Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων
Πληροφορικής

Επανεκπαίδευση με επίγνωση κόστους για μοντέλα κωδικοποίησης κόμβων σε δυναμικά δίκτυα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΔΡΕΑΣ Γ. ΜΑΝΙΑΤΗΣ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιανουάριος 2024

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων
Πληροφορικής

Επανεκπαίδευση με επίγνωση κόστους για μοντέλα κωδικοποίησης κόμβων σε δυναμικά δίκτυα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΔΡΕΑΣ Γ. ΜΑΝΙΑΤΗΣ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 14η Μαρτίου 2024.

.....	.....	.....
Συμεών Παπαβασιλείου	Μιχαήλ Μαθιουδάκης	Ελένη Στάη
Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής University of Helsinki	Επίκουρη Καθηγήτρια Ε.Μ.Π.

Αθήνα, Ιανουάριος 2024

.....

Ανδρέας Γ. Μανιάτης

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

Copyright © Ανδρέας Γ. Μανιάτης, 2024

Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Η παρούσα διπλωματική εργασία εξετάζει μοντέλα αναπαράστασης κόμβων γράφων σε δυναμικό περιβάλλον και προτείνει μια ολοκληρωμένη πρόταση για την αποδοτική τους επανεκπαίδευση με βάση το κόστος εκπαίδευσης, αλλά και την τρέχουσα ακρίβεια τους. Αρχικά, παρουσιάζεται το θεωρητικό πλαίσιο της ολίσθησης δεδομένων (data drift) και πώς αυτό παρουσιάζεται σε προβλήματα μηχανικής μάθησης για ευκλείδεια δεδομένα, επηρεάζοντας την ακρίβεια των online μοντέλων και ακολουθεί η συσχέτιση του φαινομένου αυτού με το κόστος επανεκπαίδευσης για την εξαγωγή ενός συνολικού κόστους στρατηγικής. Ακολουθείται η θεμελίωση της λογική της στασιμότητας μοντέλου (model staleness) που εισάγεται από προηγούμενη έρευνα, έτσι ώστε η υποβάθμιση της προβλεπτικής ικανότητας του μοντέλου να υπολογιστεί με βάση συγκεκριμένο όγκο εργασίας (queries). Γίνεται τέλος η σύνδεση του προβλήματος με δεδομένα γράφων με την πρόταση απεικόνισης των γραφικών δεδομένων σε διανυσματικό πεδίο και τη χρήση του νευρωνικού μοντέλου γράφων, GraphSAGE. Τέλος παρουσιάζουμε πώς θα χρησιμοποιήσουμε τον αλγόριθμο CARA, που έχει προταθεί για ευκλείδεια δεδομένα, για το πρόβλημά μας και την τελική υλοποίηση της συνάρτησης απόφασης για την πληροφορημένη επανεκπαίδευση μοντέλων.

Αρχικά, προχωρούμε με μια μελέτη της συμπεριφοράς του μοντέλου GraphSAGE για δυναμικούς γράφους και το πώς η ακρίβειά του στην εργασία της πρόβλεψης ετικετών κόμβων γράφου επηρεάζεται από τον τρόπο δυναμικής ανάπτυξής του (θεωρία ομοφιλίας). Στη συνέχεια, εξετάζουμε μια πλήρη σειρά πειραμάτων πάνω σε συνθετικά δεδομένα, ώστε να δοκιμάσουμε τις υποθέσεις μας και λαμβάνουμε αποτελέσματά που επιβεβαιώνουν την αποτελεσματικότητα του μοντέλου απόφασης σε δυναμικούς γράφους. Τέλος, δοκιμάζουμε το μοντέλο μας σε ένα σύνολο δεδομένων που προσεγγίζει αριθμό κόμβων της τάξης 10^6 και παίρνουμε ικανοποιητικά αποτελέσματα που ξεπερνούν συμβατικά baselines και προσεγγίζουν τη βέλτιστη λύση.

Λέξεις Κλειδιά

Δυναμικοί γράφοι, Διανυσματικές αναπαραστάσεις κόμβων γραφου (node embeddings), Νευρωνικά Μοντέλα Γράφων (Neural Graph Networks), Ολίσθηση δεδομένων (Data Drift), GraphSAGE, Στασιμότητα Μοντέλου (Model Staleness), Κόστος επανεκπαίδευσης (Retraining cost), Αλγόριθμος απόφασης επανεκπαίδευσης

Abstract

This thesis examines graph representation models in a dynamic environment and proposes a comprehensive proposal for their efficient retraining based on both cost and accuracy. First, the theoretical framework of data drift and how it occurs in machine learning problems for Euclidean data, affecting the accuracy of online models is presented, followed by the correlation of this phenomenon with the retraining cost to derive an overall strategy cost. The logic of model staleness introduced by previous research is followed, so that the degradation of the model's predictive ability is calculated based on a certain workload, called queries. We finally make the connection to the graph data problem by proposing a vector field representation of the graph data and the use of the graph neural model, GraphSAGE. Finally, we show how we use the algorithm CARA, proposed for Euclidean data, for our problem and the final implementation of the decision function for informed model retraining.

First, we proceed with a study of the behavior of the GraphSAGE model for dynamic graphs and how its accuracy in the task of predicting graph node labels is affected by the way it is expanded (homophily theory). We then consider a full set of experiments on synthetic data to test our hypotheses and obtain results that confirm the effectiveness of the decision algorithms on dynamic graphs. Finally, we test our model on a big real-world dataset of the order of 10^6 nodes and obtain satisfactory results that outperform conventional baselines and approximate the optimal solution.

Keywords

Dynamic Graphs, Node Embeddings, Neural Graph Networks, Data Drift, Gra- phSAGE, Model Staleness, Retraining Cost, Retraining Decision Algorithm

Ευχαριστίες

Ολοκληρώνοντας τον προπτυχιακό μου κύκλο σπουδών, θα ήθελα να ευχαριστήσω θερμά τους ανθρώπους που συνέβαλαν σε αυτή την προσπάθεια:

- Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής εργασίας, κ. Συμεών Παπαβασιλείου, για την καλή συνεργασία που είχαμε και την εμπιστοσύνη του στην ανάθεση ενός θέματος που ανέπτυξε σημαντικά τις γνώσεις μου. Χάρη στο μάθημά του 'Ανάλυσης Κοινωνικών Δικτύων' ενδιαφέρθηκα αρχικά για το συγκεκριμένο ερευνητικό πεδίο στο οποίο αναπτύχθηκε η διπλωματική μου.

- Ιδιαίτερες ευχαριστίες θα ήθελα να απευθύνω στον καθηγητή και επόπτη μου στο πανεπιστήμιο του Ελσίνκι, κ. Μιχαήλ Μαθιουδάκη, για την πολύτιμη καθοδήγησή του κατά τη διάρκεια της πρόσφατης πρακτικής μου, η οποία ωρίμασε τελικά στη παρούσα διπλωματική. Ιδιαίτερη μνεία θα ήθελα να κάνω και στους υποψήφιους διδάκτορες του εργαστηρίου του, Ananth και Sachith για την σημαντική υποστήριξή τους και τις παραγωγικές μας συζητήσεις που ανέδειξαν την ομορφιά της ερευνητικής διαδικασίας.

- Οφείλω ακόμη να ευχαριστήσω όλους τους φίλους και συμφοιτητές μου για την αδιαπραγμάτευτη και ανιδιοτελή τους υποστήριξη όλα αυτά τα χρόνια. Χάρη σε αυτούς και αυτές η ζωή μου έγινε πιο πλούσια και οι δυσκολίες πιο μικρές.

- Θέλω να ευχαριστήσω τους γονείς μου, Ζωή και Γιώργο για την αμέριστη στήριξή και ενθάρρυνσή τους που με ωθούσε από μικρό να εξερευνώ τα ενδιαφέροντα και τις ανησυχίες μου, καθώς και για την μετάδοση της αγάπης για το διάβασμα. Τέλος, ευχαριστώ τον αδελφό μου Κωνσταντίνο για την υποστήριξή του καθόλη τη διάρκεια των σπουδών μου και του εύχομαι να λάβει την ίδια ικανοποίηση από το εκπαιδευτικό ταξίδι που έχει ξεκινήσει και αυτός πια.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Στο διαρκώς εξελισσόμενο πεδίο της επιστήμης δικτύων και της μηχανικής μάθησης, η αναπαράσταση των κόμβων μέσω embeddings χαμηλών διαστάσεων έχει φέρει επανάσταση στην κατανόηση και τη χρήση πολύπλοκων συστημάτων. Τα node embeddings, που χρησιμεύουν ως συμπυκνωμένες αλλά πλούσιες σε πληροφορίες απεικονίσεις των δομών δικτύων, έχουν γίνει μάρτυρες αξιοσημείωτων επιστημονικών εξελίξεων. Οι πρόσφατες ανακαλύψεις στον τομέα δεν έχουν μόνο ενισχύσει την ικανότητά μας να αποτυπώνουμε περίπλοκες σχέσεις εντός στατικών δικτύων, αλλά έχουν επίσης προκαλέσει το ενδιαφέρον για την αντιμετώπιση των προκλήσεων που θέτουν τα δυναμικά δίκτυα, όπου οι δομικές αλλαγές συμβαίνουν με την πάροδο του χρόνου. Η έλευση προηγμένων μοντέλων και αλγορίθμων μηχανικής μάθησης έχει διαδραματίσει καθοριστικό ρόλο στην εξέλιξη της αναπαράστασης κόμβων. Μοντέλα όπως τα Νευρωνικά Δίκτυα Γράφων (GNNs) έχουν επιδείξει πρωτοφανή επιτυχία στην εκμάθηση και διατήρηση περίπλοκων δομών δικτύων, οδηγώντας σε πιο εκφραστικά και context-aware node embeddings. Η συγχώνευση των μεθοδολογιών που βασίζονται σε γράφους με τη βαθιά μάθηση έχει ανοίξει νέα σύνορα, επιτρέποντας την αναπαράσταση των κόμβων με τρόπο που ενσωματώνει τόσο τα τοπικά όσο και τα παγκόσμια χαρακτηριστικά του δικτύου.

Παρά τα βήματα αυτά, ο δυναμισμός που ενυπάρχει στα δίκτυα του πραγματικού κόσμου έχει ωθήσει σε μια πρόσφατη αλλαγή παραδείγματος προς την κατεύθυνση της κατανόησης και της προσαρμογής των node embeddings σε δυναμικά σενάρια. Οι επιστημονικές προσπάθειες σε αυτόν τον τομέα έχουν αναγνωρίσει τους περιορισμούς των παραδοσιακών προσεγγίσεων που έχουν σχεδιαστεί για στατικά δίκτυα και ασχολούνται τώρα με τις χρονικές πτυχές της εξέλιξης των δικτύων, διερευνώντας στρατηγικές για τον μετριασμό της υποβάθμισης των αναπαραστάσεων κόμβων καθώς εξελίσσονται οι δομικές αλλαγές. Η διασταύρωση της χρονικής δυναμικής, της μηχανικής μάθησης και της επιστήμης των δικτύων έχει γίνει εστία εξερεύνησης, με πρωταρχικό στόχο να καταστεί δυνατή η απρόσκοπτη εφαρμογή της αναπαράστασης κόμβων σε δυναμικά περιβάλλοντα.

Συνήθως, καθώς το αποτέλεσμα αυτών των αλλαγών συσσωρεύεται και η ποιότητα της υπάρχουσας αναπαράστασης υποβαθμίζεται, μια (συχνά δαπανηρή) επανεκπαίδευση του μοντέλου αναπαράστασης δικτύου είναι τελικά αναγκαία για να να ληφθούν ενημερωμένα node embeddings. Αυτό εγείρει το ερώτημα: Πότε πρέπει να ληφθεί ένα ενημερωμένο node embedding;

Καθώς όμως τα τελευταία χρόνια τα νευρωνικά δίκτυα γίνονται όλο και πιο περίπλοκα η εκπαίδευσή τους γίνεται ολοένα και πιο υπολογιστικά κοστοβόρα, με το τίμημα να είναι μεγάλο για τους οργανισμούς ανάπτυξής τους. Για παράδειγμα, σύμφωνα με εκτιμήσεις [1][2] μία εκπαίδευση του μεγάλου γλωσσικού μοντέλου (large language model) GPT-3 απαιτεί περίπου 500.000

δολάρια.
Το κόστος όμως είναι σημαντικό και για το περιβάλλον, καθώς τα μεγάλα υπολογιστικά κόστη οδηγούν σε αυξημένες ανάγκες ενέργειας και αυξάνουν το αποτύπωμα άνθρακα μιας εφαρμογής. Με την στροφή της έρευνας προς την κλιμάκωση (scaling) των εφαρμογών και του υπολογισμού μια μεγαλύτερη κατανόηση των ενεργειακών κόστων αναδύεται με πολλές νέες έρευνες που μελετούν το οικολογικό αποτύπωμα της εκπαίδευσης μεγάλων μοντέλων.[3][4][5][6] Παρακάτω παρουσιάζουμε έναν πίνακα με συγκριτικά δεδομένα διαφορετικών μοντέλων βαθιάς μάθησης μεγάλης κλίμακας:

Πίνακας 1.1: AI Model Comparison[7]: Όνομα μοντέλου - Αριθμός παραμέτρων προς εκπαίδευση - Οργανισμός ανάπτυξης - Χρόνος εκπαίδευσης (σε μέρες) - Συνολικοί υπολογισμοί - Κατανάλωση ενέργειας - Ακαθάριστοι μετρικοί τόνοι CO2 εκπομπών για την εκπαίδευση

Model	Number of Parameters (B)	Developer	Training time (days)	Total Computation (FLOP)	Energy Consumption (MWh)	Gross tCO2 e for Model Training
Evolved Transformer NAS	0.064 per model	Google	6.8	2.91×10^{21}	7.5	3.2
T5	11	Google	20	4.05×10^{22}	85.7	46.7
Meena	2.6	Google	30	1.12×10^{23}	232	96.4
Gshard-600B	619	Google	3.1	1.33×10^{22}	24.1	4.8
Switch Transformer	1500	Google	27	8.22×10^{22}	179	72.2
GPT-3	175	OpenAI	14.8	3.14×10^{23}	1,287	552.1

Πέρα από τα καθαρά κόστη εκπαίδευσης ενός μοντέλου από την αρχή, υπάρχουν και πολλά άλλα 'κρυφά' κόστη. Αυτά εισέρχονται στη διαδιακασία και μειώνουν την αξία μιας επανεκπαίδευσης του μοντέλου για τον διαχειριστή του συστήματος ή αλλιώς προσθέτουν ένα επιπλέον κόστος. Αυτά μπορεί να αποτελούν:

- Ο χρόνος εκπαίδευσης, ο οποίος μπορεί να οδηγήσει το σύστημα εκτός σύνδεσης με τα επακόλουθα οικονομικά κόστη
- Το κόστος καθαρισμού και επεξεργασίας των νέων δεδομένων
- Το κόστος εργασίας (ML Engineers, Data Engineer, Data Scientist κ.τ.λ.)

Κατανοούμε επομένως πόσο σημαντική είναι η απόφαση επανεκπαίδευσης ενός μοντέλου και το πώς αυτή είναι αναγκαίο να συμπλακεί με τις γενικότερες επιδιώξεις του οργανισμού ή σε γενικότερο πλαίσιο του διαχειριστή. Από τη μία πλευρά, η πολύ συχνή επανεκπαίδευση του μοντέλου αναπαράστασης μπορεί να είναι περιττά δαπανηρή, ιδίως για πολύ μεγάλα δίκτυα- από την άλλη πλευρά, η λιγότερο συχνή επανεκπαίδευση μπορεί να έχει το κόστος της μειωμένης απόδοσης σε ένα downstream task πάνω στο οποίο εργαζόμαστε, ιδιώς όταν δουλεύουμε με δεδομένα των οποίων οι στατιστικές ιδιότητες μεταβάλλονται στο χρόνο, όπως οι δυναμικοί γράφοι. Το ερώτημα που μας παρουσιάζεται είναι πώς να επιλύσουμε το trade-off αυτό που αναδύεται μεταξύ κόστους και απόδοσης.

1.1 Συνεισφορά Διπλωματικής Εργασίας

Αυτή η εργασία θα καθορίσει και θα αξιολογήσει δυναμικούς αλγορίθμους λήψης αποφάσεων για το παραπάνω ερώτημα. Η απόδοση ενός αλγορίθμου απόφασης θα οριστεί και θα μετρηθεί για συγκεκριμένους φόρτους εργασίας που σχετίζονται με το downstream task. Για παράδειγμα, εάν ένα network embedding χρησιμοποιείται για την ταξινόμηση των κόμβων, ο σχετικός φόρτος εργασίας θα οριστεί ως η κατανομή των κόμβων για τους οποίους ζητείται η κλάση, και η απόδοση θα μετρηθεί ως το σφάλμα στην ταξινόμηση κόμβων για αυτούς τους κόμβους με την πάροδο του χρόνου. Η αξιολόγηση θα περιλαμβάνει αλγορίθμους απόφασης με επίγνωση του φόρτου εργασίας σε σύγκριση με βασικούς αλγορίθμους χωρίς επίγνωση του φόρτου εργασίας

1.2 Δομή Διπλωματικής Εργασίας

Στο κεφάλαιο 2 δίνουμε τον ορισμό της Μηχανικής Μάθησης, τα προβλήματα που μπορεί να επιλύσει, κάποιες βασικές κατηγορίες και μερικά απλά μοντέλα, καθώς και θεμελιώδεις αρχές που βρίσκονται παρούσες στην βάσιση ακόμα και πιο πολύπλοκων μοντέλων. Στο κεφάλαιο 3 αναλύουμε τον επαναστατικό τομέα της Βαθιάς Μάθησης και μετά από μια εισαγωγή στις δομές δεδομένων που αποτελούν οι γράφοι στο κεφάλαιο 4, επεκτείνουμε την ανάλυσή μας στο κεφάλαιο 5 για τη Βαθιά Μάθηση συγκεκριμένα σε γράφους και σε εφαρμογές τους. Στο κεφάλαιο 6 αναλύουμε σε βάθος το βασικό μοντέλο στο οποίο στηριχθήκαμε για τη μελέτη μας και εξηγούμε το θεωρητικό υπόβαθρο της συνεισφοράς μας σε βάθος. Στο κεφάλαιο 7 παρέχουμε μερικές λεπτομέρειες για την υλοποίησή μας και συνεχίζουμε στο κεφάλαιο 8 με τα πλήρη πειράματά μας και τα αποτελέσματα που είχαμε. Τέλος, στο κεφάλαιο 9 συνοψίζουμε τα αποτελέσματά μας και παρέχουμε μερικές κατευθύνσεις περαιτέρω ανάπτυξης της εργασίας μας.

Μέρος I

Θεωρητικό υπόβαθρο

Κεφάλαιο 2

Μηχανική Μάθηση

2.1 Ορισμός Μηχανικής Μάθησης

Ένας δημοφιλής ορισμός της Μηχανικής Μάθησης (Machine Learning - ML) σύμφωνα με τον Tom Mitchell[8] είναι:

- Ένα πρόγραμμα υπολογιστή θεωρείται ότι μαθαίνει από την εμπειρία E (experience)σε σχέση με κάποια κατηγορία εργασιών T (task), και το μέτρο απόδοσης Π (performance measure), αν η απόδοσή του στις εργασίες της T, όπως μετράται από το Π, βελτιώνεται με την εμπειρία E.

Προκύπτει λοιπόν ότι υπάρχουν πολλά διαφορετικά είδη μηχανικής μάθησης, ανάλογα με τη φύση της εργασίας T που επιθυμούμε να μάθει το σύστημα, τη φύση του μέτρου απόδοσης Π που χρησιμοποιούμε για την αξιολόγηση του συστήματος, και τη φύση του σήματος εκπαίδευσης ή της εμπειρίας E που του δίνουμε. Στη συνέχεια του κεφαλαίου θα αναφερθούμε σε μερικά από αυτά τα είδη.
Η εμπειρία E, έρχεται με τη μορφή παραδειγμάτων της οντότητας που μελετάμε ή πιο απλά με τη μορφή δεδομένων D. Το σύνολο των δεδομένων που το σύστημά μας χρησιμοποιεί για να μάθει καλείται σύνολο εκπαίδευσης (training set) και κάθε μέλος του δείγμα, με συνήθη συμβολισμό x.

2.2 Παραδείγματα εφαρμογών της Μηχανικής Μάθησης

Παραθέτουμε παρακάτω μερικά παραδείγματα εργασιών Μηχανικής Μάθησης μαζί με τις τεχνικές που μπορούν να χρησιμοποιηθούν για να τις επιλύσουν:

- Εντοπισμός καρκινικών όγκων σε τομογραφίες εγκεφάλων ή ταξινόμηση άγριων μανιταριών σε ασφαλή για κατανάλωση ή όχι, χρησιμοποιώντας συνήθως Βαθιά Συνελικτικά Δίκτυα.
- Αναπαράσταση πολυδιάστατων συνόλων δεδομένων σε ξεκάθαρες οπτικοποιήσεις δύο διαστάσεων, χρησιμοποιώντας τεχνικές μείωσης διαστατικότητας.
- Πρόβλεψη μελλοντικών τιμών μετοχών στο χρηματιστήριο, χρησιμοποιώντας τεχνικές ανάλυσης χρονοσειρών, παραδείγματος χάρη με Αναδρομικά Νευρωνικά Δίκτυα
- Εκπαίδευση ρομπότ σε ελεύθερη κινησιολογία για αποφυγή εμποδίων και εκμάθηση περιβάλλοντος χώρου, χρησιμοποιώντας τεχνικές Ενισχυμένης Μάθησης

- Ανίχνευση ύποπτων τραπεζικών συναλλαγών με μεθόδους εντοπισμού ανωμαλίας (Anomaly Detection)
- Ανίχνευση ρητορικής μίσους σε κοινωνικά δίκτυα και γενικότερα στο διαδίκτυο, χρησιμοποιώντας τεχνικές επεξεργασίας φυσικής γλώσσας (NLP).
- Αυτόματη παραγωγή στίχων από τραγούδια, με τεχνικές αναγνώρισης φωνής.

2.3 Κατηγορίες συστημάτων Μηχανικής Μάθησης

Ανάλογα με τον τύπο προβλημάτων στα οποία καλούνται να αντεπεξέλθουν τα συστήματα Μηχανικής Μάθησης, μπορούν να εφαρμοστούν διαφορετικά επίπεδα ανθρώπινης επίβλεψης (supervision) στη διαδικασία εκπαίδευσής τους. Μελετάμε λοιπόν στη συνέχεια τις βασικότερες κατηγορίες σύμφωνα με το κριτήριο της επίβλεψης.

2.3.1 Επιτηρούμενη Μάθηση

Η Επιτηρούμενη Μάθηση (Supervised Learning) είναι η πιο συνηθισμένη μορφή ML. Σε αυτό το πρόβλημα, η εργασία T είναι να μάθει μια απεικόνιση f από τις εισόδους $x \in \boldsymbol{X}$ στις εξόδους $y \in \boldsymbol{Y}$, όπου $\boldsymbol{X}$ και $\boldsymbol{Y}$ είναι διανύσματα με την ίδια διάσταση.
Οι είσοδοι x ονομάζονται επίσης χαρακτηριστικά ή προγνωστικοί παράγοντες και πρόκειται συχνά για ένα διάνυσμα σταθερής διάστασης που χαρακτηρίζει τις οντότητες με τις οποίες εκπαιδεύεται το μοντέλο μας, όπως το μήκος και το πλάτος ενός πετάλου λουλουδιού, ή ο αριθμός εικονοστοιχείων (pixels) μιας εικόνας. Η έξοδος y είναι επίσης γνωστή ως ετικέτα ή στόχος.
Η εμπειρία E δίνεται με τη μορφή ενός συνόλου N ζευγών εισόδου-εξόδου $D = (x_n, y_n)_{n=1}^{N}$, γνωστό ως σύνολο εκπαίδευσης. Το μέτρο απόδοσης Π εξαρτάται από τον τύπο της εξόδου που προβλέπουμε, όπως θα συζητήσουμε παρακάτω.
Δύο από τα κύρια tasks στην επιτηρούμενη μάθηση είναι η **ταξινόμηση** (classification) και η **παλινδρόμηση** (regression).
Στην ταξινόμηση, ο χώρος εξόδου (labels) είναι ένα σύνολο από C αμοιβαία αποκλειόμενες ετικέτες, γνωστά ως κλάσεις, $Y = 1, 2, ..., C$. Το πρόβλημα της πρόβλεψης της ετικέτας κλάσης δεδομένης μιας εισόδου ονομάζεται επίσης αναγνώριση προτύπων. Στην ειδική περίπτωση που υπάρχουν μόνο δύο κλάσεις, συνήθως $y \in \{0, 1\}$, το πρόβλημα ονομάζεται δυαδική ταξινόμηση.
Στην παλινδρόμηση, σε αντίθεση με την ταξινόμηση η ετικέτα που καλούμαστε να προβλέψουμε πρόκειται για πραγματικό αριθμό, είναι δηλαδή συνεχής και όχι διακριτή.

Παράδειγμα ταξινόμησης: Άνθοι ίριδας
Ένα από τα πιο μελετημένα σύνολα δεδομένων είναι αυτό των ανθών της ίριδας (Iris flower data set), το οποίο και έγινε γνωστό από τον Βρετανό επιστήμονα στατιστικής και βιολογίας Ronald Fisher[9]. Το σύνολο εκπαίδευσης αποτελείται από διανύσματα με τέσσερα στοιχεία: το μήκος και το πλάτος του πετάλου (petal) και το μήκος και το πλάτος του σεπάλου (sepal) ενός λουλουδιού, τιμές οι οποίες προέκυψαν από πραγματικές μετρήσεις. Οι κλάσεις του συνόλου δεδομένων είναι τρεις και αποτελούνται από τρία είδη ίριδας, τα: Iris setosa, Iris virginica, Iris versicolor.

Παράδειγμα παλινδρόμησης: Άνθοι ίριδας
Αντίθετα, στην παλινδρόμηση το task αποτελεί την πρόβλεψη μιας πραγματικής τιμής $y \in R$.

(α′) Iris setosa

(β′) Iris virginica

(γ′) Iris versicolor

Σχήμα 2.1: Διαφορετικά είδη-κλάσεις ίριδας

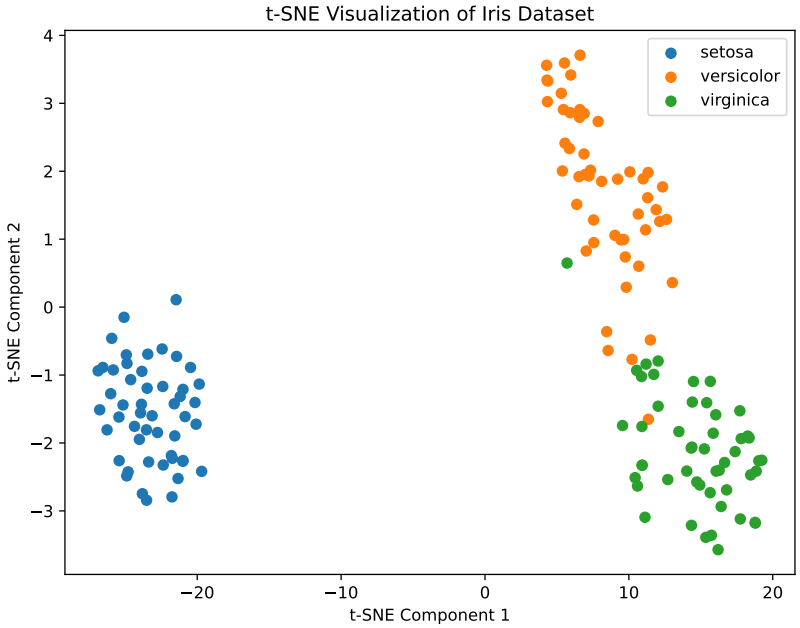
Στο παράδειγμα της ίριδας, η τιμή που θα προβλέπαμε θα μπορούσε να είναι οι μέρες ανθοφορίας του φυτού ή το ύψος του βλαστού του λουλουδιού. Λόγω του ότι στις δύο διαφορετικές περιπτώσεις έχουμε διαφορετικά είδη προβλέψεων, διαφορετικό θα είναι και το μέτρο της απόδοσης ή σφάλμα (loss).

2.3.2 Μη επιτηρούμενη Μάθηση

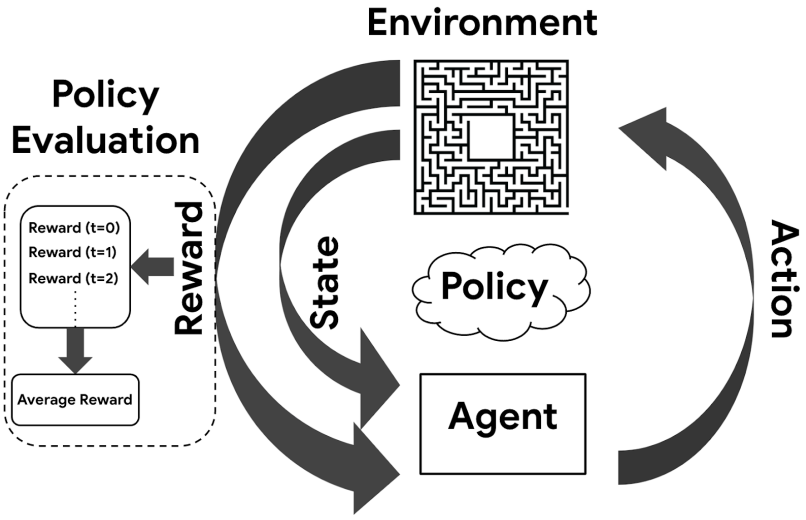
Στην Μη Επιτηρούμενη Μάθηση (Unsupervised Learning) το σύνολο δεδομένων δεν έχει ετικέτες, οπότε το σύστημά μας καλείται να εκπαιδευτεί χωρίς τιμές καθορισμένες από τον ανθρώπινο παράγοντα.
Ο στόχος σε τέτοια μη επιτηρούμενα προβλήματα μάθησης μπορεί να είναι η ανακάλυψη ομάδων παρόμοιων παραδειγμάτων μέσα στα δεδομένα, που καλείται ομαδοποίηση (clustering), ο προσδιορισμός της κατανομής των δεδομένων εντός του χώρου χαρακτηριστικών, γνωστό ως εκτίμηση πυκνότητας (density estimation), ή η προβολή των δεδομένων από έναν υψηλής διάστασης χώρο σε δύο ή τρεις διαστάσεις με σκοπό την οπτικοποίηση (visualization).[10]

2.3.3 Ενισχυτική Μάθηση

Στην Ενισχυτική Μάθηση (Reinforcement Learning), το σύστημα εκπαίδευσης, που ονομάζεται πράκτορας (agent), μπορεί να παρατηρήσει το περιβάλλον, να επιλέξει και να εκτελέσει δράσεις και να λάβει αμοιβές (reward) σε ανταπόδοση. Οι αμοιβές μπορούν να είναι θετικές, είτε αρνητικές (penalty). Στη συνέχεια, ο πράκτορας πρέπει να μάθει ποια είναι η καλύτερη στρατηγική ή πολιτική (policy), ώστε να μεγιστοποιήσει την αμοιβή στον χρονικό ορίζοντα δράσης. Μια πολιτική καθορίζει ποιες πρέπει να είναι οι αποφάσεις του πράκτορα σε κάθε δεδομένη κατάσταση. Ένα παράδειγμα επιτυχημένης χρήσης ενισχυτικής μάθησης ήταν το πρόγραμμα AlphaGo, το οποίο έχει σχεδιαστεί να παίζει παρτίδες του επιτραπέζιου παιχνιδιού Go, και το 2017 κατάφερε να νικήσει τον πρώτο σε κατάταξη επαγγελματία αθλητή στον κόσμο.



Σχήμα 2.2: Οπτικοποίηση Iris dataset με χρήση τεχνικής t-SNE [11]



Σχήμα 2.3: Ενισχυτική Μάθηση [12]

2.4 Μοντέλα Μηχανικής Μάθησης

2.4.1 Γραμμική Παλινδρόμηση

Το πιο απλό μοντέλο παλινδρόμησης είναι αυτό της Γραμμικής Παλινδρόμησης (Linear Regression), όπου η έξοδος προκύπτει από έναν γραμμικό μετασχηματισμό των τιμών εισόδου (χαρακτηριστικών) με κατάλληλα βάρη (weights):

$$\hat{y} = w_0 + w_1 * x_1 + w_2 * x_2 + ... + w_N * x_N$$

Στην παραπάνω εξίσωση:

- $\hat{y}$ είναι η πρόβλεψη του μοντέλου
- N είναι η διάσταση του διανύσματος εισόδου (χαρακτηριστικών)
- w_0 το σφάλμα (bias).

2.4.2 Λογιστική Παλινδρόμηση

Η Λογιστική Παλινδρόμηση (Logistic Regression) είναι ένα μοντέλο δυαδικής ταξινόμησης το οποίο αντί να υπολογίσει απλά ένα σταθμισμένο άθροισμα των χαρακτηριστικών εισόδου, εφαρμόζει σε αυτό τη σιγμοειδή συνάρτηση:

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

όπου η έξοδός είναι ένας φυσικός αριθμός από το 0 εώς το 1. Αυτός ο αριθμός εκφράζει την πιθανότητα της θετικής κλάσης, οπότε αν η έξοδος της σιγμοειδούς συνάρτησης είναι μεγαλύτερη από 0.5, το μοντέλο προβλέπει ότι το δείγμα ανήκει στην κλάση (λογικό 1), διαφορετικά όχι (λογικό 0). Έχουμε δηλαδή:

$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0 \\ 1 & \text{if } \hat{p} \geq 0 \end{cases}$$

με $\hat{p} = \sigma(\theta^T x)$.

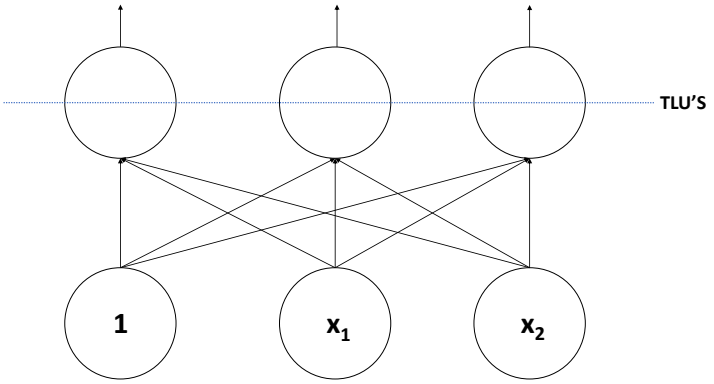
2.4.3 Μοντέλο Perceptron

Το μοντέλο Perceptron[13] εισήχθη το 1957 από τον Frank Rosenblatt και πρόκειται για δυαδικό ταξινομητή και πρόδρομο της Βαθιάς Μάθησης, στην οποία είναι αφιερωμένο το επόμενο κεφάλαιο. Η έξοδός του απλούστερου μοντέλου Perceptron αποτελεί το σταθμισμένο άθροισμα χαρακτηριστικών εισόδου με την προσθήκη ενός όρου σφάλματος (συνήθως τίθεται ίσο με 1) και με την σιγμοειδή συνάρτηση να αντικαθίσταται από τη συνάρτηση βήματος Heaviside:

$$f(x_n; \theta) = \mathbb{I}\{w^T x_n + b > 0\}.$$

Αυτό υλοποιείται από μια λογική μονάδα κατωφλίου (threshold logic unit - TLU). Παρομοίως με τη λογιστική παλινδρόμηση η έξοδος του μοντέλου Perceptron 'δείχνει' την θετική ή την αρνητική κλάση.
Στη γενική περίπτωση το μοντέλο Perceptron αποτελείται από μια συστάδα τέτοιων πυλών TLU, με κάθε μία από αυτές να είναι συνδεδεμένη με το σύνολο των χαρακτηριστικών εισόδου, έτσι ώστε να αποτελεί έναν ταξινομητή πολλαπλών ετικετών (multilabel classifier). Η εκπαίδευση του μοντέλου βασίζεται στον κανόνα του Hebb (Hebbian learning)[14], ο οποίος συνοψίζεται ως "Cells that fire together; wire together" και προέκυψε από τη μελέτη του στους βιολογικούς νευρώνες, σύμφωνα με την οποία οι νευρώνες που ενεργοποιούν συχνά ο ένας τον άλλον, αναπτύσσουν πιο ισχυρές συνδέσεις. Στην περίπτωση του μοντέλου Perceptron, ο κανόνας αυτός ερμηνεύεται ως αύξηση του βάρους των συνδέσεων που μειώνουν τα λάθη προβλέψεων. Συγκεκριμένα, το μοντέλο κάνει μία πρόβλεψη για κάθε δείγμα στο σύνολο δεδομένων εκπαίδευση. Για κάθε νευρώνα εξόδου που παρήγε λάθος πρόβλεψη, ο κανόνας ενισχύει τα βάρη των συνδέσεων από τα χαρακτηριστικά εισόδου που θα είχαν συνεισφέρει σε μια σωστή πρόβλεψη και αντίστροφα για αυτά στη λάθος με την παρακάτω στοχαστική εξίσωση:

$$w_{t+1} = w_t - \eta_t(\hat{y}_n - y_n)x_n$$



Σχήμα 2.4: Perceptron με δύο νευρώνες εισόδου και όρος σφάλματος και τρεις νευρώνες εξόδου

,όπου $\hat{y}_n$ η έξοδος του μοντέλου για το δείγμα n και η_t το βήμα εκμάθησης (learning rate) τη χρονική στιγμή t.
Αποδεικνύεται ότι αν τα δεδομένα είναι γραμμικώς διαχωρίσιμα (linearly seperable), τότε ο αλγόριθμος συγκλίνει σε μία λύση (Perceptron convergence theorem), η οποία δεν είναι όμως μοναδική και καθορίζεται από την αρχικοποίηση των βαρών και τη σειρά των δειγμάτων στη διαδικασία εκπαίδευσης.

2.5 Εκπαίδευση μοντέλων Μηχανικής Μάθησης

Όπως είδαμε και στα προηγούμενα μοντέλα, γενικός σκοπός της Μηχανικής Μάθησης είναι η εκτίμηση των παραμέτρων (parameter estimation) ή αλλιώς η 'εφαρμογή' του μοντέλου πάνω στα δεδομένα (model fitting). Αυτό απαιτεί την επίλυση ενός προβλήματος βελτιστοποίησης (optimization problem), ζητούμενο του οποίου είναι η εύρεση των τιμών των μεταβλητών $\theta \in \Theta$, που ελαχιστοποιούν μια συνάρτηση απώλειας ή κόστους $L : \Theta \to R$ [15]:

$$\theta^* \in \arg\min_{\theta \in \Theta} L(\theta)$$

2.5.1 Συνάρτηση Κόστους

Η συνάρτηση κόστους σε ένα πρόβλημα μηχανικής μάθησης ποσοτικοποιεί πόσο καλά ανταποκρίνεται το μοντέλο στην εργασία του ή πιο συγκεκριμένα είναι ένα μέτρο του πόσο καλά ευθυγραμμίζονται οι προβλέψεις του μοντέλου με τις πραγματικές τιμές (ground truth) της μεταβλητής-στόχου. Όσο χειρότερη είναι η απόδοση του μοντέλου τόσο περισσότερα ή μεγαλύτερα σφάλματα κάνει και άρα το κόστος χρήσης του αυξάνεται. Η συνάρτηση κόστους για ένα σύνολο δεδομένων είναι το άθροισμα των επιμέρους σφαλμάτων. Με την ελαχιστοποίηση της συνάρτησης κόστους, ο αλγόριθμος μηχανικής μάθησης στοχεύει στην εύρεση των βέλτιστων παραμέτρων για το μοντέλο, καθιστώντας το καλύτερο στο να κάνει ακριβείς προβλέψεις σε νέα, αθέατα δεδομένα.
Η επιλογή μιας συγκεκριμένης συνάρτησης κόστους εξαρτάται από τη φύση του προβλήματος που επιλύεται. Διαφορετικές εργασίες, όπως η παλινδρόμηση ή η ταξινόμηση, απαιτούν αναπόφευκτα διαφορετικούς τύπους συναρτήσεων κόστους:

- Μέσο τετραγωνικό σφάλμα (Mean Squared Error - MSE): Το MSE μετρά τη μέση τετραγωνική διαφορά μεταξύ των προβλεπόμενων και των πραγματικών τιμών και χρησιμοποιείται σε προβλήματα παλινδρόμησης:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Απώλεια διασταυρούμενης εντροπίας (Cross-Entropy Loss - Log Loss): Χρησιμοποιούμενη σε προβλήματα ταξινόμησης, η απώλεια διασταυρούμενης εντροπίας μετρά τη διαφορά μεταξύ των προβλεπόμενων κατανομών πιθανοτήτων και της πραγματικής κατανομής των ετικετών:

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

2.5.2 Μέθοδος Στοχαστικής Καθόδου Κλίσης

Η μέθοδος Στοχαστικής Καθόδου Κλίσης (Stochastic Gradient Descent) είναι ένας αλγόριθμος βελτιστοποίησης, στόχος του οποίου είναι να βρει το ολικό ή τοπικό ελάχιστο μιας διαφορίσιμης συνάρτησης κόστους.
Η μέθοδος αυτή είναι επαναληπτική και επαναλαμβάνεται για ένα σύνολο εποχών (epochs) για κάθε δείγμα εκπαίδευσης (συνήθως με τυχαία σειρά), όπου εποχή ονομάζουμε ένα πέρασμα κάθε δείγματος του συνόλου εκπαίδευσης από τον αλγόριθμο μάθησης. Για συνάρτηση κόστους $L = \sum_n L_n$, αν το δείγμα n έχει σφάλμα L_n, το διάνυσμα βαρών $\mathbf{w}$ ενημερώνεται σύμφωνα με τον κανόνα:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla L_n$$

όπου η το learning rate και ∇L_n η παράγωγος της συνάρτησης σφάλματος για το δείγμα n ως προς τις παραμέτρους w.
Λόγω της τυχαιότητας της διαδικασίας, οι παράγωγοι παρουσιάζουν διακυμάνσεις και η συνάρτηση κόστους αυξομοιώσεις, βαίνοντας όμως μειούμενη κατά μέσο όρο προς κάποιο ελάχιστο. Όταν τα βάρη προσεγγίσουν τις τιμές που δίνουν προσεγγιστικά την μικρότερη δυνατή συνάρτηση κόστους πάνω σε όλο το σύνολο δεδομένων, το κόστος θα συνεχίσει να αυξομοιώνεται και για αυτό το λόγο είναι σύνηθες να εισάγεται μια προσαρμοστική (adaptive) μέθοδος μείωσης του βήματος εκπαίδευσης. Έτσι, ο αλγόριθμος θα προσεγγίσει τελικά με μεγάλη πιθανότητα το ολικό ελάχιστο σε περίπτωση που η συνάρτηση κόστους είναι κυρτή (convex)[16].

2.5.3 Μέθοδος Καθόδου Κλίσης Δέσμης

Η Μέθοδος Καθόδου Κλίσης Δέσμης (Batch Gradient Descent) σε αντίθεση με τη στοχαστική παραλλαγή επεξεργάζεται όλα τα δείγματα του συνόλου εκπαίδευσης μαζί και επομένως υπολογίζεται η παράγωγος της συνολικής συνάρτησης κόστους:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla L$$

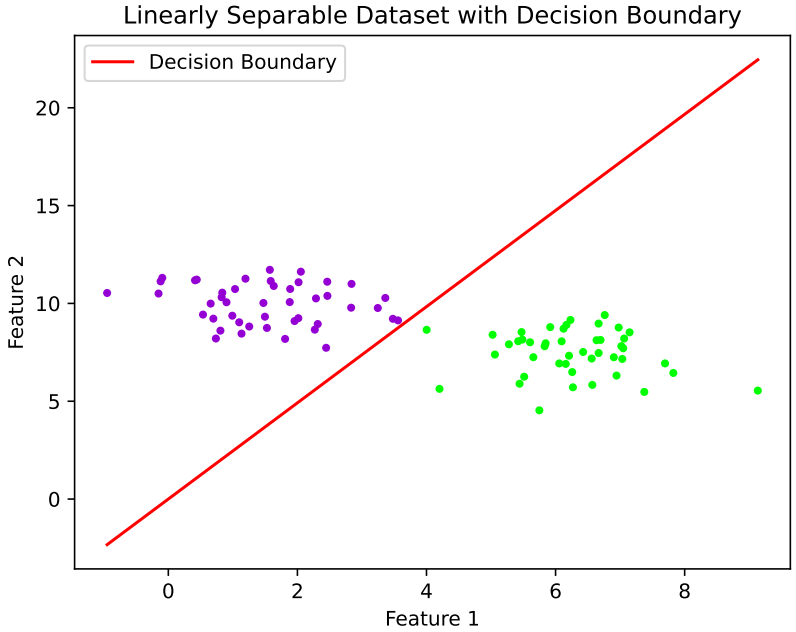
Η μέθοδος αυτή σε περίπτωση που η συνάρτηση κόστους είναι κυρτή θα προσεγγίσει σε όλες τις περιπτώσεις σταδιακά το ελάχιστο και έπειτα θα σταθεροποιηθεί, σε αντίθεση με την non-adaptive stochastic εκδοχή. Σε περίπτωση όμως που η συνάρτηση κόστους έχει τοπικά και ολικά ελάχιστα η μέθοδος Batch Gradient Descent είναι πιθανό να παγιδευτεί σε κάποιο τοπικό ελάχιστο, σε αντίθεση με την stochastic εκδοχή, η οποία μπορεί να 'ξεφύγει' εύκολα από

τέτοιες περιοχές. Ένα ακόμα μειονέκτημά της είναι ότι για μεγάλα σύνολα δεδομένων εκπαίδευσης μπορεί να έχει σημαντική μείωση ταχύτητας ή ακόμα και να μην μπορεί να εκτελεστεί από ένα υπολογιστικό σύστημα με σχετικά μικρή μνήμη.

Μια τρίτη παραλλαγή του αλγορίθμου είναι η Mini-batch εκδοχή, η οποία είναι στη μέση των δύο προηγούμενων μεθόδων νοηματικά, αφού χρησιμοποιεί για τον υπολογισμό των παραγώγων δέσμες (mini batches) από δείγματα του συνόλου. Πρόκειται για αλγόριθμο ο οποίος προσφέρει μεγάλη επιτάχυνση όταν χρησιμοποείται σε συνδυασμό με κάρτες γραφικών (GPU), οι οποίες βελτιστοποιούν τις πράξεις μεταξύ πινάκων στους υπολογισμούς παραγώγων.

2.5.4 Όριο Απόφασης

Το όριο απόφασης (Decision Boundary) πρόκειται για έννοια που συναντάται σε προβλήματα ταξινόμησης. Αντιπροσωπεύει την οριοθετική γραμμή ή επιφάνεια (σε δεδομένα με διάσταση μεγαλύτερη από 2) που διαχωρίζει ιδανικά τις διαφορετικές κλάσεις στο χώρο των χαρακτηριστικών. Το όριο απόφασης καθορίζεται από το μοντέλο κατά τη διάρκεια της φάσης εκπαίδευσης και είναι το κατώφλι στο οποίο το μοντέλο αποφασίζει να αντιστοιχίσει ένα σημείο δεδομένων στη μία ή την άλλη κλάση.
Στο σχήμα 2.5 το σύνολο δεδομένων αποτελείται από δύο κλάσεις που μπορούν να διαχωριστούν γραμμικά όπως φαίνεται. Εκπαιδεύουμε ένα μοντέλο Perceptron πάνω σε αυτά και λαμβάνουμε το όριο απόφασης, το οποίο προκύπτει από τις παραμέτρους του μοντέλου. Στη γενική περίπτωση το όριο απόφασης δεν χωρίζει τέλεια τις δύο κλάσεις, υπάρχουν δηλαδή misclassifications στις διάφορες περιοχές που ορίζει.



Σχήμα 2.5: Όριο απόφασης σε γραμμικώς διαχωρίσιμα δεδομένα

Κεφάλαιο 3

Βαθιά Μάθηση

3.1 Κίνητρο ανάπτυξης Βαθιάς Μάθησης

Ένας από τους περιορισμούς των κλασικών μοντέλων μηχανικής μάθησης, όπως της γραμμικής (linear) ή λογιστικής (logistic) παλινδρόμησης, είναι ότι πολλές φορές δεν μπορούν να μάθουν ουσιαστικές σχέσεις στο σύνολο δεδομένων ή να διαχωρίσουν μη γραμμικώς διαχωρίσιμες κλάσεις (πρόβλημα XOR), λόγω του ότι περιορίζονται από την υπόθεση ότι τα δεδομένα εισόδου σχετίζονται με γραμμικές (γενικευμένες) σχέσεις με τα δεδομένα εξόδου.
Ένας τρόπος παράκαμψης αυτού του περιορισμού είναι η εισαγωγή μη-γραμμικοτήτων με αναδρομικό τρόπο, κάτι που αποτελεί την βασική ιδέα πίσω από τα βαθιά νευρωνικά δίκτυα.

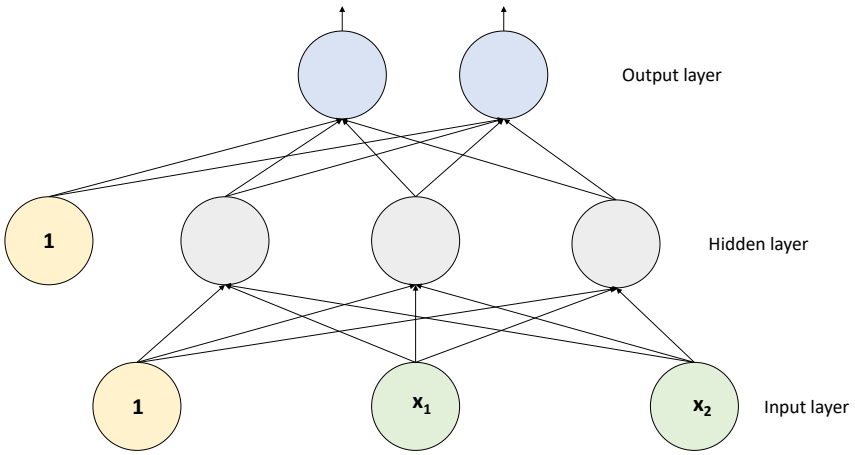
3.2 Τεχνητά Νευρωνικά Δίκτυα

Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks - ANN) ονομάζονται τα μοντέλα που χρησιμοποιούνται στη Μηχανική Μάθηση και είναι εμπνευσμένα από τα δίκτυα βιολογικών νευρώνων που βρίσκονται στον εγκέφαλό μας. Ένα ANN αποτελείται από κάποιες συνδεδεμένες μονάδες όμοια με κόμβους σε γράφους, που ονομάζονται τεχνητοί νευρώνες και που οι ακμές του έχουν βάρη που αντιστοιχούν σε πραγματικούς αριθμούς και μοντελοποιούν τις συνάψεις του εγκεφάλου. Τα βάρη των ακμών μπορούν να προσαρμόζονται κατά τη διάρκεια μιας διαδικασίας εκμάθησης, όπου οι νευρώνες τροφοδοτούν με σήματα (πραγματικούς αριθμούς) τους νευρώνες με τους οποιούς έχουν κατευθυνόμενες συνδέσεις.

3.3 Μοντέλα Perceptron πολλών επιπέδων

Ένας διαισθητικός τρόπος για να αυξήσουμε την εκφραστικότητα των μοντέλων μας και να λύσουμε προβλήματα όπως το Exclusive OR είναι να στοιβάξουμε πολλούς Perceptron το έναν πάνω στον άλλο. Αυτή η δομή ονομάζεται Perceptron πολλών επιπέδων (multilayer perceptron - MLP). Ένας MLP αποτελείται από το στρώμα εισόδου (input layer), ένα ή περισσότερα στρώματα από TLUs, που ονομάζονται κρυφά στρώματα (hidden layers) επειδή οι τιμές τους δεν παρατηρούνται και ένα τελευταίο στρώμα από TLUs, που ονομάζεται output layer. Συμβατικά, ένα νευρωνικό δίκτυο ονομάζεται βαθύ νευρωνικό δίκτυο και η μελέτη του βαθιά μάθηση όταν έχει τουλάχιστον δύο στρώματα κρυφών στρωμάτων. Λόγω της μεγάλης τους πολυπλοκότητας και τη δυσκολία εκμάθησης, η μελέτη τους γνώρισε ένα τέλμα, μέχρι την έκδοση της

καθοριστικής σημασίας για την μετέπειτα επανάσταση της Μηχανικής Μάθησης μελέτη των Rumelhart, Hinton, Williams[17] για την οπισθοδιάδοση.



Σχήμα 3.1: Perceptron πολλών επιπέδων

3.4 Οπισθοδιάδοση

Ο αλγόριθμος της οπισθοδιάδοσης (Backpropagation) υπολογίζει τη μερική παράγωγο της συνάρτησης κόστους της εξόδου (ή των εξόδων) ως προς κάθε παράμετρο του μοντέλου, με δύο περάσματα: ένα προς τα εμπρός (forward) και ένα προς τα πίσω (backward).

- Στο (forward pass) έχουμε τον σταδιακό υπολογισμό των εξόδων όλων των νευρώνων ξεκινώντας από τα input layers και μεταδίδοντας την πληροφορία μέχρι τα output layers. Στη συνέχεια υπολογίζεται το σφάλμα του δικτύου με τη συνάρτηση κόστους.
- Στο (backward pass) υπολογίζεται ουσιαστικά η συνεισφορά της κάθε σύνδεσης στο τελικό λάθος. Αυτό γίνεται εφαρμόζοντας τον κανόνα της αλυσίδας (chain rule) μετρώντας την παράγωγο του σφάλματος σε όλα τα βάρη των συνδέσεων στο δίκτυο, διαδίδοντας την προς τα πίσω μέσω του δικτύου.
- Όταν οι παράγωγοι έχουν συσσωρευτεί σε όλες τις ακμές ο αλγόριθμος βελτιστοποίησης (π.χ. Gradient Descent) ενημερώνει τα βάρη των ακμών χρησιμοποιώντας τις στον υπολογισμό.

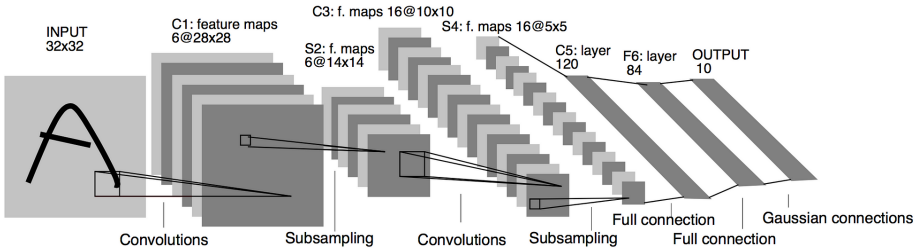
Μια σημαντική σημείωση είναι ότι επειδή η αρχική συνάρτηση βήματος στην κλασική MLP αρχιτεκτονική περιέχει μόνο επίπεδα τμήματα οι συγγραφείς την αντικατέστησαν με τη σιγμοειδή συνάρτηση:

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

Η συνάρτηση ενεργοποίησης είναι απαραίτητη για την αποτελεσματικότητα των μοντέλων, καθώς αν δεν ήταν μέρος του υπολογιστικού γράφου η αλυσίδα των γραμμικών μετασχηματισμών θα κατέληγε τελικά σε έναν απλό γραμμικό μετασηματισμό ενός απλού στρώματος.

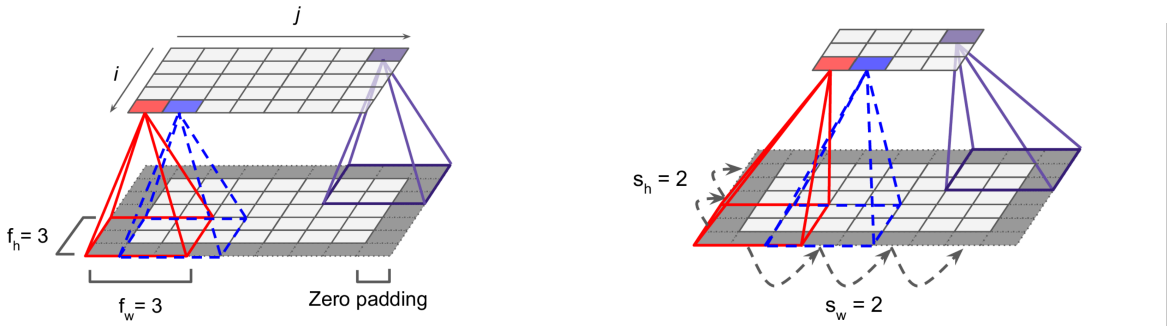
3.5 Συνελικτικά Νευρωνικά Δίκτυα

Τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks -CNNs) εμπνεύστηκαν από τον τρόπο λειτουργίας του οπτικού φλοιού (visual cortex) του ματιού και βρήκαν μεγάλη επιτυχία σε εφαρμογές Όρασης Υπολογιστών (Machine Vision). Οι έρευνες στο φλοιό του ματιού σε γάτες[18] ανέδειξαν την ύπαρξη τοπικών δεκτικών πεδίων σε συγκεκριμένες περιοχές του οπτικού πεδίου που με κατάλληλη επικάλυψη καλύπτουν όλο το πεδίο και μπορούν ακόμα να συνδυαστούν ιεραρχικά για την αναγνώριση πιο περίπλοκων προτύπων. Οι μελέτες αυτές ενέπνευσαν το μοντέλο του neocognitron[19] το οποίο εξελίχθηκε σταδιακά στη σημερινή μορφή των συνελικτικών νευρωνικών δικτύων.
Το μοντέλο LeNet-5 εισήχθη το 1998 με σκοπό την αναγνώριση χειρόγραφων αριθμών και ήταν ουσιαστικά το πρώτο μοντέρνο δίκτυο CNN που εισήγαγε συνελικτικά στρώματα και στρώματα συγκέντρωσης (pooling). Ένα συνελικτικό στρώμα σε δύο διαστάσεις (εφαρμογές όρασης όπου



Σχήμα 3.2: Αρχιτεκτονική μοντέλου LeNet-5

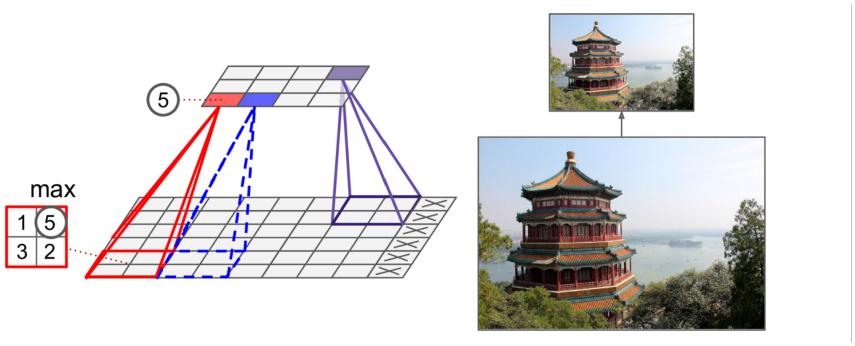
οι εικόνες αποτελούνται από ένα παραλληλόγραμο εικονοστοιχείων) αποτελείται από τη συνέλιξη παραλληλόγραμμων φίλτρων (πινάκων βαρών) τα οποία περνούν σταδιακά (συγκεκριμένο βήμα) πάνω από όλο το πεδίο κάλυψής τους (για ένα πρώτο στρώμα αυτό θα ήταν όλη η εικόνα). Τα συνελικτικά στρώματα μπορούν ακόμα να στοιβαχτούν έτσι ώστε τα φίλτρα για παράδειγμα του δεύτερου στρώματος να εφαρμόζονται στο πρώτο στρώμα που προκύπτει από τα αντίστοιχα φίλτρα. Για να αυξήσουμε το μέγεθος ενός στρώματος μπορούμε να γεμίσουμε περιγραμματικά το προηγούμενο στρώμα με επιπλέον στοιχεία, οι έξοδοι των οποίων τίθενται συνήθως ίσες με μηδέν (zero pudding). Μια τεχνική για να μειώσουμε το στρώμα που παράγεται από ένα δεδομένο φίλτρο είναι να αυξήσουμε το βήμα (stride) με το οποίο το περνάμε πάνω από το στρώμα εισόδου του.



(α′) Zero Padding (β′) Stride

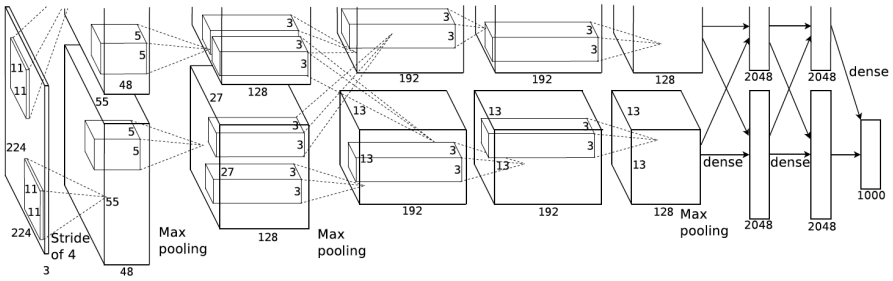
Σχήμα 3.3: (α′) Εφαρμόζουμε ένα φίλτρο διάστασης 3×3 σε στρώμα εισόδου διάστασης 5×7 και έχοντας zero pudding λαμβάνουμε στρώμα εξόδου 5×7 (α′) Με stride ίσο με 2 η διάσταση της εξόδου είναι τώρα 3×4 [20]

Τα στρώματα pooling χρησιμεύουν για τη μείωση της διάστασης του στρώματος εξόδου και εφαρμόζονται με τον ίδιο τρόπο πάνω στα στρώματα εισόδου όμοια με τα συνελικτικά δίκτυα με τη διαφορά ότι δεν έχουν βάρη, αλλά υλοποιούν κάποιους operators που αποτελούν συναρτήσης 'συγκέντρωσης' (aggregation function - aggregators), όπως μέγιστο, ελάχιστο, ή μέσο όρο. Τα συνελικτικά νευρωνικά δίκτυα συνήθως αναπτύσσουν μερικά συνελικτικά στρώματα, στη



Σχήμα 3.4: Φίλτρο max pooling διάστασης 2×2 και stride ίσο με 2 που μειώνει το στρώμα εισόδου 6×8 σε 3×4[20]

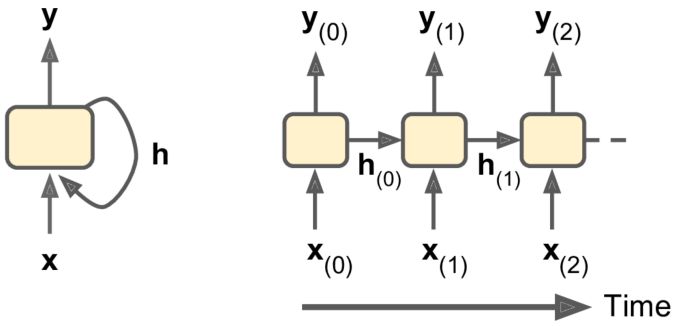
συνέχεια ένα pooling στρώμα κ.ο.κ. με τις μοντέρνες αρχιτεκτονικές να φτάνουν σε πάρα πολύ μεγάλο βάθος. Μερικά γνωστά μοντέλα και ο αριθμός των στρωμάτων τους είναι: AlexNet (8 στρώματα). VGG16 και VGG19[21] (16 και 19 στρώματα αντίστοιχα), GoogLeNet[22] (22 στρώματα), ResNet-50[23] (50 στρώματα), κ.τ.λ.



Σχήμα 3.5: Αρχιτεκτονική AlexNet[24]

3.6 Αναδρομικά Νευρωνικά Δίκτυα

Τα Αναδρομικά Νευρωνικά Δίκτυα (Recurrent neural networks - RNNs) είναι νευρωνικά δίκτυα ικανά να χειρίζονται ακολουθίες δεδομένων. Στις εφαρμογές τους περιλαμβάνονται η ταξινόμηση ακολουθιών, η παραγωγή ή μετάφραση ακολουθιών κ.α.
Σε αντίθεση με όλα τα μοντέλα που είδαμε μέχρι στιγμής στα οποία οι ενεργοποιήσεις των νευρώνων ρέουν προς μία κατεύθυνση, τα RNNs έχουν συνδέσεις που δείχνουν και προς τα πίσω. Έτσι, η έξοδος y_t δεν εξαρτάται μόνο από την είσοδο x_t, αλλά και από μια κρυφή (hidden) κατάσταση του συστήματος h_t, η οποία ενημερώνεται από την είσοδο και με βάση την προηγούμενη κατάσταση, δηλαδή $h_t = f(h_{t-1}, x_t)$. Στην εικόνα 3.6 βλέπουμε την αρχιτεκτονική αυτού του 'κυττάρου μνήμης' που είναι η κρυφή κατάσταση, ξεδιπλωμένο στο χρόνο, ώστε να φανεί



Σχήμα 3.6: Αρχιτεκτονική RNN[20]

καλύτερα η παραγωγή και χρονική εξάρτηση των μεταβλητών του συστήματος.
Ανάλογα με την εργασία για την οποία χρησιμοποιούμε το μοντέλο RNN μπορούμε να λάβουμε υπόψη μας όλες τις χρονικές εξόδους (sequence-to-sequence), για παράδειγμα για την πρόβλεψη των τιμών θερμοκρασίας των επόμενων ημερών, μόνο την τελευταία έξοδο (sequence-to-vector) για εντοπισμό ύποπτων συναλλαγών σε ένα ιστορικό συναλλαγών ή σε περίπτωση που η είσοδος είναι σταθερή και παρακολουθείται η έξοδος (vector-to-sequence) για την παραγωγή μιας περιγραφικής λεζάντας για μια φωτογραφία.
Η εκπαίδευση του μοντέλου RNN γίνεται ξετυλίγοντας τις καταστάσεις του στο χρόνο (παρόμοια με σχήμα 3.6) και χρησιμοποιώντας κανονικά backpropagation (backpropagation through time - BPTT).
Τα δίκτυα RNN πάσχουν από αρκετά προβήματα, όπως αυτό της βραχείας μνήμης όπου μετά από κάποιες χρονικές στιγμές η κρυφή κατάσταση δεν περιέχει κάποια σημαντική συνεισφορά από τις πρώτες εισόδους ή της εξαφανιζόμενης κλίσης (vanishing gradient problem), όπου οι παράγωγοι των παραμέτρων μειώνονται εκθετικά με την πάροδο του χρόνου και καθιστούν δύσκολη την αποτελεσματική ενημέρωση των βαρών από το δίκτυο για μακρινά χρονικά βήματα. Για να αντιμετωπιστούν αυτά τα προβλήματα νέα μοντέλα έχουν εισηχθή, όπως τα Δίκτυα Μακράς Βραχύχρονης Μνήμης (Long Short-term Memory - LSTM)[25] τα οποία έχουν αντικαταστήσει πια τα απλά RNN.

Κεφάλαιο 4

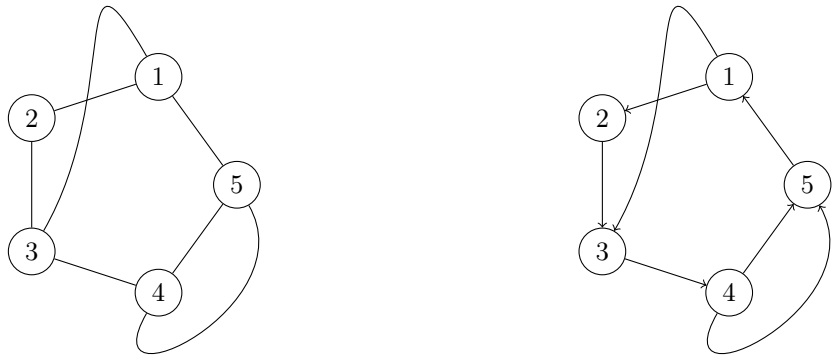
Γράφοι

4.1 Ορισμοί

Οι γράφοι ή και γραφήματα αποτελούν αναπαραστάσεις μεταξύ μιας συλλογής οντοτήτων. Χρησιμοποιούνται για τη μαθηματική μοντελοποίηση πολλών προβλημάτων π.χ. δικτύων δρομολόγησης και ανεφοδιασμού, κοινωνικών δικτύων, βιολογικών δικτύων κ.α.

Ορισμός (Γράφημα)**.** *Ένα (στατικό) γράφημα είναι ένα ζεύγος* $\boldsymbol{G = (V, E)}$*, όπου* $\boldsymbol{V}$ *είναι ένα σύνολο του οποίου τα στοιχεία ονομάζονται κορυφές ή κόμβοι (vertices), και* $\boldsymbol{E}$ *είναι ένα σύνολο ζευγαρωμένων κορυφών, του οποίου τα στοιχεία ονομάζονται ακμές (edges).*

Ανάλογα με το αν οι ακμές έχουν προσανατολισμό, δηλαδή το σύνολο Ε αποτελείται από διατεταγμένα ζεύγη κορυφών ή όχι το γράφημα ονομάζεται κατευθυνόμενο (directed) ή μη-κατευθυνόμενο (undirected). Ακόμα είναι δυνατόν να οριστούν και γραφήματα με βάρη στις ακμές τους (weighted graph) ώστε να οριστούν ποιοτικές αντί για διακριτές σχέσεις μεταξύ των κορυφών του γραφήματος.



(α′) Μη-κατευθυνόμενο γράφημα (β′) Κατευθυνόμενο γράφημα

Σχήμα 4.1: Μη-κατευθυνόμενο και κατευθυνόμενο γράφημα

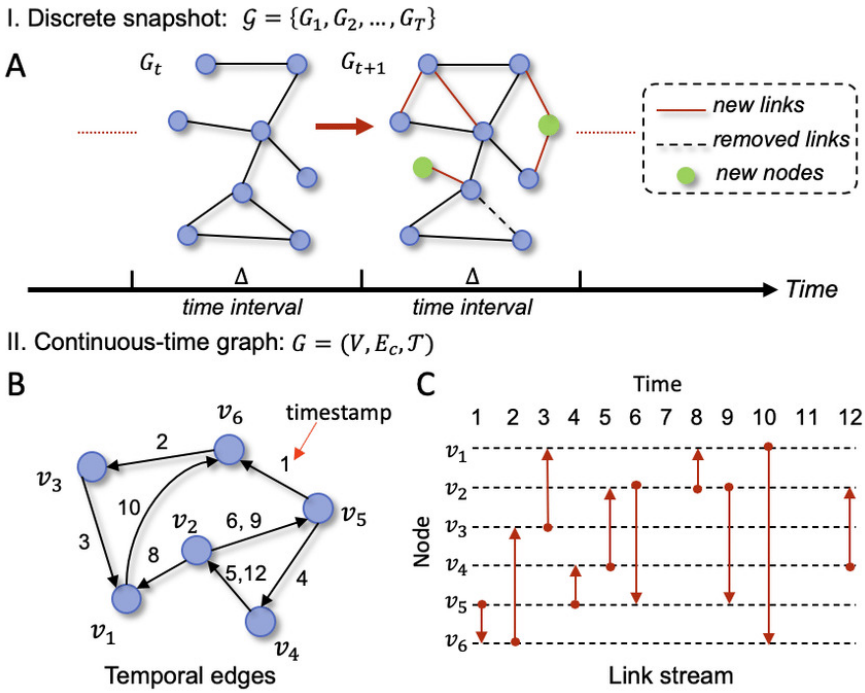
Ορισμός (Υπογράφημα)**.** *Το υπογράφημα (subgraph) είναι ένα γράφημα το οποίο περιέχει ένα υποσύνολο κορυφών και ακμών του γραφήματος βάση του οποίου ορίζεται. Το υποσύνολο των*

κορυφών πρέπει να περιλαμβάνει όλες τις κορυφές του υποσυνόλου ακμών, αλλά μπορεί επίσης να περιλαμβάνει και επιπλέον κορυφές.

Ορισμός (Δυναμικός γράφος)**.** *Ως δυναμικό γράφο ορίζουμε μια ακολουθία*

$$\Gamma := (G_1, G_2, \ldots, G_n)$$

όπου $G_i := (V_i, E_i)$ *είναι στατικοί γράφοι και οι δείκτες αναφέρονται σε μια ακολουθία από χρονικά βήματα* $\tau := (t_1, t_2, \ldots, t_n)$*. Ο ορισμός αυτός του δυναμικού γράφου ως μια ακολουθία διακριτών στιγμιοτύπων (snapshots) είναι ο επικρατών και αυτός που ακολουθούμε στην λογική αυτής της εργασίας. Εναλλακτικά, όμως, μπορούμε να ορίσουμε έναν δυναμικό γράφο και με μια πιο fine-grained λογική, όπου κάθε διαγραφή ή προσθήκη ακμής θεωρείται ένα ξεχωριστό γεγονός με χρονική σφραγίδα. Στο σχήμα 4.2 βλέπουμε τις δύο αυτές διαφορετικές προσεγγίσεις.*



Σχήμα 4.2: Διακριτοί και συνεχείς δυναμικοί γράφοι[26]

Ορισμός (Μονοπάτι)**.** *Μονοπάτι (path) ενός γραφήματος ονομάζουμε μια ακολουθία κόμβων με την ιδιότητα ότι κάθε διαδοχικό ζεύγος στην ακολουθία είναι γειτονικά, δηλαδή συνδέονται με μια ακμή.*

Μονοπάτι μεταξύ δύο κόμβων α και β είναι κάθε ακολουθία, όπου το πρώτο στοιχείο της είναι ο κόμβος α και τελευταίο στοιχείο της ο κόμβος β.
Στη γενική περίπτωση, ένα μονοπάτι μπορεί να παρουσιάζει επανάληψη κόμβων. Στο εξής θεωρούμε μονοπάτια που δεν επαναλαμβάνουν κόμβους, τα οποία αναφέρονται και ως απλά (simple) μονοπάτια.

4.2 Εγγύτητα κορυφών

4.2.1 Εγγύτητα πρώτης τάξης

Κόμβοι που συνδέονται με μια ακμή έχουν εγγύτητα πρώτης τάξης. Τα βάρη των ακμών αντιπροσωπεύουν μέτρα εγγύτητας, έτσι ώστε το μεγαλύτερο βάρος να υποδηλώνει μεγαλύτερη ομοιότητα μεταξύ των δύο συνδεδεμένων κόμβων.

4.2.2 Εγγύτητα δεύτερης τάξης

Η εγγύτητα δεύτερης τάξης αντιπροσωπεύει τις σχέσεις 2 βημάτων μεταξύ κάθε ζεύγους κορυφών και ορίζεται ως ο αριθμός των κοινών γειτόνων που μοιράζονται οι δύο κορυφές.

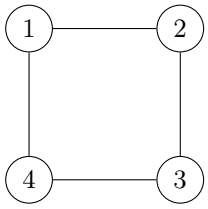
4.2.3 Εγγύτητα ανώτερης τάξης

Η γειτνίαση ανώτερης τάξης μεταξύ δύο κορυφών v και u μπορεί να οριστεί ως η πιθανότητα μετάβασης από την κορυφή v στην κορυφή u με βήμα k ($k \geq 3$).

4.3 Αναπαράσταση γραφημάτων

4.3.1 Πίνακας Γειτνίασης

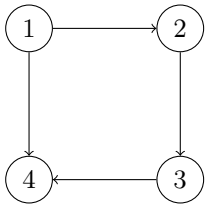
Ο πίνακας γειτνίασης (Adjacency Matrix) είναι μια μέθοδος αναπαράστασης ενός γραφήματος με έναν πίνακα τιμών αληθείας: το λογικό 1 και 0 (boolean values).
Για μη-κατευθυνόμενο γράφημα, έστω πίνακας γειτνίασης A και η τιμή του A_{ij}. Αν $A_{ij} = 1$ οι κορυφές i και j συνδέονται με ακμή, ενώ αν $A_{ij} = 0$ όχι. Προκύπτει ότι ο πίνακας γειτνίασης μη-κατευθυνόμενου γράφου είναι συμμετρικός. Αντίθετα, για κατευθυνόμενο γράφημα η αληθοτιμή 1 δηλώνει την ύπαρξη ακμής με προσανατολισμό από την κορυφή i προς την κορυφή j και ο πίνακας μπορεί να είναι ασυμμετρικός.
Τέλος, αν έχουμε να κάνουμε με γράφημα με βάρη οι τιμές του πίνακα είναι ίσες με το βάρος των ακμών $A_{ij} = w(v_i, v_j)$.



(α′) Μη-κατευθυνόμενο γράφημα

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

(β′) Πίνακας Γειτνίασης

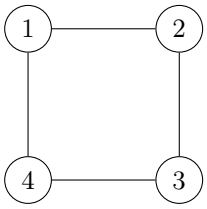


(γ′) Κατευθυνόμενο γράφημα

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

(δ′) Πίνακας Γειτνίασης

Σχήμα 4.3: Γραφήματα και πίνακες γειτνίασης



(α′) Γράφημα

Κόμβος	Γειτονιά
1	2, 4
2	1, 3
3	2, 4
4	1, 3

(β′) Λίστα γειτνίασης

Σχήμα 4.4: Γράφημα και λίστα γειτνίασης

4.3.2 Λίστα Γειτνίασης

Η λίστα γειτνίασης (Adjacency List) είναι μια συλλογή μη ταξινομημένων λιστών που χρησιμοποιείται για την αναπαράσταση ενός πεπερασμένου γράφου. Κάθε μη ταξινομημένη λίστα μέσα σε μια λίστα γειτνίασης περιέχει το σύνολο των γειτόνων μιας συγκεκριμένης κορυφής στο γράφημα.
Ένα σημαντικό πλεονέκτημα των λιστών γειτνίασης έναντι των πινάκων γειτνίασης, είναι ότι όταν ένα γράφημα είναι αραιό (sparse), δηλαδή $|E| \sim |V|$, ο χώρος που απαιτείται για την αναπαράστασή του με λίστα γειτνίασης είναι πολύ μικρότερος από ότι με πίνακα γειτνίασης, αφού στη δεύτερη περίπτωση τα περισσότερα στοιχεία του πίνακα θα ήταν 0. Για πυκνά (dense) γραφήματα, όπου ισχύει $|E| \sim |V|^2$ προτιμούμε αντίστοιχα την αναπαράσταση/αποθήκευση του πίνακα με πίνακα γειτνίασης.

4.3.3 Λαπλασιανός Πίνακας

Για να ορίσουμε τον λαπλασιανό (Laplacian) πίνακα ενός γραφήματος ορίζουμε αρχικά τον πίνακα βαθμού (Degree matrix) D, ο οποίος είναι ένας διαγώνιος πίνακας, για τον οποίο ισχύει $D_{i,i} = deg(i)$, δηλαδή κάθε διαγώνιο στοιχείο ισούται με τον αντίστοιχο βαθμό κόμβου.
Τελικά ο (Laplacian) ενός γραφήματος ορίζεται ως:

$$L = D - A$$

Ο συμμετρικός, normalized Laplacian ορίζεται ως:

$$\tilde{L} = I - D^{-1/2} W D^{-1/2}$$

4.4 Συνεκτικές Συνιστώσες

Ορισμός (Συνεκτικό γράφημα)**.** *Ένα γράφημα ονομάζεται συνεκτικό (connected) εάν υπάρχει μονοπάτι μεταξύ κάθε ζευγαριού κορυφών του.*

Ορισμός (Συνιστώσα)**.** *Συνιστώσα (component) ενός μη κατευθυνόμενου γραφήματος είναι ένα συνεκτικό υπογράφημα που δεν αποτελεί μέρος οποιουδήποτε μεγαλύτερου συνεκτικού υπογραφήματος.*

4.5 Εξερεύνηση Γραφημάτων

Στη συνέχεια παρουσιάζουμε μερικούς αλγόριθμους εξερεύνησης γραφημάτων με πληθώρα εφαρμογών, μεταξύ των οποίων και τεχνικές παραγωγής διανυσματικών αναπαραστάσεων κόμ-

βων, οι οποίες αποτελούν σημαντικό κομμάτι αυτής της εργασίας και θα αναλυθούν περαιτέρω στα επόμενα κεφάλαια.

4.5.1 Αναζήτηση κατά βάθος

Ο αλγόριθμος Αναζήτησης Κατά Βάθος (DFS - Depth-first search) εφαρμόζει διάσχιση/αναζήτηση σε ένα γράφημα. Η διάσχιση ξεκινά από έναν αρχικό κόμβο (source node) και απομακρύνεται σταδιακά από αυτόν εξερευνόντας αναδρομικά όλους τους γείτονές (adjacents) του που δεν έχει ήδη επισκεφτεί (not visited). Στη συνέχεια παραθέτουμε ψευδοκώδικα του αλγορίθμου:

Algorithm 1: Depth-First Search (DFS)

```
1  Function DFS(G, v):
       Input: Graph G and source node s
2      Mark v as visited;
3      for each adjacent vertex u of v in G do
4          if u is not visited then
5              DFS(G, u);
6          end
7      end
```

4.5.2 Αναζήτηση κατά πλάτος

Ο αλγόριθμος Αναζήτησης Κατά Πλάτος (BFS - Breadth-first search) εξελίσσεται σε φάσεις. Αρχικά, ο αλγόριθμος διασχίζει τους γειτονικούς κόμβους του αρχικού κόμβου, έπειτα στους γείτονες του επόμενου επιπέδου κ.ο.κ.
Για την υλοποίησή του χρησιμοποιούμε μια ουρά (queue), όπου το πρώτο στοιχείο που εισέρχεται σε αυτήν είναι και το πρώτο που εξέρχεται (First In, First Out - FIFO).

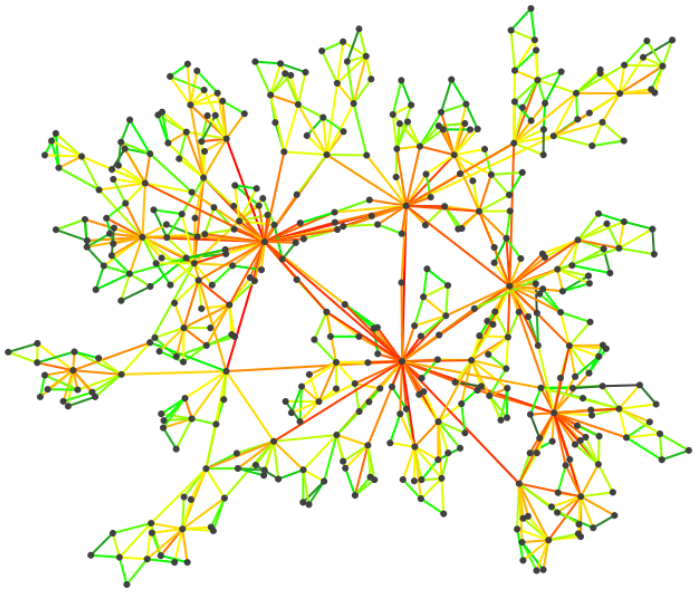
Algorithm 2: Breadth-First Search (BFS)

```
1  Function BFS(G, s):
       Input: Graph G and source node s
2      Q ← queue;
3      Q.enqueue(s) ;                          // Inserting s in queue
4      mark s as visited;
5      while Q is not empty do
6          v ← Q.dequeue();
7          for each neighbor w of v in Graph G do
8              if w is not visited then
9                  Q.enqueue(w) mark w as visited;
10             end
11         end
12     end
```

4.6 Τυχαίος Περίπατος

Ένας τυχαίος περίπατος μήκους k σε ένα γράφημα G με ρίζα v_0 είναι μια στοχαστική διαδικασία με τυχαίες μεταβλητές $X_1, X_2, ..., X_k$, έτσι ώστε $X_1 = v_0$ και κάθε ζεύγος κόμβων στον

περίπατο αυτό συνδέονται με ακμή. Σε αντίθεση με τους αλγορίθμους αναζήτησης, ο επόμενος κόμβος επιλέγεται από μια πιθανοτική κατανομή, συνήθως ομοιόμορφη και δεν υπάρχει κανένας περιορισμός στον περίπατο που να τον εμποδίζει από το να γυρίσει πίσω ή να επισκεφθεί τον ίδιο κόμβο πολλές φορές. Συγκεκριμένα, όπως φαίνεται και στο σχήμα 4.5 οι κόμβοι με περισσότερες ακμές είναι πιο πιθανό να 'επιλεχθούν' ως επόμενοι κόμβοι της αλληλουχίας και έτσι είναι λογικό οι επαγώμενες σε αυτούς ακμές να έχουν διασχιστεί πολλές φορές (κόκκινο χρώμα) σε αντίθεση με απομακρυσμένους κόμβους που οι επαγόμενες ακμές του έχουν διασχιστεί εκθετικά λιγότερες φορές (πράσινο χρώμα). Τέλος, οι τυχαίοι περίπατοι βρίσκουν χρήσης σε πολυάριθμες εφαρμογές, όπως για παράδειγμα την κατατάξη κόμβων Pagerank, ή στη δημιουργία διανυσματικών αναπαραστάσεων κόμβων που θα μελετήσουμε στα επόμενα κεφάλαια.



Σχήμα 4.5: Τυχαίος περίπατος

4.7 Μετρικές Ανάλυσης

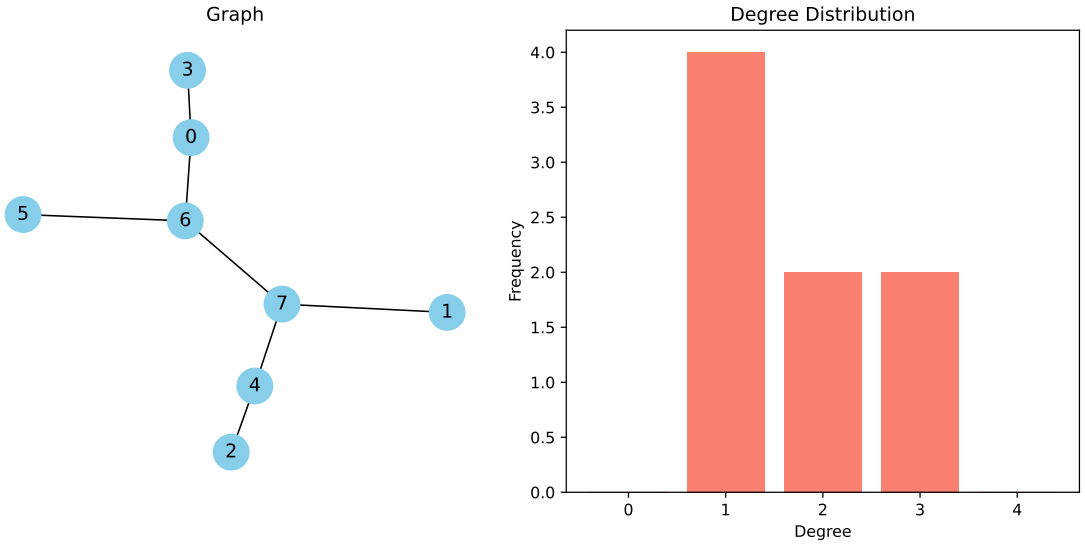
Στη συνέχεια παραθέτουμε μερικές χρήσιμες μετρικές ανάλυσης γραφημάτων, οι οποίες μπορούν να μας δώσουν μία πρώτη και γενική εικόνα του γραφήματος που επιθυμούμε να αναλύσουμε.

4.7.1 Κατανομή βαθμού

Ορισμός (Βαθμός κόμβου)**.** *Βαθμός (degree) ενός κόμβου σε ένα μη-κατευθυνόμενο γράφημα είναι ο αριθμός των ακμών που έχουν ως άκρο τον κόμβο αυτό.*

Για κατευθυνόμενα γραφήματα ορίζονται αντίστοιχα ο εσωτερικός βαθμός (in-degree) και ο εξωτερικός βαθμός κόμβου (out-degree), δηλαδή ο αριθμός των ακμών με προσανατολισμό προς τον κόμβο και από τον κόμβο αντίστοιχα. Για το υπόλοιπο του κεφαλαίου, αλλά και της μελέτης θα ασχοληθούμε με μη-κατευθυνόμενα γραφήματα.

Ορισμός (Κατανομή Βαθμού κόμβου)**.** *Κατανομή βαθμού κόμβου είναι η κατανομή πιθανότητας βαθμού κόμβου πάνω σε όλο το γράφημα. Με δεδομένο γράφημα η κατανομή αυτή προκύπτει στατιστικά.*



Σχήμα 4.6: Κατανομή Βαθμού κόμβου

4.7.2 Συντελεστής συσταδοποίησης

Ο συντελεστής συσταδοποίησης (Clustering Coefficient) είναι ένα μέτρο του βαθμού στον οποίο οι κόμβοι ενός γραφήματος τείνουν να συγκεντρώνονται ή να σχηματίζουν στενά συνδεδεμένες ομάδες.

Ορισμός (Τριπλέτα)**.** *Μια τριπλέτα είναι τρεις κόμβοι που συνδέονται είτε με δύο (ανοικτή τριπλέτα) είτε με τρεις (κλειστή τριπλέτα) μη κατευθυνόμενους δεσμούς (ακμές).*

Ένα τριγωνικό γράφημα περιλαμβάνει επομένως τρεις κλειστές τριπλέτες, μία με κέντρο κάθε κόμβο του γραφήματος, δηλαδή οι τρεις τριπλέτες σε ένα τρίγωνο προέρχονται από επικαλυπτόμενες επιλογές κόμβων.

Ο ολικός (global) συντελεστής συσταδοποίησης C ορίζεται ως εξής:

$$C = \frac{\text{Αριθμός κλειστών τριπλέτων}}{\text{Συνολικός αριθμός τριπλέτων}}$$

4.7.3 Διάμετρος

Ορισμός (Απόσταση κόμβων)**.** *Απόσταση (distance) δύο κόμβων είναι ο αριθμός των ακμών στο μικρότερο μονοπάτι που συνδέει τους δύο κόμβους.*

Ορισμός (Διάμετρος)**.** *Διάμετρος (diameter) ενός γραφήματος είναι η μεγαλύτερη απόσταση μεταξύ όλων των ζευγαριών κόμβων του γραφήματος.*

Κεφάλαιο 5

Βαθιά Μάθηση σε Γράφους

Σε αντίθεση με άλλα είδη δεδομένων στη Μηχανική Μάθηση, οι γράφοι παρουσιάζουν μοναδικές προκλήσεις που μας υποχρεώνουν να επεκτείνουμε και να καταλήξουμε πολλές φορές σε εξ ολοκλήρου νέες τεχνικές που απαντούν στα μοναδικά αυτά προβλήματα. Για παράδειγμα, αν αναλογιστούμε το πρόβλημα της αναγνώρισης οντοτήτων σε εικόνες στον τομέα του Machine Vision, μια εικόνα αποτελείται από ένα πλέγμα εικονοστοιχείων τα οποία μπορούν όμως να αναπαρασταθούν ως κόμβοι με τις ακμές να αποτελούν σχέσεις γειτονίας στην εικόνα. Κάθε εικόνα έχει ακριβώς την ίδια δομή γράφου με αποτέλεσμα οι συναρτήσεις που προκύπτουν από την εκμάθηση των βαθιών νευρωνικών δικτύων να μπορούν να γενικευτούν σε νέες εικόνες. Εν αντιθέσει, οι γράφοι γενικά έχουν διαφορετικές συνδέσεις και κάθε κόμβος διαφορετική δομή γειτονιάς. Αυτή ακριβώς η υποκείμενη μη-ευκλείδεια δομή ή δομή που δεν προσομοιάζει σε πλέγμα οδήγησε και στην ανάπτυξη τεχνικών Geometric Learning[27] που σκοπό έχουν να γενικεύσουν ήδη υπάρχουσες τεχνικές από τον τομέα της Βαθιάς Μάθησης.

5.1 Αναπαραστάσεις Γράφων

Το πρόβλημα της αναπαράστασης γράφων (Network-Graph embedding) που αναφέρεται και με τον όρο Graph Representation Learning συνίσταται στην εκμάθηση μιας συνάρτησης από ένα διακριτό γράφημα σε ένα συνεχές πεδίο. Τυπικά, δεδομένου ενός γράφου $G = (V, E)$, με σταθμισμένο πίνακα γειτνίασης $W \in R^{|V|\times|V|}$, ο στόχος είναι να μάθουμε χαμηλής διάστασης διανυσματικές αναπαραστάσεις $\{Zi\}_{i\in V}$ για τους κόμβους του γραφήματος $\{v_i\}_{i\in V}$, έτσι ώστε οι σημαντικότερες ιδιότητες του γράφου (π.χ. τοπική ή ολική δομή) να διατηρούνται στο χώρο αναπαράστασης. Για παράδειγμα, εάν δύο κόμβοι έχουν παρόμοιες συνδέσεις στον αρχικό γράφο ή λαμβάνουν ίδιους ρόλους σε τοπικές περιοχές του γράφου, οι διανυσματικές αναπαραστάσεις τους που μαθαίνονται θα πρέπει να είναι αντίστοιχα παρόμοιες. Έστω $Z \in R^{|V|\times d}$ συμβολίζει τη μήτρα αναπαράστασης κόμβου (node embedding matrix). Στην πράξη, συχνά επιθυμούμε χαμηλής διάστασης αναπαραστάσεις ($d << |V|$) για λόγους κλιμάκωσης. Δηλαδή, τα node embeddings μπορούν να θεωρηθούν ως μια τεχνική μείωσης της διαστατικότητας για δεδομένα με δομή γράφου, όπου τα δεδομένα εισόδου ορίζονται σε ένα μη ευκλείδειο, πολυδιάστατο, διακριτό πεδίο[28].
Το πρόβλημα γίνεται πιο περίπλοκο όταν κάθε κόμβος v έχει ένα διάνυσμα χαρακτηριστικών X (node features), το οποίο δεν συνδέεται απαραίτητα με την δομική πληροφορία (structural) γράφου του κόμβου, αλλά αποτελεί τη σημασιολογική (semantic) πληροφορία γράφου. Σε αυτές τις περιπτώσεις θέλουμε τα embeddings μας να αποτελούν ουσιαστικές αναπαραστάσεις

της συνδυασμένης πια πληροφορίας. Επομένως, ανάλογα με τη χρήση των node features ή όχι, σκοπός ενός μοντέλου είναι να μάθει τις χαρτογραφήσεις:

$$W \to Z$$

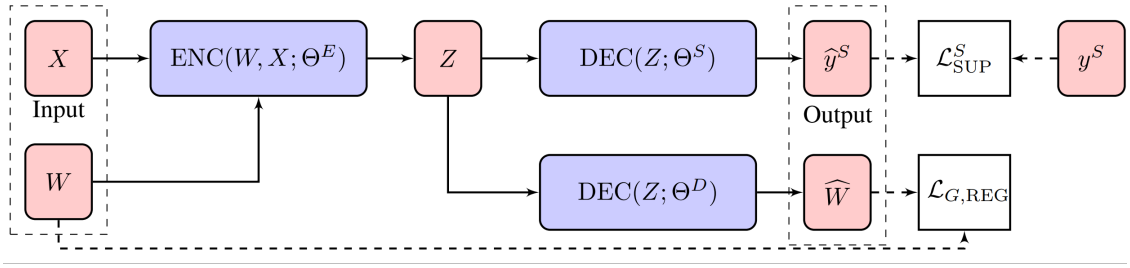
ή

$$W, X \to Z$$

Τέλος και σπανιότερα, υπάρχουν και αλγόριθμοι που μπορούν να χρησιμοποιήσουν χαρακτηριστικά ακμών (edge features), ως μέθοδο κανονικοποίησης (regularization) για τα node embeddings[29] ή σε message passing δίκτυα.

5.1.1 Μοντέλο κωδικοποιητή - αποκωδικοποιητή

Παρουσιάζουμε το μοντέλο κωδικοποιητή - αποκωδικοποιητή (encoder-decoder model - **GraphEDM**), όπως παρουσιάστηκε από τους Chami et al.[30], μέσω του οποίου μπορούν να περιγραφούν οι διαφορετικές μέθοδοι αναπαράστασης κόμβων.



Σχήμα 5.1: *Illustration of the GRAPHEDM framework. Based on the supervision available, methods will use some or all of the branches. In particular, unsupervised methods do not leverage label decoding for training and only optimize the similarity or dissimilarity decoder (lower branch). On the other hand, semi-supervised and supervised methods leverage the additional supervision to learn models' parameters (upper branch).*[30]

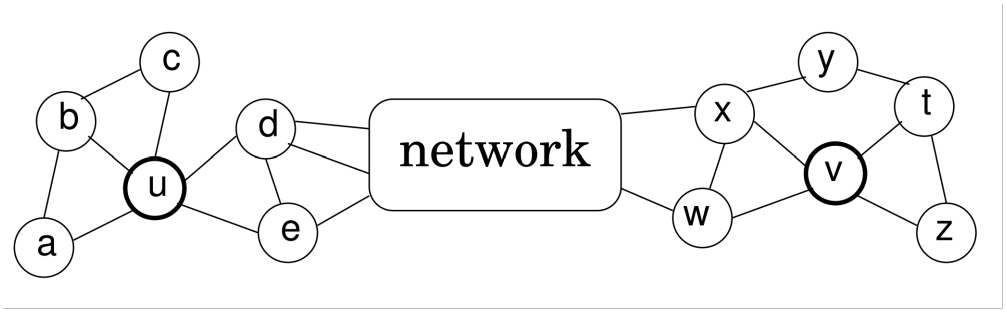
Για γράφο με σταθμισμένο πίνακα βαρών $\mathbf{W} \in R^{N\times N}$, (προαιρετικά) πίνακα χαρακτηριστικών κόμβων $\mathbf{X} \in R^{N\times D}$ και για (semi-)supervised πλαίσιο ετικέτες $S \in \{N, E, G\}$ ανάλογα αν πρόκειται για κόμβους, ακμές ή το σύνολο του γράφου το μοντέλο χωρίζεται σε 3 μέρη (ακολουθώντας το συμβολισμό του αρχικού paper):

- Δίκτυο κωδικοποιητή (**Graph encoder network**) $ENC_{\Theta^E} : R^{N\times N} \times R^{N\times D} \to R^{N\times L}$ το οποίο παράγει τον πίνακα των αναπαραστάσεων $Z \in R^{N\times L}$, παραμετροποιημένο από Θ^E.
- Δίκτυο αποκωδικοποιητή (**Graph decoder network**) $DEC_{\Theta^D} : R^{N\times L} \to R^{N\times N}$ το οποίο χρησιμοποιεί τον πίνακα αναπαράστασης κόμβων Z για τον υπολογισμό των τιμών ομοιότητας για όλα τα ζεύγη κόμβων στον πίνακα $W \in R^{N\times N}$, παραμετροποιημένο από Θ^D.
- Δίκτυο ταξινόμησης (**Classification network**)$DEC_{\Theta^S} : R^{N\times L} \to R^{N\times|y|}$ Το δίκτυο αυτό χρησιμοποιείται σε (ημι)εποπτευόμενες ρυθμίσεις και παραμετροποιείται από το Θ^S. Η έξοδος είναι μια κατανομή πάνω στις ετικέτες $\hat{y}^S$.

5.1.2 Διαφοροποιήσεις μεθόδων αναπαράστασης κόμβων

- Μία αρχική διαφοροποίηση που έχει πολύ μεγάλη σημασία για τη μελέτη μας και τις αποφάσεις αυτής είναι η κατηγοριοποίηση μεθόδων για node embeddings σε μεταγωγικές (**transductive**) και επαγωγικές (**inductive**). Μια μέθοδος ονομάζεται transductive όταν μπορούμε να παράξουμε αναπαραστάσεις μόνο για κόμβους που έχουν ήδη παρατηρηθεί στον γράφο κατά τη στιγμή της εκπαίδευσης και μπορεί να έχουν ή όχι μια ετικέτα. Οι transductive μέθοδοι δεν δημιουργούν προγνωστικό μοντέλο και η έξοδός τους είναι ένας πίνακας αναζήτησης της αναπαράστασης Z. Ως εκ τούτου, αυτές οι μέθοδοι αποκαλούνται επίσης μερικές φορές μέθοδοι ρηχής αναπαράστασης (shallow embedding methods) ή κωδικοποιητές (encoders). Σε πολλές περιπτώσεις, ωστόσο, μπορεί να είναι επιθυμητή μια inductive προσέγγιση, όπου οι προβλέψεις μπορούν να γίνουν σε δείγματα που δεν έχουν παρατηρηθεί στο γράφημα που βλέπουμε κατά την χρόνο εκπαίδευσης[31]. Το πλεονέκτημα λοιπόν των inductive μοντέλων είναι ότι μπορούν να χρησιμοποιηθούν σε δυναμικούς γράφους, στους οποίους προστίθενται νέοι κόμβοι και ακμές μετά την εκπαίδευση αλλά ακόμα και σε περιπτώσεις νέων γράφων.

- Μια ακόμα κατηγοριοποίηση μεθόδων αναπαράστασης κόμβων που έχει αναδυθεί είναι αν οι αναπαραστάσεις είναι θέσεως (positional) ή δομής (structural). Συνήθως, η πρώτη κατηγορία βασίζεται σε τεχνικές τυχαίου περιπάτου που ενσωματώνουν στις αναπαραστάσεις την πληροφορία εγγύτητας των κόμβων, δηλαδή κόμβοι που βρίσκονται σε κοντινή απόσταση στο γράφο θα έχουν και κοντινή απόσταση στο χώρο αναπαράστασης. Αυτό συμφωνεί και με τις θεωρίες της ομοφυλίας (homophily)[32] και της κοινωνικής επιρροής (social influence)[33], όπου για παράδειγμα σε ένα κοινωνικό δίκτυο είναι πιο πιθανό μια στενά συνδεδεμένη ομάδα να έχει παρόμοια ενδιαφέροντα. Όμως, πολλές φορές η εγγύτητα δύο κόμβων δεν αποτελεί σωστό κριτήριο αναπαράστασης, και δεν καταφέρνει να καταγράψει το **ρόλο** ενός κόμβου μέσα στο γράφο. Οι ρόλοι ενός κόμβου καταφέρνουν να καταγράψουν μια πιο τοπική πληροφορία και μπορούν να αντιπροσωπεύουν 'μοτίβα συνδεσιμότητας κορυφών, όπως κόμβοι, αστέρια-κέντρα, κόμβοι αστέρι-ακμή, σχεδόν-κλειδιά ή κορυφές που δρουν ως γέφυρες σε διαφορετικές περιοχές του γράφου'[34]. Έτσι, έχουν αναδειχθεί διάφορες structural μέθοδοι αναπαράστασης[35][36][37], όπου 'κόμβοι με παρόμοια χαρακτηριστικά κόμβων ή παρόμοιους δομικούς ρόλους σε ένα δίκτυο θα πρέπει να έχουν παρόμοιες αναπαραστάσεις, ανεξάρτητα από το πόσο μακριά βρίσκονται στον αρχικό γράφο'[28] (βλέπε σχήμα 5.2).



Σχήμα 5.2: Source: Struc2Vec

- Τέλος, οι μέθοδοι αναπαράστασης μπορούν να χωριστούν σε επιβλεπόμενες (**supervised**) και μη-επιβλεπόμενες (**unsupervised**), μια κατηγοριοποίηση που θα ακολουθήσουμε στη

συνέχεια. Στις unsupervised τεχνικές η βελτιστοποίηση γίνεται μόνο στο δίκτυο κωδικοποιητή με σκοπό τη διατήρηση των ιδιοτήτων του γράφου (κόστος ανακατασκευής γραφήματος ή κόστος απόστασης αναπαραστάσεων), ενώ στις supervised τεχνικές η βελτιστοποίηση γίνεται στο σύνολο του encoder-decoded με βάση κάποια συγκεκριμένη εργασία (downstream task), όπως για παράδειγμα ταξινόμηση κόμβοη ή ακμής, πρόβλεψη ακμής (link prediction), αναζήτηση γειτονιάς κτλ.

5.2 Μη-επιτηρούμενη μάθηση αναπαραστάσεων

Όπως αναφέραμε και παραπάνω οι τεχνικές που χρησιμοποιούν μόνο το πρώτο δίκτυο του GRAPHEDM framework, δηλαδή τον κωδικοποιητή για τη δημιουργία ενός πίνακα αναζήτησης αναπαραστάσεων είναι transductive και ονομάζονται **Shallow embedding methods** (μπορούν μάλιστα να θεωρηθούν και τεχνικές μείωσης διαστατικότητας, όμοια με μη-γραμμικές μορφές PCA). Αυτές χωρίζονται σε δύο βασικές κατηγορίες: στις μεθόδους που βασίζονται στην απόσταση (distance-based) και στις μεθόδους που βασίζονται στο εξωτερικό γινόμενο (outer product-based). Από αυτές θα επικεντρωθούμε στη δεύτερη κατηγορία, παρουσιάζοντας με τη σειρά τους δύο υποκατηγορίες της.

5.2.1 Factorization-based Models

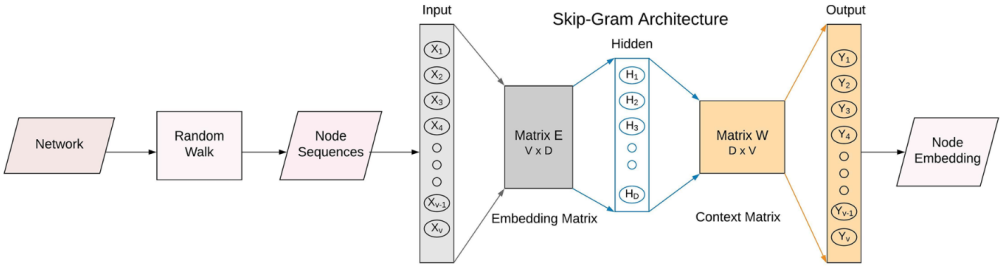
Οι μέθοδοι αυτοί βασίζονται στην ελαχιστοποίηση μιας συνάρτησης κόστους ανακατασκευής του γράφου:

$$L = ||s(W) - \hat{W}||_2^2$$

, όπου $\hat{W}$ ο ανακατασκευασμένος σταθμισμένος πίνακας γειτνίασης του γράφου και $s(W)$ αναπαράσταση χαμηλής βαθμίδας κάποιου πίνακα ομοιότητας που προκύπτει από κάποιον μετασχηματισμό, π.χ. λαπλασιανός πίνακας[38], high-order proximity matrix[39] κτλ. Αυτές οι μέθοδοι είναι όμως συχνά αναποτελεσματικές για μεγάλο αριθμό κόμβων γράφου λόγω της εκθετικής αύξησης του υπολογιστικού κόστους και του κόστους αποθήκευσης στη μνήμη του υπολογιστή, κάτι που έχει οδηγήσει στην εμφάνιση περισσότερων μεθόδων βασισμένων σε τυχαίους περιπάτους.

5.2.2 Random-walk based Models

Στη συνέχεια παρουσιάζουμε δύο από πιο σημαντικές και επιδραστικές τεχνικές αναπαράστασης κόμβων γράφου με βάση τυχαίους περιπάτους τα τελευταία χρόνια. Αυτές βασίζονται στο μοντέλο επεξεργασίας φυσικής γλώσσας, SkipGram[40] και η λογική τους φαίνεται στο σχήμα5.3.



Σχήμα 5.3: Source: Sumit Kumar's blog

DeepWalk

Το αρχικό στάδιο του DeepWalk[41] είναι η εκτέλεση τυχαίων (unbiased) περιπάτων στο γράφημα. Πραγματοποιούνται πολλαπλοί τυχαίοι περίπατοι ξεκινώντας από κάθε διαφορετικό κόμβο του γράφου, δημιουργώντας έτσι ένα σύνολο ακολουθιών (corpus), που μπορεί να προσομοιωθεί με ακολουθίες λέξεων που σχηματίζουν προτάσεις (ανάλογο). Ένα μοντέλο Skip-Gram εφαρμόζεται στη συνέχεια στο παραγόμενο corpus, που βελτιστοποιείται με στόχο τη μεγιστοποίηση της εξαρτημένης πιθανότητας των κόμβων πλαισίου (γειτονικών στους τυχαίους περιπάτους) δεδομένων του κόμβου-στόχου. Οι αναπαραστάσεις που μαθαίνονται αντιστοιχίζουν τους κόμβους σε έναν συνεχή διανυσματικό χώρο και οι κόμβοι που βρίσκονται κοντά (σε βήματα) στη δομή του γράφου αναμένεται να έχουν παρόμοιες αναπαραστάσεις.

Node2vec

Η μέθοδος Node2vec[42] ακολουθεί παρόμοια λογική με το DeepWalk, αλλά επεκτείνοντας την έννοια του context σε μια γειτονιά με την εισαγωγή προκατειλημμένων (biased) τυχαίων περιπάτων (μαρκοβιανές διαδικασίες δεύτερης τάξης), που συνδυάζουν την εξερεύνηση γράφου των αλγορίθμων breadth first search (BFS) και depth first search (DFS), με παραμέτρους που ελέγχουν την πιθανότητα απομάκρυνσης και backtracking.

Μερικά άλλα μοντέλα που χρησιμοποιούν το μοντέλο SkipGram είναι τα LINE[43] και HARP[44].

5.3 Νευρωνικά Δίκτυα Γράφων

Παρουσιάσαμε προηγουμένως μερικές τεχνικές παραγωγής αναπαραστάσεων που λειτουργούν σε unsupervised πλαίσιο. Στη συνέχεια, θα μπορούσαμε να τις χρησιμοποιήσουμε ως εισόδους για κάποια κλασικά MLP δίκτυα ώστε να πάρουμε προβλέψεις για τα κατάλληλα downstream tasks. Το πρόβλημα που δημιουργείται σε αυτήν τη περίπτωση είναι ότι τα node embeddings που δημιουργούνται από unsupervised τεχνικές δεν έχουν βελτιστοποιηθεί για κάποιο συγκεκριμένο downstream task με μια end-to-end λογική. Επιπλέον, οι ρηχές αναπαραστάσεις οδηγούν συχνά σε μη βέλτιστες λύσεις, επειδή δεν μπορούν να μοντελοποιήσουν αποτελεσματικά τη μη γραμμικότητα που είναι εγγενής στα δεδομένα του δικτύου. Αυτοί είναι μερικοί από τους λόγους ανάπτυξης και εντατικής μελέτης των νευρωνικών δικτύων γράφων (**Graph Neural Networks**) τα τελευταία χρόνια, και τα οποία θα μελετήσουμε στη συνέχεια.

5.3.1 Αρχικό Δίκτυο και Λογική Περάσματος Μηνύματος

Το αρχικό graph neural network (GNN)[45][46] βασίστηκε στην τεχνική περάσματος μηνύματος **Message Passing**. Μπορούμε να προσεγγίσουμε αυτήν τη τεχνική ως μια σταδιακή διάχυση πληροφορίας, όπου όλοι οι κόμβοι του γράφου διαδίδουν ένα μήνυμα (το οποίο σχετίζεται με μια κρυφή κατάσταση - αναπαράσταση κάθε κόμβου) σε επιπλέον απόσταση ενός βήματος τη φορά, μέχρι να προσεγγίστει μια κατάσταση ισορροπίας:

$$Z^{t+1} = ENC(X, W, Z^t; \Theta^E)$$

Αφού οι αναπαραστάσεις έχουν φτάσει πια σε ισορροπία μετά από T βήματα τις χρησιμοποιούμε για τις τελικές προβλέψεις χρησιμοποιώντας είτε το Graph decoder network είτε το Classifica-

tion network. Έστω λοιπόν για πρόβλημα ταξινόμησης κόμβων οι τελικές προβλέψεις:

$$\hat{y}^S = DEC(X, Z^t; \Theta^S)$$

Τα δύο στάδια επαναλαμβάνονται μέχρι κάποιο στάδιο (αριθμός εποχών κτλ.) και τα διαφορίσιμα βάρη Θ^{E} και Θ^{Σ} μαθαίνονται με backpropagation μέσω του αλγορίθμου Almeida–Pineda[47][48].

5.3.2 Συνελικτικά Νευρωνικά Δίκτυα Γράφων

Τα Συνελικτικά Νευρωνικά Δίκτυα Γράφων (**Convolutional graph neural networks - ConvGNNs**) είναι μία σημαντική κατηγορία μοντέρνων νευρωνικών δικτύων γράφων με μεγάλη επιτυχία τα τελευταία χρόνια και επεκτείνουν τα αντίστοιχα μοντέλα συνελίξεων **CNN**[49], που βρήκαν απήχηση σε προβλήματα εικόνας, σε γράφους. Μπορούν να χωριστούν σε δύο κατηγορίες: σε φασματικά (spectral) και χωρικά (spatial). Στη συνέχεια παρουσιάζουμε δύο σημαντικά μοντέλα, όπου το πρώτο είναι spectral και το δεύτερο spatial.

GCN

Το μοντέλο GCN[50] είναι ένα μοντέλο για semi-supervised learning και λειτουργεί συγκεντρώνοντας (aggregation) πληροφορία από τους γείτονες ενός κόμβου με κατάλληλα συνελικτικά φίλτρα, έτσι ώστε παρόμοια με τα μοντέλα CNN να μην επηρεάζεται από την αλλαγή διάταξης των κόμβων (permutation invariance) ή τη σχετική θέση των μοτίβων που αναγνωρίζει μέσα στο γράφο. Επίσης, όμοια με τα γνωστά CNN μοντέλα τα φίλτρα ή aggregators μοιράζουν τα ίδια βάρη σε όλες τις περιοχές του γραφήματος (parameter sharing) και μπορούν να αναπτυχθούν σε στρώματα, με το ένα να εφαρμόζεται πάνω στα προηγούμενα με ιεραρχική δομή.
Ο κανόνας μετάδοσης ενός συνελικτικού στρώματος δίνεται από τον κανόνα:

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in N(v_i)} \frac{1}{\sqrt{d_i \cdot d_j}} \cdot h_j^{(l)} \cdot W^{(l)}\right)$$

, όπου $h_i^{(l+1)}$ είναι η αναπαράσταση του κόμβου v_i στο στρώμα l, $W^{(l)}$ είναι ένας πίνακας εκμάθησης βαρών για το στρώμα l, σ είναι μια συνάρτηση ενεργοποίησης και d_i και d_j είναι αντίστοιχα οι βαθμοί των κόμβων i και j. Μια πιο γρήγορη παραλλαγή του μοντέλου GCN είναι το μοντέλο FastGCN[51] το οποίο με κατάλληλους μετασχηματισμούς εφαρμόζει τεχνικές Monte-Carlo, που επιτρέπουν την batched εκπαίδευση του μοντέλου με αποτέλεσμα πολύ μεγαλύτερη ταχύτητα. Το μειονέκτημα του μοντέλου GCN είναι ότι απαιτεί την αποθήκευση ολόκληρου του πίνακα γειτνίασης του γραφήματος, κάτι που για μεγάλους γράφους είναι υπολογιστικά δαπανηρό.

GraphSAGE

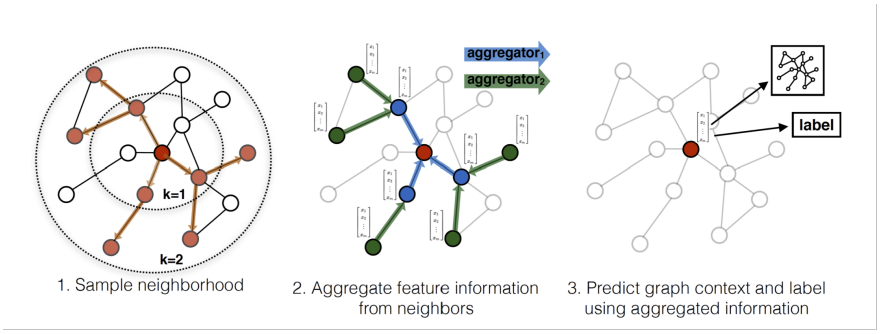
Το μοντέλο GraphSAGE[52] απαντάει σε δύο σημαντικούς περιορισμούς των προηγούμενων μοντέλων, όπως το GCN, τα οποία είναι ότι στην ουσία χρησιμοποιώντας ολόκληρο τον πίνακας γειτνίασης ή τον λαπλασιανό είναι εγγενώς transductive μοντέλα και δεύτερων το υπολογιστικό κόστος που αναφέραμε. Το μοντέλο μπορεί να μάθει aggregation συναρτήσεις (πίνακες) που συνδυάζουν τοπικές πληροφορίες (η K hop-distance αντιστοιχεί στον αριθμό των συνελικτικών στρωμάτων που διαδίδουν επαναληπτικά πληροφορίες) έτσι ώστε να μπορεί να παράγει

αναπαραστάσεις για νέους κόμβους (επαγωγικός αλγόριθμος). Συγκεκριμένα, στην αρχή της διαδικασίας ο αλγόριθμος δειγματοληπτεί γειτονικούς κόμβους για κάθε κόμβο v ώστε να συνθέσει τη γειτονιά του $N(v)$ και αναθέτει στον καθένα μια αρχική αναπαράσταση $h_0^v \leftarrow x_v$, όπου x το διάνυσμα χαρακτηριστικών του κόμβου που δίνεται ως είσοδος. Στη συνέχεια για κάθε βήμα από 1 εώς K και για κάθε κόμβο, ο αλγόριθμος κάνει aggregate την αναπαράσταση του κόμβου μαζί με αυτών της γειτονιάς του:

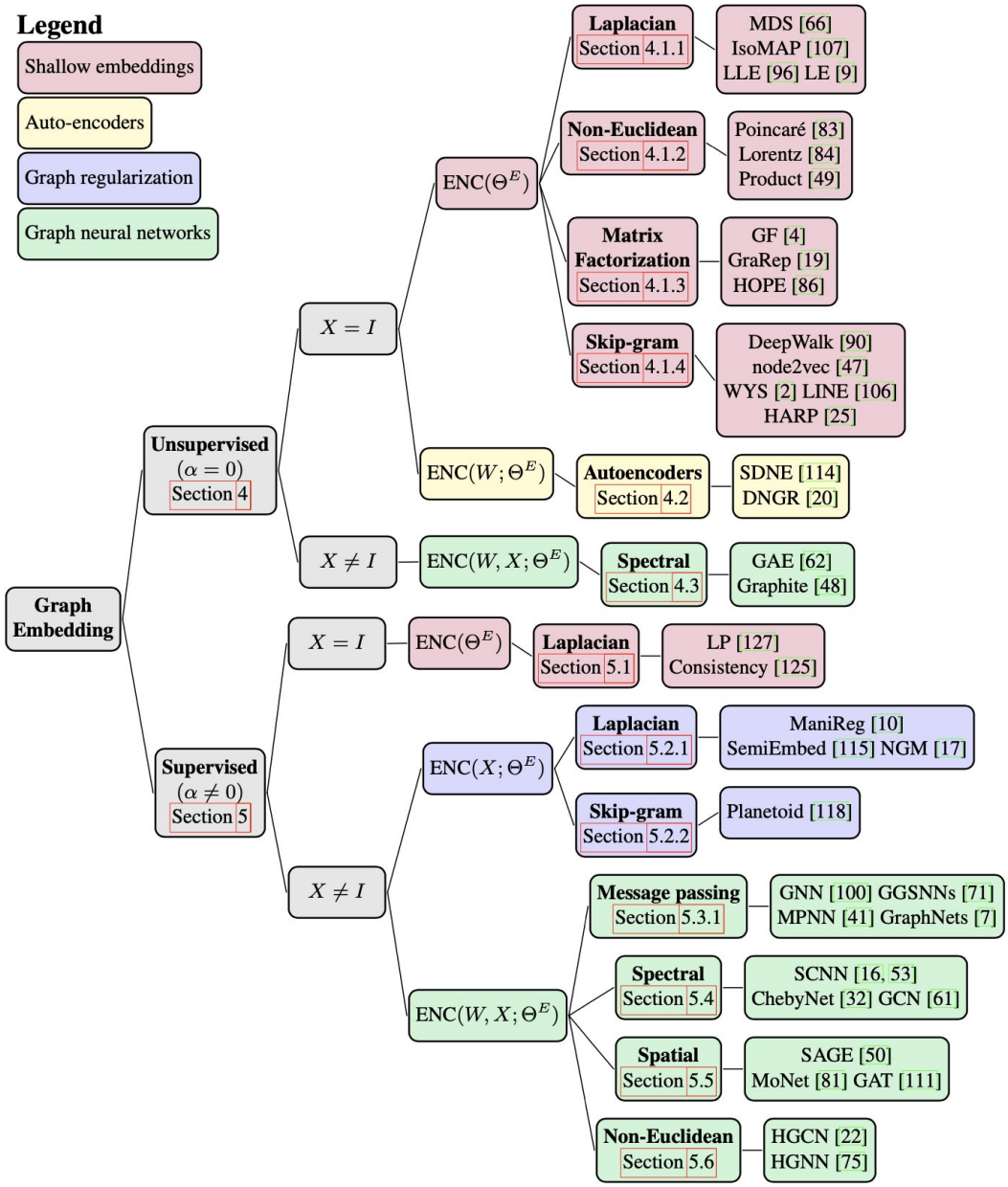
$$h_{N(v)}^k \leftarrow AGGREGATE_k\left(\{h_u^{k-1}, \forall u \in N(v)\}\right)$$

και έπειτα χαρτογραφεί το concatenated διάνυσμα στην δεδομένη διάσταση του, εφαρμόζοντας στο τέλος μια συνάρτηση ενεργοποίησης:

$$h_v^k \leftarrow \sigma\left(W_k \cdot CONCAT(h_v^{k-1}, h_{N(v)}^k)\right)$$



Σχήμα 5.4: Λογική δειγματοληψίας και συγκέντρωσης GraphSAGE



Σχήμα 5.5: Μια unified ταξινόμηση μεθόδων μάθησης σε γράφους[30]

Μέρος II

Κύριο Μέρος

Κεφάλαιο 6

Θεωρητική τεκμηρίωση

6.1 Σχετικές έρευνες

Το πεδίο της εκμάθησης αναπαραστάσεων για δυναμικούς γράφους βρίσκεται ακόμα σε αρχικό στάδιο με πολλά διαφορετικά μοντέλα να προτείνονται σχετικά με τη φύση των αναπαραστάσεων και τι ακριβώς πρέπει να περιγράφουν στο χρόνο. Τα Temporal Graph Networks που εισήχθησαν από τους Rossi et al.[53] και γενικεύουν προηγούμενα μοντέλα[54][55], προτείνουν μια μέθοδο ενημέρωσης των node embeddings στο χρόνο εισάγοντας ένα είδος μνήμης για κάθε κόμβο και δουλεύοντας με συνεχείς δυναμικούς γράφους. Εναλλακτικά, έχουν προταθεί διάφορα άλλα μοντέλα ενημέρωσης των node embeddings χρησιμοποιώντας μονάδες LSTM και GRU ώστε να λάβουν υπόψη τη χρονική συσχέτιση[56][57][58]. Στα συγκεκριμένα μοντέλα δεν προβλέπονται επανεκπαιδεύσεις παρά μόνο ενημερώσεις, αφού δίνεται περισσότερο έμφαση στην τροχιά ενός κόμβου στο πεδίο αναπαράστασης.
Στον αντίποδα, υπάρχουν σχετικές μελέτες, οι οποίες επικεντρώνονται στην ανίχνευση του concept drift σε γράφους. Για παράδειγμα, στη μελέτη των Paudel et al.[59], η οποία μελετά graph streams υιοθετείται μια λογική ενός παράθυρου ολίσθησης πάνω σε discriminative υπογράφους που παράγονται από κάθε γράφο και ανίχνευση concept drift μέσω του υπολογισμού της εντροπίας παράθυρου, όπως και στη μελέτη των Yao et al.[60]. Η τεχνική αυτή είναι unsupervised και ανεξάρτητη του μοντέλου ταξινομητή, όμως επικεντρώνεται στην εργασία της ταξινόμησης γράφων και δεν προτείνει κάποιο ολοκληρωμένο σχέδιο δράσης μετά την ανίχνευση του concept drift, παρά υπονοείται κάποιο retraining.
Στην μελέτη "Instant Graph Neural Networks for Dynamic Graphs"[61], η οποία μελετά το πρόβλημα του graph representation learning σε συνεχείς δυναμικούς γράφους, υπάρχει ένας καθορισμένος αριθμός πιθανών επανεκπαιδεύσεων καθ' όλη τη διάρκεια της ζωής του δυναμικού γράφου ως περιορισμός-προϋπολογισμού[1]. Ο αλγόριθμος διατηρεί έναν πίνακα αναπαράστασης γραφήματος Z, ο οποίος ενημερώνεται χωρίς να διατρέχει ολόκληρο τον γράφο, αλλά με ενημερώσεις διάδοσης στους επηρεαζόμενους κόμβους, όπως με Personalized PageRank. Στη συνέχεια, χρησιμοποιεί ιστορικά δεδομένα για τον υπολογισμό ενός κατωφλίου που θα πυροδοτεί επανεκπαιδεύσεις. Η μέθοδος αυτή είναι ιδιαίτερα περιοριστική καθώς θέτει εκ προοιμίου τον αριθμό πιθανών επανεκπαιδεύσεων και δεν μπορεί να προσαρμοστεί σε μη αναμενόμενες δυναμικές εξελίξεις γράφων π.χ. σε καθεστώς έντονου concept drift (βλέπε ενότητα 6.2.1).
Παρ'όλα αυτά, η σύνδεσή μοντέλων με την πλευρά του κόστους επανεκπαίδευσης είναι ισχνή από

[1] Οι συγγραφείς της μελέτης δεν έχουν ανοιχτό κώδικα ούτε πλήρεις οδηγίες υλοποίησης, οπότε και δεν έγινε πειραματική σύγκριση των μεθόδων.

πλευράς βιβλιογραφίας σε σχέση με τον τομέα της μηχανικής μάθησης για ευκλείδεια δεδομένα. Έτσι κατά κύριο λόγο βασιζόμαστε στην έρευνα από τους Mahadevan και Mathioudakis[62], η οποία μελετάει εμπεριστατωμένα το trade-off μεταξύ της ακρίβειας σε ένα predictive task και το κόστος επανεκπαιδεύσεων του μοντέλου, εισάγοντας την έννοια του model staleness.

6.2 Διατύπωση του προβλήματος

6.2.1 Ολίσθηση δεδομένων

Η ολίσθηση δεδομένων (**data drift**) είναι ένας καλά τεκμηριωμένος όρος που μπορεί να χωριστεί σε δύο κατηγορίες: **concept drift** (ή και model drift), η οποία είναι όταν η κατανομή πιθανότητας των μεταβλητών-στόχων επί των μεταβλητών εισόδου δεδομένων στο διάνυσμα χαρακτηριστικών των δεδομένων αλλάζει, και **covariate drift**, η οποία είναι όταν η κατανομή των δεδομένων εισόδου αλλάζει, κάτι που μπορεί επίσης να περιγραφεί ως λήψη δεδομένων από διαφορετικές περιοχές του χώρου των χαρακτηριστικών. Προχωρούμε στη συνέχεια σε μια πιο αναλυτική περιγραφή των εννοιών, πώς προκύπτουν και ποια είναι η σημασία τους.

Concept Drift

Η έννοια του Concept Drift εισάγεται για πρώτη φορά στο paper με τίτλο "Incremental learning from noisy data" [63] και ορίζεται ως η αλλαγή των στατιστικών ιδιοτήτων μιας εργασίας πρόβλεψης (target domain) με το χρόνο. Μπορούμε να ορίσουμε μαθηματικά το concept drift μεταξύ των χρονικών στιγμών $t1$ και $t2$ ως:

$$\exists X : p_{t1}(X, y) \neq p_{t2}(X, y)$$

όπου p_t δηλώνει την κοινή κατανομή τη χρονική στιγμή τ μεταξύ του συνόλου των μεταβλητών εισόδου X και της μεταβλητής στόχου y. Μπορεί να χωριστεί περαιτέρω [64] στις κατηγορίες:

- Sudden Drift
- Gradual Drift
- Incremental Drift
- Recurring Concepts

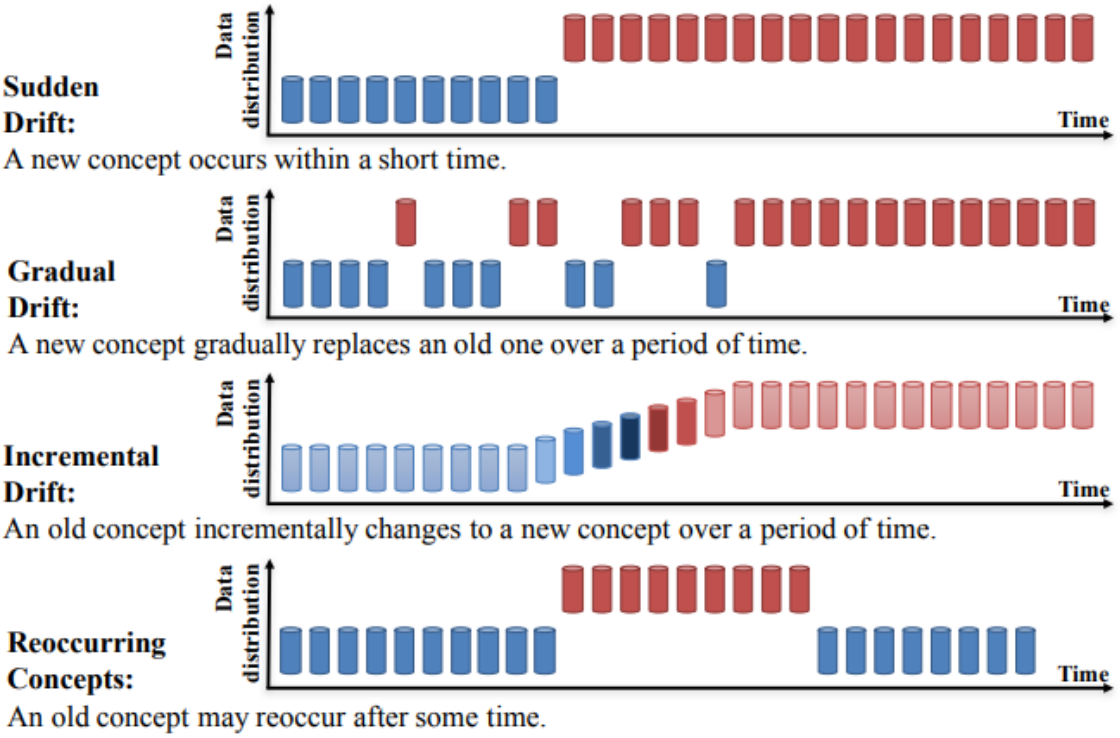
που συνοψίζονται στο σχήμα 6.1.
Μερικά κοινά παραδείγματα concept drift αποτελούν οι καταναλωτικές συμπεριφορές, οι οποίες αλλάζουν με βάση τις εποχές του χρόνου ή και λόγω τυχαίων γεγονότων και η ικανότητα των spam filters να κάνουν flag σωστά μηνύματα που έχουν εξελιχθεί χρησιμοποιώντας ή όχι συγκεκριμένες λέξεις και εκφράσεις.

Covariate Drift

Ο αντίστοιχος μαθηματικός ορισμός για ύπαρξη covariate drift μεταξύ των χρονικών στιγμών $t1$ και $t2$ ορίζεται ως:

$$\exists X : p_{t1}(X) \neq p_{t2}(X)$$

Συνήθως οι χρονικές στιγμές $t1$ και $t2$ θεωρούνται οι χρόνοι εκπαίδευσης (training) και αξιολόγησης (testing) αντίστοιχα.
Ένα παράδειγμα covariate drift θα μπορούσε να είναι στην περίπτωση ενός μοντέλου πρόβλεψης



Σχήμα 6.1: Concept drift [64]

πωλήσεων με βάση μεταξύ άλλων το εισόδημα πελατών και εκπαιδευμένο σε πελάτες χαμηλού εισοδήματος, η εμφάνιση πολλών περισσότερων πελατών υψηλού εισοδήματος.
Και στις δύο περιπτώσεις είναι γνωστό ότι τελικά το data drift (χρησιμοποιείται ως γενικότερος όρος και για τις δύο περιπτώσεις) οδηγεί σε μειωμένο model accuracy και πλήττει ουσιαστικά την αξία χρήσης του προβλεπτικού μοντέλου, ειδικά αν πρόκειται για συστήματα που είναι σχεδιασμένα να είναι online, δηλαδή να παρέχουν προβλέψεις σε πραγματικό χρόνο.
Σε πολλές περιπτώσεις, στο πρόβλημα των οποίων θα επικεντρωθούμε σε αυτή τη μελέτη, τα μοντέλα μηχανικής μάθησης δεν μπορούν να προσαρμοστούν με online τρόπο ή χρειάζεται να έχουν εκπαιδευτεί σε όλα τα δεδομένα. Ειδικά στις περιπτώσεις των γράφων, όπου όλη η υποκείμενη δομή μπορεί να αλλάξει με μόνο κάποιες προσθήκες ακμών, ένα retraining σε όλα ή μόνο στα νέα δεδομένα μπορεί να είναι επιβεβλημένο. Στη συνέχεια κάνουμε αναφορά στην έννοια της συνεχούς μάθησης (Continuous Learning) για λόγους πληρότητας και έπειτα μεταφέρουμε πια το πρόβλημα αποκλειστικά σε δεδομένα γράφων.

Continuous Learning

Ο όρος Continuous Learning, ο οποίος πολλές φορές χρησιμοποιείται έναντι ή εναλλακτικά του όρου Online Learning (κυρίως αναφέρεται στο πλαίσιο ενός data stream), όπου τα δεδομένα έρχονται ένα-ένα) αναφέρεται στην εκμάθηση των μοντέλων από νέα δεδομένα στο χρόνο, χωρίς την ανάγκη ενός retraining στο σύνολο των ιστορικών δεδομένων. Περαιτέρω, σε αντίθεση με τη σταδιακή εκμάθηση (Incremental Learning), όπου το μοντέλο ML μαθαίνει και βελτιώνει τις γνώσεις του σταδιακά, χωρίς να ξεχνά ενεργά τις πληροφορίες που έχει αποκτήσει προη-

γουμένως (αυτό μπορεί να συμβεί σταδιακά στην περίπτωση ύπαρξης covariate drift), μπορούμε να ορίσουμε και την έννοια (Adaptive Learning), στην οποία μελετώνται ρητές στρατηγικές ώστε τα μοντέλα να μπορούν να ξεχνούν άσχετες, δηλαδή μη χρήσιμες πια πληροφορίες. Οι Adaptive learning learners μπορούν να ερμηνευτούν και ως 'προηγμένοι' incremental αλγόριθμοι μάθησης που μπορούν να προσαρμόζονται στις αλλαγές σε μια ροή δεδομένων[65][66][67]. Τέλος, ένα φαινόμενο που παρατηρείται περισσότερο στο πλαίσιο του Continuous Learning βαθιών νευρωνικών δικτύων και αποτελεί επίσης αντικείμενο έντονης έρευνας είναι το catastrophic forgetting[68]. Το φαινόμενο αυτό μπορεί να δικαιολογηθεί ως αποτέλεσμα του διλήμματος σταθερότητας-πλαστικότητας[69]. Πιο συγκεκριμένα, ένα νευρωνικό δίκτυο απαιτεί επαρκή πλαστικότητα για να αποκτήσει νέες ερμηνευτικές ικανότητες, αλλά οι μεγάλες αλλαγές βάρους θα προκαλέσουν λήθη διαταράσσοντας τις αναπαραστάσεις που είχαν γίνει γνωστές προηγουμένως. Αν διατηρούσαμε σταθερά τα βάρη του δικτύου, τα καθήκοντα που εκτελούνταν με επιτυχία προηγουμένως δεν θα 'ξεχνιόντουσαν', με το κόστος όμως της αδυναμίας για εκμάθηση νέων. Τα τελευταία χρόνια έχουν εμφανιστεί πολλές διαφορετικές προσεγγίσεις στην αντιμετώπιση του προβήματος αυτού τόσο σε supervised setting[70][71][72] όσο και σε unsupervised[73][74][75][76].
Οι τεχνικές Adaptive Learning που αναπτύσσονται μπορούν λοιπόν να εντοπίσουν και να ανταπεξέλθουν στην εμφάνιση data drift, χωρίς την ανάγκη μιας επανεκπαίδευσης. Βέβαια οι τεχνικές αυτές είναι πολλές φορές model-specific και δεν έχουν βρει ακόμα ευρεία εφαρμογή σε νευρωνικά δίκτυα γράφων με τα οποία ασχολούμαστε σε αυτή την έρευνα. Όπως θα αναπτυχθεί στη συνέχεια, αντί για μερικές ενημερώσεις υιοθετούμε την κλασική μέθοδο των περιστασιακών επανεκπαιδεύσεων με έμφαση στη συχνότητα και το κόστος αυτών.

Data drift σε γράφους

Σημειώνουμε ότι χρησιμοποιούμε έναν αλγόριθμο με επίγνωση του φόρτου εργασίας, πράγμα που σημαίνει ότι το data drift λαμβάνεται υπόψη μόνο σε σχέση με το φόρτο εργασίας, που στο εξής αναφέρεται ως **queries**. Τα queries θεωρούνται μια εντελώς διαφορετική ροή δεδομένων (data stream) χωρίς ετικέτες, π.χ. κόμβοι, οι οποίοι μπορεί να έχουν ανεξάρτητη κατανομή χαρακτηριστικών από τη ροή δεδομένων ή να είναι υποσύνολο αυτής.
Παρ΄ όλα αυτά, τώρα το ερώτημα είναι πώς θα μπορούσαμε να ορίσουμε και να ανιχνεύσουμε το data drift σε γραφήματα, όπου αντί για σημεία/δείγματα με ένα πολυδιάστατο διάνυσμα χαρακτηριστικών, έχουμε κόμβους σε ένα γράφο. Για να αντιμετωπίσουμε αυτό το πρόβλημα, χρησιμοποιούμε νευρωνικά δίκτυα γράφων (GNN), τα οποία μπορούν να αποτυπώσουν ουσιώδεις αναπαραστάσεις των κόμβων. Τα μοντέλα GNN έχουν φέρει επανάσταση στον τομέα της μηχανικής μάθησης για γράφους τα τελευταία χρόνια και έχουν λειτουργήσει με τη χρήση πολλαπλών στοίχων στρωμάτων που μπορούν να εκτελέσουν πράξεις aggregation και pooling την πληροφορία γειτονιάς ενός κόμβου. Με την απεικόνιση κάθε κόμβο σε ένα embedding, μπορούμε τώρα να επεκτείνουμε προηγούμενες εργασίες [62], οι οποίες ποσοτικοποιούσαν το ανεπιθύμητο data drift ή το staleness cost του μοντέλου ('η απώλεια στην απόδοση λόγω της διατήρησης ενός ML μοντέλο εκπαιδευμένο σε παλιά δεδομένα') σε σχέση με το κόστος επανεκπαίδευσης του μοντέλου. Ως εκ τούτου, από εδώ και στο εξής, όταν αναφερόμαστε σε δεδομένα ή σημεία ερωτήσεων (queries), αναφερόμαστε στα node embeddings που ανήκουν σε κάθε ροή (data stream και query stream).

6.2.2 Model Staleness

Μέχρι στιγμής έχουμε μιλήσει για το πώς η διατήρηση ενός μοντέλου που έχει εκπαιδευτεί σε παλιά δεδομένα και έχει γίνει παρωχημένο μπορεί συνεπώς να οδηγήσει σε μειωμένη προβλεπτική ικανότητα σε σχέση με την κατανομή των queries. Επειδή τα queries δεν έχουν ετικέτες, η απώλεια ακρίβειας είναι ουσιαστικά μια αναμενόμενη απώλεια, την οποία ορίζουμε στη συνέχεια. Ορίζουμε σε ακολουθία με το αρχικό paper, 'το κόστος model staleness για ένα μόνο query q δεδομένου ενός μοντέλου M και δεδομένων D' ως:

$$\psi(q, D, M) = \frac{1}{|D|} \sum_{(x,y)\in D} \text{sim}(q, x) \cdot \ell(M, x, y)$$

, όπου sim (.,.) ένα μέτρο ομοιότητας μεταξύ των διανυσμάτων q και x (στην υλοποίησή μας χρησιμοποιούμε ευκλείδεια απόσταση), και l είναι η απώλεια του μοντέλου M, δεδομένων των χαρακτηριστικών εισόδου x και των τιμών της ετικέτας y. [2]
Το model staleness για ένα query batch Q_t δίνεται από το άθροισμα του κόστους στασιμότητας επί όλων των queries:

$$\Psi(Q_t, D_t, M_t) = \sum_{q\in Q_t} \psi(q, D_t, M_t)$$

Μπορούμε, λοιπόν, να σκεφτούμε το μοντέλο σταθερότητα του μοντέλου ως αναμενόμενη απόδοση ή απώλεια των queries.
Ορίζουμε το σχετικό (relative) model staleness ως εξής:

$$\overline{\Psi_{t,t'}} = \Psi(Q_t, D'_t, M_t) - \Psi(Q_t, D_t, M_t)$$

, όπου t' είναι η χρονοσφραγίδα της τρέχουσας παρτίδας δεδομένων και t είναι η χρονοσφραγίδα της τελευταίας εκπαίδευσης του μοντέλου. Από εδώ και στο εξής χρησιμοποιούμε το relative model staleness cost ως μέτρο του κόστους, επειδή επιθυμούμε το staleness cost να αποτυπώνει το συνδυασμό του expected query loss με το data drift. Διαφορετικά, το απόλυτο staleness θα ήταν ένας απλός δείκτης της κατανομής των queries (όπως θα συζητήσουμε πιο διεξοδικά στη συνέχεια).

6.2.3 Κόστος επανεκπαίδευσης - Trade-off

Ωστόσο, όπως τονίστηκε μέχρι στιγμής, οι δυνατότητες της μεθόδου μας δεν περιορίζονται απλώς στην ανίχνευση data drift, αλλά στην πρόταση μιας πλήρους στρατηγικής που λαμβάνει επίσης υπόψη τη σχετική με το κόστος επανεκπαίδευσης ακρίβεια του μοντέλου και συνεπώς ενημερώνει αποτελεσματικά σε σχέση με τις δύο μετρικές το μοντέλο. Ιδανικά, θα θέλαμε ο αλγόριθμός επανεκπαίδευσης μας να εξάγει το ελάχιστο κόστος στρατηγικής, δηλαδή το μικρότερο δυνατό συνολικό κόστος μιας σειράς αποφάσεων για κάθε batch, οι οποίες είτε παράγουν ένα κόστος επανεκπαίδευσης είτε ένα αναμενόμενο κόστος ερωτημάτων (expected query loss). Έτσι, προκειμένου να ενσωματώσουμε ουσιαστικά το συμβιβασμό μεταξύ υψηλότερου κόστους επανεκπαίδευσης και μειωμένης ακρίβειας, πρέπει να ορίσουμε το κόστος επανεκπαίδευσης 'ως τη σχετική αξία των πόρων που δαπανώνται στις κοινές μονάδες του model staleness cost'[62]. Η παράμετρος $\boldsymbol{\kappa}$ θα σηματοδοτεί τότε πόσες λανθασμένες ταξινομήσεις ερωτημάτων είμαστε

[2] Μία μετάφραση του όρου, η οποία όμως δεν ανταποκρίνεται πλήρως στον αγγλικό ορισμό θα μπορούσε να είναι και "κόστος στασιμότητας μοντέλου' ή απλά "κόστος στασιμότητας".

πρόθυμοι να 'ανταλλάξουμε' για μια πλήρη επανεκπαίδευση του μοντέλου. Το κ εξαρτάται από το μέγεθος του batch, τον αριθμό των παραμέτρων προς εκπαίδευση του μοντέλου, των πόρων του συστήματός μας και φυσικά της αναμενόμενης ή απαιτούμενης ακρίβειας του online μοντέλου. Ως εκ τούτου, ορίζεται σε μια τιμή που αντικατοπτρίζει όλες αυτές τις παραμέτρους από το διαχειριστή του συστήματος.

6.2.4 Δεδομένα

Τα δεδομένα μας αποτελούνται από μια σειρά στιγμιότυπων συνεκτικών γραφημάτων (graph snapshots) G^i, όπου κάθε στιγμιότυπο γραφήματος μπορεί να προκύψει από το προηγούμενο με οποιονδήποτε αριθμό προσθηκών ή διαγραφών κόμβων και ακμών. Στην ειδική περίπτωση που μελετάμε, τα data και query streams περιέχουν σε κάθε batch δύο μη επικαλυπτόμενα σύνολα κόμβων, τα μεν με ετικέτες και τα δε χωρίς, που το άθροισμά τους αποτελεί το σύνολο των κόμβων του graph snapshot της δεδομένης χρονικής στιγμής. Εργαζόμαστε έτσι σε ένα διακριτό (discreet) περιβάλλον σε σύγκριση με διαφορετικές μελέτες, οι οποίες διερευνούν συνεχείς δυναμικούς γράφους, όπου κάθε προσθήκη ή διαγραφή είναι ένα ξεχωριστό χρονικό γεγονός. Τέλος, κάθε κόμβος γράφου έχει ένα διάνυσμα χαρακτηριστικών V καθορισμένου μεγέθους και μια ετικέτα y αν ανήκει στο data stream.

6.2.5 Μοντέλο

Το μοντέλο M που εφαρμόζουμε από άκρο σε άκρο (end-to-end) κάνει map το διάνυσμα χαρακτηριστικών ενός κόμβου σε συνδυασμό με τη σχετική πληροφορία του γραφήματος σε έναν προβλεπόμενο στόχο y. Στο streaming setting, το M_t αναφέρεται σε ένα μοντέλο που έχει εκπαιδευτεί στο σύνολο των δεδομένων από το G_t με επιβλεπόμενο τρόπο. Το κρίσιμο σημείο είναι ότι κάθε μοντέλο θα πρέπει να είναι εξοπλισμένο με μια συνάρτηση embedding που μπορεί να αντιστοιχίσει έναν κόμβο στη χαμηλής διάστασης αναπαράστασή του X, καθώς αυτές οι αναπαραστάσεις χρειάζονται για τον υπολογισμό του κόστους στασιμότητας, το οποίο χρησιμοποιεί μετρικές ομοιότητας μεταξύ σημείων δεδομένων (κόμβων) στο χώρο αναπαράστασης (embedding space). Για το λόγο αυτό, πρέπει να αποσυνδέσουμε το embedding task από το classification task με τη χρήση ενός απλού MLP στρώματος ή απλά ένα πλήρες συνδετικό (fully connected) στρώμα βάθους ίσου με ένα (όπως κάνουμε στη δικιά μας μελέτη) που απεικονίζει το embedding X, που έχει ήδη ενσωματώσει τις πληροφορίες του γράφου, στον προβλεπόμενο στόχο y.

Για τους σκοπούς της μελέτης μας και το πλαίσιο της online μας αξιολόγησης χρειαζόμαστε ένα μοντέλο που μπορεί να παράγει αναπαραστάσεις (και προβλέψεις κλάσεων) για νέα δεδομένα, δηλαδή για κόμβους που δεν υπήρχαν στον αρχικό γράφο στον οποίο εκπαιδεύτηκε και προστέθηκαν αργότερα ή για κόμβους που υπήρχαν αλλά η γειτονιά τους έχει αλλάξει λόγω προσθήκης ακμών και συνεπώς οι αναπαραστάσεις τους στο γράφημα αναμένεται να αλλάξουν επίσης, αν γίνει νέα επανεκπαίδευση. Ένα online μοντέλο θα πρέπει επίσης να είναι σε θέση να προσομοιώνει αυτή την αλλαγή.

Για το λόγο αυτό, επιλέγουμε ως πρότυπο μοντέλο αναπαράστασης το GraphSAGE, το οποίο σε αντίθεση με τα περισσότερα κλασικά μοντέλα node embedding που βασίζονται στον πλήρη πίνακα γειτνίασης του ενός γράφου κατά τη διάρκεια της εκπαίδευσης και συνεπώς δεν μπορούν να γενικεύσουν σε αόρατους κόμβους (μεταγωγική μάθηση) μπορεί να παράγει αναπαραστάσεις για αθέατους κόμβους. Ένας δεύτερος λόγος για τον οποίο επιλέξαμε το GraphSAGE ως το τυπικό μας μοντέλο είναι ότι σε αντίθεση με το προηγούμενο μοντέλο GCN, αυτό χρησιμοποιεί

τεχνικές δειγματοληψίας και έτσι μπορεί να επεκταθεί αποτελεσματικά για μεγάλους γράφους όπως αυτός που χρησιμοποιούμε για την τελική μας αξιολόγηση. Σημειώνουμε ότι η θεωρητική σύγκριση διαφορετικών μοντέλων για node embeddings δεν είναι το αντικείμενο της μελέτης μας, οπότε το GraphSAGE χρησιμοποιείται σε όλη την παρούσα μελέτη.

6.2.6 Στρατηγική Επανεκπαίδευσης

Για N παρτίδες δεδομένων, η στρατηγική ισούται με μια ακολουθία από N

$$d_t \in \{\text{Διατήρηση}, \text{Επανεκπαίδευση}\} \quad (\{\text{ Keep, Retrain}\})$$

αποφάσεις και αποτελεί τη φάση της διαδικτυακής αξιολόγησης (Online Evaluation Phase).

Κόστος στρατηγικής (Strategy cost)
Το κόστος μιας απόφασης για μια μόνο παρτίδα σε χρόνο t και εκπαιδευμένο τελευταία φορά σε χρόνο t' δίνεται από τη σχέση:

$$C[t', t] = \begin{cases} \overline{\Psi_{t,t'}}, & \text{if } t' < t \\ \kappa, & \text{if } t' = t \\ \infty, & \text{otherwise} \end{cases}$$

Το κόστος μιας στρατηγικής είναι το άθροισμα του κόστους όλων των αποφάσεων κατά τη διάρκεια της online φάσης αξιολόγησης.
Αλγόριθμος επανεκπαίδευσης (Retraining algorithm) Οι αλγόριθμοι που παρακολουθούν το online μοντέλο και ενεργοποιούν τις επανεκπαιδεύσεις αποτελούν ουσιαστικά μια συνάρτηση απόφασης:

$$R : (D_t, Q_t, M_t|\theta) \rightarrow \{\text{Keep, Retrain}\}$$

, όπου D_t, Q_t, M_t είναι η τρέχουσα παρτίδα δεδομένων, η παρτίδα ερωτημάτων και το μοντέλο αντίστοιχα και ϑ είναι ορισμένες παράμετροι που μαθαίνονται από τους αλγορίθμους κατά τη διάρκεια της offline φάσης βελτιστοποίησης.
Οι μόνες δύο πιθανές αποφάσεις που μελετάμε στο πλαίσιο αυτής της μελέτης είναι οι:

- Keep, που σημαίνει ότι δεν υπάρχει επανεκπαίδευση και
- Retrain, η οποία σημαίνει ότι αφού το μοντέλο M_t χρησιμοποιηθεί για την παραγωγή embeddings ή προβλέψεων για τρέχοντα queries Q_t, στη συνέχεια επανεκπαιδεύεται στα δεδομένα D_t

6.2.7 Αλγόριθμοι

Δοκιμάζουμε τρεις διαφορετικούς αλγορίθμους επανεκπαίδευσης σε ακολουθία με την αρχική εργασία[62]:

- Ευριστική απόλυτου κατωφλίου (Threshold Heuristic):

$$R(t, t', \{\tau\}) = \begin{cases} \text{Keep}, & \text{if } \overline{\Psi_{t,t'}} \leq \tau \\ \text{Retrain}, & \text{otherwise} \end{cases}$$

- Ευριστική συνολικού κατωφλίου (Cumulative Threshold Heuristic):

$$R(t, t', \{T\}) = \begin{cases} \text{Keep}, & \text{if } \sum_{j=t'+1}^{T} \overline{\Psi_{t,t'}} \leq T \\ \text{Retrain}, & \text{otherwise} \end{cases}$$

, όπου j είναι ο χρόνος της τελευταίας επανεκπαίδευσης

- Περιοδική ευριστική (Periodic Heuristic):

$$R(t, t', \{\phi, \alpha\}) = \begin{cases} \text{Keep}, & \text{if } (t-a) \mod \phi = 0 \\ \text{Retrain}, & \text{otherwise} \end{cases}$$

, όπου φ η περίοδος και α το offset.

6.2.8 Baselines

- **ADWIN**
  Η βασική ιδέα πίσω από το ADWIN[77] είναι η διατήρηση ενός παραθύρου ολίσθησης (sliding window) μεταβλητού μήκους που αποθηκεύει πρόσφατα στοιχεία δεδομένων. Η μέθοδος ADWIN χρησιμοποιεί ένα στατιστικό τεστ (ADWIN change detector) για τη συνεχή παρακολούθηση της απόδοσης ενός μοντέλου σε αυτό το παράθυρο ολίσθησης. Παρακολουθεί δύο υποπαράθυρα εντός του παράθυρου ολίσθησης και υπολογίζει τον μέσο όρο μιας επιλεγμένης μετρικής απόδοσης (π.χ. ακρίβεια, ποσοστό σφάλματος) για κάθε υποπαράθυρο.
  Καθώς καταφθάνουν νέα δεδομένα, το ADWIN προσαρμόζει δυναμικά το μέγεθος των υποπαραθύρων για να διατηρήσει τη στατιστική σημαντικότητα. Όταν υπάρχει σημαντική διαφορά στη μετρική απόδοσης μεταξύ των δύο υπο-παραθύρων, υποδεικνύεται πιθανό concept drift, η μέθοδος ADWIN προσαρμόζεται απορρίπτοντας το παλαιότερο υποπαράθυρο και ενημερώνοντας το μοντέλο με τα πιο πρόσφατα δεδομένα.
  Η ικανότητα προσαρμογής του μεγέθους του παραθύρου με βάση τα εισερχόμενα δεδομένα του καθιστά τη μέθοδο ένα χρήσιμο εργαλείο για το χειρισμό της μετατόπισης εννοιών σε ροές δεδομένων, χωρίς να απαιτούνται προκαθορισμένα κατώτατα όρια ή σταθερά μεγέθη παραθύρων.

- **DDM**
  Η βασική ιδέα της μεθόδου DDM [78] περιλαμβάνει στατιστικό έλεγχο για την ανίχνευση σημαντικών μεταβολών στο ποσοστό σφάλματος ενός ταξινομητή κατά το χειρισμό εισερχόμενων δεδομένων. Διατηρεί ένα ολισθαίνον παράθυρο σταθερού μεγέθους και καταγράφει το παρατηρούμενο ποσοστό σφάλματος εντός αυτού του παραθύρου. Καθώς καταφθάνουν νέες περιπτώσεις, η μέθοδος DDM ενημερώνει το ποσοστό σφάλματος και υπολογίζει την τυπική απόκλιση του σφάλματος.
  Η μέθοδος DDM συγκρίνει δυναμικά το ποσοστό σφάλματος με την εκτιμώμενη τυπική απόκλιση. Όταν η διαφορά μεταξύ του ποσοστού σφάλματος και της εκτιμώμενης τυπικής απόκλισης υπερβαίνει ένα προκαθορισμένο όριο, υποδεικνύοντας μια πιθανή αλλαγή στην κατανομή των δεδομένων (concept drift), το DDM το επισημαίνει ως προειδοποίηση. Εάν η προειδοποίηση εξακολουθεί να υφίσταται και η διαφορά συνεχίζει να αυξάνεται, το DDM το προσδιορίζει ως πραγματικό concept drift και ενεργοποιεί μια ενημέρωση του μοντέλου για να προσαρμοστεί στην αλλαγή των δεδομένων.

- **Markov**
  Η ευρετική μέθοδος λαμβάνει υπόψη μόνο το τρέχον model staleness cost και χρησιμοποιεί την παράμετρο κ ως κατώφλι επανεκπαίδευσης:

$$R(t) = \begin{cases} \text{Keep}, & \text{if } \overline{\Psi_{t,t'}} \leq \kappa \\ \text{Retrain}, & \text{otherwise} \end{cases}$$

- **Never Retrain**
  Καμία επανεκπαίδευση καθ'όλη τη διάρκεια του online evaluation phase.

- **Oracle**
  Ο αλγόριθμος Oracle[62] παίρνει το όνομά του από το γεγονός ότι επιστρέφει τη βέλτιστη λύση χρησιμοποιώντας το σύνολο των online δεδομένων. Επομένως, ο αλγόριθμος Oracle δεν μπορεί να χρησιμοποιηθεί σε κανονικές συνθήκες πειράματος, αλλά έχει σημασία για λόγους αξιολόγησης της στρατηγικής μας με βάση την καλύτερη δυνατή λύση. Χρησιμοποιώντας τον συμπληρωμένο πίνακα δύο διαστάσεων $C[t', t]$ (εκπαίδευση μοντέλων σε όλα τα batches δεδομένων) μπορούμε να βρούμε τη βέλτιστη λύση, επιλύοντας ένα κλασικό πρόβλημα δυναμικού προγραμματισμού δύο μεταβλητών για το συνολικό κόστος στρατηγικής τη χρονική στιγμή t μοντέλου εκπαιδευμένου τελευταία φορά τη χρονική στιγμή t'.

6.2.9 Data Batching

Στα περισσότερα σύνολα δεδομένων, τα επόμενα στιγμιότυπα γράφων κατασκευάζονται με προσθήκες ακμών, οπότε έχουμε γνώση των νέων κόμβων που προστέθηκαν στο γράφημα και των κόμβων που απέκτησαν νέες γείτονες. Παρόλο που η πιο απλή επιλογή θα ήταν να θεωρούμε τους νέους κόμβους μόνο ως 'νέα δεδομένα', αυτό θα ήταν περιοριστικό για τους παρακάτω λόγους:

- Βασιζόμαστε σε έναν αλγόριθμο (GraphSAGE) που εξετάζει την τοπική γειτονιά ενός κόμβου για να παράξει τα embeddings και επομένως το σύνολο των embeddings όλων των κόμβων θα επηρεαστεί εξαιτίας της λογικής μεταβίβασης μηνυμάτων (message passing).
- Μερικές φορές, έχουμε μόνο προσθήκες ακμών μεταξύ υφιστάμενων κόμβων.
- Λαμβάνουμε υπόψη νέους κόμβους για να επεκτείνουμε το πλαίσιο σε γραφήματα με δυναμικές ετικέτες.

Έτσι, για τους λόγους που αναφέρθηκαν παραπάνω, κάθε παρτίδα αποτελείται από όλους τους κόμβους-αναπαραστάσεις του του τρέχονταος στιγμιότυπου γραφήματος. Για περαιτέρω δοκιμές, θα μπορούσαμε επίσης να εξετάσουμε το ενδεχόμενο να προσθέσουμε μια μάσκα για τους νέους κόμβους, αν χρειαστεί να τους διαχωρίσουμε από τους υπάρχοντες.

6.2.10 Queries

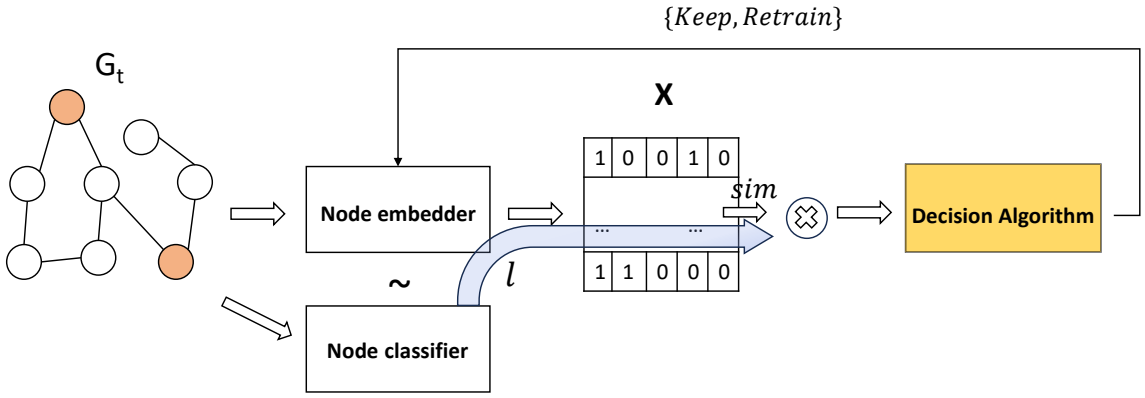
Τα queries (ερωτήματα) δίνονται γενικά από μια τυχαία δειγματοληψία των σημείων δεδομένων (5%), αλλά δοκιμάζονται επίσης πειράματα με στατιστικά ανεξάρτητες ροές ερωτημάτων, ιδίως για υποδειγματικές περιπτώσεις, όπως αναλύεται παρακάτω.

6.2.11 Δήλωση του προβλήματος (Problem Formulation)

Δεδομένης μιας ροής data-query batches, πρέπει να βρούμε μια στρατηγική που να ελαχιστοποιεί το συνολικό κόστος για ολόκληρη την εκτέλεση:

$$C[s] = \sum_{t=1}^{T} C[t]$$

Η εκμάθηση της συνάρτησης απόφασης γίνεται κατά τη διάρκεια μιας offline φάσης βελτιστοποίησης, όπου χρησιμοποιούμε ιστορικά δεδομένα, όπως σε μια αποκλειστική ακολουθία παρτίδων, για να μάθουμε πολύτιμα μοτίβα με τη μορφή κάποιων function-specific parameters. Συγκεκριμένα, χρησιμοποιούμε τρεις διαφορετικές ευριστικές τεχνικές, οι οποίες μαθαίνουν κατώφλια staleness cost για επανεκπαίδευση (απόλυτο και σωρευτικό) και μια περίοδο επανεκπαίδευσης, αντίστοιχα, για την εύρεση του ολικού ελάχιστου κόστους με την υλοποίηση της τεχνικής βελτιστοποίησης διπλής ανόπτησης (Dual Annealing) της βιβλιοθήκης scipy. Η συνολική λογική του framework μας φαίνεται συνοπτικά στο σχήμα 6.2



Σχήμα 6.2: Λειτουργία embedder και αλγορίθμου απόφασης

6.2.12 Αξιολόγηση

Για να βρούμε τη στρατηγική ελάχιστου κόστους, επιλύουμε τον πίνακα κόστους χρησιμοποιώντας την τεχνική του δυναμικού προγραμματισμού και εξετάζουμε επίσης τα data-query accuracy και τον αριθμό των επανεκπαιδεύσεων για να να συγκρίνουμε τη στρατηγική μας με απλούστερα μοντέλα που δεν λαμβάνουν υπόψη το κόστος επανεκπαίδευσης. Σημειώνουμε ότι παρόλο που κατά τη διάρκεια της εκπαίδευσης και της εκτέλεσής μας, τα queries είναι κόμβοι χωρίς ετικέτες, υποθέτουμε ότι ότι έχουν κάποιες κρυφές ετικέτες τις οποίες μπορούμε να χρησιμοποιήσουμε αυστηρά στην τελική αξιολόγηση των αλγορίθμων.

Κεφάλαιο 7

Λεπτομέρειες υλοποίησης

7.1 Περιβάλλον

Καθ' όλη τη διάρκεια των πειραμάτων μας χρησιμοποιούμε την ανοιχτή βιβλιοθήκη για Μηχανική Μάθηση Pytorch. Επίσης, για τον καλύτερο χειρισμό δεδομένων γράφων και για την καλύτερη εκπαίδευση των νευρωνικών δικτύων γράφων βασιζόμαστε στη βιβλιοθήκη **PyG** (PyTorch Geometric)[79].
Δημιουργούμε το περιβάλλον εργασίας μας με την έκδοση **python=3.9.7**, χρησιμοποιώντας το διαχειριστή πακέτων (package manager) **micromamba**. Στη συνέχεια ακολουθούμε τα παρακάτω βήματα εγκατάστασης βιβλιοθηκών:

```
conda install pytorch pytorch-cuda=12.1 -c pytorch -c nvidia
micromamba install pyg -c pyg
micromamba install pytorch-cluster
pip install pyg-lib
micromamba install matplotlib pandas umap-learn seaborn dill ray
```

7.2 Προεπεξεργασία μεγάλου συνόλου δεδομένων (Patent Dataset)

Στην αρχική του μορφή το σύνολο δεδομένων μας αποτελείται από δύο αρχεία, ένα αρχείο με τρεις στήλες που περιέχουν όλες τις ακμές του γράφου με τις αντίστοιχες χρονικές σφραγίδες (timestamps) προσθήκης και ένα αρχείο που περιέχει τα χαρατηριστικά κάθε κόμβου. Κάθε ακμή δίνεται από τα id των δύο κόμβων που την αποτελούν και είναι μοναδικά για κάθε κόμβο-ευρεσιτεχνία. Τα χαρακτηριστικά των ευρεσιτεχνιών παρουσιάζονται στον πίνακα 7.1.

Στον πίνακα 7.2 έχουμε τις αντιστοιχίσεις των υποκατηγοριών σε κατηγορίες, καθώς και το είδος ευρεσιτεχνιών κάθε υποκατηγορίας.

- Ως ετικέτα (label) του κάθε κόμβου επιλέγουμε τη στήλη Cat, δηλαδή έχουμε έξι (6) διαφορετικές κλάσεις.
- Αφαιρούμε τις ακμές των οποίων οι κόμβοι δεν έχουν χαρακτηριστικά και ετικέτες
- Από τις αρχικές στήλες επιλέγουμε αρχικά τις {'PATENT', 'GYEAR', 'GDATE', 'COUNTRY', 'POSTATE', 'NCLASS', 'CAT', 'SUBCAT'}
- Επιλέγουμε να χωρίσουμε το σύνολο των timestamped ακμών σε 25 ισάριθμες παρτίδες.

Attribute Name	Description
PATENT	Patent number
GYEAR	Grant year
GDATA	Grant date, given as the number of days elapsed since January 1, 1960
APPYEAR	Application year (available only for patents granted since 1967)
COUNTRY	Country of first inventor
POSSTATE	State of first inventory (if country is U.S.)
ASSIGNEE	Numeric identifier for assignee (i.e. patent owner)
ASSCODE	One-digit (1-9) assignee type. (The assignee type includes U.S. individual, U.S. government, U.S. organization, non-U.S. individual, etc.)
CLAIMS	Number of claims (available only for patents granted since 1975)
NCLASS	3-digit main patent class
CAT	Technological Category
SUBCAT	Technological Sub-Category

Πίνακας 7.1: Πίνακας χαρακτηριστικών συνόλου δεδομένων ευρεσιτεχνιών

- Χρησιμοποιούμε one-hot encoding για τις κατηγορικές (categorical) μεταβλητές 'COUNTRY', 'POSTATE'

Για τη δημιουργία των παρτίδων δεδομένων θα επεκτείνουμε την κλάση "Data" από τη βιβλιοθήκη torch_geometric.data υλοποιώντας και την κατάλληλη προ-επεξεργασία:

- Αρχικοποιούμε έναν άδειο γράφο και έπειτα για τον αριθμό των παρτιδών προσθέτουμε τον ίδιο αριθμό ακμών
- Επιλέγουμε τη μεγαλύτερη συνεκτική συνιστώσα
- Αρχικοποιούμε το αντικείμενο της παρτίδας ως εξής $Data(edge\_index = edge\_index, x = x, y = y)$, όπου $edge\_index$ το σύνολο των ακμών του γράφου, x τα χαρακτηριστικά του και y η ετικέτα του
- Προσθέτουμε στο αντικείμενο ως χαρακτηριστικό τον αριθμό των κλάσεων $n\_classes$
- Επιλέγουμε να κρατήσουμε 16 από τις 25 παρτίδες για τη φάση της offline βελτιστοποίησης και τις υπόλοιπες 9 για τη φάση της online αξιολόγησης.
- Επιλέγουμε τυχαία το 10% των κόμβων ως queries
- Προσθέτουμε σε κάθε αντικείμενο τις μάσκες που σημειώνουν ποιοι κόμβοι θα θεωρηθούν queries και ποιοι data, ώστε οι πρώτοι να μην ληφθούν υπόψη στο στάδιο της εκπαίδευσης του νευρωνικού δικτύου γράφου (GraphSAGE)

7.3 Κύριος κώδικας

Χρησιμοποιούμε τη standard PyG υπολοίηση του GraphSAGE:

```
class SAGE(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels, out_channels, sigmoid_loss=False,
                                                    gcn_agg=False,p=0.5):
        super().__init__()
        self.convs = torch.nn.ModuleList()
        if gcn_agg:
            agg = GCNConv
            self.convs.append(agg(in_channels, hidden_channels))
        else:
            agg = SAGEConv
            self.convs.append(agg(in_channels, hidden_channels))
        self.fc = nn.Linear(hidden_channels, out_channels)
        self.sigmoid_loss = sigmoid_loss
        self.dropout = torch.nn.Dropout(p=p)

    def forward(self, x, edge_index):
        for i, conv in enumerate(self.convs):
            x = conv(x, edge_index)
            if i < len(self.convs):
                x = x.relu_()
                x = self.dropout(x)
        return self.fc(x)

    @torch.no_grad()
    def embed(self, x_all, subgraph_loader):

        device = x_all.device
        for i, conv in enumerate(self.convs):
            xs = []
            for batch in subgraph_loader:
                x = x_all[batch.n_id.to(x_all.device)].to(device)
                x = conv(x, batch.edge_index.to(device))
                if i < len(self.convs):
                    x = x.relu_()

                xs.append(x[:batch.batch_size].cpu())
            x_all = torch.cat(xs, dim=0)
        return x_all
```

Σχόλια:

- Όπως προείπαμε χρησιμοποιούμε μόνο ένα συνελικτικό layer και στη συνέχεια ένα πλήρως συνεκτικό.
- Η υλοποίηση παρέχει τη δυνατότητα και για τη χρήση συνελικτικών δικτύων τύπου GCN, αλλά εμείς χρησιμοποιούμε συνελικτικά δίκτυα GraphSAGE, οπότε και $gcn\_agg = False$.
- Το αντικείμενο subgraph_loader της κλάσης NeighborLoader είναι ο φορτωτής δεδομένων (data loader) που φορτώνει τους γείτονες του κάθε κόμβου στη δημιουργία των παρτιδών για την εκπαίδευση με mini-batching και ουσιαστικά επιστρέφει έναν υπογράφο.

Για την εκπαίδευση των μοντέλων σε κάθε παρτίδα δεδομένων επιλέγουμε τις εξής τιμές παραμέτρων:

- $epochs = 10$, ο αριθμός των εποχών εκπαίδευσης
- $lr = 5e - 1$, το βήμα μάθησης

- $hidden = 512$, το μέγεθος του κρυφού λαψερ
- $batch_{size} = 512$, ο αριθμός κόμβων δέσμης κατά την εκπαίδευση

7.4 Hardware

Τα πειράματα εκτελέστηκαν σε μηχάνημα Linux με 32G RAM και για την πιο κοστοβόρα εκπαίδευση των νευρωνικών δικτύων έγινε επιτάχυνση σε κάρτες γραφικών p100 VRAM: 16GB, a100 VRAM: 80GB, ανάλογα με τη διαθεσιμότητα στο cluster.

Cat	SubCat	SubCatName	CatNameShort	CatNameLong
1	11	Agriculture, Food, Textiles	Chemical	Chemical
1	12	Coating	Chemical	Chemical
1	13	Gas	Chemical	Chemical
1	14	Organic Compounds	Chemical	Chemical
1	15	Resins	Chemical	Chemical
1	19	Miscellaneous-Chemical	Chemical	Chemical
2	21	Communications	Cmp&Cmm	Computers & Communications
2	22	Computer Hardware & Software	Cmp&Cmm	Computers & Communications
2	23	Computer Peripherals	Cmp&Cmm	Computers & Communications
2	24	Information Storage	Cmp&Cmm	Computers & Communications
3	31	Drugs	Drgs&Med	Drugs & Medical
3	32	Surgery & Med Inst.	Drgs&Med	Drugs & Medical
3	33	Biotechnology	Drgs&Med	Drugs & Medical
3	39	Miscellaneous-Drgs&Med	Drgs&Med	Drugs & Medical
4	41	Electrical Devices	Elec	Electrical & Electronic
4	42	Electrical Lighting	Elec	Electrical & Electronic
4	43	Measuring & Testing	Elec	Electrical & Electronic
4	44	Nuclear & X-rays	Elec	Electrical & Electronic
4	45	Power Systems	Elec	Electrical & Electronic
4	46	Semiconductor Devices	Elec	Electrical & Electronic
4	49	Miscellaneous-Elec	Elec	Electrical & Electronic
5	51	Mat. Proc & Handling	Mech	Mechanical
5	52	Metal Working	Mech	Mechanical
5	53	Motors & Engines + Parts	Mech	Mechanical
5	54	Optics	Mech	Mechanical
5	55	Transportation	Mech	Mechanical
5	59	Miscellaneous-Mechanical	Mech	Mechanical
6	61	Agriculture, Husbandry, Food	Others	Others
6	62	Amusement Devices	Others	Others
6	63	Apparel & Textile	Others	Others
6	64	Earth Working & Wells	Others	Others
6	65	Furniture, House Fixtures	Others	Others
6	66	Heating	Others	Others
6	67	Pipes & Joints	Others	Others
6	68	Receptacles	Others	Others
6	69	Miscellaneous-Others	Others	Others

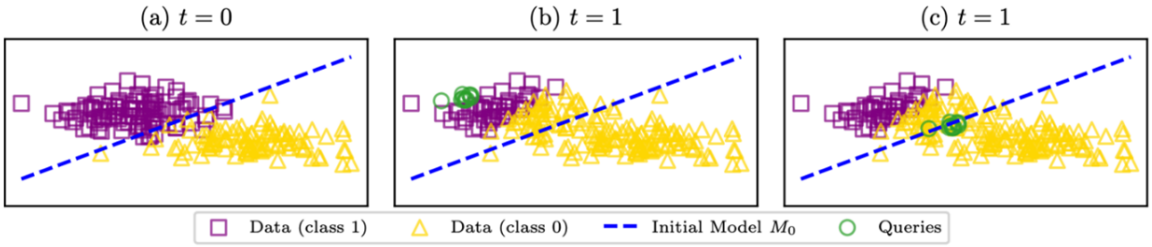
Πίναχας 7.2: Category and Subcategory Data

Κεφάλαιο 8

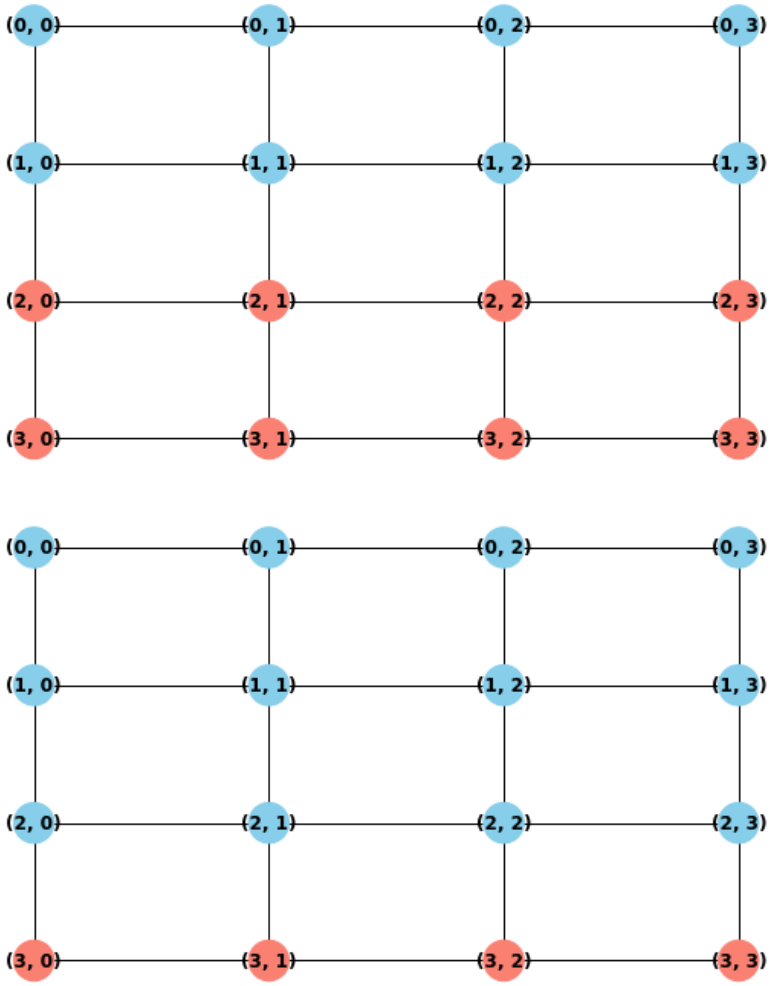
Πειραματικό Μέρος

8.1 Παραδειγματικές περιπτώσεις - Συνθετικά δεδομένα

Για να μελετήσουμε κάποιες υποδειγματικές περιπτώσεις και πώς πρέπει να συμπεριφέρεται η στρατηγική μας σε αυτές πρέπει να χρησιμοποιήσουμε συνθετικά δεδομένα προσαρμοσμένα στα αντίστοιχα σενάρια. Για το σκοπό αυτό, θα πειραματιστούμε κυρίως με γραφήματα πλέγματος (lattice graphs), όπου μπορούμε να ελέγξουμε και να δοκιμάσουμε κάθε διαφορετικό σενάριο και κάθε φορά σταδιακά να χτίζουμε μεγαλύτερη πολυπλοκότητα σε αυτό.
Συγκεκριμένα, αρχικά μελετάμε την περίπτωση στην οποία η υπό συνθήκη κατανομή των ετικετών δεν αλλάζει (δεν υπάρχει concept drift) - έχουμε περίπου τον ίδιο αριθμό λανθασμένων ταξινομήσεων κοντά στο decision boundary. Με αυτόν τον τρόπο, χρησιμοποιώντας το relative model staleness cost, το οποίο μετράει τη διαφορά μεταξύ του staleness δύο παρτίδων δεδομένων, παίρνουμε ένα χαμηλό misclassification cost, λιγότερο πιθανό να προκαλέσει απόφαση επανεκπαίδευσης, ανεξάρτητα από την κατανομή του ερωτήματος. Για να ελέγξουμε αυτό το σενάριο άμεσα, τροφοδοτούμε το μοντέλο μας με το ίδιο στιγμιότυπο γραφήματος σε κάθε παρτίδα και ένα σύνολο ερωτημάτων που μπορεί να έχουν την ίδια κατανομή (χωρίς covariate drift) ή διαφορετική ακόμη και για να επικυρώσουμε ότι η στρατηγική μας λαμβάνει μόνο αποφάσεις "Keep".
Μια σημαντική σημείωση εδώ που εξηγεί την απόφαση μοντελοποίησης να χρησιμοποιήσουμε το σχετικό κόστος παλαιότητας είναι ότι αν χρησιμοποιούσαμε ένα absolute staleness cost, στο σενάριο όπου δεν υπάρχει μετατόπιση δεδομένων αλλά η κατανομή των ερωτημάτων αλλάζει (ενδεχομένως μετακινείται σε περιοχές εσφαλμένης ταξινόμησης), θα θα μπορούσαμε να έχουμε περιττές αποφάσεις "Retrain" που δεν θα αύξαναν την ακρίβεια του μοντέλου μας (αφού τα δεδομένα εκπαίδευσης παραμένουν ίδια), αλλά απλώς θα σπαταλούσαν πόρους.
Μια αναπόσπαστη πρόκληση στη δοκιμή σημαντικών περιπτώσεων που δείχνουν πώς η στρατηγική μας μπορεί να μάθει από δεδομένα είναι το πώς ακριβώς μπορούμε να επιλέξουμε ερωτήματα σε ελεγχόμενες αποστάσεις από το όριο απόφασης ή, ακριβέστερα, με ελεγχόμενο model staleness cost. Και πάλι, μπορούμε να κατανοήσουμε καλύτερα την κατανομή του staleness cost χρησιμοποιώντας ένα lattice graph και να επιλέξουμε εκ των υστέρων τα επιθυμητά ερωτήματα.
Η δεύτερη περίπτωση που πρέπει να μελετήσουμε είναι όταν υπάρχει concept drift μεταξύ διαδοχικών παρτίδων δεδομένων, δηλαδή αλλάζει η κατανομή πιθανότητας $p(y|x)$ (Σχήμα 8.1).
Ένας απλός τρόπος για να μοντελοποιήσουμε αυτή τη συμπεριφορά είναι να διατηρήσουμε το ίδιο lattice graph (ίδια χαρακτηριστικά) και μόνο ολισθαίνοντας το όριο των ετικετών κατά μήκος ή πλάτος (Σχήμα 8.2).



Σχήμα 8.1: Concept drift [62]



Σχήμα 8.2: Sliding labels - lattice graph [62]

8.2 Real world dataset

Το σύνολο δεδομένων που χρησιμοποιούμε είναι το Δίκτυο παραπομπών διπλωμάτων ευρεσιτεχνίας (Patent citation network) [80]. Το σύνολο δεδομένων για τα διπλώματα ευρεσιτεχνίας των ΗΠΑ συντηρείται από το Εθνικό Γραφείο Οικονομικών Ερευνών. Τα δεδομένα καλύπτουν

37 έτη (1 Ιανουαρίου 1963 έως 30 Δεκεμβρίου 1999) και περιλαμβάνουν όλα τα διπλώματα ευρεσιτεχνίας χρησιμότητας, που χορηγήθηκαν κατά τη διάρκεια αυτής της περιόδου, συνολικά 3.923.922 διπλώματα ευρεσιτεχνίας. Το γράφημα παραπομπών περιλαμβάνει όλες τις αναφορές που έγιναν από διπλώματα ευρεσιτεχνίας που χορηγήθηκαν μεταξύ 1975 και 1999, συνολικά 16.522.438 αναφορές. Για το σύνολο δεδομένων των διπλωμάτων ευρεσιτεχνίας υπάρχουν 1.803.511 κόμβοι για τους οποίους δεν έχουμε πληροφορίες σχετικά με τις παραπομπές τους (έχουμε μόνο τους εσωτερικούς συνδέσμους).

Πίνακας 8.1: Dataset Statistics

Metric	**Value**
Nodes in the dataset	3,774,768
Edges in the dataset	16,518,948
Nodes in largest WCC	3,764,117 (0.997)
Edges in largest WCC	16,511,741 (1.000)
Nodes in largest SCC	1 (0.000)
Edges in largest SCC	0 (0.000)
Average clustering coefficient	0.0757
Number of triangles	7,515,023
Fraction of closed triangles	0.02343
Diameter (longest shortest path)	22
90-percentile effective diameter	9.4

Source: https://snap.stanford.edu/data/cit-Patents.html

Συνοπτικά χρησιμοποιούμε για τις ανάγκες της έρευνάς μας:

- Πλέγματα (Lattices)
- Caveman Graphs (βλέπε επόμενη ενότητα)
- Patent citation network

8.3 Διερευνητική ανάλυση μοντέλου Graph-SAGE (Exploratory Analysis)

Για να μάθουμε πώς λειτουργεί το GraphSAGE σε διάφορα σύνολα δεδομένων και να εντοπίσουμε πιθανούς περιορισμούς, το δοκιμάζουμε σε δυναμικά γραφήματα που αναπτύσσονται πιθανοτικά με διαφορετικούς τρόπους και αναφέρουμε τα συνολικά δεδομένα ακρίβειας (χωρίς αναφορά ακόμα για staleness cost).
Για τα πειράματά μας, θα χρησιμοποιήσουμε γράφους caveman, οι οποίοι αποτελούνται από ένα αριθμό κλικών (cliques), όπου μια ακμή από κάθε κλίκα επανασυνδέεται με μια άλλη, έτσι ώστε να είναι όλες συνδεδεμένες κυκλικά. Η ύπαρξη πυκνών συστάδων κόμβων μας βοηθάει να απεικονίσουμε καλύτερα τα embeddings που μαθαίνουμε από το GraphSAGE και να είμαστε σίγουροι ότι μπορούμε να ξεκινήσουμε από ένα τέλειο διαχωρίσιμο σύνολο κόμβων.
Αρχικοποιούμε όλους τους κόμβους έτσι ώστε όλα τα μέλη της κλίκας να μοιράζονται το ίδιο διάνυσμα χαρακτηριστικών (one-hot κωδικοποίηση) και ετικέτα, τα οποία είναι διαφορετικά για κάθε κλίκα. Στη συνέχεια πειραματιζόμαστε με διαφορετικούς τρόπους προσθήκης νέων κόμβων για έναν αριθμό (4 στα παρακάτω σχήματα) παρτίδων. Σε κάθε επανάληψη, για κάθε κλίκα

i και για έναν αριθμό N προστιθέμενων κόμβων (που συνδέονται με κάθε υπάρχοντα κόμβο της κλίκας), με πιθανότητα:

- p_1 , προσθέτουμε έναν κόμβο στην κλίκα i με το ίδιο χαρακτηριστικό και την ίδια ετικέτα
- p_2 , προσθέτουμε έναν κόμβο στην κλίκα i με το ίδιο χαρακτηριστικό αλλά διαφορετική ετικέτα
- p_3 , προσθέτουμε έναν κόμβο σε μια κλίκα i με διαφορετικό χαρακτηριστικό αλλά την ίδια ετικέτα

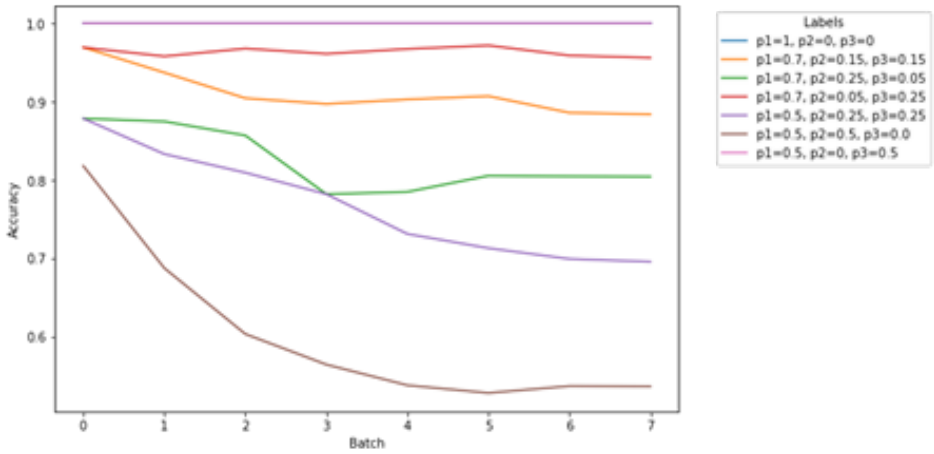
Ουσιαστικά, με τις περιπτώσεις αυτές επιθυμούμε να μελετήσουμε την απόδοση του μοντέλου σε διαφορετικά επίπεδα ομοφιλίας (homophily) στο γράφο.

Ορισμός (Homophily). *Συνδεδεμένοι κόμβοι έχουν παρόμοια χαρακτηριστικά και ετικέτες.*

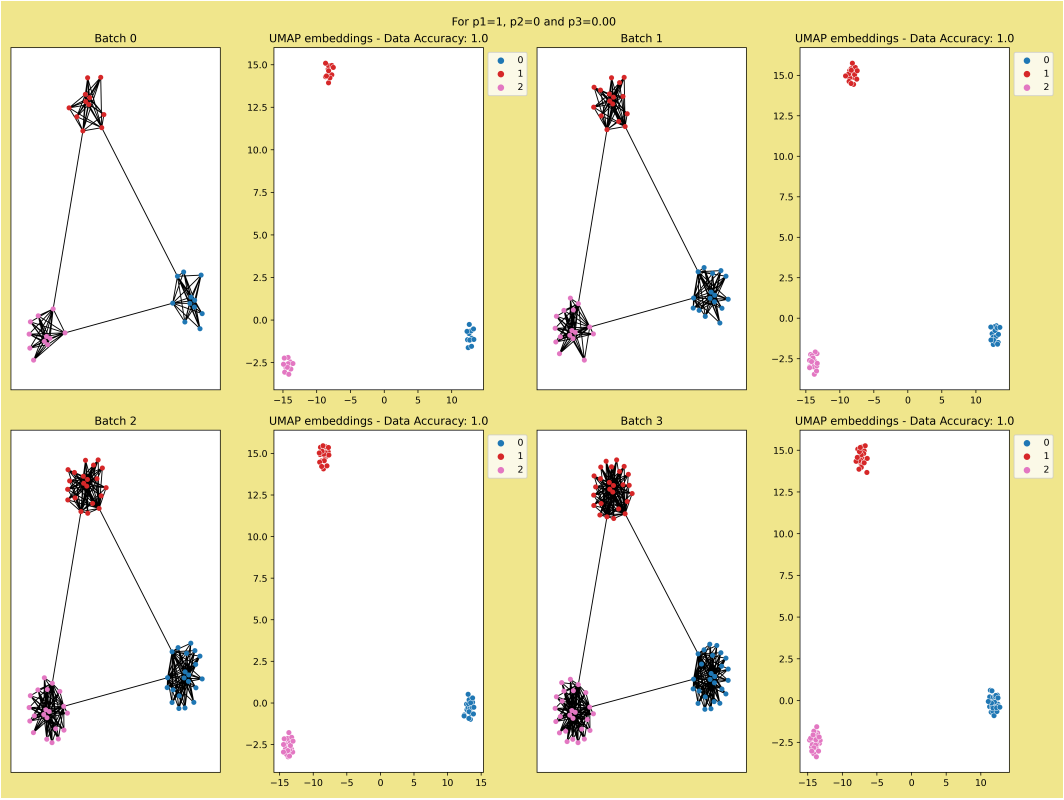
Έχει αποδειχθεί ότι τα συνελικτικά δίκτυα γράφων έχουν μειωμένη απόδοση σε καθεστώς heterophily σε στατικούς όμως γράφους, οπότε και θέλουμε να μελετήσουμε τη συμπεριφορά τους σε δυναμικούς γράφους.
Αρχικά, αντιμετωπίσαμε το πρόβλημα της αλλαγής προσανατολισμού των embeddings σε δύο διαστάσεις για κάθε batch, οπότε χρησιμοποιήσαμε τη μέθοδο AlignedUMAP για να βελτιστοποιήσουμε όλα τα embeddings ταυτόχρονα και να απεικονίσουμε καλύτερα πώς οι ενσωματώσεις αλλάζουν κατά την ανάπτυξη του γράφου.

> Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualisation similarly to t-SNE, but also for general non-linear dimension reduction. The algorithm is founded on three assumptions about the data
>
> - The data is uniformly distributed on Riemannian manifold;
> - The Riemannian metric is locally constant (or can be approximated as such);
> - The manifold is locally connected.[81]

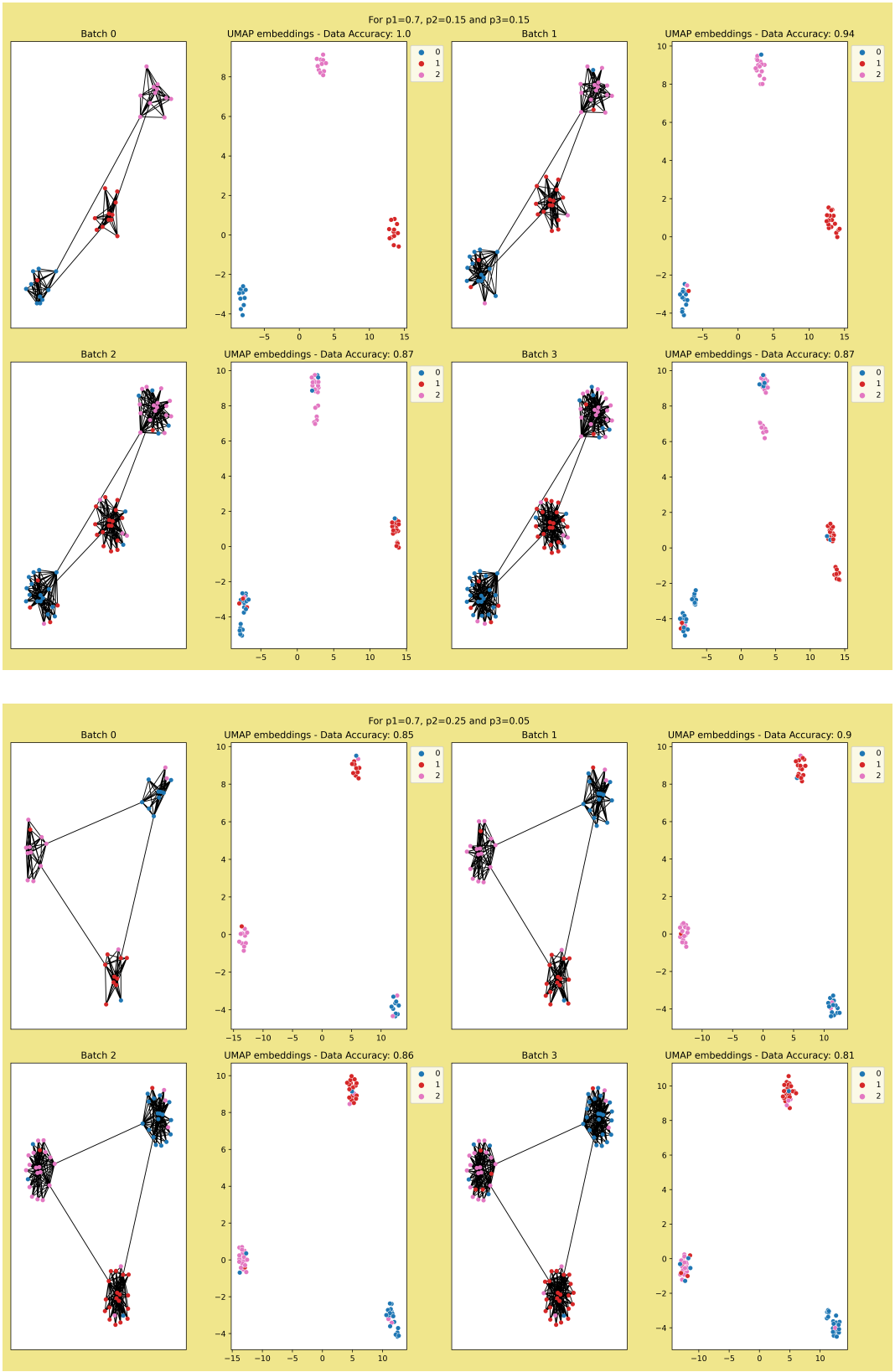
Στα σχήματα 8.4-8.7 παρουσιάζουμε για διαφορετικές τιμές των p1, p2 και p3 ακολουθίες τεσσάρων στιγμιότυπων του γράφου, μαζί με τις προκύπτουσες αναπαραστάσεις και το data accuracy. Σημειώνουμε ότι πραγματοποιείται επανεκπαίδευση σε κάθε παρτίδα για να βρούμε την κατά προσέγγιση μέγιστη ακρίβεια που μπορεί να επιτύχει το GraphSAGE.
Μπορούμε να δούμε από τις παραπάνω περιπτώσεις ότι το GraphSAGE μπορεί να αποδώσει καλά για υψηλότερες τιμές του $p1$ και $p3$, αλλά γρήγορα υποβαθμίζει την ακρίβειά του για υψηλότερες τιμές του $p2$, επειδή προβλέπει τις ετικέτες των κόμβων συγκεντρώνοντας μόνο τα χαρακτηριστικά των γειτονικών κόμβων. Σημειώστε ότι έχουμε ορίσει το βάθος σε ένα- διαφορετικά, οι πληροφορίες θα διαδίδονταν από όλες τις κλίκες. Τέλος, μπορούμε να παρατηρήσουμε ότι παρόλο που οι υψηλότερες τιμές του $p3$ εξακολουθούν να οδηγούν σε τέλεια ακρίβεια, η ομαδοποίηση των αναπαραστάσεων δεν είναι τόσο ομοιογενής σε σύγκριση με τις καθαρές προσθήκες στην πρώτη περίπτωση ($p1 = 1$), όπου όλες οι κλίκες διατηρούν τα ίδια χαρακτηριστικά και τις ίδιες ετικέτες μεταξύ των κόμβων τους καθ΄ όλη τη διάρκεια του πειράματος.
Τώρα, εξετάζοντας την αντίθετη περίπτωση, όπου η ευρετική μας απόφαση θα είχε μόνο "Keep" αποφάσεις, παρουσιάζουμε στο σχήμα 8.3 πώς η ακρίβεια του αρχικού μοντέλου μεταβάλλεται (degradation) σε αυτά τα διαφορετικά σενάρια δυναμικά αναπτυσσόμενων γραφημάτων. Μια



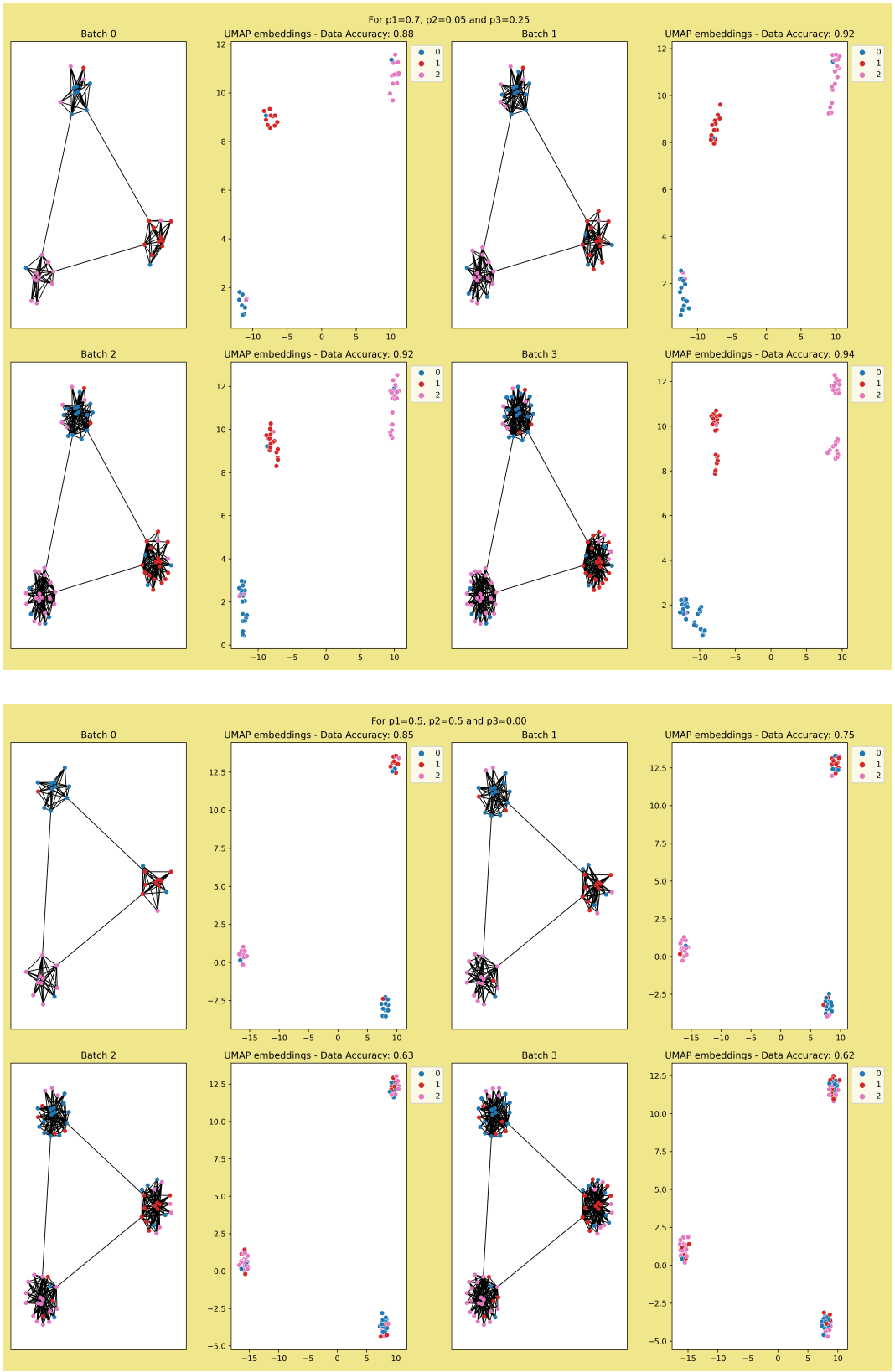
Σχήμα 8.3: Caveman Graph comparative



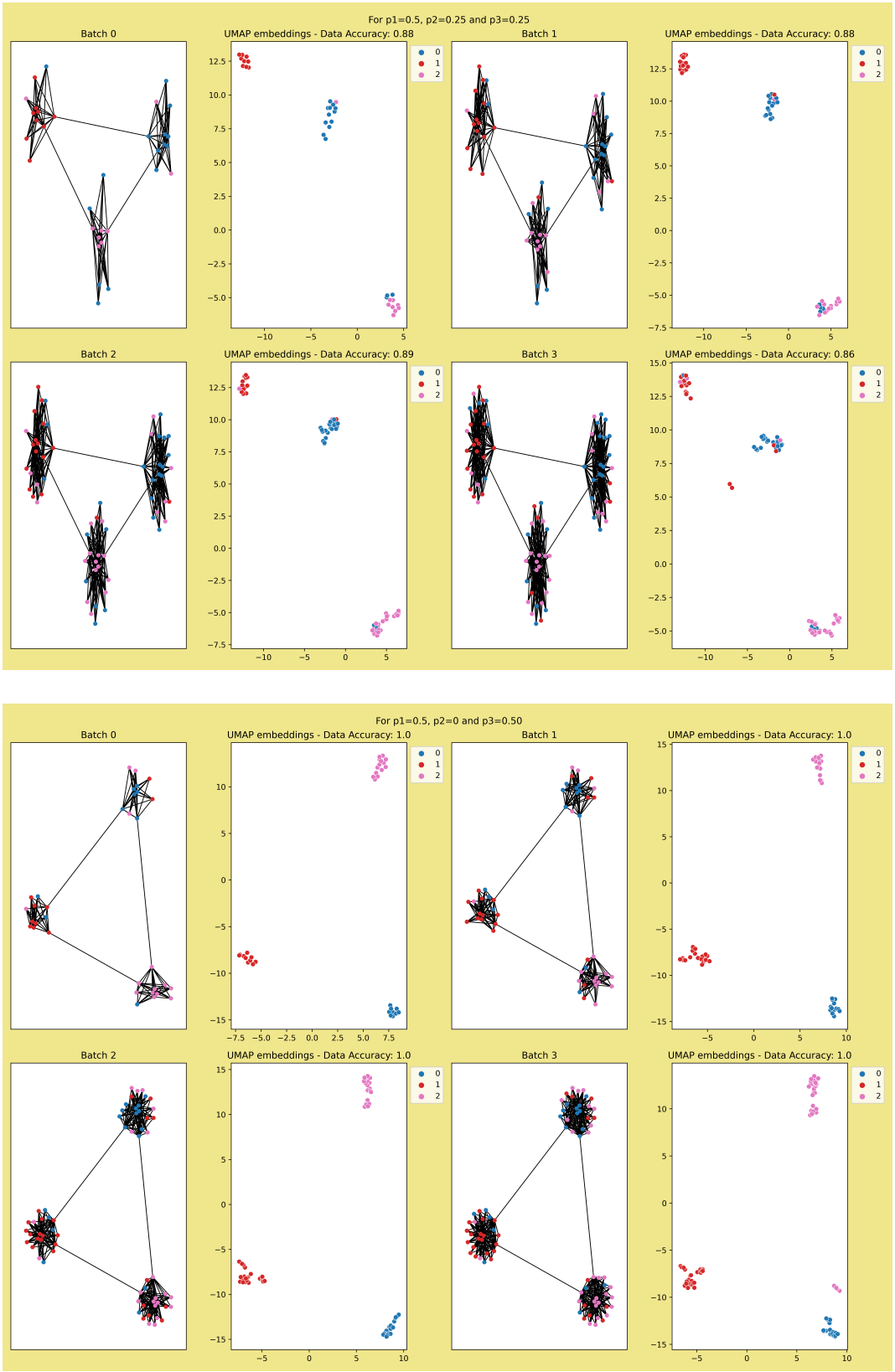
Σχήμα 8.4: Caveman Graph I



Σχήμα 8.5: Caveman Graph II



Σχήμα 8.6: Caveman Graph III



Σχήμα 8.7: Caveman Graph IV

ενδιαφέρουσα παρατήρηση είναι ότι όταν $p2 = 0$, το GraphSAGE μπορεί να διατηρήσει ένα τέλειο σκορ ακρίβειας ακόμη και όταν δεν επανεκπαιδεύεται και αντίστροφα με υψηλότερες τιμές του $p2$ έχουμε εκθετική μείωση της ακρίβειας. Το γεγονός αυτό αναδεικνύει λοιπόν την ανάγκη ικανότητας επανεκπαίδευσης σε ένα online σύστημα και αναδεικνύει την μειωμένη απόδοση συνελικτικών δικτύων γράφων σε αυξημένο επίπεδο heterophily ως ουσιαστικά ύπαρξη data drift.

8.4 Πειράματα σε συνθετικά δεδομένα

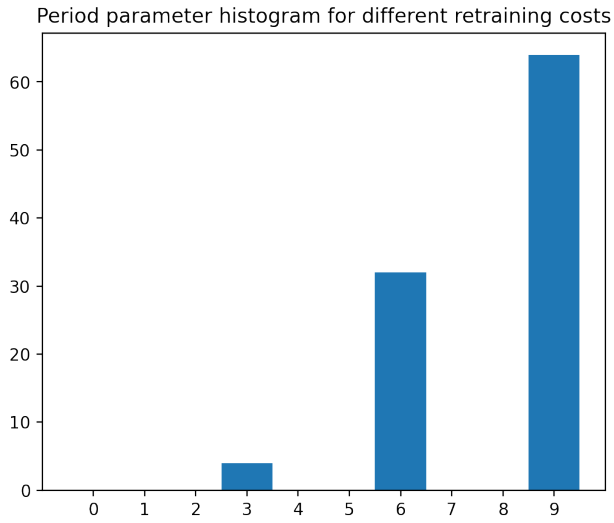
Στη συνέχεια δοκιμάζουμε μερικά σενάρια πλήρους πειράματος πάνω σε συνθετικά δεδομένα, ώστε να επιβεβαιώσουμε τις υποθέσεις μας και να προχωρήσουμε τελικά σε πειράματα μεγάλης κλίμακας.

- **Ίδια δεδομένα σε κάθε παρτίδα**
  Οι αλγόριθμοί μας αναμένεται να μάθουν ότι δεν υπάρχει καμία αλλαγή μεταξύ των offline παρτίδων και να εξάγουν αποφάσεις "Keep" σε κάθε χρονικό βήμα εκτός ενδεχομένως από όταν έχουμε μηδενικό επανεκπαίδευση κόστος.
  Αυτό είναι πράγματι αλήθεια. Για 6 διαφορετικούς αλγορίθμους (συμπεριλαμβανομένου του Oracle) και 100 διαφορετικά κόστη επανεκπαίδευσης έχουμε μόνο μία περίπτωση με τουλάχιστον μία απόφαση "Retrain" για την παραλλαγή Periodic και μηδενικό κόστος επανεκπαίδευσης.
  Αυτό το αποτέλεσμα είναι ιδιαίτερα σημαντικό, καθώς η λύση του δυναμικού προγραμματισμού κατά τη φάση της offline βελτιστοποίησης συγκρίνει τα staleness costs των διαφορετικών μοντέλων που εκπαιδεύονται σε διαφορετικές παρτίδες δεδομένων (δηλαδή σε διαφορετικές χρονικές σφραγίδες), θα ήταν δυνατό να μάθουν διαφορετικά embedding mappings, ακόμη και όταν εκπαιδεύονται στα ίδια δεδομένα και συνεπώς να παράγουν διαφορετικά staleness costs, οδηγώντας σε αστάθεια.
  Η διασφάλιση της συνέπειας του αλγορίθμου μας σε διαφορετικές παρτίδες δεδομένων είναι αναπόσπαστο στοιχείο για επιτυχία της τεχνικής μας και μας επιτρέπει να έχουμε σταθερά και 'ανθεκτικά στην περιστροφή' αποτελέσματα.

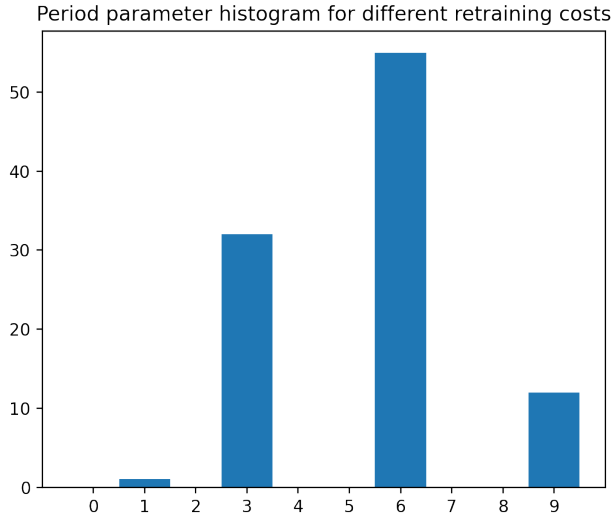
- **Περιοδικό concept drift**
  Στο παρόν πείραμα εισάγαμε concept drift στο γράφημά μας με περιοδικό τρόπο για να ελέγξουμε αν η περιοδική ευριστική μέθοδος θα μπορούσε να συμπεράνει σωστά την παράμετρο περιόδου για μια απόφαση επανεκπαίδευσης.
  Για παράδειγμα, στο παρακάτω πείραμα διατηρούμε το γράφημα ίδιο και εισάγουμε concept drift μόνο κάθε τρία χρονικά βήματα. Όπως μπορούμε να δούμε στο σχήμα 8.8, η περιοδική παραλλαγή μπορεί να μάθει αυτό το μοτίβο και να διαπιστώσει με επιτυχία ότι πρέπει να γίνει επανεκπαίδευση σε πολλαπλάσια των τριών παρτιδών.

  Μπορούμε επίσης να δούμε ότι στις περισσότερες περιπτώσεις έχουμε αποφάσεις "Retrain" μετά από έξι ή εννέα χρονικά βήματα, αλλά αυτό εξαρτάται από το επίπεδο του concept drift, το οποίο σε αυτή την περίπτωση δεν ήταν σημαντικό και επειδή με την αύξηση του κόστους επανεκπαίδευσης είναι λογικό ότι η παραλλαγή θα τείνει να επιλέγει μεγαλύτερα πολλαπλάσια της περιόδου.
  Για παράδειγμα, για πιο σημαντικό concept drift το ιστόγραμμα αλλάζει σε σε συμφωνία με την υπόθεσή μας (υπάρχει μόνο μία περίπτωση όπου η περίοδος που μαθαίνεται είναι 1, όταν το κόστος επανεκπαίδευσης είναι μηδενικό), όπως φαίνεται και στο σχήμα 8.9.

- **Ίδιο γράφημα (πλέγματος) - ίδια χαρακτηριστικά και ετικέτες - διαφορετικά queries**
  Για αυτό το πείραμα χρησιμοποιούμε το ίδιο γράφημα πλέγματος σε κάθε παρτίδα δεδομένων, αλλά χρησιμοποιούμε μια γραμμή ως query set, ξεκινώντας από την κορυφή και ολισθαίνοντας προς τα κάτω (στην περίπτωση που επιλέξαμε το decision boundary χωρίζει το γράφημα πλέγματος οριζόντια). Για να το κάνουμε αυτό, προσθέτουμε θόρυβο στο γράφημά μας από μια κατανομή Gauss με επίκεντρο τη μεσαία σειρά του πλέγματος και



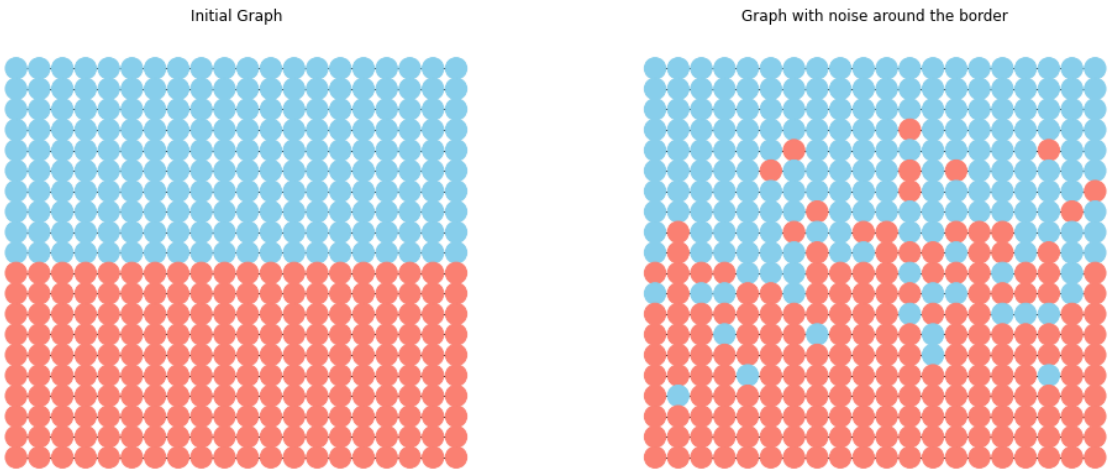
Σχήμα 8.8: Periodic concept drift (low)



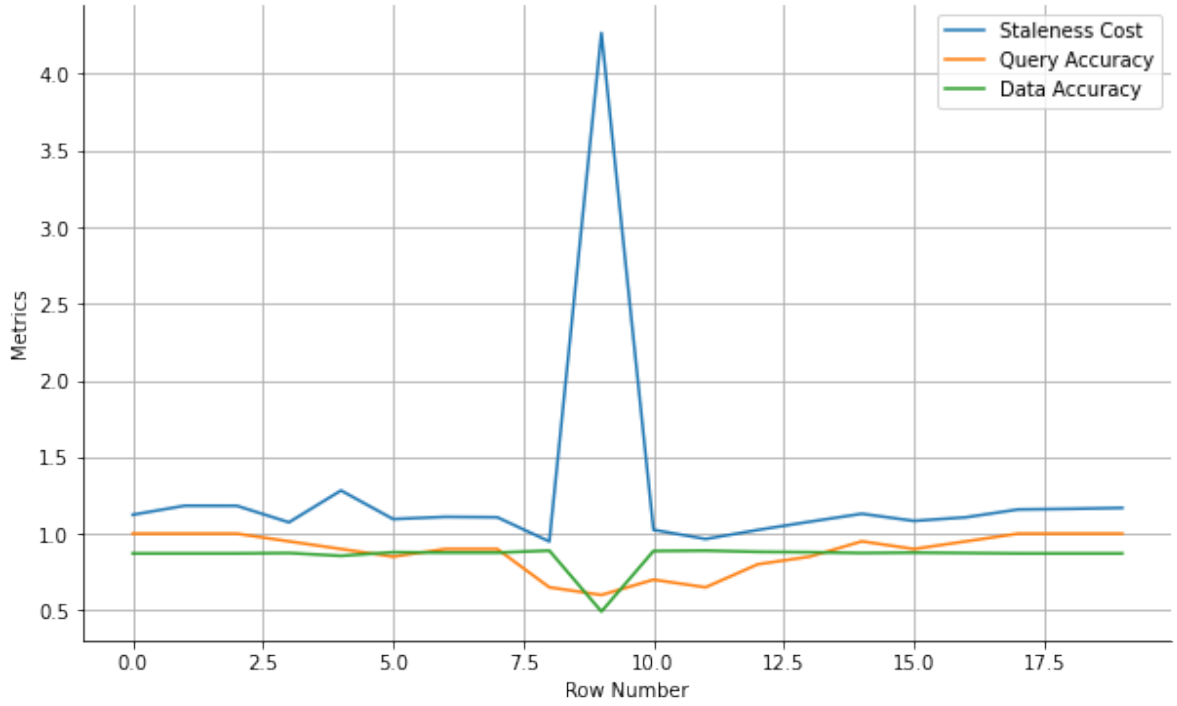
Σχήμα 8.9: Periodic concept drift (high)

έτσι μπορούμε να πάρουμε μια σειρά από αυξανόμενο και στη συνέχεια μειούμενο staleness cost, το οποίο επαληθεύουμε ξεχωριστά εκπαιδεύοντας το μοντέλο με κάθε γραμμή του συνόλου δοκιμών.

Όπως μπορούμε να δούμε στο σχήμα 8.12 οι εκπαιδευμένοι αλγόριθμοί μας είναι σε θέση να διατηρούν χαμηλό αριθμό επανεκπαιδεύσεων ανεξάρτητα από το κόστος επανεκπαίδευσης και συμπεριφέρονται όπως θα έπρεπε σε απουσία concept drift. Ακόμη και με concept drift λόγω των queries που πλησιάζουν το όριο απόφασης μια επανεκπαίδευση δεν θα είχε ως αποτέλεσμα καλύτερη ακρίβεια του μοντέλου, καθώς θα είχε εκπαιδευτεί σε παρόμοια δεδομένα.



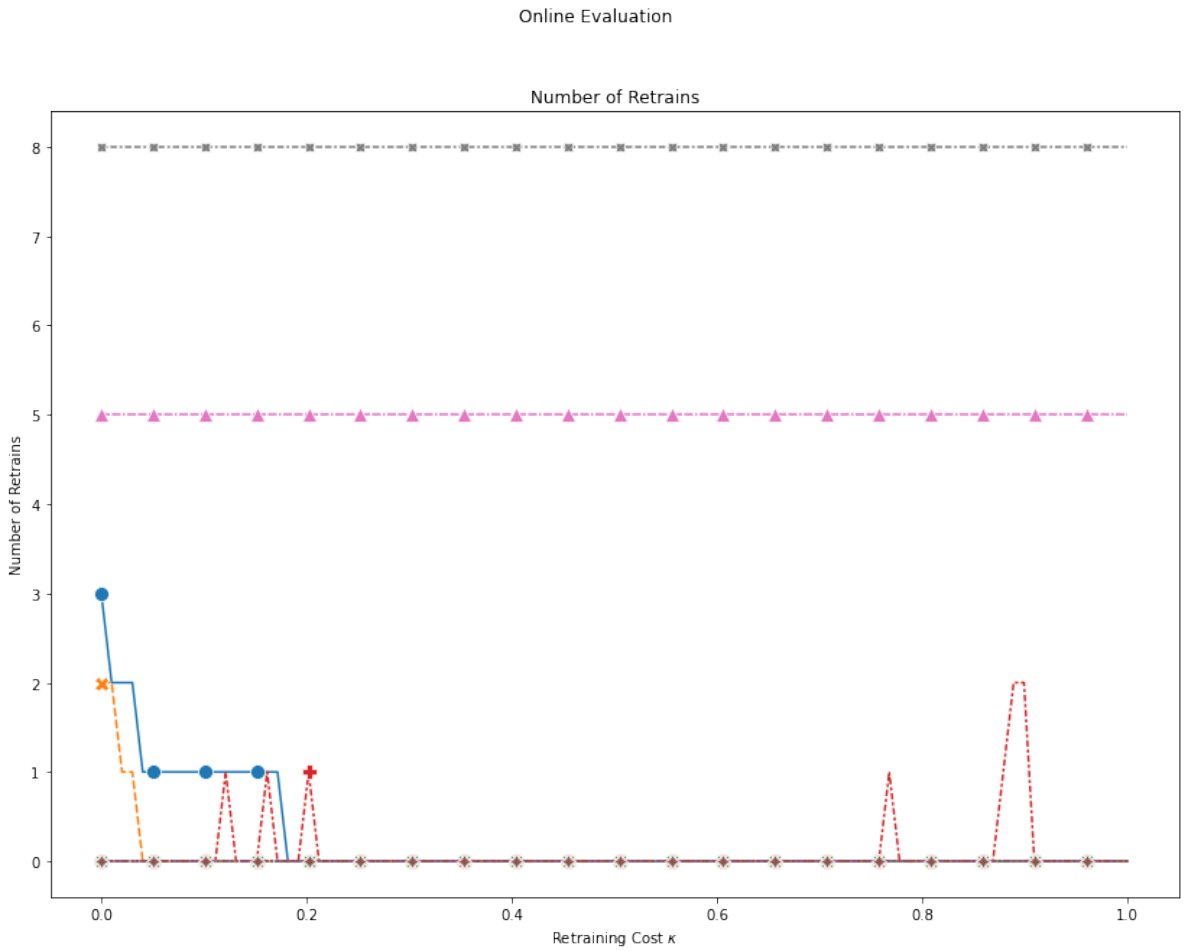
Σχήμα 8.10: Noisified lattice



Σχήμα 8.11: Corresponding staleness and accuracies

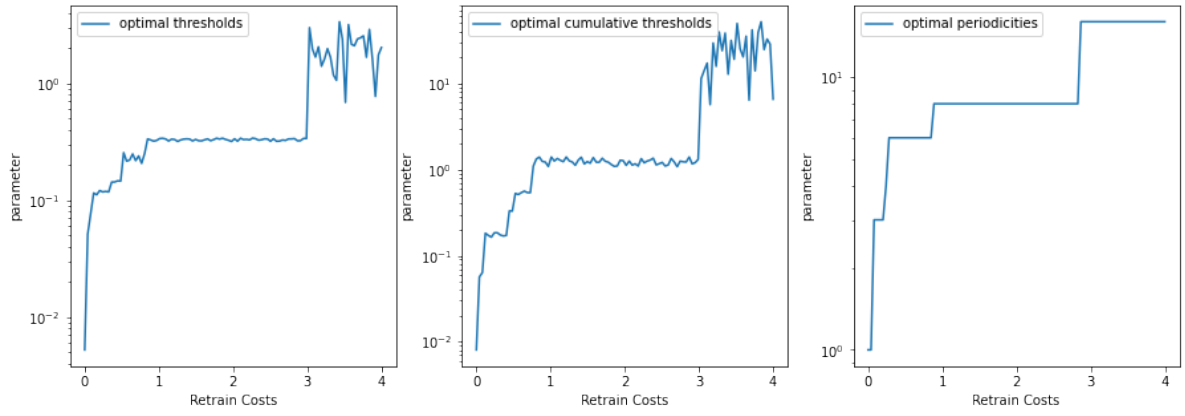
8.5 Σύνολο δεδομένων πραγματικού κόσμου

Αποτελέσματα:
Πρώτον, όπως μπορούμε να δούμε, αυξάνοντας το κόστος επανεκπαίδευσης, υπάρχει μια αρχική απότομη αύξηση για τα thresholds που μαθαίνουν οι αλγόριθμοί μας και στη συνέχεια ακολουθούν κυρίως μια γραμμική συνολική αύξηση. Αυτό επιβεβαιώνει την υπόθεσή μας ότι το χαμηλό κόστος επανεκπαίδευσης θα πρέπει να αξιοποιείται από τον αλγόριθμό μας και να προκαλεί περισσότερες, φτηνές επανεκπαιδεύσεις και έτσι να μαθαίνει χαμηλά thresholds κατά



Σχήμα 8.12: Same data - increasing staleness

τη διάρκεια της βελτιστοποίησης. Σε αντίθεση, όταν το κόστος επανεκπαίδευση αυξάνεται, ο αλγόριθμός μας μαθαίνει σωστά υψηλότερα thresholds, πράγμα που σημαίνει ότι τα staleness costs θα πρέπει επίσης να αυξηθούν σημαντικά, προκειμένου να προκαλέσουν μια δαπανηρή επανεκπαίδευση. Τα παραπάνω γίνονται διακριτά στο Σχήμα 8.13.

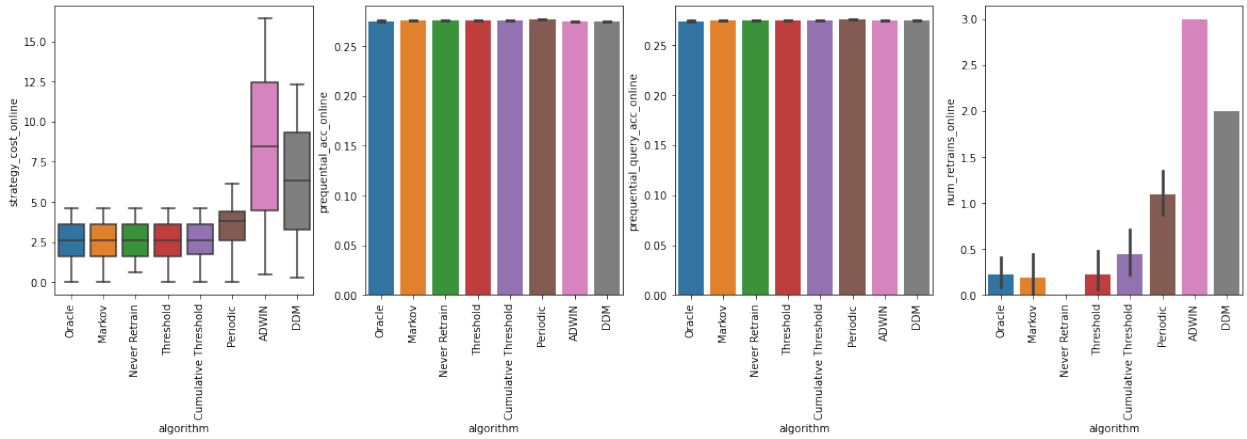


Σχήμα 8.13: Retraining cost and learned thresholds

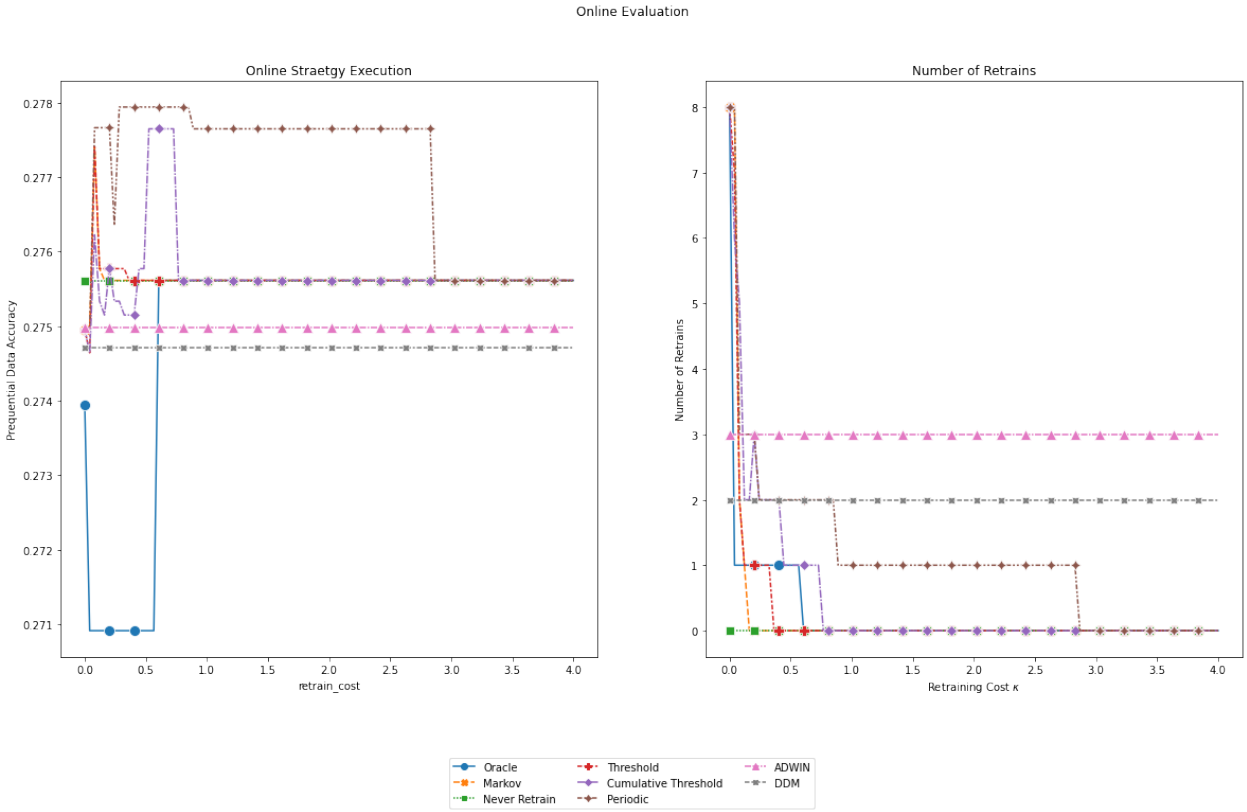
Online Μετρικές αξιολόγησης:
Η προγενέστερη (**prequential**) ακρίβεια αντιπροσωπεύει τη μέση ακρίβεια για όλη τη διάρκεια εκτέλεσης του μοντέλου μας. Στη περίπτωση μιας απόφασης Retrain σε χρονική στιγμή t υπολογίζουμε πρώτα την ακρίβεια του υπάρχοντος μοντέλου M_t για τα ερωτήματα Q_t και στη συνέχεια εκπαιδεύουμε το μοντέλο στα νέα δεδομένα. Στο σχήμα 8.14, μπορούμε να δούμε την απόδοση των τριών αλγορίθμων μας, οι οποίοι χρησιμοποιούν ένα απόλυτο κατώφλι, ένα αθροιστικό κατώφλι και μια παράμετρο περιόδου, σε σύγκριση με τα baselines και τη βέλτιστη λύση, Oracle.
Παρακάτω, δοκιμάζουμε τις διάφορες ευρετικές μέθοδοι σε τέσσερις διαφορετικές μετρήσεις:

- Κόστος στρατηγικής
- Prequential data accuracy
- Prequential query accuracy
- Αριθμός επανεκπαιδεύσεων

Όπως μπορούμε να παρατηρήσουμε, κάθε ένας από τους τρεις αλγορίθμους μας επιτυγχάνει καλύτερα αποτελέσματα από τα baselines όσον αφορά το συνολικό κόστος της στρατηγικής και τον αριθμό των επανεκπαιδεύσεων. Αυτό που είναι επίσης εντυπωσιακό είναι ότι είμαστε σε θέση να επιτύχουμε καλύτερες τιμές prequential data/query accuracy σε ολόκληρη τη διαδικτυακή εκτέλεση με πολύ μικρότερο αριθμό επανεκπαιδεύσεων, σε σύγκριση με τα baselines ADWIN - DDM. Στο σχήμα 8.15, μπορούμε επίσης να δούμε πώς επηρεάζει το κόστος επανεκπαίδευσης κάθε έναν από τους αλγορίθμους. Σημειώνουμε ότι τα baselines, τα οποία δεν έχουν επίγνωση του κόστους, έχουν πάντα σταθερό αριθμό επανεκπαιδεύσεων και συνεπώς, prequential accuracy. Στο σχήμα 8.16 παρατηρούμε τα αντίστοιχα γραφήματα για τη φάση της offline βελτιστοποίησης.
Συμπληρωματικά, μπορούμε να παρατηρήσουμε τους πίνακες κόστους staleness και τις αποφάσεις για επανεκπαίδευση ή όχι που έλαβε ενδεικτικά ο αλγόριθμος Threshold. Για παράδειγμα στην πάνω σειρά που η παράμετρος $\varkappa$ είναι χαμηλή, δηλαδή τα retrains είναι φτηνά, δεν υπάρχει επανεκπαίδευση μέχρι το έκτο βήμα, όπου παρατηρείται αισθητά μεγάλο staleness cost (γίνεται ορατό από το κίτρινο χρώμα) και γίνεται ένα πρώτο retrain. Αντίθετα, στην κάτω σειρά, όπου έχουμε υψηλά κόστη επανεκπαίδευσης, παρατηρείται μόνο ένα retrain στην offline βελτιστοποίηση και κανένα στο online run.
Παρατηρούμε ξανά, πώς ακριβώς για χαμηλό κόστος (Σχήμα 8.17) έχουμε περισσότερες αποφάσεις επανεκπαίδευσης απ΄ ότι με υψηλότερο κόστος (Σχήμα 8.18).
Παρουσιάζουμε τέλος τους μέσους όρους των διάφορων αλγορίθμων στο τελικό πείραμα (Πίνακας 8.2). Όσων αφορά το κόστος στρατηγικής της online φάσης, όπως θα περιμέναμε ο βέλτιστος αλγόριθμος Oracle έχει το χαμηλότερο κόστος με τις στρατηγικές μας να είναι πολύ κοντά με απόκλιση από το βέλτιστο από 1.3% εώς 3.96%. Όσων αφορά τη μέση ακρίβεια σε data και queries ο αλγόριθμος Periodic παρουσιάζει κατά κύριο λόγο τα καλύτερα αποτελέσματα. Παρατηρούμε ότι ο αλγόριθμος ελάχιστου κόστους στρατηγικής έχει εδώ από τα χαμηλότερα αποτελέσματα κατά μέσο όρο. Παρ΄ όλα αυτά, όλοι οι αλγόριθμοι επιτυγχάνουν σχετικά χαμηλές ακρίβειες κάτι που οφείλεται στην απλότητα του μοντέλου που χρησιμοποιήσαμε (ένα μόνο συνελικτικό στρώμα) και στον χαμηλό αριθμό εποχών που εκπαιδεύσαμε τα μοντέλα, από τη στιγμή που οι υψηλές ακρίβειες δεν ήταν στους σκοπούς αυτής της μελέτης, παρά η μελέτη τους σε σχέση πάντα με το κόστος εκπαίδευσης. Τέλος, στον μέσο όρο επανεκπαιδεύσεων

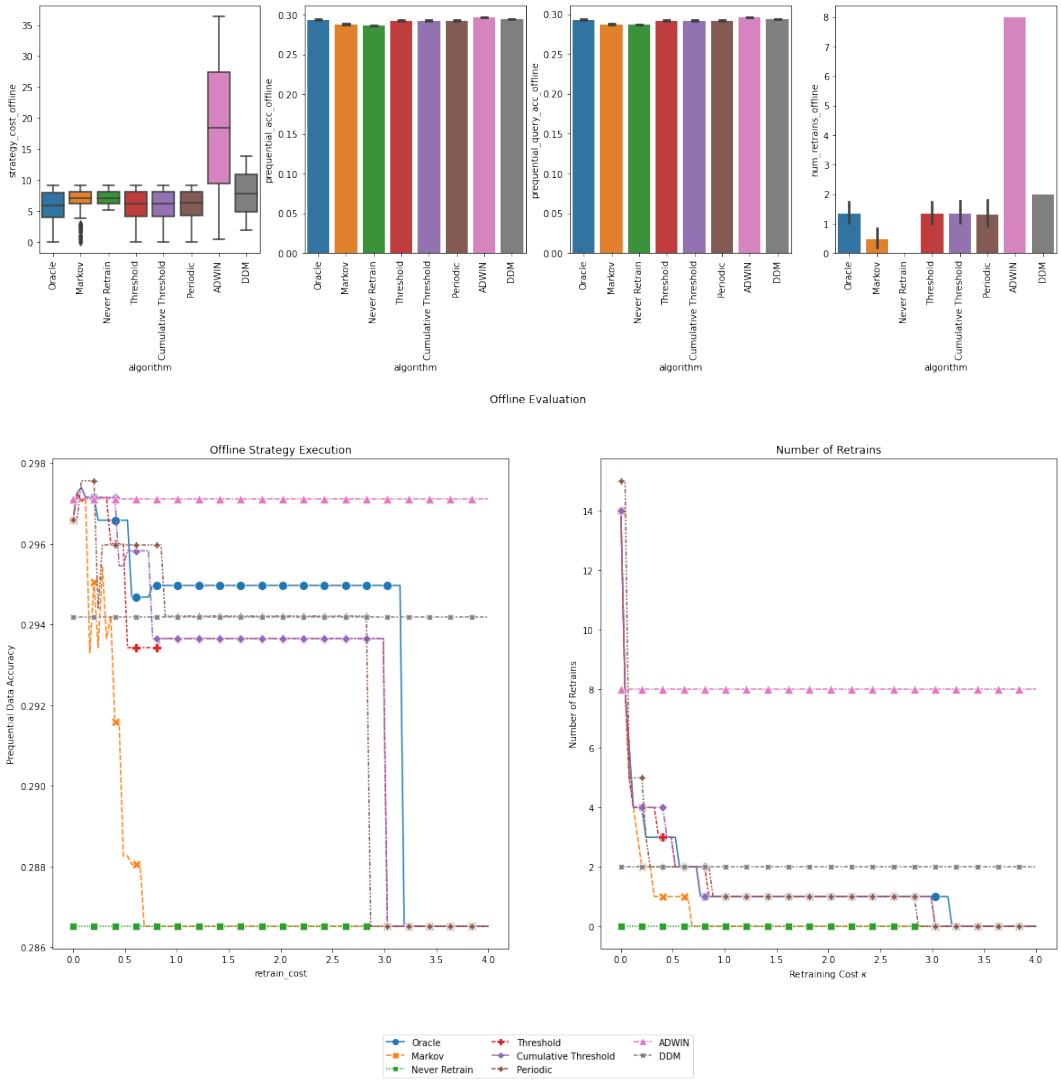


Σχήμα 8.14: Online evaluation I



Σχήμα 8.15: Online evaluation

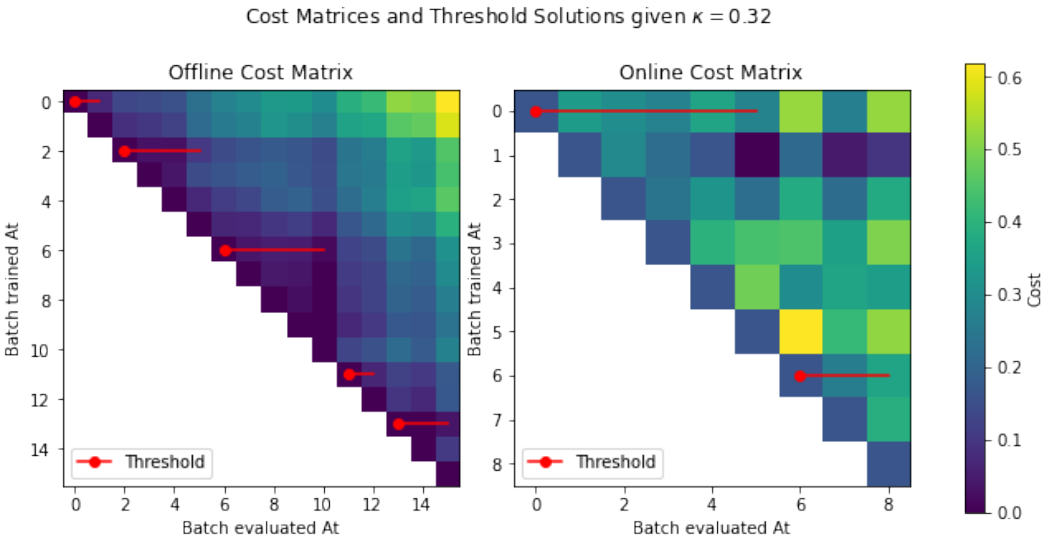
βλέπουμε ότι τα δύο baselines έχουν υπερβολικά πολλές επανεκπαιδεύσεις σε σχέση με τις άλλες μεθόδους, εκτινάσσοντας έτσι και το μέσο όρο κόστους στρατηγικής αφού αδυνατούν να προσαρμοστούν.



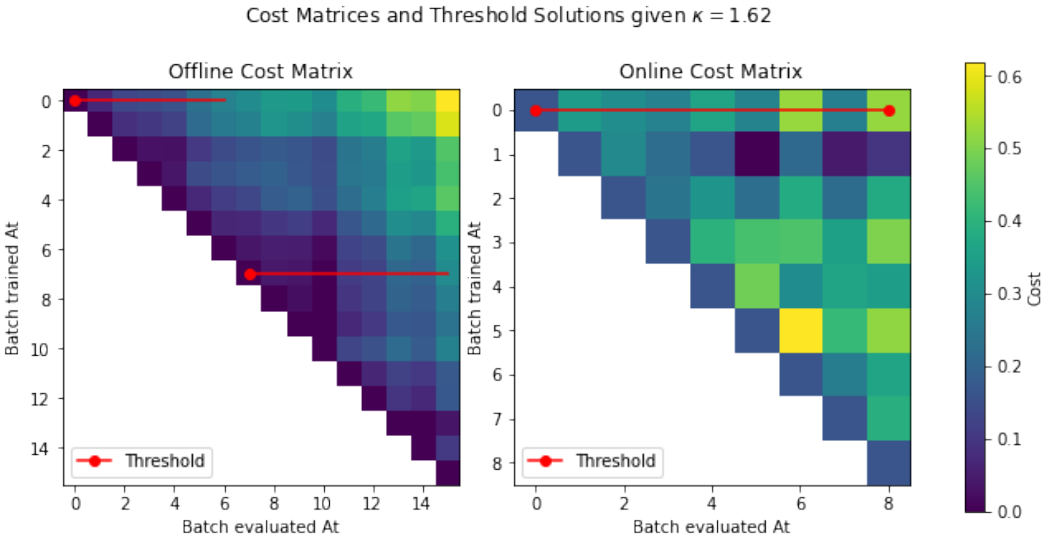
Σχήμα 8.16: Offline evaluation

Algorithm	Strategy Cost Online	Prequential Accuracy Online	Prequential Query Accuracy Online	Number of Retrains Online
ADWIN	8.454176	0.274985	0.274501	3.00
Cumulative Threshold	2.642524	0.275704	0.274841	0.44
DDM	6.300772	0.274713	0.274559	2.00
Markov	2.574816	0.275620	0.274711	0.19
Never Retrain	2.584497	0.275614	0.274700	0.00
Oracle	**2.541722**	0.274939	0.274177	0.22
Periodic	3.556269	**0.277036**	**0.276053**	1.09
Threshold	2.579207	0.275625	0.274726	0.23

Πίνακας 8.2: Average Metrics for Each Algorithm



Σχήμα 8.17: Cost matrix and Threshold retrains high retraining cost



Σχήμα 8.18: Cost matrix and Threshold retrains high retraining cost

Κεφάλαιο 9

Κατακλείδα

9.1 Συμπεράσματα

Στην παρούσα διπλωματική μελετήσαμε πώς μπορούμε να επανεκπαιδεύσουμε αποδοτικά όσον αφορά το κόστος μοντέλα αναπαράστασης γράφων, όταν αυτά αντιμετωπίζουν φαινόμενα data drift σε streaming πλαίσιο. Βασιζόμενοι στον αλγόριθμο CARA των Mahadevan et al.[62] καταφέρνουμε να επεκτείνουμε τη συνάρτηση απόφασης για γράφους. Στην αρχή, προχωρήσαμε σε μια μελέτη του μοντέλου GraphSAGE[82], το οποίο συνιστά το μοντέλο με την καλύτερη απόδοση σε πολλά datasets στατικών γράφων και χρησιμοποιείται έτσι ως κύριος embedder και ταξινομητή στα πειράματά μας. Αναδεικνύουμε έτσι πόσο γρήγορα μπορεί η απόδοσή του να πέσει σε συγκεκριμένες περιπτώσεις δυναμικής εξέλιξης γράφων και άρα διαπιστώνουμε την ανάγκη μιας εκ νέου εκπαίδευσης. Στη συνέχεια, προχωρήσαμε σε μια σειρά πειραμάτων σε συνθετικά δεδομένα για να διαπιστώσουμε την αποτελεσματικότητα των αλγορίθμων απόφασης, που επιβεβαιώνουν τελικά την επιτυχία τους στην λήψη σωστών αποφάσεων, για παράδειγμα μη αλλαγής των δεδομένων ή σε περιοδικές εισαγωγές αλλαγών στους γράφους με τη μορφή νέων κόμβων και ακμών. Τέλος, δοκιμάσαμε τις διαφορετικές παραλλαγές του αλγορίθμου απόφασης σε ένα μεγάλο σύνολο δεδομένων δυναμικού γράφου και επιτύχαμε χαμηλότερα κόστη στρατηγικής (έχοντας συσχετικοποιήσει το κόστος επανεκπαίδευσης με το κόστος λανθασμένων ταξινομήσεων) από συμβατικά baselines, που προσεγγίζουν με μεγάλη ακρίβεια (μέχρι και 1.3%) το ελάχιστο δυνατό κόστος στρατηγικής. Εκτιμούμε πώς η μέθοδός μας παράγει ασφαλή αποτελέσματα τα οποία προσφέρουν μια καινοτόμα λύση στο πρόβλημα της αποδοτικής επανεκπαίδευσης μοντέλων λαμβάνοντας υπόψη το κόστος αυτής σε συνάρτηση με την απόδοση του μοντέλου.

9.2 Μελλοντικό Έργο

Με αφορμή την παρούσα μελέτη προτείνουμε μερικές προεκτάσεις αυτής και το πώς θα μπορούσε να χρησιμοποιηθεί σε περαιτέρω μελέτες.

- Ανάπτυξη και σύγκριση αποτελεσμάτων με πρόσθετα νευρωνικά μοντέλα γράφων, όπως παραδείγματος χάρη τα GCN ή GAT και εξαγωγή συμπερασμάτων για τις ιδιότητες και τη συμπεριφορά που πρέπει να πληρεί ο graph embedder.
- Εξέταση προσθήκης μιας τρίτης πιθανής απόφασης 'Βελτιστοποίηση' για τον αλγόριθμό μας, που θα βελτιώσει δραστικά το μοντέλο όταν τα πρότυπα που μαθαίνονται από τις

παρτίδες εκτός σύνδεσης έχουν αλλάξει και, επομένως, μια νέα φάση βελτιστοποίησης εκτός σύνδεσης απαιτείται.

- Το μοντέλο GraphSAGE δεν λαμβάνει υπόψη του τους χρονικούς παράγοντες, σε αντίθεση με τα Temporal Graph Networks (TGN), τα οποία αντικατοπτρίζουν το ιστορικό ενός κόμβου διατηρώντας μια μνήμη. Στην περίπτωσή μας, η επανεκπαίδευση του μοντέλου αναδεικνύει την απόφαση να ενδιαφερθούμε μόνο για την ακεραιότητα του μοντέλου στα τρέχοντα δεδομένα και όχι για το πώς έχει εξελιχθεί ο γράφος. Παρ΄ όλα αυτά, τα TGN θα μπορούσαν ακόμα να χρησιμοποιηθούν σε μια συγκριτική μελέτη, όπου μόνο μετράμε την ακρίβεια, καθώς η σύγκριση του συνολικού κόστους δεν είναι εφικτή, μαζί με μια πλήρης θεωρητική ανάλυση αυτών και του τρόπου με τον οποίο διαφέρουν από τα κλασικά μοντέλα GNN.
- Επέκταση σε γράφους με δυναμικές ετικέτες.
- Πειράματα σε διαφορετικά tasks στον τομέα του graph learning, όπως edge classification.

Βιβλιογραφία

[1] M. B. Guido Appenzeller and M. Casado, *Navigating the high cost of ai compute*, [Online; accessed January 11, 2024].

[2] L. L. Abhinav Venigalla, *Mosaic llms (part 2): Gpt-3 quality for ¡500k*, [Online; accessed January 11, 2024].

[3] L. F. W. Anthony, B. Kanding, and R. Selvan, "Carbontracker: Tracking and predicting the carbon footprint of training deep learning models", *CoRR*, vol. abs/2007.03051, 2020. arXiv: 2007.03051. [Online]. Available: https://arxiv.org/abs/2007.03051.

[4] N. C. Thompson, K. H. Greenewald, K. Lee, and G. F. Manso, "The computational limits of deep learning", *CoRR*, vol. abs/2007.05558, 2020. arXiv: 2007.05558. [Online]. Available: https://arxiv.org/abs/2007.05558.

[5] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Compute and energy consumption trends in deep learning inference", *CoRR*, vol. abs/2109.05472, 2021. arXiv: 2109.05472. [Online]. Available: https://arxiv.org/abs/2109.05472.

[6] D. Li, X. Chen, M. Becchi, and Z. Zong, "Evaluating the energy efficiency of deep convolutional neural networks on cpus and gpus", in *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, 2016, pp. 477–484. DOI: 10.1109/BDCloud-SocialCom-SustainCom.2016.76.

[7] D. A. Patterson, J. Gonzalez, Q. V. Le, *et al.*, "Carbon emissions and large neural network training", *CoRR*, vol. abs/2104.10350, 2021. arXiv: 2104.10350. [Online]. Available: https://arxiv.org/abs/2104.10350.

[8] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[9] R. A. Fisher, "The use of multiple measurements in taxonomic problems", *Annals of Eugenics*, vol. 7, pp. 179–188, 1936, Reproduced with permission of Cambridge University Press.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006, ISBN: 978-0387310732.

[11] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne", *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[12] A. R. Ali Mousavi and G. R. Lihong Li Research Scientist, *Reinforcement learning*, [Online; accessed January 10, 2024].

[13] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[14] D. Hebb, *The organization of behavior. emphnew york*, 1949.

[15] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012, ISBN: 0262018020.

[16] H. Robbins and D. Siegmund, "A convergence theorem for non-negative almost supermartingales and some applications", *Optimizing Methods in Statistics*, vol. 1, pp. 233–257, 1971.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", *Nature*, vol. 323, pp. 533–536, 1986. [Online]. Available: https://api.semanticscholar.org/CorpusID:205001834.

[18] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex", *The Journal of Physiology*, vol. 148, no. 3, pp. 574–591, 1959. DOI: 10.1113/jphysiol.1959.sp006308.

[19] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980. DOI: 10.1007/BF00344251.

[20] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*. O'Reilly Media, 2019, https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/, ISBN: 978-1492032632.

[21] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2015. arXiv: 1409.1556 [cs.CV].

[22] C. Szegedy, W. Liu, Y. Jia, *et al.*, *Going deeper with convolutions*, 2014. arXiv: 1409.4842 [cs.CV].

[23] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25, Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[26] M. Xu, "Understanding graph embedding methods and their applications", *CoRR*, vol. abs/2012.08019, 2020. arXiv: 2012.08019. [Online]. Available: https://arxiv.org/abs/2012.08019.

[27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data", *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, Jul. 2017, ISSN: 1558-0792. DOI: 10.1109/msp.2017.2693418. [Online]. Available: http://dx.doi.org/10.1109/MSP.2017.2693418.

[28] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, and K. Murphy, "Machine learning on graphs: A model and comprehensive taxonomy", *CoRR*, vol. abs/2005.03675, 2020. arXiv: 2005.03675. [Online]. Available: https://arxiv.org/abs/2005.03675.

[29] H. Chen, X. Sun, Y. Tian, B. Perozzi, M. Chen, and S. Skiena, "Enhanced network embeddings via exploiting edge labels", Oct. 2018, pp. 1579–1582. DOI: 10.1145/3269206.3269270.

[30] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, and K. Murphy, *Machine learning on graphs: A model and comprehensive taxonomy*, 2022. arXiv: 2005.03675 [cs.LG].

[31] Z. Yang, W. W. Cohen, and R. Salakhutdinov, *Revisiting semi-supervised learning with graph embeddings*, 2016. arXiv: 1603.08861 [cs.LG].

[32] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks", *Annual Review of Sociology*, vol. 27, pp. 415–444, 2001, ISSN: 03600572, 15452115. [Online]. Available: http://www.jstor.org/stable/2678628 (visited on 01/19/2024).

[33] P. V. MARSDEN and N. E. FRIEDKIN, "Network studies of social influence", *Sociological Methods & Research*, vol. 22, no. 1, pp. 127–151, 1993. DOI: 10.1177/0049124193022001006. eprint: https://doi.org/10.1177/0049124193022001006. [Online]. Available: https://doi.org/10.1177/0049124193022001006.

[34] N. K. Ahmed, R. Rossi, J. B. Lee, *et al.*, *Learning role-based graph embeddings*, 2018. arXiv: 1802.02896 [stat.ML].

[35] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "Struc2vec: Learning node representations from structural identity", in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17, ACM, Aug. 2017. DOI: 10.1145/3097983.3098061. [Online]. Available: http://dx.doi.org/10.1145/3097983.3098061.

[36] K. Henderson, B. Gallagher, T. Eliassi-Rad, *et al.*, "Rolx: Structural role extraction & mining in large graphs", in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '12, Beijing, China: Association for Computing Machinery, 2012, pp. 1231–1239, ISBN: 9781450314626. DOI: 10.1145/2339530.2339723. [Online]. Available: https://doi.org/10.1145/2339530.2339723.

[37] K. Henderson, B. Gallagher, L. Li, *et al.*, "It's who you know: Graph mining using recursive structural features", in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11, San Diego, California, USA: Association for Computing Machinery, 2011, pp. 663–671, ISBN: 9781450308137. DOI: 10.1145/2020408.2020512. [Online]. Available: https://doi.org/10.1145/2020408.2020512.

[38] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation", in *Advances in Neural Information Processing Systems 16*, MIT Press, 2003, pp. 585–591.

[39] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding", in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 1105–1114, ISBN: 9781450342322. DOI: 10.1145/2939672.2939751. [Online]. Available: https://doi.org/10.1145/2939672.2939751.

[40] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality*, 2013. arXiv: 1310.4546 [cs.CL].

[41] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations", in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '14, ACM, Aug. 2014. DOI: 10.1145/2623330.2623732. [Online]. Available: http://dx.doi.org/10.1145/2623330.2623732.

[42] A. Grover and J. Leskovec, *Node2vec: Scalable feature learning for networks*, 2016. arXiv: 1607.00653 [cs.SI].

[43] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding", in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15, International World Wide Web Conferences Steering Committee, May 2015. DOI: 10.1145/2736277.2741093. [Online]. Available: http://dx.doi.org/10.1145/2736277.2741093.

[44] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, *Harp: Hierarchical representation learning for networks*, 2017. arXiv: 1706.07845 [cs.SI].

[45] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model", *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, Jan. 2009, PMID: 19068426, ISSN: 1045-9227. DOI: 10.1109/TNN.2008.2005605. [Online]. Available: https://doi.org/10.1109/TNN.2008.2005605.

[46] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains", *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, 729–734 vol. 2, 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:20480879.

[47] F. J. Pineda, "Generalization of back-propagation to recurrent neural networks", *Phys. Rev. Lett.*, vol. 59, pp. 2229–2232, 19 Nov. 1987. DOI: 10.1103/PhysRevLett.59.2229. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.59.2229.

[48] L. B. Almeida, "A learning rule for asynchronous perceptrons with feedback in a combinatorial environment", in *Artificial Neural Networks: Concept Learning*. IEEE Press, 1990, pp. 102–111, ISBN: 0818620153.

[49] Y. LeCun, B. Boser, J. S. Denker, *et al.*, "Backpropagation applied to handwritten zip code recognition", *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989. DOI: 10.1162/neco.1989.1.4.541.

[50] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, 2017. arXiv: 1609.02907 [cs.LG].

[51] J. Chen, T. Ma, and C. Xiao, *Fastgcn: Fast learning with graph convolutional networks via importance sampling*, 2018. arXiv: 1801.10247 [cs.LG].

[52] W. L. Hamilton, R. Ying, and J. Leskovec, *Inductive representation learning on large graphs*, 2018. arXiv: 1706.02216 [cs.SI].

[53] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, *Temporal graph networks for deep learning on dynamic graphs*, 2020. arXiv: 2006.10637 [cs.LG].

[54] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs", *arXiv preprint arXiv:2002.07962*, 2020.

[55] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks", in *Proceedings of the 25th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2019.

[56] J. Chen, J. Zhang, X. Xu, *et al.*, *E-lstm-d: A deep learning framework for dynamic network link prediction*, 2019. arXiv: 1902.08329 [cs.SI].

[57] P. Goyal, S. R. Chhetri, and A. Canedo, "Dyngraph2vec: Capturing network dynamics using dynamic graph representation learning", *Knowledge-Based Systems*, vol. 187, p. 104 816, Jan. 2020, ISSN: 0950-7051. DOI: 10.1016/j.knosys.2019.06.024. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2019.06.024.

[58] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, *Gated graph sequence neural networks*, 2017. arXiv: 1511.05493 [cs.LG].

[59] R. Paudel and W. Eberle, "An approach for concept drift detection in a graph stream using discriminative subgraphs", *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 6, Sep. 2020, ISSN: 1556-4681. DOI: 10.1145/3406243. [Online]. Available: https://doi.org/10.1145/3406243.

[60] Y. Yao and L. B. Holder, "Detecting concept drift in classification over streaming graphs", 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:16440679.

[61] Y. Zheng, H. Wang, Z. Wei, J. Liu, and S. Wang, *Instant graph neural networks for dynamic graphs*, 2022. arXiv: 2206.01379 [cs.LG].

[62] A. Mahadevan and M. Mathioudakis, *Cost-effective retraining of machine learning models*, 2023. arXiv: 2310.04216 [cs.LG].

[63] J. C. Schlimmer and R. H. Granger, "Incremental learning from noisy data", *Machine Learning*, vol. 1, no. 3, pp. 317–354, 1986, ISSN: 1573-0565. DOI: 10.1007/BF00116895. [Online]. Available: https://doi.org/10.1007/BF00116895.

[64] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review", *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2018, ISSN: 2326-3865. DOI: 10.1109/tkde.2018.2876857. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2018.2876857.

[65] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation", *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014.

[66] A. L. Suárez-Cetrulo, D. Quintana, and A. Cervantes, "A survey on machine learning for recurring concept drifting data streams", *Expert Systems with Applications*, vol. 213, p. 118 934, 2023, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2022.118934. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422019522.

[67] S. Zhang, J. Liu, and X. Zuo, "Adaptive online incremental learning for evolving data streams", *CoRR*, vol. abs/2201.01633, 2022. arXiv: 2201.01633. [Online]. Available: https://arxiv.org/abs/2201.01633.

[68] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem", in ser. Psychology of Learning and Motivation, G. H. Bower, Ed., vol. 24, Academic Press, 1989, pp. 109–165. DOI: https://doi.org/10.1016/S0079-7421(08)60536-8. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0079742108605368.

[69] W. C. Abraham and A. Robins, "Memory retention – the synaptic stability versus plasticity dilemma", *Trends in Neurosciences*, vol. 28, no. 2, pp. 73–78, 2005, ISSN: 0166-2236. DOI: https://doi.org/10.1016/j.tins.2004.12.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166223604003704.

[70] R. M. French, "Catastrophic forgetting in connectionist networks", *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[71] R. Kemker, A. Abitino, M. McClure, and C. Kanan, "Measuring catastrophic forgetting in neural networks", *CoRR*, vol. abs/1708.02072, 2017. arXiv: 1708.02072. [Online]. Available: http://arxiv.org/abs/1708.02072.

[72] J. Serrà, D. Surís, M. Miron, and A. Karatzoglou, *Overcoming catastrophic forgetting with hard attention to the task*, 2018. arXiv: 1801.01423 [cs.LG].

[73] T. J. Draelos, N. E. Miner, C. C. Lamb, *et al.*, "Neurogenesis deep learning: Extending deep networks to accommodate new classes", in *2017 International Joint Conference on Neural Networks (IJCNN)*, IEEE, May 2017. DOI: 10.1109/ijcnn.2017.7965898. [Online]. Available: http://dx.doi.org/10.1109/IJCNN.2017.7965898.

[74] A. Rannen, R. Aljundi, M. B. Blaschko, and T. Tuytelaars, "Encoder based lifelong learning", in *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Oct. 2017. DOI: 10.1109/iccv.2017.148. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2017.148.

[75] B. Goodrich and I. Arel, "Unsupervised neuron selection for mitigating catastrophic forgetting in neural networks", in *2014 IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2014, pp. 997–1000. DOI: 10.1109/MWSCAS.2014.6908585.

[76] K. Lee, K. Lee, J. Shin, and H. Lee, "Overcoming catastrophic forgetting with unlabeled data in the wild", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 312–321.

[77] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing", vol. 7, Apr. 2007. DOI: 10.1137/1.9781611972771.42.

[78] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection", in *Advances in Artificial Intelligence – SBIA 2004*, A. L. C. Bazzan and S. Labidi, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 286–295, ISBN: 978-3-540-28645-5.

[79] M. Fey, J. E. Lenssen, F. Weichert, H. Müller, and K. Kersting, "Fast graph representation learning with pytorch geometric", in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[80] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations", in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, ser. KDD '05, Chicago, Illinois, USA: Association for Computing Machinery, 2005, pp. 177–187, ISBN: 159593135X. DOI: 10.1145/1081870.1081893. [Online]. Available: https://doi.org/10.1145/1081870.1081893.

[81] L. McInnes, J. Healy, and J. Melville, *Umap: Uniform manifold approximation and projection for dimension reduction*, 2020. arXiv: 1802.03426 [stat.ML].

[82] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs", *CoRR*, vol. abs/1706.02216, 2017. arXiv: 1706.02216. [Online]. Available: http://arxiv.org/abs/1706.02216.