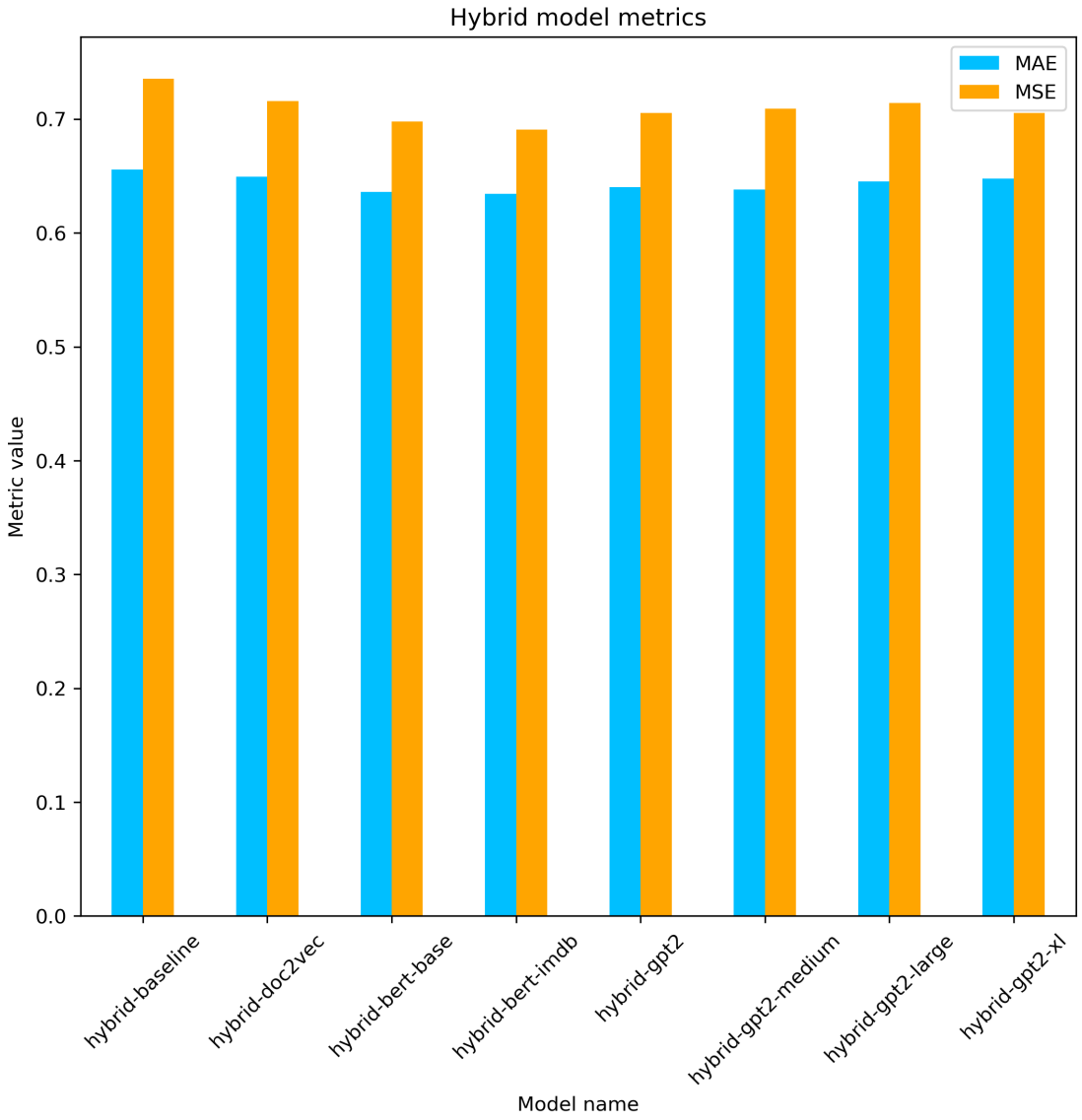


Figure 6.3: The results of the hybrid models.

The analysis of hybrid models reveals several key observations that underscore the effectiveness of combining different approaches and data representations in enhancing model performance. Notably, all hybrid models surpass the standard Neural Collaborative Filtering (NCF) implementation that was used as a baseline. This improvement indicates the value of integrating additional features and methodologies beyond just conventional collaborative filtering techniques.

The best performer among the evaluated models is ”hybrid-bert-imdb”, registering the lowest MAE and MSE, with values of 0.634 and 0.691 respectively. This represents an improvement of 2.16% and 3.76% over the ”hybrid-baseline” model. The model's

superior performance underscores the effectiveness of combining textual data with the standard collaborative filtering techniques.

The incorporation of information from Doc2Vec is observed to enhance performance, albeit modestly. This highlights the potential limitations of static embeddings in capturing the full complexity or all the relevant nuances that might be beneficial for the task of movie recommendation.

Similarly to the content-based models, "hybrid-bert-imdb", which has been adapted to the movie recommendation context by utilizing a language model that has been fine-tuned on the IMDb dataset, outperforms "hybrid-bert-base". This further supports the value of tailoring models to the specific characteristics and requirements of their application domain.

Furthermore, out findings show that simply increasing the size of GPT models does not guarantee better performance. This follows a similar observation we made on the results of the content-based models. This outcome challenges the common assumption that larger models will always perform better. It appears that there is a point of diminishing returns or that other factors, such as the quality of data or the specificity of the model to the task, play a more significant role in determining performance.

6.3 Results of Few-shot Learning Models

Figure 6.4 shows the results of the experiments we carried out in all few-shot learning models.

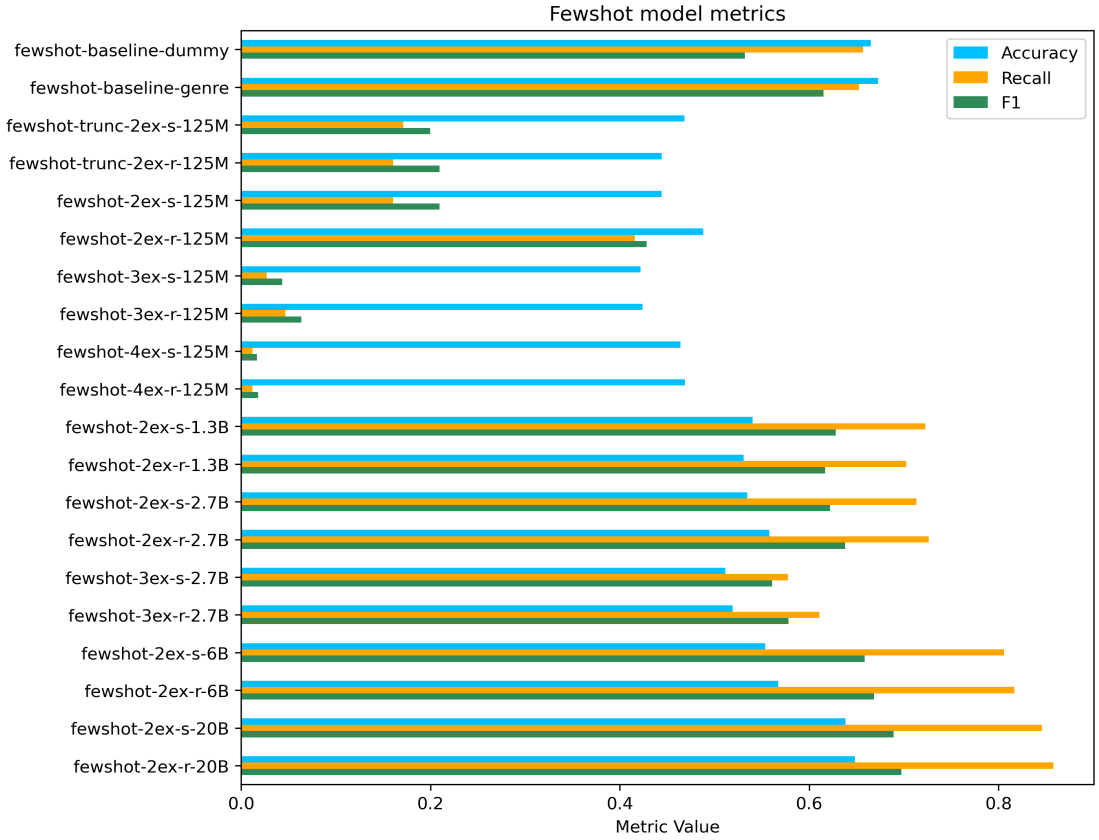


Figure 6.4: The results of the few-shot learning models.

The analysis of model performance across various configurations yields insightful observations. First of all, an increase in model size correlates with improved performance, indicating that larger models have a greater capacity to capture and generalize from the data provided. It also seems that we have not yet reached an upper limit of performance, suggesting that further scaling up model size may continue to enhance the accuracy metrics.

The optimal amount of examples in each query seems to be two. Models trained with two examples in each query consistently outperform those trained with three, and the performance deteriorates further when four examples are used. The explanation might be that additional examples may introduce noise to the model, or bias the model to a specific label.

An unexpected observation is that, among the models tested, smaller ones, specifically those with a size of 125M parameters, underperform not only in comparison to larger counterparts but also against baselines and even dummy models. This underlines the importance of model capacity in achieving competitive performance, where small models lack the necessary complexity to effectively learn from the examples.

In general, ordering examples in a ”reverse” order, where inputs are presented from

negative to positive, tends to outperform the standard order of positive to negative. This suggests that the sequence in which information is processed can impact model effectiveness, potentially due to how models prioritize or weight the first-seen data. The difference may also be attributed to the imbalance in the rating data, as movies tend to be more frequently recommended than not recommended.

Lastly, incorporating summarization as a preprocessing step yields a slight improvement in performance over simple truncation. This indicates that summarization is able to capture the essential aspects of the input. Summarization may perform even better if more sophisticated summarization models are utilized.

6.4 Overview of Results

Overall, the inclusion of plot synopsis information has proven to improve the accuracy of movie recommendation systems. Our findings underscore the value of leveraging detailed content descriptions to increase the predictive capabilities of these systems.

In content-based recommendation, it is clear that integrating plot synopsis with traditional metadata, such as genres, cast, and director information, yields the best results. The combination of different kinds of data allows for a more nuanced understanding of movies, thus facilitating more accurate and relevant recommendations.

There appears to be an inherent limit to the effectiveness of each technique, where the addition of supplemental information alone cannot fully overcome these barriers. This limitation suggests that while augmenting recommendation systems with additional data is beneficial, there are diminishing returns to such enhancements beyond a certain point. In those cases, exploring changes in the model's architecture or the techniques used may be more effective strategies for improving performance.

On a promising note, few-shot recommendation techniques have showed great potential, seemingly constrained only by the model's number of parameters or the context window size. This approach is particularly exciting as it hints at the scalability of recommendation systems with advancements in model architecture and computational capacity.

Chapter 7

Conclusion

7.1 Synopsis

In this work, we explored the potential of leveraging movie synopsis information to enhance the accuracy and relevance of movie recommendation systems. We analyzed the different types of recommendation systems and suggested various architectures in which movie plot information can be integrated. Next, we created various content-based and hybrid models that utilize modern machine learning techniques, as well as a new dataset to include textual data of movie synopses in the recommendation process. We also experimented with using movie synopses in a few-shot learning environment in order to provide more accurate recommendations. By comparing models that solely relied on traditional user-item interaction data against those enriched with synopsis-derived information, our research demonstrates a slight but measurable advantage in adding movie synopsis information to the recommendation process. The successful application of movie synopsis data also highlights the broader potential of content-based approaches in enhancing the sophistication and user relevance of recommendation systems across various domains.

The primary question of the thesis is whether movie synopsis data can improve movie recommendation systems. The experiments that we executed have demonstrated a positive impact. The integration of detailed synopsis data into the recommendation methods we examined enhanced the accuracy and relevance of the suggestions provided to users. We attribute this improvement to the ability of LLMs to deeply analyze a movie's content beyond what traditional metadata can convey. However, the improvement of recommendation quality is modest, with an upper limit that is determined by the specific technique that is used.

Regarding the secondary question of whether movie synopsis data can substitute for traditional metadata in providing accurate recommendations, the findings suggest a more nuanced outcome. In content-based models, completely substituting traditional metadata with synopses does not always yield better results. Traditional metadata, such as genre, cast, or production details, still plays a crucial role in the recommenda-

tion ecosystem. It provides essential, structured information that, when combined with supplemental textual data from movie synopses, creates a comprehensive dataset. On the other hand, few-shot learning has shown impressive results when using sufficiently large LMs. In those cases, standard metadata is not necessary to provide accurate recommendations.

7.2 Future work

The exploration of leveraging textual data through modern Large Language Models (LLMs) to enhance recommender systems is a vast and multifaceted subject, far too extensive to be comprehensively covered by a single diploma thesis. In this work, we have showed promising results that underscore the necessity for further investigation. We propose the following directions for future research:

- **Incorporate other textual information and metadata**: A possible direction for future work involves incorporating different textual information about movies, such as critic reviews or scripts from movie trailers. Furthermore, exploring the integration of all types of movie metadata, such as the cast, crew, or production details, could provide a more holistic view of each movie. Analyzing different combinations of textual data and metadata will be crucial in identifying the most effective mix for enhancing recommendation accuracy.

- **Testing different methods or algorithms**: Although our research has explored various methods in content-based and hybrid recommendation systems, they are far from the only methods that are studied in this field. Future studies should consider experimenting with other popular machine learning methods like decision trees and Bayesian network classifiers to understand their efficacy in this domain.

- **Utilizing movie scripts**: Investigating whether movie scripts can be used as an additional textual source for improving recommendation is an interesting research opportunity. Movie scripts offer a detailed narrative and dialogue context that cannot be described by a synopsis that is orders of magnitude shorter. However, the script's length necessitates the development of techniques to fit it within the specified context windows. Developing new methodologies for summarizing texts can open new pathways for leveraging more data in recommendation systems.

- **Exploring few-shot learning with larger models**: Our work showed the potential of few-shot learning in recommendation tasks. Future research could focus on testing even larger models to assess the scalability of performance improvements in recommendation accuracy. Additionally, experimenting with the model's context window could provide insights into how LLMs process textual data for recommendation purposes.

- **Testing in other domains**: Expanding the application of textual data analysis with LLMs to other domains represents an exciting frontier for future research.

For instance, analyzing the lyrics of songs could enhance music recommendation systems by tapping into the emotional and thematic content of music. Exploring these different domains would not only validate the versatility of the approach but also contribute to the development of more sophisticated and context-aware recommendation systems across various fields.

Bibliography

[1] G. Adomavicius and A. Tuzhilin. "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions." In: *IEE Transactions on Knowledge and Data Engineering* 17.6 (2005), pp. 734–749.

[2] M. Balabanović and Y. Shoham. "Fab: Content-Based, Collaborative Recommendation." In: *Communications of the ACM* 40.3 (1997), pp. 66–72.

[3] James Bennett, Stan Lanning, et al. "The Netflix prize." In: *Proceedings of KDD cup and workshop*. Vol. 2007. New York. 2007, p. 35.

[4] Daniel Billsus, Michael J Pazzani, et al. "Learning collaborative information filters." In: *ICML*. Vol. 98. 1998, pp. 46–54.

[5] Tom Brown et al. "Language models are few-shot learners." In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[6] R. Burke. "Hybrid Recommender Systems: Survey and Experiments." In: *User Modeling and User-Adapted Interaction* 12 (2002), pp. 331–370.

[7] Cyril W. Cleverdon, Jack Mills, and Michael Keen. *Factors determining the performance of indexing systems; Volume 1, Design*. Tech. rep. 1966.

[8] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[9] *Doc2vec paragraph embeddings - gensim*. Feb. 10, 2024. URL: https://radimrehurek.com/gensim/models/doc2vec.html.

[10] "Encyclopedia of Machine Learning." In: Springer, 2010, pp. 829–836.

[11] David Goldberg et al. "Using collaborative filtering to weave an information tapestry." In: *Communications of the ACM* 35.12 (1992), pp. 61–70.

[12] *Google Colaboratory*. Feb. 10, 2024. URL: https://colab.google/.

[13] David Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. 2001.

[14] Stephen P Harter. "Variations in relevance assessments and the measurement of retrieval effectiveness." In: *Journal of the American Society for Information Science* 47.1 (1996), pp. 37–49.

[15] Xiangnan He et al. "Neural collaborative filtering." In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 173–182.

[16] Jonathan L. Herlocker et al. "Evaluating collaborative filtering recommender systems." In: *ACM Transactions on Information Systems* 22.1 (2004), pp. 5–53.

[17] *Huggingface | lvwerra/distilbert-imdb*. Feb. 10, 2024. URL: https://huggingface.co/lvwerra/distilbert-imdb.

[18] *Huggingface | textattack/bert-base-uncase-imdb*. Feb. 10, 2024. URL: https://huggingface.co/textattack/bert-base-uncased-imdb.

[19] *IMDb Non-Commercial Dataset*. Jan. 5, 2024. URL: https://developer.imdb.com/non-commercial-datasets/.

[20] Dan Jurafsky and James H. Martin. *Speech and Language Processing*. Third Edition draft. 2023.

[21] Joseph A Konstan et al. "Lessons on applying automated recommender systems to information-seeking tasks." In: *AAAI*. Vol. 6. 2006, pp. 1630–1633.

[22] Yehuda Koren, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." In: *Computer* 42.8 (2009), pp. 30–37.

[23] Quoc V. Le and Tomas Mikolov. *Distributed Representations of Sentences and Documents*. 2014. arXiv: 1405.4053 [cs.CL].

[24] *Making LLMs even more accessible with bitsandbytes, 4-bit quantization and QLoRA*. Feb. 10, 2024. URL: https://huggingface.co/blog/4bit-transformers-bitsandbytes.

[25] *Matrix Factorization | Machine Learning | Google for Developers*. Feb. 10, 2024. URL: https://developers.google.com/machine-learning/recommendation/collaborative/matrix.

[26] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[27] M. Montaner, B. López, and de la Rosa J. L. "A Taxonomy of Recommender Agents on the Internet." In: *Artificial Intelligence Review* 19 (2003), pp. 285–330.

[28] *MovieLens*. Sept. 7, 2023. URL: https://grouplens.org/datasets/movielens/.

[29] *MovieLens 20M Dataset*. June 1, 2023. URL: https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset/data.

[30] Luong Vuong Nguyen, Tri-Hai Nguyen, and Jason J Jung. "Content-based collaborative filtering using word embedding: a case study on movie recommendation." In: *Proceedings of the international conference on research in adaptive and convergent systems*. 2020, pp. 96–100.

[31] M. Pazzani and D. Billsus. "Content-Based Recommendation Systems." In: *The Adaptive Web*. Ed. by P. Brusilovsky, A. Kobsa, and W. Nejdl. Vol. 4321. Lecture Notes in Computer Science. 2007, pp. 325–441.

[32] Ivens Portugal, Paulo Alencar, and Donald Cowan. "The Use of Machine Learning Algorithms in Recommender Systems: A Systematic Review." In: *Expert Systems with Applications* 97 (2018), pp. 205–227.

[33] Alec Radford et al. "Improving language understanding by generative pre-training." In: (2018).

[34] P. Resnick and H. Varian. "Recommender systems." In: *Communications of the ACM* 40.3 (1997), pp. 56–58.

[35] Paul Resnick et al. "Grouplens: An open architrecture for collaborative filtering of netnews." In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 1994, pp. 175–186.

[36] F. Ricci et al. *Recommender Systems Handbook*. Springer, 2010.

[37] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. *A Primer in BERTology: What we know about how BERT works*. 2020. arXiv: 2002.12327 [cs.CL].

[38] Gerard Salton. "Automatic Text Processing: The transformation, analysis, and retrieval of information by computer." In: *Reading: Addison-Wesley* 169 (1989), pp. 279–282.

[39] J. B. Schafer et al. "Collaborative Filtering Recommender Systems." In: *The Adaptive Web*. Ed. by P. Brusilovsky, A. Kobsa, and W. Nejdl. Vol. 4321. Lectures Notes in Computer Science. 2007, pp. 291–324.

[40] *Wikipedia Movie Plots*. June 1, 2023. URL: https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots.

[41] YY Yao. "Measuring retrieval effectiveness based on user preference of documents." In: *Journal of the American Society for Information science* 46.2 (1995), pp. 133–145.

[42] Qian Zhang, Jie Lu, and Yaochu Jin. "Artificial intelligence in recommender systems." In: *Complex & Intelligent Systems* 7 (2021), pp. 439–457.