Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής – Ανάπτυξη Λογισμικού και Τεχνητής Νοημοσύνης»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Συγκριτική Μελέτη Αλγορίθμων Κοινωνικής Σύστασης μέσω Νευρωνικών Δικτύων Γράφων.** **A Comparative Study of Social Recommendation Algrithms via Graph Neural Networks.**
Ονοματεπώνυμο Φοιτητή	**Μάρκος Σταθόπουλος**
Πατρώνυμο	**Ιωάννης**
Αριθμός Μητρώου	**ΜΠΣΠ/19045**
Επιβλέπων	**Διονύσιος Σωτηρόπουλος, Επίκουρος Καθηγητής**

Ημερομηνία Παράδοσης **Σεπτέμβριος 2021**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Διονύσιος Σωτηρόπουλος Επίκουρος Καθηγητής	Γεώργιος Τσιχριντζής Καθηγητής	Ευάγγελος Σακκόπουλος Επίκουρος Καθηγητής

ABSTRACT

This postgraduate dissertation is a comparative study of social recommendation algorithms via graph neural networks. In recent years, the importance of graph NN in solving machine learning problems have grown and a large amount of this type of architectures has been generated yearly with applications in social networks, and ecommerce. There are many classical implementations on matrix completion and collaborative filtering methods, and in recent years graphs representing the complex structure of modern databases have been used for inference. Problems arise from feature architecting and latent object embeddings, which become more prominent by the lack of topology and permutation variance of said structures. There is a push towards end to end training and use of state of the art neural models such as GAT, CGN etc. In this paper some of the best current implementations of these social recommendation architectures are tested and compared together, using common opensource datasets, that contain users items and interactions in graph form.

Keywords: GNN, CGN, RecSYS, Graph Neural Networks, Social Recommendation

ΠΕΡΙΛΗΨΗ

Η παρούσα μεταπτυχιακή διατριβή είναι μια συγκριτική μελέτη αλγορίθμων κοινωνικής σύστασης μέσω νευρωνικών δικτύων. Τα τελευταία χρόνια, η σημασία των νευρωνικών δικτύων Γράφων στην επίλυση προβλημάτων μηχανικής μάθησης έχει αυξηθεί και μεγάλο μέρος αυτού του τύπου αρχιτεκτονικών δημιουργείται ετησίως, με εφαρμογές σε κοινωνικά δίκτυα και ηλεκτρονικό εμπόριο. Υπάρχουν πολλές κλασικές εφαρμογές για την ολοκλήρωση της μήτρας και τις μεθόδους συνεργατικού φιλτραρίσματος, και τα τελευταία χρόνια έχουν χρησιμοποιηθεί για συστήματα συστάσεων, γραφήματα που αντιπροσωπεύουν τη σύνθετη δομή των σύγχρονων βάσεων δεδομένων. Αρκετά προβλήματα προκύπτουν από την αναπαράσταση των χαρακτηριστικών και τις λανθάνουσες ενσωματώσεις αντικειμένων, οι οποίες γίνονται πιο εμφανείς από την έλλειψη τοπολογίας των εν λόγω δομών. Επίσης υπάρχει ώθηση για εκπαίδευση των δικτύων από άκρη σε άκρη χωρίς χειρωνακτική προσαρμογή της εισόδου όπως και χρήση τεχνικών GAT, CGN κλπ. Σε αυτήν την εργασία δοκιμάζονται και συγκρίνονται μερικές από τις καλύτερες τρέχουσες εφαρμογές αυτών των αρχιτεκτονικών κοινωνικής σύστασης, χρησιμοποιώντας κοινά σύνολα δεδομένων ανοιχτού κώδικα, που περιέχουν στοιχεία χρηστών και αλληλεπιδράσεις σε μορφή γραφήματος.

Λέξεις -κλειδιά: Νευρωνικά Δίκτυα Γράφων, GNN, CGN, RecSYS, Graph Neural Networks, Social Recommendation

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ τον επιβλέπων καθηγητή κ.Σωτηρόπουλο για την βοήθεια και υποστήριξη, όπως και την οικογένεια μου για την έμπνευση και συμπαράσταση τους.

Πίνακας Περιεχομένων

Πρόλογος 7
Εισαγωγή 8
ΚΕΦΑΛΑΙΟ 1ο : Κλασικές αρχιτεκτονικές συστημάτων συστάσεων. 9
1.1 Συστάσεις βάσει συνεδρίας: 9
1.2 Συστάσεις σχετικές με το πλαίσιο – context aware recsys: 9
1.3 Συστάσεις βάση περιεχομένου (Content-based) 10
Συστάσεις με Φιλτράρισμα περιεχομένου 10
ΚΕΦΑΛΑΙΟ 2: Παραδοσιακά μοντέλα συστημάτων συστάσεων 12
2.1 Φιλτράρισμα βάσει περιεχομένου - Content based filtering (CF) 12
2.2 Μέθοδοι γειτνίασης - εκπαίδευσης αναπαραστάσεων (representation learning) ... 13
Μοντέλα λανθανόντων παραγόντων (Latent Factor models) 13
2.2.1 Παραγοντοποιηση μήτρας - Matrix Factorization 14
2.2.2 Συνεργατικό φιλτράρισμα - collaborative filtering 14
2.2.3 Μέθοδοι Συνεργατικού φιλτραρίσματος με χαρακτηριστικά γειτονιάς - CF Neighborhood methods 15
ΚΕΦΑΛΑΙΟ 3: Μοντέλα Βαθιάς μάθησης, συνυπολογίζοντας δευτερεύουσα πληροφορία 16
3.1 Συνελικτική παραγοντοποίηση μήτρας – convolutional matrix factorization 17
3.1.1 Αρχιτεκτονική CNN των Συνελικτικών δικτύων παραγοντοποίησης μήτρας (ConvMF) 17
3.1.2 Συνεργατικό Φιλτράρισμα με χρήση νευρωνικού δικτύου - Neural collaborative filtering 18
3.1.3 Το μοντέλο AUTOREC 19
ΚΕΦΑΛΑΙΟ 4: Αρχιτεκτονικές συστημάτων σειριακών συστάσεων και ανά συνεδρία - Sequential - Session based recommendation systems 21
4.1 Εισαγωγή 21
4.2 Παραγοντοποίηση αλυσίδων Markov για συστάσεις ανά συνεδρία. 22
4.3 Σειριακές Συστάσεις με Αναδρομικά Νευρωνικά Δίκτυα (RNNs) 23
4.4 Transformers σε μοντέλα αλληλεπιδράσεων με τεχνικές αυτοεκτίμησης (self attention) 25
4.5 Τα δυνατά σημεία των συστημάτων συστάσεων με αλγόριθμους που στηρίζονται σε βαθιά νευρωνικά δίκτυα. 25
4.5 Πιθανοί περιορισμοί των πολυεπίπεδων νευρωνικών δικτύων - Deep NN 26
ΚΕΦΑΛΑΙΟ 5: Αρχιτεκτονικές Νευρωνικών Δικτύων Γράφων 28
5.1 Βασικά χαρακτηριστικά των νευρωνικών δικτύων γράφων. 28
5.1.1 Τα δυνατά σημεία των συστημάτων συστάσεων με αλγόριθμους που στηρίζονται σε βαθιά νευρωνικά δίκτυα. 28

5.1.2 Πιθανοί περιορισμοί των πολυεπίπεδων νευρωνικών δικτύων - Deep NN 29
5.2 Διαφορετικές υποκατηγορίες GNN 30
5.2.1 GCN (Kipf & Welling, ICLR 2017) 33
5.3 Graph Attention Networks (GATs) 34
5.4 Gated Graph Networks 35
5.4.1 Αρχιτεκτονική GraphRec 36
ΚΕΦΑΛΑΙΟ 6. Συστήματα συστάσεων με Γράφους γνώσης - Knowledge Graph-based Recommendations 41
6.1 Γνωσιακοί Γράφοι 41
6.2 Επίπεδο ενσωμάτωσης Γνωσιακών Γράφων (KGE) 43
6.3 KGCN 43
6.3.1 Αρχιτεκτονική DGREC 45
6.3.2 Δίκτυο προσοχής σε Γράφο - Graph-Attention Network 47
Dataset της υλοποίησης 48
6.4 Dual Graph Attention Networks (DANSER) 50
6.4.1 Η αρχιτεκτονική του DANSER 50
6.4.2 Επίπεδο ενσωμάτωσης χαρακτηριστικών 50
6.4.3 Ομοφυλία και κοινωνική επιρροή 52
6.5 DIFFNET ++ 52
6.5.1 Επίπεδο Ενσωμάτωσης 53
6.5.2 Επίπεδο σύντηξης. 53
ΚΕΦΑΛΑΙΟ 7. Μετρικές απόδοσης - Performance Measures 54
ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ 56
ΒΙΒΛΙΟΓΡΑΦΙΑ 57
ΠΑΡΑΡΤΗΜΑ – ΕΚΤΕΛΕΣΗ ΥΛΟΠΟΙΗΣΕΩΝ 59
DANSER-WWW-19-master dataset Epinions σε pickle 59
GraphRec 62
Graphrec pytorch 64
KGCN LS 67
DIFFNET++ 70
LightGCN 74
KGCN 77

Πρόλογος

Σκοπός της διπλωματικής μεταπτυχιακής εργασίας είναι η διερεύνηση μεθόδων νευρωνικών δικτύων πάνω σε Γράφους για την λύση του προβλήματος συστάσεων, σε βάσεις μεγέθους κοινωνικών δικτύων.

Η διερεύνηση αυτή χωρίζεται σε δύο μέρη: (ι) σε μια βιβλιογραφική έρευνα πάνω στις μεθόδους ταξινόμησης γράφων από τα πεδία των νευρωνικών δικτύων, καθώς και των αναπαραστάσεων δεδομένων σε μορφή γράφων, (ιι) στην υλοποίηση και εφαρμογή των μεθόδων αυτών πάνω σε προβλήματα ταξινόμησης γνωστών σετ αναφοράς από το πεδίο των συστάσεων ταινιών και φυσικών προϊόντων.

Με την ταχεία ανάπτυξη του Διαδικτύου, ο όγκος των δεδομένα έχει αυξηθεί εκθετικά. Λόγω της υπερφόρτωσης του πληροφορίες, είναι δύσκολο για τους χρήστες να διαλέξουν ποια ενδιαφέροντα τους μεταξύ ενός μεγάλου αριθμού επιλογών. Για να βελτιώσετε το Εμπειρία χρήστη, Συστήματα σύστασης έχουν εφαρμοστεί για σενάρια όπως μουσική πρόταση [1], ταινία σύσταση [2] και διαδικτυακές αγορές [3]. Ο αλγόριθμος προτάσεων είναι το βασικό στοιχείο του Συστήματος σύστασης

“*Μια μεγάλη αλλαγή συμβαίνει στη συνδεδεμένη κοινωνία μας: αφήνουμε την εποχή της πληροφορίας και μπαίνουμε στην εποχή της σύστασης. Σήμερα έχουμε υπερπληθόρα άμεσης πληροφόρησης. Η συλλογή πληροφοριών δεν είναι πλέον το ζήτημα - η λήψη έξυπνων αποφάσεων με βάση τις πληροφορίες είναι τώρα το ζητούμενο... Επομένως, οι συστάσεις λειτουργούν ως η χαρτογράφηση της διαδρομής μέσα στην θάλασσα πληροφοριών οδηγώντας μας στην σωστή ή "αρκετά σωστή" απάντηση.”* Adam Richardson

Εισαγωγή

Ο όρος σύστημα συστάσεων (RS) αναφέρεται σε όλα τα εργαλεία και τις τεχνικές λογισμικού που, χρησιμοποιώντας τις γνώσεις που μπορούν να συγκεντρώσουν για τους εν λόγω χρήστες και αντικείμενα, και να παρέχουν προτάσεις για νέα αντικείμενα που πιθανότατα ενδιαφέρουν έναν συγκεκριμένο χρήστη [Ricci et al. , 2015]. Οι προτάσεις μπορεί να σχετίζονται με διάφορες διαδικασίες λήψης αποφάσεων, όπως ποια προϊόντα να αγοράσουν, ποια μουσική να ακούσουν ή ποιες ταινίες θα παρακολουθήσουν. Αντικείμενα είναι ο όρος που χρησιμοποιείται για να προσδιορίζει το στοιχείο που συνιστά το σύστημα στους χρήστες. Ένα σύστημα συστάσεων συνήθως επικεντρώνεται σε μια κατηγορία αντικειμένων, όπως βιβλία, άρθρα ή ξενοδοχεία για κράτηση. Οι μέθοδοι και οι τεχνικές που χρησιμοποιούνται για τη δημιουργία των συστάσεων προσαρμόζονται σε συγκεκριμένο τύπο αντικειμένων.

Σκοπός των συστημάτων σύστασης είναι να βοηθήσει τις εταιρείες να πουλήσουν περισσότερα είδη, όπως επίσης να παρέχουν μια «προσαρμοσμένη» εμπειρία, βοηθώντας τους ανθρώπους να βρουν αυτό που ψάχνουν ή σχετικό με τα ενδιαφέροντά τους πιο γρήγορα. Αποτέλεσμα είναι ότι η βελτίωση της ικανοποίηση των χρηστών. Αναλυτικότερα οι επιμέρους στόχοι είναι οι παρακάτω:

Αύξηση του αριθμού των αντικειμένων ανά καλάθι.
Η πιο σημαντική λειτουργία για ενα RS: να βοηθήσει έναν πάροχο να πουλήσει περισσότερα αντικείμενα από ότι αν δεν έδινε καμία πρόταση. Με την πληθώρα διαθέσιμων στοιχείων, οι χρήστες συχνά δεν μπορούν να βρουν αυτό που ψάχνουν, και ως αποτέλεσμα οι συνεδρίες τους δεν ολοκληρώνονται με κάποια αγορά. Τα RS προσφέρουν βοήθεια για να καλυφθούν οι ανάγκες και οι προσδοκίες των χρηστών.

Αύξηση όγκου πωλήσεων.
Ένα RS επιτρέπει στον χρήστη να επιλέξει αντικείμενα που μπορεί να είναι δύσκολο να βρεθούν χωρίς στοχευμένη σύσταση. Με την παροχή εξατομικευμένων προτάσεων, ο πωλητής μειώνει δραματικά το διαφημιστικό ρίσκο προωθώντας στοιχεία που δεν ταιριάζουν στα γούστα του χρήστη. Προτείνοντας μη δημοφιλή στοιχεία σε χρήστες, το RS μπορεί να βελτιώσει την ποιότητα της συνολικής εμπειρίας τους και να επιτρέψει την ανακάλυψη και την ανάδειξη νέων δημοφιλών αντικειμένων.

Αύξηση της ικανοποίησης χρήστη.
Εάν ο χρήστης βρίσκει τις προτάσεις ενδιαφέρουσες, σχετικές και δικαιολογημένες από ένα καλά σχεδιασμένο μπροστινό UI, θα αυξήσει την υποκειμενική αξιολόγηση του συστήματος και θα αυξήσει την πιθανότητα να γίνει σταθερός πελάτης. Ως εκ τούτου αυτές οι τεχνικές αυξάνουν τη δημοφιλία του συστήματος, τα διαθέσιμα δεδομένα για το μοντέλο συστάσεων, την ποιότητα των συστάσεων και τέλος την ικανοποίηση των χρηστών.

Αφοσίωση των χρηστών.
Ιστότοποι και πελατοκεντρικές εφαρμογές εκτιμούν και ενθαρρύνουν την αφοσίωση αναγνωρίζοντας τους πελάτες που επιστρέφουν και τους αντιμετωπίζουν ως αξιόλογους επισκέπτες. Η παρακολούθηση επιστρεφόμενων χρηστών είναι μια βασική απαίτηση για τα RS (με αρκετές εξαιρέσεις που συζητούνται αργότερα) επειδή οι αλγόριθμοι που χρησιμοποιούνται αξιοποιούν τις πληροφορίες που αποκτήθηκαν από τους χρήστες κατά τη διάρκεια προηγούμενων αλληλεπιδράσεων, όπως η βαθμολογία τους για στοιχεία για την υποβολή προτάσεων κατά την επόμενη επίσκεψη του χρήστη. Κατά συνέπεια, όσο πιο συχνά ένας χρήστης αλληλεπιδρά με τον ιστότοπο ή την εφαρμογή, τόσο πιο αξιόπιστο και αναλυτικό γίνεται το μοντέλο του χρήστη, των προτιμήσεών του και αυξάνεται η αποτελεσματικότητα της παραγωγής του συστάσεων.

ΚΕΦΑΛΑΙΟ 1ο : Κλασικές αρχιτεκτονικές συστημάτων συστάσεων.

Η δημιουργία μοντέλων χρηστών και αντικειμένων είναι στην καρδία κάθε συστήματος προτάσεων. Ωστόσο, ο τρόπος με τον οποίο συλλέγονται και αξιοποιούνται αυτές οι πληροφορίες εξαρτάται από τη αρχιτεκτονική του κάθε συστήματος και τους εκάστοτε αλγόριθμους μηχανικής μάθησης. Σύμφωνα με τον τύπο των πληροφοριών που χρησιμοποιούνται για την κατασκευή των μοντέλων και την προσέγγιση που χρησιμοποιείται για την πρόβλεψη των ενδιαφερόντων των χρηστών και την παραγωγή προβλέψεων, μπορούν να εφαρμοστούν διαφορετικοί τύποι συστημάτων σύστασης. Οι τέσσερις προσεγγίσεις που θα εξετάσουμε είναι:

Συστάσεις βάσει περιεχομένου – content based:
Ο αλγόριθμος προτάσεων αξιοποιεί τις περιγραφές αντικειμένων και το προφίλ χρηστών που αποδίδουν βάρη σε διαφορετικά χαρακτηριστικά. Στην συνέχεια εκπαιδεύεται να βρίσκει αντικείμενα παρόμοια σε περιεχόμενο, με αυτά που άρεσαν (αλληλοεπίδρασε) στο χρήστη στο παρελθόν. Ένα τυπικό παράδειγμα είναι ένας σύμβουλος ειδήσεων που συγκρίνει τα άρθρα που έχει διαβάσει ο χρήστης προηγουμένως με τα πιο πρόσφατα με βάση το περιεχόμενο.

Συνεργατικό φιλτράρισμα – collaborative filtrering:
Η βασική ιδέα πίσω από το συνεργατικό φιλτράρισμα είναι ότι εάν οι χρήστες είχαν τα ίδια ενδιαφέροντα στο παρελθόν - για παράδειγμα, εάν αγόρασαν παρόμοια βιβλία ή παρακολούθησαν παρόμοιες ταινίες - θα έχουν την ίδια συμπεριφορά και στο μέλλον. Τέτοιο σύστημα διαμορφώνει την αρχική σελίδα εμπορικών ιστό τόπων όπως το Amazon με σχετικά προϊόντα.

1.1 Συστάσεις βάσει συνεδρίας:

Η μηχανή συστάσεων κάνει προβλέψεις με βάση τα δεδομένα της προηγουμένης περιόδου σύνδεσης, με βάση τα κλικ και τις περιγραφές των στοιχείων. Μια τέτοια αρχιτεκτονική είναι χρήσιμη όταν δεν είναι διαθέσιμα προφίλ χρηστών και ιστορικό προηγούμενων δραστηριοτήτων. Χρησιμοποιεί πληροφορίες σχετικά με τις τρέχουσες αλληλεπιδράσεις των χρηστών και τις αντιστοιχεί με προηγούμενες αλληλεπιδράσεις άλλων χρηστών. Μια σχετική εφαρμογή είναι ένας ταξιδιωτικός ιστότοπος που παρέχει λεπτομέρειες για ξενοδοχεία, οικίες και διαμερίσματα, όπου οι χρήστες συχνά δεν ταυτοποιούνται παρα μόνο στο τέλος της διαδικασίας, όταν ολοκληρώνεται η κράτηση. Σε αυτές και άλλες περιπτώσεις, δεν υπάρχει διαθέσιμο ιστορικό για τον χρήστη.

1.2 Συστάσεις σχετικές με το πλαίσιο – context aware recsys:

Η μηχανή προτάσεων δημιουργεί σχετικές προτάσεις προσαρμόζοντάς τις στο συγκεκριμένο πλαίσιο του χρήστη [Adomavicius et al., 2011]. Οι πληροφορίες με βάση τα συμφραζόμενα θα μπορούσαν να περιλαμβάνουν τοποθεσία, ώρα ή εταιρεία (με τον χρήστη του χρήστη). Για παράδειγμα, πολλές εφαρμογές για κινητά χρησιμοποιούν δευτερεύον πληροφορίες από τα meta-δεδομένα (τοποθεσία, καιρός, ώρα και ούτω καθεξής) για να βελτιώσουν τις συστάσεις που παρέχονται στους χρήστες.

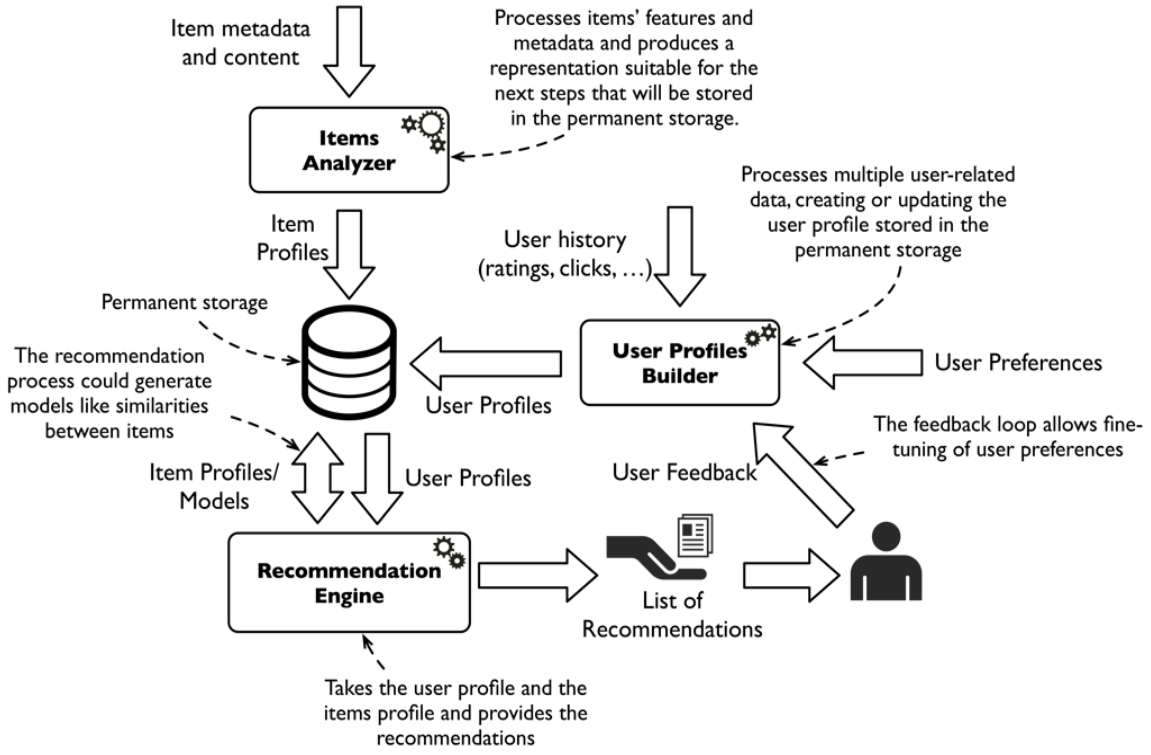
Κάθε μία από τις παραπάνω προσεγγίσεις έχει πλεονεκτήματα και μειονεκτήματα, τα οποία επισημαίνονται στις επόμενες ενότητες. Τα υβριδικά συστήματα συστάσεων συνδυάζουν διαφορετικές προσεγγίσεις για να ξεπεράσουν κάποια από τα ζητήματα και παρέχουν καλύτερες συστάσεις στους τελικούς χρήστες.

1.3 Συστάσεις βάση περιεχομένου (Content-based)

Συστάσεις με Φιλτράρισμα περιεχομένου

Σε σύγκριση με τα μοντέλα που βασίζονται σε CF, το οποίο μαθαίνει την αναπαράσταση του χρήστη και του αντικειμένου από τα δεδομένα αλληλεπίδρασης μεταξύ χρηστών-αντικειμένων, οι μέθοδοι που βασίζονται στο περιεχόμενο αναπαριστούν τον χρήστη και το αντικείμενο από το περιεχόμενο των αντικειμένων. Η θεωρία του φιλτραρίσματος βάσει περιεχομένου στηρίζεται στο στην υπόθεση ότι οι χρήστες μπορεί να ενδιαφέρονται για αντικείμενα παρόμοια με τα παρελθόντα αντικείμενα που έχουν αλληλοεπιδράσει. Η αναπαράσταση των αντικειμένων γίνεται εξάγοντας χαρακτηριστικά από τις συνοδευτικές πληροφορίες τους, συμπεριλαμβανομένων κειμένων, εικόνων κ.λπ., ενώ η αναπαράσταση του χρήστη βασίζεται στις ιδιότητες των αντικειμένων που έχουν δει. Η διαδικασία σύγκρισης των υποψηφίων αντικειμένων με το προφίλ χρήστη είναι ουσιαστικά ένα ταίριασμα με τα προηγούμενα αρχεία του χρήστη. Επομένως, αυτή η προσέγγιση τείνει να προτείνει στοιχεία που είναι παρόμοια με στοιχεία που άρεσαν στον χρήστη στο παρελθόν.

Το παρακάτω σχήμα επισημαίνει την αρχιτεκτονική υψηλού επιπέδου συστήματος συστάσεων βάσει περιεχομένου CF



Εικόνα 1.1: Ο μηχανισμός σύστασης βάσει περιεχομένου.
Πηγή: Semantics-Aware Content-Based Recommender Systems

Το παραπάνω διάγραμμα αποσυνθέτει τον μηχανισμό σύστασης σε τρία κύρια συστατικά:

Ανάλυση στοιχείων

Ο κύριος σκοπός αυτού του στοιχείου είναι η ανάλυση στοιχείων, η εξαγωγή ή να προσδιορίσετε σχετικά χαρακτηριστικά και να αναπαραστήσετε τα στοιχεία σε μορφή κατάλληλη για το επόμενα βήματα επεξεργασίας. Λαμβάνει ως είσοδο το περιεχόμενο του στοιχείου (όπως

το περιεχόμενο του ένα βιβλίο ή μια περιγραφή προϊόντος) και μεταπληροφόρηση (όπως ένα βιβλίο συγγραφέα, τους ηθοποιούς σε μια ταινία ή είδη ταινιών) από μία ή περισσότερες πληροφορίες πηγές και τα μετατρέπει σε μοντέλο στοιχείων που χρησιμοποιείται αργότερα για παροχή σύστασης. Στην παραπάνω προσέγγιση που παράγονται μοντέλα γράφων, διαφορετικών τύπων. Αυτή η αναπαράσταση γράφου χρησιμοποιείται για να τροφοδοτήσει τη διαδικασία σύστασης.

Δημιουργία προφίλ χρηστών

Αυτή η διαδικασία συλλέγει δεδομένα αντιπροσωπευτικά των προτιμήσεων των χρηστών και συνάγει τα προφίλ χρηστών. Αυτά περιλαμβάνουν πραγματικές προτιμήσεις που έχουν συγκεντρωθεί ρωτώντας τους χρήστες για τα ενδιαφέροντά τους ή από τα έμμεσα δεδομένα που συλλέγονται από παρατήρηση και αποθήκευση της συμπεριφοράς των χρηστών. Το αποτέλεσμα είναι ένα μοντέλο σε μορφή γράφου που αντιπροσωπεύει το ενδιαφέρον του χρήστη για κάποιο συγκεκριμένο στοιχείο, χαρακτηριστικό στοιχείου, ή και τα δύο. Στην αρχιτεκτονική του του παραπάνω σχήματος τα προφίλ στοιχείων (δημιουργήθηκαν κατά τη διάρκεια του στάδιο ανάλυσης στοιχείων) και τα προφίλ χρηστών (που δημιουργήθηκαν σε αυτό το στάδιο) συγκλίνουν σε μια κοινή βάση δεδομένων. Επιπλέον, επειδή και οι δύο διαδικασίες επιστρέφουν ένα Γράφο, οι έξοδοι μπορούν να συνδυαστούν σε ένα ενιαίο, συνδεδεμένο και εύκολο στην πρόσβαση μοντέλο που θα χρησιμοποιηθεί ως είσοδος της επόμενης φάσης.

Σύστημα σύστασης

Αυτή η ενότητα εκμεταλλεύεται τα προφίλ χρηστών και τις αναπαραστάσεις στοιχείων για να προτείνει σχετικά στοιχεία αντιστοιχίζοντας τα ενδιαφέροντα των χρηστών με τα χαρακτηριστικά των στοιχείων.

Σε αυτή τη φάση, παράγεται ένα μοντέλο πρόβλεψης που χρησιμοποιείται για την πρόβλεψη για τις βαθμολογίες συνάφειας για κάθε στοιχείο. Αυτή η βαθμολογία χρησιμοποιείται για την κατάταξη και ιεράρχηση των στοιχείων που προτείνονται στον χρήστη. Ορισμένοι αλγόριθμοι προτάσεων υπολογίζουν αρχικά σχετικές τιμές, για παράδειγμα τις ομοιότητες μεταξύ στοιχείων, για να γίνει πιο γρήγορη η φάση πρόβλεψης.

Υβριδικές μέθοδοι.

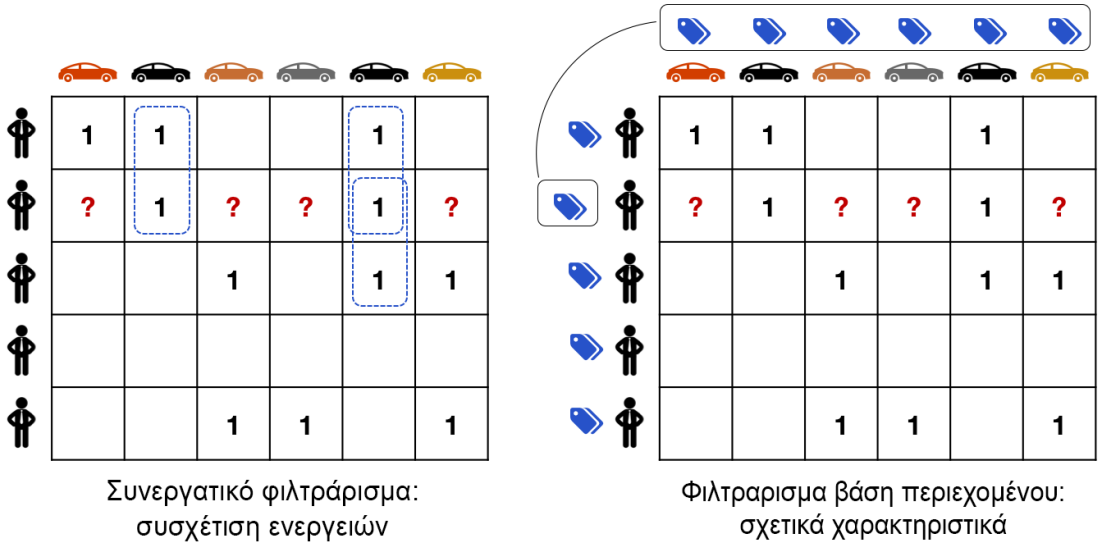
Η υβριδικές μέθοδοι αξιοποιούν πολλαπλούς αλγόριθμους προτάσεων με στόχο να ξεπεραστούν οι περιορισμοί της χρήσης μιας μόνο μεθόδου. Ένα σημαντικό μειονέκτημα των συστημάτων που βασίζονται σε CF είναι η έλλειψη δεδομένων αλληλεπίδρασης χρήστη-στοιχείου, γεγονός που καθιστά δύσκολη την εύρεση παρόμοιων στοιχείων ή χρηστών από την άποψη της αλληλεπίδρασης. Μια ειδική περίπτωση για αυτό το ζήτημα είναι το πρόβλημα εκκίνησης – cold start, που σημαίνει ότι η συστάσεις για νέους χρήστες ή στοιχεία είναι δύσκολη, καθώς η ομοιότητα χρήστη-χρήστη και στοιχείου δεν μπορεί να προσδιοριστεί χωρίς υπάρχουσες εγγραφές αλληλεπίδρασης. Ενσωματώνοντας τις πληροφορίες των χρηστών και των αντικειμένων, μπορεί να επιτευχθεί καλύτερη απόδοση προτάσεων. Μερικές πληροφορίες που χρησιμοποιούνται συνήθως από πλευράς αντικειμένων περιλαμβάνουν χαρακτηριστικά αντικειμένων όπως επωνυμία, κατηγορία, πληροφορίες πολυμέσων των στοιχείων, όπως περιγραφή με κειμένου, χαρακτηριστικά εικόνας, ήχου και κριτικές αντικειμένων. Οι συχνές επιλογές για πληροφορίες από πλευράς χρήστη περιλαμβάνουν τις δημογραφικές πληροφορίες του χρήστη, όπως εργασία, το φύλου, τα χόμπι και δίκτυο χρηστών. Σε αυτήν την κατηγορία ανήκουν, τα συστήματα σύστασης που βασίζονται σε Γράφους γνώσεων KG αξιοποιούν το KG ως πηγή πληροφοριών, συνδυάζοντας την τεχνική που βασίζεται σε CF για καλύτερο αποτέλεσμα.

ΚΕΦΑΛΑΙΟ 2: Παραδοσιακά μοντέλα συστημάτων συστάσεων

2.1 Φιλτράρισμα βάσει περιεχομένου - Content based filtering (CF)

Το φιλτράρισμα βάσει περιεχομένου χρησιμοποιεί τα χαρακτηριστικά των στοιχείων για να προτείνει στοιχεία παρόμοια με αυτά που αρέσουν στον χρήστη, με βάση τις προηγούμενες ενέργειες ή τις απαντήσεις του.

Στο παρακάτω σχήμα βλέπουμε μια μήτρα χαρακτηριστικών όπου κάθε γραμμή αντιπροσωπεύει μια εφαρμογή και κάθε στήλη αντιπροσωπεύει ένα χαρακτηριστικό. Οι στήλες θα μπορούσαν να περιλαμβάνουν κατηγορίες (όπως Εκπαίδευση, Περιστασιακό, Υγεία), τον εκδότη της εφαρμογής και πολλές άλλες. Για απλοποίηση, οι τιμές είναι δυαδικές: μια μη μηδενική τιμή σημαίνει ότι η εφαρμογή διαθέτει αυτήν τη δυνατότητα.



Εικόνα 2.1: Σύγκριση κλασικών συστημάτων συστάσεων
Πηγή: https://ebaytech.berlin/deep-learning-for-recommender-systems-48c786a20e1a

Παρουσιάζουμε στον χρήστη το ίδιο πλαίσιο λειτουργιών ώστε να επιλέξει κάποιες από αυτές. Για παράδειγμα, ένας χρήστης επιλέγει "Εφαρμογές ψυχαγωγίας" στο προφίλ του. Άλλες κατηγορίες μπορεί να είναι σιωπηρές, με βάση τις εφαρμογές που έχουν εγκαταστήσει προηγουμένως.

Το μοντέλο θα πρέπει να προτείνει στοιχεία κατάλληλα για αυτόν τον χρήστη. Έτσι πρέπει πρώτα να υπολογίσουμε τον βαθμό ομοιότητας (για παράδειγμα, το γινόμενο). Στη συνέχεια το σύστημα βαθμολογεί κάθε υποψήφιο στοιχείο σύμφωνα με αυτήν τη μέτρηση. Οι συστάσεις είναι συγκεκριμένες για αυτόν τον χρήστη, καθώς το μοντέλο δεν χρησιμοποιεί πληροφορίες για άλλους χρήστες.

Μειονεκτήματα

Οι συστάσεις από σύστημα φιλτραρίσματος με βάση το περιεχόμενο είναι περιορισμένης εμβέλειας και απαιτούν στοιχεία και χαρακτηριστικά που να είναι αναγνωρίσιμα από το σύστημα. Δεν μπορεί να φιλτράρει στοιχεία με αξιολόγηση της ποιότητας, του στυλ ή της υποκειμενική άποψης επειδή δεν μπορεί να αξιολογήσει την εμπειρία των χρηστών και επίσης υπάρχει απουσία εξατομικευμένων συστάσεων.

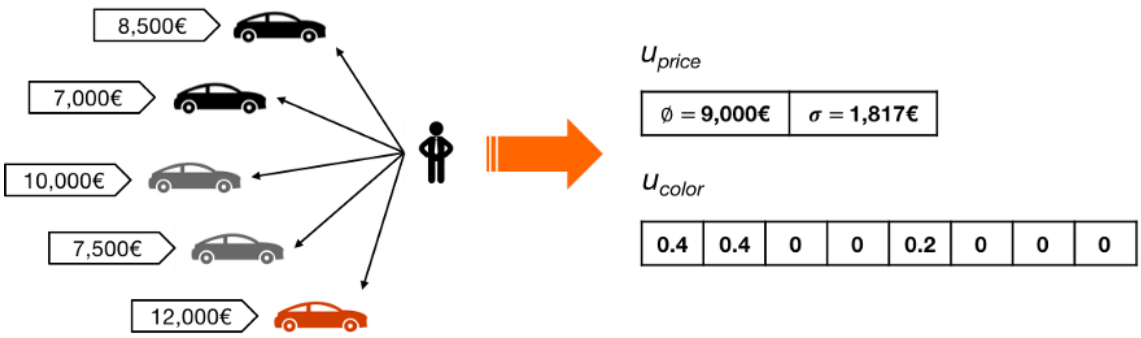
Στο σύστημα φιλτραρίσματος περιεχομένου δεν υπάρχει Serendipity - είναι η ικανότητα του σύστημα για να δώσει ένα στοιχείο που να προκαλέσει έκπληξη και ενδιαφέρον σε έναν χρήστη, αλλά μόνο στοιχεία που ενδεχομένως προβλέπονται ως ελκυστικά από τον χρήστη. Για παράδειγμα, εάν έχει προταθεί ένα βιβλίο του ίδιου συγγραφέα, ο χρήστης πιθανότατα γνωρίζει ήδη για το βιβλίο και, ως εκ τούτου, δεν ενθουσιάζεται από την σύσταση.

Το σύστημα φιλτραρίσματος βάσει περιεχομένου πάσχει από Συνωνυμίες. Αν υπάρχουν δύο λέξεις που γράφονται διαφορετικά αλλά έχουν την ίδια έννοια, το φιλτράρισμα βάσει περιεχομένου θα τα αναγνωρίσει ως δύο ανεξάρτητες λέξεις και δεν θα βρει ομοιότητες σε χαρακτηριστικά.

2.2 Μέθοδοι γειτνίασης - εκπαίδευσης αναπαραστάσεων (representation learning)

Μοντέλα λανθανόντων παραγόντων (Latent Factor models)

Τα συστήματα συστάσεων βάσει περιεχομένου που είδαμε προηγούμενα ταίριαζαν αντικείμενα με βασικά συνήθειες χρηστών με βάση την συχνότητα εμφάνισης. Αυτή η προσέγγιση είχε νόημα, αλλά συναντήσαμε τη δυσκολία συλλογής πληροφοριών από τους χρήστες. Για παράδειγμα, πολλά παιδιά αντιστοιχούν διαισθητικά σε μεγαλύτερο αριθμό υπνοδωματίων σε ένα σπίτι, αλλά ρωτώντας πόσα παιδιά έχει κάποιος, μαζί με πολλές άλλες απαραίτητές για τη δημιουργία ενός πλήρους προφίλ χρήστη ερωτήσεις, είναι ενοχλητικό. Έτσι οι εφαρμογές διαδικτύου βασίζονται κυρίως στα δεδομένα που λαμβάνουν μέσω αλληλεπιδράσεων μεταξύ της υπηρεσίας τους και των χρηστών τους.



Εικόνα 2.2: Διανυσματική αναπαράσταση χαρακτηριστικών Uprice για την τιμή και Ucolor το χρώμα του στοιχείου
Πηγή: https://ebaytech.berlin/deep-learning-for-recommender-systems-48c786a20e1a

Τα μοντέλα που βασίζονται στα εν λόγω δεδομένα ονομάζονται μοντέλα λανθανουσών παραγόντων Ένας λανθάνων παράγοντας είναι μία από τις ιδιότητες σε ένα προφίλ χρήστη χωρίς τη ρητή γνώση αυτής της ιδιότητας. Οι λανθάνοντες παράγοντες είναι οι κινητήρια δύναμη πίσω από τις αποφάσεις που λαμβάνουν οι χρήστες. Για παράδειγμα, ακόμη και αν δεν γνωρίζουμε με βεβαιότητα ότι ένας αγοραστής σπιτιού έχει παιδιά, μπορούμε να το συμπεράνουμε, αν βλέπει σπίτια με πολλά υπνοδωμάτια. Έτσι, εάν δύο αγοραστές κατοικιών δείχνουν ενδιαφέρον για σπίτια με πολλά υπνοδωμάτια, τότε θα ήταν λογικό να δείξουμε στους αγοραστές τα σπίτια που οι ίδιοι δεν έχουν δει αλλά ο ομόλογός τους είχε δει. Επιπλέον, τα μοντέλα λανθάνοντος παράγοντα είναι πολύ καλά στην αναγνώριση χαρακτηριστικών που είναι δύσκολο να συμπεριληφθούν συγκεκριμένα ως ιδιότητες ενός αντικειμένου. Για παράδειγμα, οι χρήστες με παιδιά μπορεί επίσης να ενδιαφέρονται για σπίτια κοντά σε καλά σχολεία. Ακόμα κι

αν μας εφαρμογή μας δεν περιείχε αυτές τις πληροφορίες, ένα καλό μοντέλο λανθάνοντος παράγοντα θα τον συμπεράνει από τα δεδομένα με τρόπο χωρίς επίβλεψη.

Οι λανθάνοντες παράγοντες, όπως τα γνωστά χαρακτηριστικά, μπορεί να είναι διακριτές / κατηγορηματικές ή συνεχείς τιμές. Διατηρούνται σε μια λίστα ή μια σειρά τιμών, όπως συζητείται περαιτέρω στην επόμενη ενότητα. μοντέλα Συνεργατικού φιλτραρίσματος που χρησιμοποιούν λανθάνουσες παραμέτρους πετυχαίνουν καλύτερη ακρίβεια όσο αυξάνεται ο αριθμός των λανθάνων παραγόντων.

2.2.1 Παραγοντοποιηση μήτρας - Matrix Factorization

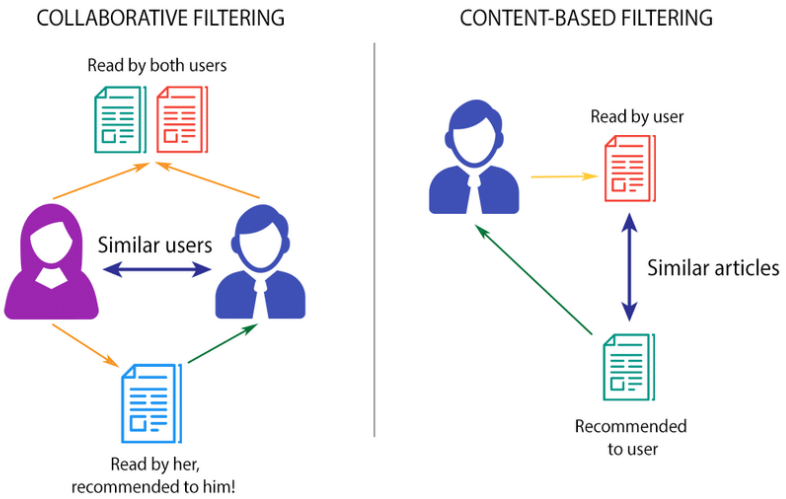
Η Παραγοντοποίηση μήτρας (MF) συσχετίζει κάθε χρήστη και στοιχείο με ένα διάνυσμα τιμών λανθανουσών χαρακτηριστικών. Αν τα $\mathbf{p}_u$ και $\mathbf{q}_i$ δηλώνουν το λανθάνον διάνυσμα για το χρήστη u και το στοιχείο i, αντίστοιχα. Η MF υπολογίζει μια αλληλεπίδραση y_{ui} ως το εσωτερικό γινόμενο των $\mathbf{p}_u$ και $\mathbf{q}_i$:

$$\hat{y}_{ui} = f(u, i \mid \mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u^T \mathbf{q}_i = \sum_{k=1}^{K} p_{uk} q_{ik}$$

όπου το K δηλώνει τη διάσταση του λανθάνοντος χώρου. Όπως μπορούμε να δούμε, το MF μοντελοποιεί τη αμφίδρομη αλληλεπίδραση χρηστών και στοιχείων λανθάνων παραγόντων, υποθέτοντας ότι κάθε διάσταση του λανθάνοντος χώρου είναι ανεξάρτητη και συνδυάζοντας τους γραμμικά με το ίδιο βάρος.

2.2.2 Συνεργατικό φιλτράρισμα - collaborative filtering

Το συνεργατικό φιλτράρισμα είναι μια τεχνική για τον προσδιορισμό των στοιχείων προς σύσταση χωρίς να απαιτείται από τους χρήστες να εισάγουν προσωπικά δεδομένα. Χρησιμοποιεί το ιστορικό των ενεργειών των χρηστών για να ταιριάξουν χρήστες με αντικείμενα και χρήστες με παρόμοιο ιστορικό θεωρείται ότι μοιράζονται τα ίδια χαρακτηριστικά. Αυτή η προσέγγιση είναι συνήθως πιο ακριβής για την πρόβλεψη συγγένειας χρήστη-στοιχείων από τις τεχνικές που βασίζονται σε συγκριση περιεχομένου (content based), καθώς χρησιμοποιούν το ιστορικό όλων των χρηστών για να κάνουν καλύτερες συστάσεις.



Εικόνα 2.3 : Η αρχιτεκτονική του ConvMF
Πηγή: https://hugrypiggykim.com/wp-content/uploads/2018/01/Convolutional-Matrix-Factorization-for-Document-Context-Aware-Recommendation.pdf

2.2.3 Μέθοδοι Συνεργατικού φιλτραρίσματος με χαρακτηριστικά γειτονιάς - CF Neighborhood methods

Οι μέθοδοι συνεργατικού φιλτραρίσματος CF που βασίζονται στη γειτονιά επικεντρώνονται στη σχέση μεταξύ στοιχείων (CF βάσει στοιχείων) ή εναλλακτικά, μεταξύ χρηστών (CF που βασίζεται σε χρήστες).

- Τα CF που βασίζονται στον χρήστη βρίσκουν χρήστες που έχουν παρόμοια προτίμηση για τα στοιχεία όπως και προτείνει νέα στοιχεία με βάση αυτά που τους αρέσουν.
- Τα CF που βασίζεται σε στοιχεία προτείνει στοιχεία παρόμοια με αυτά που αρέσουν στον χρήστη, όπου η ομοιότητα βασίζεται σε ομάδες στοιχείων (π.χ. χρήστες που αγόρασαν το x, αγόρασαν επίσης το y).

Μερικές από τις μεθόδους που χρησιμοποιούνται συνήθως για υπολογισμούς CF βάσει γειτονίας είναι:

- K-Nearest Neighbors (KNN)
- k-Means
- **k** -d Trees
- Locality Sensitive Hashing

Η βασική ιδέα στις μεθόδους που βασίζονται στη γειτονιά είναι να χρησιμοποιήσετε είτε ομοιότητα χρήστη-χρήστη είτε ομοιότητα στοιχείου για να κάνετε συστάσεις από έναν πίνακα αξιολογήσεων. Η έννοια της γειτονιάς υποδηλώνει ότι πρέπει να καθορίσουμε είτε παρόμοιους χρήστες είτε παρόμοια στοιχεία για να κάνουμε προβλέψεις. Στη συνέχεια, θα συζητήσουμε πώς μπορούν να χρησιμοποιηθούν μέθοδοι βάσει γειτονίας για την πρόβλεψη των βαθμολογιών συγκεκριμένων συνδυασμών στοιχείων χρήστη. Υπάρχουν δύο βασικές αρχές που χρησιμοποιούνται σε μοντέλα με βάση τη γειτονιά:

Μοντέλα βασισμένα σε χρήστες: Παρόμοιοι χρήστες έχουν παρόμοιες αξιολογήσεις για το ίδιο στοιχείο. Επομένως, εάν δύο χρήστες έχουν βαθμολογήσει τις ταινίες με παρόμοιο τρόπο στο παρελθόν, τότε κάποιος μπορεί να χρησιμοποιήσει τις βαθμολογίες σε ένα αντικείμενο για να προβλέψουν τις μη παρατηρούμενες βαθμολογίες ενός άλλου χρήστη για αυτό το αντικείμενο.

Μοντέλα που βασίζονται σε αντικείμενα: Παρόμοια στοιχεία βαθμολογούνται με παρόμοιο τρόπο από τον ίδιο χρήστη. Ως εκ τούτου, οι βαθμολογίες ενός αντικειμένου μπορούν να χρησιμοποιηθούν για κάποιο άλλο αντικείμενο για τον ίδιο χρήστη.

ΚΕΦΑΛΑΙΟ 3: Μοντέλα Βαθιάς μάθησης, συνυπολογίζοντας δευτερεύουσα πληροφορία

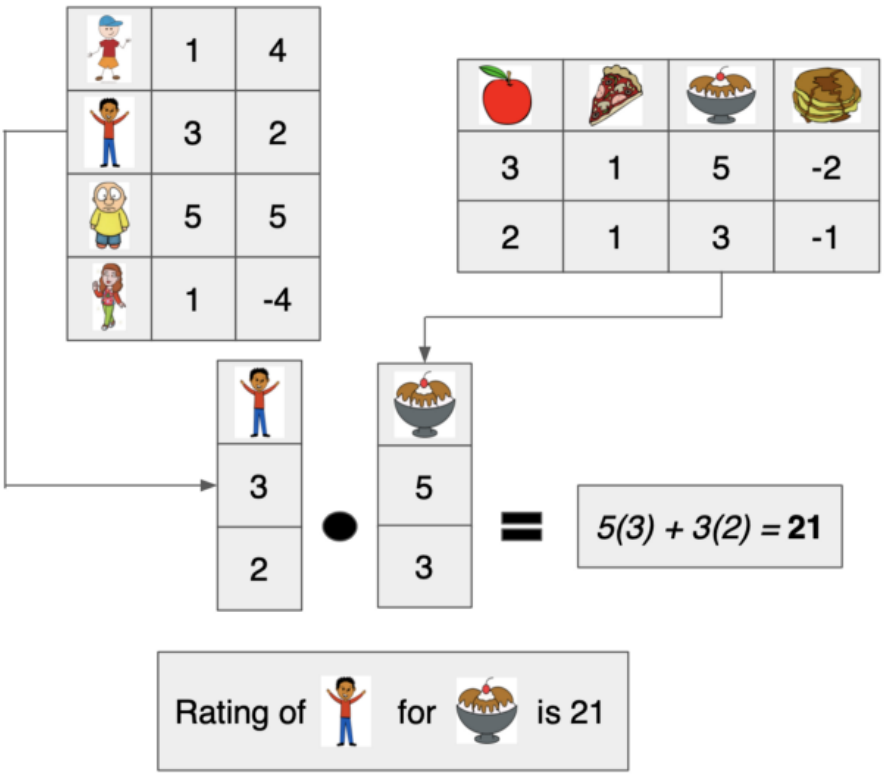
Λαμβάνοντας υπόψιν συνοδευτικές πληροφορίες αλγόριθμοι με τεχνικές βαθιάς μάθησης καταφέρνουν πιο αποτελεσματικές προβλέψεις για νέους και παλαιούς χρήστες. Συναντάμε τα παρακάτω μοντέλα:

PMF : Το πιθανολογικό μοντέλο παραγοντοποίησης μήτρας - Probabilistic Matrix Factorization είναι ένα τυπικό μοντέλο πρόβλεψης βαθμολογίας που χρησιμοποιεί μόνο βαθμολογίες για συνεργατικό φιλτράρισμα.

CTR : Το συνεργατικό Collaborative Topic Regression είναι ένα μοντέλο σύστασης state-of-the art, το οποίο συνδυάζει συλλογικό φιλτράρισμα (PMF) και μοντελοποίηση θεμάτων (LDA) για τη χρήση τόσο αξιολογήσεων όσο και εγγράφων.

CDL : Η συνεργατική βαθιά μάθηση είναι ένα άλλο υπερσύγχρονο μοντέλο σύστασης, το οποίο βελτιώνει την ακρίβεια της πρόβλεψης αξιολόγησης αναλύοντας έγγραφα χρησιμοποιώντας SDAE

ConvMF: Το Convolutional Matrix Factorization είναι το μοντέλο που χρησιμοποιείται σε μια από τις υλοποιήσεις



Εικόνα 3.1 : Το πιθανολογικό μοντέλο παραγοντοποίησης μήτρας
Πηγή: https://towardsdatascience.com/building-a-music-recommendation-engine-with-probabilistic-matrix-factorization-in-pytorch-7d2934067d4a

3.1 Συνελικτική παραγοντοποίηση μήτρας – convolutional matrix factorization

3.1.1 Αρχιτεκτονική CNN των Συνελικτικών δικτύων παραγοντοποίησης μήτρας (ConvMF)

Ο στόχος της αρχιτεκτονικής του CNN είναι να δημιουργήσει λανθάνοντα διανύσματα από σύνολα στοιχείων, τα οποία χρησιμοποιούνται για τη σύνθεση των λανθάνουσων μοντέλων των στοιχείων με μεταβλητές Εψιλον. Το παρακάτω σχήμα δείχνει την αρχιτεκτονική του CNN που αποτελείται από τέσσερα επίπεδα. 1) Επίπεδο ενσωμάτωσης, 2) Επίπεδο συνέλιξης, 3) Επίπεδο συγκέντρωσης και 4) Διάταξη εξόδου.

Η αρχιτεκτονική Συνελικτικής παραγοντοποίησης μήτρας (ConvMF), περιγράφεται με 3 μοντέλα:

Το πιθανολογικό μοντέλο του ConvMF και περιγράφουμε τη βασική ιδέα να γεφυρώσουμε PMF και CNN για να χρησιμοποιήσουμε και τις δύο αξιολογήσεις και έγγραφα περιγραφής αντικειμένου.

Η αρχιτεκτονική του CNN, η οποία δημιουργεί λανθάνον μοντέλο εγγράφων αναλύοντας έγγραφα περιγραφής στοιχείων.

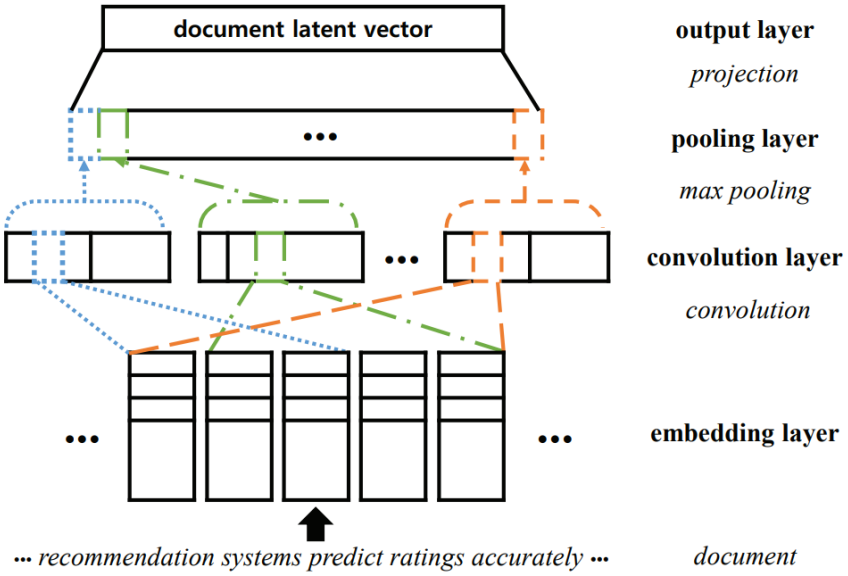
Βελτιστοποίηση λανθανουσών τελεστών του ConvMF.

Πιθανολογικό μοντέλο του ConvMF

Ας υποθέσουμε ότι έχουμε N χρήστες και M στοιχεία και οι παρατηρούμενες βαθμολογίες αντιπροσωπεύονται από τον πίνακα $\boldsymbol{R} \in \mathbb{R}^{N \times M}$. Στη συνέχεια, ο στόχος μας είναι να βρούμε λανθάνοντα μοντέλα χρηστών και στοιχείων ($\boldsymbol{U} \in \mathbb{R}^{k \times N}$ και $\boldsymbol{V} \in \mathbb{R}^{k \times M}$) των οποίων το προϊόν ($\boldsymbol{U}^T\boldsymbol{V}$) ανακατασκευάζει τον πίνακα βαθμολογίας $\boldsymbol{R}$. Η υπό όρους κατανομή επί των παρατηρούμενων βαθμολογιών δίνεται από την:

$$p(\boldsymbol{R} \mid \boldsymbol{U}, \boldsymbol{V}, \sigma^2) = \prod_i^N \prod_j^M N\left(r_{ij} \mid \boldsymbol{u}_i^T \boldsymbol{v}_j, \sigma^2\right)^{I_{ij}} \quad (3.1)$$

όπου $N(\boldsymbol{x} \mid \boldsymbol{\mu}, \sigma^2)$ είναι η συνάρτηση πυκνότητας πιθανότητας της κανονικής κατανομής του Gauss με μέσο $\boldsymbol{\mu}$ και διακύμανση σ^2, ενώ το $\boldsymbol{I}_{ij}$ είναι μια συνάρτηση δείκτη.



Εικόνα 3.2: Η αρχιτεκτονική του ConvMF
Πηγή: https://hugrypiggykim.com/wp-content/uploads/2018/01/Convolutional-Matrix-Factorization-for-Document-Context-Aware-Recommendation.pdf

Επίπεδο ενσωμάτωσης

Το επίπεδο ενσωμάτωσης μετατρέπει ένα ακατέργαστο έγγραφο σε πυκνό πίνακα, για το επόμενο στρώμα συνέλιξης. Αν το έγγραφο θεωρηθεί ως ακολουθία l λέξεων, αναπαριστούμε το έγγραφο ως μήτρα, ενώνοντας διανύσματα λέξεων με λέξεις στο έγγραφο. Οι λέξεις διανύσματα αρχικοποιούνται τυχαία ή με προ-εκπαιδευμένο μοντέλο ενσωμάτωσης λέξεων όπως το GLOVE. Τα διανύσματα λέξεων αλλάζουν περαιτέρω μέσω διαδικασίας βελτιστοποίησης. Στη συνέχεια, ο πίνακας εγγράφων $\boldsymbol{D} \in \mathbb{R}^{p \times l}$ γίνεται:

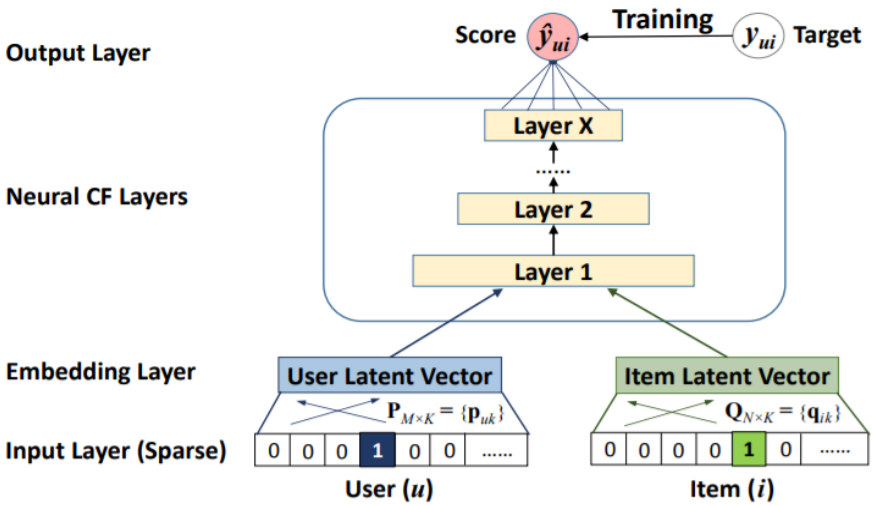
$$\boldsymbol{D} = \begin{bmatrix} \cdots & | & | & | & \cdots \\ & \boldsymbol{w}_{i-1} & \boldsymbol{w}_i & \boldsymbol{w}_{i+1} & \\ & | & | & | & \end{bmatrix} \quad (3.2)$$

όπου $\boldsymbol{l}$ είναι το μήκος του εγγράφου και $\boldsymbol{p}$ είναι το μέγεθος της διάστασης ενσωμάτωσης για κάθε λέξη $\boldsymbol{w}_i$.

Το επίπεδο συνέλιξης εξάγει συμφραζόμενα χαρακτηριστικά.

Το επίπεδο συγκέντρωσης εξάγει αντιπροσωπευτικά χαρακτηριστικά από το επίπεδο συνέλιξης και ασχολείται επίσης με μεταβλητά μήκη σειρών μέσω μιας λειτουργίας συγκέντρωσης που κατασκευάζει ένα διάνυσμα χαρακτηριστικών σταθερού μήκους.

3.1.2 Συνεργατικό Φιλτράρισμα με χρήση νευρωνικού δικτύου - Neural collaborative filtering



Εικόνα 3.3: Η αρχιτεκτονική Neural collaborative filtering
Πηγή: http://staff.ustc.edu.cn/~hexn/papers/www17-ncf.pdf

Αρχιτεκτονική

Για να σκεφτούμε την αντιμετώπιση του συνεργατικού φιλτραρίσματος με διάταξη νευρωνικού δικτύου, υιοθετούμε μια αναπαράσταση πολλαπλών επιπέδων για να μοντελοποιήσουμε μια αλληλεπίδραση στοιχείου χρήστη y_{ui} , όπου η έξοδος ενός επιπέδου χρησιμεύει ως είσοδος του

επόμενου. Το κάτω επίπεδο εισόδου αποτελείται από δύο διανύσματα χαρακτηριστικών $\mathbf{v}_u^U$ και $\mathbf{v}_i^I$ που περιγράφουν το χρήστη $\boldsymbol{u}$ και το στοιχείο $\boldsymbol{i}$, αντίστοιχα. Μπορούν επίσης να υποστηρίζουν ένα ευρύ φάσμα μοντελοποίησης χρηστών και στοιχείων, όπως το περιβάλλον, το περιεχόμενο, και την γειτονιά. Με μια τέτοια γενική αναπαράσταση χαρακτηριστικών για τις εισόδους, η μέθοδος αντιμετωπίζει το πρόβλημα της ψυχρής εκκίνησης, χρησιμοποιώντας δυνατότητες περιεχομένου για την αναπαράσταση χρηστών και στοιχείων.

Το επίπεδο ενσωμάτωσης είναι ένα πλήρως συνδεδεμένο επίπεδο που προβάλλει την αραιή αναπαράσταση σε ένα πυκνό διάνυσμα. Η ενσωμάτωση του χρήστη (στοιχείου) που λαμβάνεται μπορεί να θεωρηθεί ως το λανθάνον διάνυσμα για τον χρήστη (στοιχείο) στο πλαίσιο του μοντέλου λανθάνοντος παράγοντα. Η ενσωμάτωση του χρήστη και η ενσωμάτωση στοιχείων εισάγονται στη συνέχεια σε μια νευρωνική αρχιτεκτονική πολλαπλών επιπέδων, την οποία ονομάζουμε ως νευρωνικό συνεργατικό φιλτράρισμα, για να χαρτογραφήσουμε τα λανθάνοντα διανύσματα σε βαθμολογίες πρόβλεψης.

Κάθε επίπεδο εκπαιδεύεται για να ανακαλύψει ορισμένες λανθάνουσες αλληλεπιδράσεις στοιχείων-χρήστη. Η διάσταση του τελευταίου κρυμμένου επιπέδου X καθορίζει την ικανότητα του μοντέλου. Το τελικό επίπεδο εξόδου είναι η προβλεπόμενη βαθμολογία y⊥ˆ_ui και η εκπαίδευση πραγματοποιείται ελαχιστοποιώντας τη σημειακή απώλεια μεταξύ $\hat{\boldsymbol{y}}_{\boldsymbol{ui}}$, και της τιμής στόχου $\boldsymbol{y_{ui}}$. Εναλακτικός τρόπος εκπαίδευσης του μοντέλου είναι η εκπαίδευση ανά ζεύγη, όπως η χρήση Bayesian Personalized Ranking και η margin-based loss. Έτσι διατυπώνουμε το προγνωστικό μοντέλο του NCF ως:

$$\hat{\boldsymbol{y}}_{\boldsymbol{ui}} = \boldsymbol{f}\left(\mathbf{P}^T\mathbf{v}_u^U, \mathbf{Q}^T\mathbf{v}_i^I \mid \mathbf{P}, \mathbf{Q}, \boldsymbol{\Theta}_f\right) \quad (3.3)$$

όπου $\mathbf{P} \in \mathbb{R}^{M \times K}$ και $\mathbf{Q} \in \mathbb{R}^{N \times K}$, δηλώνοντας τον πίνακα λανθανόντων συντελεστών για χρήστες και στοιχεία, αντίστοιχα · και το $\boldsymbol{\Theta}_f$υποδηλώνει τις παραμέτρους της συνάρτησης αλληλεπίδρασης $\boldsymbol{f}$.

3.1.3 Το μοντέλο AUTOREC

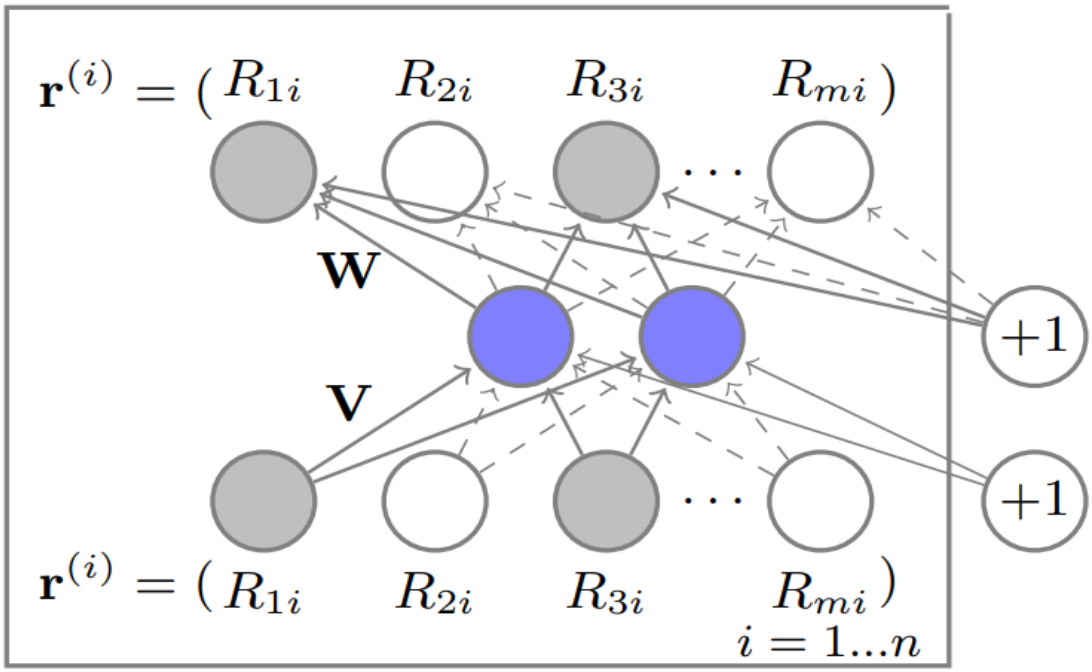
AutoRec: Πρόβλεψη αξιολόγησης με autoencoders.

Ο AutoRec [Sedhain et al., 2015]. προσεγγίζει το συνεργατικό φιλτράρισμα (CF) με μια αρχιτεκτονική autoencoder και στοχεύει να ενσωματώσει μη γραμμικούς μετασχηματισμούς σε CF με βάση ρητή ανατροφοδότηση. Έχει αποδειχθεί ότι τα νευρωνικά δίκτυα είναι σε θέση να προσεγγίζουν κάθε συνεχή συνάρτηση, καθιστώντας το κατάλληλο να αντιμετωπίσει τον περιορισμό της παραγοντοποίησης της μήτρας και να εμπλουτίσει την εκφραστικότητα της παραγοντοποίησης μήτρας.

Ο AutoRec έχει την ίδια δομή με autoencoder, που αποτελείται από ένα επίπεδο εισόδου, ένα κρυφό επίπεδο και ένα κρυφό ανακατασκευής (εξόδου). Οι autoencoders είναι νευρωνικά δίκτυα που μαθαίνουν να αντιγράφουν την είσοδό τους στην έξοδο για να κωδικοποιήσουν τις εισόδους σε κρυφές (και συνήθως χαμηλής διάστασης) αναπαραστάσεις.

Ο Auto-Rec, αντί να ενσωματώνει ρητά τους χρήστες/στοιχεία σε χώρο χαμηλών διαστάσεων, χρησιμοποιεί τη στήλη/γραμμή του πίνακα αλληλεπίδρασης ως είσοδο και στη συνέχεια ανακατασκευάζει τον πίνακα αλληλεπίδρασης στο επίπεδο εξόδου. Χρησιμοποιεί έναν μερικώς παρατηρούμενο πίνακα αλληλεπίδρασης ως είσοδο, με στόχο την ανασυγκρότηση ενός

ολοκληρωμένου πίνακα αξιολόγησης. Εν τω μεταξύ, τα κενά της εισόδου συμπληρώνονται στο επίπεδο εξόδου μέσω ανακατασκευής για παραχθούν προτάσεις.



Εικόνα 3.4: Μοντέλο AutoRec ανά στοιχείο. Χρησιμοποιούμε σημειογραφία πινακίδας για να υποδείξουμε ότι υπάρχουν n αντίγραφα του νευρωνικού δικτύου (ένα για κάθε στοιχείο), όπου W και V είναι δεμένα σε όλα τα αντίγραφα.

Πηγή: https://users.cecs.anu.edu.au/~u5098633/papers/www15.pdf

Μοντέλο

Αν θεωρήσουμε ότι το $\mathbf{R}_{*i}$ υποδηλώνει την ι στήλη του πίνακα αξιολόγησης, όπου οι άγνωστες βαθμολογίες έχουν οριστεί ως μηδενικά. Η αρχιτεκτονική του δικτύου ορίζεται ως:

$$\boldsymbol{h}(\mathbf{R}_{*i}) = \boldsymbol{f}(\mathbf{W} \cdot \boldsymbol{g}(\mathbf{V}\mathbf{R}_{*i} + \boldsymbol{\mu}) + \boldsymbol{b}) \quad (3.4)$$

με $\boldsymbol{f}(\cdot)$ και $\boldsymbol{g}(\cdot)$ να αντιπροσωπεύουν συναρτήσεις ενεργοποίησης, $\mathbf{W}$ και $\mathbf{V}$ είναι πίνακες βαρών, $\boldsymbol{\mu}$ και $\boldsymbol{\mu}$ το κατώφλι (bias). Το $\boldsymbol{h}(\cdot)$ δηλώσει ολόκληρο το δίκτυο του AutoRec. Η έξοδος $\boldsymbol{h}(\mathbf{R}_{*i})$ είναι η ανακατασκευή της στήλης $\boldsymbol{I}$ της μήτρας βαθμολογιών.

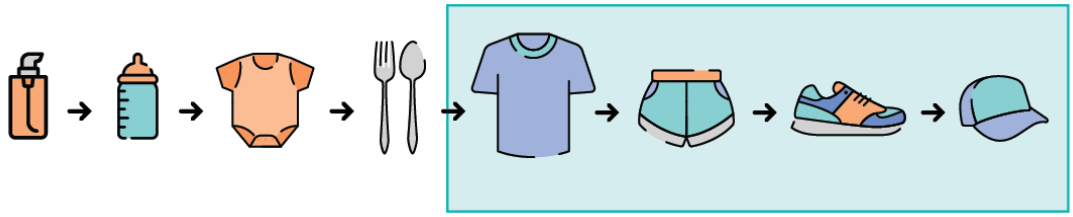
Η ακόλουθη αντικειμενική συνάρτηση στοχεύει στην ελαχιστοποίηση του σφάλματος ανασυγκρότησης:

$$\underset{\mathbf{W},\mathbf{V},\boldsymbol{\mu},\boldsymbol{b}}{\operatorname{argmin}} \sum_{i=1}^{M} \|\mathbf{R}_{*i} - \boldsymbol{h}(\mathbf{R}_{*i})\|_{O}^{2} + \boldsymbol{\lambda}\left(\| \mathbf{W} \|_{F}^{2} + \| \mathbf{V} \|_{F}^{2}\right) \quad (3.5)$$

όπου $\|\cdot\|_O$ υποδηλώνει ότι λαμβάνονται υπόψη μόνο οι παρατηρούμενες βαθμολογίες κατά την οπισθοδιάδοση.

ΚΕΦΑΛΑΙΟ 4: Αρχιτεκτονικές συστημάτων σειριακών συστάσεων και ανά συνεδρία - Sequential - Session based recommendation systems

4.1 Εισαγωγή



Εικόνα 4.1 Με βάση τις υπάρχουσες αγορές από τον χρήστη, προβλέπουμε τις επόμενες αγορές (γαλάζιο πλαίσιο)

Για μερικώς γνωστές πληροφορίες περιόδου σύνδεσης, π.χ. μέρος μιας περιόδου σύνδεσης ή πρόσφατων περιόδων σύνδεσης ένα σύστημα συστάσεων που βασίζεται σε συνεδρίες θα πρέπει να μπορεί να εκμεταλλευτεί τις εξελισσόμενες προτιμήσεις ενός χρήστη. Αυτές υποθέτουμε ότι είναι ένα μείγμα βραχυπρόθεσμων και μακροπρόθεσμων ενδιαφερόντων.

Στους αλγόριθμους σύστασης που βασίζονται σε δεδομένα από την εκάστοτε συνεδρία, τα μοναδικά στοιχεία που εμπλέκονται σε αλληλουχίες αλληλεπίδρασης $\boldsymbol{V} = \{\boldsymbol{v}_i\}_{i=1}^{\boldsymbol{m}}$ και οι αλληλουχίες αλληλεπίδρασης των χρηστών μπορούν να ταξινομηθούν με χρονικές σημάνσεις (δηλαδή, $\boldsymbol{s} = \left[\boldsymbol{v}_{s,i}\right]_{i=1}^{s_n}$), όπου ένα συμβάν αλληλεπίδρασης που εμπλέκεται στην αλληλουχία αλληλεπίδρασης s συμβολίζεται ως $\boldsymbol{v}_{s,i} \in \boldsymbol{V}$. Με βάση τα παραπάνω σύμβολα και περιγραφές, το πρόβλημα που πρέπει να επιλυθεί για σύσταση που βασίζεται σε συνεδρίες είναι:

Δεδομένης μιας αλληλουχίας συμβάντων s, πρέπει να προβλέψουμε το συμβάν αλληλεπίδρασης του χρήστη $\boldsymbol{v}_{s,s_n+1}$ στο εγγύς μέλλον. Η αρχιτεκτονική του μοντέλου της προσέγγισής μας φαίνεται παρακάτω.

$$\boldsymbol{s}^{\boldsymbol{u}} = \left[\boldsymbol{i}_{s,1}, \boldsymbol{i}_{s,2}, \ldots, \boldsymbol{i}_{s,n}\right] \quad (4.1)$$

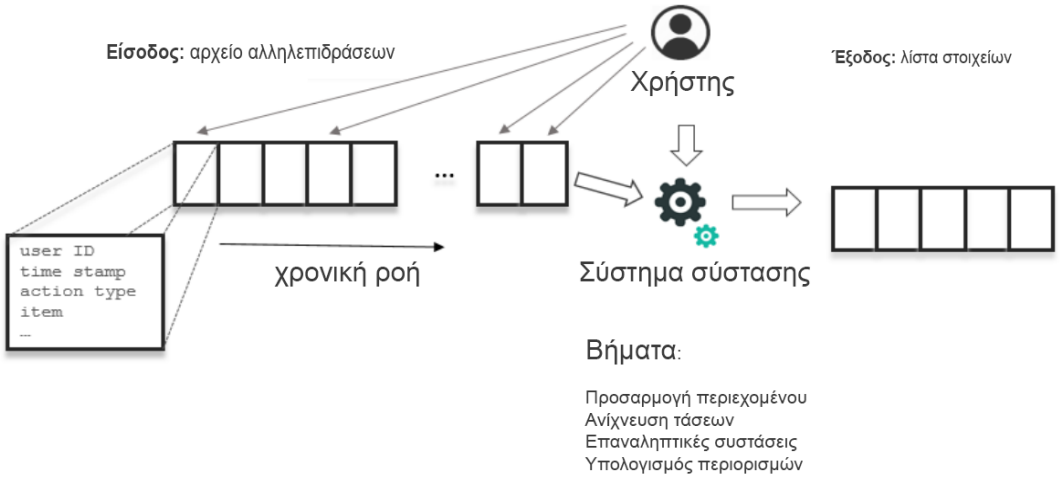
$$\boldsymbol{i}_{s,n+1}^{*} = \arg\max_{\boldsymbol{i} \in \boldsymbol{I}} \boldsymbol{P}\left(\boldsymbol{i}_{s,n+1} = \boldsymbol{i} \mid \boldsymbol{s}^{\boldsymbol{u}}\right) \quad (4.2)$$

Το πρόβλημα διατυπώνεται ως εξής:

Αν $[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n-1}, \boldsymbol{x}_n]$ να είναι μια συνεδρία διαδράσεων, όπου $\boldsymbol{x}_i \in \boldsymbol{I} (1 \leq \boldsymbol{i} \leq \boldsymbol{n})$ είναι ο δείκτης ενός στοιχείου που επέλεξε ο χρήστης από ένα συνολικό αριθμό $\boldsymbol{m}$ στοιχείων. Φτιάχνουμε ένα μοντέλο M έτσι ώστε για οποιοδήποτε πρόθεμα της ακολουθίας $\mathbf{x} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t], 1 \leq \boldsymbol{t} \leq \boldsymbol{n}$, να έχουμε την έξοδο $\mathbf{y} = \mathbf{M}(\mathbf{x})$, όπου $\mathbf{y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{m-1}, \boldsymbol{y}_m]$ αντιστοιχεί στη βαθμολογία σύστασης του στοιχείου j. Δεδομένου ότι συνήθως να γίνουν περισσότερες από μία προτάσεις προς τον χρήστη, τα κορυφαία στοιχεία (1≤k≤m) συνιστώνται από την y.

Η κύρια πρόκληση των συστημάτων διαδοχικών συστάσεων είναι να μάθουμε μια ακολουθία ενεργειών που αντικατοπτρίζει την τρέχουσα προτίμηση του χρήστη. Οι πρώτες λύσεις υιοθέτησαν αλυσίδες Μαρκόφ - Markov Chains (MC) για να καταγράψουν τη ροή μετάβασης αντικειμένων με βάση την παραδοχή ότι το πιο πρόσφατο στοιχείο που επιλέγει ο χρήστης είναι έναυσμα για τις αλλαγές στις προτιμήσεις του. Ακολούθως επιστρατεύθηκαν τα Αναδρομικά

Νευρωνικά δίκτυα Recurrent Neural Network (RNN) στη μοντελοποίηση ακολουθιών, λόγω του πλεονεκτημάτων τους ως προς τη σύλληψη μοτίβων ακολουθίας. Για να βελτιωθεί περαιτέρω η αναπαράσταση της εκάστοτε συνεδρίας, νεότερες μέθοδοι αξιοποιούν μηχανισμό προσοχής για να ενσωματώσουν ολόκληρη την ακολουθία εκτός του πλέον πρόσφατου στοιχείου. Εμπνευσμένο από τις επιδόσεις των Transformer [128] σε προβλήματα NLP, οι SASRec [45] και BERT4Rec [96] αξιοποιούν την τεχνική αυτο-προσοχής για να μοντελοποιήσουν τις αλληλεπιδράσεις στοιχείων, η οποία επιτρέπει περισσότερη ευελιξία στις μεταβάσεις μεταξύ αντικειμένων. Με την εμφάνιση των Νευρωνικών Δικτύων Γράφων GNN, η χρήση των GNN για τη σύλληψη σύνθετων μοτίβων μετάβασης των στοιχείων έχει γίνει δημοφιλής σε συστήματα διαδοχικών προτάσεων.



Εικόνα 4.2: Το πρόβλημα των συστάσεων ανα συνεδρία

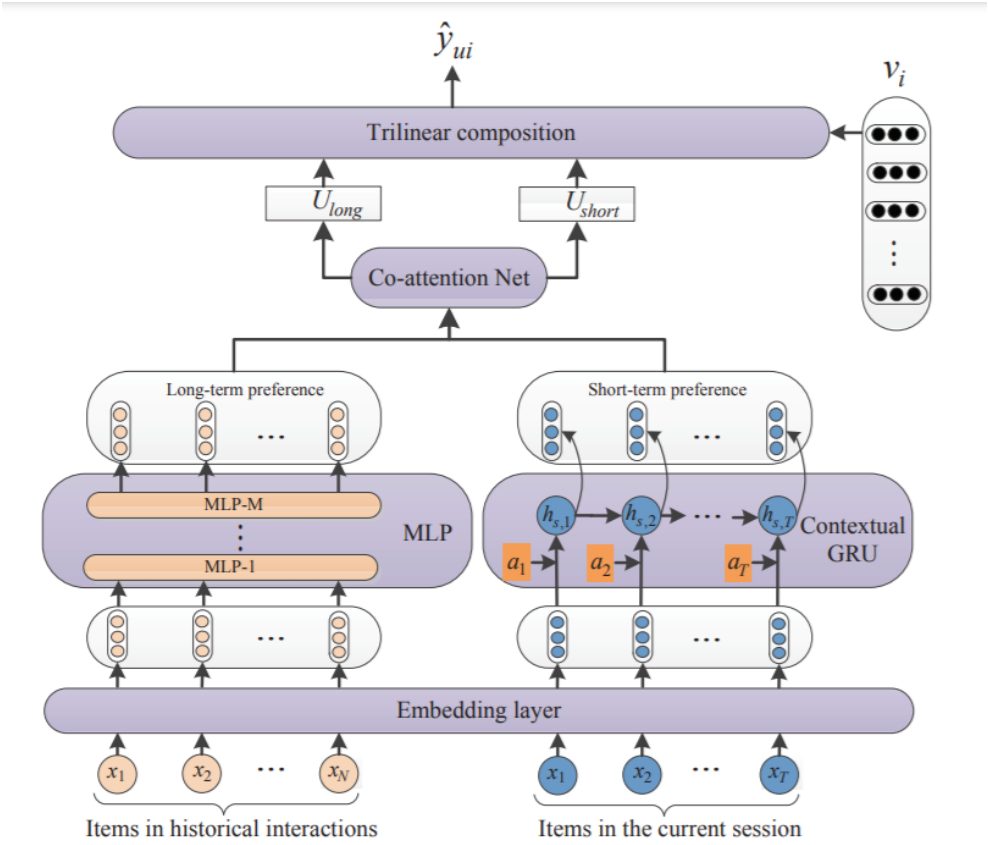
4.2 Παραγοντοποίηση αλυσίδων Markov για συστάσεις ανά συνεδρία.

Συνοπτικά, οι κύριες προκλήσεις που αντιμετωπίζει η σύσταση βάσει περιόδου σύνδεσης είναι

- Η ενσωμάτωση τόσο των μακροπρόθεσμων όσο και των βραχυπρόθεσμων προτιμήσεων του χρήστη για συστάσεις
- Η ανακάλυψη των δυναμικών προτιμήσεων των χρηστών από έμμεσα δεδομένα.

Μια αρχιτεκτονική που λύνει αυτό το πρόβλημα είναι το Δυναμικό Δίκτυο Συνεργασίας για Συστάσεις βάσει Συνεδρίας (DCN-SR). Το DCN-SR έχει τρία βασικά συστατικά:

- Το πρώτο είναι ένα δίκτυο Concurrent GR Unit (CGRU) με βάση τα συμφραζόμενα για τη μοντελοποίηση των βραχυπρόθεσμων προτιμήσεων ενός χρήστη, που αναπαριστούμε ως συνδυασμό κρυφών καταστάσεων αλληλεπιδράσεων κατά την τρέχουσα συνεδρία.
- Το δεύτερο είναι ένα Multi-Layer Perceptron (MLP) που ασχολείται με τις ιστορικές αλληλεπιδράσεις ενός χρήστη και συνάγει μακροπρόθεσμες προτιμήσεις.
- Το τρίτο είναι ένα δίκτυο συν-προσοχής που χρησιμοποιεί τα αποτελέσματα των δύο πρώτων συνιστωσών για να συλλάβει αλληλεπιδράσεις μεταξύ ενεργειών σε ιστορικά μακροπρόθεσμης και βραχυπρόθεσμης αλληλεπίδρασης ενός χρήστη και να παράγει συν-εξαρτημένες αναπαραστάσεις των μακροπρόθεσμων και βραχυπρόθεσμων προτιμήσεων.



Εικόνα 4.3: Αρχιτεκτονική DCN-SR Συστάσεων ανά συνεδρία.
Πηγή: DCN-SR

4.3 Σειριακές Συστάσεις με Αναδρομικά Νευρωνικά Δίκτυα (RNNs)

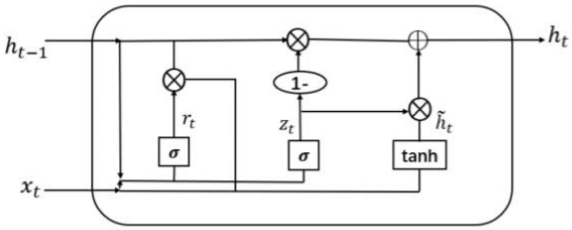
Τα αναδρομικά νευρωνικά δίκτυα έχουν επινοηθεί για να μοντελοποιήσουν ακολουθίες μεταβλητού μήκους. Η κύρια διαφορά μεταξύ των RNN και των συμβατικών μοντέλων feed forward είναι η ύπαρξη μιας εσωτερικής κρυφής κατάστασης στις μονάδες που συνθέτουν το δίκτυο. Τα τυπικά RNN ενημερώνουν την κρυφή τους κατάσταση $\boldsymbol{h}$ χρησιμοποιώντας την ακόλουθη συνάρτηση ενημέρωσης:

$$\mathbf{h_t} = \boldsymbol{g}(\boldsymbol{W}\mathbf{x_t} + \boldsymbol{U}\mathbf{h_{t-1}}) \quad (4.3)$$

Όπου $\boldsymbol{g}$ είναι μια σιγμοειδής συνάρτηση $\mathbf{x_t}$ είναι η είσοδος της μονάδας τη στιγμή $\boldsymbol{t}$. Ένα RNN εξάγει μια κατανομή πιθανοτήτων στο επόμενο στοιχείο της ακολουθίας, δεδομένης της τρέχουσας κατάστασής του $\mathbf{h_t}$.

GRU

Μια Gated Recurrent Unit (GRU) (Cho et al., 2014) είναι ένα εξελιγμένο μοντέλο μιας μονάδας RNN που στοχεύει στην αντιμετώπιση του προβλήματος ομαλοποίησης (vanishing gradient problem). Οι πύλες GRU ουσιαστικά μαθαίνουν πότε και πόσο να ενημερώσουν την κρυφή κατάσταση της μονάδας.



Εικόνα 4.4: Μια Gated Recurrent Unit (GRU)
Πηγή: Cho et al., 2014

Η ενεργοποίηση του GRU είναι μια γραμμική παρεμβολή μεταξύ της προηγούμενης ενεργοποίησης και της υποψήφιας ενεργοποίησης $\hat{\mathbf{h}}_\mathbf{t}$:

$$\mathbf{h_t} = (1 - \mathbf{z_t}) \otimes \mathbf{h_{t-1}} + \mathbf{z_t} \otimes \hat{\mathbf{h}}_\mathbf{t} \quad (4.4)$$

Ο τελεστής $\otimes$ δηλώνει πολλαπλασιασμό (element-wise multiplication). Η πύλη ενημέρωσης δίνεται από:

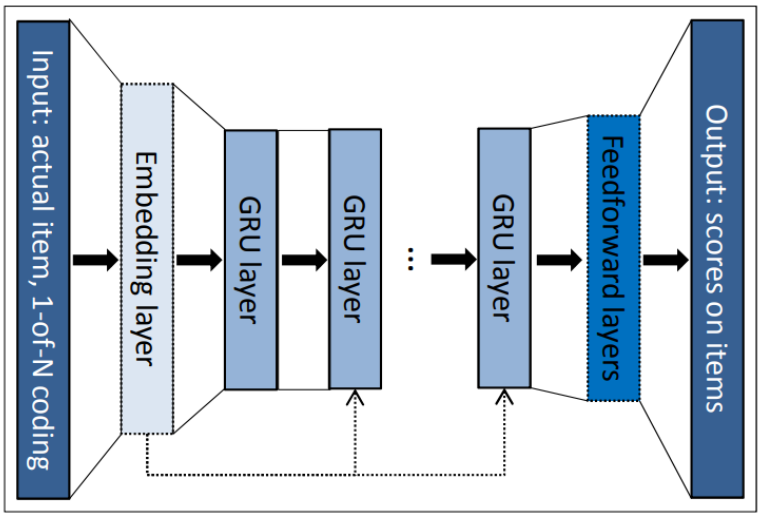
$$\mathbf{z_t} = \boldsymbol{\sigma}(\boldsymbol{W_z}\mathbf{x_t} + \boldsymbol{U_z}\mathbf{h_{t-1}}) \quad (4.5)$$

ενώ η υποψήφια συνάρτηση ενεργοποίησης $\hat{\mathbf{h}}_\mathbf{t}$ υπολογίζεται με παρόμοιο τρόπο:

$$\hat{\mathbf{h}}_\mathbf{t} = \mathbf{tanh}\left(\boldsymbol{W}\mathbf{x_t} + \boldsymbol{U}(\mathbf{r_t} \otimes \mathbf{h_{t-1}})\right) \quad (4.6)$$

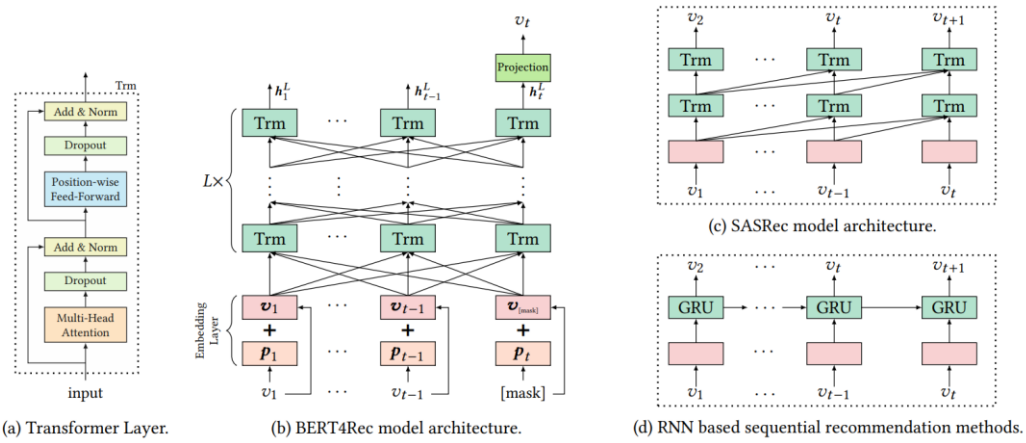
και τέλος η πύλη επαναφοράς $\mathbf{r_t}$ δίνεται από:

$$\mathbf{r_t} = \boldsymbol{\sigma}(\boldsymbol{W_r}\mathbf{x_t} + \boldsymbol{U_r}\mathbf{h_{t-1}}) \quad (4.7)$$



Εικόνα 4.5: Γενική αρχιτεκτονική του δικτύου. Επεξεργασία ενός συμβάντος της ροής συμβάντων παράλληλα.
Πηγή: Cho et al., 2014

4.4 Transformers σε μοντέλα αλληλεπιδράσεων με τεχνικές αυτοεκτίμησης (self attention)



(a) Transformer Layer. (b) BERT4Rec model architecture. (c) SASRec model architecture. (d) RNN based sequential recommendation methods.

Εικόνα 4.6: Διαφορές στις αρχιτεκτονικές μοντέλων σύστασης συνεδριών. Το BERT4Rec μαθαίνει ένα αμφίδρομο μοντέλο μέσω της εργασίας Cloze, ενώ οι μέθοδοι που βασίζονται σε SASRec και RNN προβλέπουν διαδοχικά προς τα δεξιά το επόμενο στοιχείο.
Πηγή: Cho et al., 2014

4.5 Τα δυνατά σημεία των συστημάτων συστάσεων με αλγόριθμους που στηρίζονται σε βαθιά νευρωνικά δίκτυα.

Μη γραμμικός μετασχηματισμός (non linear transformation)

Σε αντίθεση με τα γραμμικά μοντέλα, όπως η όπως η παραγοντοποίηση μήτρας, τα βαθιά νευρωνικά δίκτυα είναι ικανά μοντελοποίησης με μη γραμμικές ενεργοποιήσεις όπως relu, sigmoid, tanh κ.λπ.

Για παράδειγμα, η παραγοντοποίηση μήτρας διαμορφώνει την αλληλεπίδραση χρήστη-στοιχείου συνδυάζοντας γραμμικά τους χρήστες και τις λανθάνουσες παραμέτρους των αντικειμένων. Η μηχανή παραγοντοποίησης είναι μέλος μιας γραμμικής οικογένειας πολλαπλών παραλλαγών. Τα νευρωνικά δίκτυα είναι σε θέση να προσεγγίζουν οποιαδήποτε συνεχή λειτουργία με ορισμένη ακρίβεια μεταβάλλοντας τις επιλογές ενεργοποίησης και τους συνδυασμούς. Αυτή η ιδιότητα τα καθιστά ικανά να εφαρμοστούν σε πολύπλοκα προβλήματα αλληλεπίδρασης και να καθορίσουν με ακρίβεια την προτίμηση του χρήστη.

Εκμάθηση αναπαραστάσεων (representation learning)

Τα βαθιά νευρωνικά δίκτυα είναι αποτελεσματικά στην εκμάθηση των υποκείμενων επεξηγηματικών παραγόντων και χρήσιμων αναπαραστάσεων από δεδομένα εισόδου, ειδικά όταν αυτά είναι διαθέσιμα σε μεγάλους όγκους, όπως στην περίπτωση των συστημάτων συστάσεων.

Έτσι, είναι φυσική επιλογή να εφαρμόζουμε βαθιά νευρωνικά δίκτυα στην εκπροσώπηση της μάθησης σε μοντέλα συστάσεων. Τα πλεονεκτήματα της χρήσης βαθιών νευρωνικών δικτύων για να βοηθήσουν την εκμάθηση της αναπαράστασης είναι διττά:

Μείωση του χειρωνακτικού σχεδιασμού χαρακτηριστικών.

Τα βαθιά νευρωνικά δίκτυα επιτρέπουν την αυτόματη εκμάθηση χαρακτηριστικών από ανεπεξέργαστα δεδομένα χωρίς επίβλεψη ή με ημιεπίβλεψη επιτρέπει στα μοντέλα προτάσεων να περιλαμβάνουν ετερογενείς πληροφορίες περιεχομένου όπως κείμενο, εικόνες, ήχος και ακόμη και βίντεο. Τα δίκτυα βαθιάς μάθησης έχουν πραγματοποιήσει σημαντικές ανακαλύψεις από την επεξεργασία πολυμεσικών δεδομένων και έχουν σημαντικές δυνατότητες για μάθηση αναπαραστάσεων από διάφορες πηγές.

Μοντελοποίηση ακολουθιών (Sequence modeling)

Τα βαθιά νευρωνικά δίκτυα έχουν δείξει πολλά υποσχόμενα αποτελέσματα σε μια σειρά διαδοχικών εργασιών μοντελοποίησης σειριακών δεδομένων όπως στην αυτόματη μετάφραση, κατανόηση φυσικής γλώσσας, αναγνώριση ομιλίας, chatbots και πολλά άλλα. Τα RNN και CNN διαδραματίζουν κρίσιμο ρόλο σε αυτές τις εργασίες. Τα RNN το επιτυγχάνουν αυτά με καταστάσεις εσωτερικής μνήμης, ενώ το CNN το επιτυγχάνουν αυτό με κυλιόμενα φίλτρα στο χρόνο. Και τα δύο είναι ευρέως εφαρμόσιμα και ευπροσάρμοστα στην εξόρυξη δομών διαδοχικών δεδομένων Η μοντελοποίηση διαδοχικών σημάτων είναι ένα σημαντικό θέμα για την πρόβλεψη της δυναμικής συμπεριφοράς των χρηστών και της εξέλιξης των στοιχείων. Για παράδειγμα, η πρόβλεψη επόμενου αντικειμένου / καλαθιού και η πρόταση βάσει χαρακτηριστικών μιας συγκεκριμένης περιόδου σύνδεσης είναι δυο τυπικές εφαρμογές. Ως εκ τούτου, τα βαθιά νευρωνικά δίκτυα είναι το κατάλληλο εργαλείο για αυτό το διαδοχικό έργο εξόρυξης καθοριστικών χαρακτηριστικών.

Ευελιξία.

Οι τεχνικές βαθιάς μάθησης διαθέτουν υψηλή ευελιξία, ειδικά με την εμφάνιση πολλών δημοφιλών Frameworks βαθιάς μάθησης όπως Tensor*f*ow3, Keras4, MXnet6, DeepLearning4j7, PyTorch8, Θeano9, κ.λπ. Τα περισσότερα από αυτά τα εργαλεία έχουν αναπτυχθεί με αρθρωτό τρόπο, έχουν ενεργή κοινότητα και υποστήριξη.

Για παράδειγμα, είναι εύκολο να συνδυάσουμε δομές νευρωνικών δικτύων για κατασκευή πολύπλοκων υβριδικών μοντέλων όπως και μοντέλα για να καταγράφουν ταυτόχρονα ειδικά χαρακτηριστικά και τους παράγοντες.

4.5 Πιθανοί περιορισμοί των πολυεπίπεδων νευρωνικών δικτύων - Deep NN

Ερμηνεία αποτελεσμάτων.

Παρά την επιτυχία της, η βαθιά μάθηση είναι γνωστό ότι συμπεριφέρεται ως μαύρα κουτιά και παρέχοντας επεξηγημένες προβλέψεις φαίνεται να είναι ένα πραγματικά δύσκολο έργο. Ένα κοινό επιχείρημα κατά του βαθιών νευρωνικών δίκτυων είναι ότι τα κρυμμένα βάρη και οι ενεργοποιήσεις είναι γενικά μη ερμηνεύσιμα, όμως τα τελευταία χρόνια το πρόβλημα έχει λυθεί με την εμφάνιση των νευρικών μοντέλων προσοχής (Neural Attention Models). Παρόλα αυτά η διερμηνεία των μεμονωμένων νευρώνων εξακολουθεί να αποτελεί πρόκληση για όλες τις εφαρμογές των νευρωνικων μοντέλων.

Απαιτήσεις όγκου δεδομένων.

Ένας δεύτερος πιθανός περιορισμός είναι ότι η βαθιά μάθηση απαιτεί τα μεγάλο πλήθος δεδομένων προκειμένου να υποστηρίξει το ικανό εύρος παραμετροποίησής. Ωστόσο, σε σύγκριση με άλλους τομείς (όπως η γλώσσα ή η όραση) στα οποία τα σημασμένα με ετικέτα δεδομένα είναι σπάνια, είναι σχετικά εύκολο να συγκεντρώσουμε ικανό πλήθος δεδομένων στο

πλαίσιο των συστημάτων συστάσεων για έρευνα, ενώ επίσης διατίθενται μεγάλα σύνολα δεδομένων για ακαδημαϊκή χρήση.

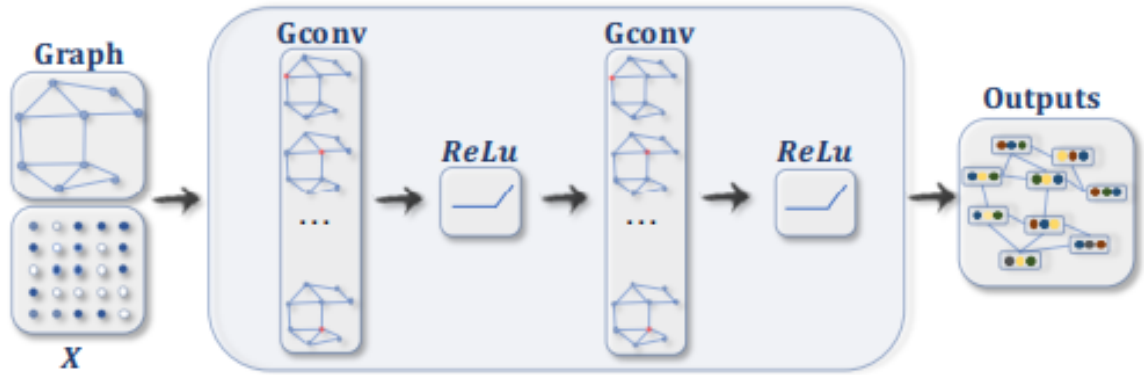
ΚΕΦΑΛΑΙΟ 5: Αρχιτεκτονικές Νευρωνικών Δικτύων Γράφων

5.1 Βασικά χαρακτηριστικά των νευρωνικών δικτύων γράφων.

Η μηχανική μάθηση με κατάλληλες αναπαραστάσεις δεδομένων λέγεται μάθηση αναπαραστάσεων (representation learning). Αυτά προέρχονται από τα χαρακτηριστικά κόμβων και την μήτρα γειτνίασης. Το GNN εξάγει νέες αναπαραστάσεις τις οποίες ονομάζουμε ενσωματώσεις (embeddings) για κάθε κόμβο (node level embeddings), όπως και σε επίπεδο γράφου (graph level embeddings).

Οι ενσωματώσεις χρησιμεύουν για λειτουργίες πρόβλεψης, όπου στην περίπτωση ενός κόμβου με άγνωστη ετικέτα (label). Το μέγεθος του πλήθους των ενσωματωμένων χαρακτηριστικών καθορίζεται από μια υπερπαράμετρο (hyper parameter).

Τα κύρια δομικά στοιχεία ενός GNN, τα επίπεδα μεταφοράς μηνυμάτων (message passing layers), συλλέγουν πληροφορία από τους γειτονικούς κόμβους, την συνδυάζουν σε ενσωματώσεις και ενημερώνουν με αυτές τα χαρακτηριστικά του κόμβου. Αυτή η διαδικασία λέγεται Graph Convolution.



Εικόνα 5.1: Συνέλιξη Γράφων στην διαδικασία μηχανικής μάθησης

Πηγή: A Comprehensive Survey on Graph Neural Networks. https://arxiv.org/abs/1901.00596

5.1.1 Τα δυνατά σημεία των συστημάτων συστάσεων με αλγόριθμους που στηρίζονται σε βαθιά νευρωνικά δίκτυα.

Μη γραμμικός μετασχηματισμός (non linear transformation)
Σε αντίθεση με τα γραμμικά μοντέλα, όπως η όπως η παραγοντοποίηση μήτρας, τα βαθιά νευρωνικά δίκτυα είναι ικανά μοντελοποίησης με μη γραμμικές ενεργοποιήσεις όπως relu, sigmoid, tanh κ.λπ.

Για παράδειγμα, η παραγοντοποίηση μήτρας διαμορφώνει την αλληλεπίδραση χρήστη-στοιχείου συνδυάζοντας γραμμικά τους χρήστες και τις λανθάνουσες παραμέτρους των αντικειμένων. Η μηχανή παραγοντοποίησης είναι μέλος μιας γραμμικής οικογένειας πολλαπλών παραλλαγών. Τα νευρωνικά δίκτυα είναι σε θέση να προσεγγίζουν οποιαδήποτε συνεχή λειτουργία με ορισμένη ακρίβεια μεταβάλλοντας τις επιλογές ενεργοποίησης και τους συνδυασμούς. Αυτή η ιδιότητα τα καθιστά ικανά να εφαρμοστούν σε πολύπλοκα προβλήματα αλληλεπίδρασης και να καθορίσουν με ακρίβεια την προτίμηση του χρήστη.

Εκμάθηση αναπαραστάσεων (representation learning)

Τα βαθιά νευρωνικά δίκτυα είναι αποτελεσματικά στην εκμάθηση των υποκείμενων επεξηγηματικών παραγόντων και χρήσιμων αναπαραστάσεων από δεδομένα εισόδου, ειδικά όταν αυτά είναι διαθέσιμα σε μεγάλους όγκους, όπως στην περίπτωση των συστημάτων συστάσεων.

Έτσι, είναι φυσική επιλογή να εφαρμόζουμε βαθιά νευρωνικά δίκτυα στην εκπροσώπηση της μάθησης σε μοντέλα συστάσεων. Τα πλεονεκτήματα της χρήσης βαθιών νευρωνικών δικτύων για να βοηθήσουν την εκμάθηση της αναπαράστασης είναι πολλαπλά:

Μείωση του χειρωνακτικού σχεδιασμού χαρακτηριστικών.

Τα βαθιά νευρωνικά δίκτυα επιτρέπουν την αυτόματη εκμάθηση χαρακτηριστικών από ανεπεξέργαστα δεδομένα χωρίς επίβλεψη ή με ημιεπίβλεψη επιτρέπει στα μοντέλα προτάσεων να περιλαμβάνουν ετερογενείς πληροφορίες περιεχομένου όπως κείμενο, εικόνες, ήχος και ακόμη και βίντεο. Τα δίκτυα βαθιάς μάθησης έχουν πραγματοποιήσει σημαντικές ανακαλύψεις από την επεξεργασία πολυμεσικών δεδομένων και έχουν σημαντικές δυνατότητες για μάθηση αναπαραστάσεων από διάφορες πηγές.

Μοντελοποίηση ακολουθιών (Sequence modeling)

Τα βαθιά νευρωνικά δίκτυα έχουν δείξει πολλά υποσχόμενα αποτελέσματα σε μια σειρά διαδοχικών εργασίων μοντελοποίησης σειριακών δεδομένων όπως στην αυτόματη μετάφραση, κατανόηση φυσικής γλώσσας, αναγνώριση ομιλίας, chatbots και πολλά άλλα. Τα RNN και CNN διαδραματίζουν κρίσιμο ρόλο σε αυτές τις εργασίες. Τα RNN το επιτυγχάνουν αυτά με καταστάσεις εσωτερικής μνήμης, ενώ το CNN το επιτυγχάνουν αυτό με κυλιόμενα φίλτρα στο χρόνο. Και τα δύο είναι ευρέως εφαρμόσιμα και ευπροσάρμοστα στην εξόρυξη δομών διαδοχικών δεδομένων Η μοντελοποίηση διαδοχικών σημάτων είναι ένα σημαντικό θέμα για την πρόβλεψη της δυναμικής συμπεριφοράς των χρηστών και της εξέλιξης των στοιχείων. Για παράδειγμα, η πρόβλεψη επόμενου αντικειμένου / καλαθιού και η πρόταση βάσει χαρακτηριστικών μιας συγκεκριμένης περιόδου σύνδεσης είναι δυο τυπικές εφαρμογές. Ως εκ τούτου, τα βαθιά νευρωνικά δίκτυα είναι το κατάλληλο εργαλείο για αυτό το διαδοχικό έργο εξόρυξης καθοριστικών χαρακτηριστικών.

Ευελιξία.

Οι τεχνικές βαθιάς μάθησης διαθέτουν υψηλή ευελιξία, ειδικά με την εμφάνιση πολλών δημοφιλών Frameworks βαθιάς μάθησης όπως Tensor*f*ow3, Keras4, MXnet6, DeepLearning4j7, PyTorch8, Θeano9, κ.λπ. Τα περισσότερα από αυτά τα εργαλεία έχουν αναπτυχθεί με αρθρωτό τρόπο, έχουν ενεργή κοινότητα και υποστήριξη.

Για παράδειγμα, είναι εύκολο να συνδυάσουμε δομές νευρωνικών δικτύων για κατασκευή πολύπλοκων υβριδικών μοντέλων όπως και μοντέλα για να καταγράφουν ταυτόχρονα ειδικά χαρακτηριστικά και τους παράγοντες.

5.1.2 Πιθανοί περιορισμοί των πολυεπίπεδων νευρωνικών δικτύων - Deep NN

Ερμηνεία αποτελεσμάτων.

Παρά την επιτυχία της, η βαθιά μάθηση είναι γνωστό ότι συμπεριφέρεται ως μαύρα κουτιά και παρέχοντας επεξηγημένες προβλέψεις φαίνεται να είναι ένα πραγματικά δύσκολο έργο. Ένα κοινό επιχείρημα κατά του βαθιών νευρωνικών δίκτυων είναι ότι τα κρυμμένα βάρη και οι ενεργοποιήσεις είναι γενικά μη ερμηνεύσιμα, όμως τα τελευταία χρόνια το πρόβλημα έχει λυθεί με την εμφάνιση των νευρικών μοντέλων προσοχής (Neural Attention Models). Παρόλα αυτά η διερμηνεία των μεμονωμένων νευρώνων εξακολουθεί να αποτελεί πρόκληση για όλες τις εφαρμογές των νευρωνικων μοντέλων.

Απαιτήσεις όγκου δεδομένων.

Ένας δεύτερος πιθανός περιορισμός είναι ότι η βαθιά μάθηση απαιτεί τα μεγάλο πλήθος δεδομένων προκειμένου να υποστηρίξει το ικανό εύρος παραμετροποίησής. Ωστόσο, σε σύγκριση με άλλους τομείς (όπως η γλώσσα ή η όραση) στα οποία τα σημασμένα με ετικέτα δεδομένα είναι σπάνια, είναι σχετικά εύκολο να συγκεντρώσουμε ικανό πλήθος δεδομένων στο πλαίσιο των συστημάτων συστάσεων για έρευνα, ενώ επίσης διατίθενται μεγάλα σύνολα δεδομένων για ακαδημαϊκή χρήση.

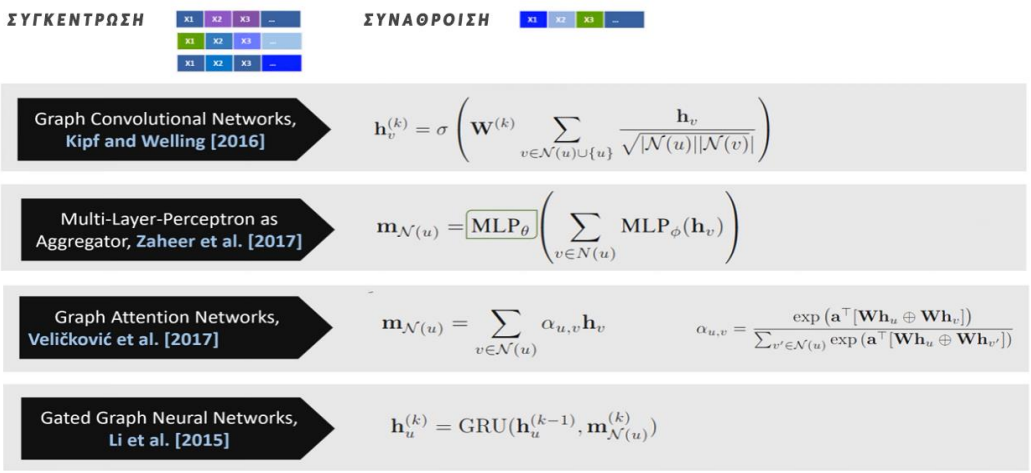
Βήμα 1: Για τον κόμβο 1, σύλλεξε πληροφορίες από γειτονικούς κόμβους $h_1^{(k)}$, $h_2^{(k)}$, $h_3^{(k)}$

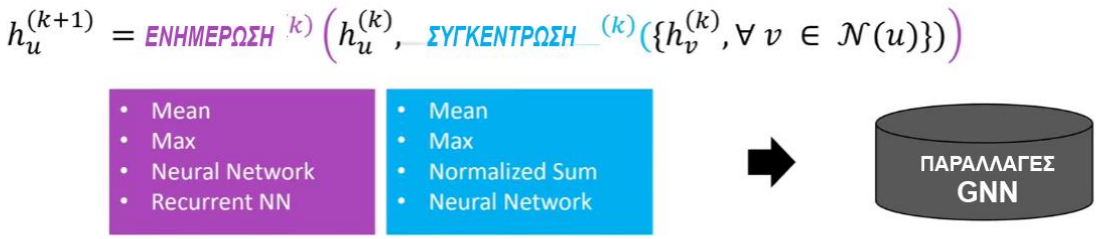
Βήμα 2: Συνάθροισε την παραπάνω πληροφορία (Aggregate)

Βήμα 3: Στο επόμενο βήμα / συνελικτικό επίπεδο k+1 ενημερώνουμε την κατάσταση του κόμβου 1 συνδυάζοντας την υπάρχουσα κατάσταση με την συναθροισμένη πληροφορία των γειτόνων $h_1^{(k+1)}$

Αυτά τα βήματα εκτελούνται για κάθε έναν από τους κόμβους του γράφου και σε κάθε νέο βήμα (neighborhood hop) η γειτονία ευρύνεται. Η διαδικασία συνάθροισης αντιστοιχεί στην εκμάθηση του πυρήνα ενός συνελικτικού νευρωνικού δικτύου CNN και οδηγεί μετά από τις ενημερώσεις στην αναπαράσταση του υπολογιστικού Γράφου που είναι μια αναδιάταξη των κόμβων ανάλογα την επίγνωση ενός ως προς τους υπολοίπους. Με την χρήση ικανού αριθμού επιπέδων οι ενσωματώσεις είναι όμοιες για όλους τους κόμβους.

5.2 Διαφορετικές υποκατηγορίες GNN





Εικόνα 5.2: Παραλλαγές αρχιτεκτονικών GNN

GCN

- Συνάθροιση γειτονικών χαρακτηριστικών ως κανονικοποιημένο άθροισμα των καταστάσεων.
- Ενημέρωση κατάστασης κάθε κόμβου ενσωματώνοντας τις πληροφορίες με συνάθροιση με την χρήση ενός αυτοβρόχου, που συμπεριλαμβάνεται στο άθροισμα (η ενημέρωση και η συνάθροιση συνδυάζονται σε έναν υπολογισμό).

Multilayer perceptrons

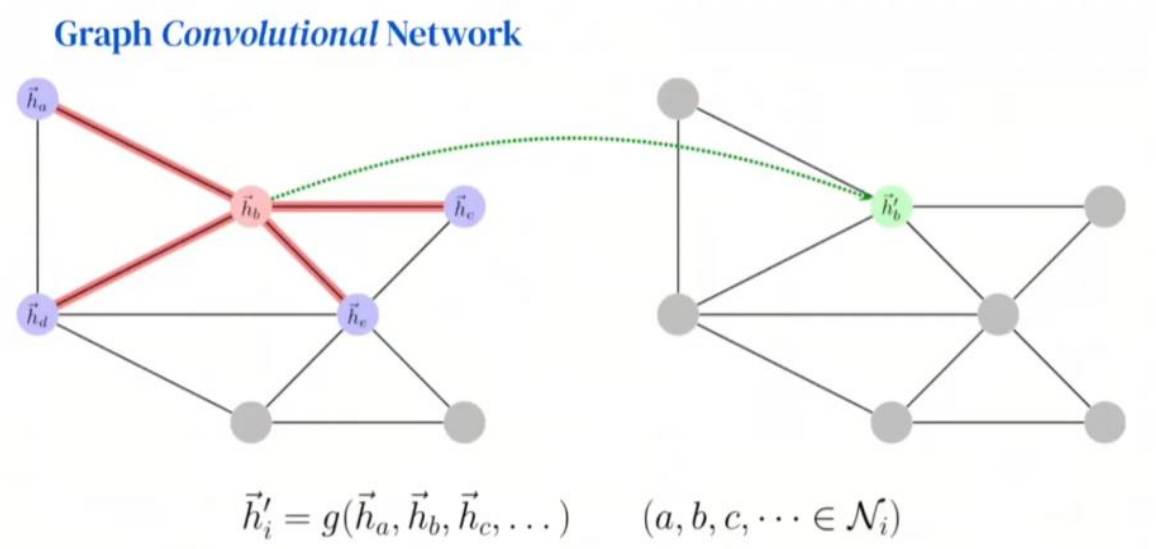
- Χρήση feed forward νευρωνικών δικτύων για την εκτέλεση της λειτουργίας ενσωμάτωσης. Ο αλγόριθμος βελτιστοποιεί τα βάρη για την αποτελεσματικότερη συνάθροιση των γειτονικών κόμβων.

Μηχανισμός Προσοχής στα GNNs

- Η σημασία των χαρακτηριστικών των γειτονικών κόμβων λαμβάνεται υπόψη κατά την ενσωμάτωση. Ως αποτέλεσμα, η ενημερωμένη ενσωμάτωση περιέχει περισσότερες πληροφορίες σχετικά με τα χαρακτηριστικά των γειτόνων.

Gated Graph Neural Networks

- Γίνεται χρήση μιας αναδρομικής μονάδας για ενημέρωση της κατάστασης του κόμβου επαναλαμβανόμενα με την πάροδο του χρόνου.



Εικόνα 5.3: Συνελικτικό δίκτυο Γράφων
Πηγή: Will Hamilton

Σε κάθε επανάληψη, εκπαιδεύουμε έναν γραμμικό ταξινομητή κόμβων.

Προκλήσεις στις συνελίξεις γραφημάτων

Επιθυμητές ιδιότητες για ένα συνελικτικό επίπεδο γραφήματος:

- Χαμηλή πολυπλοκότητα επεξεργασίας και μικρός χώρος αποθήκευσης στην μνήμη (~O (V + E))
- Σταθερός αριθμός παραμέτρων (ανεξάρτητα από το μέγεθος γραφήματος).
- Τοπικότητα: Επικέντρωση του ενδιαφέροντος σε γειτονίες κόμβων.
- Καθορισμός ειδικού βάρους σε συγκεκριμένους κόμβους.
- Εφαρμογή σε επαγωγικά προβλήματα.

Η διαδικασία εφαρμόζεται σε γράφους που περιγράφονται με πολυωνυμικές συναρτήσεις. Υπάρχουν 3 γενικά μοντέλα για την επίλυση του προβλήματος: GCNs, GATs και MPNNs.

GCN

Συνδυασμός ενημέρωσης και συνάθροισης

- Για έναν Γράφο με μη κατευθυνόμενες ακμές και χωρίς βάρη έχουμε

$$\mathbf{A}_{ij} = \mathbf{A}_{ji} = \begin{cases} \boldsymbol{i \leftrightarrow j} & \mathbf{1} \\ \boldsymbol{\alpha\lambda\lambda\iota\acute{\omega}\varsigma} & 0 \end{cases} \qquad (5.1)$$

- Στη συνέχεια, μπορούμε να συναθροίσουμε γείτονες πολλαπλασιάζοντας με τον πίνακα γειτνίασης.

$$\mathbf{H}' = \boldsymbol{\sigma}(\mathbf{AHW}) \quad (5.2)$$

όπου το W είναι ένας εκπαιδευόμενος γραμμικός μετασχηματισμός ανά κόμβο, και το σ είναι μια μη-γραμμικότητα.

- Μερικά πράγματα πρέπει να διορθωθούν ...Πρώτον, αυτός ο κανόνας ενημέρωσης απορρίπτει τον κεντρικό κόμβο. Με μια απλή διόρθωση: $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$.

$$\mathbf{H}' = \boldsymbol{\sigma}(\widetilde{\mathbf{A}}\mathbf{H}\mathbf{W}) \quad (5.3)$$

Ο κανόνας ενημέρωσης ανά κόμβο μπορεί πλέον να ξαναγραφεί ως:

$$\vec{\boldsymbol{h}}_{\boldsymbol{i}}' = \boldsymbol{\sigma}\left(\sum_{\boldsymbol{j}\in\mathcal{N}_{\boldsymbol{i}}} \mathbf{W}\vec{\boldsymbol{h}}_{\boldsymbol{j}}\right) \quad (5.4)$$

Ο κανόνας ενημέρωσης mean-pooling

Ο πολλαπλασιασμός με το A μπορεί να αυξήσει την κλίμακα των χαρακτηριστικών εξόδου. Πρέπει να ομαλοποιήσουμε κατάλληλα, π.χ. με

$$\mathbf{H}' = \boldsymbol{\sigma}\left(\mathbf{D}^{-1}\widetilde{\mathbf{A}}\mathbf{H}\mathbf{W}\right) \quad (5.5)$$

Όπου $\mathbf{D}$ είναι η μήτρα συνδέσεων του $\mathbf{A}$. Έτσι φτάνουμε στον κανόνα ενημέρωσης mean pooling:

$$\vec{\boldsymbol{h}}_{\boldsymbol{i}}' = \boldsymbol{\sigma}\left(\sum_{\boldsymbol{j}\in\mathcal{N}_{\boldsymbol{i}}} \frac{\mathbf{1}}{|\mathcal{N}_{\boldsymbol{i}}|}\mathbf{W}\vec{\boldsymbol{h}}_{\boldsymbol{j}}\right) \quad (5.6)$$

5.2.1 GCN (Kipf & Welling, ICLR 2017)

GCN: Ο μαθηματικός τύπος

$$\boldsymbol{H}^{(\boldsymbol{l}+\mathbf{1})} = \boldsymbol{\sigma}\left(\boldsymbol{A}\boldsymbol{H}^{(\boldsymbol{l})}\boldsymbol{W}^{(\boldsymbol{l})}\right) \quad (5.7)$$

Η Ομαλοποιημένη μήτρα γειτνίασης - normalized adjacency matrix.

Όπου $\boldsymbol{A}$ είναι η μήτρα γειτνίασης του $\boldsymbol{G}$ και $\boldsymbol{D} = \mathbf{diag}\,(\boldsymbol{d})$ με $\boldsymbol{d}(\boldsymbol{i})$ ο βαθμός του κόμβου $\boldsymbol{i}$.
Για ένα συνεκτικό Γράφο $\boldsymbol{G}$ έχουμε:

Αν αντί αυτού χρησιμοποιούμε συμμετρική ομαλοποίηση:

$$\mathbf{H}' = \boldsymbol{\sigma}\left(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\right) \quad (5.8)$$

Καταλήγουμε στην συνάρτηση ενημέρωσης του GCN που αποτελεί σήμερα το πιο δημοφιλές επίπεδο συνέλιξης γράφων. Ως προς κάθε κόμβο γράφεται ως εξής:

$$\vec{\boldsymbol{h}}_{\boldsymbol{i}}' = \boldsymbol{\sigma}\left(\sum_{\boldsymbol{i}\in\mathcal{N}_{\boldsymbol{j}}} \frac{\mathbf{1}}{\sqrt{|\mathcal{N}_{\boldsymbol{i}}||\mathcal{N}_{\boldsymbol{j}}|}}\mathbf{W}\vec{\boldsymbol{h}}_{\boldsymbol{j}}\right) \quad (5.9)$$

Απλό και ισχυρό, αλλά μειονεκτεί ως προς την επαγωγική μάθηση των βαρών.

MLP

Συσσωματώσεις αλληλουχίας κόμβων

Η εξομάλυνση γειτονιάς (Neighborhood normalization) χρησιμοποιείται για την βελτίωση της απόδοσης του GNN, όμως μπορούμε να βελτιώσουμε και τον τελεστή AGGREGATE; Υπάρχει ίσως κάτι πιο εξελιγμένο από το απλό άθροισμα των γειτονικών ενσωματώσεων. Η λειτουργία συσσωμάτωσης γειτονιάς είναι βασικά μια καθορισμένη συνάρτηση. Μας δίνεται ένα σύνολο ενσωματώσεων γειτόνων $\{\mathbf{h}_v, \forall v \in \mathcal{N}(u)\}$ και πρέπει να το αναπαραστήσουμε με ένα διάνυσμα $\mathbf{m}_{\mathcal{N}(u)}$.

Το γεγονός ότι το $\{\mathbf{h}_v, \forall v \in \mathcal{N}(u)\}$ είναι ένα σύνολο είναι στην πραγματικότητα πολύ σημαντικό: δεν υπάρχει προκαθορισμένη διάταξη γειτόνων κόμβων και οποία συνάρτηση συνάθροισης ορίσουμε πρέπει να εφαρμόζεται σε όλες τις πιθανές μεταθέσεις των κόμβων (permutation invariance).

Ομαδοποίηση αλληλουχίας κόμβων

Μια βασισμένη προσέγγιση για τον καθορισμό μιας συνάρτησης συνάθροισης βασίζεται στη θεωρία των αμετάθετων νευρωνικών δικτύων (permutation invariant neural networks). Για παράδειγμα, οι Zaheer et al. [2017] προτείνουν μια συνάρτηση συνάθροισης που προσεγγίζει καθολικά το σύνολο των κόμβων

$$\mathbf{m}_{\mathcal{N}(u)} = \mathbf{MLP}_{\theta}\left(\sum_{v \in N(u)} \mathbf{MLP}_{\phi}(\mathbf{h}_v)\right) \textit{(5.10)}$$

όπου ως συνήθως χρησιμοποιούμε πολύ-επίπεδα perceptron που παραμετροποιούνται από εκπαιδευόμενη παράμετρο θ. Σύμφωνα με τους Zaheer et al. [2017] οποιαδήποτε permutation invariant συνάρτηση που αντιστοιχεί ένα σύνολο ενσωματώσεων σε μία ενιαία ενσωμάτωση μπορεί να προσεγγιστεί με καθορισμένη ακρίβεια με την παραπάνω Εξίσωση.

Κάθε ορισμός προσεγγίσεων συγκέντρωσης με βάση την παραπάνω εξίσωση οδηγεί σε μικρές αυξήσεις στην απόδοση, ενώ εισάγουν κίνδυνο overfitting, ανάλογα με το βάθος των MLP που χρησιμοποιούνται. Είναι συνηθισμένο να χρησιμοποιούνται MLP που έχουν μόνο ένα κρυφό επίπεδο, δεδομένου ότι αυτά τα μοντέλα είναι επαρκή για να ικανοποιήσουν τη θεωρία, αλλά δεν είναι τόσο υπερ-παραμετροποιημένα, ώστε να διατρέχουν κίνδυνο overfitting.

5.3 Graph Attention Networks (GATs)

Μια επίσης δημοφιλής στρατηγική συνάθροισης συνόλων που βελτιώνει το επίπεδο συγκέντρωσης στα GNNs είναι η προσοχή [Bahdanau et al., 2015]. Η βασική ιδέα είναι να δοθεί ένα ατομικό βάρος ή αξία σε κάθε γειτονικό κόμβο, το οποίο χρησιμοποιείται για να σταθμίσει την επιρροή αυτού του γείτονα κατά τη διάρκεια του βήματος συγκέντρωσης. Το πρώτο μοντέλο GNN που εφάρμοσε αυτό το στυλ προσοχής ήταν αυτό των Veličković et al. [2018] Το Graph Attention Network (GAT), το οποίο χρησιμοποιεί βάρη προσοχής για να καθορίσει ένα σταθμισμένο άθροισμα των γειτόνων συμβολίζεται με την παρακάτω συνάρτηση:

$$\mathbf{m}_{\mathcal{N}(u)} = \sum_{v \in \mathcal{N}(u)} \alpha_{u,v} \mathbf{h}_v \quad \textit{(5.11)}$$

όπου $\alpha_{u,v}$ υποδηλώνει την προσοχή στον γείτονα $v \in \mathcal{N}(u)$ όταν συγκεντρώνουμε πληροφορίες στον κόμβο u. Στη αρχική πρόταση GAT, τα βάρη προσοχής ορίζονται ως

$$\alpha_{u,v} = \frac{\exp\left(\mathbf{a}^{\top}[\mathbf{W}\mathbf{h}_u \oplus \mathbf{W}\mathbf{h}_v]\right)}{\sum_{v' \in \mathcal{N}(u)} \exp\left(\mathbf{a}^{\top}[\mathbf{W}\mathbf{h}_u \oplus \mathbf{W}\mathbf{h}_{v'}]\right)} \quad (5.12)$$

όπου το a είναι ένα εκπαιδευμένο διάνυσμα προσοχής, το W είναι εκπαιδευμένη μήτρα και το$\oplus$ υποδηλώνει τη λειτουργία συνένωσης.

Ο υπολογισμός προσοχής με χρήση GAT λειτουργεί αποτελεσματικά με δεδομένα σε μορφή γράφου. Μια δημοφιλή παραλλαγή μοντέλου προσοχής είναι το μοντέλο διπλής προσοχής

$$\boldsymbol{\alpha_{u,v}} = \frac{\mathbf{exp}\ (\mathbf{h}_u^\top \mathbf{W}\mathbf{h}_v)}{\sum_{v' \in \mathcal{N}(u)}\ \mathbf{exp}\ (\mathbf{h}_u^\top \mathbf{W}\mathbf{h}_{v'})} \quad (\mathbf{5.13})$$

Όπως και παραλλαγές επιπέδων προσοχής με χρήση MLPs

$$\boldsymbol{\alpha_{u,v}} = \frac{\mathbf{exp}\left(\mathbf{MLP}\ (\mathbf{h}_u, \mathbf{h}_v)\right)}{\sum_{v' \in \mathcal{N}(u)}\ \mathbf{exp}\left(\mathbf{MLP}\ (\mathbf{h}_u, \mathbf{h}_{v'})\right)} \quad (\mathbf{5.14})$$

5.4 Gated Graph Networks

Η αναδρομική μονάδα πύλης (GRU) προτάθηκε από τους Cho et al. [2014] για να κάνει κάθε επαναλαμβανόμενη μονάδα να προσαρμόζει προσαρμοστικά τις εξαρτήσεις διαφορετικών χρονικών κλιμάκων. Ομοίως με τη μονάδα LSTM, η GRU διαθέτει μονάδες πύλης που ρυθμίζουν τη ροή πληροφοριών μέσα στη μονάδα, ωστόσο, χωρίς να έχουν ξεχωριστά κελιά μνήμης.

Η ενεργοποίηση $\boldsymbol{h_t^j}$ του GRU στο χρόνο t είναι μια γραμμική παρεμβολή μεταξύ της προηγούμενης ενεργοποίησης $\boldsymbol{h_{t-1}^j}$ και της υποψήφιας ενεργοποίησης $\boldsymbol{\tilde{h}_t^j}$:

$$\boldsymbol{h_t^j} = \left(\mathbf{1} - \boldsymbol{z_t^j}\right)\boldsymbol{h_{t-1}^j} + \boldsymbol{z_t^j}\boldsymbol{\tilde{h}_t^j} \quad \textit{(5.15)}$$

όπου μια πύλη ενημέρωσης $\boldsymbol{z_t^j}$ αποφασίζει πόσο ενημερώνει η μονάδα την ενεργοποίηση ή το περιεχόμενό της. Η πύλη ενημέρωσης υπολογίζεται από την συνάρτηση

$$\boldsymbol{z_t^j} = \boldsymbol{\sigma}(\boldsymbol{W_z}\mathbf{x_t} + \boldsymbol{U_z}\mathbf{h_{t-1}})^j \quad \textit{(5.16)}$$

Αυτή η διαδικασία λήψης γραμμικού αθροίσματος μεταξύ της υπάρχουσας κατάστασης και της νέας υπολογισμένης κατάστασης μοιάζει με μια μονάδα LSTM. Η GRU, ωστόσο, δεν διαθέτει μηχανισμό ελέγχου του βαθμού στον οποίο εκτίθεται η κατάστασή του, αλλά εκθέτει ολόκληρη την κατάσταση κάθε φορά.

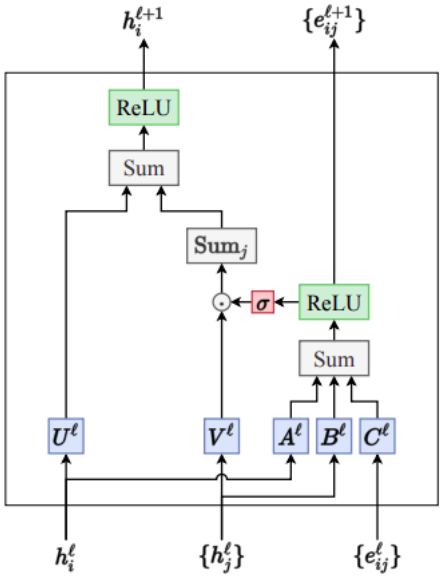
Η υποψήφια ενεργοποίηση $\boldsymbol{\tilde{h}_t^j}$ υπολογίζεται παρόμοια με εκείνη της παραδοσιακής αναδρομικής μονάδας ως εξής:

$$\boldsymbol{\tilde{h}_t^j} = \mathbf{tanh}\left(\boldsymbol{W}\mathbf{x_t} + \boldsymbol{U}(\mathbf{r_t} \odot \mathbf{h_{t-1}})\right)^j \quad \textit{(5.17)}$$

όπου το $\mathbf{r_t}$ είναι ένα σύνολο πυλών επαναφοράς και το $\odot$ είναι ένας πολλαπλασιασμός ανά στοιχείο. ^ 1 Όταν είναι είναι κλειστή ($\boldsymbol{r_t^j}$τείνει στο 0), η πύλη επαναφοράς κάνει τη μονάδα να ενεργεί σαν να διαβάζει το πρώτο σύμβολο μιας ακολουθίας εισόδου, επιτρέποντάς της να ξεχάσει την προηγούμενη υπολογισμένη κατάσταση.

Η πύλη επαναφοράς $\boldsymbol{r_t^j}$ υπολογίζεται παρόμοια με την πύλη ενημέρωσης:

$$\boldsymbol{r_t^j} = \boldsymbol{\sigma}(\boldsymbol{W_r}\mathbf{x_t} + \boldsymbol{U_r}\mathbf{h_{t-1}})^j \quad \textit{(5.18)}$$



Εικόνα 5.4: Γραφική αναπαράσταση επιπέδου GRU του Gated GCN
Πηγη: Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2021. *Empirical evaluation of gated recurrent neural networks on sequence modeling* https://export.arxiv.org/pdf/1412.3555

Οι αρχιτεκτονικές πύλης (gated) χρησιμοποιούνται για τη βελτίωση της σταθερότητας και της μαθησιακής ικανότητας των επαναλαμβανόμενων νευρωνικών δικτύων (RNNs). Συγκεκριμένα, ένας τρόπος για να προβάλετε τον αλγόριθμο μετάδοσης μηνυμάτων GNN είναι ότι η συνάρτηση συνάθροισης λαμβάνει μια παρατήρηση από τους γείτονες, η οποία στη συνέχεια χρησιμοποιείται για την ενημέρωση της κρυφής κατάστασης κάθε κόμβου. Με αυτόν τον τρόπο μπορούμε να εφαρμόσουμε απευθείας, μεθόδους που χρησιμοποιούνται για την ενημέρωση της κρυφής κατάστασης των αρχιτεκτονικών RNN βάσει παρατηρήσεων. Μια από τις πρώτες αρχιτεκτονικές GNN [Li et al., 2015] ορίζει τη συνάρτηση ενημέρωσης ως

$$\mathbf{h}_u^{(k)} = \mathbf{GRU}\left(\mathbf{h}_u^{(k-1)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right) \quad (5.19)$$

Στην συνέχεια ασχολούμαστε με την εξέταση αρχιτεκτονικών νευρωνικών δικτύων γράφων ως προς την παραγωγή μοντέλων χρήστη, αντικειμένων, κοινωνικής επιρροής και την ενσωμάτωση τους.

5.4.1 Αρχιτεκτονική GraphRec

Εισαγωγή στην αρχιτεκτονική

Η αρχιτεκτονική του προτεινόμενου μοντέλου φαίνεται στο Σχήμα 2. Το μοντέλο αποτελείται από τρία στοιχεία: μοντελοποίηση χρήστη, μοντελοποίηση στοιχείων και πρόβλεψη βαθμολογίας. Το πρώτο στοιχείο αφορά εκμάθηση λανθάνων παραγόντων των χρηστών. Δεδομένου της εισόδου από δύο διαφορετικά γραφήματα, δηλαδή, ένα γράφημα κοινωνικών δικτύων και ένα γράφημα στοιχείων χρήστη, μπορούμε να μάθουμε τις διανυσματικές αναπαραστάσεις χρηστών από διαφορετικά χαρακτηριστικά. Επομένως, εισάγονται δύο συγκεντρώσεις για να επεξεργαστούν αντίστοιχα αυτά τα δύο διαφορετικά γραφήματα. Το ένα είναι η συγκέντρωση στοιχείων, η οποία μπορεί να χρησιμοποιηθεί για την αναπαράσταση των χρηστών μέσω από τις αλληλεπιδράσεων μεταξύ χρηστών και αντικειμένων στο γράφημα στοιχείων χρήστη (ή κοινωνικό δίκτυο). Το άλλο είναι η συσσωμάτωση σχέσεων μεταξύ των

χρηστών στο κοινωνικό γράφο, η οποία μπορεί να βοηθήσει το μοντέλο των χρηστών λαμβάνοντας στοιχεία από το κοινονικό τους προφίλ. Στη συνέχεια, είναι εύκολη η εξαγωγή χαρακτηριστικών για τις παραμέτρους του χρήστη συνδυάζοντας πληροφορίες τόσο από τον χώρο των αντικειμένων όσο και από τον κοινωνικό χώρο. Η δεύτερη συνιστώσα του GRAPHREC είναι η μοντελοποίηση αντικειμένων, η οποία είναι η εκμάθηση λανθάνουσων χαρακτηριστικών των αντικειμένων. Προκειμένου να ληφθούν υπόψη τόσο οι αλληλεπιδράσεις όσο και η βαθμολόγηση στο γράφημα χρήστων - αντικειμένων, κάνουμε ενσωμάτωση των χρηστών και των απόψεών τους στη μοντελοποίηση των στοιχείων. Στο τρίτο στάδιο προβλέπουμε τις παραμέτρους του μοντέλου μέσω μηχανικής μάθησης, ενσωματώνοντας στοιχεία μοντέλου χρήστη και αντικειμένων Στη συνέχεια, θα αναλύσουμε κάθε στοιχείο του μοντέλου.

Μοντελοποίηση χρηστών

Η μοντελοποίηση χρηστών στοχεύει στην εκμάθηση λανθανουσών παραγόντων χρηστών, ως $\mathbf{h}_i \in \mathbb{R}^d$ για τον χρήστη u_i. Το πρόβλημα στην συνέχεια εντοπίζεται στο πως να συνδυάσουμε εγγενώς το γράφημα χρήστη και το γράφημα κοινωνικής δικτύωσης.

Για να αντιμετωπίσουμε αυτήν την πρόκληση, χρησιμοποιούμε πρώτα δύο τύπους συνάθροισης για να μάθουμε παράγοντες από δύο γραφήματα, Η πρώτη συγκέντρωση, που δηλώνεται ως συγκέντρωση στοιχείων, χρησιμοποιείται για την εκμάθηση του κρυφού παράγοντα $\mathbf{h}_i^I \in \mathbb{R}^d$από τον γράφο αλληλεπίδρασης χρηστών-αντικειμένων. Η δεύτερη συνάθροιση είναι η κοινωνική συγκέντρωση όπου ο λανθάνων παράγοντας $\mathbf{h}_i^S \in \mathbb{R}^d$του κοινωνικού χώρου μαθαίνεται από το γράφημα κοινωνικής δικτύωσης.

Στη συνέχεια, αυτοί οι δύο παράγοντες συνδυάζονται για να σχηματίσουν τους τελικούς λανθάνοντες παράγοντες $\mathbf{h}_i$. Στη συνέχεια, εισάγουμε τη συνάθροιση στοιχείων, την κοινωνική συγκέντρωση και τον τρόπο συνδυασμού των λανθάνουσών παραγόντων των χρηστών τόσο από τον χώρο στοιχείων όσο και από τον κοινωνικό χώρο.

Συγκέντρωση στοιχείων (item aggregation)

Το γράφημα στοιχείων-χρήστη περιέχει αλληλεπιδράσεις μεταξύ χρηστών και αντικειμένων, αλλά και προτιμήσεις χρηστών (ή βαθμολογίες βαθμολογίας) για αντικείμενα, παρέχουμε μια καταρχήν προσέγγιση για την από ταυτόχρονη λήψη αλληλεπιδράσεων και προτιμήσεων για την εκμάθηση λανθανόντων παραγόντων των χρηστών $\mathbf{h}_i^I$, που χρησιμοποιούνται για τη μοντελοποίηση των λανθανόντων παραγόντων του χρήστη μέσω αλληλεπιδράσεων στο γράφημα του αντικειμένων-χρήστη.

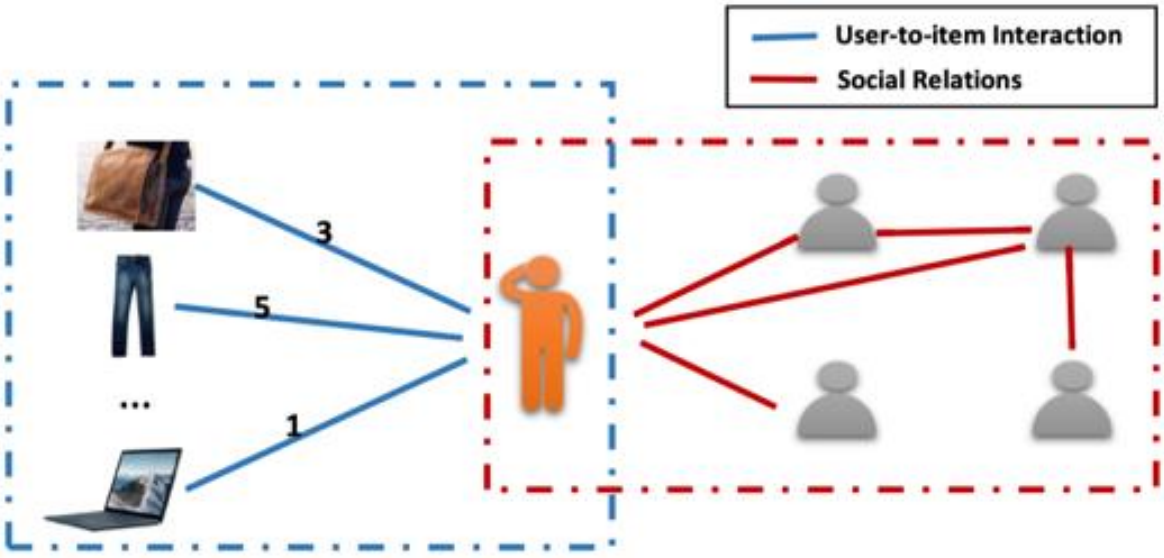
Γενικές κατευθύνσεις

Γράφος στοιχείου-χρήστη

- Αλληλεπιδράσεις μεταξύ χρηστών και στοιχείων.
- Γνώμες χρηστών για στοιχεία (δηλ. Ρητή ανατροφοδότηση, βαθμολογίες).

Γράφημα χρήστη-χρήστη

- Οι κοινωνικές σχέσεις έχουν ετερογενή βάρη.
- Ισχυροί και αδύναμοι δεσμοί συνυπάρχουν.
- Οι χρήστες είναι πιθανό να μοιράζονται περισσότερα παρόμοια γούστα με ισχυρούς δεσμούς παρά αδύναμους δεσμούς.



Εικόνα 5.5: Συγκέντρωση στοιχείων από δύο Γράφους
Πηγή: Fan et al. Graph Neural Networks for Social Recommendation.WWW2019

Ορολογία

- Γράφος: $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$
- Χαρακτηριστικά κόμβου: $\mathbf{H} = \{\vec{\boldsymbol{h}}_1, \vec{\boldsymbol{h}}_2, \dots, \vec{\boldsymbol{h}}_N\}; \vec{\boldsymbol{h}}_i \in \mathbb{R}^F$
- Μήτρα γειτνίασης: $\mathbf{A} \in \mathbb{R}^{N \times N}$
- Γειτονιά: $\boldsymbol{\mathcal{N}}_i = \{\boldsymbol{j} \mid \boldsymbol{i} = \boldsymbol{j} \vee \mathbf{A}_{ij} \neq 0\}$
- Χαρακτηριστικά ακμής: $\vec{\boldsymbol{e}}_{ij} \in \mathbb{R}^{F'}; (\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{E}$

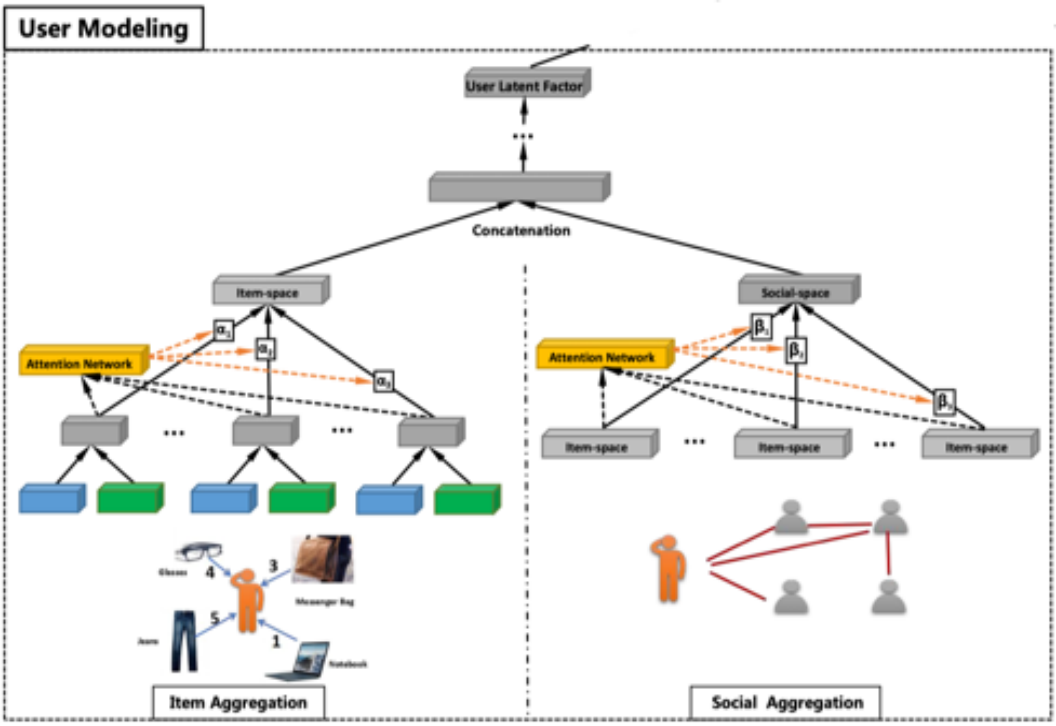
Οι 2 γράφοι παρέχουν πληροφορίες χρηστών από διαφορετικές οπτικές γωνίες

Συγκέντρωση στοιχείων
Χώρος στοιχείων: αξιοποιούμε τις αλληλεπιδράσεις στοιχείων-χρήστη για να λάβουμε τις αναπαραστάσεις χρηστών:

$$\mathbf{h}_i^I = \boldsymbol{\sigma}\big(\mathbf{W} \cdot \textbf{ Aggre }_{\boldsymbol{items}}(\{\mathbf{x}_{ia}, \forall \boldsymbol{a} \in \boldsymbol{C}(\boldsymbol{i})\}) + \mathbf{b}\big) \quad \textit{(5.20)}$$

Κοινωνική Συγκέντρωση
Κοινωνικός χώρος: χρησιμοποιούμε τις κοινωνικές σχέσεις για να λάβουμε αναπαραστάσεις χρηστών:

$$\mathbf{h}_i^S = \boldsymbol{\sigma}\left(\mathbf{W} \cdot \textbf{ Aggre }_{\textbf{neigbhors}}(\{\mathbf{h}_o^I, \forall \boldsymbol{o} \in \boldsymbol{N}(\boldsymbol{i})\}) + \mathbf{b}\right) \quad \textit{(5.21)}$$



Εικόνα 5.6 Συνδυασμός μοντέλων χρήστη και κοινωνικού δικτύου
Πηγή: Social rec comparison and methodology https://arxiv.org/pdf/2011.04797.pdf

Συγκέντρωση χρηστών

- Εξετάζουμε τόσο τις αλληλεπιδράσεις όσο και τις απόψεις για να λάβουμε αναπαραστάσεις στοιχείων

Πρόβλεψη βαθμολογίας

- Τροφοδοτούμε τη συνένωση των αναπαραστάσεων χρήστη και στοιχείων σε νευρωνικό δίκτυο
- Δίκτυο προσοχής για τη διαφοροποίηση των σημασία βάρους
- $\mathbf{z}_j = \sigma(\mathbf{W} \cdot \text{Aggre } \mathbf{users}(\{\mathbf{f}_{jt}, \forall \boldsymbol{t} \in \boldsymbol{B}(\boldsymbol{j})\}) + \mathbf{b})$ *(5.22)*
- Αναπαράσταση γνώσης γνώμης μιας αλληλεπίδρασης

Dataset της υλοποίησης του GraphRec

Epinion dataset (με /και χωρίς timestamps)

rating.mat

Το rating.mat περιλαμβάνει τις πληροφορίες αξιολόγησης. Υπάρχουν πέντε στήλες: userid, productid, categoryid, βαθμολογία, χρησιμότητα,]

****

Για παράδειγμα η γραμμή

(1,2,3,4,5)

Σημαίνει ότι ο χρήστης 1 δίνει βαθμολογία 4 στο προϊόν 2 από την κατηγορία 3. Η χρησιμότητα αυτής της βαθμολογίας είναι 5.

trustnetwork.mat

Το trustnetwork.mat περιλαμβάνει τις σχέσεις εμπιστοσύνης μεταξύ των χρηστών. Υπάρχουν δύο στήλες και οι δύο είναι userid.

****

Για παράδειγμα η γραμμή:

(1,2)

Σημαίνει ότι ο χρήστης 1 εμπιστεύεται τον χρήστη 2

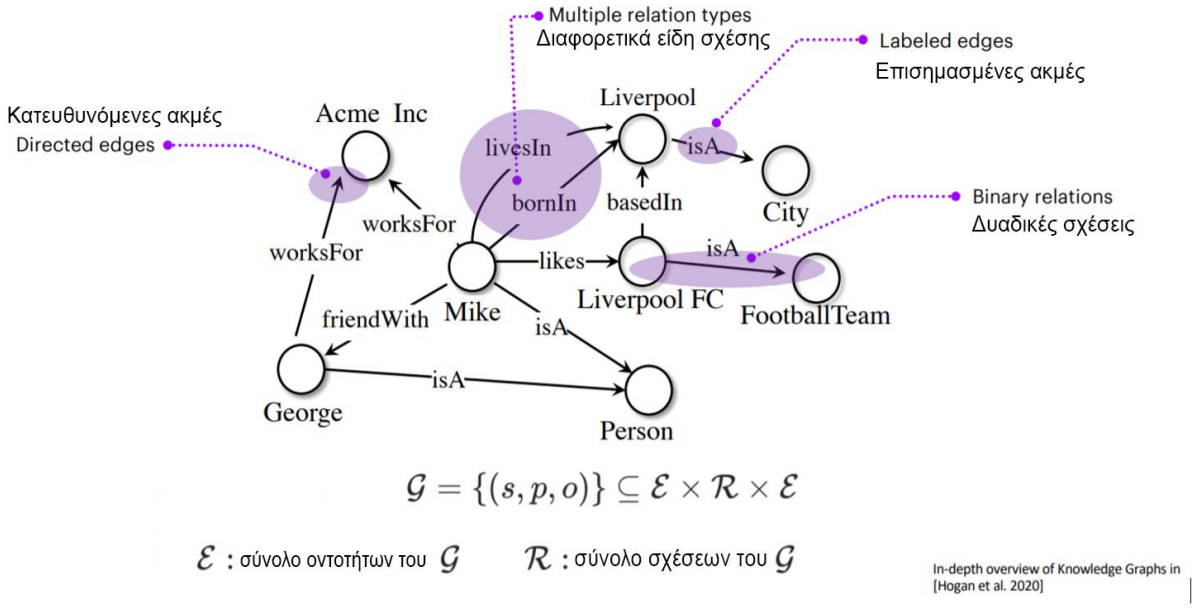
ΚΕΦΑΛΑΙΟ 6. Συστήματα συστάσεων με Γράφους γνώσης - Knowledge Graph-based Recommendations

Οι γνωσιακοί Γράφοι (KG) υιοθετούν την Υπόθεση του Open World (OWA): η απουσία ενός γεγονότος δεν σημαίνει ότι το γεγονός είναι ψευδές. Απλά δεν ξέρουμε.

Τα Συστήματα σύστασης βάσει KG (KG-RecSys) χρησιμοποιούν υπάρχουσα γνώση για προϊόντα και χρήστες για να δημιουργήσουν ένα κριτήριο βασισμένο στη γνώση για τη δημιουργία συστάσεων. Ένα KG-RecSys δεν απαιτεί αρχικό μεγάλο όγκο δεδομένων, καθώς οι συστάσεις του είναι ανεξάρτητες από τις αξιολογήσεις του χρήστη. Προτείνει στοιχεία με βάση τις προτιμήσεις του χρήστη αξιολογώντας τα προϊόντα που ικανοποιούν τις ανάγκες του.

6.1 Γνωσιακοί Γράφοι

Οι γνωσιακοί γράφοι (KG) αντιπροσωπεύουν τις βάσεις γνώσης (KB) ως κατευθυνόμενο γράφημα του οποίου οι κόμβοι και οι ακμές αντιπροσωπεύουν οντότητες και σχέσεις μεταξύ οντοτήτων, αντίστοιχα. Οι σχέσεις οργανώνονται με τριπλές σχέσεις (head, relation, tail) (h, r, t).



Εικόνα 6.1: Αναπαράσταση Γράφου Γνώσης
Πηγή: Hogan et al. 2020

Γράφημα γνώσης (KG):	**Οφέλη ως προς τα συστήματα σύστασης**
• Ιστορικό γνώσεων σχετικά με τα αντικείμενα	• Περιορίζουν τον χώρο αναζήτησης
• Πλούσια σημασιολογία & σχέσεις	• Εξερευνούν τα ενδιαφέροντα των χρηστών
	• Προσφέρουν εξηγήσιμα αποτελέσματα

Πρόβλεψη διασύνδεσης / Τριπλή ταξινόμηση

- Βάρος διασύνδεσης ανάλογο της πιθανότητας να ισχύει..

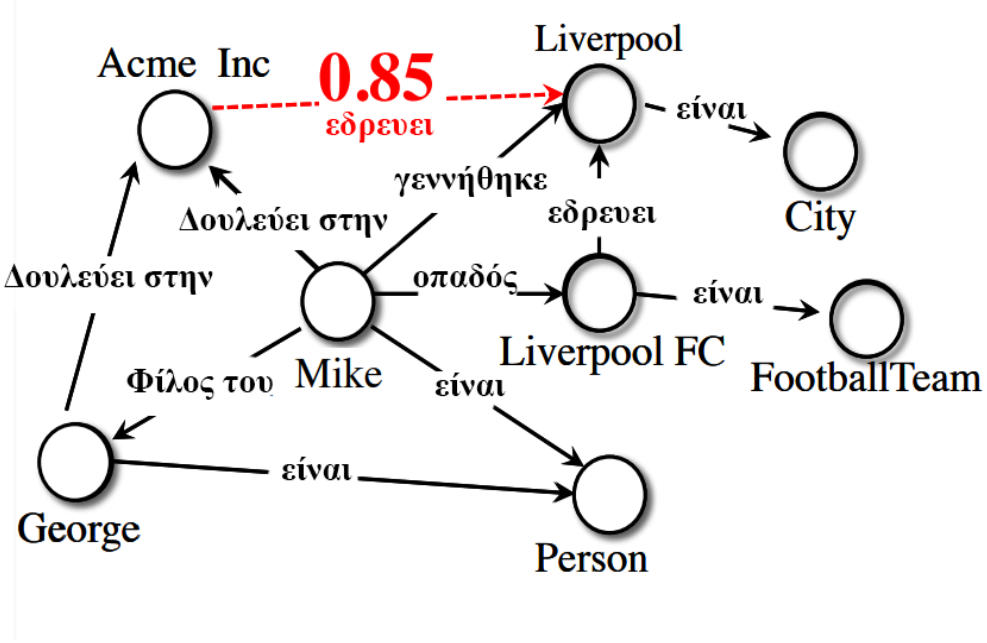
- Ολοκλήρωση γραφήματος γνώσεων.
- Σύσταση περιεχομένου.
- Απάντηση ερωτήσεων.

Πρόβλεψη γειτόνων

- Πρόβλημα Εκμάθησης Θέσης (Learning rank problem).
- Μετρικές ανάκτησης πληροφοριών.

Τριπλή Ταξινόμηση

- Δυαδική εργασία ταξινόμησης.
- Μετρικές δυαδικής ταξινόμησης.
- Το σετ δοκιμών απαιτεί εξακριβωμένα θετικά και αρνητικά.



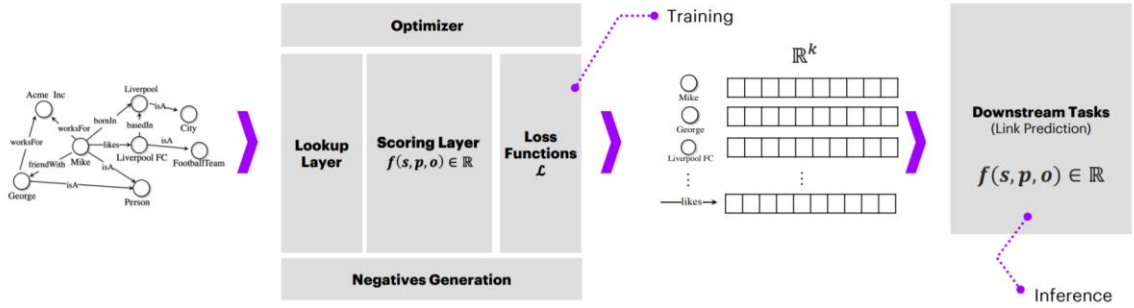
Εικόνα 6.2: Αναπαράσταση σχέσεων Γράφου Γνώσης
Πηγή: Hogan et al. 2020

Γιατί KG σε συστήματα συστάσεων:

- Για ένα σύστημα συστάσεων, το βασικό βήμα είναι να εξαγάγουμε πολλά υποψήφια στοιχεία να τα αξιολογήσουμε και τους τέλος να τα στείλουμε στους χρήστες.
- Τα KG είναι χτισμένα ως σημασιολογικά δίκτυα, που σημαίνει ότι θα μπορούσαμε να υπολογίσουμε τις σημασιολογικές ομοιότητες μεταξύ των στοιχείων. Και σημασιολογικά παρόμοιες οντότητες προσφέρουν κάποια διαφορετικά είδη υποψηφίων στοιχείων προς σύσταση.
- Τέλος τα KG παρέχουν εξηγήσιμες συστάσεις και επίσης επιλύουν το πρόβλημα της αραιότητας δεδομένων.

6.2 Επίπεδο ενσωμάτωσης Γνωσιακών Γράφων (KGE)

Η αρχιτεκτονική ενσωμάτωσης αναπαραστάσεων σε ένα γνωσιακό γράφο



Εικόνα 6.3: Επίπεδο KGCN
Πηγή: Knowledge Graph Convolutional Networks for Recommender Systems

Ορολογία

- Knowledge Graph (KG) $\mathcal{G}$
- Scoring function for a triple $f(t)$
- Loss function $\mathcal{L}$
- Optimization algorithm
- Negatives generation strategy

6.3 KGCN

Οι γράφοι γνώσης (knowledge graphs) περιλαμβάνουν αλληλοσυνδεόμενες πληροφορίες μεταξύ οντοτήτων, που είναι κατάλληλες προς αξιοποίηση από τα συστήματα προτάσεων. Πολλές αρχιτεκτονικές συστημάτων προτάσεων που χρησιμοποιούν γραφήματα γνώσης προϋποθέτουν δημιουργία χειροκίνητων χαρακτηριστικών (manual feature engineering), δεν επιτρέπουν εκπαίδευση από άκρο σε άκρο (end to end training) και παρέχουν χαμηλή απόδοση σε μεγάλα δίκτυα. Η αρχιτεκτονική (KGCN), είναι μια αρχιτεκτονική εκπαίδευσης δικτύου από άκρη σε άκρη που αξιοποιεί τις σχέσεις των στοιχείων που περιγράφονται στο το γράφο γνώσεων με σκοπό να παρέχει καλύτερες προτάσεις.

Εννοιολογικά, το KGCN υπολογίζει τις ενσωματώσεις ειδικών στοιχείων για τον χρήστη εφαρμόζοντας πρώτα μια εκπαιδευόμενη συνάρτηση που προσδιορίζει σημαντικές σχέσεις γραφημάτων γνώσης για έναν δεδομένο χρήστη και στη συνέχεια μετατρέπει το γράφημα γνώσης σε ένα σταθμισμένο γράφημα για έναν συγκεκριμένο χρήστη. Στη συνέχεια, το KGCN εφαρμόζει ένα περιελικτικό νευρωνικό δίκτυο γράφου με σκοπό να υπολογίσει την ενσωμάτωση ενός κόμβου (αντικειμένου) μέσω της διάδοσης και της συγκέντρωσης πληροφοριών των γειτονικών κόμβων (propagating and aggregating). Επιπλέον, για να υπάρχει αξιόπιστη επαγωγική μεροληψία (inductive bias), το KGCN χρησιμοποιεί την εξομάλυνση ετικέτας (label smoothness), η οποία παρέχει εξομάλυνση σε βάρη ακμών, μια διαδικασία ισοδύναμη της διάδοσης ετικετών (label propagation). Το KGCN-LS επιτυγχάνει επίσης εξαιρετική απόδοση σε αραιά δεδομένα (sparse data) και είναι εξαιρετικά κλιμακώσιμο ανεξάρτητα μεγέθους του γράφου γνώσης.

Το KGCN-LS επεκτείνει τα GCN στα KGs συγκεντρώνοντας μεροληπτικά τις πληροφορίες γειτονιάς, οι οποίες μπορούν να μάθουν τη δομή, τη σημασιολογική πληροφορία του KG, καθώς και τα εξατομικευμένα ενδιαφέροντα των χρηστών. Η προτεινόμενη σταθεροποίηση LS και η απώλεια «one-out» παρέχουν ισχυρή πρόσθετη καθοδήγηση για τη μαθησιακή διαδικασία. Εφαρμόζουμε επίσης το KGCN-LS με έναν επεκτάσιμο τρόπο μίνι παρτίδας. Μέσα από εκτεταμένα πειράματα, το KGCN-LS αποδεικνύεται ότι αποδίδει ικανοποιητικά και επιτυγχάνει την επιθυμητή επεκτασιμότητα ως προς το μέγεθος του γνωσιακού γράφου KG.

Γενική ιδέα

- Συγκεντρώνει και ενσωματώνει πληροφορίες γειτονιάς με μεροληψία κατά τον υπολογισμό της αναπαράστασης μιας δεδομένης οντότητας στον γνωσιακό γράφο.
- Μέσω της λειτουργίας της συγκέντρωσης γειτονιών, η τοπική δομή καταγράφεται και αποθηκεύεται με επιτυχία σε κάθε οντότητα.
- Οι γείτονες σταθμίζονται με βαθμολογίες, οι οποίες αντιπροσωπεύουν τόσο τη σημασιολογική πληροφορία του Γράφου γνώσης όσο και τα εξατομικευμένα ενδιαφέροντα των χρηστών ως προς τις κοινωνικές σχέσεις.

Μοντέλα χρηστών & στοιχείων.

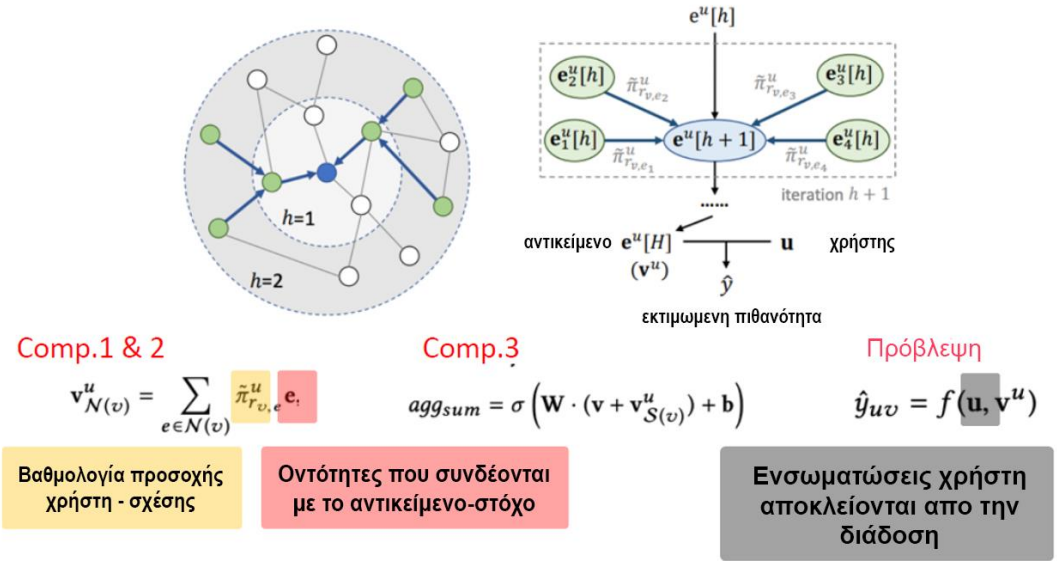
Ορίζουμε ένα επίπεδο KGCN ως:

- Σχέσεις χρηστών: Εξερεύνηση του λανθάνοντος σημείου ενδιαφέροντος του χρήστη, δηλαδή τη δημιουργία σχέσεων μεταξύ χρηστών και χαρακτηριστικών των στοιχείων ή χαρακτηριστικών (τα επισημαίνουμε με r) $\boldsymbol{\pi_r^u = g(u, r)}$, όπου το g (·) είναι εσωτερικό γινόμενο.
- Επίπεδο ενσωμάτωσης (συνέλιξη): βασίζεται σε γειτονικές αναπαραστάσεις του παραπάνω επιπέδου για να αντιπροσωπεύσει την οντότητα του επόμενου στρώματος

$$\boldsymbol{v_{N(v)}^u = \sum_{e \in N(v)} \tilde{\pi}_{r_{v,e}}^u e, \tilde{\pi}_{r_{v,e}}^u = \frac{\exp\left(\pi_{r_{v,e}}^u\right)}{\sum_{e \in N(v)} \exp\left(\pi_{r_{v,e}}^u\right)}} \quad (6.1)$$

Όπου, e είναι η αναπαράσταση της οντότητας και το $\tilde{\pi}_{r_{v,e}}^u$ μπορεί να θεωρηθεί ως βάρος κάθε οντότητας, ή κανονικοποιημένη βαθμολογία χρήστη- σχέσης που αντικατοπτρίζει την προτίμηση των χρηστών.

Τελικό επίπεδο: συγκεντρώνει την αναπαράσταση της οντότητας και την αναπαράσταση γειτονιάς της σε ένα διάνυσμα τιμών.



Εικόνα 6.4: Επίπεδο KGCN
Πηγή: Knowledge Graph Convolutional Networks for Recommender Systems

Η αρχιτεκτονική KGNN-LS είναι μια επέκταση της KGCN από τους [Wang et al, KDD'2019]

Η εξομάλυνση ετικέτας στηρίζεται στην παραδοχή ότι γειτονικά στοιχεία στο KG είναι πιθανό να έχουν παρόμοιες ετικέτες/βαθμολογίες συνάφειας χρηστών.

Για ένα νευρωνικό δίκτυο γνωσιακών γράφων KG : $\mathbf{v}_u = \boldsymbol{KGNN}(\mathbf{E}, \mathbf{A}_u)$ (διάδοση χαρακτηριστικών)

Και την συνάρτηση πρόβλεψης $\hat{\boldsymbol{y}}_{uv} = \boldsymbol{f}(\mathbf{u}, \mathbf{v}_u)$

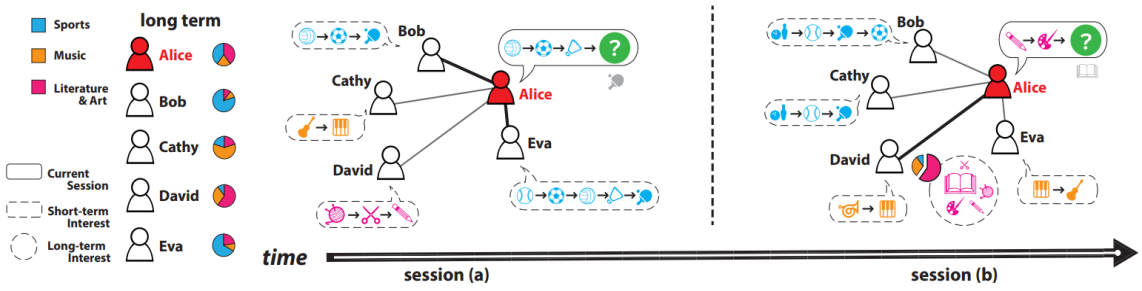
Για την κανονικοποίηση εξομάλυνσης ετικετών σε βάρη KG: $\boldsymbol{R}(\mathbf{A}_u)$ (διάδοση ετικέτας) τα παραπάνω συνδυάζονται ως εξής:

$$\mathcal{L} = \boldsymbol{J}\left(\hat{\boldsymbol{y}}_{uv}, \boldsymbol{y}_{uv}\right) + \boldsymbol{\lambda R}(\mathbf{A}_u) \quad (6.2)$$

$$\boldsymbol{E}(\boldsymbol{l}_u, \mathbf{A}_u) = \frac{1}{2}\sum_{\boldsymbol{e}_i \in \mathcal{E}, \boldsymbol{e}_j \in \mathcal{E}} \boldsymbol{A}_u^{\boldsymbol{ij}}\left(\boldsymbol{l}_u(\boldsymbol{e}_i) - \boldsymbol{l}_u(\boldsymbol{e}_j)\right)^2 \quad (6.3)$$

6.3.1 Αρχιτεκτονική DGREC

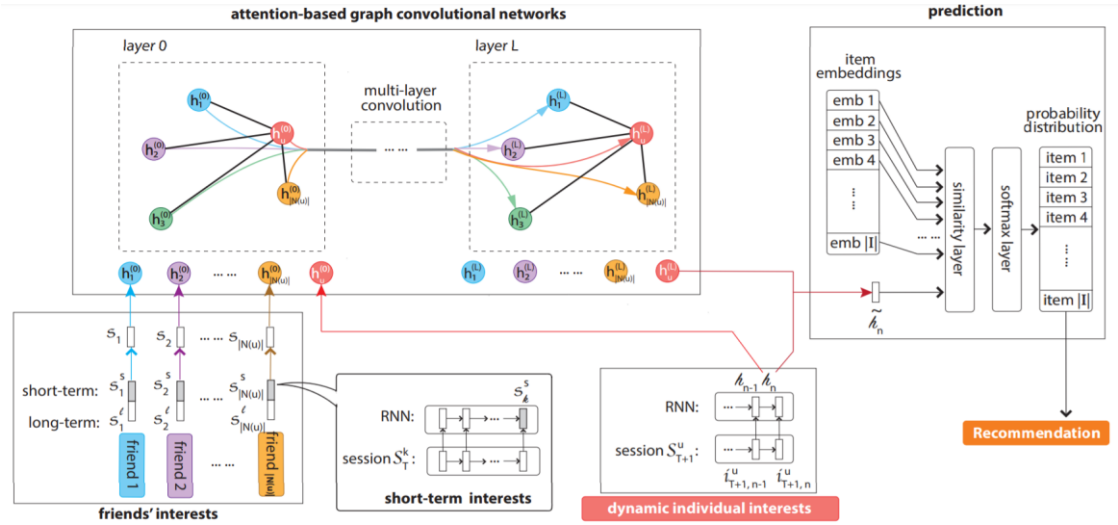
Στα κοινωνικά δίκτυα οι προτιμήσεις των χρηστών επηρεάζονται από το κοινωνικό περιβάλλον και τους φίλους. Για παράδειγμα αν ένας φίλος μου έχει δει μια ταινία, είναι πιθανό να ενδιαφερθώ και εγώ λόγω του φαινομένου της κοινωνικής επιρροής. Ο βαθμός που αυτή παίζει ρόλο εξαρτάται από την περίσταση. Εμπιστευόμαστε ορισμένους φίλους σε συγκεκριμένα θέματα ανάλογα με την επαγγελματική ή προσωπική ενασχόληση τους με την κατηγορία του αντικειμένου. Η επιρροή αυτή αφορά τρέχοντα και παλαιότερα ενδιαφέροντα των φίλων.



Εικόνα 6.5: Βραχυπρόθεσμες και μακροπρόθεσμες κοινωνικές επιρροές
Πηγή: Knowledge Graph Convolutional Networks for Recommender Systems

Η αρχιτεκτονική DGRec φτιάχνει μοντέλο ενδιαφερόντων του χρήστη και σχετικών επιρροών από το κοινωνικό περιβάλλον.

Η αρχιτεκτονική DGRec αποτελείται από τέσσερις ενότητες. Πρώτα ένα αναδρομικό νευρωνικό δίκτυο (RNN) με είσοδο την ακολουθία των στοιχείων που επιλέχθηκαν στην τρέχουσα συνεδρία από τον χρήστη. Τα ενδιαφέροντα των φίλων του διαμορφώνονται χρησιμοποιώντας ένα συνδυασμό των βραχυπρόθεσμων και μακροπρόθεσμων προτιμήσεών τους. Οι βραχυπρόθεσμες προτιμήσεις, για παράδειγμα, τα αντικείμενα που είδαν στην πιο πρόσφατη συνεδρία τους, κωδικοποιούνται και αυτά χρησιμοποιώντας RNN. Οι μακροχρόνιες προτιμήσεις των φίλων κωδικοποιούνται με μια εξατομικευμένη ενσωμάτωση. Στη συνέχεια, το μοντέλο συνδυάζει την αναπαράσταση χαρακτηριστικών του τρέχοντος χρήστη με τις αναπαραστάσεις των φίλων χρησιμοποιώντας ένα δίκτυο προσοχής. Ο προτεινόμενος μηχανισμός μαθαίνει να σταθμίζει την επιρροή κάθε φίλου με βάση τα τρέχοντα ενδιαφέροντα του χρήστη. Ως τελικό βήμα, το μοντέλο παράγει συστάσεις συνδυάζοντας τις τρέχουσες προτιμήσεις ενός χρήστη με τις κοινωνικές επιρροές του (ανάλογα με την περίσταση).



Εικόνα 6.6: Σχηματική αναπαράσταση του μοντέλου των δυναμικών κοινωνικών συστάσεων DGRec
Πηγή: Knowledge Graph Convolutional Networks for Recommender Systems

Δυναμικά ενδιαφέροντα χρήστη

Για να καταγράψουμε τα μεταβαλλόμενα ενδιαφέροντα ενός χρήστη, χρησιμοποιούμε ένα RNN για να μοντελοποιήσουμε τις ενέργειες του χρήστη (στόχου) κατά την τρέχουσα συνεδρία. Το

RNN συνάγει την αναπαράσταση της συνεδρίας ενός χρήστη $\mathbf{S_{uT+1}} = \{\mathbf{i_{uT+1,1}}, \dots, \mathbf{i_{uT+1,n}}\}$, συνδυάζοντας tokens αναδρομικά την αναπαράσταση όλων των προηγούμενων tokens με το τελευταίο, δηλ.

$$\mathbf{h_n} = \mathbf{f}(\mathbf{i_{uT+1,n}}, \mathbf{h_{n-1}}) \quad (6.4)$$

όπου το $\mathbf{h_n}$ αντιπροσωπεύει τα ενδιαφέροντα ενός χρήστη και το $\mathbf{F}(\cdot,\cdot)$ είναι μια μη γραμμική συνάρτηση που συνδυάζει και τις δύο πηγές πληροφοριών.

Αναπαράσταση ενδιαφερόντων των φίλων.

Οι ενέργειες κάθε φίλου $\mathbf{S_{kT}} = \{\mathbf{i_{kT,1}}, \mathbf{i_{kT,2'}}, \dots, \mathbf{i_{kT,N_{k,T}}}\}$διαμορφώνονται χρησιμοποιώντας ένα RNN. Τέλος, συνδυάζουμε τις βραχυπρόθεσμες και μακροπρόθεσμες προτιμήσεις των φίλων χρησιμοποιώντας έναν μη γραμμικό μετασχηματισμό:

$$\mathbf{s_k} = \mathbf{Re}\ \boldsymbol{LU}(\mathbf{W_1}[\mathbf{S_{sk}}; \mathbf{S_{lk}}]) \quad (6.5)$$

όπου $\mathbf{ReL}\ \boldsymbol{U}(\boldsymbol{x}) = \mathbf{max}(0, \boldsymbol{x})$ είναι μια μη γραμμική συνάρτηση ενεργοποίησης και $\mathbf{W_1}$ είναι ο πίνακας μετασχηματισμού.

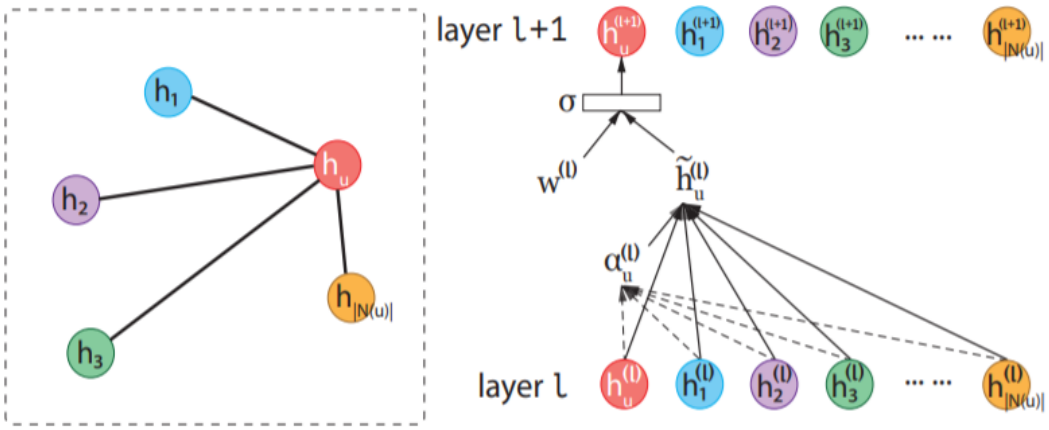
Κοινωνικές επιρροές σε πλαίσιο

Γράφημα δυναμικών χαρακτηριστικών

Για κάθε χρήστη, δημιουργούμε ένα γράφημα όπου οι κόμβοι αντιστοιχούν σε αυτόν τον χρήστη και τους φίλους της. Για συγκεκριμένο χρήστη u με $|\boldsymbol{N}(\boldsymbol{u})|$ φίλοι, το γράφημα έχει $|\boldsymbol{N}(\boldsymbol{u})| + 1$ κόμβους. Η αρχική αναπαράσταση του χρήστη $\mathbf{h_n}$ χρησιμοποιείται ως χαρακτηριστικά του κόμβου *h(o)u* (τα χαρακτηριστικά ενημερώνονται κάθε φορά που αλληλοεπιδρά με ένα νέο στοιχείο στο → $\boldsymbol{S_{uT+1}}$). Για έναν φίλο k, η λειτουργία κόμβου έχει οριστεί σε $\boldsymbol{s_k}$ και παραμένει αμετάβλητη για το χρονικό διάστημα $\boldsymbol{T} + 1$. Έτσι τα χαρακτηριστικά του κόμβου είναι $\boldsymbol{h_{(0)u}} = \boldsymbol{h_n}$ και $\{\boldsymbol{h_{(0)k}} = \boldsymbol{s_k}, \boldsymbol{k} \in \boldsymbol{N}(\boldsymbol{u})\}$.

6.3.2 Δίκτυο προσοχής σε Γράφο - Graph-Attention Network

Με τα χαρακτηριστικά του κόμβου να ορίζονται παραπάνω, περνάμε στη συνέχεια μηνύματα (χαρακτηριστικά) για να συνδυάσουμε τα ενδιαφέροντα των χρηστών των φίλων του χρήστη-στόχου. Αυτή η διαδικασία εφαρμόζεται σε ένα συνελικτικό δίκτυο γραφημάτων (Kipf and Welling, 2017).



Εικόνα 6.7: Το γραφικό μοντέλο του ενιαίου συνελικτικού επιπέδου χρησιμοποιώντας μηχανισμό προσοχής. Το αποτέλεσμα που εξαρτάται από το τρέχον ενδιαφέρον ερμηνεύεται ως κοινωνικές επιρροές εντός πλαισίου.
Πηγή: Kipf and Welling, 2017

Dataset της υλοποίησης

Douban data

Συλλογή των αξιολογήσεων των χρηστών σε τρεις τομείς (ταινίες, βιβλία και μουσική) από το Douban (www.douban.com), που είναι ένας δημοφιλής ιστότοπος αξιολόγησης στην Κίνα.

Τα στατιστικά των συνόλων δεδομένων Douban συνοψίζονται ως εξής:

Dataset	#user	#item	#event
DoubanMovie	94,890	81,906	11,742,260
DoubanMusic	39,742	164,223	1,792,501
DoubanBook	46,548	212,995	1,908,081

Εκτός από τις αξιολογήσεις υπάρχει και καταγραφή των κοινωνικών διασυνδέσεων.

	#node	#edge
SocialNet	695,800	1,758,302

Train.tsv

UserId	ItemId	Rating	Timestamp	TimeId	SessionId
0	1	4	1229529600.0	48	0_48
0	2	4	1229529600.0	48	0_48
0	3	4	1229529600.0	48	0_48

User_id_map

228054	23191
681607	18285
228056	245

Item_it_map

20051	11434
11541	7764
11545	10191

Adj.tsv

Follower	Followee	Weight
17711	18755	1.0
17711	20243	1.0
21575	4946	1.0
21575	9608	1.0

valid.tsv

UserId	ItemId	Rating	Timestamp	TimeId	SessionId
8	773	4	1473523200.0	452	8_452
8	789	4	1473523200.0	452	8_452
8	1105	4	1473696000.0	452	8_452

Τα παράκάτω από το raw DoubanMovie dataset με την εκτέλεση του preprocess_DoubanMovie.py

Η μορφη του Doubanmovie:

UserId	ItemId	Rating	Timestamp
630157	0	5	1182009600.0
630157	1	5	1182009600.0
630157	2	4	1182009600.0
630157	3	5	1182355200.0

Και του socialnet.tsv

Follower	Followee	Weight
48899	127372	1.0
48899	149248	1.0
48899	674863	1.0

6.4 Dual Graph Attention Networks (DANSER)

Οι περισσότερες αρχιτεκτονικές συστημάτων κοινωνικών συστάσεων υποθέτουν ότι η κοινωνική επιρροή από τους φίλους είναι στατική και απεικονίζεται με σταθερά βάρη ή/και περιορισμούς. Με τα δίκτυα γραφήματος διπλής προσοχής (dual GAT) μαθαίνουμε συνεργατικά τις αναπαραστάσεις των κοινωνικών επιδράσεων σε δυο επίπεδα, όπου το ένα διαμορφώνεται από το βάρος του συγκεκριμένου χρήστη και το άλλο είναι υπολογίζει με δυναμικό τρόπο το βάρος για κάθε αντικείμενο. Επεκτείνονται έτσι και οι κοινωνικές επιδράσεις του κοινωνικού δικτύου στον δίκτυο των αντικειμένων, έτσι ώστε οι πληροφορίες από σχετικά αντικείμενα να μπορούν να αξιοποιηθούν για την αντιμετώπιση του προβλήματος σπανιότητας δεδομένων. Επιπλέον, λαμβάνοντας υπόψη ότι οι διαφορετικές σχέσεις επιδράσεις στους 2 παραπάνω τομείς αλληλοεπιδρούν μεταξύ τους και επηρεάζουν από κοινού τις προτιμήσεις των χρηστών, η παρούσα στρατηγική σύντηξης με τεχνική contextual multi-armed bandit για των καθορισμό των εκάστοτε κοινωνικών επιπτώσεων.

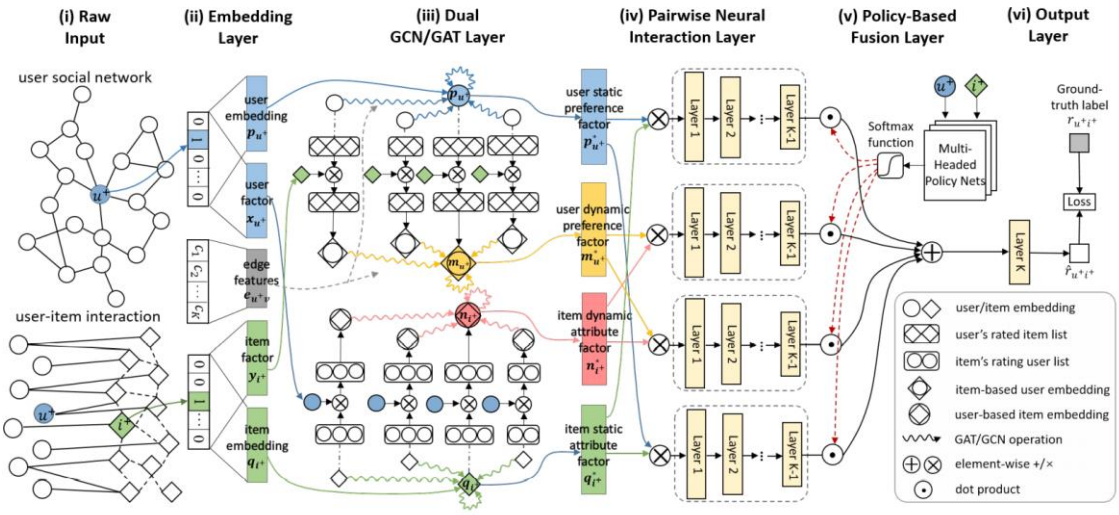
6.4.1 Η αρχιτεκτονική του DANSER

The model requires user-item interaction matrix $\boldsymbol{R}$ and user social network $\boldsymbol{G_U}$ as input. Social recommendation methods focus on leveraging the user social network to solve the data sparsity problem in recommender systems. Most existing methods treat items independently since there is no prior information that explicitly expresses the relationship between items. One way to calculate the similarity or relevance between two items is by the common users who clicked or rated them [25]. For any item $\boldsymbol{i}$ and item $\boldsymbol{j}$, we define their similarity coefficient $\boldsymbol{s_{ij}}$ as the number of users who clicked both items. These coefficients induce an equivalence relation over items as follows: item $\boldsymbol{i}$ is related to item $\boldsymbol{j}$ if $\boldsymbol{s_{ij} > \tau}$ with $\boldsymbol{\tau}$ a fixed threshold. We define the item implicit network as the graph $\boldsymbol{G_I = (V_I, E_I)}$ where $\boldsymbol{V_I}$ is the set of items and $\boldsymbol{E_I}$ is the set of edges that connects two related items.

Η μέθοδος απαιτεί ως είσοδο τον πίνακα με δεδομένα αλληλεπίδρασης μεταξύ χρηστών – αντικειμένων, και τον Γράφο του κοινωνικού δικτύου των χρηστών. Έτσι ξεπερνάμε το πρόβλημα αραιού πίνακα, και την έλλειψη πληροφορίων που να εκφράζουν ρητά τη σχέση μεταξύ των αντικειμένων, αξιοποιώντας το κοινωνικό δίκτυο των χρηστών. Ένας τρόπος για να υπολογιστεί η ομοιότητα ή η συνάφεια μεταξύ δύο στοιχείων είναι από τους κοινούς χρήστες που τους έκαναν κλικ ή τα βαθμολόγησαν. Για οποιοδήποτε στοιχείο i και j, ορίζουμε ως συντελεστή ομοιότητας $\boldsymbol{s_{ij}}$ ισο με τον αριθμό των χρηστών που έκαναν κλικ και στα δύο στοιχεία. Αυτοί οι συντελεστές προκαλούν μια σχέση ισοδυναμίας για τα στοιχεία ως εξής: το στοιχείο i σχετίζεται με το στοιχείο j εάν $\boldsymbol{s_{ij} > \tau}$ όπου τ ένα σταθερό κατώφλι. Ορίζουμε το αποκομμένο δίκτυο αντικειμένων ως το γράφημα $\boldsymbol{G_I = (V_I, E_I)}$ όπου το $\boldsymbol{V_I}$ είναι το σύνολο των αντικειμένων και το $\boldsymbol{E_I}$ είναι το σύνολο των ακμών που συνδέουν δύο σχετικά στοιχεία.

6.4.2 Επίπεδο ενσωμάτωσης χαρακτηριστικών

Η ακατέργαστη είσοδος κάθε χρήστη (που αντιστοιχούν σε αντικείμενο) είναι ένα one hot διάνυσμα πολλών διαστάσεων και η λειτουργία ενσωμάτωσης προβάλλει κάθε χρήστη σε μια αναπαράσταση χαμηλής διάστασης σε μια ενσωμάτωση του συγκεκριμένου χρήστη και των στοιχεία που βαθμολογούνται από αυτήν. Η πρώτη αναπαράσταση αντικατοπτρίζει τα ενδιαφέροντα του χρήστη, ενώ η δεύτερη (που ονομάζεται ενσωμάτωση χρήστη βάσει στοιχείων) αποτυπώνει την έμμεση επίδραση του ιστορικού βαθμολογίας στην τρέχουσα απόφαση σύστασης.



Εικόνα 6.8: Αρχιτεκτονική του DANSER. Οι μπλε κύκλοι υποδηλώνουν έναν στοχευμένο χρήστη ενώ οι πράσινοι ρόμβοι ένα υποψήφιο στοιχείο.
Πηγή: Kipf and Welling, 2017

i) Το μοντέλο απαιτεί δεδομένα αλληλεπίδρασης χρήστη-στοιχείου και κοινωνικό δίκτυο των χρηστών ως αρχική είσοδο. Λαμβάνουμε κοινούς χρήστες που βαθμολογούν και τα δύο στοιχεία για τον υπολογισμό της συνάφειας των στοιχείων και συνδέουμε τα σχετικά στοιχεία για να σχηματίσουν ένα έμμεσο δίκτυο στοιχείων που συνδέονται με διακεκομμένες γραμμές.

ii) Στο επίπεδο ενσωμάτωσης, αναπαριστούμε έναν χρήστη (αντίστοιχο στοιχείο) ως ένα διάνυσμα ενσωμάτωσης χαμηλής διάστασης και έναν λανθάνοντα παράγοντα. Επιπλέον, οι συχνότητες αλληλεπίδρασης μεταξύ των χρηστών χρησιμοποιούνται ως βάρος ακμής.

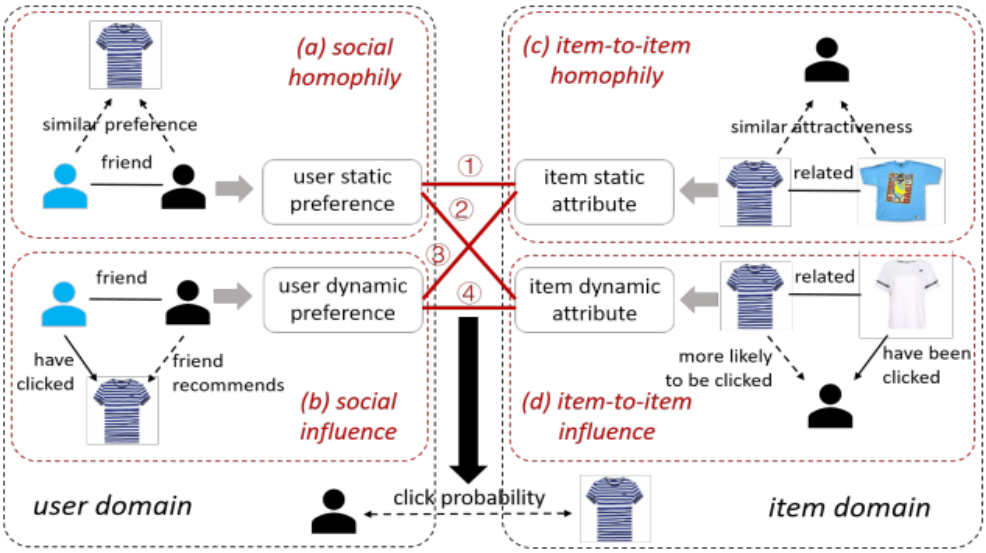
iii) Στο διπλό επίπεδο GCN/GAT, τέσσερα διαφορετικά δίκτυα προσοχής γράφου πρέπει να καταγράφουν τις κοινωνικές επιρροές με διπλή επίπτωση, όπου τα επάνω (αντίστοιχα κάτω) δύο εξάγουν αναπαραστάσεις για τις στατικές και δυναμικές προτιμήσεις του χρήστη (αντίστοιχο στοιχείο) (αντίστοιχα χαρακτηριστικά) υπό την επίδραση της ομοφυλίας και της επιρροής, αντίστοιχα.

iv) Αυτοί οι τέσσερις παράγοντες θα συνδυαστούν κατά ζεύγη ως τέσσερα αλληλοεπιδρώμενα χαρακτηριστικά, τα οποία στη συνέχεια τροφοδοτούν τέσσερα ανεξάρτητα νευρωνικά δίκτυα για να ληφθούν πιο συμπυκνωμένες αναπαραστάσεις.

v) Στη συνέχεια, ένα δίκτυο πολιτικής με την εισαγωγή των στοιχείων $\boldsymbol{i}^+$ και των ενσωματώσεων του χρήστη $\boldsymbol{u}^+$ πληροφοριών περιβάλλοντος, βγάζει βάρη για τέσσερις συνδυαστικά χαρακτηριστικά, τα οποία θα συγκεντρωθούν σε ένα συνθετικό διάνυσμα.

vi) Τέλος, το συνθετικό διάνυσμα εισάγεται στο επίπεδο εξόδου για να δώσει την τελική πρόβλεψη βαθμολογίας $\hat{\boldsymbol{r}}_{\boldsymbol{u}^+\boldsymbol{i}^+}$.

6.4.3 Ομοφυλία και κοινωνική επιρροή



Εικόνα 6.9: Αναπαράσταση των διπλών κοινωνικών επιπτώσεων, δηλαδή της επιρροής και της συχνότητας αλληλεπιδράσεων μεταξύ παρόμοιων ατόμων σε κοινωνικά δίκτυα, καθώς και μεταξύ σχετικών αντικειμένων. Οι τέσσερις κοινωνικές επιπτώσεις από κοινού επηρεάζουν την προτίμηση ενός χρήστη για ένα αντικείμενο.
Πηγή: Kipf and Welling, 2017

DATASETS

- Το Epinions είναι ιστότοπος αξιολόγησης καταναλωτών, όπου οι χρήστες μπορούν να αξιολογήσουν ορισμένα στοιχεία και να προσθέσουν άλλους χρήστες στις λίστες εμπιστοσύνης τους. Το σύνολο δεδομένων Epinions περιέχει δύο είδη πληροφοριών: τα ζεύγη αλληλεπίδρασης χρήστη-στοιχείου, όπου τα στοιχεία βαθμολογούνται από 1 έως 5, καθώς και σχέσεις εμπιστοσύνης μεταξύ των χρηστών (κατευθυνόμενες ακμές). Το σύνολο δεδομένων έχει χρησιμοποιηθεί ευρέως ως σημεία αναφοράς για κοινωνικές προτάσεις. Η εφαρμογή επιλέγει τυχαία το 80% των αλληλεπιδράσεων στοιχείων χρήστη ως σύνολο εκπαίδευσης και το υπόλοιπο 20% ως σύνολο δοκιμών.

6.5 DIFFNET ++

Το προκαταρκτικό στάδιο του DiffNet υιοθετεί τη διαδικασία διαδοχικής διάχυσης επιρροής για επαναληπτική εκμάθηση ενσωμάτωσης χρηστών, έτσι ώστε η δομή κοινωνικού δικτύου μέχρι την K-απόσταση να ενημερώνει την διαδικασία κοινωνικής πρότασης. Το DiffNet ++, ένα βελτιωμένο μοντέλο του DiffNet που συγχωνεύει τόσο τη διάχυση στο κοινωνικό δίκτυο $\boldsymbol{G_S}$ όσο και τη διάδοση ενδιαφέροντος στο δίκτυο συμφερόντων $\boldsymbol{G_I}$ για κοινωνικές προτάσεις.

Η αρχιτεκτονική του DiffNet ++ περιέχει τέσσερα κύρια μέρη: ένα στρώμα ενσωμάτωσης, ένα στρώμα σύντηξης, τα επίπεδα επιρροής και διάχυσης ενδιαφέροντος και ένα επίπεδο πρόβλεψης βαθμολογίας. Συγκεκριμένα, λαμβάνοντας σχετικές εισόδους, το επίπεδο ενσωμάτωσης εξάγει ενσωματώσεις χρηστών και αντικειμένων, και το στρώμα σύντηξης συγχωνεύει τόσο τις δυνατότητες περιεχομένου όσο και τις δωρεάν ενσωματώσεις. Στα επίπεδα επιρροής και διάχυσης ενδιαφέροντος, σχεδιάζουμε προσεκτικά μια δομή προσοχής πολλαπλών επιπέδων που θα μπορούσε αποτελεσματικά να διαχέει κοινωνικά δίκτυα και ενδιαφέροντα υψηλότερης τάξης. Αφού η διαδικασία διάχυσης φτάσει σταθερή, το επίπεδο

εξόδου προβλέπει τη βαθμολογία προτίμησης κάθε ζεύγους μη παρατηρούμενων στοιχείων χρήστη.

6.5.1 Επίπεδο Ενσωμάτωσης

Κωδικοποιεί χρήστες και στοιχεία με αντίστοιχες δωρεάν διανυσματικές αναπαραστάσεις. Αν τα $\mathbf{P} \in \mathbb{R}^{M \times D}$ και τα $\mathbf{Q} \in \mathbb{R}^{N \times D}$ αντιπροσωπεύουν τους ελεύθερους λανθάνοντες πίνακες ενσωμάτωσης χρηστών και στοιχείων με D διαστάσεις. Λαμβάνοντας υπόψη τις μοναδικές αναπαραστάσεις του χρήστη a, το επίπεδο ενσωμάτωσης εκτελεί μια επιλογή ευρετηρίου και εξάγει τη ελεύθερη λανθάνουσα ενσωμάτωση$\mathbf{p}_a$ του χρήστη, δηλαδή τη μεταφορά μιας σειράς από τον πίνακα ενσωμάτωσης χωρίς χρήστη $\mathbf{p}_a$. Ομοίως, η ενσωμάτωση $\mathbf{q}_i$του στοιχείου i' s είναι η μεταφορά της i -ης σειράς της μήτρας ενσωμάτωσης $\mathbf{Q}$.

6.5.2 Επίπεδο σύντηξης.

Για κάθε χρήστη a, το επίπεδο σύντηξης παίρνει ως είσοδο το $\mathbf{p}_a$ και το σχετικό διάνυσμα χαρακτηριστικών $\mathbf{x}_a$ και εξάγει μια σύντηξη χρήστη ενσωματώνοντας το $\mathbf{u}_a^0$ που αποτυπώνει τα αρχικά ενδιαφέροντα του χρήστη από διαφορετικά είδη δεδομένων εισόδου. Αναπαριστούμε το επίπεδο σύντηξης ως εξής:

$$\mathbf{u}_a^0 = g(\mathbf{W}_1 \times [\mathbf{p}_a, \mathbf{x}_a,]) \quad (6.6)$$

όπου το $\mathbf{W}_1$ είναι μια μήτρα μετασχηματισμού και το g (x) είναι μια συνάρτηση μετασχηματισμού. Χωρίς σύγχυση, παραλείπουμε τον όρο προκατάληψης. Αυτό το στρώμα σύντηξης θα μπορούσε να γενικεύσει πολλές τυπικές λειτουργίες σύντηξης, όπως η λειτουργία συνένωσης $\mathbf{u}_a^0 = [\mathbf{p}_a, \mathbf{x}_a]$ ορίζοντας το $\mathbf{W}_1$ ως μήτρα ταυτότητας και $g(x)$ μια συνάρτηση ταυτότητας.

Ομοίως, για κάθε στοιχείο i, το επίπεδο σύντηξης μοντελοποιεί την ενσωμάτωση αντικειμένου $\mathbf{v}_i^0$ως συνάρτηση μεταξύ του ελεύθερου λανθάνοντος διανύσματος q_i και του διανύσματος χαρακτηριστικών του $\mathbf{y}_i$ ως εξής:

$$\mathbf{v}_i^0 = g(\mathbf{W}_2 \times [\mathbf{q}_i, \mathbf{y}_i]) \quad (6.7)$$

ΚΕΦΑΛΑΙΟ 7. Μετρικές απόδοσης - Performance Measures

MAE - Μέσο απόλυτο Σφάλμα

Ορίζουμε το Μέσο απόλυτο Σφάλμα (MAE) στην παρακάτω εξίσωση όπου N είναι ο συνολικός αριθμός αξιολογήσεων που πρέπει να προβλέψουμε, Pu, i είναι η προβλεπόμενη τιμή της βαθμολογίας του στοιχείου i από τον χρήστη u και Ru, i είναι η πραγματική βαθμολογία του στοιχείου i από τον χρήστη u.

Ακολούθως το MAE μετρά τη μέση απόλυτη απόκλιση μεταξύ της προβλεπόμενης βαθμολογίας Pu, i του αλγορίθμου και της πραγματικής βαθμολογίας χρήστη Ru, i

RMSE

Το Root Mean Square Error (RMSE). Χρησιμοποιούμε το RMSE καθώς και το MAE καθώς μας δίνει μια ευρύτερη εικόνα της απόδοσης κάθε αλγορίθμου. Με το RMSE, τα μεγαλύτερα σφάλματα στην πρόβλεψη έχουν μεγαλύτερο σχετικό βάρος, καθώς η διαφορά μεταξύ της προβλεπόμενης βαθμολογίας Pu, i από την πραγματική βαθμολογία Ru, i στο τετράγωνο. Αυτό σημαίνει ότι το RMSE τιμωρεί τις προβλεπόμενες τιμές βαθμολογίας που αποκλίνουν περισσότερο από την πραγματική τιμή βαθμολογίας. Ως αποτέλεσμα, οι χαμηλότερες τιμές RMSE υποδεικνύουν ότι ένας αλγόριθμος είναι πιο ακριβής υπό την έννοια ότι οι προβλεπόμενες τιμές του δεν αποκλίνουν πολύ από την πραγματική βαθμολογία.

$$MAE = \frac{\sum_{i=1}^{N} |P_{u,i} - R_{u,i}|}{N} \quad (7.1)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (R_{u,i} - P_{u,i})^2}{N}} \quad (7.2)$$

Coverage - Κάλυψη

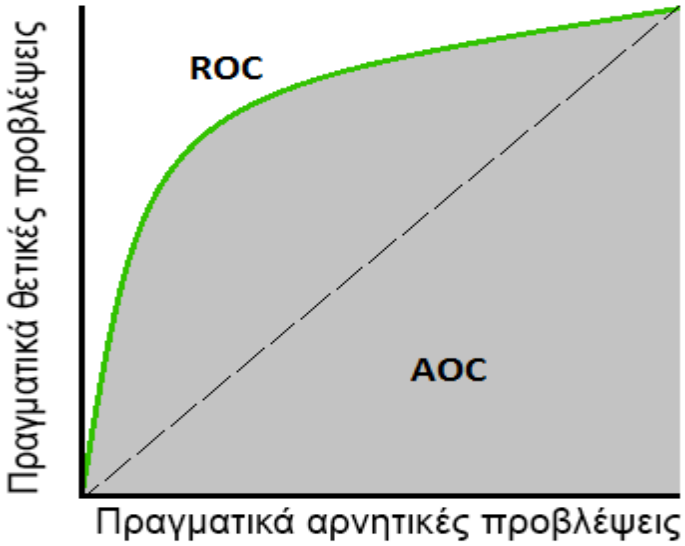
Εκτός από αυτά τις μετρήσεις ακρίβειας, μπορούμε να αναλύσουμε την απόδοση όπως συζητήθηκε στο και μετρώντας την κάλυψη (coverage) των αλγορίθμων. Η κάλυψη είναι το ποσοστό των κενών αξιολογήσεων χρηστών σε στοιχεία για τα οποία ο αλγόριθμος μπόρεσε να παράγει κάποιο είδος πρόβλεψης. Την υπολογίζουμε λαμβάνοντας τον αριθμό των αξιολογήσεων για τις οποίες ο αλγόριθμος έκανε μια σωστή πρόβλεψη και τον διαιρεί με τον συνολικό αριθμό των κενών αξιολογήσεων που επρόκειτο να προβλεφθούν, όπως φαίνεται στην παρακάτω εξίσωση, όπου P είναι ο συνολικός αριθμός προβλέψεων που δημιουργήθηκαν από τον αλγόριθμο και το N είναι ο συνολικός αριθμός αξιολογήσεων χρηστών σε στοιχεία που ζητήθηκε από τον αλγόριθμο για πρόβλεψη.

$$\text{Coverage} = P / N \quad (7.3)$$

Η κάλυψη των προβλέψεων έχει αποδειχθεί ότι είναι ένα σημαντικό μέτρο απόδοσης για αλγόριθμους συστάσεων, καθώς ορισμένες τεχνικές έχουν αποδειχθεί ότι δεν μπορούν να παράγουν μεγάλο αριθμό προβλέψεων βαθμολογιών. Τέτοιοι αλγόριθμοι συνήθως στηρίζονται στην μέθοδο συλλογικού φιλτραρίσματος (Collaborative Filtering) της Resnick (εξίσωση 3) όταν δεν υπάρχουν χρήστες που έχουν βαθμολογήσει ένα στοιχείο i που έχουν θετική συσχέτιση με τον Pearson u.

Καμπύλη AUC – ROC

Η καμπύλη AUC - ROC είναι μια μέτρηση απόδοσης για τα προβλήματα ταξινόμησης σε διάφορες ρυθμίσεις κατωφλίου. Η ROC είναι μια καμπύλη πιθανότητας και το AUC αντιπροσωπεύει το βαθμό ή το μέτρο του διαχωρισμού. Λέει πόσο το μοντέλο είναι ικανό να διακρίνει μεταξύ τάξεων. Όσο υψηλότερη είναι η AUC, τόσο καλύτερα το μοντέλο προβλέπει 0 κλάσεις ως 0 και 1 ως 1. Κατ 'αναλογία, όσο υψηλότερη είναι η AUC, τόσο καλύτερο είναι το μοντέλο στη διάκριση μεταξύ περιπτώσεων που ανήκουν ή δεν ανήκουν σε μια κατηγορία.



Εικόνα 7.1: Η καμπύλη ROC απεικονίζεται με TPR έναντι του FPR όπου το TPR βρίσκεται στον άξονα y και το FPR στον άξονα x.

$$\boldsymbol{AUC = 1 - \sum_{k=1}^{n} (X_k - X_{k-1})(Y_k + Y_{k-1})} \quad (7.4)$$

Οι τιμές όλων των παραπάνω βρίσκονται στο σύνολο [0, 1], με το 1 να αποτελεί δείκτη απόλυτης ευστοχίας της ταξινόμησης, ενώ το 0 το αντίθετο.

F1

Ως F1 ορίζουμε τον αρμονικό μέσο της ευαισθησίας και της ακρίβειας, ορισμένης υπό την έννοια του ποσοστού των πραγματικά θετικών έναντι των πραγματικά και λανθασμένα θετικών.

Το εμβαδόν που καλύπτει η ROC ονομάζεται Περιοχή Κάτω από την Καμπύλη (Area Under the Curve - AUC) και οι τιμές της ανήκουν, όπως αναφέρθηκε, στο [0 ,1]. Ένα σύστημα που είναι πολύ εύστοχο θα έχει AUC πολύ κοντά στη μονάδα, ενώ όταν θα αποτυγχάνει τις μισές περιπτώσεις η ROC θα έχει τη μορφή της δεύτερης διαγώνιου.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Recommendation Systems	Συστήματα Συστάσεων
Content based filtering	Φιλτραρισμα Βάσει Περιεχομένου
Collaborative filtrering	Συνεργατικό φιλτράρισμα
Context aware recsys	Συστάσεις σχετικές με το πλαίσιο
Pepresentation learning	Μάθηση αναπαραστάσεων
Latent Factor models	Μοντέλα λανθανόντων παραγόντων
Matrix Factorization	Παραγοντοποιηση Μήτρας
CNN	Συνελικτικά Νευρωνικά Δίκτυα
Neural collaborative filtering	Συνεργατικό Φιλτράρισμα με χρήση νευρωνικού δικτύου
RNNs	Αναδρομικά Νευρωνικά Δίκτυα
Gated Recurrent Unit	Αναδρομική Μονάδα με Πύλη
Non linear transformation	Μη γραμμικός μετασχηματισμός (συνάρτηση)
Deep Neural Network	Πολυεπίπεδο Νευρωνικό Δίκτυο
embeddings	Ενσωματώσεις (αναπαραστάσεις)
graph level embeddings	Ενσωμάτωση Γράφου
Graph Convolution	Συνέλιξη Γράφου
Multilayer perceptrons	Διάταξη Πέρσεπτρον σε πολλά επίπεδα
Graph Attention Networks (GATs)	Νευρωνικά Δίκτυα Γράφων με Πύλες

ΒΙΒΛΙΟΓΡΑΦΙΑ

Adomavicius, G. and Tuzhilin, A., 2021. Context-Aware Recommender Systems.SHIWEN WU, FEI SUN, WENTAO ZHANG, BIN CUI, Graph Neural Networks in Recommender Systems: A Survey arXiv:2011.02260v2 [cs.IR] 19 Apr 2021

Hamilton, W., 2020. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3), pp.1-159.

ZHANG, Q., ZHANG, L., QIN, C., WANG, C., ZHU, H., XIONG, H., CHEN, E., GUO, Q. and ZHUANG, F., 2020. A survey on knowledge graph-based recommender systems. SCIENTIA SINICA Informationis, 50(7), pp.937-956.

Wang, B. and Cai, W., 2021. *Attention-Enhanced Graph Neural Networks for Session-Based Recommendation*.

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2021. *Empirical evaluation of gated recurrent neural networks on sequence modeling*. [online] NYU Scholars. Available at: <https://nyuscholars.nyu.edu/en/publications/empirical-evaluation-of-gated-recurrent-neural-networks-on-sequen> [Accessed 22 August 2021].

Wang, M., Qiu, L. and Wang, X., 2021. A Survey on Knowledge Graph Embeddings for Link Prediction. *Symmetry*, 13(3), p.485.

Anwar, K., Siddiqui, J. and Sohail, S., 2020. Machine learning-based book recommender system: a survey and new perspectives. *International Journal of Intelligent Information and Database Systems*, 13(2/3/4), p.231.

Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019

Neural Collaborative Filtering, c 2017 International World Wide Web Conference Committee(IW3C2), WWW 2017, April 3–7, 2017, Perth, Australia.ACM 978-1-4503-4913-0/17/04. http://dx.doi.org/10.1145/3038912.3052569

Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie AutoRec: Autoencoders Meet Collaborative Filtering. WWW 2015 Companion, May 18–22, 2015, Florence, Italy. ACM 978-1-4503-3473-0/15/05. http://dx.doi.org/10.1145/2740908.2742726 .

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J. and Yin, D., 2021. *Graph Neural Networks for Social Recommendation*. [online] arXiv.org. Available at: <https://arxiv.org/abs/1902.07243> [Accessed 22 August 2021].

DeepAI. 2021. *Session-based Social Recommendation via Dynamic Graph Attention Networks*. [online] Available at: <https://deepai.org/publication/session-based-social-recommendation-via-dynamic-graph-attention-networks> [Accessed 22 August 2021].

Wu, Q., Zhang, H., Gao, X., He, P., Weng, P., Gao, H. and Chen, G., 2021. *Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems*.

Wu, L., Li, J., Sun, P., Hong, R., Ge, Y. and Wang, M., 2021. *DiffNet++: A Neural Influence and Interest Diffusion Network for Social Recommendation*. [online] arXiv.org. Available at: <https://arxiv.org/abs/2002.00844> [Accessed 22 August 2021].

ΠΑΡΑΡΤΗΜΑ – ΕΚΤΕΛΕΣΗ ΥΛΟΠΟΙΗΣΕΩΝ

Έγινε προσπάθεια εκτέλεσης υλοποιήσεων όπως περιγράφονται στις εργασίες των αρχιτεκτονικών που περιγράφονται σε αυτήν την εργασία. Στα περισσότερα βρέθηκε υλοποίηση στο github με την ακριβή μεθοδολογία και με λίγες προσαρμογές στον κώδικα, κατέστη δυνατόν να βγουν αποτελέσματα συγκρίσιμα με αυτά που αναφέρονται στις αντίστοιχες εργασίες παρουσίασής τους.

Εκτός από την LightCGN, όπου δεν ήταν δυνατόν να εκτελεστεί, οι υλοποιήσεις έτρεξαν στα υπολογιστικά περιβάλλοντα που ήταν διαθέσιμα. Συγκεκριμένα ένας Η/Υ με Ryzen5 με 6/12 threads και 16 GB μνήμη με ενσωματωμένη GPU, και οι υλοποιήσεις που χρειάζονται GPU (Cuda), έτρεξαν σε Η/Υ με Intel i5 6400 16GB ram και Nvidia GTX1060 (6GB ram)

Στον παρακάτω πίνακα αναφέρονται συνοπτικά τα απαραίτητα APIs, τα Datasets που επιλέχθηκαν, ρυθμίσεις στα επιμέρους δίκτυα, χρόνοι και μετρικές ακρίβειας.

	Περιβάλλον	GPU	Dataset	Χρόνος εκτ.	Ρυθμίσεις	Μετρικές ακρίβειας
DANSER Qitian Wu et al.	Tensorflow	X	Epinions	~11 ώρες		MAE:0,7780 RMSE:1.0277
GRAPHREC Wenqi Fan et al.	Torch.cuda Sk.Learn	V	Epinions	~2,5 ώρες		MAE:0,0,8621 RMSE:1.1901
KGCN-LS Hongwei Wang et al.	Tensorflow	V	MovieLens20M	0,5 ώρες	epoch: 405	AUC: 0.9757 F1:0,9298
DIFFNET++ Le Wu et al.	TF.GPU	V	YELP	~0,5 ώρες	epoch: 99	TOP5 : 0,1891 TOP10: 0,2864
KGCN Hongwei Wang et al.	Tensorflow, Sk.Learn	X	MovieLens20M	< 0,5 ώρες		AUC: 0.9744 F1:0,9274

Παρακάτω παρουσιάζεται αναλυτικά η έξοδος από την κάθε υλοποίηση.

DANSER-WWW-19-master dataset Epinions σε pickle

ΠΕΡΙΒΑΛΛΟΝ

Test.py με spyder

AttributeError: module 'tensorflow' has no attribute 'set_random_seed'

Περιβάλλον Tensorflow 1.7.0.

Θελει tf.random.set_seed()

Απαιτείται περιβάλλον στο anaconda με python 3.6 και tf 1.7

build_dataset.py

OSError: /home/myronwu/DANSER-WWW-19/data/ratings_data.txt not found.

Εκτελεση του Build_dataset.py

Test.py :

2021-03-02 08:25:49.091946: I tensorflow/core/platform/cpu_feature_guard.cc:140] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2

Στο jsonapi.py, κάνω σχόλια τις παρακάτω γραμμές

if isinstance(s, unicode):

  # s = s.encode('utf8')

DOKIMH 3

Με python 3.6

Tf 1.7

Pandas 1.0

Με κλειστό το avast

Και περιβάλλον spyder

CPU training only

ΑΠΟΤΕΛΕΣΜΑ ΤΟΥ PAPER

Table 2: Comparative results for Epinions and WeChat. For MAE, RMSE, the smaller value is better, and vice versa for P@10, AUC.

	Epinions		WeChat	
	MAE	RMSE	P@10	AUC
SVD++ [15]	0.8321	1.0772	0.0653	0.7304
DELF [2]	0.8115	1.0561	0.0752	0.7818
TrustPro [37]	0.9130	1.1124	0.0561	0.6482
TrustMF [36]	0.8214	1.0715	0.0625	0.7005
TrustSVD [10]	0.8144	1.0492	0.0664	0.7325
NSCR [31]	0.8044	1.0425	0.0736	0.7727
SREPS [16]	0.8014	1.0393	0.0725	0.7745
DANSER	**0.7781**	**1.0268**	**0.0823**	**0.8165**
Impv.[1]	2.87%	1.25%	9.33%	4.48%

[1] The improvement compares DANSER with the best competitor (underlined).

Αποτελέσματα

runfile('C:/Users/pc/Downloads/REC SYS CODE/DANSER-WWW-19-master/DANSER-WWW-19-master/train.py', wdir='C:/Users/pc/Downloads/REC SYS CODE/DANSER-WWW-19-master/DANSER-WWW-19-master')

Reloaded modules: eval, input, model

19:48 μμ εως 6:53 πμ

Epoch 21 Step 179000 Train_loss: 0.8919 Test_loss: 1.1500 P@3: 0.9033 P@5: 0.8855 P@10: 0.8253 NDCG@3: 0.8189 NDCG@5: 0.8178 NDCG@10: 0.8554 MAE: 0.7882 RMSE: 1.0290 Best_MAE: 0.7780
Epoch 21 Step 180000 Train_loss: 0.8893 Test_loss: 1.1480 P@3: 0.9039 P@5: 0.8855 P@10: 0.8255 NDCG@3: 0.8196 NDCG@5: 0.8183 NDCG@10: 0.8556 MAE: 0.7859 RMSE: 1.0277 Best_MAE: 0.7780
Epoch 21 Step 181000 Train_loss: 0.9098 Test_loss: 1.1490 P@3: 0.9056 P@5: 0.8845 P@10: 0.8261 NDCG@3: 0.8205 NDCG@5: 0.8169 NDCG@10: 0.8555 MAE: 0.7890 RMSE: 1.0283 Best_MAE: 0.7780
Epoch 21 Step 182000 Train_loss: 0.9139 Test_loss: 1.1481 P@3: 0.9062 P@5: 0.8853 P@10: 0.8259 NDCG@3: 0.8209 NDCG@5: 0.8180 NDCG@10: 0.8557 MAE: 0.7866 RMSE: 1.0277 Best_MAE: 0.7780

Evaluate your Recommendation Engine using NDCG

https://towardsdatascience.com/evaluate-your-recommendation-engine-using-ndcg-759a851452d1

Recall@20, Recall@50, and NDCG@100

GraphRec

ΠΕΡΙΒΑΛΛΟΝ

python: 3.6

pytorch: 0.2+

pickle του epinions ένα αρχειο

εγκατάσταση torch.cuda

Πειρβάλλον copy το env του danser και εβαλα pytorch 1.0.1

Traceback (most recent call last):

File "C:\Users\DeskMini\Documents\REC SYS CODE\GraphRec-WWW19-master\GraphRec-WWW19-master\run_GraphRec_example.py", line 1, in <module>

import torch

File "C:\Users\DeskMini\anaconda3\envs\danser graphrec pytorch 03\lib\site-packages\torch__init__.py", line 102, in <module>

from torch._C import *

ImportError: DLL load failed: The specified procedure could not be found.

Μάλλον φταίει ότι δεν έχω cuda, θα δοκιμάσω στο pc με την GTX

έκανα copy το env του danser (+ torch cuda for graphrec)

conda install pytorch torchvision torchaudio cudatoolkit=11.0 -c pytorch

File "C:\Users\pc\Downloads\REC SYS CODE\GraphRec-WWW19-master\GraphRec-WWW19-master\run_GraphRec_example.py", line 16, in <module>

from sklearn.metrics import mean_squared_error

ModuleNotFoundError: No module named 'sklearn'

install sklearn

τρέχω το toy_dataset.pickle για input

ΑΠΟΤΕΛΕΣΜΑΤΑ PAPER

Table 3: Performance comparison of different recommender systems

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

Για το epinions : MAE 0.8168 RMSE 1.0631

ΑΠΟΤΕΛΕΣΜΑΤΑ ΔΙΚΑ ΜΟΥ

Με toy dataset

runfile('C:/Users/pc/Downloads/REC SYS CODE/GraphRec-WWW19-master/GraphRec-WWW19-master/run_GraphRec_example.py', wdir='C:/Users/pc/Downloads/REC SYS CODE/GraphRec-WWW19-master/GraphRec-WWW19-master')

Reloaded modules: UV_Encoders, UV_Aggregators, Attention, Social_Encoders, Social_Aggregators

CUDA OK

[22, 0] loss: 0.005, The best rmse/mae: 0.805929 / 0.616961

[22, 100] loss: 0.679, The best rmse/mae: 0.805929 / 0.616961

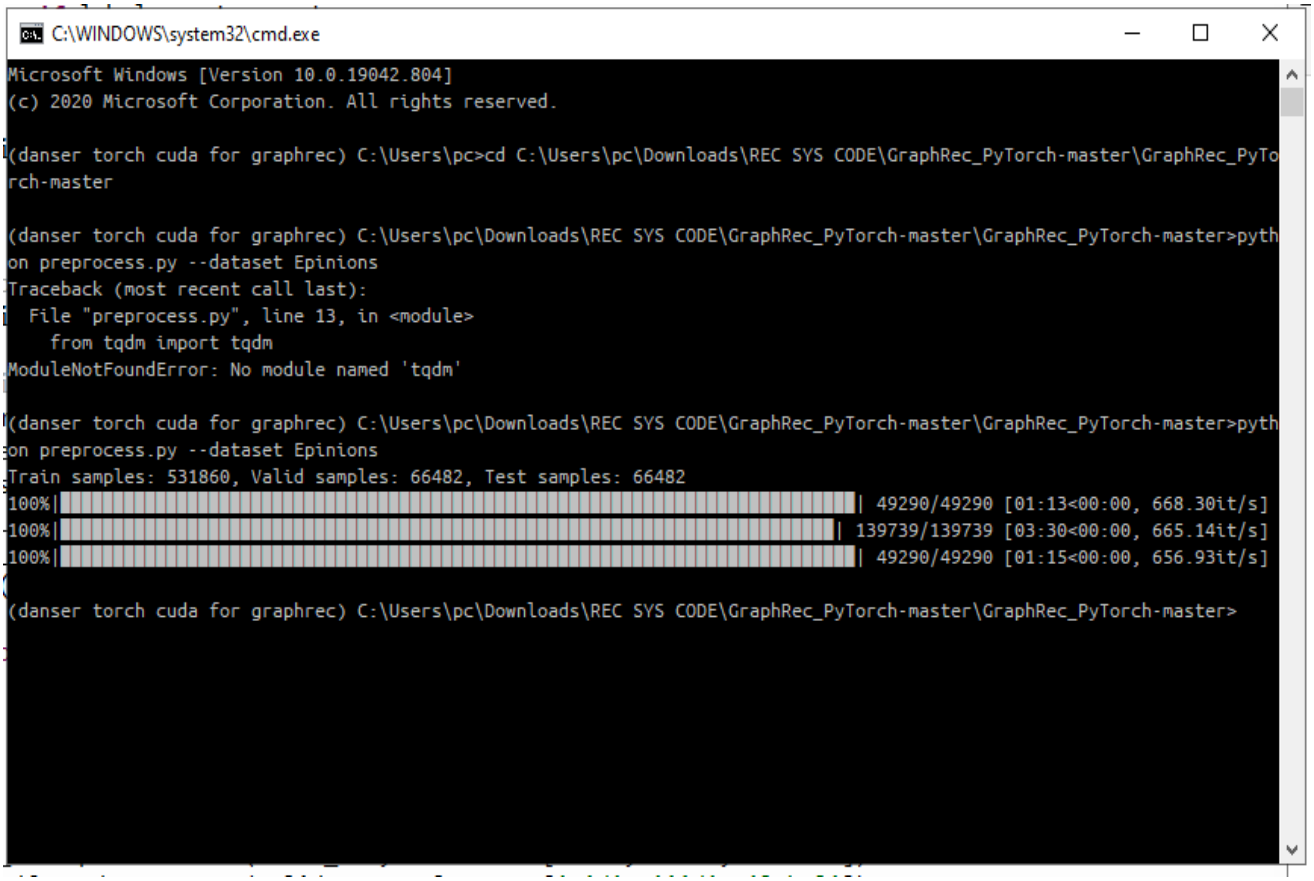
rmse: 0.8320, mae:0.6239

Graphrec pytorch

Περιβάλλον Danser Torch Cuda

python preprocess.py --dataset **Epinions**

θελει tqdm έκδοση 4.4.5

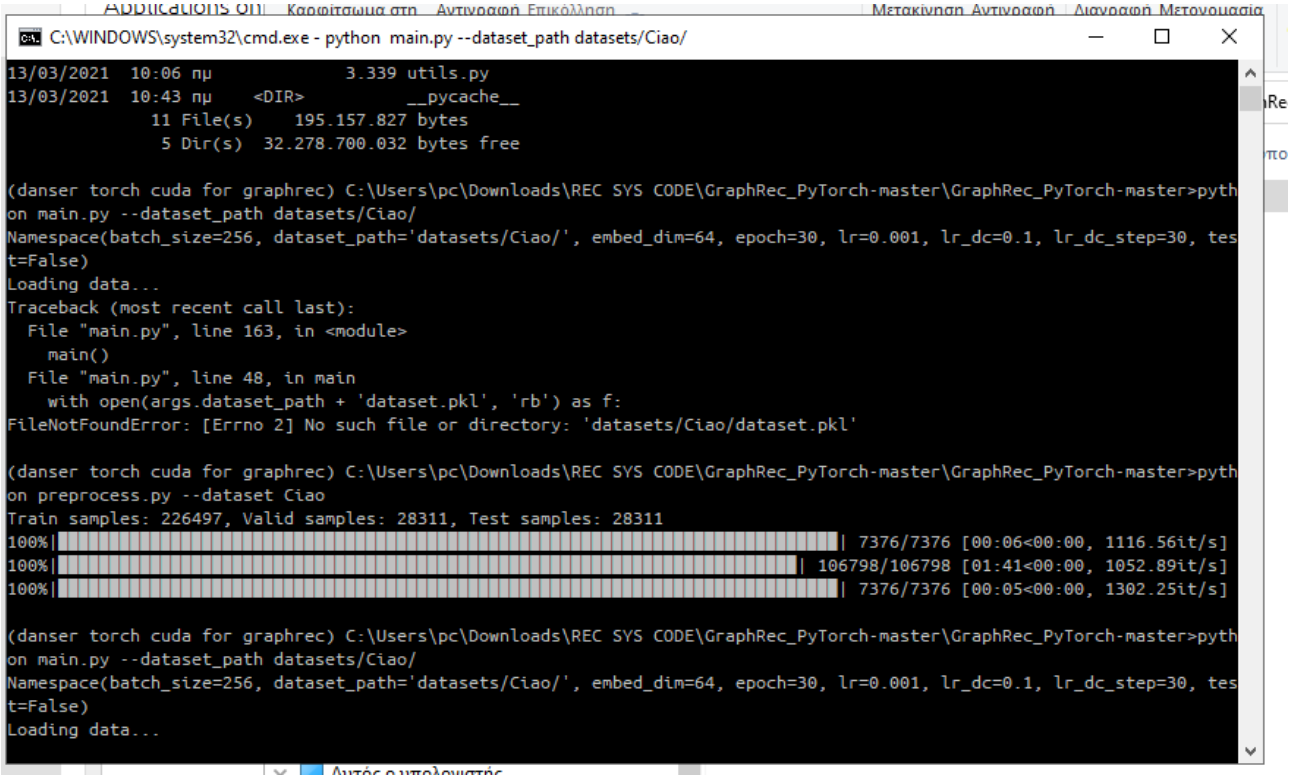


python main.py --dataset_path datasets/Epinions/

θελει pytorch μικροτερο από 1.1.0 αλλάζω το περιβαλλον της anaconda se torch 1.0.1

Άρχισε στις 10πμ περιπου 10λεπτα x 30 = περιπου 5 ωρες στο παλιο PC

CIAO dataset:



Training time από 2,5 ωρες GPU utilization περίπου 8%, 3,3 iterations per second

Με batch size 1024 , ανεβαίνει η χρήση της ram της GPU σε 4 GB. Με 1,15 it/sec

conda install -c peterjc123 pytorch-cpu

μετα κολαει στο

from torch._C import * ImportError: DLL load failed: The specified procedure could not be found.

Μετα

from torch._six import nan

ImportError: cannot import name 'nan'

Αποτελέσματα Εκτέλεσης

Μετα την εκτέλεση του main.py για training τρεχω το

python main.py –-test

[TRAIN] epoch 29/30 batch loss: 0.4730 (avg 0.4768) (828.48 im/s)| 1900/2078 [09:19<00:53, 3.34it/s]

[TRAIN] epoch 29/30 batch loss: 0.4644 (avg 0.4768) (867.79 im/s)| 2000/2078 [09:48<00:23, 3.39it/s]

100%| 2078/2078 [10:11<00:00, 3.40it/s]

100%| 260/260 [01:03<00:00, 4.12it/s]

Epoch 28 validation: MAE: 0.8651, RMSE: 1.1904, Best MAE: 0.8651| 260/260 [01:03<00:00, 4.68it/s]

97%| | 29/30 [5:42:26<11:29, 689.74s/it] [TRAIN] epoch 30/30 batch loss: 0.4818 (avg 0.4818) (744.22 im/s) | 0/2078 [00:00<?, ?it/s]

[TRAIN] epoch 30/30 batch loss: 0.5182 (avg 0.4617) (888.90

.....

im/s)| 2000/2078 [09:48<00:22, 3.53it/s]

100%| 2078/2078 [10:11<00:00, 3.40it/s]

100%| 260/260 [01:03<00:00, 4.11it/s]

Epoch 29 validation: MAE: 0.8832, RMSE: 1.2150, Best MAE: 0.8651| 260/260 [01:03<00:00, 4.61it/s]

100%| 30/30 [5:53:40<00:00, 707.37s/it]

(danser torch cuda for graphrec) C:\Users\pc\Downloads\REC SYS CODE\GraphRec_PyTorch-master\GraphRec_PyTorch-master>python main.py --dataset_path datasets/Epinions/ --test

Namespace(batch_size=256, dataset_path='datasets/Epinions/', embed_dim=64, epoch=30, lr=0.001, lr_dc=0.1, lr_dc_step=30, test=True)

Loading data...

Load checkpoint and testing...

100%|| 260/260 [01:23<00:00, 3.13it/s]

Test: MAE: 0.8629, RMSE: 1.1901

ΑΠΟΤΕΛΕΣΜΑΤΑ PAPER

Table 3: Performance comparison of different recommender systems

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

KGCN LS

ΠΕΡΙΒΑΛΛΟΝ

Υλοποιήθηκε σε danser περιβάλλον με py 3.6 tf 1.7

Dataset το movielense 20M

Εκτέλεση του preprocess.py

runfile('C:/Users/DeskMini/Documents/REC SYS CODE/KGNN-LS-master/KGNN-LS-master/src/preprocess.py', wdir='C:/Users/DeskMini/Documents/REC SYS CODE/KGNN-LS-master/KGNN-LS-master/src')

reading item index to entity id file: ../data/movie/item_index2entity_id.txt ...

reading rating file ...

converting rating file ...

number of users: 138159

number of items: 16954

converting kg file ...

number of entities (containing items): 102569

number of relations: 32

done

Νέο environment με sklearn από copy του danser

conda install -c anaconda scikit-learn

εγκατάσταση του sklearn 0.23.2

ΑΠΟΤΕΛΕΣΜΑΤΑ PAPER

Model	MovieLens-20M				Book-Crossing				Last.FM				Dianping-Food			
	R@2	*R@10*	*R@50*	*R@100*	*R@2*	*R@10*	*R@50*	*R@100*	*R@2*	*R@10*	*R@50*	*R@100*	*R@2*	*R@10*	*R@50*	*R@100*
SVD	0.036	0.124	0.277	0.401	0.027	0.046	0.077	0.109	0.029	0.098	0.240	0.332	0.039	0.152	0.329	0.451
LibFM	0.039	0.121	0.271	0.388	0.033	0.062	0.092	0.124	0.030	0.103	0.263	0.330	0.043	0.156	0.332	0.448
LibFM + TransE	0.041	0.125	0.280	0.396	0.037	0.064	0.097	0.130	0.032	0.102	0.259	0.326	0.044	0.161	**0.343**	0.455
PER	0.022	0.077	0.160	0.243	0.022	0.041	0.064	0.070	0.014	0.052	0.116	0.176	0.023	0.102	0.256	0.354
CKE	0.034	0.107	0.244	0.322	0.028	0.051	0.079	0.112	0.023	0.070	0.180	0.296	0.034	0.138	0.305	0.437
RippleNet	**0.045**	0.130	0.278	0.447	0.036	0.074	0.107	0.127	0.032	0.101	0.242	0.336	0.040	0.155	0.328	0.440
KGNN-LS	0.043	**0.155**	**0.321**	**0.458**	**0.045**	**0.082**	**0.117**	**0.149**	**0.044**	**0.122**	**0.277**	**0.370**	**0.047**	**0.170**	0.340	**0.487**

Table 3: The results of *Recall@K* in top-K recommendation.

Model	Movie	Book	Music	Restaurant
SVD	0.963	0.672	0.769	0.838
LibFM	0.959	0.691	0.778	0.837
LibFM + TransE	0.966	0.698	0.777	0.839
PER	0.832	0.617	0.633	0.746
CKE	0.924	0.677	0.744	0.802
RippleNet	0.960	0.727	0.770	0.833
KGCN-LS	**0.979**	**0.744***	**0.803***	**0.850**
KGCN-avg	0.975	0.722	0.774	0.844

Πίνακας: Αποτέλεσμα μέτρησης AUC ως προς την πρόβλεψη επιλογής στοιχείου (CTR)

Αποτελέσματα Εκτέλεσης

reading rating file ...

splitting dataset ...

reading KG file ...

constructing knowledge graph ...

constructing adjacency matrix ...

data loaded.

2021-03-03 07:39:45.561798: I tensorflow/core/platform/cpu_feature_guard.cc:140] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2

epoch 8 train auc: 0.9977 f1: 0.9817 eval auc: 0.9760 f1: 0.9301 test auc: 0.9761 f1: 0.9305

epoch 9 train auc: 0.9979 f1: 0.9827 eval auc: 0.9756 f1: 0.9296 test auc: 0.9757 f1: 0.9298

training time 07:39am - 8:11 am

DIFFNET++

python2.7, tensorflow-gpu-1.12.0

εβαλα py 3.6 γιατι το tf gpu δεν παιζει σε pt2.7

conda create -n tensorflow python=2.7

conda activate tensorflow

import numpy as np

ImportError: No module named numpy

Np 1.9.3

yelp and flikr

.npy

.Links

.ratings

(base) mark@mark-ubu:~/Downloads/diffnet++/diffnet-master/Diffnet++$ python entry.py --data_name=yelp --model_name=diffnetplus --gpu=0

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/conf/yelp_diffnetplus.ini

System start to prepare parser config file...

('num_users', 'int 17237')

('num_items', 'int 38342')

('gpu_device', 'int 1')

('data_name', 'string yelp')

```
('model_name', 'string diffnetplus')

('dimension', 'int 64')

('learning_rate', 'float 0.0005')

('epochs', 'int 450')

('num_negatives', 'int 8')

('num_evaluate', 'int 1000')

('num_procs', 'int 16')

('top10', 'int 10')

('top5', 'int 5')

('top15', 'int 15')

('evaluate_batch_size', 'int 2560')

('training_batch_size', 'int 512')

('epoch_notice', 'int 300')

('pretrain_flag', 'int 0')

('pre_model', 'string diffnet_hr_0.3437_ndcg_0.2092_epoch_98.ckpt')

System start to load data...

Data has been loaded successfully, cost:16.6082s

System start to load graph...

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:71: RuntimeWarning: divide by zero
encountered in true_divide

  self.consumed_items_num_input = 1.0/np.reshape(data_dict['CONSUMED_ITEMS_NUM_INPUT'], [-1,1])

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:98: RuntimeWarning: divide by zero
encountered in true_divide

  self.item_customer_num_input = 1.0/np.reshape(data_dict['ITEM_CUSTOMER_NUM_INPUT'],[-1,1])

WARNING:tensorflow:From /home/mark/anaconda2/lib/python2.7/site-
packages/tensorflow/python/ops/sparse_grad.py:281: calling sparse_reduce_sum (from
tensorflow.python.ops.sparse_ops) with keep_dims is deprecated and will be removed in a future version.

Instructions for updating:

keep_dims is deprecated, use keepdims instead

()

Following will output the evaluation of the model:

Traceback (most recent call last):

  File "entry.py", line 44, in <module>

    executeTrainModel(config_path, model_name)

  File "entry.py", line 25, in executeTrainModel
```

```
    starter.start(conf, data, model, evaluate)
  File "/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/train.py", line 140, in start
    negative_predictions = getNegativePredictions()
  File "/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/train.py", line 128, in getNegativePredictions
    feed_dict=eva_feed_dict
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/client/session.py", line 929, in run
    run_metadata_ptr)
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/client/session.py", line 1152, in _run
    feed_dict_tensor, options, run_metadata)
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/client/session.py", line 1328, in _do_run
    run_metadata)
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/client/session.py", line 1348, in _do_call
    raise type(e)(node_def, op, message)
tensorflow.python.framework.errors_impl.ResourceExhaustedError: OOM when allocating tensor with
shape[2560000,256] and type float on /job:localhost/replica:0/task:0/device:GPU:0 by allocator GPU_0_bfc
         [[node GatherNd (defined at /home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:529)  =
GatherNd[Tindices=DT_INT32, Tparams=DT_FLOAT,
_device="/job:localhost/replica:0/task:0/device:GPU:0"](concat_4, _arg_Placeholder_1_0_1/_235)]]
Hint: If you want to see a list of allocated tensors when OOM happens, add report_tensor_allocations_upon_oom to
RunOptions for current allocation info.
Caused by op u'GatherNd', defined at:
  File "entry.py", line 44, in <module>
    executeTrainModel(config_path, model_name)
  File "entry.py", line 25, in executeTrainModel
    starter.start(conf, data, model, evaluate)
  File "/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/train.py", line 42, in start
    model.startConstructGraph()
  File "/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py", line 16, in startConstructGraph
    self.constructTrainGraph()
  File "/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py", line 529, in constructTrainGraph
    latest_user_latent = tf.gather_nd(self.final_user_embedding, self.user_input)
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/ops/gen_array_ops.py", line 3240, in
gather_nd
    "GatherNd", params=params, indices=indices, name=name)
```

```
  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/framework/op_def_library.py", line 787, in
_apply_op_helper

    op_def=op_def)

  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/util/deprecation.py", line 488, in new_func

    return func(*args, **kwargs)

  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/framework/ops.py", line 3274, in create_op

    op_def=op_def)

  File "/home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/framework/ops.py", line 1770, in __init__

    self._traceback = tf_stack.extract_stack()


ResourceExhaustedError (see above for traceback): OOM when allocating tensor with shape[2560000,256] and type
float on /job:localhost/replica:0/task:0/device:GPU:0 by allocator GPU_0_bfc

         [[node GatherNd (defined at /home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:529)  =
GatherNd[Tindices=DT_INT32, Tparams=DT_FLOAT,
_device="/job:localhost/replica:0/task:0/device:GPU:0"](concat_4, _arg_Placeholder_1_0_1/_235)]]

Hint: If you want to see a list of allocated tensors when OOM happens, add report_tensor_allocations_upon_oom to
RunOptions for current allocation info.



(base) mark@mark-ubu:~/Downloads/diffnet++/diffnet-master/Diffnet++$ python entry.py --data_name=yelp --
model_name=diffnetplus --gpu=1

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/conf/yelp_diffnetplus.ini

System start to prepare parser config file...

('num_users', 'int 17237')

('num_items', 'int 38342')

('gpu_device', 'int 1')

('data_name', 'string yelp')

('model_name', 'string diffnetplus')

('dimension', 'int 64')

('learning_rate', 'float 0.0005')

('epochs', 'int 450')

('num_negatives', 'int 8')

('num_evaluate', 'int 1000')

('num_procs', 'int 16')

('top10', 'int 10')

('top5', 'int 5')
```

('top15', 'int 15')

('evaluate_batch_size', 'int 2560')

('training_batch_size', 'int 512')

('epoch_notice', 'int 300')

('pretrain_flag', 'int 0')

('pre_model', 'string diffnet_hr_0.3437_ndcg_0.2092_epoch_98.ckpt')

System start to load data...

Data has been loaded successfully, cost:16.7098s

System start to load graph...

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:71: RuntimeWarning: divide by zero encountered in true_divide

self.consumed_items_num_input = 1.0/np.reshape(data_dict['CONSUMED_ITEMS_NUM_INPUT'], [-1,1])

/home/mark/Downloads/diffnet++/diffnet-master/Diffnet++/diffnetplus.py:98: RuntimeWarning: divide by zero encountered in true_divide

self.item_customer_num_input = 1.0/np.reshape(data_dict['ITEM_CUSTOMER_NUM_INPUT'],[-1,1])

WARNING:tensorflow:From /home/mark/anaconda2/lib/python2.7/site-packages/tensorflow/python/ops/sparse_grad.py:281: calling sparse_reduce_sum (from tensorflow.python.ops.sparse_ops) with keep_dims is deprecated and will be removed in a future version.

Instructions for updating:

keep_dims is deprecated, use keepdims instead

()

Following will output the evaluation of the model:

Epoch:99, compute loss cost:22.6094s, train loss:721.8492, val loss:881.5691, test loss:4598.2949

Evaluate cost:14.4630s

Top5: hr:0.2500, ndcg:0.1856

Top10: hr:0.3628, ndcg:0.2262

Top15: hr:0.4418, ndcg:0.2499

LightGCN

- tensorflow == 1.11.0
- numpy == 1.14.3
- scipy == 1.1.0
- sklearn == 0.19.1
- cython == 0.29.15

https://github.com/kuandeng/LightGCN

Περιβάλλον ίδιο με του danser με τα παρακάτω

tensorflow 1.11

(LightCGN) C:\Users\DeskMini\Documents\REC SYS CODE\LightGCN-parallelized-version-master\LightGCN-parallelized-version-master>python LightGCN.py --dataset yelp2018 --regs [1e-4] --embed_size 64 --layer_size [64,64,64,64] --lr 0.001 --batch_size 16384 --epoch 10 --verbose 1
n_users=31668, n_items=38048
n_interactions=1561406
n_train=1237259, n_test=324147, sparsity=0.00130
already load adj matrix (69716, 69716) 0.20804834365844727
use the pre adjcency matrix
using xavier initialization
without pretraining.
Epoch 0 [200.0s]: train==[0.68774=0.68774 + 0.00000]
Epoch 1 [172.0s]: train==[0.48967=0.48958 + 0.00009]
Epoch 2 [184.3s]: train==[0.25370=0.25341 + 0.00029]
Epoch 3 [179.0s]: train==[0.18970=0.18925 + 0.00044]
Epoch 4 [179.3s]: train==[0.17016=0.16962 + 0.00054]
Epoch 5 [181.5s]: train==[0.16231=0.16170 + 0.00060]
Epoch 6 [178.1s]: train==[0.15661=0.15596 + 0.00065]
Epoch 7 [176.5s]: train==[0.15404=0.15335 + 0.00069]
Epoch 8 [179.6s]: train==[0.15081=0.15009 + 0.00071]
Epoch 9 [177.2s]: train==[0.14778=0.14705 + 0.00074]
Traceback (most recent call last):
  File "LightGCN.py", line 715, in <module>
    best_rec_0 = max(recs[:, 0])
IndexError: too many indices for array

train== [loss= mf_loss + emb_loss]

Table 4: The comparison of overall performance among LightGCN and competing methods.

Dataset	Gowalla		Yelp2018		Amazon-Book	
Method	recall	ndcg	recall	ndcg	recall	ndcg
NGCF	0.1570	0.1327	0.0579	0.0477	0.0344	0.0263
Mult-VAE	0.1641	0.1335	0.0584	0.0450	0.0407	0.0315
GRMF	0.1477	0.1205	0.0571	0.0462	0.0354	0.0270
GRMF-norm	0.1557	0.1261	0.0561	0.0454	0.0352	0.0269
LightGCN	**0.1830**	**0.1554**	**0.0649**	**0.0530**	**0.0411**	**0.0315**

perf_str = 'Epoch %d [%.1fs]: train==[%.5f=%.5f + %.5f]' % (epoch, time() - t1, loss, mf_loss, emb_loss)

(LightCGN) C:\Users\DeskMini\Documents\REC SYS CODE\LightGCN-parallelized-version-master\LightGCN-parallelized-version-master>python LightGCN.py --dataset yelp2018 --regs [1e-4] --embed_size 64 --layer_size [64,64,64,64] --lr 0.001 --batch_size 16384 --epoch 10 --verbose 1
n_users=31668, n_items=38048
n_interactions=1561406
n_train=1237259, n_test=324147, sparsity=0.00130
already load adj matrix (69716, 69716) 0.20804834365844727
use the pre adjcency matrix
using xavier initialization
without pretraining.
Epoch 0 [200.0s]: train==[0.68774=0.68774 + 0.00000]

Epoch 0 [200.0s]: train==[0.68774=0.68774 + 0.00000]
Epoch 1 [172.0s]: train==[0.48967=0.48958 + 0.00009]
Epoch 2 [184.3s]: train==[0.25370=0.25341 + 0.00029]

train== [loss= mf_loss + emb_loss]

(LightCGN) C:\Users\DeskMini\Documents\REC SYS CODE\LightGCN-parallelized-version-master\LightGCN-parallelized-version-master>python LightGCN.py --dataset yelp2018 --regs [1e-4] --embed_size 64 --layer_size [64,64,64,64] --lr 0.001 --batch_size **8192** --epoch 1000 --verbose 1
n_users=31668, n_items=38048
n_interactions=1561406
n_train=1237259, n_test=324147, sparsity=0.00130
already load adj matrix (69716, 69716) 0.21204280853271484
use the pre adjcency matrix

using xavier initialization
without pretraining.
Epoch 0 [367.0s]: train==[0.59747=0.59743 + 0.00004]

On 3/5/2021 8:39 AM, Markos S wrote:

Epoch 0 [1373.6s]: train==[0.32521=0.32490 + 0.00031]
Epoch 1 [1353.5s]: train==[0.16059=0.15993 + 0.00066]

On 3/5/2021 8:25 AM, Markos S wrote:

θελει 18,89 ημέρες

(LightCGN) C:\Users\DeskMini\Documents\REC SYS CODE\LightGCN-parallelized-version-master\LightGCN-parallelized-version-master>python LightGCN.py --dataset yelp2018 --regs [1e-4] --embed_size 64 --layer_size [64,64,64,64] --lr 0.001 --batch_size 2048 --epoch 1000 --verbose 1
n_users=31668, n_items=38048
n_interactions=1561406
n_train=1237259, n_test=324147, sparsity=0.00130
already load adj matrix (69716, 69716) 0.2112736701965332
use the pre adjcency matrix
using xavier initialization
without pretraining.
Epoch 0 [1373.6s]: train==[0.32521=0.32490 + 0.00031]

KGCN

ΠΕΡΙΒΑΛΛΟΝ

Python 3.6

Tf 1.2

Scikit learn 1.21.2

unzip ml-20m.zip

python preprocess.py -d movie

reading item index to entity id file: ../data/movie/item_index2entity_id.txt ...

reading rating file ...

converting rating file ...

number of users: 138159

number of items: 16954

converting kg file ...

number of entities (containing items): 102569

number of relations: 32

done

ΑΠΟΤΕΛΕΣΜΑ ΤΟΥ PAPER

Table 3: *AUC* result of KGCN with different neighbor sampling size K.

K	2	4	8	16	32	64
MovieLens-20M	0.978	**0.979**	0.978	0.978	0.977	0.978
Book-Crossing	0.680	0.727	**0.736**	0.725	0.711	0.723
Last.FM	0.791	0.794	**0.795**	0.793	0.794	0.792

Table 4: *AUC* result of KGCN with different depth of receptive field H.

H	1	2	3	4
MovieLens-20M	0.972	**0.976**	0.974	0.514
Book-Crossing	**0.738**	0.731	0.684	0.547
Last.FM	**0.794**	0.723	0.545	0.534

Table 5: *AUC* result of KGCN with different dimension of embedding.

d	4	8	16	32	64	128
MovieLens-20M	0.968	0.970	0.975	**0.977**	0.973	0.972
Book-Crossing	0.709	0.732	0.733	0.735	**0.739**	0.736
Last.FM	0.789	0.793	**0.797**	0.793	0.790	0.789

is more sensitive to H compared to K. We observe the occurrence of serious model collapse when $H = 3$ or 4, as a larger H brings massive noises to the model. This is also in accordance with our intuition, since a too long relation-chain makes little sense when inferring inter-item similarities. An H of 1 or 2 is enough for real cases according to the experiment results.

4.4.3 Impact of dimension of embedding. Lastly, we examine the

Αποτελέσματα Εκτέλεσης

epoch 8 train auc: 0.9965 f1: 0.9751 eval auc: 0.9749 f1: 0.9279 test auc: 0.9750 f1: 0.9283

epoch 9 train auc: 0.9968 f1: 0.9765 eval auc: 0.9743 f1: 0.9272 test auc: 0.9744 f1: 0.9274