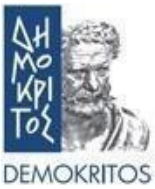

Εφαρμογές τεχνικών μηχανικής μάθησης σε δεδομένα γράφων

Από

Καμπάνης Αλέξανδρος Ιάσονας

Υποβάλλεται

για την εκπλήρωση των προϋποθέσεων λήψης

Μεταπτυχιακού Διπλώματος

στην «Τεχνητή Νοημοσύνη»

στο

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

04 2023

Συγγραφέας Καμπάνης Αλέξανδρος Ιάσονας

ΔΠΜΣ «Τεχνητή Νοημοσύνη»

04 05, 2023

Έγινε αποδεκτό από/ Certified by. . . . . .

Μαρία Χαλκίδη
Αναπληρώτρια Καθηγήτρια
Επιβλέπουσα

Έγινε αποδεκτό από/ Certified by. . . . . .

Μιχάλης Φιλιππάκης
Καθηγητής
Μέλος Εξεταστικής Επιτροπής

Έγινε αποδεκτό από/ Certified by. . . . . .

Ηλίας Μαγκλογιάννης
Καθηγητής
Μέλος Εξεταστικής Επιτροπής

Εφαρμογές τεχνικών μηχανικής μάθησης σε δεδομένα γράφων

Από

Καμπάνης Αλέξανδρος Ιάσονας

Υποβλήθηκε στο ΔΠΜΣ «Τεχνητή Νοημοσύνη» την ΧΧ Μηνός 20ΧΧ ως υποχρέωση για την λήψη Μεταπτυχιακού Διπλώματος Σπουδών

Abstract

In recent years advancements in Machine Learning and the massive production of data with the natural occurrence of relations lead organizations and research to seek new methods to exploit them or generalize traditional machine learning technologies for this purpose. The most well known occurrence is social networks. The purpose of this thesis is to present and explain the basic principles and terms related to the domain of Graph Representation Learning. We present all basic terms needed for the reader to understand all the algorithms explained in the literature and explain the inner workings of those algorithms according to characteristics and taxonomies. In the use case of fraud detection and the learning objective of semi supervised learning we choose the most appropriate algorithms based on the characteristics that are useful for this scenario and we assess their performance based on the selected use case and the dataset chosen. We perform and comment on the results of several experiments designed to test the performance of those algorithms of the GRL domain in those specific data.

Περίληψη

Τα τελευταία χρόνια οι εξελίξεις στον τομέα της Μηχανικής Μάθησης σε συνδυασμό με την παραγωγή δεδομένων μεγάλης κλίμακας με σχέσεις που προκύπτουν μεταξύ των δειγμάτων να παράγονται αυτόματα οδήγησαν τους οργανισμούς και ερευνητές να βρουν καινούργιους τρόπους ή να γενικεύσουν υπάρχοντες ώστε να αξιοποιήσουν αυτά τα δεδομένα. Το πιο χαρακτηριστικό παράδειγμα είναι αυτό των κοινωνικών δικτύων. Σε

αυτήν την εργασία σκοπός είναι να περιγράψουμε τις βασικές έννοιες που σχετίζονται με τον τομέα του Graph Representation Learning. Παρουσιάζουμε όλες τις βασικές έννοιες που απαιτούνται καθώς και τους κύριους αλγορίθμους και ταξινομίες που τους κατατάσουν σε κατηγορίες με βάση τα χαρακτηριστικά τους. Στο σενάριο χρήσης της ανίχνευσης απάτης σε συναλλαγές επιλέγουμε τους πλέον κατάλληλους και αξιολογούμε την επίδοση τους στο σενάριο της ημι επιβλεπόμενης μάθησης. Παρουσιάζουμε και σχολιάζουμε τα αποτελέσματα με βάση πειράματα που αξιολογούν πολλά σενάρια όπως end to end εκπαίδευση αλλά και παραγωγή αναπαραστάσεων με μη επιβλεπόμενο τρόπο.

Επιβλέπουσα: Μαρία Χαλκίδη
Ακαδημαϊκή Θέση: Αναπληρώτρια Καθηγήτρια

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Table of Figures

Index of Tables

ΚΕΦΑΛΑΙΟ 1ο

Εισαγωγή

Τα δίκτυα είναι ένα θεμελιώδες εργαλείο για τη μοντελοποίηση πολύπλοκων κοινωνικών, τεχνολογικών και βιολογικών συστημάτων **[25]**. Σε συνδυασμό με την εμφάνιση διαδικτυακών κοινωνικών δικτύων και τη διαθεσιμότητα δεδομένων μεγάλης κλίμακας υπάρχουν πλέον οι αναγκαίες προϋποθέσεις και τα εργαλεία για την επεξεργασία των διαθέσιμων δεδομένων και την εξαγωγή γνώσης από αυτά. Με βάση αυτά, η εργασία εστιάζει στην μελέτη τεχνικών ανάλυσης μεγάλων δικτύων που παρουσιάζουν αρκετές υπολογιστικές και αλγοριθμικές προκλήσεις.

Θα εστιάσουμε το πειραματικό μέρος της εργασίας στη μελέτη περίπτωσης της ανίχνευσης απάτης σε συναλλαγές **[18]**.

Στόχος της εργασίας είναι να παρουσιάσει τις βασικές έννοιες σχετικά με την ανίχνευση απάτης σε συναλλαγές και την επίλυση του προβλήματος με χρήση τεχνικών μηχανικής μάθησης. Στα πλαίσια της εργασίας θα μελετήσουμε τις ιδιαιτερότητες **[21]** που προκύπτουν από δεδομένα γράφων στην Μηχανική Μάθηση και τις λύσεις **[4,5,17]** που έχουν προταθεί για την ανάλυση δεδομένων γράφων. Επίσης, θα εφαρμόσουμε και θα συγκρίνουμε αλγορίθμους από αυτές τις λύσεις στα πλαίσια της μελέτης περίπτωσης που επιλέχθηκε καθώς και θα αξιολογήσουμε τα αποτελέσματα τους.

Όπως θα δούμε σε επόμενες ενότητες ο στόχος για το πρόβλημα **[18]** που θέλουμε να αντιμετωπίσουμε έχει πολλές δυσκολίες που πρέπει να αντιμετωπιστούν σε δύο άξονες.

Ο πρώτος αφορά γενικά την περίπτωση της ανίχνευσης απάτης σε συναλλαγές. Η διαδικασία αυτή παρουσιάζει δυσκολίες σχετικά με το ότι τα δεδομένα εκπαίδευσης αλλά και οι περιπτώσεις που θα αντιμετωπιστούν κατά την διάρκεια λειτουργίας ενός τέτοιου συστήματος θα έχουν το χαρακτηριστικό του ότι τα διαθέσιμα δείγματα για κάθε κλάση δεν θα είναι ίδια σε πλήθος (highly imbalanced) **[26]** φαινόμενο το οποίο παρουσιάζεται και σε άλλους τομείς που βρίσκουν εφαρμογή αυτές οι τεχνολογίες. Με βάση αυτό παρουσιάζονται δυσκολίες όπως τον τρόπο που θα γίνει η εκπαίδευση αλλά και τον τρόπο

που θα αξιολογηθεί η απόδοση ενός τέτοιου μοντέλου. Θα δούμε και θα δοκιμάσουμε τρόπους **[26]** για να αντιμετωπίσουμε τα δύο προηγούμενα προβλήματα και θα τα αξιολογήσουμε με βάση το σενάριο χρήσης και τα δεδομένα που έχουμε.

Ο δεύτερος άξονας αφορά την περίπτωση ύπαρξης γράφων **[2]**. Στην περίπτωση της κλασικής μηχανικής μάθησης έχουμε την περίπτωση όπου έχουμε δείγματα τα οποία δεν έχουν κάποια σχέση μεταξύ τους και με βάση αυτά εκπαιδεύουμε τον αλγόριθμο. Επίσης, υπάρχουν περιπτώσεις όπως στην περίπτωση της υπολογιστικής όρασης (computer vision) **[27]** και της επεξεργασίας φυσικής γλώσσας **[28]** όπου τα δεδομένα που αναλύουμε λαμβάνουν υπόψη γειτονικά δεδομένα όπως πχ σε ένα CNN **[29]** για επεξεργασία εικόνας ή ένα RNN **[30]** για επεξεργασία κειμένου. Σε αυτές τις περιπτώσεις εκτός από τα δεδομένα που επεξεργαζόμαστε λαμβάνουμε υπόψη και γειτονικά δεδομένα. Αυτές οι περιπτώσεις αναφέρονται σαν Ευκλείδεια **[21]** δεδομένα καθώς η δομή τους είναι προκαθορισμένη σαν δομή μίας ή δύο διαστάσεων ή τριών διαστάσεων. Για παράδειγμα, ένα εικονοστοιχείο (pixel) σε ένα CNN έχει συγκεκριμένο αριθμό γειτονικών pixels πάντα στην ίδια απόσταση, δηλαδή το κάθε γειτονικό pixel έχει την ίδια αξία. Στην περίπτωση της επεξεργασίας φυσικής γλώσσας επίσης η κάθε λέξη ακολουθείται πάντα από μία μόνο λέξη και προηγείται μίας μόνο λέξης. Με βάση αυτά η εικόνα έχει σταθερή δομή 2D και η περίπτωση του nlp σταθερή δομή 1D.

Στην περίπτωση γράφων όμως οι παραπάνω περιπτώσεις δεν ισχύουν. Σε ένα γράφο πχ για κοινωνικά δίκτυα ο κάθε κόμβος εκτός από τα χαρακτηριστικά (features) τα οποία μπορεί να έχει, έχει και σχέση ή σχέσεις με τους γείτονες του. Αυτό θεωρητικά δίνει επιπλέον πληροφορία την οποία μπορούμε να αξιοποιήσουμε για να λύσουμε κάποιο πρόβλημα **[2,17]** που μας ενδιαφέρει όπως κατηγοριοποίηση κόμβων, γράφων ή την πρόβλεψη κάποιας σχέσης (link prediction) μεταξύ κόμβων για να αναφέρουμε τα πιο γνωστά. Το πρόβλημα που προκύπτει **[21]** και περιορίζει την χρήση μεθόδων παραδοσιακής μηχανικής μάθησης στο πρόβλημα είναι ότι η δομή του γράφου δεν είναι προκαθορισμένη και δεν είναι πάντα ίδια όπως στις προηγούμενες περιπτώσεις που αναφέραμε αλλά υπάρχει άγνωστος αριθμός γειτόνων και πιθανώς διαφορετική αξία ή είδος σε κάθε σχέση που συνδέει κόμβους. Με βάση αυτά γίνεται έρευνα **[17]** για το πως μπορούν παραδοσιακές και δοκιμασμένες τεχνικές όπως η συνέλιξη σε τέτοια δεδομένα να γενικευτούν και να βρουν εφαρμογή και σε δομές δεδομένων όπως αυτές που περιγράψαμε. Θεωρητικά η χρήση αυτής της επιπλέον πληροφορίας θα αυξήσει την

απόδοση σε σχέση με το σενάριο όπου δεν συνυπολογίζεται. Επίσης, οι τεχνολογίες που εκμεταλλεύονται αυτήν την πληροφορία μπορούν να βελτιώσουν τις παραδοσιακές τεχνικές σε σενάρια χρήσης όπου συνδυάζουμε **[17]** για παράδειγμα υπολογιστική όραση (computer vision) με επεξεργασία φυσικής γλώσσας.

Ο τομέας αυτός στην βιβλιογραφία αναφέρεται σαν Graph Representation Learning **[2]** και ασχολείται με αυτά τα προβλήματα όπου η δομή του γράφου είναι πολλές φορές δεδομένη όπως κοινωνικά δίκτυα και άλλα. Γενικά σκοπός είναι να παραχθούν αναπαραστασεις (embendings) σε χαμηλή διάσταση οι οποίες κωδικοποιούν πληροφορία για την δομή του γράφου και πολλές φορές όπως θα δούμε ανακατασκευάζουν **[4,5,2,1]** κάποιο χαρακτηριστικό με βάση αυτές τις αναπαραστάσεις αν και αυτό δεν είναι απόλυτο.

Ο τομέας αυτός έχει αναπτυχθεί πολύ τα τελευταία χρόνια και έχουν προταθεί λύσεις πέρα από τις παραδοσιακές **[4,5,2,1]** οι οποίες βασίζονται κυρίως σε χειροποίητα (handcrafted) χαρακτηριστικά για να αντιμετωπιστεί αυτό το πρόβλημα, λύσεις που βασίζονται σε νευρωνικά δίκτυα **[17]**. Η περίπτωση των νευρωνικών δικτύων είναι ιδιαίτερα ελκυστική για πολλά σενάρια χρήσης όπως ανάλυση κοινωνικών δικτύων, δικτύων συναλλαγών και άλλα καθώς μπορεί να αξιοποιήσει ταυτόχρονα **[17,5]** και την πληροφορία από την δομή και την συνδεσιμότητα του γράφου αλλά και την πληροφορία από χαρακτηριστικά (features) που πιθανόν συνοδεύουν τον κάθε κόμβο ή ακμή του γράφου.

Σε αντίθεση με τις λύσεις των νευρωνικών και πριν από αυτά η πληροφορία που αφορά τον γράφο σχετικά με την συνδεσιμότητα εξάγεται με χρήση χειροποίητων χαρακτηριστικών (handcrafted features) **[31]** όπως betweenness, centrality, node degree και άλλα. Στην συνέχεια αυτή η πληροφορία χρησιμοποιούνταν **[31]** με χρήση κάποιου αλγορίθμου κατηγοριοποίησης σε συνδυασμό με τα χαρακτηριστικά κόμβων/ακμών. Γενικά υπάρχουν πολλοί τρόποι **[5,2]** για την εκπαίδευση των αναπαραστάσεων τις οποίες θα αναφέρουμε αναλυτικά στην εργασία αυτή. Κύριος στόχος της εργασίας είναι να αξιολογήσουμε τις τεχνικές που αφορούν τα νευρωνικά δίκτυα για τον λόγο του ότι είναι καινούριες λύσεις και η έρευνα γύρω από αυτές είναι σε εξέλιξη καθώς επίσης και επειδή ταιριάζουν καλύτερα στο σενάριο χρήσης που ασχολούμαστε.

Με βάση τα παραπάνω και το σενάριο χρήσης της ανίχνευσης απάτης σε συναλλαγές **[18]** θα επιλέξουμε να αξιολογήσουμε τις νέες αυτές τεχνολογίες με βάση τα δεδομένα

που έχουμε στην διάθεση μας και θα τις συγκρίνουμε μεταξύ τους αλλά και με μία τεχνολογία παραδοσιακών αλγορίθμων Μηχανικής Μάθησης. Οι τεχνολογίες Βαθιάς Μηχανικής Μάθησης με βάση την βιβλιογραφία **[22,6]** μπορούν να κατηγοριοποιηθούν ως spectral και spatial. Για την πρώτη κατηγορία έχουμε το σενάριο μάθησης όπου όλα τα δεδομένα του γράφου δηλαδή οι κόμβοι και οι ακμές πρέπει να είναι διαθέσιμα κατά την διάρκεια της εκπαίδευσης **[4,5]** και οι αναπαραστάσεις του κάθε κόμβου βελτιστοποιούνται με τους υπολογισμούς να βασίζονται στον πίνακα γειτνίασης (Adjacency matrix) κάποια μορφή του Laplacian ή την βάση Fourier του Laplacian. Αυτή η περίπτωση παρουσιάζει τις δυσκολίες όπου περιλαμβάνει κοστοβόρους υπολογισμούς πινάκων και τον περιορισμό του ότι δεν μπορεί να γενικεύσει την γνώση σε νέους γράφους. Για την δεύτερη περίπτωση έχουμε το σενάριο όπου οι αναπαραστάσεις προκύπτουν από την γειτονιά ενός κόμβου **[4,5]**. Οι αλγόριθμοι της δεύτερης περίπτωσης έχουν πολλά πλεονεκτήματα για το δικό μας σενάριο χρήσης όπως θα δούμε στην συνέχεια αλλά και γενικά με πιο σημαντικό χαρακτηριστικό το ότι μπορούν εύκολα να χρησιμοποιηθούν σε πολύ μεγάλους γράφους **[6,19,5]** αλλά και το ότι κάποιοι από αυτούς μπορούν να παράξουν αναπαραστάσεις για γράφους που δεν έχουν ξαναδεί.

Το υπόλοιπο της εργασίας οργανώνεται ως εξής. Στην ενότητα 2 θα κάνουμε μία επισκόπηση των βασικών τεχνικών μηχανικής μάθησης σε γράφους, στην ενότητα 3 θα παρουσιάσουμε την μεθοδολογία που θα ακολουθήσουμε, στην ενότητα 4 θα περιγράψουμε το πειραματικό μέρος και την υλοποίηση καθώς και τα αποτελέσματα των πειραμάτων και στην ενότητα 5 θα σχολιάσουμε τα συμπεράσματα.

ΚΕΦΑΛΑΙΟ 2ο

Επισκόπηση τεχνικών μηχανικής μάθησης σε γράφους

Σε αυτή την ενότητα θα παρουσιάσουμε τις βασικές αρχές για τον τομέα που θα παρουσιάσουμε δηλαδή την μηχανική Μάθηση σε γράφους **[2,4]**. Επειδή αποτελεί υποκλάδο της Μηχανικής Μάθησης θα κάνουμε μία σύντομη αναφορά στην Μηχανική Μάθηση και την Βαθιά Μηχανική Μάθηση καθώς θα εξετάσουμε λύσεις για το σενάριο χρήσης που θα ασχοληθούμε **[18]** που χρησιμοποιούν και τις δύο αν και θα δώσουμε μεγαλύτερη έμφαση στις λύσεις Βαθιάς Μηχανικής Μάθησης καθώς είναι καινούργιες τεχνολογίες και έχουν χαρακτηριστικά τα οποία είναι πολύ χρήσιμα για το σενάριο χρήσης αυτής της εργασίας.

Στην συνέχεια θα παρουσιάσουμε βασικούς όρους που σχετίζονται με τον τομέα των γράφων και αποτελούν προαπαιτούμενα για να καταλάβει ο αναγνώστης το πρόβλημα και τις λύσεις που προτείνονται στην βιβλιογραφία.

Στην συνέχεια θα παρουσιάσουμε την θεωρία που αναδεικνύει τις ιδιαιτερότητες και τα προβλήματα που σχετίζονται με αυτόν τον τομέα και τρόπους που μπορούν να αντιμετωπιστούν.

Στην συνέχεια θα παρουσιάσουμε τους πιο συνηθισμένους στόχους και προβλήματα που καλούνται να λύσουν αυτές οι τεχνικές Μηχανικής Μάθησης.

Στην συνέχεια θα παρουσιάσουμε τις κύριες κατηγορίες αλγορίθμων με βάση κριτήρια που θα αναφέρουμε τα πλεονεκτήματα και τα μειονεκτήματα τους.

Τέλος θα παρουσιάσουμε κάποια ανοικτά ερευνητικά θέματα που προκύπτουν από τη βιβλιογραφία.

2.1 Εισαγωγή στις βασικές τεχνικές μηχανικής μάθησης

Η μηχανική Μάθηση **[32]** και ο υποκλάδος της, η Βαθιά Μηχανική Μάθηση **[32]** αποτελούν τομέα της Τεχνητής Νοημοσύνης ο οποίος έχει γνωρίσει μεγάλη εξέλιξη τα τελευταία χρόνια και η έρευνα που πραγματοποιείται αφορά πολλούς στόχους. Η Τεχνητή Νοημοσύνη γενικά περιλαμβάνει πολλούς υποκλάδους όπως της υπολογιστικής όρασης (computer vision), επεξεργασία φυσικής γλώσσας (natural language processing) και άλλα. Ο απόλυτος στόχος της Τεχνητής Νοημοσύνης είναι αυτό που στην βιβλιογραφία αναφέρεται συχνά σαν general AI **[33]**. Αυτός ο στόχος περιγράφει έναν ευφυή πράκτορα ο οποίος έχει και υλική υπόσταση, δηλαδή δεν είναι απλά λογισμικό. Ιδανικά αυτός ο πράκτορας έχει την δυνατότητα να αξιοποιεί αισθητήρες όπως για παράδειγμα κάμερες και μικρόφωνα και έχει την δυνατότητα να κινείται στον χώρο να καταλαβαίνει την φυσική γλώσσα αλλά και να μπορεί να παράξει πληροφορία σε φυσική γλώσσα και να μπορεί να πραγματοποιεί εξερεύνηση επιλογών σχετικά με ενέργειες που μπορεί να πραγματοποιήσει ώστε να πετύχει έναν σκοπό. Με λίγα λόγια να μπορεί να λειτουργήσει όπως νοήμονες οργανισμοί σαν τους ανθρώπους και τα ζώα. Παρόλο που ο στόχος του general AI ίσως να απέχει πολύ από το να πραγματοποιηθεί έχει καταγραφεί μεγάλη πρόοδος σε πιο συγκεκριμένα προβλήματα που προκύπτουν στον πραγματικό κόσμο. Τέτοια προβλήματα έχει στόχο να λύσει και η Μηχανική Μάθηση/Βαθιά Μηχανική Μάθηση.

Βασικά συστατικά μέρη της Μηχανικής Μάθησης αποτελούν τα δεδομένα και οι αλγόριθμοι μάθησης **[34]** και αυτά τα δύο μέρη συμπληρώνουν το ένα το άλλο. Χωρίς δεδομένα οι αλγόριθμοι είναι άχρηστοι και χωρίς αλγορίθμους τα δεδομένα είναι άχρηστα. Τα τελευταία χρόνια παράγονται δεδομένα κάθε είδους από εικόνες και βίντεο μέχρι αλληλεπιδράσεις στα κοινωνικά δίκτυα και υπάρχουν πλέον διαθέσιμα είτε δημόσια είτε ιδιωτικά πολλά δεδομένα που έχουν προοπτική να αξιοποιηθούν.

Οι μέθοδοι της Μηχανικής Μάθησης στόχο έχουν να βρουν κρυμμένα μοτίβα (patterns) σε δεδομένα που αλλιώς θα ήταν αδύνατο να βρεθούν. Αυτά τα κρυμμένα μοτίβα και η γνώση χρησιμοποιούνται στην συνέχεια για να γίνονται προβλέψεις σε άγνωστα νέα δεδομένα. Οι περισσότεροι από εμάς δεν αντιλαμβανόμαστε ότι αλληλεπιδρούμε **[34]** καθημερινά με αλγορίθμους Μηχανικής Μάθησης. Κάθε φορά που κάνουμε μία αναζήτηση σε κάποια μηχανή αναζήτησης ή ανεβάζουμε κάποιο video ή φωτογραφία σε

κάποιο κοινωνικό δίκτυο ή κάνουμε μία συναλλαγή με πιστωτική κάρτα αλληλεπιδρούμε με κάποιον αλγόριθμο Μηχανικής Μάθησης.

Γενικά ένα πρόβλημα όταν είναι απλό, είναι εύκολο να δημιουργηθεί ένα πρόγραμμα που να εφαρμόζει μία λύση από κάποιον που γνωρίζει το αντικείμενο (domain expert). Για πολύπλοκα προβλήματα αντίθετα μπορεί να είναι πολύ δύσκολο έως αδύνατο ένας άνθρωπος να βρει μία λύση και να την υλοποιήσει σε έναν αλγόριθμο όσο καλά και αν γνωρίζει το αντικείμενο. Αντί για αυτό είναι πιο αποδοτικό ο υπολογιστής να προσπαθήσει να δημιουργήσει έναν αλγόριθμο από παραδείγματα αντί να προσπαθούμε να βρούμε λύση για κάθε βήμα του αλγορίθμου. Γενικά η Μηχανική Μάθηση χωρίζεται σε δύο κατηγορίες **[35]** αν και στην περίπτωση της μηχανικής μάθησης σε δεδομένα γράφων αυτός ο διαχωρισμός δεν είναι πλέον τόσο ευδιάκριτος **[1]**. Η πρώτη κατηγορία είναι η Επιβλεπόμενη Μηχανική Μάθηση **[36]** και η δεύτερη η μη Επιβλεπόμενη **[37]** και θα τις περιγράψουμε παρακάτω.

- Επιβλεπόμενη Μάθηση (Supervised Learning)

Πρόκειται για έναν από τους πιο απλούς τρόπους εκπαίδευσης ενός αλγορίθμου. Στην περίπτωση αυτή **[36]** έχουμε ένα σύνολο δεδομένων εκπαίδευσης τα οποία ο αλγόριθμος θα χρησιμοποιήσει για να εξάγει γνώση και στην συνέχεια να την γενικεύσει. Παρόλη την ευκολία και δύναμη που προσφέρουν τα δεδομένα με ετικέτες (labels) πρέπει να ξανα αναφέρουμε ότι η επιτυχία βασίζεται στην ποιότητα και ακρίβεια της διαδικασίας του annotation **[38]** για το οποίο υπάρχει μεθοδολογία και αποτελεί από μόνο του ξεχωριστό πρόβλημα **[38]**. Σαν παράδειγμα προβλημάτων αναφέρουμε την διαφωνία μεταξύ annotators και την περίπτωση ανίχνευσης λάθος annotation. Εκτός από αυτά η διαδικασία του annotation μπορεί να γίνει ιδιαίτερα κοστοβόρα. Υπάρχουν πολλές περιπτώσεις στις οποίες είναι πολύ δύσκολο να δημιουργηθεί ένα σύνολο δεδομένων το οποίο έχει προοπτική να αξιοποιηθεί και να δώσει καλά αποτελέσματα. Πέρα από αυτούς τους περιορισμούς και δυσκολίες η διαδικασία εξελίσσεται με τον αλγόριθμο κατά την φάση της εκπαίδευσης να βρίσκει πολύπλοκα μοτίβα μεταξύ των δειγμάτων τα οποία δεν είναι δυνατόν να βρεθούν χωρίς τους υπολογιστικούς πόρους τα οποία συσχετίζει με τις δοσμένες ετικέτες και δημιουργεί ένα μοντέλο για να εξηγήσει αυτές τις συσχετίσεις. Φυσικά αυτό αποτελεί έναν γενικό ορισμό και ο κάθε αλγόριθμος χρησιμοποιεί

διαφορετικές τακτικές για να λύσει το πρόβλημα. Παρόλη την δύναμη της τεχνικής αυτής μπορούν να παρουσιαστούν προβλήματα με πιο χαρακτηριστικά αυτά του overfitting, της έλλειψης επαρκών δεδομένων εκπαίδευσης, του θορύβου στα δεδομένα , και του curse of dimensionality για να αναφέρουμε τα πιο γνωστά. Σε αυτά τα προβλήματα θα αναφερθούμε στην συνέχεια. Τέλος μετά την επιτυχή εκπαίδευση του μοντέλου ακολουθεί η διαδικασία της ανάπτυξης (deployment) του συστήματος. Αυτό το τελευταίο στάδιο είναι πολύ σημαντικό και μπορεί να καθορίσει την επιτυχία ή την αποτυχία του εγχειρήματος. Παρόλο που μπορεί να έχει πετύχει κάποιος αποδεκτά αποτελέσματα όσον αφορά μετρικές επίδοσης μπορεί το μοντέλο να έχει την απαίτηση να λειτουργεί σε πραγματικό χρόνο οπότε αν ο χρόνος επεξεργασίας ξεπεράσει τον ρυθμό που φτάνουν δεδομένα τότε το μοντέλο δεν λειτουργεί σωστά. Επίσης το overhead σε υπολογιστικούς πόρους μπορεί να είναι σημαντική παράμετρος. Αυτό ισχύει ειδικά στην περίπτωση deep μοντέλων νευρωνικών δικτύων που έχουν την απαίτηση να λειτουργούν στο edge του δικτύου ή σε κινητές συσκευές η σε microcontrollers. Τα frameworks που είναι διαθέσιμα έχουν προσπαθήσει να αντιμετωπίσουν τα παραπάνω προβλήματα και ως έναν βαθμό το έχουν καταφέρει αλλά εξαρτάται από την περίπτωση. Τέλος είναι αυτονόητο ότι το μοντέλο πρέπει να τροφοδοτείται με δεδομένα ίδιου τύπου με αυτά που εκπαιδεύτηκε. Το αν τελικά είναι εφικτό και εύκολο να βρούμε αυτά τα δεδομένα είναι ένα θέμα που εξετάζεται ξεχωριστά οπότε πολλές φορές η επιλογή χαρακτηριστικών (feature selection) **[39]** είναι προαπαιτούμενο και έχει πολλές φορές σκοπό να διευκολύνει την διαδικασία αυτή πέρα από την συνεισφορά των χαρακτηριστικών αυτών στην απόδοση του αλγορίθμου.

- Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning)

Η μέθοδος της μη επιβλεπόμενης μάθησης **[37]** έχει το πλεονέκτημα του ότι δεν χρειάζεται την ακριβή διαδικασία του annotation. Αντί να συσχετίζει χαρακτηριστικά των δειγμάτων με ετικέτες και έχει στόχο έχει να βρει μοτίβα που χαρακτηρίζουν τα δείγματα χρησιμοποιώντας τις έννοιες της ομοιότητας και της απόστασης **[40]**. Με αυτόν τον τρόπο βρίσκει παρόμοια χαρακτηριστικά που είναι αντιπροσωπευτικά των δειγμάτων με βάση τα οποία μπορεί να διαχωρίσει τα δεδομένα σε ομάδες. Η διαδικασία που περιγράφηκε είναι η πλέον αντιπροσωπευτική και ονομάζεται συσταδοποίηση (clustering) **[41]**. Η διαδικασία της συσταδοποίησης είναι μία ενότητα από μόνη της και

θα μπορούσαμε να πούμε πολλά περισσότερα αλλά δεν είναι σκοπός της εργασίας. Γενικά πάντως αποτελεί την πρώτη λύση στην διαδικασία exploratory data analysis **[42]**. Όσο για το πλήθος των αλγορίθμων συσταδοποίησης αποτελείται από δεκάδες έως εκατοντάδες **[41]**. Παρόλο που η ομαδοποίηση των δεδομένων είναι η προφανής εφαρμογή αυτής της μεθόδου, μπορεί να χρησιμοποιηθεί σε προβλήματα classification **[43]**, anomaly detection, outlier detection όπου εμέσως κατηγοριοποιεί ένα δείγμα ώς συνηθισμένο ή απόκλιση στο τομέα της ανίχνευσης απάτης σε συναλλαγές ή στην ανίχνευση μη συνηθισμένης συμπεριφοράς ενός κόμβου ή ενός χρήστη σε ένα δίκτυο ή πληροφοριακό σύστημα. Άλλες χρήσεις **[17]** είναι η εύρεση κοινοτήτων σε δεδομένα γράφων και άλλα. Τέλος αξίζει να κάνουμε αναφορά στην εφαρμογή της μείωσης διαστάσεων (dimensionality reduction) **[44]**. Σε αυτή την χρήση και με άμεση σχέση με το πρόβλημα μείωσης διάστασης υπάρχει η δυνατότητα να αναπαραστήσουμε τα δεδομένα σε πολύ μικρότερες διαστάσεις από τις αρχικές χάνοντας ελεγχόμενο μέγεθος πληροφορίας καθώς και για σκοπούς οπτικοποίησης (visualization). Αυτό μπορεί να επιτευχθεί χωρίς επίβλεψη με πιο γνωστή μέθοδο την PCA **[45]**. Η μέθοδος αυτή έχει ως στόχο να μειώσει την επανάληψη της πληροφορίας καθώς και να συμπιέσει το σύνολο δεδομένων (dataset). Αποτελεί γραμμικό μετασχηματισμό στα δεδομένα και αναπαράσταση σε νέο χώρο όπου τα principal components είναι ορθογώνια. Με αυτό τον τρόπο κρατώντας το πρώτο principal component, το οποίο έχει το μέγιστο variance δηλαδή φέρει την μεγαλύτερη πληροφορία, έχουμε μεγάλο μέρος της πληροφορίας σε μία διάσταση. Στην συνέχεια κρατώντας το δεύτερο προσθέτουμε το μεγαλύτερο μέρος της εναπομείνασας πληροφορίας και ούτω καθεξής. Άλλη μία μέθοδος μείωσης της διάστασης είναι η SVD **[46]**. Τέλος αξίζει να αναφέρουμε τα νευρωνικά δίκτυα Autoencoders **[47]**. Η μέθοδος αυτή λειτουργεί ως εξής **[47]**. Πρώτα ο κωδικοποιητής (encoder) μειώνει την διάσταση της εισόδου περνώντας την από ένα feed forward δίκτυο μέχρι να φτάσει στο ενδιάμεσο επίπεδο (bottleneck) το οποίο αποτελεί την αναπαράσταση σε μικρότερη διάσταση. Η αναπαράσταση αυτή στην συνέχεια περνάει στον decoder ο οποίος προσπαθεί να ανακατασκευάσει την αρχική πληροφορία. Το αποτέλεσμα μετά την εκπαίδευση είναι να έχουμε έναν κωδικοποιητή ο οποίος παράγει σωστές αναπαραστάσεις σε μικρή διάσταση.

2.2 Όροι σχετικοί με γράφους

- **Γράφος**

Ένας γράφος αποτελείται από ένα σύνολο κόμβων και ακμών.

- **Κατευθυνόμενος/ Μη κατευθυνόμενος γράφος**

Ένας γράφος είναι μη κατευθυνόμενος (digraph) όταν οι σχέσεις(ακμές) που συνδέουν τους κόμβους είναι συμμετρικές και κατευθυνόμενος οταν δεν είναι.

- **Bipartite**

Ένας γράφος λέγεται bipartite όταν οι κόμβοι χωρίζονται σε δύο κατηγορίες και δεν υπάρχουν ακμές μεταξύ κόμβων της ίδιας ομάδας.

- **Multigraph**

Ένας γράφος όπου μπορούν να υπάρχουν πολλές διαφορετικές σχέσεις(ακμές) ανάμεσα στους κόμβους.

- **Attributed Graph**

Όταν οι κόμβοι περιλαμβάνουν και χαρακτηριστικά (features).

- **Knowledge graph**

Ονομάζεται ένας γράφος multi digraph με labels στις ακμές που αναπαριστούν τον τύπο της σχέσης.

- **Adjacency Matrix**

Πίνακας που αναπαριστά τις ακμές του γράφου. Συμμετρικός για μή κατευθυνόμενους γράφους.

- **Degree Matrix**

Διαγώνιος πίνακας που στα στοιχεία της διαγωνίου έχει τον αριθμό των ακμών του κάθε κόμβου.

- **Laplacian Matrix**

Υπάρχουν πολλές εκδοχές αυτού του πίνακα **[48]**. Η πιο συνηθισμένη είναι L = D - A. Όπου D ο degree πίνακας και A ο Adjacency.

- **CTDG**

Δυναμικός Γράφος συνεχούς χρόνου **[3]**. Αναπαρίσταται ως ένας αρχικός γράφος και αλλαγές(γεγονότα) σε αυτόν τα οποία συνοδεύονται από ένα timestamp συνεχών τιμών.

- **DTDG**

Δυναμικός Γράφος διακριτού χρόνου **[3]**. Αναπαρίσταται ως σύνολο στιγμιοτύπων του γράφου σε συνήθως ίδιου εύρους χρονικά πεδία.

- **Higher order Proximity Matrix**

Το Adjacency Matrix παρέχει πληροφορία μόνο για το αν υπάρχει σύνδεση μεταξύ δύο κόμβων. Πολλές φορές ορίζονται άλλα μέτρα ομοιότητας **[3]** μεταξύ των κόμβων τα οποία μπορούν από μόνα τους να χρησιμοποιηθούν σαν αναπαραστάσεις. Κάποια από τα πιο συνηθισμένα **[49]** είναι τα Common Neighbors Matrix, Jaccard's Coefficient, Adamic Adar, Katz Index.

2.3 Ιδιαιτερότητες Μηχανικής Μάθησης για δεδομένα Γράφων

Οι βασικές ιδιαιτερότητες που προκύπτουν από δεδομένα γράφων στην μηχανική Μάθηση είναι δύο. Η πρώτη **[50]** αφορά την επιπλέον πληροφορία που καλούμαστε να ενσωματώσουμε στην διαδικασία μάθησης δηλαδή την συνδεσιμότητα μεταξύ των δειγμάτων που αναλύουμε η οποία έχει μη καθορισμένη δομή σε αντίθεση με άλλα σενάρια Μηχανικής Μάθησης. Η δεύτερη **[1]** είναι το σενάριο μάθησης που παρουσιάζεται πολύ συχνά το οποίο ονομάζεται ημι επιβλεπόμενη μάθηση (semi supervised learning). Σε αυτό το σενάριο **[1]** υπάρχουν δύο βασικές διαδικασίες. Η πρώτη αφορά την δημιουργία ενός γράφου από τα δείγματα και η δεύτερη αφορά την πρόβλεψη της κλάσης κάθε δείγματος που δεν έχει ετικέτα (label). Για την πρώτη φάση δηλαδή την δημιουργία του γράφου δεν θα πούμε πολλά σε αυτή την εργασία διότι όπως θα δούμε στην συνέχεια στο σύνολο δεδομένων που έχουμε ο γράφος είναι έτοιμος. Απλά σαν σύντομη αναφορά μπορούμε να πούμε ότι υπάρχουν και μη επιβλεπόμενοι και επιβλεπόμενοι τρόποι για να γίνει αυτό. Πιο συνηθισμένα σενάρια είναι αυτά των μη επιβλεπόμενων μεθόδων με πιο γνωστούς τους KNN και e-neighborhood. Στον τομέα αυτό γίνεται έρευνα **[17]** για πιο αποδοτικούς τρόπους σε σχέση με τους παραδοσιακούς Στην συνέχεια περιγράφουμε αναλυτικά τις ιδιαιτερότητες αυτές.

2.3.1 Μη ευκλείδεια δεδομένα

Τις περισσότερες περιπτώσεις στην Βαθιά Μηχανική μάθηση τα δεδομένα που έχουμε στην διάθεση μας έχουν συγκεκριμένη δομή. Για παράδειγμα **[21]** στον τομέα της υπολογιστικής όρασης (computer vision) είτε για δεδομένα video είτε για εικόνες η δομή των δεδομένων είναι ίδια. Το κάθε εικονοστοιχείο (pixel) έχει συγκεκριμένο αριθμό γειτόνων σε ίδια πάντα απόσταση. Στον τομέα της επεξεργασίας φυσικής γλώσσας (Natural Language Processing) ισχύει το ίδιο. Αυτές οι περιπτώσεις δεδομένων αναφέρονται στην βιβλιογραφία ως Ευκλείδεια δεδομένα **[21]** και με λίγα λόγια έχουν συγκεκριμένη δομή. Αντίθετα με αυτά στα δεδομένα γράφων ο κάθε κόμβος μπορεί να έχει οποιοδήποτε αριθμό γειτόνων ο καθένας με άλλα χαρακτηριστικά και σε διαφορετική απόσταση δηλαδή η σχέση να περιλαμβάνει βάρη. Επίσης πολλές φορές παρουσιάζεται η περίπτωση όπου έχουμε διαφορετικού τύπου σχέση ανάμεσα σε κάποιους κόμβους δηλαδή να υπάρχουν παραπάνω από ένα είδος σχέσης. Επίσης το κάθε datapoint (εικόνα) θεωρείται ανεξάρτητο από τα άλλα για ένα CNN κάτι που δεν ισχύει για κόμβους γράφων στους οποίους υπάρχουν σχέσεις μεταξύ των κόμβων. Επίσης άλλη μία ιδιαιτερότητα που παίζει ρόλο είναι το μέγεθος των δεδομένων **[19]**. Στην πρώτη περίπτωση για παράδειγμα μία εικόνα έχει μικρό μέγεθος αν θεωρήσουμε το κάθε pixel σαν έναν κόμβο και ακόμα και αν η ανάλυση είναι μεγάλη(κάτι που δεν συνηθίζεται) και πάλι το μέγεθος θεωρείται μικρό. Αντίθετα ένας γράφος κοινωνικών δικτύων μπορεί να έχει εκατομμύρια κόμβους και δισεκατομμύρια ακμές κάτι που μπορεί να αποτελέσει πρόβλημα ειδικά για τους αλγορίθμους spectral **[22]** οι οποίοι βασίζονται σε πολύπλοκους και χρονοβόρους υπολογισμούς πάνω στους πίνακες. Ο όρος Geometric deep Learning **[21]** αναφέρεται στον τομέα αυτό που έχει στόχο να γενικεύσει υπάρχουσες τεχνολογίες Βαθιάς Μηχανικής Μάθησης σε δεδομένα χωρίς σταθερή δομή όπως περιγράφηκαν παραπάνω.

2.3.2 Ημι Επιβλεπόμενη Μηχανική Μάθηση (Semi supervised learning)

Αυτό το σενάριο μάθησης **[1,2,17]** αποτελεί μία έννοια που δεν παρουσιάζεται συχνά στο τομέα της Μηχανικής Μάθησης. Βασικό χαρακτηριστικό του σεναρίου αυτού το οποίο καλούμαστε να αντιμετωπίσουμε και στο σενάριο χρήσης αυτής της εργασίας είναι η

χρήση όλης της πληροφορίας κατά την διάρκεια της εκπαίδευσης, δηλαδή όλων των ακμών του γράφου, όλων των κόμβων και όλων των χαρακτηριστικών που τους συνοδεύουν αλλά στον υπολογισμό του σφάλματος εκπαίδευσης λαμβάνονται υπόψη μόνο τις ετικέτες ενός μικρού ποσοστού των διαθέσιμων που έχουμε **[5,8]**. Αυτό συμβαίνει κυρίως γιατί στις περισσότερες περιπτώσεις ο στόχος είναι να κατηγοριοποιήσουμε όλους τους κόμβους του δικτύου που δεν έχουν ετικέτα και η αναλογία κόμβων χωρίς ετικέτα έναντι αυτών που έχουν ετικέτα είναι πολύ μεγάλη. Επίσης πολλές φορές δεν είναι εφικτό ή είναι πολύ απαιτητική η διαδικασία του annotation. Με βάση αυτό χρησιμοποιούμε **[5,8]** όσο λιγότερα δείγματα με ετικέτα μπορούμε ώστε το αποτέλεσμα της πρόβλεψης στο σύνολο δεδομένων αξιολόγησης (test set) να έχει μεγαλύτερη βεβαιότητα. Άλλος ένας λόγος είναι ότι θέλουμε να βασιστούμε περισσότερο στην πληροφορία που μας δίνει η δομή του γράφου. Με βάση τα παραπάνω ο διαχωρισμός των δεδομένων σε train/validation/test δεν είναι αυτός που πραγματοποιείται συνήθως σε σενάρια μηχανικής μάθησης **[51]**.

Άλλη μία ιδιαιτερότητα που προκύπτει είναι ότι οι αλγόριθμοι που επιλέγουμε για το σενάριο χρήσης αυτής της εργασίας είναι ιδανικοί για σενάρια inductive learning **[1,5,19]**. Παρόλο που τους επιλέγουμε για αυτό το χαρακτηριστικό τους, το πρωτεύον σενάριο που αντιμετωπίζουμε σε αυτή την εργασία είναι αυτό της ημι επιβλεπόμενης μάθησης (semi supervised learning) σε όλα τα διαθέσιμα δεδομένα σενάριο που δεν εμφανίζεται συχνά για τους αλγορίθμους inductive στην βιβλιογραφία **[1]**. Το πιο συνηθισμένο σενάριο που εμφανίζεται είναι αυτο όπου όλοι οι κόμβοι εκπαίδευσης έχουν ετικέτα και η αξιολόγηση γίνεται σε καινούριους γράφους ή μέρη του αρχικού γράφου που κρατήθηκε εκτός εκπαίδευσης για αυτό το σκοπό.

2.4 Περιγραφή κύριων στόχων Μηχανικής Μάθησης σε δεδομένα Γράφων

Οι κύριοι στόχοι για τον τομέα αυτό βασίζονται στα περισσότερα στις τεχνικές Μηχανικής Μάθησης αλλά προκύπτουν και κάποια επιπλέον σενάρια τα οποία σχετίζονται με τους γράφους μόνο. Η κατηγοριοποίηση σε επιβλεπόμενη και μη

επιβλεπόμενη γενικά ισχύει και εδώ. Στην συνέχεια καταγράφουμε όλα τα σενάρια που συνήθως παρουσιάζονται στον τομέα αυτό.

Ημι Επιβλεπόμενη μάθηση σε γράφους (Graph Semi-supervised Learning)

Στον τομέα του Graph Representation Learning **[2]** όπως αναφέραμε προηγουμένως ισχύουν και εδώ οι όροι επιβλεπόμενη και μη επιβλεπόμενη μάθηση αλλά τα όρια μεταξύ τους δεν είναι ξεκάθαρα **[1]**. Το πιο συνηθισμένο σενάριο είναι αυτό της ημι επιβλεπόμενης μάθησης (semi-supervised learning) **[1, 2]**. Αυτό το σενάριο δεν αφορά μόνο γράφους **[52]**. Για την περίπτωση των γράφων στα περισσότερα σενάρια έχουμε μόνο έναν πολύ μικρό αριθμό από δεδομένα με ετικέτα (labeled data) τα οποία είναι ή ετικέτες για κόμβους ή για ακμές και έναν πολύ μεγάλο αριθμό από δεδομένα χωρίς ετικέτα (unlabeled data) τα οποία θέλουμε να κατηγοριοποιήσουμε **[1]**. Το πρόβλημα αντιμετωπίζεται εκπαιδεύοντας το μοντέλο λαμβάνοντας υπόψη εκτός από τους κόμβους/ακμές με ετικέτα (labeled) και την πληροφορία από την δομή του γράφου που αφορά όλους τους κόμβους καθώς και τα χαρακτηριστικά (features) των κόμβων/ακμών αν αυτά υπάρχουν. Με αυτό το τρόπο και παρόλο που στον υπολογισμό του σφάλματος (loss function) χρησιμοποιούνται μόνο λίγα από τα διαθέσιμα δεδομένα με ετικέτα επιτυγχάνεται η εκπαίδευση σε αντίθεση με σενάρια κλασικής Μηχανικής Μάθησης **[34]** όπου θεωρείτε ότι τα δείγματα (data points) είναι ασυσχέτιστα. Γενικά όπως και στην κλασική περίπτωση ενός προβλήματος Μηχανικής Μάθησης χωρίζουμε το σύνολο δεδομένων σε train/validation/test αλλά εδώ τα labeled data στο train είναι ένα πολύ μικρό ποσοστό των συνολικών, αυτά στο validation είναι ένα μεγαλύτερο ποσοστό και τέλος αυτά στο test είναι το μεγαλύτερο ποσοστό **[5,8]**.

Αυτο επιβλεπόμενη μάθηση σε Γράφους (Graph self supervised Learning)

Αυτός ο όρος **[24]** αναφέρεται στην διαδικασία εκμάθησης αναπαραστάσεων χωρίς την χρήση ετικετών (labels). Ο όρος προκύπτει από το γεγονός του ότι βελτιστοποιούνται οι αναπαραστάσεις με βάση κάποιον optimizer ώστε να ανακατασκευάζεται ένα μέτρο ομοιότητας **[24,49]** οπότε τεχνικά έχουμε ετικέτες (labels). Αυτό το μέτρο ομοιότητας που προσπαθεί να ανακατασκευάσει μπορεί να είναι το Adjacency Matrix ή κάποιο higher order proximity matrix **[49]** και άλλα.

Επιβλεπόμενη μάθηση σε Γράφους (Graph Supervised Learning)

Σε αυτήν την κατηγορία **[1,17]** έχουμε τρία σενάρια στόχων. Αυτά είναι η κατηγοριοποίηση κόμβων (node classification), η πρόβλεψη κάποιας σχέσης που θα προκύψει μελλοντικά (link prediction) και η κατηγοριοποίησης ολόκληρων γράφων (graph classification) τα οποία περιγράφουμε στην συνέχεια.

- **Κατηγοριοποίηση Κόμβων (Node Classification)**

Πρόκειται για το πιο συνηθισμένο πρόβλημα **[2]** και αφορά και την εργασία μας. Εδώ θέλουμε να κατηγοριοποιήσουμε έναν κόμβο. Η εκπαίδευση μπορεί να γίνει από άκρο σε άκρο (end to end) δηλαδή να εκπαιδεύσουμε τον κωδικοποιητή (encoder)για να παράγει αναπαραστάσεις (embendings) και τον κατηγοριοποιητή (classifier) για να κάνει την κατηγοριοποίηση ταυτόχρονα. Αποτελεί το πιο συνηθισμένο πρόβλημα για γράφους και βρίσκει εφαρμογή σε πολλούς τομείς **[53]**.

- **Πρόβλεψη σχέσης (Link Prediction)**

Εδώ **[2]** στόχος είναι να βγάλουμε συμπέρασμα για το αν υπάρχει ή πρόκειται να εμφανιστεί κάποια σχέση μεταξύ δύο κόμβων. Και πάλι για την εκπαίδευση χρειαζόμαστε σενάριο επιβλεπόμενης μάθησης. Εδώ στα περισσότερα σενάρια **[54]** που παρουσιάζονται στην βιβλιογραφία εκπαιδεύεται μία συνάρτηση (function) η οποία δέχεται ως είσοδο δύο αναπαραστάσεις (embendings) και έχει ως έξοδο μία τιμή που αποτελεί το μέτρο ή την πιθανότητα να προκύψει μία σχέση ανάμεσα σε αυτούς τους κόμβους. Η συνάρτηση αυτή μπορεί να είναι ένα νευρωνικό δίκτυο. Είναι πρόβλημα που βρίσκει εφαρμογή σε περιπτώσεις **[2]** όπως πρόβλεψη πιθανών γνωστών σε κοινωνικά δίκτυα ή ομάδων και σελίδων που μπορεί να ενδιαφέρουν έναν χρήστη.

- **Κατηγοριοποίηση Γράφου (Graph Classification)**

Εδώ **[2]** μας ενδιαφέρει να προβλέψουμε την κλάση ενός ολόκληρου γράφου ή κάποιο χαρακτηριστικό του. Και πάλι για την εκπαίδευση χρειαζόμαστε σενάριο επιβλεπόμενης μάθησης. Επίσης χρησιμοποιείται μία συνάρτηση η οποία στην βιβλιογραφία αναφέρεται **[55]** με τον όρο Readout function η οποία συνδυάζει αναπαραστάσεις από κόμβους ή/και ακμές με στόχο να παράξει μία αναπαράσταση για όλο το γράφο και να την χρησιμοποιήσει σε έναν classifier. Ένα παράδειγμα τέτοιας συνάρτησης είναι το mean των αναπαραστάσεων των κόμβων ή των ακμών σε συνδυασμό με μία συνάρτηση ενεργοποίησης (activation function). Αυτό το σενάριο βρίσκει εφαρμογή σε περιπτώσεις όπου για παράδειγμα οι γράφοι είναι μόρια και στόχος είναι να προβλεφθεί κάποια

ιδιότητα τους. Επίσης βρίσκει εφαρμογή στον αλγόριθμο DGI **[13]** που θα εξετάσουμε.

Μη επιβλεπόμενη μάθηση σε Γράφους (Graph Unsupervised Learning)

- **Συσταδοποίηση και Εύρεση κοινοτήτων (Clustering/Community Detection)**

Εδώ **[2]** ο στόχος είναι να συγκεντρώσουμε μαζί κόμβους που έχουν κοινά χαρακτηριστικά και μπορούν να οργανωθούν σε ομάδες. Η έννοια αυτή δεν αφορά μόνο γράφους και είναι η πιο γνωστή προσέγγιση μη επιβλεπόμενης μάθησης. Η έννοια της εύρεσης κοινοτήτων αποτελεί υποπερίπτωση **[56]** της έννοιας συσταδοποίηση και προκύπτει για σενάρια χρήσης σε κοινωνικά δίκτυα.

2.5 Παρουσίαση και επεξήγηση βασικών αλγορίθμων μηχανικής μάθησης σε Γράφους

Σε αυτή την ενότητα θα αναφερθούμε σε αλγορίθμους μηχανικής μάθησης σε δεδομένα γράφων με βάση την βιβλιογραφία και θα εξηγήσουμε τις βασικές κατηγορίες που προκύπτουν και τα χαρακτηριστικά της κάθε μιας. Δεν είναι στόχος της εργασίας να απαριθμήσει όλους τους αλγορίθμους που υπάρχουν και για αυτό θα παρουσιάσουμε ήδη υπάρχουσα ταξινομία. Πριν αναφερθούμε στους αλγορίθμους θα κάνουμε μία αναφορά στους όρους που έχουν επικρατήσει για την περιγραφή **[2,4,5]** των αλγορίθμων μηχανικής μάθησης σε γράφους.

2.5.1 Μοντέλο αναπαράστασης Γράφων

Εδώ θα περιγράψουμε τους όρους που έχουν επικρατήσει για την περιγραφή μοντέλων GRL **[2,4,5]**. Το μοντέλο έχει δύο βασικά μέρη. Αυτά είναι ο κωδικοποιητής (encoder) και ο αποκωδικοποιητής (decoder). Ο ρόλος του encoder είναι ο πιο σημαντικός και το μεγαλύτερο μέρος της βιβλιογραφίας **[2,4,5]** ασχολείται με αυτό. Ο encoder είναι ένα μία συνάρτηση που μπορεί να υλοποιηθεί με πολλές τεχνολογίες και σαν είσοδο έχει κόμβους και ακμές για τα shallow **[2,4]** μοντέλα και για τα deep **[5]** χρησιμοποιούνται και τα χαρακτηριστικά (features) του κάθε κόμβου ή/και ακμής. Η έξοδος του είναι ένα

διάνυσμα (vector) ή ένας πίνακας (matrix) ή ένα βαθμωτό μέγεθος (scalar) ή κάποιος συνδυασμός από αυτά για κάθε κόμβο ή/και ακμή. Ο decoder συνήθως έχει ως είσοδο δύο αναπαραστάσεις (embendings), αν και αυτό δεν είναι απόλυτο όπως για παράδειγμα οι τεχνολογίες Autoencoders οι οποίοι χρησιμοποιούν μόνο μία αναπαράσταση. Τα δύο αυτά συστατικά μέρη (components) μπορούν να εκπαιδευτούν ταυτόχρονα (end to end). Ο encoder μπορεί να είναι **[4]** ένας πίνακας αναζήτησης σε αναπαραστάσεις που βελτιστοποιήθηκαν ανεξάρτητα για κάθε κόμβο έως και ένα πολύπλοκο Νευρωνικό δίκτυο με πολλά κρυμμένα επίπεδα **[5]**. Ο decoder για την περίπτωση των Νευρωνικών είναι ένας MLP. Επίσης πολλές φορές συνδυάζονται encoders βασισμένοι σε Νευρωνικά και ο decoder ένα shallow μοντέλο ML σενάριο που θα εξετάσουμε και σε αυτήν την εργασία.

2.5.2 Αλγόριθμοι Μηχανικής Μάθησης σε γράφους

Όπως αναφέραμε στο προηγούμενο οι αλγόριθμοι αφορούν ζεύγη encoders decoders. Εδώ θα παρουσιάσουμε τους κύριους **[4,5]** αλγορίθμους για encoders. Αυτοί χωρίζονται σε δύο κατηγορίες shallow και deep. Η κατηγορία των deep **[5]** χωρίζεται επίσης με βάση το κριτήριο **[5]** σε ποιά τεχνολογία βασίζεται αλλά και στο αν βασίζει τους υπολογισμούς για την διαδικασία της εκπαίδευσης των αναπαραστάσεων του κάθε κόμβου στον πίνακα γειτνίασης ή κάποιον Laplacian ή την βάση Fourier του Laplacian ή βασίζει τους υπολογισμούς στην άμεση γειτονιά του κάθε κόμβου. Περισσότερα για τα deep μοντέλα στην συνέχεια.

2.5.2.1 Αλγόριθμοι κλασικής μηχανικής μάθησης(shallow models)

Αυτά τα μοντέλα **[4]** αποτέλεσαν τις πρώτες λύσεις για το πρόβλημα αναπαράστασης δεδομένων γράφων. Περιγράφονται από τέσσερα μέρη **[4]**. Αυτά είναι ο κωδικοποιητής (encoder), ο αποκωδικοποιητής (decoder), η συνάρτηση σφάλματος (loss function) και το μέτρο ομοιότητας (similarity) που χρησιμοποιούν. Χωρίζονται γενικά με βάση το μέτρο ομοιότητας που χρησιμοποιούν σε δύο κατηγορίες **[2,4]**. Αυτές είναι Matrix Factorization και Random walk. Και στις δύο αυτές περιπτώσεις οι αναπαραστάσεις για

κάθε κόμβο βελτιστοποιούνται ξεχωριστά δηλαδή δεν έχουμε parameter sharing.

Για την περίπτωση Matrix Factorization **[4,57]** το μέτρο ομοιότητας είναι ντετερμινιστικό (deterministic) και η βελτιστοποίηση (optimization) πραγματοποιείται κατευθείαν στις αναπαραστάσεις και βασίζεται σε κάτι στατικό. Ο αποκωδικοποιητής (decoder) τις περισσότερες φορές είναι το εσωτερικό γινόμενο (dot product) δύο αναπαραστάσεων και η συνάρτηση σφάλματος (loss function) κάποια απόσταση ανάμεσα στην πρόβλεψη και την πραγματική τιμή (ground truth).
Αυτές οι μέθοδοι περιλαμβάνουν τους αλγορίθμους όπως 1.Laplacian Eigenmaps 2.Matrix Factorization 3.GraRep 4.Hope. Αυτοί οι αλγόριθμοι διαφέρουν στο πως ορίζουν κωδικοποιητή/αποκωδικοποιητή/μέτρο ομοιότητας και συνάρτηση σφάλματος.

Για την περίπτωση των Random Walks **[58]** το μέτρο ομοιότητας που χρησιμοποιείται είναι στοχαστικό και μη συμμετρικό και βασίζεται στην πιθανότητα που προκύπτει δύο κόμβοι να προκύψουν σε ένα μονοπάτι μεταβάσεων σταθερού μήκους. Η συνάρτηση σφάλματος είναι η cross entropy loss. Ένας από τους πιο γνωστούς αλγορίθμους που βασίζονται σε Random Walks είναι ο node2vec **[59]** τον οποίο θα χρησιμοποιήσουμε σε αυτή την εργασία και θα τον περιγράψουμε αναλυτικά σε επόμενη ενότητα.

2.5.2.2 Αλγόριθμοι Βαθιάς Μηχανικής Μάθησης

Όπως αναφέραμε και πριν χωρίζονται με βάση την τεχνολογία που χρησιμοποιούν και το αν είναι μεταδοτικοί (transductive) ή επαγωγικοί (inductive). Εδώ τα παρουσιάζουμε όπως στο **[5]** με βάση την τεχνολογία. Οι κατηγορίες είναι τρείς και θα τις παρουσιάσουμε στην συνέχεια. Σε κάθε περίπτωση θα αναφέρουμε αν είναι transductive ή inductive. Αυτοι οι όροι παρουσιάζονται πολύ συχνά **[1,5,6,12,19,13]** οπότε τους χρησιμοποιούμε για κατηγοριοποιήσουμε τους αλγορίθμους. Οι inductive αλγόριθμοι μπορούν να χρησιμοποιηθούν σε καινούριους γράφους με χαρακτηριστικά (features) ίδια με τον γράφο που εκπαιδεύτηκαν. Οι transductive δεν έχουν αυτή τη δυνατότητα και βασίζουν όλους τους υπολογισμούς στον πίνακα γειτνίασης ή τον Laplacian. Σε περίπτωση νέων δεδομένων, δηλαδή με προσθήκες κόμβων ή/και ακμών οι πρώτοι μπορούν να παράξουν **[6,14,19]** αποτελέσματα ενώ για τους transductive πρέπει να

εκπαιδευτεί **[4]** από την αρχή το μοντέλο ή να πραγματοποιηθεί μία συμπληρωματική εκπαίδευση. Επίσης οι inductive μπορούν να χειριστούν καλύτερα σενάρια **[5,19]** με πολύ μεγάλους γράφους διότι βασίζονται σε υπολογισμούς στην γειτονιά του κάθε κόμβου. Η λίστα των αλγορίθμων δεν είναι εξαντλητική και προέκυψε από την βιβλιογραφία **[1,2,4,5]**. Επίσης πολλές φορές υπάρχει και κατηγοριοποίηση spectral και non spectral (spatial) **[6,22]**. Με βάση αυτό **[22]** μπορούμε να χρησιμοποιήσουμε τους όρους αυτούς για όλα τα convolutional και attention μοντέλα όπως περιγράφονται εδώ **[5]**. Το αν μπορούμε να το χρησιμοποιήσουμε και για τα όλα τα μοντέλα Autoencoders δεν είναι σαφές. Είναι προφανές με βάση το προηγούμενο ότι οι spectral **[22]** τεχνικές βασίζονται σε υπολογισμούς όπου χρησιμοποιείται συνήθως το eigendecomposition του Laplacian και η συνέλιξη ορίζεται στο πεδίο του μετασχηματισμού Fourier του Laplacian **[5]**. Αντίθετα στις non spectral(spatial) **[6,14,19]** τεχνικές οι υπολογισμοί βασίζονται στην γειτονιά του κάθε κόμβου.

Στην συνέχεια παρουσιάζουμε συνοπτικά τους αλγορίθμους βαθιάς μηχανικής μάθησης με βάση την ταξινομία **[5]** που τους χωρίζει με βάση την τεχνολογία. Οι τρεις κατηγορίες είναι Συνελικτικός Μηχανισμός (Convolutional), μηχανισμός προσοχής(attention) και νευρωνικά δίκτυα Autoencoders.

2.5.2.2.1 Συνελικτικός Μηχανισμός(Convolutional)

Εδώ παρουσιάζουμε τους αλγορίθμους Νευρωνικών δικτύων που βασίζονται στην έννοια της συνέλιξης.

- ChebNet

Το ChebNet **[7]** υλοποιεί την συνέλιξη στον γράφο που δέχεται σαν είσοδο χρησιμοποιώντας πολυώνυμα Chebyshev. Χρησιμοποιώντας K πολυώνυμα στην συνέλιξη έχουμε πληροφορία από K-hops απόσταση για το τρέχοντα κόμβο που επεξεργαζόμαστε. Ο αλγόριθμος αυτός βασίζεται στην εκπαίδευση σε κάποια μορφή του Laplacian πίνακα και βασίζει τις πράξεις εκεί συνεπώς κατατάσσεται στους transductive δηλαδή δεν μπορεί να γενικεύσει και να δώσει αναπαραστάσεις για νέους γράφους.

- Graph Convolutional Network

Βασίζεται **[8]** στο παραπάνω αλλά θέτει k=1 και με κάποιες άλλες αλλαγές και χρησιμοποιώντας πολλά επίπεδα στο δίκτυο που δημιουργείται καταφέρνει να έχει

πληροφορία από k-hops γειτονιά κάθε κόμβου που επεξεργάζεται όπως και ο προηγούμενος. Βασίζει τους υπολογισμούς σε κάποια μορφή του Laplacian στην διαδικασία της εκπαίδευσης συνεπώς ανήκει και αυτός στην κατηγορία transductive.

- APPNP

Συνδυάζει **[9]** το GCN με τον αλγόριθμο PageRank **[60]**. Βασίζεται στο GCN συνεπώς ανήκει στους transductive.

- SGC

Ένα πολύ απλό μοντέλο **[10]** αλλά δίνει καλά αποτελέσματα. Αποτελείται από softmax της k-δύναμης του πίνακα γειτνίασης επι τα features επι παραμετρους που μαθαίνονται. Λειτουργεί με πράξεις που βασίζονται στον πίνακα γειτνίασης στην εκπαίδευση άρα είναι transductive.

- GraphSAGE

Αυτός ο αλγόριθμος **[19]** λειτουργεί ως εξής. Ο αλγόριθμος χρησιμοποιεί τα features του κάθε κόμβου και πληροφορία από τις αναπαραστάσεις των γειτονικών κόμβων. Εκπαιδεύει k aggregators έναν για κάθε επίπεδο του δικτύου. Αντί να χρησιμοποιεί στους υπολογισμούς όλο το Adjacency Matrix ή το Laplacian όπως προηγούμενες μέθοδοι χρησιμοποιεί μόνο κόμβους από την γειτονιά του κάθε κόμβου που επεξεργάζεται συνεπώς κατατάσσεται στους spatial αλγορίθμους. Οι k aggregators μπορούν να είναι ή mean ή pooling ή LSTM. Επίσης σε κάθε επίπεδο εκπαιδεύεται και ένα σύνολο βαρών μετασχηματισμού. Η δειγματοληψία (sampling) που χρησιμοποιείται αποτελεί υπερ παράμετρο και παραμένει σταθερή για την διάρκεια της εκπαίδευσης των αναπαραστάσεων κάθε κόμβου. Η στρατηγική που ακολουθείται εδώ είναι να μην χρησιμοποιούνται όλοι οι κόμβοι που συνδέονται με τον τρέχοντα κόμβο που επεξεργαζόμαστε σε αντίθεση με άλλους αλγορίθμους όπως ο GAT **[6]** Αυτό λειτουργεί και σαν regularization **[61]** αλλά και μειώνει την πολυπλοκότητα κρατώντας το υπολογιστικό κόστος σταθερό. Συνήθως με βάση τους δημιουργούς χρησιμοποιούνται δύο επίπεδα και ο αριθμός των δειγμάτων (samples) στο επόμενο επίπεδο είναι μικρότερος από το προηγούμενο. Επίσης με βάση τα βάρη μετασχηματισμού που αναφέραμε προηγουμένως μειώνεται η διάσταση των αναπαραστάσεων σταδιακά. Ο αλγόριθμος συσωρεύει πληροφορία από τους γείτονες σε κάθε επίπεδο κάτι που ονομάζεται γενικά message passing **[62]** όρος που χρησιμοποιείται και για άλλους αλγορίθμους και συνδυάζει αυτήν την πληροφορία με την προηγούμενη αναπαράσταση

του κάθε κόμβου. Επειδή χρησιμοποιεί την γειτονιά κάθε κόμβου στους υπολογισμούς και όχι τον πίνακα γειτνίασης ή κάποιο Laplacian κατηγοριοποιείται ως inductive δηλαδή μπορεί να δώσει αποτελέσματα για γράφους που δεν είχε διαθέσιμους στην εκπαίδευση. Να σημειωθεί ότι οι δημιουργοί **[19]** αναφέρουν και μη επιβλεπόμενο τρόπο εκπαίδευσης αναπαραστάσεων με την τεχνική του negative sampling **[20].**

2.5.2.2.2 Νευρωνικά Δίκτυα Autoencoders

Εδώ παρουσιάζουμε την λίστα Νευρωνικών Δικτύων που βασίζονται σε Autoencoders.

- VGAE

Πρόκειται για αλγόριθμο μη επιβλεπόμενης μάθησης **[11]** με την έννοια ότι δεν χρησιμοποιεί κάποιο μέτρο ομοιότητας. Αποτελείται από ένα μοντέλο πρόβλεψης (inference) και ένα generative μοντέλο. Το μοντέλο πρόβλεψης χρησιμοποιεί ένα GCN δύο επιπέδων και το generative είναι απλά ένα dot product. Δεδομένου ότι χρησιμοποιεί το GCN τον κατατάσει στην κατηγορία transductive για τους ίδιους λόγους που θεωρείται transductive και το GCN.

- G2G

Αυτό που κάνει αυτό το μοντέλο **[12]** είναι να αναπαριστά κάθε κόμβο σαν χαμηλής διάστασης Γκαουσιανή (low-dimensional Gaussian) κατανομή. Εδώ πρώτα ένας κωδικοποιητής (encoder) παράγει μία ενδιάμεση αναπαράσταση για κάθε κόμβο και στην συνέχεια αυτή χρησιμοποιείται σαν είσοδος για δύο άλλες συναρτήσεις που δίνουν την μέση τιμή και τον πίνακα συνδιακύμανσης (covariance matrix) για κάθε κόμβο. Τις αναπαραστάσεις τις έχουμε αφού περάσουμε το mean vector και covariance matrix από τη Γκαουσιανή κατανομή. Για να χρησιμοποιήσει την δομή του γράφου χρησιμοποιεί ένα μέτρο ομοιότητας και ένα αλγόριθμο τύπου pagerank **[60]** καθώς και περιορισμούς (constraints) που πρέπει να διατηρηθούν κατα την διάρκεια της εκπαίδευσης. Ο αλγόριθμος αυτός είναι επαγωγικός (inductive).

- DGI

Ο αλγόριθμος που περιγράφουμε εδώ **[13]** σκοπό έχει να συμπεριλάβει και χαρακτηριστικά συνολικά από όλο τον γράφο και να παράξει αναπαραστάσεις που έχουν σχέση και με την τοπική πληροφία κάθε κόμβου αλλά και συνολικά με όλο το γράφο . Χρησιμοποιεί readout function **[55]** για να έχει αναπαράσταση για όλο τον γράφο και

χρησιμοποιεί την έννοια της παραμόρφωσης (corruption) του γράφου για να παράξει negative samples **[20]** αλλάζοντας την δομή του γράφου αλλά και τα χαρακτηριστικά (features) των κόμβων/ακμών. Με αυτό το σύνολο δεδομένων και χρησιμοποιώντας έναν discriminator εκπαιδεύει όλα τα μέρη (components) και παράγει τις αναπαραστάσεις (embendings). Για την συνάρτηση readout λαμβάνονται υπόψη μονο οι αναπαραστάσεις που προκύπτουν από τον πραγματικό γράφο και όχι αυτές που προκύπτουν απο τον καινούριο παραποιημένο γράφο. Η συνάρτηση σφάλματος λαμβάνει υπόψη και τους δύο γράφους. Αυτός ο αλγόριθμος βασίζεται σε κάποιον encoder τον οποίο εκπαιδεύει με τρόπο μη επιβλεπόμενο και στην συνέχεια οι αναπαραστάσεις που προκύπτουν χρησιμοποιούνται για κάποιο από τα προβλήματα που αναφέραμε. Ο αλγόριθμος αυτός είναι επαγωγικός (inductive) γιατί οι encoders που μπορεί να εκπαιδεύσει βασίζουν την εκπαίδευση στην γειτονια κάθε κόμβου.

2.5.2.2.3 Μηχανισμός Προσοχής (Attention)

Εδώ περιγράφουμε μοντέλα Νευρωνικών δικτύων που χρησιμοποιούν μηχανισμό Προσοχής (Attention).

- GAT

Αυτός ο αλγόριθμος **[6]** όπως και ο προηγούμενος χρησιμοποιεί την άμεση γειτονιά του κάθε κόμβου. Το κάθε επίπεδο αποτελείται από έναν γραμμικό μετασχηματισμό για την αναπαράσταση από το προηγούμενο επίπεδο και στην συνέχεια χρησιμοποιείται ένας μηχανισμός προσοχής (attention) που υπολογίζει την αξία κάθε γείτονα. Ο μηχανισμός attention είναι ένα απλό feed forward δίκτυο με είσοδο δύο αναπαραστάσεις και έξοδο ένα score. Με αυτό τον τρόπο καταφέρνει να βρίσκει ποιοι κόμβοι γείτονες είναι πιο σημαντικοί για την αναπαράσταση του τρέχοντος κόμβου. Το δίκτυο δημιουργείται δημιουργώντας ακολουθιακά επίπεδα και με αυτό το τρόπο έχουμε όπως και σε κάποια από τα προηγούμενα πληροφορία απο k hops για κάθε κόμβο. Επίσης η όλη αρχιτεκτονική επεκτείνεται χρησιμοποιώντας περισσότερα από ένα attention heads. Στην ουσία μαθαίνονται περισσότερα είδη ομοιότητας αντί για ένα και χρησιμοποιούμε την πληροφορία σωρευτικά. Αυτή η στρατηγική έχει δώσει καλά αποτελέσματα για άλλα σενάρια χρήσης όπως στον τομέα της Επεξεργασίας Φυσικής Γλώσσας. Ο αλγόριθμος είναι επαγωγικός (inductive) για τους ίδιους λόγους με τον GraphSAGE και τους

υπόλοιπους δηλαδή επειδή δεν βασίζει τους υπολογισμούς στον πίνακα γειτνίασης ή κάποια μορφή του Laplacian αλλά στην γειτονία κάθε κόμβου. Σε αντίθεση με τον GraphSAGE **[19]** χρησιμοποιεί όλους τους γείτονες του κάθε κόμβου.

- AGNN

Ο αλγόριθμος αυτός **[14]** βασίζεται επίσης στην έννοια του attention. Είναι παρόμοιος με τον GAT αλλά διαφέρει για τον τρόπο που υπολογίζει το attention για κάθε κόμβο. Εδώ χρησιμοποιείται αντί για feedforward δίκτυο το cos των δύο αναπαραστάσεων και ο μηχανισμός attention έχει μία μόνο εκπαιδεύσιμη παράμετρο. Είναι πιο απλός μηχανισμός από αυτόν του GAT αλλά η απλότητα αυτή ίσως μπορεί να δώσει καλύτερα αποτελέσματα για σενάρια ημι επιβλεπόμενης μάθησης όπου τα δεδομένα με ετικέτα δεν είναι πολλά και είναι δύσκολο να βρεθούν. Όπως αναφέρεται **[14]** η ύπαρξη λιγότερων βαρών συνεπάγεται ανάγκη για λιγότερα δείγματα εκπαίδευσης (training samples). Επίσης χρησιμοποιεί άμεσα την γειτονία κάθε κόμβου οπότε κατηγοριοποιείται σαν inductive.

Ποιους από αυτούς τους αλγορίθμους θα επιλέξουμε για το σενάριο χρήσης και το σύνολο δεδομένων με το οποίο θα τους αξιολογήσουμε καθώς και αναλυτική περιγραφή τους θα την κάνουμε σε επόμενη ενότητα.

Ο Πίνακας 1 παρουσιάζει συνοπτικά τους αλγορίθμους με βάση την τεχνολογία που χρησιμοποιούν και σε τί είδους σύνολα δεδομένων (datasets) μπορούν να εφαρμοστούν.

Table 1: Αλγόριθμοι GRL

Αλγόριθμος	Τεχνολογία	Transductive	Inductive
Matrix Factorization	Βελτιστοποίηση της κάθε αναπαράστασης ξεχωριστά	ΝΑΙ	ΟΧΙ
RandomWalk(shall	Βελτιστοποίηση της	ΝΑΙ	ΟΧΙ

ow)	κάθε αναπαράστασης ξεχωριστά		
ChebNET	Convolution	NAI	OXI
GCN	Convolution	NAI	OXI
APPNP	Convolution	NAI	OXI
SGC	Convolution	NAI	OXI
GraphSAGE	Convolution	NAI	NAI
VGAE	Autoencoder	NAI	OXI
G2G	Autoencoder	NAI	NAI
DGI	Autoencoder	NAI	NAI
GAT	Attention	NAI	NAI
AGNN	Attention	NAI	NAI

2.6 Άλλες έννοιες και αλγόριθμοι που σχετίζονται με τον τομέα

Επειδή το σύνολο δεδομένων που θα ασχοληθούμε περιέχει και χρονική διάσταση θα κάνουμε μία αναφορά σε τεχνολογίες που μπορούν να την αξιοποιήσουν. Κάναμε μία σύντομη αναφορά σε CTDGs και DTDGs σε προηγούμενη ενότητα.

- TGN

Πρόκειται για έναν encoder **[15]** που χρησιμοποιείται σε έναν CTDG(και σε DTDG) και σκοπό έχει να παράγει αναπαραστάσεις (embendings) για κάθε timestamp που εξαρτάται από ένα (ή πολλά με το ίδιο timestamp) γεγονός. Μπορεί να χειριστεί και

προσθήκη και διαγραφή κόμβων/ακμών δηλαδή βρίσκει εφαρμογή σε σενάρια δυναμικών γράφων.

Το πρώτο συστατικό μέρος (module) είναι το Memory ή State. Εδώ για κάθε κόμβο διατηρείται ένα διάνυσμα το οποίο τροποποιείται μετά από κάθε γεγονός που αφορά τον κόμβο(θεωρητικά ένα event που αφορά έναν κόμβο μπορεί να αφορά κάθε άλλο κόμβο) και χρησιμεύει σαν η ιστορία του κόμβου. Για νέους κόμβους αρχικοποιείται στο μηδέν.

Το δεύτερο συστατικό μέρος (module) είναι το message function. Σκοπός του είναι να παράξει μία αναπαράσταση για κάθε γεγονός που αφορά τον κόμβο η οποία θα χρησιμοποιηθεί για να ενημερώσει την μνήμη του συγκεκριμένου κόμβου. Σε περίπτωση γεγονότος ακμής παράγονται δύο μηνύματα (messages) ένα για κάθε κόμβο.

Το τρίτο συστατικό μέρος είναι το message aggregator. Για λόγους επίδοσης αν υπάρχουν παραπάνω γεγονότα για έναν κόμβο σε ένα batch αυτά συσσωρεύονται σε ένα.

Το τέταρτο μέρος είναι το memory updater και με βάση τα messages και το memory ενημερώνει την μνήμη.

Το πέμπτο μέρος είναι αυτό που παράγει τα embendings για το χρόνο t.

Χρησιμοποιείται και ο όρος Staleness **[15]**.Όταν ένας κόμβος δεν έχει δραστηριότητα για μεγάλο χρονικό διάστημα, για παράδειγμα ένας χρήστης ενός κοινωνικού δικτύου, τότε θεωρείται ότι η πληροφορία που έχουμε για αυτόν τον κόμβο δεν είναι επίκαιρη. Για να αντιμετωπιστεί αυτό ανανεώνεται το memory του συγκεκριμένου κόμβου με βάση πληροφορία από γείτονες.

- Χρονική διάσταση για DTDGs

Όταν έχουμε χρονική πληροφορία για την εξέλιξη ενός δυναμικού γράφου διακριτού χρόνου μπορούμε να την αξιοποιήσουμε **[3]**. Η βασική προσέγγιση είναι η τελική αναπαράσταση να προκύπτει από ένα σύνολο αναπαραστάσεων (embendings) που προέκυψαν από όλα τα στιγμιότυπα του γράφου. Το τελικό embedding προκύπτει εισάγοντας όλα τα προηγούμενα από κάποιο RNN. Τα RNNs είναι ιδανικά στο να αποτυπώσουν την πληροφορία της χρονικής εξέλιξης ενός φαινομένου. Εδώ δεν έχει σημασία η αναπαράσταση που προκύπτει για κάθε timestamp αλλά η αλληλουχία των γεγονότων μπορεί να δώσει πληροφορία η οποία πιθανόν να μην αξιοποιούνταν με άλλες αρχιτεκτονικές.

- Incremental Learning

Αυτή η έννοια **[16]** αφορά την βελτίωση του μοντέλου με όλα τα νέα δεδομένα που συνοδεύονται από ετικέτες (labels) που προκύπτουν. Γενικά είναι ανοιχτό θέμα. Στην περίπτωση των transductive μοντέλων η πιο απλή περίπτωση είναι η επανεκπαίδευση του μοντέλου. Υπάρχουν κάποιες προσεγγίσεις για δεδομένα γράφων **[16]**.

- Streaming Scenario

Εδώ **[3]** μας ενδιαφέρει ιδανικά να παράγουμε άμεσα αναπαραστάσεις(embendings) για νέα δεδομένα που προστίθενται στον γράφο αλλά και να πραγματοποιούμε incremental learning με αυτά. Αποτελεί πολύ συνηθισμένη απαίτηση για πολλά σενάρια χρήσης περιλαμβάνοντας και αυτό της δικής μας εργασίας. Χαρακτηριστικό που έχουν οι inductive αλγόριθμοι είναι ότι μπορούν να πάρουν σαν είσοδο έναν τροποποιημένο γράφο και να δώσουν αναπαραστάσεις για νέους κόμβους.

- Over smoothing

Όπως αναφέρεται **[17]** τα GNNs όταν έχουμε πολλά επίπεδα δεν έχουν καλή επίδοση. Αυτό οφείλεται στο φαινόμενο όπου οι αναπαραστάσεις μοιάζουν πολύ μεταξύ τους επειδή κωδικοποιούν στην ουσία την ίδια πληροφορία. Κάποιες λύσεις για αυτό το πρόβλημα αφορούν skip connections ή concatenation απο αναπαραστάσεις του προηγούμενου επιπέδου ώστε στην τελική αναπαράσταση να μην χάνεται τελείως η αρχική πληροφορία για τον κόμβο.

2.7 Ανοικτά Ερευνητικά Θέματα στη Μηχανική Μάθηση σε Γράφους

Η έρευνα γύρω από τον τομέα του GRL **[17,16]** είναι σχετικά πρόσφατη και υπάρχουν πολλά θέματα τα οποία δεν έχουν αντιμετωπιστεί ακόμα και γίνεται έρευνα γύρω από αυτά. Εδώ παρουσιάζουμε σύντομα κάποια από αυτά.

Incremental Learning

Αυτό το ερευνητικό πεδίο **[16]** στόχο έχει να βρει τρόπους ώστε ένας αλγόριθμος να μπορεί να επικαιροποιείται με βάση νέα δείγματα (samples) όταν αυτά είναι διαθέσιμα. Αυτός ο τομέας δεν αφορά μόνο δεδομένα γράφων αλλά στους γράφους υπάρχουν πολλές ιδιαιτερότητες που πρέπει να αντιμετωπιστούν σε σχέση με σενάρια κλασικής Μηχανικής

Μάθησης σε Ευκλείδεια δεδομένα όπως η κατηγοριοποίηση εικόνας (image classification) με CNNs.

Graph Structure Learning

Ακολουθόντας την επιτυχία που προέκυψε από τα Νευρωνικά δίκτυα στο να παράγουν αυτόματα χαρακτηριστικά (features) και το ότι αυτές οι μέθοδοι ξεπέρασαν τις παραδοσιακές μεθόδους feature engineering ο τομέας αυτός **[17,63]** προσπαθεί να βρει λύσεις στο πρόβλημα της αυτόματης παραγωγής γράφων από τα διαθέσιμα δείγματα με τρόπο που να βελτιστοποιεί την επίδοση.

Εφαρμογές σε άλλα προβλήματα

Δεδομένης της επιτυχίας των GNNs γίνονται προσπάθειες να γενικευτεί η χρήση τους και σε άλλα παραδοσιακά πεδία **[17]** της Μηχανικής μάθησης όπως υπολογιστική όραση και επεξεργασία φυσικής γλώσσας. Οι προσπάθειες αυτές δικαιολογούνται από την υπόθεση του ότι αν έχουμε διαθέσιμες σχέσεις μεταξύ δειγμάτων (datapoints) αντί να τα θεωρούμε ως ανεξάρτητα έχουμε περισσότερη πληροφορία να αξιοποιήσουμε.

Δυναμικοί Γράφοι (Dynamic Graphs)

Εδώ **[17,3,15]** αναζητούνται λύσεις για το πρόβλημα της αντιμετώπισης δυναμικών γράφων όπου θέλουμε να χρησιμοποιήσουμε την πληροφορία από την χρονική εξέλιξη ενός γράφου. Τα γεγονότα που συμβαίνουν στον γράφο μπορούν να είναι προσθήκη κόμβου, διαγραφή κόμβου, δημιουργία ακμής κτλ. Αν αντιμετωπίζαμε τον τελικό γράφο σαν κάτι στατικό θα χάναμε πληροφορία.

Ανταγωνιστική Ευρωστία (Adversarial Robustness)

Εδώ **[17,64]** σκοπός είναι το μοντέλο και κατ' επέκταση η διαδικασία της εκπαίδευσης να αντιμετωπίσει την περίπτωση ενός συνόλου δεδομένων (dataset) το οποίο εσκεμμένα έχει αλλοιωθεί με σκοπό να επηρεάσει την απόδοση του μοντέλου.

ΚΕΦΑΛΑΙΟ 3ο

Παρουσίαση προτεινόμενης μεθοδολογίας

Εδώ θα παρουσιάσουμε την μεθοδολογία που θα ακολουθήσουμε για να αξιολογήσουμε τις τεχνολογίες που παρουσιάσαμε στο σενάριο χρήσης που περιγράψαμε και με το σύνολο δεδομένων που έχουμε στην διάθεση μας. Αρχικά θα ορίσουμε με σαφή τρόπο το πρόβλημα που καλούμαστε να λύσουμε και τους λόγους για τους οποίους επιλέξαμε τους αλγορίθμους. Στην συνέχεια θα παρουσιάσουμε μία τεχνική περιγραφή των αλγορίθμων που επιλέξαμε. Τέλος θα παρουσιάσουμε την μεθοδολογία ανίχνευσης απάτης σε συναλλαγές

3.1 Ορισμός του προβλήματος

Η εργασία αυτή ασχολείται με την ανίχνευση απάτης σε συναλλαγές. Πρόκειται για ένα πολύ σημαντικό πρόβλημα του πραγματικού κόσμου και περισσότερα για τις περιπτώσεις και την μεθοδολογία που ακολουθείται μπορούν να βρεθούν εδώ **[18].** Το περιβάλλον που έχουμε είναι ένας γράφος που αναπαριστά αλληλεπιδράσεις μεταξύ οντοτήτων που αφορούν χρηματικές συναλλαγές. Πιο συγκεκριμένα έχουμε τους κόμβους οι οποίοι είναι τα transactionIDs. Τα transactionIDs στο bitcoin στην ουσία είναι λογαριασμοί. Οι ακμές είναι χρηματικές συναλλαγές ανάμεσα στις οντότητες. Οι χρηματικες αυτές συναλλαγές αφορούν το αν υπήρξε μεταφορά χρημάτων από έναν κόμβο σε έναν άλλο. Δεν λαμβάνει υπόψη το ποσό συνεπώς οι ακμές δεν έχουν βάρη. Επιπλέον ο γράφος είναι κατευθυνόμενος. Επίσης έχουμε έναν μικρό αριθμό labeled οντοτήτων ως νόμιμων και παράνομων. Το πρόβλημα που θα λύσουμε θα είναι αυτό της κατηγοριοποίησης κόμβων (node classification) **[2]**. Σκοπός είναι με χρήση της μεθόδου semi-supervised learning **[1]** να βρούμε ποιες άλλες οντότητες σχετίζονται με παράνομη δραστηριότητα. Για να το πετύχουμε αυτό και να βρούμε την καλύτερη μέθοδο θα δοκιμάσουμε αλγορίθμους από την βιβλιογραφία των deep αλγορίθμων επιλέγοντας τους πλέον κατάλληλους

[6,13,14,19] για την περίπτωση αυτή. Επίσης εκτός από deep μοντέλα και για λόγους σύγκρισης θα δοκιμάσουμε και ένα shallow μοντέλο, το node2vec **[59]** χρησιμοποιώντας self-supervised learning για να πάρουμε αναπαραστάσεις(embendings) δηλαδή πληροφορία για κάθε κόμβο στον γράφο και στην συνέχεια να τα χρησιμοποιήσουμε σε συνδυασμό με τα υπόλοιπα χαρακτηριστικά (features) του κάθε κόμβου για να κάνουμε την πρόβλεψη. Όπως γίνεται στα σενάρια ημι επιβλεπόμενης μάθησης **[1]** και αναφέραμε και σε προηγούμενη ενότητα θα χρησιμοποιήσουμε μικρό αριθμό labels για την εκπαίδευση και τα υπόλοιπα για validation/test. Επίσης μας ενδιαφέρει το μοντέλο που θα προκύψει να μπορεί χειριστεί και το streaming σενάριο **[3]** κάτι που θα επηρεάσει την απόφαση για τους αλγορίθμους που θα διαλέξουμε.

3.2 Επιλογή και παρουσίαση αλγορίθμων

Σε αυτή την ενότητα θα παρουσιάσουμε την επιλογή των αλγορίθμων, τους λόγους που τους επιλέξαμε και θα τους περιγράψουμε σε τεχνικό επίπεδο.

3.2.1 Επιλογή αλγορίθμων

Για την διενέργεια των πειραμάτων επιλέγουμε τους αλγορίθμους GAT **[6]**, AGNN **[13]**, GraphSAGE **[19]**, DGI **[14]** και node2vec **[59]**.

Τους τρεις πρώτους τους επιλέγουμε διότι μπορούν να χρησιμοποιηθούν αποδότικά σε μεγάλους γράφους **[19,2]** χαρακτηριστικό που το θέλουμε διότι οι γράφοι που προκύπτουν σε δίκτυα συναλλαγών τις περισσότερες φορές έχουν μεγάλο μέγεθος. Επίσης όπως αναφέραμε σε προηγούμενη ενότητα έχουν την επαγωγική (inductive) **[1,2,5,19]** ιδιότητα. Αυτό το χαρακτηριστικό το θέλουμε διότι στο σενάριο που εξετάζουμε οι γράφοι μπορούν να αλλάξουν γρήγορα ή να παραχθούν τελείως καινούριοι γράφοι σε πολύ σύντομο χρονικό διάστημα. Το σενάριο της επανεκπαίδευσης που απαιτούν οι μη επαγωγικοί (transductive) αλγόριθμοι δεν είναι βιώσιμο στις περισσότερες περιπτώσεις. Τον αλγόριθμο DGI **[14]** τον επιλέγουμε για να αξιολογήσουμε την αποτελεσματικότητα προ εκπαιδευμένων βαρών στο πρόβλημα της

κατηγοριοποίησης κόμβων. Τέλος τον node2vec τον επιλέξαμε για λόγους σύγκρισης των προηγούμενων.

3.2.2 Παρουσίαση Αλγορίθμων

Εδώ παρουσιάζουμε σε τεχνικό επίπεδο τους αλγορίθμους της προηγούμενης ενότητας.

3.2.2.1 Αλγόριθμος GAT

Ο αλγόριθμος GAT **[6]** όπως και οι υπόλοιποι που θα εξετάσουμε λειτουργεί συνδυάζοντας πληροφορία από τους γείτονες του κάθε κόμβου για τον οποίο θέλουμε να βρούμε μία αναπαράσταση συνεπώς κατηγοριοποιείται ως spatial και inductive. Όπως αναφέρεται από τους δημιουργούς και σε αντίθεση με την περίπτωση του GraphSage που θα εξετάσουμε στην συνέχεια λειτουργεί λαμβάνοντας υπόψη όλους τους γείτονες του κάθε κόμβου. Η βασική ιδέα είναι ότι ο κάθε γείτονας μπορεί να παίζει διαφορετικό ρόλο σε κάθε περίπτωση. Για παράδειγμα ένας γείτονας ο οποίος έχει μεγαλύτερο centrality ή bettweness από κάποιον άλλο μπορεί να είναι πιο σημαντικός. Εισάγοντας έναν μηχανισμό attention το δίκτυο είναι σε θέση να βρει και να εκμεταλλευτεί τέτοιες συσχετίσεις. Ο μηχανισμός attention στο μοντέλο GAT υλοποιείται σαν ένα feedforward νευρωνικό δίκτυο που σαν είσοδο έχει δύο αναπαραστάσεις και σαν έξοδο ένα score για αυτές τις δύο αναπαραστάσεις. Οι παράμετροι που μαθαίνονται είναι ένα διάνυσμα βαρών. Ο μηχανισμός αυτός μπορεί να υλοποιηθεί και με άλλους τρόπους. Το score αυτό για να είναι συγκρίσιμο ανάμεσα σε κόμβους κανονικοποιείται με χρήση softmax activation. Επίσης οι δημιουργοί ισχυρίζονται ότι είναι πιο αποτελεσματικό να χρησιμοποιούνται περισσότερα από ένα attention head τα οποία κατά την εκπαίδευση αντιπροσωπεύουν διαφορετικά similarities που μαθαίνονται στην εκπαίδευση αν και αυτό αυξάνει την πολυπλοκότητα του μοντέλου. Το βασικό δομικό στοιχείο είναι ένα επίπεδο το οποίο υλοποιεί έναν γραμμικό μετασχηματισμό στα χαρακτηριστικά και έναν ή παραπάνω μηχανισμό attention. Στο αρχικό επίπεδο είσοδος είναι τα χαρακτηριστικά (features) και έξοδος μία αναπαράσταση η οποία με βάση τα όσα γράφονται από τους δημιουργούς αποτελεί το concatenation των αναπαραστάσεων που προκύπτουν από κάθε attention head. Στα ενδιάμεσα επίπεδα είσοδος είναι οι αναπαραστάσεις που προέκυψαν από το προηγούμενο επίπεδο. Στο τελευταίο επίπεδο αντί για concatenation

οι δημιουργοί προτείνουν να χρησιμοποιηθεί το average των αναπαραστάσεων. Δημιουργώντας επίπεδα το ένα μετά το άλλο καταφέρνουν να κωδικοποιήσουν πληροφορία από k-hops γειτονιά του κάθε κόμβου. Οι υπερ παραμετροι που μπορούν να δοκιμαστούν εδώ όπως θα δούμε και στην συνέχεια είναι πολλές και η απόδοση τους εξαρτάται από το dataset. Ακολουθεί η περιγραφή ενός τέτοιου επιπέδου.
Ο μηχανισμός attention υλοποιείται ως εξής για κάθε ένα attention head για κάθε επίπεδο.

$$\alpha_{ij} = \frac{\exp(LeakyReLU(\vec{a}^T[W\vec{h}_i]||W\vec{h}_j))}{\sum_{k\in N_i}\exp(LeakyReLU(\vec{a}^T[W\vec{h}_i]||W\vec{h}_k))}$$

Figure 1: Μηχανισμός attention GAT

Στο σχήμα 1 έχουμε τον συντελεστή ομοιότητας για τον τρέχοντα κόμβο και για τον κάθε γείτονα του. Οι εκπαιδεύσιμες παραμετροι είναι ένας πίνακας μετασχηματισμού και ένα διάνυσμα.

Και ο μηχανισμός που συνδυάζει τις αναπαραστάσεις του κάθε κόμβου με αυτές των γειτόνων του για να παράξει την νέα αναπαράσταση για την περίπτωση των ενδιάμεσων επιπέδων είναι ο παρακάτω.

$$\vec{h'}_i = ||_{k=1}^{K}\sigma(\sum_{j\in N_i}\alpha_{ij}^k W^k\vec{h}j)$$

Figure 2: Ενδιάμεσα επίπεδα GAT

Στο σχήμα 2 έχουμε την αναπαράσταση που προκύπτει από το concatenation των αναπαραστάσεων των K attention heads για το επίπεδο k.
Και για το τελικό επίπεδο ο παρακάτω.

$$\vec{h'}_i = \sigma(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i}a_{ij}^k W^k\vec{h}_j)$$

Figure 3: Τελικό επίπεδο GAT

Στο σχήμα 3 έχουμε την τελική αναπαράσταση που προκύπτει σαν ο μέσος όρος από τις αναπαραστάσεις που προκύπτουν από κάθε attention head.

3.2.2.2 Αλγόριθμος GraphSage

Ο αλγόριθμος GraphSage **[19]** μοιάζει με τον προηγούμενο διότι και αυτός παράγει αναπαραστάσεις με βάση την γειτονία του κάθε κόμβου και μπορεί να παράξει αναπαραστάσεις για κόμβους γράφων τους οποίους δεν είχε διαθέσιμους στην εκπαίδευση. Αντί να βελτιστοποιεί την αναπαράσταση κάθε κόμβου ξεχωριστά όπως κάνουν οι shallow αλγόριθμοι, μαθαίνει ένα σετ από Aggregators και στην συνέχεια γενικεύει την γνώση. Σε αντίθεση με τον αλγόριθμο GAT ,ο GraphSAGE δεν λαμβάνει υπόψιν όλους τους γείτονες του κάθε κόμβου υποχρεωτικά αλλά έναν προκαθορισμένο αριθμό ο οποίος αποτελεί υπερ παράμετρο. Η στρατηγική αυτή εξυπηρετεί δύο στόχους. Πρώτον λειτουργεί σαν regularization και δεύτερον κρατάει το υπολογιστικό κόστος σταθερό σε σχέση με το πλήθος των κόμβων. Ο αλγόριθμος περιγράφεται παρακάτω.
Αρχικά είσοδος είναι ένας γράφος καθώς και τα χαρακτηριστικά που συνοδεύουν τον κάθε κόμβο.
Στην συνέχεια η διαδικασία εξελίσσεται αρχικοποιώντας τις αναπαραστάσεις για κάθε κόμβο με τα χαρακτηριστικά του όπως φαίνεται στο σχήμα 4.

$h_v^0 \leftarrow x_v, \forall v \in \mathcal{V}$

Figure 4: Αρχικές αναπαραστάσεις GraphSAGE

Στην συνέχεια για κάθε επίπεδο του δικτύου και για κάθε κόμβο εκτελούνται τα παρακάτω βήματα.

$$h^k_{\mathcal{N}(v)} \leftarrow AGGREGATE_k(\{h^{k-1}_u, \forall u \in \mathcal{N}(v)\})$$

Figure 5: Συνδυασμός αναπαραστάσεων γειτόνων GraphSAGE

Στο σχήμα 5 δημιουργείται ένα διάνυσμα απο τις αναπαραστάσεις του προηγούμενου επιπέδου για τους γείτονες(οχι υποχρεωτικά όλους) του κόμβου που επεξεργαζόμαστε.

$$h^k_v \leftarrow \sigma(W^k \cdot CONCAT(h^{k-1}_v, h^k_{\mathcal{N}(v)}))$$

Figure 6: Συνδυασμός αναπαραστάσεων γειτόνων με προηγούμενες αναπαραστάσεις

Στο σχήμα 6 περιγράφεται το concatenation της αναπαράστασης που προέκυψε στο προηγούμενο βήμα με την αναπαράσταση του τρέχοντος κόμβου από το προηγούμενο επίπεδο σε συνδυασμό με έναν μετασχηματισμό και μία συνάρτηση ενεργοποίησης.

$$h^k_v \leftarrow h^k_v / ||h^k_v||_2, \forall v \in \mathcal{V}$$

Figure 7: Κανονικοποίηση Αναπαράστασης

Σε αυτό το βήμα κανονικοποιείται το διάνυσμα αναπαράσταση. Τέλος, η αναπαράσταση που προκύπτει από το τελευταίο επίπεδο είναι και η τελική αναπαράσταση του κόμβου.

Το κύριο μέρος του αλγορίθμου είναι οι aggregators. Εδώ με βάση το paper **[19]** υπάρχουν τρεις επιλογές mean,max και LSTM.

Ο **mean aggregator** παίρνει το element wise mean των δύο διανυσμάτων δηλαδή της αναπαράστασης των γειτόνων αλλά και του κόμβου από το προηγούμενο επίπεδο.

$$h_v^k \leftarrow \sigma(W \cdot MEAN(\{h_v^{k-1}\} \cup \{h_u^{k-1}, \forall u \in N_{(v)})$$
Figure 8: Mean aggregator GraphSAGE

Ο **LSTM aggregator** προσπαθεί να φέρει τα οφέλη των RNNs στην παραγωγή των αναπαραστάσεων. Παρόλο του ότι θεωρητικά έχει καλύτερη απόδοση υπάρχει η αδυναμία του ότι λαμβάνει υπόψη την σειρά με την οποία γίνεται η επεξεργασία. Για να αντιμετωπιστεί αυτό το πρόβλημα οι δημιουργοί εφάρμοσαν τυχαίες αναδιατάξεις των εισόδων για την γειτονία κάθε κόμβου

Ο **pooling aggregator.** Ο aggregator αυτός μοιάζει με τον πρώτο αλλά διαφέρει η σειρά που γίνονται οι υπολογισμοί όπως περιγράφεται από τα παρακάτω.

$$AGGREGATE_k^{pool} = \max(\{\sigma(W_{pool} h_{u_i} + b), \forall u_i \in N_{(v)}\})$$
Figure 9: Pool aggregator GraphSAGE

3.2.2.3 Αλγόριθμος AGNN

Ο αλγόριθμος AGNN **[14]** αρχικά πραγματοποιεί ένα γραμμικό μετασχηματισμό στα χαρακτηριστικά(features) όπως φαίνεται στο σχήμα 10.

$$H_0 = \sigma(XW^0)$$
Figure 10: Αρχικός μετασχηματισμός AGNN

και στην συνέχεια για όλους τους γείτονες του κάθε κόμβου υπολογίζει similarities όπως και ο GAT αλλά ορίζει το similarity με διαφορετικό τρόπο όπως φαίνεται παρακάτω στο σχήμα 11.

$$\alpha_{ij}^{k} = \frac{\exp(\beta^{k} \cos(H_i^k, H_j^k))}{\sum_{v_r \in N_{(i)} \cup \{v_i\}} \exp(\beta^{k} \cos(H_i^k, H_r^k))}$$

Figure 11: Μηχανισμός attention AGNN

Στην συνέχεια υπολογίζονται οι αναπαραστάσεις κάθε κόμβου όπως παρουσιάζεται στο σχήμα 12.

$$H_i^{k+1} = \sum_{v_j \in N_{(i)} \cup \{v_i\}} \alpha_{ij}^{k} H_j^{k}$$

Figure 12: Αναπαραστάσεις κρυμένου επιπέδου AGNN

Τέλος, μετά το τελευταίο επίπεδο χρησιμοποιείται ένα Linear layer για να παραχθεί η τελική αναπαράσταση όπως φαίνεται στο σχήμα 13.

$$Z = f(X, A) = \sigma(H^{K} W^{l})$$

Figure 13: Τελική αναπαράσταση AGNN

Να σημειωθεί ότι το πως υλοποιεί το framework τα layers αποτελεί black box. Για παράδειγμα δεν είναι σαφές το αν το framework γνωρίζει ότι πρόκειται για τελικό propagation layer και εφαρμόζει το γραμμικό μετασχηματισμό ή πρέπει να τον προσθέσουμε στο μοντέλο που φτιάχνουμε. Με βάση αυτό πραγματοποιούμε την υλοποίηση με βάση τα επίσημα παραδείγματα της Pytorch Geometric**[23]**.

3.2.2.4 Αλγόριθμος DGI

Ο αλγόριθμος DGI **[13]** δεν αποτελεί encoder όπως οι τρεις προηγούμενοι. Αντίθετα χρησιμοποιεί έναν encoder και τον εκπαιδεύει με unsupervised τρόπο **[20]** και παράγει αναπαραστάσεις. Αποτελείται από τρία βασικά μέρη. Αυτά είναι ο discriminator μία corruption function και μία readout function **[55]**. Ο τρόπος που λειτουργεί είναι ο

παρακάτω. Αρχικά δημιουργεί ένα dataset εκπαίδευσης αποτελούμενο από positive και negative samples. Τα positive προέρχονται από τον γράφο που εξετάζουμε και τα negative από έναν corrupted γράφο. Ο corrupted γράφος προκύπτει κρατώντας τις ακμές αλλά αλλάζοντας με τυχαία αναδιάταξη τα features. Στην συνέχεια το readout function με είσοδο τις αναπαραστάσεις του encoder παράγει μία αναπαράσταση όλου του γράφου από τα positive samples μόνο. Με βάση αυτά δημιουργείται ένα loss function αποτελούμενο από δύο μέρη ένα για τα positive samples όπως φαίνεται στο σχήμα 14 και ένα για τα negative όπως φαίνεται στο σχήμα 15.

$$\mathcal{L}_{pos} = \sum_{i=1}^{n} \mathbb{E}_{(X,A)}[\log \mathbb{D}(H_i, s)]$$

Figure 14: Συνάρτηση σφάλματος για τα positive samples DGI

Σχήμα 14. Συνάρτηση σφάλματος για τα positive samples DGI

$$\mathcal{L}_{neg} = \sum_{j=1}^{h} \mathbb{E}_{(\widehat{X},\widehat{A})}[\log(1 - \mathcal{D}(\widehat{H_j}, s))]$$

Figure 15: Συνάρτηση σφάλματος για negative samples DGI

Σχήμα 15. Συνάρτηση σφάλματος για τα negative samples DGI

Το τελικό loss function είναι το μέσο όρο των δύο παραπάνω
Έξοδος του αλγορίθμου είναι ο encoder που πλέον έχει εκπαιδευμένα βάρη και μπορεί να χρησιμοποιηθεί σε άλλα downstream tasks.

3.2.2.5 Αλγόριθμος node2vec

Ο αλγόριθμος node2vec **[59]** κατατάσσεται στους shallow και σε αντίθεση με τους προηγούμενους βελτιστοποιεί τις αναπαραστάσεις ξεχωριστά για κάθε κόμβο. Το επιτυγχάνει χρησιμοποιώντας random walks. Με βάση αυτό δημιουργεί ένα dataset και υπολογίζει πιθανότητες να βρεθεί ένας κόμβος στην γειτονία ενός άλλου κόμβου

πραγματοποιόντας ένα τυχαίο περίπατο(random walk). Στην συνέχεια με χρήση ενός decoder εσωτερικού γινομένου βελτιστοποιεί τις αναπαραστάσεις. Υπάρχουν πολλές υπερ παράμετροι για αυτόν τον αλγόριθμο. Το τελικό αποτέλεσμα είναι οι εκπαιδευμένες αναπαραστάσεις οι οποίες στην συνέχεια μπορούν να χρησιμοποιηθούν για προβλήματα όπως κατηγοριοποίηση κόμβων.

3.3 Μεθοδολογία ανίχνευσης απάτης σε συναλλαγές

Με βάση τα παραπάνω εξηγήσαμε πως λειτουργούν οι αλγόριθμοι που επιλέξαμε. Όπως αναφέραμε το σενάριο χρήσης που αντιμετωπίζουμε είναι αυτό της ανίχνευσης απάτης σε συναλλαγές **[18]** και το πρόβλημα αποτελεί κλασική περίπτωση ημι επιβλεπόμενης μάθησης **[1]**. Όπως αναφέραμε το σενάριο αυτό περιγράφεται ως imbalanced πρόβλημα **[26]** και για αυτό επιλέγουμε τρεις τρόπους για να το αντιμετωπίσουμε. Αυτοί οι τρόποι είναι από τους πιο συνηθισμένους που εφαρμόζονται στην βιβλιογραφία **[26]** και αφορούν πρώτον την χρήση βαρών στην εκπαίδευση όπου υπάρχει το χαρακτηριστικό imbalance στο σύνολο δεδομένων, δεύτερον το undersampling της majority class με σκοπό να γίνει το σύνολο δεδομένων balanced και τρίτον το oversampling της minority κλάσσης ώστε και πάλι να γίνει το σύνολο δεδομένων balanced. Αυτά για να αντιμετωπίσουμε την πρώτη διάσταση του προβλήματος όπως την αναφέραμε σε προηγούμενη ενότητα.
Με βάση αυτά τα τρία σενάρια θα δοκιμάσουμε διάφορες υπερ παραμέτρους με στόχο να βρούμε την βέλτιστη που ταιριάζει στο σύνολο δεδομένων. Αυτό το κάνουμε διότι οι προκαθορισμένες αρχιτεκτονικές που είναι υλοποιημένες στο framework είναι βελτιστοποιημένες για συγκεκριμένα datasets συνεπώς στην πράξη μπορεί να μην δώσουν καλά αποτελέσματα για το σύνολο δεδομένων που εξετάζουμε εδώ.
Στην συνέχεια θα εξετάσουμε όλα αυτά τα υποψήφια μοντέλα που προκύπτουν σε συνδυασμό με τις τρεις τεχνικές που περιγράφηκαν προηγουμένως. Στόχος είναι να αξιολογήσουμε τα αποτελέσματα που προκύπτουν.

Στην συνέχεια θα αξιολογήσουμε το σενάριο όπου αντί για νευρωνικά δίκτυα χρησιμοποιούνται αλγόριθμοι που δεν συνδυάζουν πληροφορία από την δομή του

γράφου και τα χαρακτηριστικά των κόμβων μαζί αλλά ξεχωριστά. Αυτό θα το πετύχουμε χρησιμοποιώντας τον αλγόριθμο node2vec για να εξάγουμε πληροφορία από την δομή του γράφου και την χρήση αυτών των χαρακτηριστικών είτε μόνα τους είτε σε συνδυασμό με τα υπόλοιπα χαρακτηριστικά και κάποιον classifier. Σκοπός είναι να δούμε αν η επίδοση είναι συγκρίσιμη σε σχέση με την περίπτωση της end to end εκπαίδευσης των νευρωνικών καθώς και πόσο ικανό είναι τα χαρακτηριστικά που προκύπτουν από τον αλγόριθμο node2vec να κατηγοριοποιήσουν επιτυχώς τους κόμβους χωρίς επιπλέον πληροφορία.
Επίσης θα αξιολογήσουμε το σενάριο παραγωγής αναπαραστάσεων απο νευρωνικά, δηλαδή τα μοντέλα του πειράματος 1 αλλά χωρίς τα χαρακτηριστικά των κόμβων σε συνδυασμό με κάποιον classifier. Επίσης θα αξιολογήσουμε την περίπτωση των αναπαραστάσεων που προέκυψαν από το πείραμα 1 αλλά με Random Forest classifier.

Τέλος θα εξετάσουμε και πάλι την ικανότητα ενός classifier να κατηγοριοποιήσει τους κόμβους του δικτύου με αναπαραστάσεις που έχουν προκύψει από τον αλγόριθμο DGI. Εδώ σκοπός είναι να δούμε πόση πληροφορία μας δίνει μία μέθοδος εκπαίδευσης που δεν χρησιμοποιεί καθόλου τις ετικέτες των δεδομένων. Με βάση όλα τα παραπάνω θα αποφανθούμε για το πιο είναι το καλύτερο σενάριο που προκύπτει για την περίπτωση χρήσης που αξιολογούμε.

ΚΕΦΑΛΑΙΟ 4ο

Πειραματικό Μέρος και Υλοποίηση

Σε αυτή την ενότητα θα περιγράψουμε αρχικά το σύνολο δεδομένων που έχουμε στην διάθεση μας και με το οποίο θα αξιολογήσουμε τους αλγορίθμους. Στην συνέχεια θα περιγράψουμε αναλυτικά τα πειράματα που θα πραγματοποιήσουμε και τον λόγο που τα επιλέξαμε ώστε να αξιολογήσουμε την επίπτωση παραμέτρων στα αποτελέσματα. Στην συνέχεια θα περιγράψουμε τις μετρικές αξιολόγησης που καταγράψαμε στα πειράματα ώστε να μπορεί εύκολα να επιλεγεί η κατάλληλη αρχιτεκτονική ανάλογα με το σενάριο χρήσης και το τι είναι σημαντικό να επιτευχθεί σε κάθε περίπτωση. Τέλος θα παρουσιάσουμε αναλυτικά τα ποσοτικά αποτελέσματα για κάθε πείραμα.

4.1 Περιγραφή συνόλου δεδομένων

Το σύνολο δεδομένων που έχουμε στην διάθεση μας είναι το Elliptic Dataset **[65]**. Αποτελεί καταγραφή συναλλαγών για το κρυπτονόμισμα bitcoin.

Αποτελείται από 203,769 κόμβους που ονομάζονται transactions IDs. Αντιστοιχούν στην ουσία σε τραπεζικούς λογαριασμούς. Οι ακμές είναι 234,355 και αντιστοιχούν σε μεταφορά χρημάτων από ένα transaction ID σε ένα άλλο. Με βάση αυτό ο γράφος είναι κατευθηνόμενος (directed). Στην συνέχεια βλέπουμε ότι το σύνολο δεδομένων περιλαμβάνει και την χρονική διάσταση δίνοντας ένα χαρακτηριστικό (feature) που σχετίζεται με τον χρόνο. Αποτελείται από τιμές 1-49 όπου η κάθε τιμή περιλαμβάνει transactionIDs που εμφανίστηκαν εντός δύο εβδομάδων. Όλα τα transactionIDs για κάθε χρονικό εύρος αποτελούν ένα connected component. Συνεπώς ο γράφος αποτελείται από 49 γράφους των οποίων οι κόμβοι δεν συνδέονται με ακμές. Ο κάθε κόμβος συνοδεύεται από 166 features. Τα πρώτα 94 αφορούν πληροφορία για τον συγκεκριμένο κόμβο και τα υπόλοιπα 72 αφορούν σωρευτική πληροφορία από 1-hop γείτονες του κάθε κόμβου. Όλα

τα χαρακτηριστικά (features) είναι αριθμητικά. Τέλος ο κάθε κόμβος συνοδεύεται από ένα label. Τα τρία labels είναι licit/illicit/unknown. Η πρώτη αποτελεί το 22% η δεύτερη 2% και η τρίτη τα υπόλοιπα. Να πούμε ότι δεν έχουμε πληροφορία για το τι αντιπροσωπεύουν τα 165 χαρακτηριστικά πράγμα που αποτελεί περιορισμό για την αξιολόγηση των μεθόδων.

4.2 Περιγραφή Πειραμάτων και στόχοι αξιολόγησης

Σε αυτή την ενότητα θα περιγράψουμε τα πειράματα που θα πραγματοποιήσουμε καθώς και τους λόγους που επιλέξαμε να τα πραγματοποιήσουμε και τα συμπεράσματα που θέλουμε να προκύψουν από αυτά. Θυμίζουμε ότι έχουμε το σενάριο όπου έχουμε δύο κλάσεις με αναλογία δειγμάτων περίπου 1:10. Τα σενάρια που θα αξιολογήσουμε είναι πέντε.

4.2.1 Πείραμα Πρώτο Class Imbalance και υπερ παράμετροι

Το πρώτο αφορά ημι επιβλεπόμενη εκπαίδευση (semi supervised) πάνω στο σύνολο δεδομένων που έχουμε ώστε να κατηγοριοποιήσουμε τους κόμβους που δεν έχουν ετικέτα. Σε αυτό το σενάριο θα πραγματοποιήσουμε πειράματα για να βρούμε τον κατάλληλο συνδυασμό παραμέτρων που δίνουν τα καλύτερα αποτελέσματα. Όπως είπαμε και στην ενότητα Μεθοδολογία θα αξιολογήσουμε τους αλγορίθμους GAT/AGNN/GRAPHSAGE με διάφορες υπερ παραμέτρους ταυτόχρονα με τα τρία σενάρια που δοκιμάζουμε για την αντιμετώπιση του διαφορετικού πλήθους δειγμάτων ανά κλάση. Με βάση αυτά οι περιπτώσεις είναι οι ακόλουθες.

- Αλλαγή αναλογίας δειγμάτων από 1:10 σε 1:4 και υπερ παράμετροι για κάθε ένα από τα τρία μοντέλα. 500 δείγματα για την κλάση illicit και 2000 για την κλάση licit. Αυτό αντιστοιχεί σε περίπου 10% των συνολικών διαθέσιμων για την πρώτη κλάση. Επιλέξαμε αυτό το ποσοστό διότι σε έτοιμα παραδείγματα κώδικα ή/και έτοιμα σύνολα δεδομένων **[5,8]** χρησιμοποιούν λιγότερα από αυτό το ποσοστό οπότε το αντιμετωπίσαμε σαν άνω όριο στον διαχωρισμό που κάναμε. Το ποσοστό για την δεύτερη κλάση είναι περίπου 5%.

- Αλλαγή αναλογίας δειγμάτων σε 1:1 με under sampling της κλάσης με τα περισσότερα δείγματα. 500 δείγματα για κάθε κλάση. Αυτή η μέθοδος αποτελεί μία πολύ γνωστή μέθοδο αντιμετώπισης του προβλήματος της αναλογίας των διαθέσιμων δειγμάτων για τις κλάσεις οπότε θα το δοκιμάσουμε και εδώ.
- Αλλαγή αναλογίας δειγμάτων σε 1:1. 1000 δείγματα ανά κλάση. Αυτή αποτελεί συμπληρωματική μέθοδο της προηγούμενης και αφορά την αξιοποίηση περισσότερων δειγμάτων για την μέθοδο με τα λιγότερα δείγματα. Σε αυτό το σενάριο αξιολογούμε ταυτόχρονα αυτή τη μέθοδο για αντιμετώπιση του προβλήματος της αναλογίας αλλά και το αποτέλεσμα της χρήσης παραπάνω δειγμάτων στην λύση του προβλήματος που αντιμετωπίζουμε.
- Το τελευταίο σενάριο που θα αξιολογήσουμε σε συνδυασμό με τις υπερ παραμέτρους και για την πρώτη περίπτωση μόνο δηλαδή της αναλογίας 1:4 είναι η χρήση βαρών ή όχι για τις δύο κλάσεις.

Για αυτή την περίπτωση ξεκινήσαμε δοκιμάζοντας υπερ παραμέτρους ώστε να καταλήξουμε σε αυτές που θα εξετάσουμε με βάση το πλέγμα(grid) που θα δημιουργήσουμε. Γενικά ο GraphSAGE πήγαινε καλύτερα και ακολουθούσε ο AGNN και ο GAT είχε κακά αποτελέσματα με βάση αυτά τα πρώτα πειράματα καταλήξαμε στις ακόλουθες υπερ παραμέτρους ανά αλγόριθμο για το επόμενο βήμα.

Για τον GAT θα εξετάσουμε τα παρακάτω.

- Αριθμός επιπέδων [2,3]
- Μέγεθος Αναπαράστασης [32,64]
- Loader data object και Neighbor Loader
- Αριθμός heads [4,8]
- Βάρη εκπαίδευσης [True,False] για το πρώτο σενάριο μόνο.

Για τον AGNN θα εξετάσουμε τα παρακάτω

- Αριθμός επιπέδων [2,3]
- Μέγεθος Αναπαράστασης [32,64]
- Loader data object και Neighbor Loader
- Βάρη εκπαίδευσης [True,False] για το πρώτο σενάριο μόνο.

Για τον GraphSAGE θα εξετάσουμε τα παρακάτω.

- Αριθμός επιπέδων [2,3]
- Αριθμός Αναπαράστασης [32,64,128]
- Είδος aggregator [‘mean’, ‘max’]
- Είδος Loader. Neighbor Loader μόνο. Βάρη μόνο για το πρώτο σενάριο.

Να σημειωθεί εδώ ότι εφόσον έχουμε του πόρους να πραγματοποιήσουμε την εκπαίδευση σε full batch δηλαδή με όλα τα δεδομένα σε ένα βήμα δεν θα χρησιμοποιήσουμε mini batches. Με βάση αυτό το αντικείμενο τύπου Data της Pytorch Geometric που αναπαριστά τον γράφο θα χρησιμοποιείται απευθείας σαν είσοδος στο μοντέλο και στην συνάρτηση σφάλματος. Επίσης υπάρχει άλλη μία κλάση που έχει δημιουργηθεί για τον αλγόριθμο GraphSAGE και ονομάζεται Neighbor Loader. Είδαμε παραδείγματα όπου χρησιμοποιείται και για άλλους αλγορίθμους και δεν δίνει κάποιο σφάλμα αλλά επειδή δεν ξέρουμε πως λειτουργεί θα την αξιολογήσουμε σαν υπερ παράμετρο για τους GAT/AGNN.

4.2.2 Προεκπαιδευμένα βάρη με DGI και οι καλύτερες περιπτώσεις από το πρώτο

Το δεύτερο πείραμα αφορά τον αλγόριθμο DGI. Εδώ θα αξιολογήσουμε τρία σενάρια με χρήση των τριών αλγορίθμων που εξετάσαμε και στο προηγούμενο και τις καλύτερες αρχιτεκτονικές.

- Το πρώτο αφορά εκπαίδευση του καλύτερου κωδικοποιητή(encoder) που προέκυψε από την προηγούμενη διαδικασία για κάθε έναν απο τους τρείς αλγορίθμους και εκπαίδευση με DGI και χρήση του σε έναν κατηγοριοποιητή (classifier) MLP χωρίς να εκπαιδεύσουμε τις παραμέτρους του (τα βάρη του encoder σε κατάσταση freeze).

- Το δεύτερο αφορά την χρήση του καλύτερου encoder που προέκυψε από τα παραπάνω σε έναν classifier με τα βάρη του εκπαιδεύσιμα ώστε να δούμε αν δίνει καλύτερο αποτέλεσμα με τα προεκπαιδευμένα βάρη.
- Το τρίτο σενάριο είναι να κρατήσουμε τις αναπαραστάσεις από την εκπαίδευση με τον DGI για κάθε μοντέλο και θα εκπαιδεύσουμε Random Forest Classifier.

Εδώ να σημειωθεί ότι στα παραδείγματα που υπάρχουν που χρησιμοποιούν τον αλγόριθμο αυτό αξιολογούν αυτήν την μέθοδο χρησιμοποιώντας πολύ λιγότερα δεδομένα εκπαίδευσης σε σχέση με το σενάριο ημι επιβλεπόμενης μάθησης.

4.2.3 Αλγόριθμος node2vec και κατηγοριοποιητής Random Forest

Το τρίτο πείραμα αφορά τον αλγόριθμο node2vec και έναν Random Forest αλγόριθμο. Εδώ θα εξετάσουμε τρία σενάρια.

- Το πρώτο είναι χρήση των χαρακτηριστικών(features) που προκύπτουν από τις αναπαραστάσεις και από τον node2vec που αφορά την δομή του γράφου και των χαρακτηριστικών που συνοδεύουν κάθε κόμβο.
- Το δεύτερο είναι με χρήση μόνο των χαρακτηριστικών από τον node2vec.
- Το τρίτο είναι με χρήση μόνο των χαρακτηριστικών των κόμβων.

4.2.4 Νευρωνικά χωρίς features

Εδώ αξιολογούμε την περίπτωση των νευρωνικών που μαθαίνουν μόνο από την τοπολογία του γράφου και όχι από τα features που συνοδεύουν τους κόμβους. Για να το κάνουμε αυτό **[4]** υπολογίζουμε το degree του κάθε κόμβου και αυτό το χαρακτηριστικό είναι η είσοδος στο νευρωνικό. Εναλλακτικά μπορεί να χρησιμοποιηθεί ένα one hot vector για κάθε κόμβο **[4]** Αυτό το πείραμα αφορά τις τρεις καλύτερες αρχιτεκτονικές των GAT/AGNN/GraphSAGE που προέκυψαν από το πρώτο πείραμα.

4.2.5 Αναπαραστάσεις από Νευρωνικά και Random Forest

Εδώ για τις καλύτερες αρχιτεκτονικές της πρώτης περίπτωσης θα κρατήσουμε τις αναπαραστάσεις που προκύπτουν και θα τις δώσουμε σαν είσοδο στον Random Forest για να συγκρίνουμε την επίδοση.
Για την περίπτωση αυτή αξιολογούμε τις αναπαραστάσεις που προκύπτουν από τα νευρωνικά αλλά αντί για MLP χρησιμοποιούμε Random Forest Classifier.

4.3 Μετρικές Αξιολόγησης

Με βάση όσα αναφέραμε σε προηγούμενες ενότητες και για την τελική αξιολόγηση πρέπει να διαλέξουμε ένα μετρικό με βάση το οποίο θα αξιολογήσουμε την επίδοση του μοντέλου κατά την διάρκεια των πειραμάτων. Αποφασίζουμε να καταγράψουμε πολλά από αυτά **[66]** με στόχο παρουσιάζοντας τα αποτελέσματα και ανάλογα με το τι θεωρούμε σημαντικό να επιλέγουμε την κατάλληλη αρχιτεκτονική και αλγόριθμο. Τα μετρικά είναι accuracy, auroc, recall, precision, f1. Για την εκπαίδευση μπορούμε να παρακολουθούμε το σφάλμα στο validation set ή κάποιο από τα παραπάνω. Επιλέξαμε να παρακολουθούμε το σφάλμα (loss) στο σύνολο validation για την επιλογή των καλύτερων παραμέτρων του μοντέλου, το οποίο στην συνέχεια θα αξιολογηθεί στο σύνολο test διότι αυτό συνήθως γίνεται σε αυτά τα σενάρια. Αυτό δεν αποκλείει την επιλογή κάποιου άλλου μετρικού για την επιλογή του καλύτερου μοντέλου. Στην συνέχεια θυμίζουμε τι σημαίνει το κάθε μετρικό για λόγους πληρότητας.

- Accuracy

Αυτό το μετρικό σημαίνει το πλήθος των σωστών προβλέψεων διά το σύνολο των δειγμάτων.

- Auroc

Μετρικό ανα κλάση που δείχνει πόσο καλά ξεχωρίζει ο αλγόριθμος τις κλάσεις. Δέχεται ως είσοδο τις πραγματικές τιμές (ground truth) και την έξοδο με softmax του δικτύου και υπολογίζει το εμβαδόν της καμπύλης που προκύπτει, για διαφορετικά κατώφλια (thresholds), ανάμεσα στο TPR και FPR.

- Precision

Μετρικό ανα κλάση που ορίζεται ως οι σωστές προβλέψεις για την κλάση δια τον συνολικό αριθμό προβλέψεων που έγιναν για αυτή τη κλάση.

- Recall

Μετρικό ανα κλάση που ορίζεται ως οι σωστές προβλέψεις ανά κλάση διά τον συνολικό αριθμό δειγμάτων για αυτή την κλάση.

- F1

Μετρικό ανά κλάση που ορίζεται ως 2*Recall*Precision διά Recall + Precision.

4.4 Πειραματικά αποτελέσματα

Εδώ παρουσιάζουμε τα αποτελέσματα. Η ενότητα αυτή χωρίζεται σε πέντε μέρη όπως περιγράφονται στην ενότητα 4.2.

4.4.1 Αποτελέσματα Πειράματος για GAT/AGNN/GraphSAGE

Εδώ παρουσιάζουμε τα αποτελέσματα για το πρώτο πείραμα 4.2.1 δηλαδή υπερ παράμετροι και τρία σενάρια για να αντιμετωπίσουμε το imbalance. Παρουσιάζουμε τα αποτελέσματα ανά αλγόριθμο.
Θυμίζουμε ότι σε αυτό το σενάριο έχουμε αναλογία δειγμάτων 1:4. Δίνουμε τα αποτελέσματα για τους τρεις αλγορίθμους και τα σχολιάζουμε. Οι γραμμές είναι οι αρχιτεκτονικές που δοκιμάστηκαν και οι στήλες είναι τα μετρικά. Όπου υπάρχουν δύο τιμές αναφέρονται στα αποτελέσματα για τις δύο κλάσεις και όπου μία είναι το μέσο όρο για τις δύο κλάσεις. Η πρώτη κλάση είναι η minority (illicit).

4.4.1.1 Αποτελέσματα GAT

Εδώ οι γραμμές είναι αρχιτεκτονικές και ο πρώτος αριθμός είναι τα επίπεδα, ο δεύτερος το μέγεθος της αναπαράστασης, το τρίτο το είδος του loader, το τέταρτο ο αριθμός των attention heads και το τελευταίο αν χρησιμοποιήθηκαν βάρη ή όχι στην εκπαίδευση.

- Για το πρώτο σενάριο δηλαδή το imbalance έχουμε τα παρακάτω.

Table 2: Αποτελέσματα για GAT σενάριο class imbalance

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-4-True.model	0.645	0.854	['0.192', '0.992']	['0.948', '0.616']	['0.319', '0.760']
2-32-0-4-False.model	0.913	0.858	['0.517', '0.919']	['0.086', '0.992']	['0.148', '0.954']
2-32-0-8-True.model	0.665	0.857	['0.199', '0.990']	['0.934', '0.639']	['0.328', '0.777']
2-32-0-8-False.model	0.911	0.854	['0.491', '0.940']	['0.363', '0.964']	['0.417', '0.952']
2-32-1-4-True.model	0.649	0.852	['0.192', '0.990']	['0.938', '0.621']	['0.319', '0.764']
2-32-1-4-False.model	0.911	0.852	['0.317', '0.914']	['0.018', '0.996']	['0.033', '0.953']
2-32-1-8-True.model	0.632	0.856	['0.186', '0.993']	['0.953', '0.601']	['0.312', '0.749']
2-32-1-8-False.model	0.91	0.854	['0.480', '0.942']	['0.379', '0.961']	['0.424', '0.951']
2-64-0-4-True.model	0.649	0.856	['0.190', '0.989']	['0.925', '0.623']	['0.316', '0.764']
2-64-0-4-False.model	0.914	0.855	['0.522', '0.925']	['0.170', '0.985']	['0.257', '0.954']
2-64-0-8-True.model	0.623	0.856	['0.182', '0.991']	['0.942', '0.593']	['0.305', '0.742']
2-64-0-8-False.model	0.906	0.832	['0.366', '0.919']	['0.101', '0.983']	['0.158', '0.950']
2-64-1-4-True.model	0.628	0.859	['0.185', '0.992']	['0.952', '0.597']	['0.309', '0.745']
2-64-1-4-False.model	0.914	0.856	['0.568', '0.917']	['0.057', '0.996']	['0.104', '0.955']
2-64-1-8-True.model	0.649	0.849	['0.191', '0.989']	['0.926', '0.623']	['0.316', '0.764']
2-64-1-8-False.model	0.905	0.82	['0.267', '0.915']	['0.048', '0.987']	['0.081', '0.950']
3-32-0-4-True.model	0.602	0.834	['0.173', '0.990']	['0.941', '0.570']	['0.293', '0.723']
3-32-0-4-False.model	0.905	0.83	['0.257', '0.915']	['0.045', '0.988']	['0.076', '0.950']
3-32-0-8-True.model	0.533	0.822	['0.153', '0.992']	['0.956', '0.493']	['0.264', '0.658']
3-32-0-8-False.model	0.873	0.838	['0.330', '0.945']	['0.441', '0.914']	['0.378', '0.929']
3-32-1-4-True.model	0.548	0.83	['0.157', '0.991']	['0.950', '0.509']	['0.269', '0.673']
3-32-1-4-False.model	0.89	0.832	['0.353', '0.935']	['0.313', '0.945']	['0.332', '0.940']
3-32-1-8-True.model	0.57	0.838	['0.163', '0.990']	['0.945', '0.534']	['0.278', '0.694']
3-32-1-8-	0.868	0.84	['0.328',	['0.489',	['0.393',

False.model			'0.949']	'0.904']	'0.926']
3-64-0-4-True.model	0.551	0.838	['0.157', '0.990']	['0.949', '0.513']	['0.270', '0.676']
3-64-0-4-False.model	0.893	0.839	['0.383', '0.938']	['0.352', '0.945']	['0.367', '0.942']
3-64-0-8-True.model	0.524	0.816	['0.151', '0.992']	['0.959', '0.482']	['0.261', '0.649']
3-64-0-8-False.model	0.912	0.6	['0.000', '0.912']	['0.000', '1.000']	['0.000', '0.954']
3-64-1-4-True.model	0.546	0.837	['0.157', '0.991']	['0.953', '0.507']	['0.269', '0.671']
3-64-1-4-False.model	0.906	0.833	['0.439', '0.931']	['0.251', '0.969']	['0.319', '0.950']
3-64-1-8-True.model	0.912	0.624	['0.000', '0.912']	['0.000', '1.000']	['0.000', '0.954']
3-64-1-8-False.model	0.912	0.601	['0.000', '0.912']	['0.000', '1.000']	['0.000', '0.954']

Με βάση τα αποτελέσματα του πίνακα 2 βλέπουμε ότι η αρχιτεκτονική μπορεί να παίξει πολύ καθοριστικό ρόλο. Το accuracy κυμαίνεται από 50% το οποίο είναι πολύ κακή απόδοση μέχρι και πάνω από 90%. Δεδομένου όμως του χαρακτηριστικού imbalance δεν αποτελεί καλό κριτήριο αξιολόγησης. Το δεύτερο metric το οποίο θεωρητικά έχει μεγαλύτερη σημασία δεν διαφοροποιείται σημαντικά ανάμεσα στις αρχιτεκτονικές συνεπώς δεν αποτελεί καλή επιλογή αξιολόγησης. Στην συνέχεια έχουμε το precision και το recall για κάθε μία από τις δύο κλάσεις και αυτά τα metrics δεν είναι καλό να χρησιμοποιηθούν διότι το τελευταίο δηλαδή το f1 συνδιάζει αυτές τις δύο πληροφορίες. Με βάση το f1 βλέπουμε ότι η καλύτερη περίπτωση για την minority class είναι αυτή με f1 = 0.42 και f1 της majority 0.95. Συνεπώς κατα την γνώμη μας καλύτερο μοντέλο είναι αυτό με το μεγαλύτερο average f1 με max f1 της minority. Επίσης βλέπουμε ότι υπάρχουν αρχιτεκτονικές όπου το μοντέλο μαθαίνει απλά να διαλέγει την majority class(licit). Αυτό είναι προφανές από τις τιμές 0.0 sto recall/precision της minority class. Επόμενο χαρακτηριστικό που παρατηρούμε είναι την επίδραση των βαρών κατά την εκπαίδευση στα metrics. Σε 12 από τις 16 αρχιτεκτονικές που δοκιμάστηκαν τα f1 για τις δύο κλάσεις ήταν καλύτερα χωρίς την χρήση βαρών. Για την περίπτωση της καλύτερης αρχιτεκτονικής 2-32-1-8 βλέπουμε ότι η περίπτωση χωρίς βάρη δίνει f1 = [0.42, 0.95] και η περίπτωση με βάρη δίνει [0.31, 0.74]. Στην συνέχεια εξετάζουμε τις επιπτώσεις του loader τον οποίο όπως αναφέραμε σε προηγούμενη ενότητα εξετάζουμε ως υπερ

παράμετρο. Εδώ παρατηρούμε ότι στις περισσότερες περιπτώσεις δεν φαίνεται να παίζει ρόλο αλλά σε κάποιες βλέπουμε διαφορά. Ο λόγος που δοκιμάσαμε τόσα πολλά μοντέλα είναι το ότι η απόδοση του μοντέλου GAT δεν είναι αυτή που αναμέναμε. Να σημειωθεί ότι εδώ δεν παρουσιάζουμε όλα τα μοντέλα που δοκιμάσαμε. Για παράδειγμα δοκιμάσαμε μοντέλα με ένα επίπεδο καθώς επίσης και μοντέλα όπου το output layer ήταν ενσωματωμένο σε GATConv layer. Η απόδοση τους ήταν απογοητευτική. Αρχιτεκτονική με ένα επίπεδο απλά μάθαινε την majority class.

- Για την δεύτερη περίπτωση, δηλαδή το σενάριο με under sampling έχουμε τα παρακάτω.

Table 3: Αποτελέσματα GAT για σενάριο undersampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-4-False.model	0.574	0.847	['0.161', '0.993']	['0.960', '0.539']	['0.276', '0.698']
2-32-0-8-False.model	0.615	0.851	['0.174', '0.992']	['0.949', '0.585']	['0.295', '0.736']
2-32-1-4-False.model	0.618	0.845	['0.175', '0.992']	['0.950', '0.587']	['0.296', '0.738']
2-32-1-8-False.model	0.641	0.834	['0.182', '0.989']	['0.926', '0.615']	['0.304', '0.758']
2-64-0-4-False.model	0.62	0.855	['0.177', '0.993']	['0.952', '0.589']	['0.298', '0.739']
2-64-0-8-False.model	0.661	0.812	['0.177', '0.975']	['0.824', '0.646']	['0.292', '0.777']
2-64-1-4-False.model	0.885	0.748	['0.181', '0.920']	['0.102', '0.958']	['0.130', '0.938']
2-64-1-8-False.model	0.621	0.846	['0.176', '0.992']	['0.950', '0.590']	['0.298', '0.740']
3-32-0-4-False.model	0.796	0.559	['0.091', '0.917']	['0.157', '0.856']	['0.116', '0.885']
3-32-0-8-False.model	0.527	0.83	['0.148', '0.993']	['0.960', '0.487']	['0.256', '0.653']
3-32-1-4-False.model	0.544	0.825	['0.152', '0.992']	['0.957', '0.506']	['0.262', '0.670']
3-32-1-8-False.model	0.542	0.82	['0.152', '0.992']	['0.959', '0.504']	['0.262', '0.668']
3-64-0-4-False.model	0.537	0.833	['0.150', '0.992']	['0.957', '0.498']	['0.259', '0.663']
3-64-0-8-False.model	0.534	0.822	['0.149', '0.992']	['0.957', '0.495']	['0.258', '0.660']
3-64-1-4-	0.198	0.751	['0.095',	['1.000',	['0.174',

False.model			'1.000']	'0.124']	'0.220']
3-64-1-8-False.model	0.572	0.7	['0.136', '0.961']	['0.758', '0.555']	['0.231', '0.704']

Στον πίνακα 3 βλέπουμε ότι η απόδοση είναι πολύ κακή συγκριτικά με την προηγούμενη περίπτωση. Εδώ το accuracy κυμαίνεται από 19% έως 90%. Το auc roc φαίνεται να διαφοροποιείται σημαντικά ανάμεσα στις αρχιτεκτονικές. Τέλος το καλύτερο μοντέλο εξετάζοντας και πάλι την περίπτωση του καλύτερου f1 για την minority και εν συνεχεία του καλύτερου f1 για την majority είναι το 0.30/0.75 που αντιστοιχεί 2-32-1-8 η ίδια δηλαδή με την προηγούμενη περίπτωση.

- Για το τρίτο σενάριο δηλαδή αυτό με το over sampling έχουμε τα παρακάτω.

Table 4: Αποτελέσματα GAT για σενάριο oversampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-4-False.model	0.644	0.854	['0.177', '0.982']	['0.879', '0.623']	['0.295', '0.762']
2-32-0-8-False.model	0.915	0.689	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']
2-32-1-4-False.model	0.639	0.853	['0.182', '0.990']	['0.936', '0.612']	['0.305', '0.757']
2-32-1-8-False.model	0.621	0.851	['0.174', '0.989']	['0.931', '0.592']	['0.294', '0.741']
2-64-0-4-False.model	0.622	0.856	['0.172', '0.987']	['0.912', '0.595']	['0.290', '0.742']
2-64-0-8-False.model	0.915	0.705	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']
2-64-1-4-False.model	0.663	0.86	['0.190', '0.988']	['0.916', '0.640']	['0.315', '0.777']
2-64-1-8-False.model	0.657	0.844	['0.187', '0.988']	['0.916', '0.633']	['0.311', '0.771']
3-32-0-4-False.model	0.915	0.713	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']
3-32-0-8-False.model	0.531	0.833	['0.148', '0.991']	['0.954', '0.492']	['0.256', '0.658']
3-32-1-4-False.model	0.59	0.835	['0.164', '0.990']	['0.940', '0.557']	['0.279', '0.713']
3-32-1-8-False.model	0.566	0.829	['0.157', '0.991']	['0.947', '0.531']	['0.270', '0.692']
3-64-0-4-False.model	0.915	0.577	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']
3-64-0-8-False.model	0.915	0.727	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']

3-64-1-4-False.model	0.551	0.822	['0.153', '0.991']	['0.950', '0.514']	['0.264', '0.677']
3-64-1-8-False.model	0.915	0.691	['0.000', '0.915']	['0.000', '1.000']	['0.000', '0.956']

Στον πίνακα 4 βλέπουμε ότι η απόδοση είναι χειρότερη και από τις δύο προηγούμενες περιπτώσεις με πολλά από τα μοντέλα να μαθαίνουν απλά την majority class. Το accuracy εδώ κυμαίνεται από 50% έως 91%. Το auc roc διαφοροποιείται σημαντικά. Το καλύτερο f1 όπως το διαλέξαμε στις προηγούμενες δύο περιπτώσεις είναι το [0.31,0.77] και αντιστοιχεί στις αρχιτεκτονικές 2-64-1-8-False και 2-64-1-4-False διαφορετικές από τις προηγούμενες περιπτώσεις.

Με βάση αυτά βλέπουμε πως την καλύτερη απόδοση του GAT την έχουμε για το imbalanced σενάριο με αρχιτεκτονική 2-32-1-8 δηλαδή 2 layers, μέγεθος hidden dimension 32, χρήση του data object κατευθείαν(όχι Neighbor Loader) και αριθμό attention heads 8 και μή χρήση βαρών. Η εξήγηση πρέπει να είναι δεδομένης της συνολικής απόδοση ανα περίπτωση το ότι είναι καλό να διατηρείται το imbalance στο train set. Επίσης βλέπουμε πως το oversampling δεν βοήθησε πράγμα μη αναμενόμενο.

4.4.1.2 Αποτελέσματα AGNN

Εδώ οι γραμμές είναι οι αρχιτεκτονικές και ο πρώτος αριθμός είναι τα επίπεδα του δικτύου, ο δεύτερος είναι το μέγεθος της αναπαράστασης, το τρίτο ο loader και το τέταρτο αν χρησιμοποιήθηκαν βάρη στην εκπαίδευση.

- Για το πρώτο σενάριο δηλαδή το imbalance έχουμε τα παρακάτω.

Table 5: Αποτελέσματα AGNN για το σενάριο class imbalance

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-True.model	0.804	0.941	['0.297', '0.989']	['0.904', '0.795']	['0.447', '0.881']
2-32-0-False.model	0.927	0.898	['0.656', '0.939']	['0.339', '0.983']	['0.447', '0.961']
2-32-1-True.model	0.786	0.931	['0.279', '0.989']	['0.910', '0.774']	['0.427', '0.869']

2-32-1-False.model	0.937	0.919	['0.688', '0.954']	['0.510', '0.978']	['0.586', '0.966']
2-64-0-True.model	0.788	0.942	['0.282', '0.990']	['0.916', '0.776']	['0.431', '0.870']
2-64-0-False.model	0.937	0.925	['0.661', '0.960']	['0.582', '0.971']	['0.619', '0.966']
2-64-1-True.model	0.805	0.943	['0.298', '0.989']	['0.906', '0.795']	['0.448', '0.881']
2-64-1-False.model	0.937	0.924	['0.660', '0.960']	['0.583', '0.971']	['0.619', '0.966']
3-32-0-True.model	0.785	0.925	['0.278', '0.988']	['0.906', '0.774']	['0.425', '0.868']
3-32-0-False.model	0.933	0.907	['0.713', '0.944']	['0.395', '0.985']	['0.509', '0.964']
3-32-1-True.model	0.791	0.929	['0.283', '0.988']	['0.904', '0.781']	['0.432', '0.872']
3-32-1-False.model	0.926	0.874	['0.901', '0.927']	['0.175', '0.998']	['0.293', '0.961']
3-64-0-True.model	0.742	0.931	['0.244', '0.990']	['0.924', '0.725']	['0.386', '0.837']
3-64-0-False.model	0.92	0.892	['0.565', '0.942']	['0.377', '0.972']	['0.452', '0.957']
3-64-1-True.model	0.768	0.934	['0.264', '0.990']	['0.920', '0.754']	['0.410', '0.856']
3-64-1-False.model	0.924	0.903	['0.588', '0.948']	['0.447', '0.970']	['0.508', '0.959']

Στον πίνακα 5 βλέπουμε πολύ καλύτερα αποτελέσματα σε σχέση με το GAT. Το accuracy κινείται ανάμεσα στο 78% και στο 93%. Το auc roc δεν διαφοροποιείται πολύ. Σχετικά με το f1 βλέπουμε ότι είναι σχετικά σταθερό ανάμεσα σε όλες τις αρχιτεκτονικές κάτι που μας κάνει να συμπεράνουμε ότι πρόκειται για μοντέλο που ταιριάζει στο use case που εξετάζουμε. Η καλύτερη επίδοση είναι f1 [0.62,0.95] που αντιστοιχεί στην αρχιτεκτονική 2-64-1-False δηλαδή 2 layers, μέγεθος representation 64, με χρήση data object(οχι Neighbor Loader). Συγκριτικά με την ίδια αρχιτεκτονική με χρήση DataLoader η απόδοση είναι πολύ καλύτερη και είναι πιθανό να οφείλεται σε αυτό. Περαιτέρω ανάλυση των δεδομένων για να δούμε αν παίζει ρόλο αυτό απαιτείται και εδώ.

- Για το δεύτερο σενάριο, δηλαδή το downsampling έχουμε τα παρακάτω.

Table 6: Αποτελέσματα για AGNN για σενάριο undersampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-False.model	0.816	0.936	['0.302', '0.988']	['0.895', '0.808']	['0.451', '0.889']
2-32-1-False.model	0.813	0.935	['0.298', '0.988']	['0.892', '0.805']	['0.446', '0.887']
2-64-0-False.model	0.778	0.934	['0.264', '0.989']	['0.910', '0.765']	['0.409', '0.863']
2-64-1-False.model	0.808	0.938	['0.294', '0.989']	['0.905', '0.799']	['0.443', '0.884']
3-32-0-False.model	0.773	0.925	['0.258', '0.988']	['0.901', '0.761']	['0.401', '0.860']
3-32-1-False.model	0.74	0.924	['0.234', '0.989']	['0.916', '0.723']	['0.373', '0.836']
3-64-0-False.model	0.755	0.926	['0.245', '0.990']	['0.916', '0.740']	['0.387', '0.847']
3-64-1-False.model	0.763	0.926	['0.252', '0.989']	['0.913', '0.749']	['0.395', '0.853']

Στον πίνακα 6 βλέπουμε ότι τα αποτελέσματα συνολικά είναι και πάλι ανώτερα του GAT και δεν υπάρχουν ακραίες τιμές πράγμα που σημαίνει ότι το μοντέλο αυτό και σε αυτό το σενάριο αποδίδει ως έναν βαθμό. Το accuracy δεν έχει μεγάλη απόκλιση από το μέσο όρο. Το auc roc δεν διαφοροποιείται πολύ συμπεριφορά που συμβαδίζει με τα υπόλοιπα metrics. Το καλύτερο αποτέλεσμα είναι αυτό με f1 [0.44,0.88] και αντιστοιχεί στην αρχιτεκτονική 2-64-1-False δηλαδή 2 layers μέγεθος αναπαράστασης 64 και data object. Η απόδοση του ίδιου μοντέλου με Neighbor Loader είναι κοντά αλλά υστερεί ως προς την minority class. Εδώ όπως και προηγουμένως θα πρέπει να δοκιμάσουμε να προσθέσουμε ένα Linear layer μετά το propagation layer καθώς δεν είναι σαφής ο τρόπος που υλοποιείται ο αλγόριθμος στο pytorch geometric.

- Για το τρίτο σενάριο, δηλαδή το over sampling έχουμε τα παρακάτω.

Table 7: Αποτελέσματα AGNN για σενάριο oversampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-0-False.model	0.794	0.935	['0.279', '0.988']	['0.901', '0.785']	['0.426', '0.875']
2-32-1-False.model	0.812	0.931	['0.296', '0.987']	['0.882', '0.806']	['0.443', '0.887']
2-64-0-	0.793	0.938	['0.278',	['0.906',	['0.425',

False.model			'0.989']	'0.782']	'0.873']
2-64-1-False.model	0.815	0.942	['0.301', '0.988']	['0.895', '0.808']	['0.450', '0.889']
3-32-0-False.model	0.77	0.92	['0.255', '0.987']	['0.894', '0.759']	['0.397', '0.858']
3-32-1-False.model	0.756	0.922	['0.246', '0.989']	['0.912', '0.742']	['0.388', '0.848']
3-64-0-False.model	0.782	0.926	['0.266', '0.987']	['0.892', '0.772']	['0.409', '0.866']
3-64-1-False.model	0.748	0.927	['0.241', '0.990']	['0.918', '0.732']	['0.381', '0.842']

Στον πίνακα 7 παρατηρούμε πάλι ότι δεν υπάρχουν ακραίες τιμές και η απόδοση είναι γενικά ίδια ανάμεσα στις διαφορετικές αρχιτεκτονικές. Το accuracy κινείται γύρω στο 80% το auc_roc είναι σταθερό και το f1 κυμαίνεται γύρω στο 40% για την minority class. Το καλύτερο μοντέλο είναι αυτό με f1 [0.45, 0.88] και αντιστοιχεί στο 2-64-1-False δηλαδή 2 layers μέγεθος αναπαράστασης 64 και loader data object. Βλέπουμε εδώ ότι η ίδια αρχιτεκτονική δίνει τα καλύτερα αποτελέσματα και στα τρία σενάρια που εξετάσαμε.

4.4.1.3 Αποτελέσματα GraphSAGE

Εδώ οι γραμμές είναι οι αρχιτεκτονικές και ο πρώτος αριθμός είναι τα επίπεδα, ο δεύτερος είναι το μέγεθος της αναπαράστασης, ο τρίτος το είδος του aggregator, ο τέταρτος το είδος του loader και ο πέμπτος αν χρησιμοποιήθηκαν βάρη ή όχι.

- Για το πρώτο σενάριο, δηλαδή το imbalance έχουμε τα παρακάτω.

Table 8: Αποτελέσματα GraphSAGE για το σενάριο class imbalance

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-max-0-True.model	0.925	0.959	['0.547', '0.984']	['0.839', '0.933']	['0.662', '0.958']
2-32-max-0-False.model	0.959	0.961	['0.778', '0.976']	['0.752', '0.979']	['0.764', '0.978']
2-32-mean-0-True.model	0.93	0.958	['0.566', '0.984']	['0.840', '0.938']	['0.676', '0.960']
2-32-mean-0-	0.958	0.96	['0.748', '0.979']	['0.778', '0.975']	['0.762', '0.977']

False.model					
2-64-max-0-True.model	0.919	0.961	['0.522', '0.987']	['0.871', '0.923']	['0.652', '0.954']
2-64-max-0-False.model	0.96	0.96	['0.776', '0.978']	['0.768', '0.979']	['0.772', '0.978']
2-64-mean-0-True.model	0.933	0.96	['0.583', '0.984']	['0.838', '0.942']	['0.688', '0.963']
2-64-mean-0-False.model	0.963	0.962	['0.796', '0.979']	['0.778', '0.981']	['0.787', '0.980']
2-128-max-0-True.model	0.934	0.961	['0.588', '0.984']	['0.840', '0.944']	['0.692', '0.963']
2-128-max-0-False.model	0.963	0.962	['0.809', '0.976']	['0.753', '0.983']	['0.780', '0.980']
2-128-mean-0-True.model	0.927	0.96	['0.556', '0.985']	['0.853', '0.935']	['0.673', '0.959']
2-128-mean-0-False.model	0.963	0.963	['0.790', '0.979']	['0.781', '0.980']	['0.785', '0.980']
3-32-max-0-True.model	0.928	0.962	['0.560', '0.985']	['0.851', '0.936']	['0.675', '0.960']
3-32-max-0-False.model	0.961	0.962	['0.806', '0.974']	['0.729', '0.983']	['0.766', '0.979']
3-32-mean-0-True.model	0.932	0.961	['0.575', '0.984']	['0.842', '0.940']	['0.683', '0.962']
3-32-mean-0-False.model	0.963	0.959	['0.809', '0.976']	['0.750', '0.983']	['0.778', '0.980']
3-64-max-0-True.model	0.94	0.964	['0.616', '0.984']	['0.844', '0.949']	['0.712', '0.967']
3-64-max-0-False.model	0.963	0.962	['0.818', '0.975']	['0.739', '0.984']	['0.777', '0.980']
3-64-mean-0-True.model	0.932	0.967	['0.576', '0.986']	['0.863', '0.939']	['0.691', '0.962']
3-64-mean-0-False.model	0.963	0.961	['0.809', '0.976']	['0.752', '0.983']	['0.780', '0.980']
3-128-max-0-True.model	0.936	0.958	['0.600', '0.981']	['0.805', '0.948']	['0.687', '0.964']
3-128-max-0-False.model	0.963	0.963	['0.805', '0.978']	['0.767', '0.982']	['0.785', '0.980']
3-128-mean-0-	0.919	0.965	['0.521', '0.987']	['0.876', '0.923']	['0.653', '0.954']

True.model					
3-128-mean-0-False.model	0.967	0.963	['0.843', '0.978']	['0.764', '0.986']	['0.801', '0.982']

Στον πίνακα 8 βλέπουμε ότι η απόδοση είναι πολύ καλύτερη από τα δύο προηγούμενα. Εδώ το accuracy είναι σταθερά πάνω από 90%. Το ίδιο ισχύει και για το auc roc. Στα precision και recall η απόδοση διαφοροποιείται αρκετά ανάμεσα στις διαφορετικές αρχιτεκτονικές. Τέλος με κριτήριο όπως και στα προηγούμενα βλέπουμε ότι η καλύτερη απόδοση είναι αυτή με 3-128-mean-0-False, δηλαδή 3 επίπεδα, μέγεθος αναπαράστασης 128, mean aggregator και οχι βάρη και F1 [0.80, 0.95]. Και εδώ πρέπει να δούμε την επίδραση της χρήσης βαρών στην εκπαίδευση. Συγκριτικά με την καλύτερη αρχιτεκτονική βλέπουμε ότι η απόδοση χωρίς βάρη είναι κατα πολύ ανώτερη από αυτήν με βάρη. Αξίζει να σημειωθεί ότι υπάρχουν και άλλες αρχιτεκτονικές που τα πάνε καλά και το ότι η καλύτερη είναι το πιο πολύπλοκο μοντέλο.

- Για την δεύτερη περίπτωση, δηλαδή under sampling έχουμε τα παρακάτω.

Table 9: Αποτελέσματα GraphSAGE για το σενάριο undersampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-max-0-False.model	0.911	0.946	['0.486', '0.981']	['0.808', '0.921']	['0.607', '0.950']
2-32-mean-0-False.model	0.926	0.948	['0.542', '0.981']	['0.799', '0.938']	['0.646', '0.959']
2-64-max-0-False.model	0.913	0.947	['0.491', '0.983']	['0.830', '0.920']	['0.617', '0.951']
2-64-mean-0-False.model	0.934	0.946	['0.586', '0.978']	['0.765', '0.950']	['0.663', '0.964']
2-128-max-0-False.model	0.907	0.941	['0.471', '0.980']	['0.800', '0.917']	['0.593', '0.948']
2-128-mean-0-False.model	0.935	0.947	['0.587', '0.978']	['0.771', '0.950']	['0.666', '0.964']
3-32-max-0-False.model	0.909	0.95	['0.478', '0.985']	['0.855', '0.914']	['0.613', '0.948']
3-32-mean-0-	0.923	0.947	['0.528', '0.982']	['0.814', '0.933']	['0.641', '0.957']

False.model					
3-64-max-0-False.model	0.92	0.95	['0.516', '0.983']	['0.826', '0.928']	['0.635', '0.955']
3-64-mean-0-False.model	0.941	0.931	['0.685', '0.960']	['0.560', '0.976']	['0.617', '0.968']
3-128-max-0-False.model	0.929	0.946	['0.556', '0.980']	['0.793', '0.941']	['0.653', '0.960']
3-128-mean-0-False.model	0.925	0.947	['0.537', '0.980']	['0.797', '0.936']	['0.642', '0.958']

Στον πίνακα 9 βλέπουμε ότι ο GraphSAGE αποδίδει σχετικά καλά σε σχέση με τους άλλους δύο και σε αυτό το σενάριο. Βλέπουμε ότι accuracy και aucroc είναι σταθερά ανάμεσα στις αρχιτεκτονικές οπότε δεν μας δίνουν σημαντική πληροφορία για το καλύτερο μοντέλο. Επίσης βλέπουμε πολύ καλή απόδοση γενικά και για τις δύο κλάσεις σχετικά με το recall. Τέλος όπως και στα προηγούμενα το F1 κινείται σε σταθερά επίπεδα για όλες τις αρχιτεκτονικές. Καλύτερο μοντέλο είναι το 3-128-max-0-False δηλαδή το ίδιο με πριν αλλά με aggregator max. Το αντίστοιχο με mean έχει σχεδόν ίδια απόδοση.

- Για την τρίτη περίπτωση, δηλαδή του over sampling έχουμε τα παρακάτω.

Table 10: Αποτελέσματα για GraphSAGE και σενάριο oversampling

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-max-0-False.model	0.944	0.956	['0.634', '0.980']	['0.792', '0.958']	['0.704', '0.969']
2-32-mean-0-False.model	0.94	0.946	['0.624', '0.975']	['0.736', '0.959']	['0.676', '0.967']
2-64-max-0-False.model	0.932	0.949	['0.570', '0.979']	['0.782', '0.945']	['0.659', '0.962']
2-64-mean-0-False.model	0.943	0.955	['0.628', '0.981']	['0.795', '0.956']	['0.702', '0.968']
2-128-max-0-False.model	0.932	0.952	['0.572', '0.980']	['0.796', '0.945']	['0.665', '0.962']
2-128-mean-0-	0.95	0.953	['0.711', '0.972']	['0.699', '0.974']	['0.705', '0.973']

False.model					
3-32-max-0-False.model	0.947	0.955	['0.659', '0.979']	['0.778', '0.963']	['0.714', '0.971']
3-32-mean-0-False.model	0.949	0.954	['0.685', '0.976']	['0.740', '0.969']	['0.712', '0.972']
3-64-max-0-False.model	0.943	0.96	['0.625', '0.982']	['0.808', '0.955']	['0.705', '0.968']
3-64-mean-0-False.model	0.942	0.95	['0.629', '0.978']	['0.768', '0.958']	['0.692', '0.968']
3-128-max-0-False.model	0.94	0.949	['0.621', '0.977']	['0.759', '0.957']	['0.683', '0.967']
3-128-mean-0-False.model	0.952	0.95	['0.731', '0.971']	['0.688', '0.977']	['0.709', '0.974']

Στον πίνακα 10 βλέπουμε ότι η επίδοση είναι καλύτερη από το δεύτερο σενάριο δηλαδή αυτό της περιπτώσεις balance με undersampling της majority κλάσης αλλά χειρότερα της περίπτωσης όπου διατηρείται η διαφορετική αναλογία ανάμεσα στις δύο κλάσεις.

4.4.2 Αποτελέσματα για DGI

Εδώ παρουσιάζουμε τα αποτελέσματα από το 4.2.2, δηλαδή τα καλύτερα μοντέλα που προέκυψαν από το προηγούμενο με χρήση DGI. Παρουσιάζουμε τα αποτελέσματα ανα αλγόριθμο.
Εδώ για το μοντέλο GAT θα εξετάσουμε την αρχιτεκτονική 2-32-1-8-False δηλαδή 2 layers μεγεθος αναπαράστασης 32, εκπαίδευση με το function που χρησιμοποιεί το data object, 8 attention heads και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

Για το μοντέλο AGNN θα εξετάσουμε την αρχιτεκτονική 2-64-1-False δηλαδή 2 layers μέγεθος αναπαράστασης 64 train function που χρησιμοποιεί το data object και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

Για το μοντέλο GraphSAGE θα εξετάσουμε την αρχιτεκτονική 3-128-mean-0-False δηλαδή 3 layers, μέγεθος αναπαράστασης 128, είδος aggregator mean, εκπαίδευση με το

train function που χρησιμοποιεί τον Neighbor Loader και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

4.4.2.1 Αποτελέσματα GraphSAGE

Αρχικά εκπαιδεύουμε τον DGI με τον encoder GraphSAGE. Το loss βλέπουμε ότι ξεκινάει από περίπου 1.5 και για 200 εποχές πέφτει σταθερά και φτάνει στο 0.54

Στην συνέχεια πραγματοποιούμε εκπαίδευση του classifier με τα βάρη του encoder σε κατάσταση freeze για 200 εποχές. Τα αποτελέσματα είναι τα παρακατω.

Table 11: Αποτελέσματα για DGI με encoder GraphSAGE και βάρη freeze

Architecture	Accuracy	AucRoc	Precision	Recall	F1
sage_freeze	0.912	0.745	[0.000, 0.912]	[0.000, 1.000]	[0.000, 0.954]

Στον πίνακα 11 βλέπουμε ότι με βάρη σε κατάσταση freeze ο Linear classifier μαθαίνει στην ουσία την majority class.

Στην συνέχεια τρέχουμε το ίδιο με τα βάρη του encoder εκπαιδεύσιμα. Τα αποτελέσματα είναι τα εξής.

Table 12: Αποτελέσματα DGI με encoder GraphSAGE και βάρη εκπαιδεύσιμα

Architecture	Accuracy	AucRoc	Precision	Recall	F1
sage_train_enc	0.96	0.963	[0.769, 0.979]	[0.780, 0.977]	[0.774, 0.978]

Στον πίνακα 12 βλέπουμε ότι η απόδοση είναι περίπου ίδια με την απόδοση χωρίς προ εκπαίδευση που είχαμε για το σενάριο end to end που αξιολογήσαμε στην προηγούμενη

ενότητα. Βλέπουμε ότι τα αποτελέσματα δεν είναι καλύτερα από αυτά που είχαμε χωρίς DGI.

Στην συνέχεια έχουμε τα αποτελέσματα από τα embendings και τον Random Forest.

Table 13: Αποτελέσματα για DGI με GraphSAGE και Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
sage_RF	0.926		[0.615, 0.945]	[0.413, 0.975]	[0.494, 0.960]

Παρατηρούμε ότι τα αποτελέσματα είναι πολύ χειρότερα από την εκπαίδευση end to end με Linear Classifier. Χειρότερα κατα 30 μονάδες στο f1 της minority.

4.4.2.2 Αποτελέσματα GAT

Αρχικά εκπαιδεύουμε τον DGI με τον encoder GAT. Το loss βλέπουμε ότι ξεκινάει από περίπου 1.5 και για 200 εποχές πέφτει σταθερά και φτάνει στο 1.32. Σε αντίθεση με τον SAGE βλέπουμε ότι το loss δεν πέφτει πολύ, σχεδόν καθόλου.

Εκπαιδεύοντας με βάρη σε κατάσταση freeze έχουμε τα παρακάτω.

Table 14: Αποτελέσματα για DGI με encoder GAT και βάρη freeze

Architecture	Accuracy	AucRoc	Precision	Recall	F1
gat_freeze	0.912	0.706	[0.000, 0.912]	[0.000, 1.000]	[0.000, 0.954]

Στον πίνακα 14 βλέπουμε το ίδιο αποτέλεσμα με τον SAGE, ο αλγόριθμος μαθαίνει να επιλέγει την majority class.

Στην συνέχεια τρέχουμε το ίδιο με τα βάρη του encoder εκπαιδεύσιμα. Τα αποτελέσματα είναι τα εξής.

Table 15: Αποτελέσματα για DGI με encoder GAT και βάρη εκπαιδεύσιμα

Architecture	Accuracy	AucRoc	Precision	Recall	F1
gat_train	0.917	0.859	[0.553, 0.934]	[0.275, 0.979]	[0.367, 0.956]

Στον πίνακα 15 βλέπουμε ότι η απόδοση είναι λίγο χειρότερη για αυτήν που είχαμε χωρίς αρχικοποίηση βαρών από τον DGI αλλά αυτό μπορεί να είναι και τυχαίο.

Στην συνέχεια έχουμε την περίπτωση embendings και Random Forest. Τα αποτελέσματα είναι τα εξής

Table 16: Αποτελέσματα για DGI με encoder gat και Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
gat_RF	0.921		[0.559, 0.952]	[0.505, 0.961]	[0.530, 0.957]

Στον πίνακα 16 βλέπουμε ότι η απόδοση είναι πολύ καλύτερη από αυτή που έχουμε από εκπαίδευση με το καλύτερο σενάριο end to end με Linear classifier. Καλύτερη σχεδόν 10 μονάδες για το F1 της minority.

4.4.2.3 Αποτελέσματα AGNN

Το loss ξεκινάει από 1.43 και φτάνει στο 1.36. Και εδώ σε αντίθεση με τον SAGE το loss δεν πέφτει σημαντικά.

Για την περίπτωση με τα βάρη του encoder σε κατάσταση freeze έχουμε τα παρακάτω αποτελέσματα.

Table 17: Αποτελέσματα για DGI με ecoder AGNN και βάρη freeze

Architecture	Accuracy	AucRoc	Precision	Recall	F1
agnn_freeze	0.912	0.66	[0.000, 0.912]	[0.000, 1.000]	[0.000, 0.954]

Με βάση τον πίνακα 17 βλέπουμε ότι και εδώ ο classifier μαθαίνει την majority class.

Για την περίπτωση με τα βάρη του encoder εκπαιδεύσιμα έχουμε τα παρακάτω αποτελέσματα.

Table 18: Αποτελέσματα για DGI με ecoder AGNN και βάρη εκπαιδεύσιμα

Architecture	Accuracy	AucRoc	Precision	Recall	F1
agnn_train	0.936	0.926	[0.667, 0.956]	[0.532, 0.975]	[0.592, 0.965]

Παρατηρούμε από τον πίνακα 18 ότι τα αποτελέσματα είναι σχεδόν ίδια με την περίπτωση όπου δεν ήταν αρχικοποιημένα από τον DGI.

Για την περίπτωση με τα embendings που προκύπτουν και χρήση Random Forest τα αποτελέσματα είναι τα εξής.

Table 19: Αποτελέσματα για DGI με encoder AGNN και Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
agnn_RF	0.938		[0.692, 0.956]	[0.532, 0.977]	[0.601, 0.966]

Παρατηρούμε στον πίνακα 19 ότι τα αποτελέσματα είναι συγκρίσιμα με αυτά που είχαμε με την εκπαίδευση end to end για την καλύτερη περίπτωση.

Βλέπουμε ότι για την περίπτωση μας και για τα δύο πρώτα πειράματα που δοκιμάσαμε δεν προσφέρει κάτι σημαντικό. Για το τρίτο όμως όπου δίνουμε τα embendings στον Random Forest η επίδοση είναι ίδια για το AGNN, πολύ καλύτερη για το GAT και πολύ χειρότερη για το GraphSAGE, Αξιοσημείωτο είναι το γεγονός ότι για την περίπτωση των GAT/AGNN το loss στην εκπαίδευση με DGI δεν πέφτει σχεδόν καθόλου αλλά για τον GraphSAGE παρόλο που πέφτει τα αποτελέσματα είναι απογοητευτικά. Αυτό πολύ

πιθανόν να συμβαίνει διότι έχουμε πολλά features σε σύγκριση με τους άλλους δύο για την περίπτωση του GraphSAGE κάτι που μπορεί να οδηγεί σε overfitting.

4.4.3 Αποτελέσματα node2vec Random Forest

Εδώ παρουσιάζουμε τα αποτελέσματα για την περίπτωση 4.2.3 δηλαδή node2vec και Random Forest.

- Για την πρώτη περίπτωση δηλαδή για τα features από κάθε κόμβο αλλά και τα features από τον node2vec έχουμε τα παρακάτω αποτελέσματα.

Table 20: Αποτελέσματα για node2vec + node features

Architecture	Accuracy	AucRoc	Precision	Recall	F1
node2vec + node features	0.984		[0.964, 0.985]	[0.850, 0.996]	[0.903, 0.991]

Με βάση τον πίνακα 20 βλέπουμε ότι η επίδοση είναι πολύ καλύτερη από την βέλτιστη που πέτυχαν τα νευρωνικά.

- Για την δεύτερη περίπτωση θα τρέξουμε τα ίδια αλλά χωρίς τα 64 features από τα embendings.

Table 21: Αποτελέσματα με χρήση μόνο node features

Architecture	Accuracy	AucRoc	Precision	Recall	F1
node features	0.985		[0.959, 0.987]	[0.869, 0.996]	[0.911, 0.991]

Στον πίνακα 21 βλέπουμε ότι παρόλο που δεν έχουμε πληροφορία από την δομή και την συνδεσιμότητα του γράφου τα χαρακτηριστικά που συνοδεύουν τους κόμβους είναι αρκετά για να δώσουν πολύ καλά αποτελέσματα.

- Για την τρίτη περίπτωση θα τρέξουμε το ίδιο αλλά μόνο με τα 64 features από τα embendings που προέκυψαν από τον node2vec.

Table 22: Αποτελέσματα για node2vec embendings

Architecture	Accuracy	AucRoc	Precision	Recall	F1
node2vec	0.913		[0.537, 0.916]	[0.048, 0.996]	[0.088, 0.954]

Με βάση τον πίνακα 22 βλεπουμε ότι τα features που προέκυψαν από την πληροφορία της δομής του γράφου δεν δίνουν καλά αποτελέσματα.

4.4.4 Αποτελέσματα για Νευρωνικά χωρίς node features

Εδώ για το μοντέλο GAT θα εξετάσουμε την αρχιτεκτονική 2-32-1-8-False δηλαδή 2 layers μέγεθος αναπαράστασης 32, εκπαίδευση με το function που χρησιμοποιεί το data object, 8 attention heads και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

Για το μοντέλο AGNN θα εξετάσουμε την αρχιτεκτονική 2-64-1-False δηλαδή 2 layers μέγεθος αναπαράστασης 64 train function που χρησιμοποιεί το data object και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

Για το μοντέλο GraphSAGE θα εξετάσουμε την αρχιτεκτονική 3-128-mean-0-False δηλαδή 3 layers, μέγεθος αναπαράστασης 128, είδος aggregator mean, εκπαίδευση με το train function που χρησιμοποιεί τον Neighbor Loader και εκπαίδευση χωρίς βάρη στο imbalance σενάριο.

4.4.4.1 Αποτελέσματα για GAT

Τα αποτελέσματα είναι τα παρακάτω.

Table 23: Αποτελέσματα απο αναπαραστάσεις GAT χωρίς node features

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-32-1-8-False	0.461	0.472	[0.081, 0.912]	[0.522, 0.455]	[0.141, 0.607]

Βλέπουμε από τον πίνακα 23 ότι χωρίς πληροφορία για τα χαρακτηριστικά που συνοδεύουν κάθε κόμβο η πληροφορία από την δομή και μόνο δεν είναι αρκετή να δώσει καλά αποτελέσματα.

4.4.4.2 Αποτελέσματα για GraphSAGE

Τα αποτελέσματα είναι τα παρακάτω.

Table 24: Αποτελέσματα απο GraphSAGE χωρίς node features

Architecture	Accuracy	AucRoc	Precision	Recall	F1
3-128-mean-0-Flase	0.915	0.515	[0.000, 0.915]	[0.000, 1.000]	[0.000, 0.956]

Από τον πίνακα 24 βλέπουμε ότι ο classifier μαθαίνει την μία κλάση.

4.4.4.3 Αποτελέσματα για AGNN

Τα αποτελέσματα είναι τα παρακάτω.

Table 25: Αποτελέσματα για AGNN χωρίς node features

Architecture	Accuracy	AucRoc	Precision	Recall	F1
2-64-1-False	0.915	0.461	[0.000, 0.915]	[0.000, 1.000]	[0.000, 0.956]

Από τον πίνακα 25 βλέπουμε αντίστοιχα αποτελέσματα όπως και για τους προηγούμενους.

Και για τις τρεις περιπτώσεις παρατηρούμε ότι οι αναπαραστάσεις που προκύπτουν από την τοπολογία του γράφου μόνο δεν μπορούν να δώσουν καλό αποτέλεσμα. Στις δύο από τις τρεις περιπτώσεις απλά ο αλγόριθμος μαθαίνει την την majority class ενώ στην τρίτη όλα τα metrics όπως accuracy και aucroc δείχνουν ότι η επιλογές είναι στα όρια του τυχαίου.

4.4.5 Αποτελέσματα για αναπαραστάσεις από νευρωνικά με Random Forest

Εδώ κρατάμε τις αναπαραστάσεις που προέκυψαν από τις τρεις καλύτερες αρχιτεκτονικές νευρωνικών και θα τις χρησιμοποιήσουμε σαν είσοδο στον Random Forest.

4.4.5.1 Αποτελέσματα για AGNN

Τα αποτελέσματα είναι τα παρακάτω.

Table 26: Αποτελέσματα για AGNN με Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
agnn_RF	0.933		[0.604, 0.971]	[0.706, 0.955]	[0.651, 0.963]

Παρατηρούμε στον πίνακα 26 ότι είναι πολύ κοντά στα αποτελέσματα που είχαμε στο πρώτο πείραμα δηλαδή χρήση ενός Linear επιπέδου σαν classifier.

4.4.5.2 Αποτελέσματα για GAT

Τα αποτελέσματα είναι τα παρακάτω.

Table 27: Αποτελέσματα απο αναπαραστάσεις GAT με Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
gat_RF	0.906		[0.470, 0.958]	[0.579, 0.937]	[0.519, 0.947]

Παρατηρούμε στον πίνακα 27 ότι τα αποτελέσματα είναι πολύ καλύτερα από αυτά που είχαμε στο πρώτο πείραμα για την minority κλάση περίπου δέκα μονάδες πάνω στο F1.

4.4.5.3 Αποτελέσματα για GraphSAGE

Τα αποτελέσματα είναι τα παρακάτω.

Table 28: Αποτελέσματα απο αναπαραστάσεις GraphSAGE με Random Forest

Architecture	Accuracy	AucRoc	Precision	Recall	F1
sage_RF	0.936		[0.600, 0.982]	[0.829, 0.947]	[0.696, 0.964]

Παρατηρούμε στον πίνακα 28 ότι η απόδοση πέφτει σε σχέση με το πρώτο πείραμα δηλαδή με ένα Linear επίπεδο για classifier κατά περίπου δέκα μονάδες.

Για όλα τα πειράματα αυτής της ενότητας τα δοκιμάσαμε και με την παράμετρο class_weights = 'balanced'. Η απόδοση ήταν συγκρίσιμη και συμπεράνουμε ότι για το σενάριο χρήσης και το σύνολο δεδομένων που έχουμε ούτε εδώ τα βάρη βελτιώνουν την απόδοση.

ΚΕΦΑΛΑΙΟ 5ο

Συμπεράσματα

Παρουσιάσαμε τις βασικές αρχές για τον τομέα την Μηχανικής Μάθησης σε δεδομένα γράφων, τους βασικούς αλγορίθμους και τις βασικές αρχές. Δώσαμε έμφαση στις τεχνολογίες που βασίζονται σε Βαθιά Μηχανική Μάθηση διότι αποτελούν καινούριες λύσεις που έχουν την δυνατότητα να εκμεταλλεύονται ταυτόχρονα πληροφορία και από τα χαρακτηριστικά που συνοδεύουν τον κάθε κόμβο/ακμή αλλά και από την τοπολογία του δικτύου. Στο σενάριο χρήσης που ασχοληθήκαμε με βάση τα αποτελέσματα που προέκυψαν είδαμε ότι πρόκειται για δύσκολο πρόβλημα και το κάθε σενάριο χρήσης έχει διαφορετικές παραμέτρους που πρέπει να ληφθούν υπόψιν. Για τις περιπτώσεις γνωστών συνόλων δεδομένων που παρουσιάζονται στην βιβλιογραφία όπως το cora οι λύσεις νευρωνικών όπως το GAT δίνουν πολύ καλά αποτελέσματα αλλά σε άλλα σενάρια χρήσης και σύνολα δεδομένων δεν είναι όπως είδαμε δεδομένη η καλή επίδοση. Στα πειράματα που πραγματοποιήσαμε είδαμε ότι τα καλύτερα αποτελέσματα προέκυψαν από τον αλγόριθμο GraphSAGE. Η τεχνικές που χρησιμοποιεί, δηλαδή οι aggregators είναι σχετικά απλές αλλά δώσανε τα καλύτερα αποτελέσματα. Συγκριτικά με τα άλλα δύο μοντέλα που βασίζονται στον μηχανισμό attention και με βάση τα πολύ καλά αποτελέσματα που δίνει αυτός ο μηχανισμός σε άλλα προβλήματα Μηχανικής Μάθησης στο δικό μας σενάριο χρήσης δεν έδωσαν καλά αποτελέσματα κάτι που δεν το περιμέναμε. Επίσης είδαμε ότι ο αλγόριθμος DGI και οι αναπαραστάσεις που προκύπτουν με μη επιβλεπόμενο τρόπο για τον αλγόριθμο GAT σε συνδυασμό με κατηγοριοποιητή Random Forest έδωσε σημαντικά καλύτερα αποτελέσματα από την end to end εκπαίδευση με semi supervised τρόπο του encoder και του Linear classifier. Αντίθετα η επίδοση του GraphSAGE για το ίδιο σενάριο δηλαδή μη επιβλεπόμενης μάθησης για την παραγωγή των αναπαραστάσεων έδωσε χειρότερα αποτελέσματα από την αντίστοιχη end to end. Είδαμε επίσης ότι σε σενάριο εκπαίδευσης shallow μοντέλου τα χαρακτηριστικά που συνοδεύουν τον κάθε κόμβο ήταν αρκετά από μόνα τους, δηλαδή χωρίς κάποια γνώση για την τοπολογία του γράφου, να κατηγοριοποιήσουν με πολύ

καλύτερα αποτελέσματα συγκριτικά με τις λύσεις των νευρωνικών τους κόμβους χρησιμοποιώντας τον ίδιο αριθμό δειγμάτων εκπαίδευσης. Το σενάριο αυτό είναι δύσκολο να το εξηγήσουμε γιατί συνέβη διότι το σύνολο δεδομένων δεν δίνει περιγραφή των χαρακτηριστικών του κόμβου όπως αναφέραμε σε αντίστοιχη ενότητα για λόγους πνευματικής ιδιοκτησίας.

Παρόλα αυτά μπορούμε να κάνουμε την υπόθεση ότι το ότι δύο transactionIDs πραγματοποίησαν μία συναλλαγή δεν μας δίνει με απόλυτο τρόπο και με μεγάλη βεβαιότητα κάποια πληροφορία για την ομοιότητα κόμβων. Τέλος θα πρέπει να τονίσουμε ότι όλοι οι επαγωγικοί (inductive) αλγόριθμοι δίνουν συνολικά μεγάλο πλεονέκτημα σε σενάρια χρήσης σαν αυτό που εξετάσαμε σε αυτή την εργασία κάτι που ήταν πολύ δύσκολο να επιτευχθεί με τις λύσεις πριν από αυτούς.

Σαν μελλοντικές εργασίες θα θέλαμε να δοκιμάσουμε και άλλους αλγορίθμους που έχουν προταθεί. Με τα frameworks να πολλαπλασιάζονται και νέες αρχιτεκτονικές ή παραλλαγές υπαρχόντων είναι βασικό να έχουν δοκιμαστεί όλες οι πιθανές λύσεις για κάποιο πιθανό πρόβλημα.

Επίσης θα μπορούσαμε να δούμε και λύσεις που αφορούν την αυτόματη δημιουργία ενός γράφου με βάση τα χαρακτηριστικά κόμβων ή/και ακμών. Πρόκειται για πολλά υποσχόμενο υπο κλάδο για τον οποίο κάναμε μία σύντομη αναφορά και θα είχε ενδιαφέρον να δοκιμάσουμε στο σενάριο χρήσης που εξετάσαμε εναλλακτικούς τρόπους να συνδέσουμε κόμβους δεδομένης της υπόθεσης ότι μία συναλλαγή δεν δίνει αρκετή πληροφορία για την τοπολογία του γράφου.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Z. Song, X. Yang, Z. Xu "Graph-based Semi-supervised Learning: A Comprehensive Review" *IEEE Transactions on Neural Networks and Learning Systems,* 2022.

[2] W. M.Hamilton "Graph Representation Learning Book" *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2020.

[3] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, "Representation Learning for Dynamic Graphs: A Survey" *Journal of Machine Learning Research*, 21, 1-73, 2020.

[4] W. L. Hamilton, R. Ying, J. Leskovec "Representation Learning on Graphs: Methods and Applications", *IEEE Data Engineering Bulletin*, 2017.

[5] S. Xiao, S. Wang, Y. Dai, W. Guo "Graph neural networks in node classification: survey and evaluation" *Machine Vision and Applications*, 2022.

[6] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio "Graph Attention Networks" *ICLR*, 2018.

[7] M. Defferrard, X. Bresson, P. Vandergheynst "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering" *NIPS*, 2016.

[8] T. N. Kipf, M. Welling "Semi-supervised Classification with Graph Convolutional Networks" *ICLR*, 2017.

[9] J. Gasteiger, A. Bojchevski, S. Gunnemann "Predict then Propagate: Graph Neural Networks Meet Personalized Pagerank" *ICLR*, 2019.

[10] F. Wu, T. Zhang, A. Holanda de Souza Jr., C. Fifty, T. Yu, K. Q. Weinberger "Simplifying Graph Convolutional Networks", 36th International Conference on Machine Learning, 2019.

[11] T. N. Kipf, M. Welling "Variational Graph Auto-Encoders", *NIPS*, 2016.

[12] A. Bojchevski, S. Gunnemann "Deep Gaussian Embedding of Graphs: Unsupervised Inductive learning via Ranking", *ICLR*, 2018.

[13] P. Velickovic, W. Fedus, W. L. Hamilton, P. Lio, Y. Bengio, R. D. Hjelm “Deep Graph Infomax”, *ICLR*, 2019.

[14] K. K. Thekumparampil, C. Wang, S. Oh, L. Li “Attention-based Graph Neural Network for Semi-supervised Learning”, *ArXiv*, 2018.

[15] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, M. Bronstein “Temporal Graph Networks for Deep Learning on Dynamic Graphs”, 37th International Conference on Machine Learning, 2020.

[16] Anonymous Authors “Incremental Learning on growing graphs”, Under review as a conference paper at ICLR, 2021.

[17] L. Wu, P. Cui, J. Pei, L. Zhao “Graph Neural Networks Foundations Frontiers and Applications”, Springer, 2022.

[18] L. Borgne, Yann-A and Siblini, Wissam and Lebichot, Bertrand and Bontempi, Gianluca “Reproducible Machine Learning for Credit Card Fraud Detection - Practical Handbook”, *Universite Libre de Bruxelles*, 2022.

[19] W. L. Hamilton, R. Ying, J. Leskovec “Inductive Representation Learning on Large Graphs”, *NIPS*, 2017.

[20] Z. Yang, M. Ding, C. Zhou, H. Yang, J. Zhou, J. Tang “Understanding Negative Sampling in Graph RepresentationLearning”, Conference on Knowledge Discovery and Data Mining, 2020.

[21] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst “Geometric deep learning: going beyond Euclidean data”, *IEEE Signal Processing Magazine*, 2016.

[22] X. Chen “Understanding Spectral Graph Neural Networks”, *ArXiv*, 2021.

[23] GitHub - pyg-team/pytorch_geometric: Graph Neural Network Library for PyTorch.

[24] Y. Liu, M. Jin, S. Pan, C. Zhou, Y. Zheng, F. Xia, P. S. Yu, “Graph Self-Supervised Learning: A Survey ”, *Journal of Latex Class Files*, vol. 14, 2021.

[25] L. Torres, A. S. Blevins, D. Bassett, T. Eliassi-Rad “The Why, How, and When of Representations for Complex Systems”, *SIAM Review,* vol. 63, No. 3, pp. 435–485, 2021.

[26] K. Md. Hasib et al. “A Survey of Methods for Managing the Classification and Solution of Data Imbalance Problem”, *Journal of Computer Science*, 2020.

[27] Unknown author, “What is computer Vision”, ibm.com, 2022.

[28] Unknown author, “What is Natural Language Processing”, ibm.com, 2022.

[29] K. O’Shea, R. Nash “An Introduction to Convolutional Neural Networks”, *ArXiv*, 2015.

[30] A. Sherstinsky “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network”, *Elsevier journal “Physica D: Nonlinear Phenomena”,* vol. 404, 2020.

[31] Z. Yang et al. “A Comprehensive Survey of Graph-level Learning”, *Journal of Latex Class Files*, vol. 14, 2021.

[32] C. Janiesch, P. Zschech, K. Heinrich “Machine Learning and Deep Learning”, Springer, 2021.

[33] S. Rathi “Approaches to Artificial General Intelligence: An Analysis”, *ArXiv*, 2022.

[34] Multiple Authors, “IntechOpen Series Artificial Intelligence”, Volume 7 “Machine learning Algorithms, Models Applications”, 2021.

[35] H. Khandelwal “A Closer Look into the Major Types of Machine Learning Models”, becominghuman.ai, 2019.

[36] Unknown Author “What is supervised Learning”, ibm.com, 2022.

[37] Unknown Author “What is unsupervised Learning”, ibm.com, 2022.

[38] SAMTHOMASONLINE “Challenges in Machine Learning Data Annotation & How to Overcome Them”, datasciencesociety.net, 2022.

[39] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, “Feature Selection: A Data Perspective”, *ACM Computer Surveys (CSUR),* 2017.

[40] S. Ontanon “An Overview of Distance and Similarity Functions for Structured Data”, *Artificial Intelligence Review* 53, 2020.

[41] K. Wong “A Short Survey on Data Clustering Algorithms”, *ArXiv*, 2015.

[42] K. Wongsuphasawat, Y. Liu, J. Heer, “Goals, Process, and Challenges of Exploratory Data Analysis: An Interview Study”, *Wiley Interdisciplinary Reviews: Computational Statistics 1*, 2009.

[43] V. Bajaj “Unsupervised Learning For Anomaly Detection”, towardsdatascience.com, 2020.

[44] C.O.S. Sorzano, J. Vargas, A. Pascual-Montano “A survey of dimensionality reduction techniques”, *ArXiv*, unknown.

[45] S. Xu, J. Vaughan, J. Chen, A. Sudjianto, V. Nair, “Supervised Linear Dimension-Reduction Methods: Review, Extensions, and Comparisons”, *ArXiv*, 2021.

[46] M. Hornick, “Using SVD for dimensionality reduction”, blogs.oracle.com, 2016.

[47] U. Michelucci “An introduction to Autoencoders”, *ArXiv*, 2022.

[48] L. Wiskott, F. Schonfeld “Laplacian Matrix for Dimensionality Reduction and Clustering”, *ArXiv*, 2019.

[49] W. Peng, X. BaoWen, W. YuRong, Z. XiaoYu “Link Prediction in Social Networks: the State-of-the-Art”, *Sci China Inf Sci*, 2015.

[50] Y. Zhu et al. “A Survey on Graph Structure Learning: Progress and Opportunities”, *DUP*, 2022.

[51] V. R. Joseph “Optimal Ratio for Data Splitting”, *ArXiv*, 2022.

[52] J. Brownlee, “What is semi supervised Learning”, machinelearningmastery.com, 2021.

[53] ODSC - Open Data Science, “A Brief Survey of Node Classification with Graph Neural Networks”, medium.com, 2020.

[54] M. Zhang, Y. Chen “Link Prediction Based on Graph Neural Networks”, *ArXiv*, 2018.

[55] D. Buterez, J. P. Janet, S. J. Kiddle, D. Oglic, P. Liò “Graph Neural Networks with Adaptive Readouts”, *ArXiv*, 2022.

[56] S. Fortunato “Community detection in graphs”, *ArXiv*, 2010.

[57] Aakanksha NS “Recommender Systems: Matrix Factorization from scratch”, towardsdatascience.com, 2020.

[58] L. Lovasz “Random Walks on Graphs a survey”, 1993.

[59] A. Grover, J. Leskovec “node2vec: Scalable Feature Learning for Networks”, Conference on Knowledge Discovery and Data Mining, 2016.

[60] G. Williams, “A summary of PageRank”, medium.com, 2021.

[61] P. Gupta “Regularization in Machine Learning”, towardsdatascience.com, 2017.

[62] K. Kubara “Introduction to message passing Neural Networks”, towardsdatascience.com, 2020

[63] [50] Y. Zhu et al. “A Survey on Graph Structure Learning: Progress and Opportunities”, 2022.

[64] W. Ruan, X. Yi, X. Huang, “Adversarial Robustness of Deep Learning: Theory, Algorithms, and Applications”, 30th ACM International Conference on Information and Knowledge Management (CIKM ’21), 2021.

[65] kaggle.com “Eliptic Dataset”.

[66] S. Minaee “20 Popular Machine Learning Metrics. Part 1: Classification & Regression Evaluation Metrics”, towardsdatascience.com, 2019.