Χαροκόπειο Πανεπιστήμιο
Τμήμα Πληροφορικής και Τηλεματικής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Αποσαφήνιση όρων με χρήση σημασιολογικής πληροφορίας»

ΣΟΥΛΙΩΤΗΣ ΝΙΚΟΛΑΟΣ

ΑΜ: 20734

Επιβλέπων:
Ηρακλής Βαρλάμης, Λέκτορας
Μέλη:
Θωμάς Καμαλάκης, Επίκουρος Καθηγητής
Κωνσταντίνος Τσερπές, Λέκτορας

Αθήνα, Σεπτέμβριος 2012

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν στην ολοκλήρωση της πτυχιακής εργασίας μου και γενικότερα των σπουδών μου στο Τμήμα Πληροφορικής και Τηλεματικής του Χαροκόπειου Πανεπιστημίου.

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της πτυχιακής μου εργασίας κ. Ηρακλή Βαρλάμη, για την καθοδήγησή του και την πολύτιμη συμβολή του σε κάθε φάση της δημιουργίας της.

Τέλος, θα ήθελα να ευχαριστήσω θερμά την οικογένειά μου για την ηθική και οικονομική συμπαράσταση όχι μόνο κατά τη διάρκεια της εκπόνησης της πτυχιακής μου εργασίας αλλά και καθ' όλη τη διάρκεια των σπουδών μου.

ΠΕΡΙΛΗΨΗ

Σκοπός της πτυχιακής εργασίας είναι η εφαρμογή αλγορίθμων επικάλυψης λέξεων (Lesk-based algorithms on word overlap) στην αποσαφήνιση της έννοιας πολύσημων λέξεων (Word Sense Disambiguation) με έμφαση στην ιατρική ορολογία.

Η έννοια της αποσαφήνισης λέξεων αφορά στην αντιμετώπιση της ασάφειας που μπορεί να εμφανιστεί στην ερμηνεία ενός όρου, ο οποίος συνδέεται στενά με δύο ή περισσότερα θέματα. Η ερμηνεία μπορεί να αλλάζει ανάλογα με το περιβάλλον στο οποίο βρίσκεται ο όρος (πρόταση, παράγραφος, κείμενο). Η ασάφεια στην έννοια μιας λέξης είναι χαρακτηριστικό της φυσικής γλώσσας. Για παράδειγμα η λέξη 'κρύο' έχει αρκετές έννοιες και μπορεί να αναφέρεται σε μια ασθένεια, στη θερμοκρασία ή σε μια περιβαλλοντική κατάσταση.

Η ασάφεια στην ερμηνεία ενός όρου δημιουργεί μεγάλες δυσκολίες σε πολλές εργασίες διαχείρισης κειμένων, π.χ. στην αναζήτηση ή στην κατηγοριοποίηση κειμένων. Η αποσαφήνιση γίνεται συνήθως με χρήση του περιβάλλοντος στο οποίο εμφανίζεται ο όρος, δηλαδή των λέξεων που εμφανίζονται στο ίδιο κείμενο, στην ίδια πρόταση, σε κοντινή απόσταση με τον όρο. Μια μεγάλη κατηγορία μεθόδων χρησιμοποιεί λεξικά και άλλους γλωσσολογικούς πόρους ώστε να εντοπίσει την καταλληλότερη έννοια για έναν όρο.

Η εργασία εντάσσεται σε αυτή την κατηγορία μεθόδων αποσαφήνισης καθώς χρησιμοποιεί γλωσσολογική γνώση και συγκεκριμένα τον ιατρικό θησαυρό UMLS (Unified Medical Language System) και στοχεύει στην αποσαφήνιση πολύσημων ιατρικών όρων.

Στο UMLS Metathesaurus που συνδυάζει πολλές βάσεις με ιατρικές ορολογίες, έγινε μια έρευνα που διερευνά την αυτόματη επίλυση της έννοιας μια λέξης χρησιμοποιώντας τεχνικές φυσικής γλώσσας. Πιο συγκεκριμένα, για την αποσαφήνιση μιας λέξης μέσα στο περιβάλλον της (παράγραφος) χρησιμοποιήθηκε η ομοιότητα του λήμματος (ορισμός) κάθε έννοιας της λέξης με την παράγραφο. Η έννοια με τη μεγαλύτερη ομοιότητα είναι η επικρατέστερη κάθε φορά. Η ομοιότητα

μετρήθηκε με ένα κλασικό αλγόριθμο που μετρά την επικάλυψη λέξεων (αλγόριθμος Lesk) μεταξύ δύο κειμένων.

Προκειμένου να υποστηριχθεί η έρευνα που έγινε και να αξιολογηθούν τα αποτελέσματά της, χρησιμοποιήθηκε μια συλλογή ιατρικών δοκιμαστικών κειμένων (περιλήψεις από την συλλογή επιστημονικών άρθρων Medline) στην οποία οι ασάφειες έχουν επιλυθεί εξαρχής με το χέρι. Η συλλογή αποτελείται από 50 πολύ συχνά διφορούμενες λέξεις. Κάθε μία από τις 50 λέξεις έχει 100 πιθανές έννοιες-περιπτώσεις.

Τα αποτελέσματα από αυτή τη μελέτη είναι η ακρίβεια των αποτελεσμάτων σε σύγκριση με τα αποτελέσματα που παίρνουμε ως δεδομένα από το UMLS. Το πρόγραμμα αυτόματης αποσαφήνισης επιλέγει το καλύτερο αποτέλεσμα και αυτό το συγκρίνουμε με το αποτέλεσμα που έχουν επιλέξει οι εκτιμητές του UMLS. Αν τα αποτελέσματα είναι τα ίδια τότε το λαμβάνουμε ως σωστό διαφορετικά ως λάθος. Η ακρίβεια είναι το πηλίκο των σωστών αποτελεσμάτων προς τις συνολικές περιπτώσεις.

ΠΕΡΙΕΧΟΜΕΝΑ

1 ΕΙΣΑΓΩΓΗ

1.1 ΒΙΟΪΑΤΡΙΚΟΙ ΟΡΟΙ ΚΑΙ ΑΠΟΣΑΦΗΝΙΣΗ

Οι μεγάλες ποσότητες των βιοϊατρικών κειμένων που υπάρχουν οδηγούν στην ανάγκη για αυτοματοποιημένες τεχνικές για ανάλυση και εξαγωγή των γνώσεων από αυτές τις πληροφορίες. Η λεκτική αμφισημία, φαινόμενο στο οποίο μια λέξη ή φράση έχει περισσότερες από μία δυνατή έννοια, παρουσιάζει ένα σημαντικό εμπόδιο για την αυτόματη επεξεργασία κειμένου. Η επίλυση της αμφισημίας των βιοϊατρικών όρων σε αυτά τα κείμενα είναι ένα σημαντικό βήμα για την ανάπτυξη αποτελεσματικών τεχνικών επεξεργασίας κειμένου. Η αποσαφήνιση των εννοιών των λέξεων (WSD) είναι μία τεχνολογία που επιλύει αυτές τις ασάφειες αυτόματα και είναι ένα σημαντικό βήμα στην κατανόηση του κειμένου. Οι πιο ακριβείς προσεγγίσεις για αποσαφήνιση λέξεων (WSD) βασίζονται σε παραδείγματα που

έχουν γίνει από ανθρώπους με το χέρι, αλλά αυτό είναι δύσκολο να γίνει αλλά και απαγορευτικά δαπανηρό. Η εργασία αυτή προσφέρει μια λύση σε αυτό το πρόβλημα χρησιμοποιώντας πληροφορίες από το UMLS Metathesaurus. Το Ενιαίο Σύστημα Ιατρικής Γλώσσας (Unified Medical Language System, UMLS) είναι μια συλλογή από πολλά λεξικά που ασχολούνται με τις βιοϊατρικές επιστήμες. Παρέχει μια δομή αντιστοίχισης μεταξύ αυτών των λεξιλογίων και έτσι επιτρέπει σε κάποιον να μεταφράσει τα διάφορα συστήματα ορολογίας. Επίσης, μπορεί να θεωρηθεί ως ένας ολοκληρωμένος λεξιλογικός θησαυρός των βιοϊατρικών εννοιών. Το UMLS παρέχει περαιτέρω διευκολύνσεις για την επεξεργασία της φυσικής γλώσσας. Προορίζεται για να χρησιμοποιηθεί κυρίως από προγραμματιστές των συστημάτων βιοϊατρικής πληροφορικής και αποτελείται από πηγές γνώσεις (βάσεις δεδομένων) και ένα σύνολο εργαλείων λογισμικού. Το UMLS σχεδιάστηκε το 1986 από τον Donald AB Lindberg[1] και συντηρείται από την Αμερικάνικη Εθνική Βιβλιοθήκη της Ιατρικής(US National Library of Medicine).

1.2 ΣΚΟΠΟΣ ΤΗΣ ΕΡΓΑΣΙΑΣ

Σκοπός της εργασίας είναι η κατανόηση του προβλήματος αποσαφήνισης λέξεων ή αλλιώς WSD, η επισκόπηση σχετικών αλγορίθμων που βασίζονται σε λεξικά ή σε βάσεις γνώσεις και η υλοποίηση ενός αλγορίθμου, που βασίζεται στον αλγόριθμο Lesk και έχει ως σκοπό την αυτόματη αποσαφήνιση. Η υλοποίηση θα δοκιμαστεί στην αποσαφήνιση ενός συνόλου βιοϊατρικών όρων. Για το σκοπό αυτό θα χρησιμοποιηθούν πληροφορίες από το UMLS. Τέλος, θα εκτιμηθεί η ακρίβεια της μελέτης.

1.3 ΟΦΕΛΗ ΑΠΟ ΤΗΝ ΕΡΓΑΣΙΑ

[1] Donald A.B. Lindberg, M.D. “Biographical Sketch” http://www.nlm.nih.gov/od/roster/lindberg.html

Τα οφέλη από την πτυχιακή μου εργασία είναι πολλά. Αρχικά, αποτελεί μια πρώτη επαφή με την έρευνα και αποδεικνύει τη γνώση σε μια επιστημονική περιοχή όπως η κατανόηση και η εφαρμογή μεθοδολογιών και τεχνικών στο κομμάτι της αποσαφήνισης. Ακόμη, αποτελεί σημαντικό εφόδιο για επαγγελματική απασχόληση ή μεταπτυχιακές σπουδές, όπως και σημαντική ενίσχυση του βιογραφικού μου. Η πτυχιακή εργασία μου έδωσε την ευκαιρία να δοκιμάσω τις γνώσεις που απέκτησα κατά τη διάρκεια των σπουδών μου αφού διερεύνησα ένα θέμα που με ενδιαφέρει, εφαρμόζοντας μια συστηματική μεθοδολογική και επιστημονική προσέγγιση. Ειδικότερα, σε επίπεδο υλοποίησης ασχολήθηκα με προγραμματισμό σε Java όπως και με Βάσεις δεδομένων (MySQL), δύο πολύ βασικά εργαλεία ανάπτυξης εφαρμογών και πληροφοριακών συστημάτων, και σε επίπεδο αλγορίθμων υλοποίησα μέτρα ομοιότητα κειμένων που βασίζονται στην επικάλυψη λέξεων και μέτρα σημαντικότητας (βαρύτητας) λέξεων που βασίζονται στη συχνότητα εμφάνισης, τεχνικές που είναι πολύ χρήσιμες σε οποιονδήποτε αναπτύσσει εφαρμογές που σχετίζονται με διαχείριση και αναζήτηση κειμένων.

2 ΥΠΟΒΑΘΡΟ

2.1 ΓΕΝΙΚΑ

Στην υπολογιστική γλωσσολογία, η αποσαφήνιση της έννοιας των λέξεων (Word sense disambiguation - WSD) είναι ένα πρόβλημα της επεξεργασίας φυσικής γλώσσας, που προσπαθεί να αναγνωρίσει το νόημα-έννοια μιας λέξης που χρησιμοποιείται σε μία πρόταση, όταν η λέξη έχει πολλαπλές σημασίες. Πολλές έρευνες έχουν γίνει και έχουν προχωρήσει σταθερά προς το σημείο που τα συστήματα αποσαφήνισης έχουν επιτύχει ένα επαρκώς υψηλό επίπεδο ακρίβειας σε λέξεις με πολλαπλές σημασίες.

Η διαδικασία αποσαφήνισης απαιτεί δύο πράγματα: Ένα λεξικό για να καθορίσει τις έννοιες κάθε λέξης που πρέπει να αποσαφηνιστεί και μια συλλογή λέξεων οι οποίες θα αποσαφηνιστούν.

2.1.1 ΙΣΤΟΡΙΑ

Η αποσαφήνιση λέξεων διατυπώθηκε για πρώτη φορά ως ξεχωριστή υπολογιστική εργασία κατά τις πρώτες ημέρες της αυτόματης μετάφρασης στη δεκαετία του 1940, καθιστώντας το ένα από τα παλαιότερα προβλήματα στην υπολογιστική γλωσσολογία. Ο Warren Weaver (1949) έθεσε για πρώτη φορά το πρόβλημα σε ένα υπολογιστικό πλαίσιο. Αρχικά είχε αναφέρει για πρώτη φορά το ενδεχόμενο της χρήσης ηλεκτρονικών υπολογιστών για τη μετάφραση εγγράφων σε επιστολή του προς την κυβέρνηση. Στα επόμενα δύο χρόνια προέτρεψε τους συναδέλφους του από το ίδρυμα Rockefeller να επεξεργαστούν τις ιδέες του. Το αποτέλεσμα ήταν μία έκθεση, με τίτλο "Translation"(“Μετάφραση”), το οποίο έγραψε τον Ιούλιο του 1949 στο Carlsbad, New Mexico. Οι πρώτοι ερευνητές κατανόησαν τη σημασία και τη δυσκολία του WSD. Ειδικότερα, ο Bar-Hillel (1960) υποστήριξε ότι η αποσαφήνιση (WSD) δεν θα μπορούσε να γίνει με ηλεκτρονικό υπολογιστή λόγω της ανάγκης σε γενικές γραμμές να διαμορφώσει όλες τις γνώσεις για τον κόσμο. Στη δεκαετία του 1970, το WSD ήταν υποεργασία των συστημάτων σημασιολογικής ερμηνείας που αναπτύσσονταν στο πεδίο της τεχνητής νοημοσύνης, αλλά από τότε που τα συστήματα αποσαφήνισης ήταν σε μεγάλο βαθμό με βάση τους κανόνες και κωδικοποιημένα στο χέρι βοήθησαν στην απόκτηση γνώσεων. Από το 1980 οι μεγάλοι κλίμακας λεξικογραφικοί πόροι, όπως το Λεξικό της Οξφόρδης (Oxford Advanced Learners’ Dictionary - OALD), έγιναν διαθέσιμοι. Έτσι, η κωδικοποίηση με το χέρι αντικαταστάθηκε αυτόματα με τη γνώση που προέρχεται από αυτους τους πόρους, αλλά η αποσαφήνιση ήταν ακόμα βασισμένη είτε στην γνώση των ειδικών γλωσσολόγων ή βασισμένη στο λεξικό. Στη δεκαετία του 1990, η στατιστική επανάσταση σάρωσε την υπολογιστική γλωσσολογία και η αποσαφήνιση (WSD) έγινε παράδειγμα προβλήματος στο οποίο εφαρμόστηκε επίβλεψη τεχνικών μηχανικής μάθησης. Τέλος τη δεκαετία του 2000, οι τεχνικές επίβλεψης έχουν φτάσει σε ένα καλό επίπεδο όσον αφορά την ακρίβεια. Τα εποπτευόμενα συστήματα συνεχίζουν να αποδίδουν καλύτερα.

2.1.2 ΔΥΣΚΟΛΙΕΣ

2.1.2.1 ΔΙΑΦΟΡΕΣ ΜΕΤΑΞΥ ΤΩΝ ΛΕΞΙΚΩΝ

Ένα πρόβλημα αποσαφήνισης της έννοιας των λέξεων είναι να αποφασιστεί ποια από τις διαφορετικές έννοιες που μπορεί να έχει η λέξη ταιριάζει καλύτερα στη εμφάνιση της λέξης μέσα σε ένα συγκεκριμένο περιεχόμενο (context) π.χ. πρόταση, παράγραφο, κείμενο. Σε μερικές περιπτώσεις οι διαφορετικές έννοιες μπορεί να είναι στενά συνδεδεμένες μεταξύ τους (μία έννοια να είναι μεταφορική ή μετωνυμική επέκταση της άλλης), και σε τέτοιες περιπτώσεις η αντιστοίχιση των λέξεων στις σωστές έννοιες γίνεται πολύ πιο δύσκολη. Διαφορετικά λεξικά και θησαυροί παρέχουν διαφορετικά τμήματα των λέξεων στις έννοιες και δίνουν διαφορετικά σύνολα εννοιών για μια λέξη. Μια λύση που ορισμένοι ερευνητές έχουν χρησιμοποιήσει είναι η επιλογή ενός συγκεκριμένου λεξικού και η χρησιμοποίηση μόνο των δικών του εννοιών. Γενικά, όμως τα αποτελέσματα των ερευνών που χρησιμοποιούσαν μεγάλο όγκο πληροφοριών στις έννοιες ήταν πολύ καλύτερα από εκείνους που χρησιμοποιούσαν μικρό όγκο πληροφοριών. Οι περισσότερες έρευνες στον τομέα της αποσαφήνισης λέξεων γίνεται με τη χρήση του WordNet ως κατάλογου εννοιών στα αγγλικά. Το WordNet είναι μία λεξιλογική βάση δεδομένων για την αγγλική γλώσσα. Ομαδοποιεί αγγλικές λέξεις σε ομάδες συνωνύμων που ονομάζονται synsets. Ακόμη, παρέχει σύντομους και γενικούς ορισμούς και καταγράφει τις διάφορες σημασιολογικές σχέσεις μεταξύ των ομάδων συνωνύμων. Ο πρώτος του στόχος του είναι να παράγει ένα συνδιασμό εννοιών από λεξικά και λεξιλογικούς θυσαυρούς ώστε να μπορεί να χρησιμοποιηθεί περισσότερο διαισθητικά και ο δεύτερος να μπορεί να υποστηρίζει τεχνικές αυτόματης ανάλυσης κειμένου αλλά και εφαρμογές τεχνητής νοημοσύνης. Η βάση δεδομένων και τα εργαλεία λογισμικού έχουν δοθεί στη δημοσιότητα (σύμφωνα με μία άδεια BSD) και είναι διαθέσιμα ώστε να χρησιμοποιηθούν. Το WordNet δημιουργήθηκε και συντηρείται στο Εργαστήριο Γνωσιακής Επιστήμης του Πανεπιστημίου του Princetonυπό την καθοδήγηση του καθηγητή ψυχολογίας George A. Miller. Η ανάπτυξή του άρχισε το 1985. Άλλοι πόροι που χρησιμοποιούνται για αποσαφήνιση είναι οι Roget's

Thesaurus[2] και η Wikipedia[3].

2.1.2.2 ΜΕΡΗ ΤΟΥ ΛΟΓΟΥ

Σε κάθε πραγματικό τεστ, το μέρος του λόγου και η έννοια είναι πολύ στενά συνδεδεμένα μεταξύ τους περιορίζοντας το ένα το άλλο. Το ερώτημα αν αυτά τα καθήκοντα θα πρέπει να διατηρούνται ή να αποσυνδεθούν μεταξύ τους δεν έχει αποφασιστεί ακόμα ομόφωνα, αλλά πρόσφατα οι επιστήμονες τείνουν να δοκιμάσουν αυτά τα πράγματα χωριστά. Είναι διδακτικό να συγκρίνουμε το πρόβλημα αποσαφήνιση της έννοιας μιας λέξης με το πρόβλημα της σήμανσης του μέρους του λόγου (Part Of Speech tagging). Και οι δυο περιλαμβάνουν αποσαφηνισμένες έννοιες ή σημάνσεις με λέξεις, είτε με τις έννοιες είτε με τα μέρη του λόγου. Ωστόσο, οι αλγόριθμοι χρησιμοποιούνται για κάποιο πρόβλημα που δεν έχει την τάση να λειτουργεί καλά για το άλλο, κυρίως γιατί το μέρος του λόγου μιας λέξης καθορίζεται κυρίως από τις αμέσως γειτονικές λέξεις, ενώ η έννοια της λέξης μπορεί να καθορίζεται από λέξεις που βρίσκονται πιο μακριά. Το ποσοστό επιτυχίας για έναν αλγόριθμο σήμανσης ενός μέρους του λόγου είναι σήμερα πολύ υψηλότερο από εκείνο ενός WSD. Βέβαια αυτά τα στοιχεία είναι χαρακτηριστικά για την αγγλική γλώσσα και μπορεί να διαφέρουν για τις άλλες γλώσσες.

2.1.2.3 ΚΟΙΝΗ ΛΟΓΙΚΗ

Μερικοί ερευνητές στο πεδίο της τεχνητής νοημοσύνης, όπως ο Douglas Lenat υποστηρίζουν ότι δεν μπορεί κανείς να αναλύσει έννοιες από τα λόγια, χωρίς κάποια κοινή έννοια οντολογίας. Για παράδειγμα, συγκρίνοντας αυτές τις δύο φράσεις:

"Jill and Mary are sisters." – (Είναι μεταξύ τους αδερφές).

"Jill and Mary are mothers." – (Είναι μητέρες ξεχωριστά η μία από την άλλη).

[2] Yarowsky 1992, pp. 454–460.

[3] Εγκυκλοπαίδεια Wikipedia. www.wikipedia.org

Για να προσδιορίσει σωστά τις έννοιες των λέξεων κάποιος πρέπει να γνωρίζει γεγονότα κοινής λογικής. Επιπλέον, μερικές φορές η κοινή λογική είναι απαραίτητη για την αποσαφήνιση αυτών των λέξεων.

2.1.3 ΠΡΟΣΕΓΓΙΣΕΙΣ ΚΑΙ ΜΕΘΟΔΟΙ

Όπως σε όλες τις επεξεργασίες φυσικών γλωσσών, υπάρχουν δυο κύριες προσεγγίσεις για την αποσαφήνιση: η βαθιά και η ρηχή προσέγγιση. Η βαθιά προσέγγιση βασίζεται στη γνώση που χρησιμοποιείται για να καθορίσει με ποια έννοια χρησιμοποιείται η λέξη. Αυτή η προσέγγιση δεν είναι πολύ επιτυχημένη στην πράξη, κυρίως επειδή τέτοια γνώση δεν υπάρχει σε μορφή αναγνώσιμη από τον υπολογιστή εκτός από πολύ περιορισμένους τομείς. Ωστόσο αν υπήρχε τέτοιου είδους γνώση, τότε η βαθιά προσέγγιση θα ήταν πολύ πιο ακριβής από τη ρηχή προσέγγιση. Επίσης, υπάρχει μια μακρά παράδοση σε θέματα υπολογιστικής γλωσσολογίας, στην προσπάθεια αυτών των προσεγγίσεων όσον αφορά την κωδικοποίηση της γνώσης. Σε ορισμένες περιπτώσεις είναι δύσκολο να πούμε με σαφήνεια κατά πόσο η γνώση που εμπλέκεται είναι γλωσσική ή παγκόσμια γνώση.

Η ρηχή προσέγγιση δεν προσπαθεί να κατανοήσει το κείμενο. Απλά εξετάζουν τις λέξεις που βρίσκονται στην πρόταση. Οι κανόνες που χρησιμοποιεί μπορούν να προέρχονται αυτόματα από τον υπολογιστή, χρησιμοποιώντας ένα σύνολο κατάρτισης των λέξεων που έχουν ως ετικέτα τις έννοιες της κάθε λέξης. Αυτή η προσέγγιση ενώ θεωρητικά δεν είναι τόσο ισχυρή όσο η βαθία προσέγγιση, δίνει εξαιρετικά αποτελέσματα στην πράξη, λόγω των περιορισμένων γνώσεων του υπολογιστή.

Υπάρχουν τέσσερις συμβατικές προσεγγίσεις για την αποσαφήνιση λέξεων :

– Οι μέθοδοι με βάση τα λεξικά και τις γνώσεις που βασίζονται κυρίως σε λεξικά, λεξιλογικούς θησαυρούς και λεξιλογικές βάσεις δεδομένων.

– Οι μέθοδοι επίβλεψης, που χρησιμοποιούν θησαυρούς με έννοιες που έχουν σχόλια.

– Οι μέθοδοι ημι-επίβλεψης ή ελάχιστα επιβλεπόμενες μέθοδοι, που

χρησιμοποιούν δευτερεύουσες πηγές γνώσης.

– Οι μέθοδοι χωρίς επίβλεψη, οι οποίες αποφεύγουν σχεδόν ολοκληρωτικά εξωτερικές πληροφορίες. Αυτές οι μέθοδοι είναι γνωστοί ακόμη και με το όνομα διάκρισης των εννοιών των λέξεων (word sense discrimination).

2.1.3.1 Μέθοδοι με βάση το λεξικό και μέθοδοι με βάση τη γνώση

Ο αλγόριθμος Lesk είναι μια δημιουργική μέθοδος με βάση το λεξικό. Βασίζεται στην υπόθεση ότι οι λέξεις που χρησιμοποιούνται μαζί με το κείμενο σχετίζονται μεταξύ τους και η σχέση μπορεί να παρατηρηθεί στους ορισμούς των λέξεων όπως και στις έννοιες τους. Δύο (ή περισσότερες) λέξεις αποσαφηνίζονται βρίσκοντας το ζεύγος των εννοιών τους στο λεξικό με τη μεγαλύτερη επικάλυψη της λέξης στο λεξικό με τους ορισμούς. Μια εναλλακτική λύση για τη χρήση των ορισμών είναι να εξεταστεί η συγγένεια της έννοιας της λέξης και να υπολογίσει τη σημασιολογική ομοιότητα του κάθε ζεύγους των εννοιών των λέξεων που βασίζεται σε μια συγκεκριμένη λεκτική βάση γνώσεων όπως το WordNet. Οι μέθοδοι με βάση τα γραφήματα εξαπλώθηκαν με την ενεργοποίηση των ερευνών τεχνητής νοημοσύνης (Artificial intelligence AI) και είχαν εφαρμοστεί με κάποια επιτυχία. Οι πιο σύνθετες μέθοδοι γραφήματος με βάση τις προσεγγίσεις έχουν αποδειχθεί οτι εκτελούνται σχεδόν τόσο καλά όσο και οι μέθοδοι εποπτείας ή ακόμη και ξεπερνώντας τους σε συγκεκριμένους τομείς. Πρόσφατα, έχει αναφερθεί ότι η αυτόματη μεταφορά γνώσεων με τη μορφή της σημασιολογικής σχέσης από την Wikipedia στο WordNet ενίσχυει τις μεθόδους που βασίζονται στην απλή γνώση, επιτρέποντάς τους να ανταγωνιστούν τα καλύτερα εποπτευόμενα συστήματα ακόμη και να τα ξεπεράσουν σε συγκεκριμένους τομείς ρύθμισης.

2.1.3.2 Μέθοδοι επίβλεψης

Οι μέθοδοι επίβλεψης βασίζονται στην προϋπόθεση ότι το κείμενο μπορεί να παρέχει αρκετά στοιχεία σχετικά με αυτό (κείμενο) για την αποσαφήνιση των λέξεων (συνεπώς η γνώση και η λογική θεωρούνται περιττά). Πιθανώς, κάθε μηχανή εκμάθησης αλγορίθμου πρόκειται να εφαρμοστεί για την αποσαφήνιση (WSD),

συμπεριλαμβανομένων των συνδεδεμένων τεχνικών, όπως η επιλογή των χαρακτηριστικών γνωρισμάτων , η βελτιστοποίηση των παραμέτρων και η συνολική μάθηση. Οι μηχανές εδραίων διανυσμάτων υποστήριξης (Support Vector Machines) και η μάθηση που είναι βασισμένη στη μνήμη (memory-based learning) έχουν αποδειχθεί ότι είναι οι πιο επιτυχημένες προσεγγίσεις μέχρι σήμερα, ίσως επειδή μπορούν να αντιμετωπίσουν την υψηλή διστακτικότητα του χώρου των χαρακτηριστικών γνωρισμάτων. Ωστόσο, οι μέθοδοι επίβλεψης υπόκεινται σε ένα νεο εμπόδιο απόκτησης γνώσεων εφόσον βασίζονται σε μεγάλο βαθμό στην επιλογή της κατάλληλης έννοιας χειροκίνητα για την κατάρτιση, η οποία είναι επίμονη και δαπανηρή.

2.1.3.3 Μέθοδοι ημιεπίβλεψης ή ελάχιστα επιβλέψιμοι μέθοδοι

Λόγω της έλλειψης δεδομένων εκπαίδευσης, πολλοί αλγόριθμοι αποσαφήνισης λέξεων χρησιμοποιούν ημι-επιβλεπόμενη μάθηση, η οποία επιτρέπει και χαρακτηρισμένα αλλά και μη χαρακτηρισμένα δεδομένα. Ο αλγόριθμος Yarowsky ένα από τα πρώτα παραδείγματα ενός τέτοιου αλγορίθμου. Χρησιμοποιεί το "Μια έννοια ανα συνδυασμό λέξεων" (One sense per collocation) και το "Μια έννοια ανα περιεχόμενο" (One sense per discourse) που είναι ιδιότητες των ανθρώπινων γλωσσών για την αποσαφήνιση της έννοιας των λέξεων. Επιπλέον, οι λέξεις έχουν την τάση να επιδεικνύουν μόνο μια έννοια τις περισσότερες φορές δεδομένου του λόγου και ενός συνδυασμού λέξεων. Η αρχική προσέγγιση ξεκινά με ένα μικρό αριθμό δεδομένων για κάθε λέξη. Τα αρχικά δεδομένα χρησιμοποιούνται για την κατάρτιση μιας αρχικής ταξινόμησης, χρησιμοποιώντας οποιαδήποτε μέθοδο επίβλεψης. Αυτή η ταξινόμηση χρησιμοποιείται στη συνέχεια σε τμήματα με μη τοποθετημένες ετικέτες για την εξαγωγή ενός μεγαλυτέρου συνόλου εκπαίδευσης, στο οποίο περιλαμβάνονται οι πιο σίγουρες ταξινομήσεις. Η διαδικασία επαναλαμβάνεται και κάθε νέος ταξινομητής εκπαιδεύεται σε διαδοχικά μεγαλύτερο διάστημα εκπαίδευσης έως ότου να επιτευχθεί ο μέγιστος αριθμός επαναλήψεων ή να ταξινομηθεί ολόκληρο το διάστημα εκπαίδευσης. Άλλες ημι-εποπτευόμενες τεχνικές χρησιμοποιούν μεγάλες ποσότητες διαστημάτων χωρίς ετικέτες για να παρέχουν πληροφορίες που συμπληρώνουν τα διαστήματα με ετικέτα. Αυτές οι τεχνικές έχουν τη δυνατότητα να βοηθήσουν στην προσαρμογή των εποπτευόμενων μοντέλων σε

διαφορετικούς τομείς. Τέλος, μια διφορούμενη λέξη σε μια γλώσσα μεταφράζεται συχνα σε διαφορετικές λέξεις σε μία άλλη γλώσσα, ανάλογα με την έννοια της λέξης.

2.1.3.4 Μέθοδοι χωρίς επίβλεψη

Η μη επιβλεπόμενη μάθηση είναι η μεγαλύτερη πρόκληση για τους ερευνητές της αποσαφήνισης λέξεων (WSD). Η βασική παραδοχή είναι ότι παρόμοιες έννοιες εμφανίζονται σε παρόμοια πλαίσια, και συνεπώς οι έννοιες μπορούν να προκληθούν από το κείμενο με ομαδοποίηση λέξεων χρησιμοποιώντας κάποιο μέτρο ομοιότητας του πλαισίου, ένα έργο που αναφέρεται ως επαγωγή της έννοιας των λέξεων ή των διαφορών που υπάρχουν στις λέξεις ανάλογα τη γλώσσα. Στη συνέχεια, οι νέες εμφανίσεις της λέξης μπορούν να ταξινομηθούν σε κοντινότερες συστάδες / έννοιες. Η απόδοση είναι χαμηλότερη από τις παραπάνω μεθόδους (Επιβλεπόμενες μέθοδοι, Ημι-επιβλεπόμενες ή ελάχιστα επιβλεπόμενες μέθοδοι) αλλά οι συγκρίσεις είναι δύσκολες δεδομένου ότι οι έννοιες που προκαλούνται πρέπει να αντιστοιχίζονται σε ένα γνωστό λεξικό των εννοιών των λέξεων. Εάν μια χαρτογράφηση από ένα σύνολο εννοιών ενός λεξικού δεν είναι επιθυμητή, μπορούν να εκτελεστούν αξιολογήσεις με βάση τη διασπορά (συμπεριλαμβανομένων των μέτρων της εντροπίας και της καθαρότητας). Εναλλακτικά, οι μέθοδοι επαγωγής εννοιών λέξεων μπορούν να εξεταστούν και να συγκριθούν μέσα σε ένα πρόγραμμα. Για παράδειγμα, έχει αποδειχθεί ότι η επαγωγή της έννοιας μιας λέξης βελτιώνει την ομαδοποίηση στα αποτελέσματα αναζήτησης στο διαδίκτυο αυξάνοντας την ποιότητα των αποτελεσμάτων και του βαθμού της διαφοροποίησης στις λίστες αποτελεσμάτων. Εκφράζεται η ελπίδα ότι η μη επιβλεπόμενη μάθηση θα ξεπεράσει το εμπόδιο της απόκτησης γνώσεων επειδή δεν εξαρτάται από την προσπάθεια που γίνεται με το χέρι.

2.1.4 ΑΞΙΟΛΟΓΗΣΗ

Η σύγκριση και η αξιολόγηση των διαφορών των συστημάτων αποσαφήνισης

λέξεων (WSD) είναι εξαιρετικά δύσκολη, λόγω των διαφορετικών συνόλων δοκιμής, τους καταλόγους εννοιών και τους πόρους των γνώσεων. Πριν από την οργάνωση των ειδικών εκστρατειών αξιολόγησης τα περισσότερα συστήματα είχαν αξιολογηθεί σε σύνολα δεδομένων μικρής κλίμακας. Προκειμένου να ελέγξουν έναν αλγόριθμο, οι προγραμματιστές πρέπει να περνούν το χρόνο τους στο σχολιασμό όλων των λέξεων που εμφανίζονται. Επίσης, δεν επιλέγουν μεθόδους σύγκρισης ακόμη και στο ίδιο διάστημα εφόσον χρησιμοποιούν διαφορετικούς καταλόγους εννοιών. Προκειμένου να καθοριστούν κοινές βάσεις δεδομένων και διαδικασίες αξιολόγησης, οι δημόσιες εκστρατείες αξιολόγησης έχουν οργανωθεί. Το Senseval (από το 2007 και έπειτα μετονομάστηκε σε SemEval) είναι ένας διεθνής διαγωνισμός αποσαφήνισης εννοιών των λέξεων, που πραγματοποιείται κάθε τρία χρόνια από το 1998. Ο στόχος του διαγωνισμού είναι να προετοιμάσει σύνολα δεδομένων για την αξιολόγηση αλγορίθμων αποσαφήνισης λέξεων (WSD) και να οργανώσει συγκριτικές αξιολογήσεις των συστημάτων αποσαφήνισης σε διάφορα είδη καθηκόντων, συμπεριλαμβανομένων όλων των λέξεων και των λεξιλογικών δειγμάτων αποσαφήνισης για διαφορετικές γλώσσες. Τα συστήματα που υποβάλλονται σε αξιολόγηση σε αυτούς τους διαγωνισμούς συνήθως ενσωματώνουν διαφορετικές τεχνικές και συχνά συνδυάζουν την εποπτεία και τις μεθόδους που βασίζονται στη γνώση (ιδίως για την αποφυγή κακών επιδόσεων σε παραδείγματα εκπαίδευσης).

2.1.5 ΑΞΙΟΛΟΓΗΣΗ ΚΑΙ ΕΠΙΛΟΓΕΣ ΣΧΕΔΙΑΣΜΟΥ ΕΡΓΑΣΙΩΝ

Αρχικά δημιουργούνται οι κατάλογοι με τις διαθέσιμες έννοιες, στη συνέχεια ορίζονται οι πολύσημες έννοιες που πρέπει να αποσαφηνιστούν, έπειτα εντοπίζονται τα αποσπάσματα κειμένων στα οποία έχουμε εμφανίσεις των λέξεων που πρόκειται να αποσαφηνίσουμε και τέλος ορίζονται τα μέτρα αξιολόγησης.

Κατάλογοι εννοιών. Σε μία από τις πρώτες ασκήσεις αποσαφήνισης συνόλων για WSD, στο 1^{ο} Senseval χρησιμοποιήθηκε ο κατάλογος εννοιών HECTOR. Ο λόγος της έγκρισης ενός άγνωστου ως τότε καταλόγου εννοιών ήταν κυρίως να αποφευχθεί η χρήση των δημοφιλών εννοιών των λέξεων (όπως το WordNet), οι οποίες θα

μπορούσαν να κάνουν τα πειράματα αθέμιτα ή μεροληπτικά. Ωστόσο, δεδομένης της έλλειψης κάλυψης τέτοιων καταλόγων, ο κατάλογος εννοιών WordNet εγκρίθηκε στο 2^{ο} Senseval.

Ένα σύνολο από δοκιμαστικές λέξεις. Η σύγκριση των μεθόδων μπορεί να χωριστεί σε 2 ομάδες με βάση τον αριθμό των λέξεων που πρέπει να αποσαφηνιστούν. Η διαφορά αυτή έχει αντίκτυπο στην ποσότητα της ανάλυσης και της επεξεργασίας που πρέπει να γίνει.

– Το έργο (task) προϋποθέτει την αποσαφήνιση όλων των λέξεων του κειμένου.

– Το έργο αφορά την αποσαφήνιση επιλεγμένων λέξεων από το σύνολο των λέξεων του κειμένου, οι οποίες παρόλα αυτά μπορούν να χρησιμοποιηθούν για την αποσαφήνιση.

Υποτίθεται ότι στην πρώτη ομάδα έχουμε μια πιο ρεαλιστική αξιολόγηση, παρόλο τον πολύ κοπιαστικό έλεγχο των αποτελεσμάτων. Αρχικά, μόνο η τελευταία ομάδα είχε χρησιμοποιηθεί για την αξιολόγηση αλλά αργότερα συμπεριλήφθηκε και η πρώτη ομάδα.

Αφετηρίες αναφοράς. Για λόγους σύγκρισης χρησιμοποιούνται αλγόριθμοι που ονομάζονται αφετηρίες αναφοράς (Baselines). Αυτές περιλαμβάνουν διάφορες παραλλαγές του αλγορίθμου Lesk ή των πιο συχνών αλγόριθμων εννοιών.

Κατάλογοι εννοιών. Οι ασκήσεις αποσαφήνισης απαιτούν ένα λεξικό για να προσδιοριστούν οι έννοιες των λέξεων που πρόκειται να αποσαφηνιστούν και ένα κείμενο το οποίο θα αποσαφηνιστεί. Το WordNet είναι το πιο δημοφιλές παράδειγμα των αποθεμάτων που περιέχουν έννοιες λέξεων.

Αξιολόγηση των μέτρων. Κατά τη διάρκεια της αξιολόγησης των συστημάτων αποσαφήνισης χρησιμοποιούνται δύο βασικά μέτρα απόδοσης:

– Η ακρίβεια (precision), το ποσοστό των εργασιών του συστήματος που πραγματοποιήθηκαν δείχνει πόσο σωστό είναι.

– Η ανάκληση (recall), το ποσοστό των συνολικών περιπτώσεων που η έννοια της λέξης έχει ανατεθεί σωστά από το σύστημα.

Αν ένα σύστημα κάνει μια ανάθεση για κάθε λέξη (100% κάλυψη – coverage), τότε η ακρίβεια και η ανάκληση γίνονται το ίδιο που μπορεί να ονομαστεί ακρίβεια (accuracy). Αυτό το μοντέλο έχει επεκταθεί για να λαμβάνονται υπόψη τα συστήματα που επιστρέφουν ένα σύνολο εννοιών με βάρη για κάθε περιστατικό.

2.2 WSD ΜΕ ΒΙΟΪΑΤΡΙΚΑ ΔΕΔΟΜΕΝΑ

Η βιοϊατρική επιστημονική βιβλιογραφία είναι πλέον τόσο μεγάλη, ώστε αυτοματοποιημένα εργαλεία είναι αναγκαία για την αποτελεσματική κατανόηση των κειμένων (Chapman και Cohen,2009). Ωστόσο, η διαδικασία αυτή γίνεται δύσκολη από το γεγονός ότι οι όροι σε φυσική γλώσσα μπορεί να είναι διφορούμενοι, δηλαδή μπορεί να αναφέρονται σε περισσότερες από μία πιθανές έννοιες. Για παράδειγμα, στον τομέα της βιοϊατρικής η λέξη «κρύο» είναι διφορούμενη και μπορεί να σημαίνει (τουλάχιστον) «χαμηλή θερμοκρασία» ή «κρύα αίσθηση». Τα συστήματα της αποσαφήνισης των εννοιών των λέξεων (WSD) έχουν ως στόχο να λύσουν αυτό το πρόβλημα με τον προσδιορισμό των εννοιών από τις διφορούμενες λέξεις (Agirre και Edmonds 2006, Navigli, 2009). Αυτές οι πληροφορίες θα μπορούσαν να χρησιμοποιηθούν για τη βελτίωση αναζητήσεων στη λογοτεχνία με την εξασφάλιση ότι στη συνέχεια τα έγγραφα που επιστρέφονται περιέχουν μόνο διφορούμενους όρους όταν πρόκειται να χρησιμοποιηθούν σε μια έννοια που είναι σχετική με την έρευνα, και είναι επίσης ωφέλιμη για άλλες εφαρμογές που είναι χρήσιμες για βιοϊατρικές έρευνες, όπως η αυτοματοποιημένη ευρετηρίαση, η άντληση πληροφοριών και γνώσεων (Aronson et al., 2000; Friedman, 2000; MacMullen and Denn, 2005). Σε πρόσφατο άρθρο του ο Agirre (Agirre, et al.,2010) περιγράφει μια προσέγγιση για την αποσαφήνιση λέξεων στον βιοϊατρικό τομέα που βασίζεται σε αλγορίθμους με βάση τα γραφήματα. Δεδομένου ότι η προσέγγιση είναι χωρίς επίβλεψη, δεν απαιτεί οποιαδήποτε επισήμανση κατάρτισης δεδομένων και βασίζεται σε πληροφορίες από το Ενιαίο Σύστημα ιατρικών Γλωσσών (UMLS) Metathesaurus (Humphreys et al., 1998).

2.2.1 ΣΧΕΤΙΚΗ ΕΡΓΑΣΙΑ

Το πρόβλημα της WSD έχει διερευνηθεί από το 1950 και θεωρείται ως ένα σημαντικό στάδιο στην επεξεργασία κειμένου (Agirre και Edmonds, 2006, Navigli, 2009). Η πλειοψηφία των προσεγγίσεων έχουν εξετάσει το πρόβλημα σε ένα περιβάλλον ανεξάρτητα από τον τομέα, αν και αρκετοί ερευνητές έχουν αναπτύξει συστήματα που προορίζονται ειδικά για την επίλυση των ασαφειών που υπάρχουν στον τομέα της βιοϊατρικής (Schuemie, 2005). Οι πιο δημοφιλείς προσεγγίσεις για την αποσαφήνιση (WSD) στην βιοϊατρική βασίζονται στην εποπτευόμενη μάθηση, όπως του Joshi (2005), του Liu (2004) και του Savova (2008). Παρότι οι μελέτες σχετικά με τους ανεξάρτητους τομείς αποσαφήνισης έχουν δείξει ότι οι εποπτευόμενες προσεγγίσεις ξεπερνούν τις εναλλακτικές, που απαιτούν παραδείγματα εκπαίδευσης που δεν μπορεί να είναι διαθέσιμα και είναι δαπανηρά να δημιουργηθούν. Αυτός ο περιορισμός σημαίνει ότι οι περισσότερες εποπτευόμενες προσεγγίσεις (συμπεριλαμβανομένων εκείνων που αναφέρθηκαν παραπάνω) μπορούν να αποσαφηνίσουν μόνο ένα μικρό δείγμα των λέξεων για τις οποίες τα δεδομένα κατάρτισης μπορούν να βρεθούν, και αυτό περιορίζει τη χρησιμότητα τους στην πράξη. Στο πλαίσιο της βιοϊατρικής, ο Humphrey το 2006 αποφεύγει αυτό το πρόβλημα χρησιμοποιώντας το Medline ως δεδομένα εκπαίδευσης και αξιοποιώντας τις πληροφορίες που περιέχεισχετικά με την πηγή κάθε περίληψης(abstract). Οι μη επιβλεπόμενες προσεγγίσεις δεν απαιτούν παραδείγματα με ετικέτες εκπαίδευσης και συχνά χρησιμοποιούν βάσεις με γνώσεις, όπως το UMLS Metathesaurus. Ο McInnes (2008) περιγράφει μια τέτοια προσέγγιση που χρησιμοποιεί το UMLS για τη δημιουργία ορισμών κειμένου για τιςπιθανές έννοιες των διφορούμενων όρων. Η αποσαφήνιση πραγματοποιείται με τη σύγκριση του πλαισίου του διφορούμενου όρου με τους ορισμούς της κάθε δυνατής έννοιας και επιλέγει αυτή με τις περισσότερες κοινές λέξεις.

2.2.2 ΑΞΙΟΛΟΓΗΣΗ

Το σύστημα αποσαφήνισης λέξεων (WSD) αξιολογήθηκε χρησιμοποιώντας την συλλογή NLM-WSD του Weeber (2001). Αυτή η συλλογή περιέχει 50 διφορούμενους όρους με 100 περιπτώσεις του καθενός. Οι περιπτώσεις είναι

περιλήψεις (abstracts) που περιέχουν τους διφορούμενους όρους τυχαία που προέρχονται από αυτά που προστέθηκαν στο Medline το 1998. Οι 5000 περιπτώσεις αποσαφηνιστήκαν από 11 σχολιαστές, οι οποίοι έβαλαν ετικέτα σε κάθε εμφάνιση του αντίστοιχου όρου με την αντίστοιχη έννοια. Σε μερικές περιπτώσεις τοποθετήθηκε η ετικέτα 'None' για να δείξει ότι οι σχολιαστές δεν θεωρούν καμία από τις πιθανές σημασίες κατάλληλη στο UMLS. Μετά την καθιερωμένη πρακτική (Humphrey το 2006, McInnes, 2008), αυτές οι περιπτώσεις δεν χρησιμοποιούνται για την αξιολόγηση, αποφέροντας συνολικά 3983 δείγματα και 49 όρους. Ο όρος 'association' είχε αποκλειστεί από όλες τις 100 περιπτώσεις που χαρακτηρίστηκαν ως 'None'. Εκτός από το πλήρες NLM-WSD σύνολο δεδομένων, ένα υποσύνολο με 13 από αυτούς τους όρους χρησιμοποιείται επίσης για την αξιολόγηση. Αυτό το υποσύνολο χρησιμοποιήθηκε από τον McInnes (2008) και αποτελείται από όρους που έχουν μια σχεδόν σίγουρη έννοια που υπάρχει για λιγότερο από το 65% των περιπτώσεων και των οποίων οι ενδεχόμενες έννοιες δεν μοιράζονται τον ίδιο σημασιολογικό τύπο. Ένα σύνολο από 20 όρους γύρω από τη λέξη-όρο που θέλουμε (δηλαδή οι 10 προηγούμενες και οι 10 ακόλουθες λέξεις χρησιμοποιείται ως πλαίσιο (context). Αυτό δημιουργείται χρησιμοποιώντας το πρόγραμμα MetaMap (το οποίο χωρίζει το κείμενο εισόδου σε φράσεις) για τον προσδιορισμό των όρων γύρω από τη λέξη-στόχο. Οποιεσδήποτε φράσεις που δεν έχουν αντιστοιχιστεί σε ένα CUI (προσδιοριστής έννοιας - concept unique identifier) απορρίπτονται και οι όροι που αποτελούν καθεμία από τις υπόλοιπες φράσεις χρησιμοποιούνται για να δημιουργήσουν το πλαίσιο. Ο συντελεστής απόσβεσης ορίστηκε σε 0,85. Οι τιμές των παραμέτρων αυτών επιλέχθηκαν με βάση τις προηγούμενες εργασίες (Agirre και Soroa, 2009). Η 2007AB[4] έκδοση του UMLS χρησιμοποιήθηκε για τα πειράματα. Αυτή η έκδοση επελέγη διότι είχαμε πρόσβαση σε μια αντιστοίχιση μεταξύ των ετικετών με τις έννοιες από το NLM-WSD και τα CUIs του UMLS. Μια τέτοια αντιστοίχιση απαιτείται μόνο για να αξιολογήσει την προσέγγισή μας και θα ήταν δυνατόν να χρησιμοποιηθεί η προσέγγιση που περιγράφεται σε αυτό το άρθρο για την εκτέλεση της αποσαφήνισης σε σχέση με οποιαδήποτε έκδοση του UMLS. Η αντιστοίχιση από την έκδοση 2007AB του UMLS δημιουργήθηκε με τη βοήθεια λογισμικού που είναι διαθέσιμο στο κοινό και την επαλήθευση των εννοιών με το χέρι.

[4] http://www.nlm.nih.gov/research/umls/licensedcontent/umlsarchives04.html

2.2.3 ΑΠΟΤΕΛΕΣΜΑΤΑ

Ο παρακάτω πίνακας (Πίνακας 1) παρουσιάζει τα αποτελέσματα της αξιολόγησης του συστήματος. Η επίδοση μετριέται με ακρίβεια, δηλαδή το ποσοστό των περιπτώσεων που η αποσαφήνιση ήταν σωστή. Σημειώστε ότι ο αλγόριθμος μας επιστρέφει μια έννοια για όλες τις περιπτώσεις. Το διάστημα εμπιστοσύνης, που υπολογίζεται χρησιμοποιώντας την αναδειγματοληψία bootstrap με επίπεδο εμπιστοσύνης 95% (Noreen, 1989), εμφανίζεται επίσης. Το πάνω μέρος του πίνακα δείχνει ακόμη τα αποτελέσματα σχετικά με το σύνολο δεδομένων NLM-WSD. Οι δύο πρώτες σειρές δείχνουν το αποτέλεσμα χρησιμοποιώντας εξατομικευμένη PageRank (με την ένδειξη PPR) χρησιμοποιώντας το γράφημα που δημιουργήθηκε από τον πίνακα MRREL και τον συνδυασμό των πινάκων MRREL και MRCOC. Επίσης, περιλαμβάνεται η απόδοση χρησιμοποιώντας δύο γραμμές βάσης. Η πρώτη, στατική, εφαρμόζει τον προσαρμοσμένο αλγόριθμο PageRank (Aggire, 2009) στο γράφημα χωρίς να κάνει χρήση του πλαισίου και η δεύτερη, τυχαία, απλά επιλέγει μια έννοια τυχαία. Όλες οι διαφορές στον πίνακα είναι στατιστικά σημαντικές. Η προσαρμοσμένη PageRank υπερτερεί σαφώς τόσο για τις τυχαίες όσο και για τις στατικές γραμμές βάσης. Η καλύτερη απόδοση επιτυγχάνεται χρησιμοποιώντας τη γραφική παράσταση που δημιουργήθηκε από τον πίνακα MRREL. Τα αποτελέσματα μειώνονται όταν οι επιπλέον σχέσεις από τον πίνακα MRCOC προστίθενται. Αυτή η πτώση στην απόδοση ήταν απροσδόκητη δεδομένου ότι οι συνυπάρχουσες πληροφορίες θεωρούνται γενικά ότι είναι πολύ χρήσιμες για την αποσαφήνιση. Ωστόσο, ο πίνακας MRCOC περιλαμβάνει μόνο τις σχέσεις για ορισμένα CUIs, σε αντίθεση με τον πίνακα MRREL που περιέχει όλες τις σχέσεις για όλα τα CUIs. Αυτό επηρεάζει αρνητικά το προσαρμοσμένο PageRank, καθώς είναι πιο πιθανό να επιλεγούν CUIs που είναι πιο πολύ συνδεδεμένα και το γεγονός ότι ορισμένα CUIs δεν εμφανίζονται στον πίνακα MRCOC δημιουργεί μια προκατάληψη σε σχέση με εκείνα που το κάνουν. Η ανάλυση έδειξε ότι υπάρχουν μόνο τέσσερις όροι (‘ganglion’, ‘man’, ‘secretion’ και ‘surgery’) για τους οποίους όλα τα πιθανά CUIs εμφανίζονται στον πίνακα MRCOC. Για τους 25 όρους, ορισμένα από τα CUIs εμφανίζονται στον πίνακα MRCOC ενώ άλλοι δεν το κάνουν. Επιπλέον, οι σχέσεις στον πίνακα MRCOC δημιουργούνται αυτόματα. Στο κάτω μέρος του πίνακα

δίνονται τα αποτελέσματα για τους 13 όρους που χρησιμοποιούνται από το McInnes (2008). Η καλύτερη προσέγγιση χρησιμοποιώντας το προσαρμοσμένο Pagerank MRREL, επιτυγχάνει καλύτερα αποτελέσματα από εκείνες του McInnes (2008). Κάποιοι πιθανοί λόγοι για αυτή τη βελτίωση των επιδόσεων είναι ότι ο πίνακας MRREL περιέχει πληροφορίες που είναι πιο χρήσιμες για αποσαφήνιση από τους ορισμούς CUI που χρησιμοποιούνται από το McInnes (2008) και ότι οι αλγόριθμοι που βασίζονται σε γράφημα χρησιμοποιούνται για δικό μας όφελος και είναι σε θέση να χρησιμοποιούν τις πληροφορίες από ολόκληρο το UMLS.

Μέθοδος	**Βάση γνώσης**	**Ακρίβεια (διάστημα εμπιστοσύνης)**
NLM-WSD σύνολο δεδομένων		
ppr	MRREL	68.1 [66.8, 69.23]
ppr	MRREL+MRCOC	65.5 [64.3, 66.73]
Στατική απόδοση	MRREL	58.4 [57.07, 59.60]
Τυχαία απόδοση	-	45.6
McInnes(2008) υποσύνολα του NLM-WSD		
ppr	MRREL	55.0 [52.60, 57.76]
McInnes(2008)	-	48.1

Πίνακας 1. Αποτελέσματα από τη μελέτη του Aggire[5] για το σύνολο των δεδομένων NLM-WSD και το υποσύνολο που χρησιμοποιείται στην έρευνα McInnes (2008)

3 ΑΛΓΟΡΙΘΜΟΣ LESK

[5] http://bioinformatics.oxfordjournals.org/content/26/22/2889.full

3.1 ΓΕΝΙΚΑ

Ο αλγόριθμος Lesk(1986) είναι ένας κλασσικός αλγόριθμος που χρησιμοποιείται για την αποσαφήνιση των εννοιών των λέξεων (WSD).

Ο αλγόριθμος Lesk βασίζεται στην υπόθεση ότι οι λέξεις σε μια δεδομένη περιοχή-πρόταση τείνουν να μοιράζονται ένα κοινό θέμα. Μια απλοποιημένη έκδοση του αλγορίθμου Lesk είναι να συγκρίνουμε τον αρχικό ορισμό που παίρνουμε από το λεξικό για μία διφορούμενη λέξη με τους όρους που υπάρχουν στην ίδια περιοχή-πρόταση. Οι εκδόσεις που έχουν προσαρμοστεί στο WordNet. Είναι κάπως έτσι :

- για κάθε έννοια μιας λέξης που αποσαφηνίζεται πρέπει κανείς να μετρήσει τον αριθμό των λέξεων που βρίσκονται τόσο στην ίδια την πρόταση με τη λέξη όσο και στον ορισμό της κάθε έννοιας στο λεξικό.

- η έννοια που πρέπει να επιλεγεί είναι η έννοια που έχει το μεγαλύτερο αριθμό λέξεων.

3.1.1 ΑΡΧΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ LESK

Ο Michael Lesk στην εργασία του δούλεψε πάνω στην αυτόματη αποσαφήνιση εννοιών με χρήση ηλεκτρονικών λεξικών. Αρχικά, προσπάθησε να αποφασίσει αυτόματα ποια από τις έννοιες μιας λέξης ταιριάζει σε κάθε περίπτωση με τη χρήση ηλεκτρονικών λεξικών. Έτσι, αναζήτησε κοινές λέξεις που υπάρχουν στον ορισμό της έννοιας στο λεξικό και στην πρόταση που υπάρχει η αμφίσημη λέξη. Το δίλλημα της απόφασης ποια έννοια της λέξης χρησιμοποιεί ο συγγραφέας είναι ένα σημαντικό πρόβλημα για συστήματα ανάκτησης πληροφοριών. Ο M. Lesk τονίζει ότι σε προηγούμενη εργασία του πάνω σε αυτό το θέμα είχε προτείνει λεπτομερή πλαίσια που περιγράφουν τις έννοιες από τη συγκεκριμένη λέξη ή παγκόσμια στατιστικά στοιχεία σχετικά με τις εμφανίσεις των λέξεων. Το παράδειγμα που ασχολήθηκε ο M. Lesk ήταν το πως να ξεχωρίσεις ένα παγωτό χωνάκι από ένα κουκουνάρι (ice cream cone – pine cone). Για τη λέξη pine στο Αγγλικό λεξικό της Οξφόρδης για προχωρημένους μαθητές (Oxford Advanced Learner's Dictionary of Current English) υπάρχουν δύο σημαντικές έννοιες, η "kind of evergreen tree with needle-shaped leaves.. ." και η "waste away through sorrow or illness...". Για τη λέξη

cone υπάρχουν τρεις διαφορετικοί ορισμοί-έννοιες. Αυτοί οι ορισμοί είναι οι “solid body which narrows to a’ point . . . . ”, “something of this shape whether solid or hollow...” και “fruit of certain evergreen trees...”. Παρατηρούμε ότι οι λέξεις 'evergreen' και 'tree' υπάρχουν σε δύο από τις διαθέσιμες έννοιες, έτσι ένα πρόγραμμα θα μπορούσε να μαντέψει ότι αν οι δύο λέξεις pine cone εμφανίζονται μαζί, οι πιθανή έννοια είναι ότι αναφέρεται σε δέντρο και στον καρπό του(κουκουνάρι).

Στην εικόνα 1 βλέπουμε και το αποτέλεσμα :

```
pine   1*   7 kinds of evergreen tree with needle-shaped
              evergreen(1) tree(6)
       2    1 pine/ -
              pine(1)
       3    0  waste away through sorrow or illness:

       4    0  / pine for sth; pine to do sth, /  have a

cone   1    0  solid body which narrows to a point from a

       2    0  sth of this shape whether solid or hollow,

       3*   8  fruit of certain evergreen trees (fir, pine,
              evergreen(1) tree(6) pine(1)
```

Εικόνα 1. Παράδειγμα εντοπισμού λήμματος για δύο πολύσημες λέξεις.

Οι πολλαπλές έννοιες των λέξεων, είτε σημασιολογικές είτε συντακτικές, είναι ένα σημαντικό πρόβλημα για πολλούς τομείς στην επεξεργασία φυσικών γλωσσών σε υπολογιστές. Η εργασία αυτή είναι μία προσπάθεια για μία φθηνή λύση στο πρόβλημα της διάκρισης των εννοιών των λέξεων.

3.1.2 ΑΠΛΟΠΟΙΗΜΕΝΟΣ ΑΛΓΟΡΙΘΜΟΣ LESK

Σε έναν απλοποιημένο αλγόριθμο Lesk , η σωστή έννοια κάθε λέξης σε ένα δεδομένο πλαίσιο καθορίζεται μεμονωμένα από τον εντοπισμό της έννοιας που υπερκαλύπτει περισσότερο από τις άλλες μεταξύ του ορισμού από το λεξικό και του

δεδομένου πλαισίου. Η προσέγγιση αυτή αντιμετωπίζει κάθε λέξη ξεχωριστά, ανεξάρτητα από την έννοια των άλλων λέξεων που βρίσκονται στο ίδιο πλαίσιο-πρόταση.

Μια συγκριτική αξιολόγηση που πραγματοποιήθηκε (Vasilescu et al., 2004) απέδειξε ότι ο απλοποιημένος αλγόριθμος Lesk μπορεί να ξεπεράσει σημαντικά τον αρχικό ορισμό του αλγορίθμου, τόσο στην ακρίβεια όσο και στην αποτελεσματικότητα. Αξιολογώντας τους αλγορίθμους αποσαφήνισης με τα λεξιλογικά δεδομένα από το Senseval-2 English, μετρήθηκε 58% ακρίβεια χρησιμοποιώντας τον απλοποιημένο αλγόριθμο Lesk σε σύγκριση με το μόλις 42% του αρχικού αλγορίθμου.

Ο απλοποιημένος αλγόριθμος Lesk (Vasilescu, et al.,2004)

```
Function SIMPLIFIED LESK(word, sentence) returns best sense of word

best-sense <- most frequent sense for word
max-overlap <- 0
context <- set of words in sentence
for each sense in senses of word do
        signature <- set of words in the gloss and examples of sense
        overlap<-COMPUTEOVERLAP (signature, context)
        if overlap > max-overlap then
            max-overlap <- overlap
            best-sense <- sense

end return (best-sense)
```

Πίνακας 2. Απλοποιημένος αλγόριθμος Lesk (Vasilescu, et al.,2004)

Με μία δομή επανάληψης ελέγχουμε όλες τις πιθανές έννοιες. Η λειτουργία COMPUTEOVERLAP επιστρέφει τον αριθμό των κοινών λέξεων μεταξύ των δύο συνόλων, αγνοώντας τις λειτουργικές (άρθρα, σύνδεσμοι, προθέσεις, κλπ.) λέξεις ή άλλες λέξεις που εμφανίζονται συχνά στο λόγο (stopword list). Τέλος, καταχωρεί την έννοια με τις περισσότερες κοινές λέξεις και την επιστρέφει στο κύριο πρόγραμμα. Ενώ ο αρχικός αλγόριθμος Lesk καθορίζει το πλαίσιο με ένα πιο σύνθετο τρόπο.

3.1.3 ΕΠΙΚΡΙΣΕΙΣ ΚΑΙ ΑΛΛΟΙ ΜΕΘΟΔΟΙ ΠΟΥ ΒΑΣΙΖΟΝΤΑΙ ΣΤΟΝ ΑΛΓΟΡΙΘΜΟ LESK

Δυστυχώς, η προσέγγιση Lesk είναι πολύ ευαίσθητη στην ακριβή διατύπωση των ορισμών, έτσι ώστε η απουσία μιας συγκεκριμένης λέξης μπορεί να αλλάξει ριζικά τα αποτελέσματα. Περαιτέρω, ο αλγόριθμος προσδιορίζει επικαλύψεις μόνο μεταξύ των εννοιών που εξετάζονται. Αυτό είναι ένας σημαντικός περιορισμός γι' αυτές τις λεξιλογικές συγκαλύψεις που τείνουν να είναι αρκετά σύντομες και δεν παρέχουν επαρκή λεξιλόγιο. Πρόσφατα, πολλές εργασίες δημιουργήθηκαν που προσφέρουν διαφορετικές τροποποιήσεις του αλγορίθμου. Οι εργασίες αυτές χρησιμοποιούν άλλους πόρους για την ανάλυση (λεξιλογικούς θησαυρούς, λεξικά με συνωνύμων ή μορφολογικά και συντακτικά μοντέλα). Υπάρχουν πολλές μελέτες σχετικά με τον αλγόριθμο Lesk και τις επεκτάσεις του (Kwong, 2001; Nastase and Szpakowicz, 2001; Wilks και Stevenson, 1998, 1999; Mahesh, 1997; Cowie, 1992; Yarowsky, 1992; Pook και Catlett, 1988; Kilgarriff & Rosensweig, 2000; Gelbukh and Sidorov, 2004).

4 ΠΕΡΙΓΡΑΦΗ ΤΟΥ UMLS-NLM

4.1 ΣΧΕΤΙΚΑ ΜΕ ΤΟ UMLS

Ο σκοπός του Ενιαίου συστήματος ιατρικής γλώσσας (Unified Medical Language System- UMLS) είναι να διευκολύνει την ανάπτυξη των υπολογιστικών συστημάτων που συμπεριφέρονται έτσι ώστε να "καταλαβαίνουν" το νόημα της βιοϊατρικής γλώσσας και της υγείας. Για το σκοπό αυτό, η (National Library of Medicine) NLM παράγει και διανέμει τις πηγές γνώσης UMLS (βάσεις δεδομένων) και τα συναφή εργαλεία λογισμικού (προγράμματα) που χρησιμοποιούνται από τους προγραμματιστές του συστήματος για την κατασκευή ή τη βελτίωση των συστημάτων ηλεκτρονικής πληροφόρησης της βιοϊατρικής των δεδομένων υγείας και

πληροφοριών, καθώς και στην έρευνα της πληροφορικής. Από τη σχεδίαση, οι πηγές γνώσης UMLS είναι πολλαπλών χρήσεων. Δεν έχουν βελτιστοποιηθεί για συγκεκριμένες εφαρμογές, αλλά μπορούν να εφαρμοστούν σε συστήματα που εκτελούν μια σειρά από λειτουργίες που αφορούν έναν ή περισσότερους τύπους πληροφοριών, π.χ. τους φακέλους των ασθενών. Τα συναφή εργαλεία λογισμικού του UMLS βοηθούν τους προγραμματιστές στην προσαρμογή ή τη χρήση των πηγών γνώσεων UMLS για συγκεκριμένους σκοπούς. Τα λεξιλογικά εργαλεία λειτουργούν αποτελεσματικότερα σε συνδυασμό με τις πηγές γνώσεις UMLS, αλλά μπορούν να χρησιμοποιηθούν και ανεξάρτητα.

4.2 ΣΧΕΤΙΚΑ ΜΕ ΤΟ NLM

Η Εθνική Βιβλιοθήκη Ιατρικής (NLM), στην πανεπιστημιούπολη του Εθνικού ινστιτούτου υγείας στην Bethesda, έγινε το κέντρο καινοτόμων πληροφοριών από την ίδρυσή της (1836) έως και σήμερα. Ως η μεγαλύτερη βιβλιοθήκη βιοϊατρικής του κόσμου, η NLM διατηρεί και διαθέτει μια τεράστια συλλογή εντύπων και παράγει ηλεκτρονικές μορφές πληροφόρησης για ένα ευρύ φάσμα θεμάτων που αναζητείται δισεκατομμύρια φορές κάθε χρόνο από εκατομμύρια ανθρώπους σε όλο τον κόσμο. Επίσης υποστηρίζει και διεξάγει έρευνα ανάπτυξης και κατάρτισης στον τομέα της πληροφορικής, στον τομέα της βιοϊατρικής και της τεχνολογίας και των πληροφοριών για την υγεία. Επιπλέον, η Βιβλιοθήκη συντονίζει 6000 μέλη του Εθνικού Δικτύου Βιβλιοθηκών της Ιατρικής που προωθεί και παρέχει πρόσβαση σε πληροφορίες υγείας στις κοινότητες των Ηνωμένων Πολιτειών.

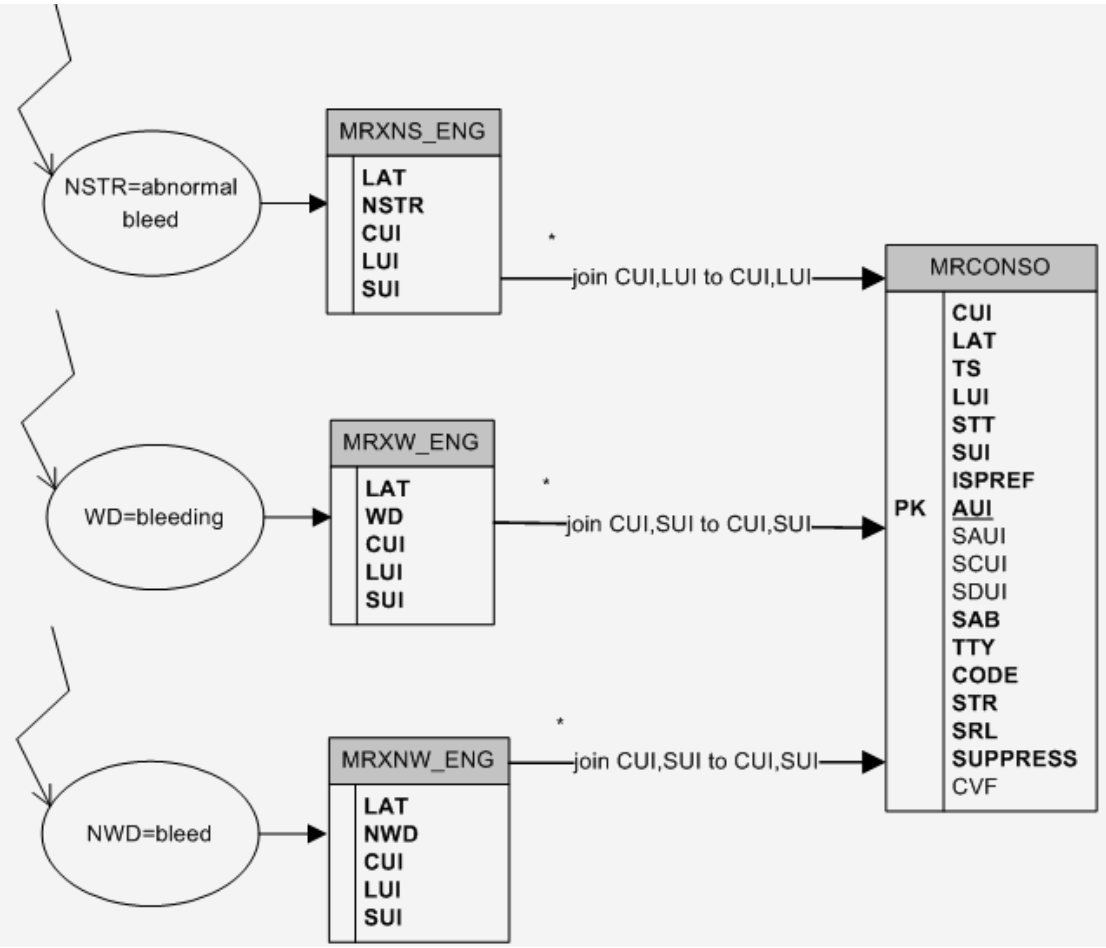
4.3 ΔΙΑΓΡΑΜΜΑΤΑ ΕΡΩΤΗΜΑΤΟΣ ΣΕ ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ UMLS

Τα UMLS RRF (Rich Release Format) ή αλλιώς αρχεία σε μορφή εμπλουτισμένου κειμένου είναι μια συλλογή από σχεσιακά αρχεία που περιέχουν τα δεδομένα του UMLS Metathesaurus. Τα δεδομένα σε αυτά τα αρχεία έχουν

επεξεργαστεί με ποικίλους τρόπους ώστε να είναι πιο εύκολα προσβάσιμα και να διατηρούν τις πορείες πρόσβασης όπως η αρχική μορφή έκδοσης (ORF). Αποτέλεσμα αυτού είναι η ανικανότητα να καθιστούν τα μοντέλα δεδομένων, όπως μια σειρά από ακριβή διαγράμματα Οντοτήτων-Συσχετίσεων (ER).

Ως εναλλακτική λύση, υπάρχει ένα σύνολο από διαγράμματα ER που ονομάζονται "διαγράμματα ερωτήματος". Κάθε διάγραμμα κρίνει μια συγκεκριμένη περίπτωση χρήσης και δείχνει πως τα αρχεία μπορούν να ενωθούν για να βρεθούν όλα τα σχετικά δεδομένα για τη συγκεκριμένη περίπτωση χρήσης. Τα διαγράμματα και τα ερωτήματα (queries) παίρνουν στοιχεία από τα αρχεία δεδομένων RRF από μια βάση δεδομένων με τη χρήση σεναρίων φόρτισης που δημιουργήθηκαν από ένα εργαλείο εγκατάστασης της MetamorphoSys (MMSYS).

Το παρακάτω διάγραμμα δείχνει πως να πραγματοποιήσουμε αναζητήσεις για κανονικοποιημένες συμβολοσειρές (normalized strings - NSTR), κανονικοποιημένες λέξεις (normalized words - NWD), ή κανονικές λέξεις (words - WD).



Εικόνα 2. Διάγραμμα ερωτημάτων σε βάση δεδομένων UMLS
Πηγή(http://www.nlm.nih.gov/research/umls/implementation_resources/query_diagrams/er6.html)

5 ΑΠΟΣΑΦΗΝΙΣΗ ΙΑΤΡΙΚΩΝ ΟΡΩΝ

5.1 ΑΞΙΟΛΟΓΗΣΗ WSD ΤΕΧΝΙΚΩΝ ΣΕ ΙΑΤΡΙΚΟΥΣ ΟΡΟΥΣ

Ασάφεια είναι το φαινόμενο ότι μια λέξη έχει περισσότερες από μία έννοια. Αυτό το φαινόμενο δημιουργεί δυσκολίες για πολλά από τα σημερινά συστήματα Φυσικής Επεξεργασίας Γλώσσας (Natural Language Processing-NLP). Οι αλγόριθμοι που

βοηθούν την ανάλυση αυτών των ασαφειών αποσαφηνίζοντας μία λέξη ή και γενικότερα ένα κείμενο ενισχύουν τις επιδόσεις των συστημάτων αυτών.

Τα συστήματα NLP για ιατρικούς όρους έχουν γενικά σχεδιαστεί για να αναλύουν ιατρικά κείμενα για την υποστήριξη λήψης αποφάσεων ή για σκοπούς ευρετηρίου. Το σύστημα MedLEE (Medical Language Extracting and Encoding) του Columbia University's που αρχικά σχεδιάστηκε για ένα μικρό ιατρικό (και γλωσσολογικό) τομέα έχει εφαρμοστεί σε διαφορετικά πεδία όπως τα φάρμακα. Ένα από τα προβλήματα που προκύπτουν κατά τη διεύρυνση του πεδίο εφαρμογής ενός τέτοιου συστήματος είναι η εισαγωγή των ασαφειών. Ένας όρος ή μια λέξη έχει διαφορετικές έννοιες σε διαφορετικές ιατρικές ειδικότητες. Το MedLEE έχει κάποιους ειδικούς κανόνες για την αντιμετώπιση των ασαφειών, αλλά υπάρχει ανάγκη για νέες τεχνικές μηχανικής μάθησης (ML) και μια καλή συλλογή από δεδομένα εκπαίδευσης. Ο στόχος των ευρετηρίων της Εθνικής Βιβλιοθήκης της Ιατρικής (NLM) είναι να διερευνήσει μεθόδους με τις οποίες οι αυτοματοποιημένες τεχνικές ευρετηρίασης μπορούν μερικώς ή και πλήρως να υποκαταστήσουν τις τρέχουσες. Τα σφάλματα ανάλυσης του συστήματος ευρετηρίασης δείχνουν ότι τα μεγάλα προβλήματα αφορούν την ασάφεια των όρων. Επίσης, το MetaMap, ένα κείμενο για την έννοια του προγράμματος αντιστοίχισης είναι σήμερα σε θέση να αποσαφηνίζει διφορούμενες έννοιες. Το σύστημα DAD είναι ένα εργαλείο που βασίζεται σε μια ιδέα για λογοτεχνικές ανακαλύψεις στην βιοϊατρική. Αυτό το σύστημα χρησιμοποιεί το MetaMap για την επεξεργασία των κειμένων MEDLINE. Στην εκτέλεση της ανακάλυψης του Swanson, που βασίζεται στη λογοτεχνία, όσον αφορά το ρόλο της έλλειψης μαγνησίου κατά την ημικρανία το σύστημα DAD έδειξε ότι η συντομογραφία mg θα μπορούσε να είναι ενδιαφέρουσα για τη θεραπεία της ημικρανίας. Ωστόσο, το MetaMap δεν είναι σε θέση να διακρίνει ποια από τις UMLS έννοιες μαγνήσιο (Magnesium) ή χιλιοστόγραμμο (Milligram) αντιστοιχεί στη συντομογραφία mg. Αυτό σημαίνει ότι ψεύτικες πληροφορίες σχετικά με το χιλιοστόγραμμο (Milligram) περιλαμβάνονται στα αποτελέσματα του συστήματος. Σε μία πρόσφατη μελέτη πάνω στο UMLS σχέδιο (Nadkami) καταλήξανε ότι μια πλήρως αυτόματη διαδικασία δεν είναι ακόμη εφικτή, εν μέρει λόγω των προβλημάτων ασάφειας. Αν και είναι σαφές ότι υπάρχει ανάγκη, λίγη έρευνα έχει αφιερωθεί στην βιοϊατρική αποσαφήνιση της έννοιας μιας λέξης. Πρόσφατα, για την αποσαφήνιση (WSD) έχει παρατηρηθεί μια αύξηση του ενδιαφέροντος στην

υπολογιστική γλωσσολογία, συνοδευόμενο από ένα ειδικό τεύχος της Υπολογιστικής Γλωσσολογίας (Computational Linguistics) το 1998 και μία ειδική έκδοση της Πληροφορικής και των Ανθρωπιστικών Επιστημών το 2000. Επιπλέον, υπάρχουν τα SENSEVAL workshops για την αξιολόγηση αλγορίθμων και τεχνικών για WSD και άλλων γλωσσολογικών εργασιών.

Ο χρόνος είναι κατάλληλος για τη δοκιμή αλγορίθμων που αναπτύχθηκαν πρόσφατα σε τομείς της βιοϊατρικής γλώσσας. Απαραίτητη για τον έλεγχο των αλγορίθμων είναι μία συλλογή από όρους κειμένου βιοϊατρικού χαρακτήρα που έχουν αποσαφηνιστεί με το χέρι και όχι αυτόματα. Αυτή η συλλογή χρησιμοποιείται ως ένα χρυσό πρότυπο.

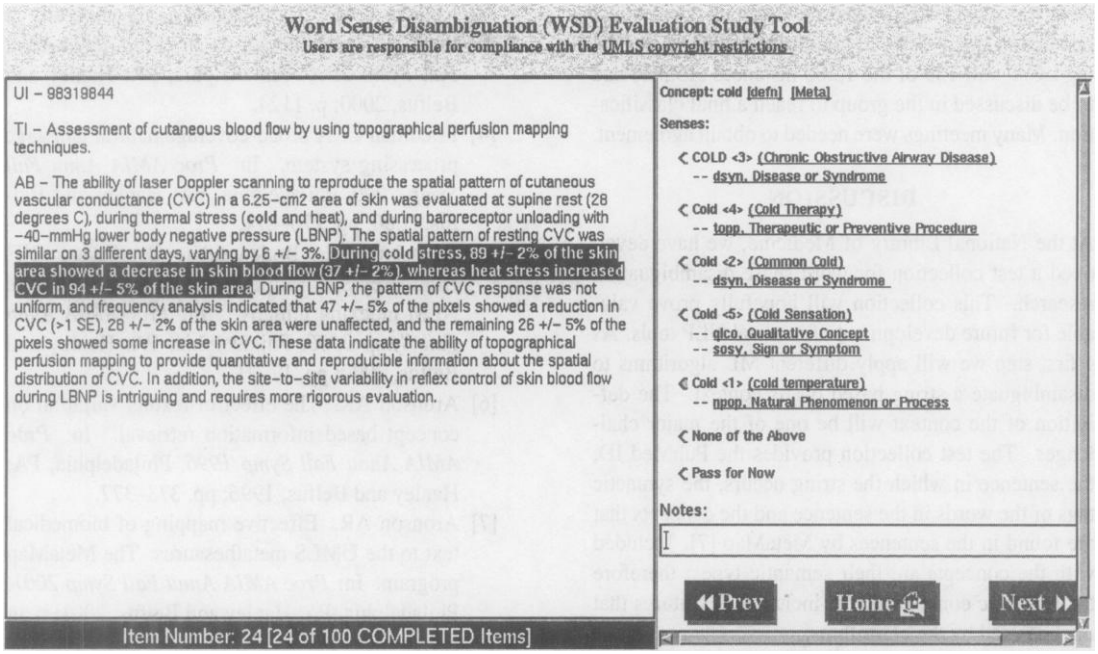
5.2 ΔΙΑΔΙΚΑΣΙΑ ΑΠΟΣΑΦΗΝΙΣΗΣ ΣΤΗ ΣΥΛΛΟΓΗ ΤΟΥ NLM

Για να ελέγξουμε τέτοιες τεχνικές στον τομέα της βιοϊατρικής γλώσσας, έχει αναπτυχθεί από το NLM μία δοκιμαστική συλλογή αποσαφήνισης των εννοιών των λέξεων (WSD) που συγκρίνει 5000 περιπτώσεις αποσαφήνισης για 50 διφορούμενες λέξεις. Περιγραφές γι' αυτό βρίσκουμε το dataset. http://wsd.nlm.nih.gov/

Η δοκιμαστική αυτή συλλογή αποτελείται από 50 πολύ συχνά εμφανιζόμενους διφορούμενους όρους του UMLS MEDLINE από το 1998. Κάθε ένας από τους 50 διφορούμενους όρους έχει 100 περιπτώσεις στις οποίες εμφανίζεται ο όρος με διαφορετική έννοια. Αυτές οι περιπτώσεις επιλέγονται τυχαία από τις αναφορές του MEDLINE από το 1998. Για ένα σύνολο από 5000 περιπτώσεις. Συνολικά υπήρχαν 11 αξιολογητές από τους οποίους οι 8 ολοκλήρωσαν το 100% των περιπτώσεων. Ένας αξιολογητής συμπλήρωσε το 56% , ένας άλλος το 44% και ο τελευταίος μόνο το 12% των περιπτώσεων. Οι αξιολογήσεις χρησιμοποιούνταν μόνο όταν όλοι οι αξιολογητές ολοκλήρωναν και τις 100 περιπτώσεις για ένα δεδομένο βιοϊατρικό όρο.

Για τη μελέτη των Marc Weeber et al., δεδομένου ότι 5.000 περιπτώσεις ασάφειας δεν είναι εύκολο έργο για να γίνει χειροκίνητα, αναπτύχθηκε μια διεπαφή που διευκολύνει τη διαδικασία αποσαφήνισης και μειώνει το πραγματικό έργο που γίνεται χειροκίνητα σε δύο κλικ του ποντικιού για κάθε περίπτωση(όπως το

παράδειγμα στην εικόνα 3 παρακάτω).



Εικόνα 3. Διεπαφή αποσαφήνισης. Το αριστερό παράθυρο εμφανίζει την αναφορά MEDLINE ως πλαίσιο για τους βαθμολογητές ώστε να αποσαφηνίζουν το κρύο string. Οι πιθανές έννοιες, με συνδέσμους προς τις UMLS, είναι στη δεξιά πλευρά.

Στο αριστερό πάνελ της διεπαφής η πρόταση στην οποία εμφανίζεται η λέξη που αποκρυσταλλώνεται είναι σε μπλε πλαίσιο, ενώ η λέξη που μας ενδιαφέρει σε κόκκινο(δεν ξεχωρίζει πολύ καλά αλλά στη συγκεκριμένη περίπτωση η λέξη είναι η 'cold'). Επιπλέον, τα υπόλοιπα από τον τίτλο και την περίληψη της παραπομπής MEDLINE είναι ορατά. Οι βαθμολογητές έχουν τη δυνατότητα να τοποθετήσουν τη λέξη που πρόκειται να αποσαφηνιστεί σε οποιαδήποτε σειρά και δεν απαιτείται η ολοκλήρωση αποσαφήνισης μιας λέξης για την έναρξη μιας άλλης. Η σειρά με την οποία οι 100 περιπτώσεις κάθε λέξης παρουσιάζονται είναι τυχαία για κάθε χρήστη. Επίσης, η σειρά με την οποία παρουσιάζονται οι έννοιες είναι και αυτή τυχαία για κάθε χρήστη. Οι διαφορετικές έννοιες είναι διαθέσιμες στον δεξιό πίνακα. Ο εκτιμητής μπορεί να επιλέξει μόνο μία έννοια από τις διαθέσιμες ή να προσπεράσει το παράδειγμα για να το επανεξετάσει αργότερα. Οι έννοιες και οι σημασιολογικοί τύποι τους έχουν υπερσυνδέσμους στο UMLS.

5.2.1 ΑΝΑΛΥΣΗ ΤΩΝ ΒΑΘΜΟΛΟΓΙΩΝ

Για να επιτευχθεί μία τελική κατάταξη για τη σωστή έννοια που θα επιλεγεί για κάθε περίπτωση ασάφειας, χρησιμοποιήθηκαν δύο προσεγγίσεις. Η πρώτη είναι η πλειοψηφία. Η έννοια που έχει επιλεγεί από τους περισσότερους βαθμολογητές είναι η τελική και σωστή έννοια. Όταν στην περίπτωση που η πλειοψηφία καταλήγει σε μία (σχεδόν) ισοπαλία, χρησιμοποιείται η δεύτερη μέθοδος, που ονομάζεται λανθάνουσα ταξική ανάλυση (LCA). Για κάθε περίπτωση, χρησιμοποιεί τα πρότυπα αξιολόγησης από τις άλλες περιπτώσεις ώστε να αποφασίσουν ποια είναι η αληθινή και τελική κατάταξη. Εκτός από αυτές τις μεθόδους, είναι ενδιαφέρον να υπολογίσουμε τον βαθμό στον οποίο οι βαθμολογητές συμφωνούν μεταξύ τους χρησιμοποιώντας τη στατιστική kappa(K)[6]. Η στατιστική kappa είναι ένα στατιστικό μέτρο για την αξιολόγηση της αξιοπιστίας της συμφωνίας μεταξύ ενός σταθερού αριθμού βαθμολογητών, στους οποίους έχει ανατεθεί η αξιολόγηση ενός αριθμού στοιχείων. Ο προσδιορισμός των τελικών ταξινομήσεων είναι διαδικασία τεσσάρων βημάτων. Στο πρώτο βήμα, υπολογίζουμε το K για κάθε συνδυασμό βαθμολογητών. Αυτό το στατιστικό (K) δείχνει ποιοι βαθμολογητές συμφωνούν μεταξύ τους και πιο σημαντικό ποιοι βαθμολογητές διαφωνούν συστηματικά από τους υπόλοιπους. Στο δεύτερο βήμα, μετράμε τις συνολικές μετρήσεις για κάθε περίπτωση λέξης. Εάν υπάρχει μια πλειοψηφία από δύο ψηφοφορίες για μία ορισμένη έννοια αυτή είναι η τελική κατάταξη. Στην περίπτωση ισοβαθμίας ή πλειοψηφίας του ενός, εξαιρείται ένας εκτιμητής-βαθμολογητής, εάν αυτός διαφωνεί συστηματικά με όλους τους υπόλοιπους. Εφαρμόζουμε το βήμα τρία, αν το δεύτερο βήμα δεν καταλήγει σε ικανοποιητικά αποτελέσματα για πολλές περιπτώσεις λέξεων, δηλαδή να υπάρχουν πολλές ισοβαθμίες και παρόλο την εξαίρεση ενός ή περισσοτέρων βαθμολογητών τα αποτελέσματα δεν βελτιώνονται. Γι' αυτές τις περιπτώσεις, χρησιμοποιούμε τη μέθοδο LCA που είδαμε και παραπάνω για να έχουμε μια ταξινόμηση. Το τέταρτο βήμα είναι η επανεξέταση των περιπτώσεων σε μια ομάδα συζήτησης από την ομάδα αποσαφήνισης για τις περιπτώσεις που δεν επιλύθηκαν από τα βήματα 2 και 3.

5.2.2 ΑΠΟΤΕΛΕΣΜΑΤΑ

[6] http://en.wikipedia.org/wiki/Fleiss%27_kappa

Ανάλογα με τις δυσκολία της υπόθεσης, οι βαθμολογητές δαπάνησαν από 30 λεπτά έως και 2 ώρες ανά διφορούμενη λέξη (100 περιπτώσεις). Η τελική βαθμολόγηση έγινε επιπλέον από τα κανονικά καθήκοντα των βαθμολογητών. Μετά από μια περίοδο τεσσάρων μηνών κατά την οποία υπήρχαν τρεις συναντήσεις στις οποίες η ομάδα συζήτησε παραδείγματα δύσκολων λέξεων και ειδικές περιπτώσεις, τα δεδομένα πάγωσαν. Οκτώ βαθμολογητές ολοκλήρωσαν και τις 5.000 περιπτώσεις, ενώ άλλοι τρεις ολοκλήρωσαν 2.800 (28 λέξεις), 2.200 (22 λέξεις) και 600 (6 λέξεις) περιπτώσεις αντίστοιχα.

Η τελική ανάλυση με την στατιστική K παρείχε πολλές ενδιαφέρουσες ιδέες. Για παράδειγμα, οι δύο βαθμολογητές που συμφώνησαν για περισσότερες από 50 λέξεις εργάζονταν και οι δύο στα ευρετήρια NLM. Επίσης, για πολλές αμφίσημες λέξεις ένας ή δύο από τους βαθμολογητές διαφώνησαν συστηματικά με το υπόλοιπο της ομάδας. Με την εξαίρεσή τους σε έντεκα περιπτώσεις η ομάδα κατάφερε να επιλύσει τα προβλήματα. Οκτώ βαθμολογητές αποκλείστηκαν τουλάχιστον μία φορά. Για 38 λέξεις δηλαδή 3.800 περιπτώσεις η επίτευξη της τελικής κατάταξης ήταν σχετικά απλή, και για τις περισσότερες από αυτές τις περιπτώσεις τα βήματα ένα, δύο και τρία ήταν επαρκείς. Μόνο για τις 162 από τις 3.800 περιπτώσεις (4,3 %) χρειάστηκε να συζητήσει η ομάδα για την τελική κατάταξη (βήμα 4). Οι δώδεκα υπόλοιπες λέξεις ήταν περισσότερο προβληματικές και υπήρξε διαφωνία μεταξύ των βαθμολογητών. Μετά τη χρήση της μεθόδου LCA ακόμα 159 από τις 1.200 περιπτώσεις (13.3%) έπρεπε να συζητηθούν στην ομάδα ώστε να επιτευχθεί μια τελική ταξινόμηση. Πολλές συναντήσεις χρειάστηκαν ώστε να επιτευχθεί συμφωνία.

6 ΥΛΟΠΟΙΗΣΗ

Στην UMLS Metathesaurus υπάρχει ένας μεγάλος αριθμός περιπτώσεων διφορούμενων όρων όπου καθεμία από αυτές αναφέρεται σε μία από τις αντίστοιχες πιθανές έννοιες. Προκειμένου να υποστηριχθεί η έρευνα, η αυτόματη επίλυση των

αμφίσημων λέξεων χρησιμοποιώντας φυσικές τεχνικές επεξεργασίας γλώσσας, κατασκεύαστηκε αυτή η συλλογή των ιατρικών κειμένων στην οποία οι ασάφειες επιλύθηκαν με το χέρι από εκτιμητές. Οι εκτιμητές εξέτασαν τις αμφίσημες λέξεις και επέλεξαν την καταλληλότερη (σύμφωνα με αυτούς) έννοια. Η δοκιμαστική συλλογή αποτελείται από 50 διφορούμενους όρους του UMLS που εμφανίζονται συχνά στο MEDLINE από το 1998. Κάθε ένας από τους 50 διφορούμενους όρους έχει 100 περιπτώσεις όπου εμφανίζεται, οι οποίες σε κάθε περίπτωση επιλέγονται τυχαία από αναφορές του MEDLINE από το 1998. Έτσι, έχουμε ένα σύνολο 5000 περιπτώσεων για αποσαφήνιση.

adjustment	**evaluation**	**implantation**	**reduction**	**transient**
association	**extraction**	**inhibition**	**repair**	**transport**
blood_pressure	**failure**	**japanese**	**resistance**	**ultrasound**
cold	**fat**	**lead**	**scale**	**variation**
condition	**fit**	**man**	**secretion**	**weight**
culture	**fluid**	**mole**	**sensitivity**	**white**
degree	**frequency**	**mosaic**	**sex**	
depression	**ganglion**	**nutrition**	**single**	
determination	**glucose**	**pathology**	**strains**	
discharge	**growth**	**pressure**	**support**	
energy	**immunosuppression**	**radiation**	**surgery**	

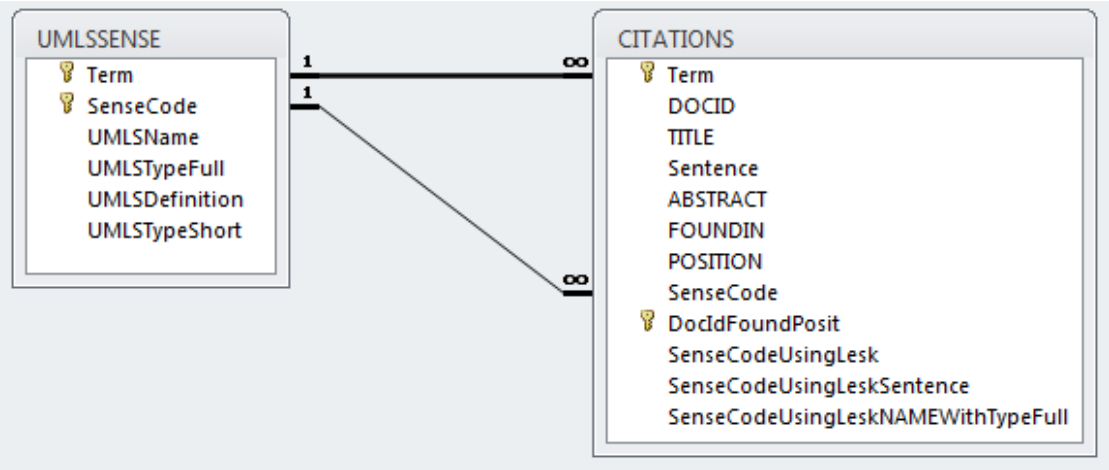
Πίνακας 3. Οι 50 όροι-λέξεις που καλούμαστε να αποσαφηνίσουμε

Στη μελέτη που έκανα χρησιμοποίησα αυτή τη δοκιμαστική συλλογή από βιοϊατρικούς όρους. Τους διφορούμενους όρους όπως και τις περιπτώσεις κάθε όρου τις αποθήκευσα σε βάσεις δεδομένων ώστε να μπορώ να τα επεξεργαστώ και να έχω αποτελέσματα.

6.1 ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΔΗΜΙΟΥΡΓΙΑ ΠΙΝΑΚΩΝ

Αρχικά για την εργασία χρησιμοποίησα βάσεις δεδομένων MySQL database για να αποθηκεύσω τους 50 όρους που θα αποσαφηνίσουμε όπως και τα 100

παραδείγματα με πιθανές έννοιες του κάθε όρου. Έφτιαξα δύο πίνακες και τους ονόμασα CITATIONS και UMLSSense όπως φαίνεται στην Εικόνα 4.



Εικόνα 4. Το σχήμα της ΒΔ που αφορά τους όρους

Η στήλη Term περιέχει το όνομα του βιοϊατρικού όρου ή της λέξης π.χ. Adjustment,Blood Pressure, Cold. Η στήλη SenseCode περιέχει τους κωδικούς των εννοιών, δηλαδή όποια έννοια έχει επιλεχθεί για το συγκεκριμένο όρο ή λέξη θα έχει και τον αντίστοιχο κωδικό π.χ. M1, M2. Ο κάθε κωδικός είναι μοναδικός. Στη στήλη UMLSName έχουμε τον τίτλο της κάθε έννοιας π.χ. Individual Adjustment, Adjustment Action. Υπάρχουν ορισμένες έννοιες οι οποίες δεν έχουν καθόλου τίτλο, σε αυτή την περίπτωση στο πεδίο του πίνακα είναι κενό. Η στήλη UMLSDefinition περιέχει τον ορισμό κάθε επιλεγμένης έννοιας που δίδεται από το database του UMLS. Τέλος, στις στήλες UMLSTypeFull και UMLSTypeShort έχουμε τον τύπο της κάθε έννοιας και πλήρες γραμμένο αλλά και σε συντομογραφία, όπως για παράδειγμα 'Individual Behavior' και 'inbe' αντίστοιχα για κάθε πεδίο. Στήλες κλειδιά (primary) είναι οι στήλες Term και SenseCode.

Ο πίνακας CITATIONS περιέχει στοιχεία για καθένα από τα 100 παραδείγματα για όλους τους βιοϊατρικούς όρους. Ο πίνακας CITATIONS συνδέεται με τον πίνακα UMLSSense με ξένα κλειδιά τις στήλες Term και SenseCode. Κάθε παράδειγμα από τα 5000 περιέχει στοιχεία που τα αποθηκεύω στις αντίστοιχες στήλες

του πίνακα. Η στήλη Term περιέχει τους βιοϊατρικούς όρους ή λέξεις π.χ. Adjustment, Blood Pressure, Cold όπως ακριβώς και στον πίνακα UMLSSense. Η στήλη DOCID περιέχει έναν οκταψήφιο αριθμό π.χ. 97452340 που αποτελεί το id του κάθε παραδείγματος. Η στήλη TITLE περιέχει τον τίτλο από το κείμενο κάθε παραδείγματος, το οποίο μπορεί να περιέχει και τη λέξη ή τον όρο που μας ενδιαφέρει. Στη στήλη με το όνομα Sentence υπάρχει η πρόταση που περιέχει τον βιοϊατρικό όρο που θα προσπαθήσουμε να αποσαφηνίσουμε. Αυτή η πρόταση μπορεί να είναι είτε ο τίτλος, αρά στη συγκεκριμένη περίπτωση θα είναι η ίδια με την προηγούμενη στήλη, είτε μία πρόταση από το γενικότερο κείμενο (ABSTRACT). Η στήλη ABSTRACT περιέχει το κείμενο που βρίσκεται ο διφορούμενος όρος. Υπάρχει βέβαια και η περίπτωση να μην βρίσκεται στο κείμενο αλλά στον τίτλο του κειμένου, παρόλα αυτά το κείμενο αναφέρεται και αποθηκεύεται στη βάση δεδομένων. Η στήλη FOUNDIN είναι η στήλη που μας δείχνει σε ποιό σημείο βρίσκεται η πολύσημη λέξη. Η στήλη αυτή περιέχει έναν μονοψήφιο αριθμό που παίρνει τις τιμές 0 και 1, αν ο αριθμός αυτός είναι 0 τότε η λέξη βρίσκεται στον τίτλο του κειμένου αντιθέτως στην περίπτωση που ο αριθμός είναι 1 τότε η λέξη βρίσκεται στο κείμενο (ABSTRACT). Η στήλη POSITION μας δείχνει σε ποια πρόταση του κειμένου βρίσκεται ο όρος ή η λέξη με τον αντίστοιχο αριθμό. Στην περίπτωση που βρίσκεται στον τίτλο του κειμένου, το πεδίο περιέχει τον αριθμό 1, αλλιώς αν βρίσκεται στο κείμενο τον αριθμό της αντίστοιχης πρότασης. Η στήλη SenseCode περιέχει τον κωδικό από την έννοια που έχει επιλεγεί από τους ανθρώπους του UMLS. Στην περίπτωση που δεν έχει επιλεγεί κάποια έννοια ή είναι απροσδιόριστο τότε το πεδίο παίρνει την τιμή ‘null’. Στη στήλη DocIdFoundPosit περιέχονται στοιχεία που έχουν αναφερθεί παραπάνω αλλά για λόγους μοναδικότητας δημιούργησα αυτή τη στήλη, η οποία είναι και πρωτεύον κλειδί για τον πίνακα CITATIONS μαζί με τη στήλη Term. Η μορφή του περιεχομένου αυτής της στήλης είναι π.χ. 97452340.ab.6, όπου ο πρώτος αριθμός είναι το DOCID, το ab δείχνει ότι η πολύσημη λέξη βρίσκεται στο κείμενο και το 6 δείχνει την πρόταση του κειμένου που βρίσκεται η λέξη. Στην περίπτωση που η λέξη βρίσκεται στον τίτλο τότε το πεδίο της στήλης DocIdFoundPosit είναι της μορφής π.χ. 98019371.ti.1, όπου ti το TITLE. Η στήλη SenseCodeUsingLesk περιέχει τα αποτελέσματα μετά από τη χρησιμοποίηση του αλγορίθμου Lesk που κάνει την αυτόματη αποσαφήνιση. Αρχικά, όλα τα πεδία της στήλης είναι κενά καθώς αποτελούν το στόχο της διαδικασίας αποσαφήνισης. Για το σκοπό αυτό ανέπτυξα ένα πρόγραμμα που υλοποιεί τον αλγόριθμο Lesk και ενημερώνει τη ΒΔ. Σε αυτή την

περίπτωση συγκρίνω τη στήλη UMLSDefinition του πίνακα UMLSSense με το κείμενο που περιέχει τον διφορούμενο όρο, δηλαδή αν βρίσκεται στον τίτλο με το TITLE ενώ αν βρίσκεται στο κείμενο με το ABSTRACT, του πίνακα CITATIONS.

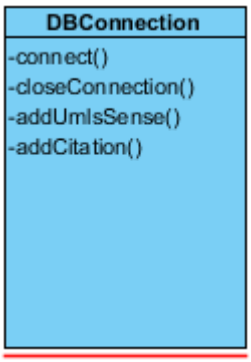
Στην περίπτωση που το πρόγραμμα δεν επιλέξει κάποια έννοια, δηλαδή το πεδίο είναι κενό ή έχει την τιμή 'null', αυτόματα επιλέγω τον κωδικό από την πρώτη έννοια (M1). Η τεχνική αυτή είναι αντίστοιχη της Most Popular Sense (First Sense) που χρησιμοποιούν πολλοί αλγόριθμοι WSD για τις περιπτώσεις που δεν μπορούν να επιλύσουν την ασάφεια του όρου.

Καθώς στην πορεία της εργασίας δοκιμάστηκαν επιπλέον παραλλαγές του βασικού αλγορίθμου Lesk, όλα τα αποτελέσματα αποθηκεύθηκαν στη ΒΔ σε επιπλέον στήλες του πίνακα Citations με παρόμοια δομή με τη στήλη SenseCodeUsingLesk. Για την στήλη SenseCodeUsingLeskSentence συγκρίνουμε τη στήλη UMLSDefinition του πίνακα UMLSSense με τη στήλη Sentence του πίνακα CITATIONS. Τέλος, για την στήλη SenseCodeUsingLeskNAMEWithTypeFull συγκρίνουμε τις στήλες UMLSName και UMLSTypeFull του πίνακα UMLSSense με τη στήλη Sentence του πίνακα CITATIONS.

6.2 ΚΑΤΑΧΩΡΗΣΗ ΤΩΝ ΔΕΔΟΜΕΝΩΝ UMLS ΣΤΟΥΣ ΠΙΝΑΚΕΣ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Αρχικά, έφτιαξα ένα project που το ονόμασα inserts και το οποίο έχει δύο πακέτα κώδικα.

Το πρώτο πακέτο που ονομάζεται DBConnection περιέχει μόνο μία κλάση, την κλάση DBConnection.java. Η κλάση αυτή έχει 4 μεθόδους.



Εικόνα 5. Η κλάση DBConnection από το project inserts.

Η πρώτη μέθοδος είναι η connect() μία void μέθοδος με την οποία συνδεόμαστε με τη βάση με χρήση JDBC driver για MySQL.

```
Class.forName("com.mysql.jdbc.Driver");
String url = "jdbc:mysql://galaxy.hua.gr:3306/it34";
Connection con = DriverManager.getConnection(url, "username", "password");
```

Η δεύτερη μέθοδος είναι η closeConnection(), άλλη μια void μέθοδος με την οποία γίνεται το κλείσιμο της σύνδεσης με τη βάση.

```
con.close();
```

Η Τρίτη μέθοδος είναι η addUmlsSense(umlssense umls) που παίρνει ένα όρισμα και είναι τύπου Boolean. Μας επιστρέφει αν έτρεξε σωστά ή όχι η μέθοδος. Αυτή η κλάση καταχωρεί στον πίνακα UMLSSense της βάσης δεδομένων που έχουμε κάθε φορά μία διαφορετική έννοια κάθε όρου από τους 50. Αυτό το κάνει παίρνοντας τα στοιχεία που χρειαζόμαστε από την αντίστοιχη arraylist που δίνεται σαν όρισμα.

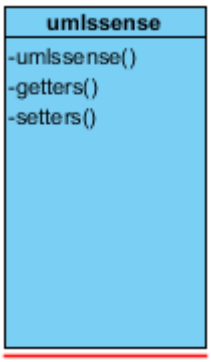
```
public boolean addUmlsSense(umlssense umls) {
      boolean success = false;
      try {
         Statement stmt;
         stmt = (Statement) this.con.createStatement();
         String sql = "insert into it34.UMLSSENSE values('" + umls.getTerm() + "','" +
umls.getSenseCode() + "','" + umls.getUmlsName() + "','" + umls.getUmlsTypeFull() + "','" + null +
"','" + umls.getUmlsTypeShort() + "');";
         success = stmt.execute(sql);
      } catch (SQLException ex) {
         Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
      }
      return success;
   }
```

Τέλος, η τέταρτη μέθοδος που έχουμε είναι η addCitation(Citations citation) που είναι ίδια με την προηγούμενη μόνο που αυτή η μέθοδος καταχωρεί κάθε φορά μία περίπτωση που εμφανίζεται ένας όρος από τις 5000 στον πίνακα CITATIONS της βάσης δεδομένων. Και αυτή η κλάση όπως και η προηγούμενη είναι τύπου Boolean και το αποτέλεσμα του ελέγχου που γίνεται μας επιστρέφεται στο τέλος. Ακόμη, σε αυτή τη μέθοδο ελέγχουμε τις περιπτώσεις που παίρνουμε από την arraylist γιατί σε μερικές δεν έχει δοθεί ο κωδικός της σωστής έννοιας από τους ερευνητές για κάποιο λόγο, με αποτέλεσμα να έχουμε πρόβλημα. Έτσι, γίνεται έλεγχος και σε περίπτωση που δεν έχουμε αποτέλεσμα, δηλαδή το SenseCode είναι UNDEF ή None, αντικαθιστούμε το περιεχόμενο του SenseCode με την τιμή “null”.

```
public boolean addCitation(Citations citation) {
      boolean success = false;
      if (citation.getSenceCode().equals("UNDEF")) {
         String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "',null,'" +
citation.getDocIdFoundPosit() + "','" + citation.getSenceCodeUsingLesk() + "');";
         try {
            Statement stmt;
            stmt = (Statement) this.con.createStatement();
            success = stmt.execute(sql);
         } catch (SQLException ex) {
            Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
         }
      } else if (citation.getSenceCode().equals("None")) {
         String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "',null,'" +
citation.getDocIdFoundPosit() + "','" + citation.getSenceCodeUsingLesk() + "');";
         try {
            Statement stmt;
            stmt = (Statement) this.con.createStatement();
            success = stmt.execute(sql);
         } catch (SQLException ex) {
            Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
         }
      } else {
         String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "','" +
citation.getSenceCode() + "','" + citation.getDocIdFoundPosit() + "','" +
citation.getSenceCodeUsingLesk() + "');";
         try {
            Statement stmt;
            stmt = (Statement) this.con.createStatement();
            success = stmt.execute(sql);
```

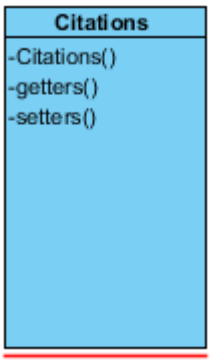
```
        } catch (SQLException ex) {
            Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    return success;
}
```

Το δεύτερο πακέτο που ονομάζεται inserts περιέχει 3 κλάσεις. Την Main.java, την Citations.java και την umlssense.java. Η κλάση umlssense.java περιέχει String μεταβλητές για όλα τα στοιχεία που θέλουμε να αποθηκεύσουμε στον αντίστοιχο πίνακα UMLSSense της βάσης μας. Ένα constructor και getters-setters για όλες τις μεταβλητές είναι οι μέθοδοι που έχουμε σε αυτή την κλάση ώστε να μπορούμε να παίρνουμε τα στοιχεία από τα αρχεία και να τα αποθηκεύουμε στις αντίστοιχες arraylist.



Εικόνα 6. Η κλάση umlssense από το project inserts

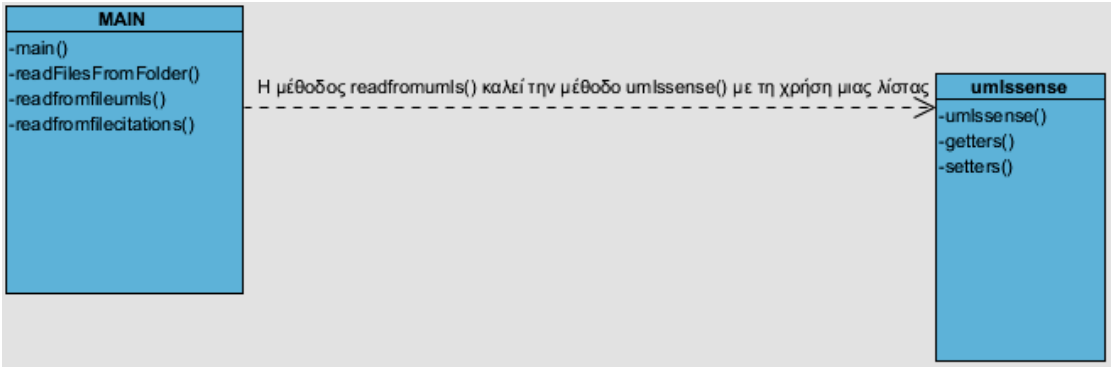
Η κλάση Citations.java περιέχει και αυτή String μεταβλητές για όλα τα στοιχεία που θα αποθηκεύσουμε στον πίνακα CITATIONS της βάσης δεδομένων που δημιουργήσαμε. Ακόμη, έχει και αυτή όπως και η προηγούμενη κλάση (umlssense.java) ένα constructor και getters-setters για όλες τις μεταβλητές.



Εικόνα 7. Η κλάση Citations από το project inserts

Η κλάση Main.java είναι η κύρια κλάση μας, αυτή που τρέχει το πρόγραμμα και που καλεί όλες τις άλλες. Αρχικά, δημιουργούμε δύο arraylists που η μία αντιστοιχεί στην κλάση Citations.java και η άλλη στην κλάση umlssense.java. Έπειτα, έχουμε τις μεθόδους οι οποίες είναι τέσσερις. Η κύρια μέθοδός μας main(String[] args), η μέθοδος readFilesFromFolder(File file), η readfromfileumls(String s) και η readfromfilecitations(String s).

Η μέθοδος readfromfileumls(String s) επεξεργάζεται τις πληροφορίες που μας δίνονται για τις διαφορετικές έννοιες κάθε όρου και τις αποθηκεύει στην arraylist που αντιστοιχεί στην κλάση umlssense.java παίρνοντας σαν όρισμα το όνομα του αρχείου.

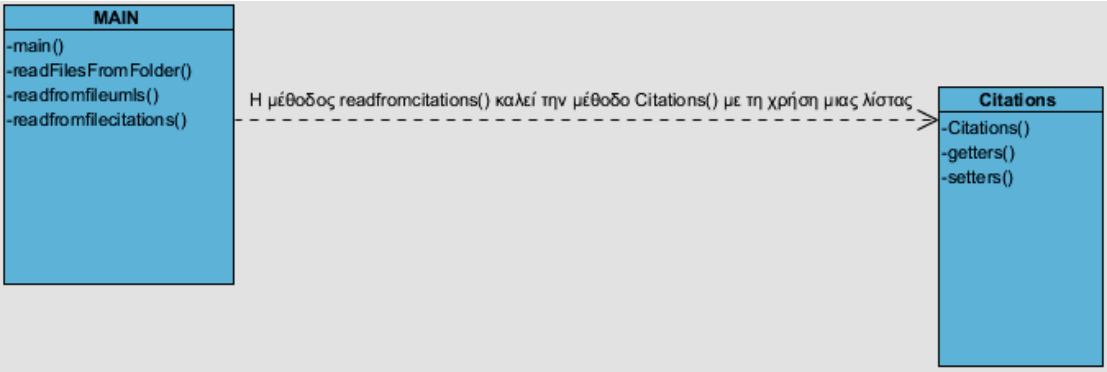


Εικόνα 8. Η μέθοδος readfromfileumls() της κλάσης MAIN και η μέθοδος που καλεί.

```
public static void readfromfileumls(String s) throws IOException {
    BufferedReader in = null;

    try {
       in = new BufferedReader(new FileReader(s));
       String str;
       while ((str = in.readLine()) != null) {
          StringTokenizer st = new StringTokenizer(str, "||");
          while (st.hasMoreTokens()) {
```

```
                String Sencode = st.nextToken();
                String T = st.nextToken();
                String Term = null;
                String num = null;

                String[] spli = T.split(" \\(", 2);
                spli[1] = (String) spli[1].subSequence(0, spli[1].length() - 1);
                String[] spli2 = T.split(" ");

                Term = spli2[0] + " " + spli2[1];
                num = "";

                if (spli2[1].contains("<")) {
                   Term = spli2[0];
                }

                String umlsname = spli[1];
                String nexttoken = st.nextToken();
                String[] tokens = nexttoken.split("[,|;]");

                String typeshort = null;
                String typefull = null;
                if (tokens.length > 2) {
                   typeshort = tokens[0] + tokens[2];
                   typefull = tokens[1] + tokens[3];
                } else {
                   typeshort = tokens[0];
                   typefull = tokens[1];
                }
                int commapos=Term.indexOf(",");
                if (commapos>=0) {
                   Term=Term.substring(0, commapos);
                }
                umlssense u = new umlssense(Term, Sencode, umlsname, typeshort, typefull, "", num);
                umls.add(u);
             }
          }
       } catch (FileNotFoundException e) {
          e.printStackTrace();
       } catch (IOException e) {
          e.printStackTrace();
       } finally {
          if (in != null) {
             in.close();
          }
       }
    }
```

Η μέθοδος readfromfilecitations(String s) επεξεργάζεται και αυτή τις πληροφορίες που μας δίνονται για κάθε περίπτωση εμφάνισης διφορούμενου όρου και τις αποθηκεύει στην arraylist που αντιστοιχεί στην κλάση Citations.java παίρνοντας σαν όρισμα το όνομα του αρχείου.



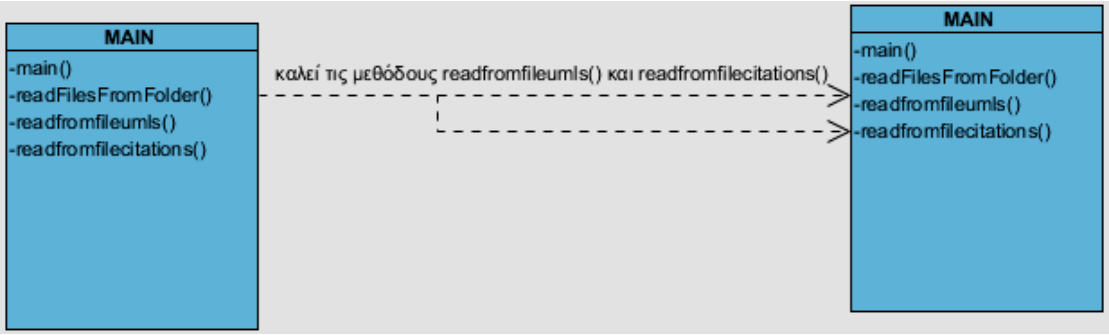
Εικόνα 9. Η μέθοδος readfromfilecitations() της κλάσης MAIN και η μέθοδος που καλεί.

```
public static void readfromfilecitations(String s) throws IOException {
      BufferedReader in = null;
      String term = null;
      String header = null;
      String pos = null;
      String sencode = null;
      String ui = null;
      String ti = null;
      String ab = null;
      String found = null;
      String DocIdPF = null;

      try {
         in = new BufferedReader(new FileReader(s));
         String str;
         String temp = null;

         while ((str = in.readLine()) != null) {
            if (str.equals("")) {
               Citations c = new Citations(term, ui, ti, header, ab, found, pos, sencode, DocIdPF, null);
               citat.add(c);
            } else {
               StringTokenizer st = new StringTokenizer(str, "");
               while (st.hasMoreTokens()) {
                  temp = st.nextToken();
                  temp = temp.trim();

                  if (temp.startsWith("UI")) {
```

```
                    int splitPOS = temp.indexOf(" - ");
                    ui = temp.substring(splitPOS + 3);
                } else if (temp.startsWith("TI")) {
                    int splitPOS = temp.indexOf(" - ");
                    ti = temp.substring(splitPOS + 3);
                    ti = ti.replaceAll("""", "\"");
                } else if (temp.startsWith("AB")) {
                    int splitPOS = temp.indexOf(" - ");
                    ab = temp.substring(splitPOS + 3);
                    ab = ab.replaceAll("""", "\"");
                } else {
                    if (temp.contains("|")) {
                        String[] tokens = temp.split("\\|");
                        if (tokens.length < 5) {
                            sencode = tokens[2];
                            DocIdPF = tokens[1];
                            String[] tokens2 = tokens[1].split("\\.");
                            pos = tokens2[2];
                            found = tokens2[1];
                            if (found.equals("ab")) {
                                found = "1";
                            } else if (found.equals("ti")) {
                                found = "0";
                            }
                        } else {
                            String[] tokens3 = temp.split("\\|");
                            term = tokens3[0];
                            term = term.replaceAll("_", " ");
                        }
                    } else {
                        header = temp;
                        header = header.replaceAll("""", "\"");
                    }
                }
            }
        }
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        if (in != null) {
            in.close();
        }
    }
}
```

Η μέθοδος readFilesFromFolder(File file) δέχεται σαν όρισμα ένα όνομα φακέλου, ο οποίος περιέχει όλα τα απαραίτητα αρχεία, και αφού κάνει τους απαραίτητους

ελέγχους, καλεί τις δύο προηγούμενες μεθόδους (readfromfilecitationsκαι readfromfileumls) ώστε να αποθηκευτούν όλες οι πληροφορίες στις αντίστοιχες arraylists.

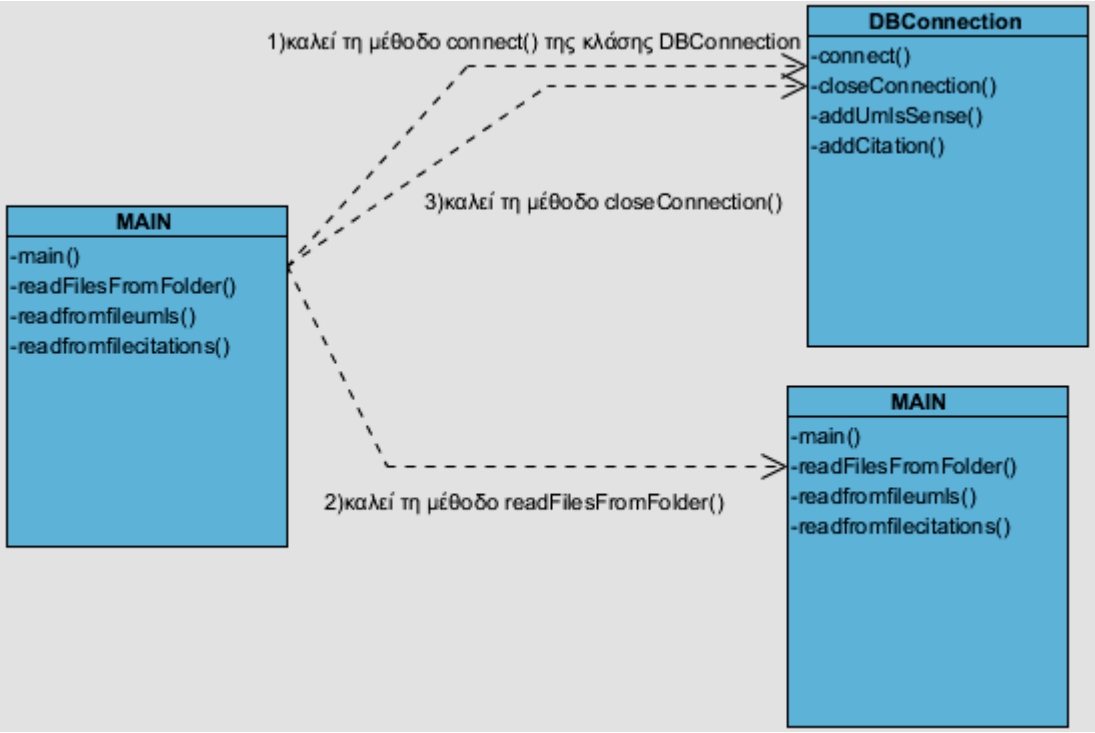


Εικόνα 10. Η μέθοδος readFilesFromFolder() και οι μέθοδοι που καλεί.

```
public static void readFilesFromFolder(File file) throws IOException {
      boolean isDirectory = file.isDirectory();
      if (isDirectory) {
         File[] files = file.listFiles();
         for (int i = 0; i < files.length; i++) {
            File folders50 = files[i];
            if (folders50.isDirectory()) {
               File[] filesInEachfolder = folders50.listFiles();
               for (int j = 0; j < filesInEachfolder.length; j++) {
                  File f = filesInEachfolder[j];

                  if (f.getName().equals("choices")) {
                     String s = f + "";
                     readfromfileumls(s);
                  } else if (f.getName().endsWith("_set")) {
                     String s1 = f + "";
                     readfromfilecitations(s1);
                  }
               }
            }
         }
      }
   }
```

Τέλος, η κύρια μέθοδος main(String[] args) καλεί τη μέθοδο connect() της κλάσης DBConnection.java για να γίνει η σύνδεση με τη βάση. Έπειτα, καλεί την μέθοδο readFilesFromFolder(File file) δίνοντας της για όρισμα τον κατάλληλο φάκελο. Αφού αποθηκευτούν οι πληροφορίες καλεί τις μεθόδους addUmlsSense και addCitation της κλάσης DBConnection.java όσες φορές χρειάζεται (με μία δομή

επανάληψης) ώστε να περάσουν στη βάση όλες οι πληροφορίες. Τέλος, κλείνει τη σύνδεση καλώντας την αντίστοιχη μέθοδο (closeConnection) της κλάσης DBConnection.java.



Εικόνα 11. Η κύρια μέθοδος main() και οι μέθοδοι που καλεί.

```
public static void main(String[] args) throws IOException {
      DBConnection db = new DBConnection();
      db.connect();
      File file = new File("Non-Reviewed_Results");
      readFilesFromFolder(file);

      for (umlssense u : umls) {
         db.addUmlsSense(u);
      }
      for (Citations c : citat) {
         db.addCitation(c);
      }
      db.closeConnection();
   }
```

6.3 ΚΑΤΑΧΩΡΗΣΗ ΤΩΝ DEFINITIONS

Μετά την επιτυχή καταχώρηση των πληροφοριών που πήραμε από το UMLS στη βάση δεδομένων μας. Πρέπει να γεμίσουμε τη στήλη UMLSDefinition του πίνακα UMLSSense με τους αντίστοιχους ορισμούς που υπάρχουν στη βάση δεδομένων του UMLS. Για να πάρουμε τους ορισμούς αποθηκεύσαμε 6 πίνακες από τη βάση δεδομένων του UMLS στη δικιά μας βάση. Αυτοί οι πίνακες είναι οι εξής: mrconso, mrdef, mrsty, mrxns_eng, mrxnw_eng και mrxw_eng. Με βάση αυτούς τους πίνακες παίρνουμε τον ορισμό κάθε έννοιας ενός βιοϊατρικού όρου κάνοντας αναζήτηση με τα κατάλληλα Queries.

Αυτό γίνεται με το πρόγραμμα FillDatabase. Αυτό το πρόγραμμα έχει δύο πακέτα κώδικα, το πρώτο ονομάζεται DBConnection και το δεύτερο πακέτο κώδικα ονομάζεται FillDatabase.

Το πακέτο DBConnection περιέχει μία κλάση, τη DBConnection.java με την οποία γίνεται όλες οι εργασίες με τη βάση.

Εικόνα 12. Η κλάση DBConnection του project FillDatabase

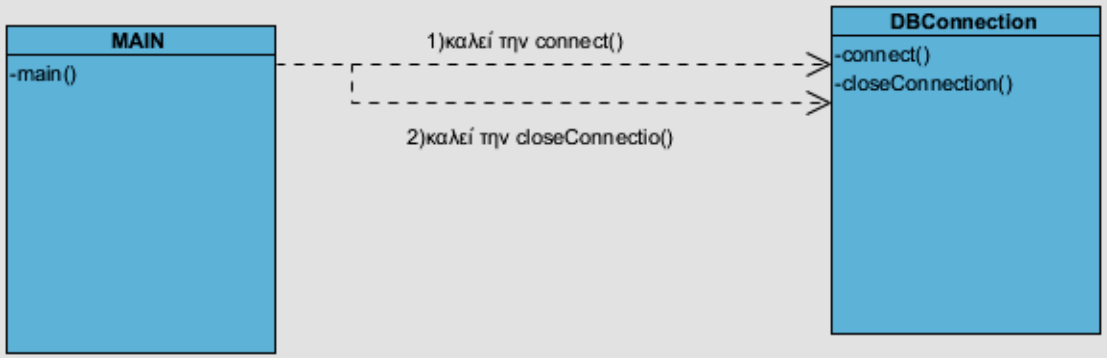
Η κλάση αυτή έχει δύο μεθόδους, τις connect() και closeConnection(). Η μέθοδος connect() κάνει τη σύνδεση με τη βάση.

```
Class.forName("com.mysql.jdbc.Driver");
        String url = "jdbc:mysql://galaxy.hua.gr:3306/it34";
        Connection con = DriverManager.getConnection(url, "username", "password");
```

Και η μέθοδος closeConnection() κλείνει τη σύνδεση με τη βάση.

```
con.close();
```

Το δεύτερο πακέτο, δηλαδή το FillDatabase, περιέχει 1 κλάση η οποία λέγεται Main.java. Η Main.java είναι η κλάση από την οποία τρέχουμε ο πρόγραμμα. Έχει μόνο μία μέθοδο τη κύρια μέθοδο main().



Εικόνα 13. Η μέθοδος main() της κλάσης MAIN και οι μέθοδοι που καλεί.

Αρχικά, καλούμε τη μέθοδο connect() της κλάσης DBConnection.java για να γίνει η σύνδεση με τη βάση. Έπειτα, εμφανίζουμε όλες τις γραμμές του πίνακα UMLSSense με τις παρακάτω εντολές

```
String QUERY = "select Term, UMLSTypeFull,UMLSName,Sensecode,UMLSDefinition from it34.UMLSSENSE where UMLSDefinition='null';";

java.sql.Statement stmt1 = conn1.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_UPDATABLE);

ResultSet rs = stmt1.executeQuery(QUERY);
```

Και διαβάζουμε κάθε γραμμή του πίνακα με μία while. Εκτελούμε τρία Queries τα οποία διαβάζουν και τους έξι πίνακες της UMLS βάσης δεδομένων και τα αποτελέσματα τους τα αποθηκεύουμε προσωρινά σε μία μεταβλητή String ώστε με τη σειρά της να τα περάσει στο αντίστοιχο πεδίο της στήλης UMLSDefinition του πίνακα UMLSSense με εντολή update. Τέλος, καλούμε τη μέθοδο closeConnection() της κλάσης DBConnection.java για να κλείσει η σύνδεση με τη βάση.

```
while (rs.next()) {
        String Query2 = "SELECT DISTINCT r.def FROM mrsty s JOIN mrxns_eng l ON (s.cui=l.cui) " + "JOIN mrdef r ON (r.cui=l.cui) " + "WHERE s.sty='" + rs.getString(2).trim() + "' AND nstr='" + rs.getString(1) + "';";
```

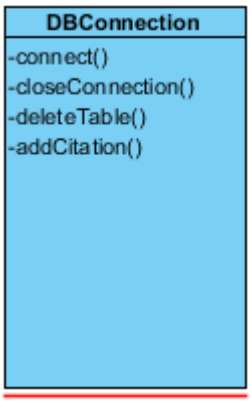
```
            String Query3 = "SELECT DISTINCT r.def FROM mrsty s JOIN mrxnw_eng l ON
(s.cui=l.cui) "  + "JOIN mrdef r ON (r.cui=l.cui) " + "WHERE s.sty='" + rs.getString(2).trim() + "'
AND NWD='" + rs.getString(1) + "';";
            String Query4 = "SELECT DISTINCT r.def FROM mrsty s JOIN mrxw_eng l ON
(s.cui=l.cui) "  + "JOIN mrdef r ON (r.cui=l.cui) "  + "WHERE s.sty='" + rs.getString(2).trim() + "'
AND WD='" + rs.getString(1) + "';";

            String def = "";
            ResultSet rs2 = stmt2.executeQuery(Query2);
            while (rs2.next()) {
               def += " " + rs2.getString(1);
            }
            rs2 = stmt2.executeQuery(Query3);
            while (rs2.next()) {
               def += " " + rs2.getString(1);
            }
            rs2 = stmt2.executeQuery(Query4);
            while (rs2.next()) {
               def += " " + rs2.getString(1);
            }
            rs.updateString(5, def);
            rs.updateRow();
         }
      db.closeConnection();
   }
```

6.4 ΕΝΑΛΛΑΚΤΙΚΟΙ ΤΡΟΠΟΙ ΜΕΤΑΤΡΟΠΗΣ ΤΟΥ ΛΗΜΜΑΤΟΣ ‘Η ΤΟΥ ΚΕΙΜΕΝΟΥ ΣΕ ΣΥΝΟΛΟ ΛΕΞΕΩΝ

Ο αλγόριθμος Lesk που χρησιμοποιώ βασίζεται στον απλοποιημένο αλγόριθμο Lesk με μερικές διαφορές. Αρχικά, στον αλγόριθμο που έφτιαξα έχω 4 κλάσεις σε δύο πακέτα κώδικα. Το ένα πακέτο λέγεται lesk και περιέχει 3 κλάσεις, τις Citations.java, umlssense.java και τη Main.java που είναι η κλάση από την οποία

τρέχει το πρόγραμμα. Στο δεύτερο πακέτο, το οποίο ονομάζεται DBConnection, έχουμε την κλάση DBConnection.java στην οποία γίνεται η σύνδεση με τη βάση.

Η κλάση DBConnection.java έχει τέσσερις μεθόδους. Τη μέθοδο connect(), την closeConnection(), την deleteTable() και την addCitation(Citations citation).



Εικόνα 14. Η κλάση DBConnection του project Lesk.

Η μέθοδος connect() συνδέει το πρόγραμμα με τη βάση όπως και στα άλλα δύο προγράμματα που είδαμε παραπάνω.

```
Class.forName("com.mysql.jdbc.Driver");
        String url = "jdbc:mysql://galaxy.hua.gr:3306/it34";
        Connection con = DriverManager.getConnection(url, "username", "password");
```

Η μέθοδος closeConnection() κλείνει τη σύνδεση με τη βάση.

```
con.close();
```

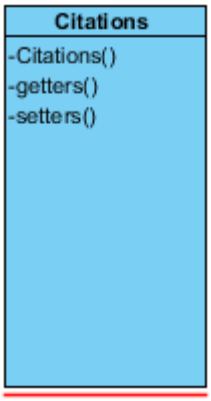
Η μέθοδος deleteTable() διαγράφει τον πίνακα CITATIONS.

```
String sql = "DELETE FROM it34.CITATIONS";
Statement stmt = (Statement) this.con.createStatement();
stmt.execute(sql);
```

Η τελευταία μέθοδος της κλάσης αυτής είναι η addCitation(Citations citation) η οποία παίρνει σαν όρισμα μία γραμμή της arraylist Citations και ελέγχει την τιμή που έχει το SenceCode. Ανάλογα με την τιμή του με τη βοήθεια μιας δομής επιλογής (if-else) γίνεται έλεγχος και σε κάθε περίπτωση περνάει στη βάση όλα τα στοιχεία της γραμμής.

```
if (citation.getSenceCode()==null || citation.getSenceCode().equals("UNDEF")) {
        String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "',null,'" +
citation.getDocIdFoundPosit() + "','" + citation.getSenceCodeUsingLesk() + "','"+
citation.getSenceCodeUsingLeskSentence() + "','" +
citation.getSenceCodeUsingLeskNameWithTypeFull() + "');";
        try {
           Statement stmt;
           stmt = (Statement) this.con.createStatement();
           success = stmt.execute(sql);
        } catch (SQLException ex) {
           Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
        }
     } else if (citation.getSenceCode()==null || citation.getSenceCode().equals("None")) {
        String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "',null,'" +
citation.getDocIdFoundPosit() + "','" + citation.getSenceCodeUsingLesk() + "','"+
citation.getSenceCodeUsingLeskSentence() + "','" +
citation.getSenceCodeUsingLeskNameWithTypeFull() + "');";
        try {
           Statement stmt;
           stmt = (Statement) this.con.createStatement();
           success = stmt.execute(sql);
        } catch (SQLException ex) {
           Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
        }
     } else {
        String sql = "insert into it34.CITATIONS values('" + citation.getTerm() + "','" +
citation.getDocid() + "','" + citation.getTitle() + "','" + citation.getSentence() + "','" +
citation.getAbstract() + "','" + citation.getFoundin() + "','" + citation.getPosition() + "','" +
citation.getSenceCode() + "','" + citation.getDocIdFoundPosit() + "','" +
citation.getSenceCodeUsingLesk() + "','"+ citation.getSenceCodeUsingLeskSentence() + "','" +
citation.getSenceCodeUsingLeskNameWithTypeFull() + "');";
        try {
           Statement stmt;
           stmt = (Statement) this.con.createStatement();
           success = stmt.execute(sql);
        } catch (SQLException ex) {
           Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
        }
     }
```

Η πρώτη κλάση από το πρώτο πακέτο lesk είναι η Citations.java η οποία περιέχει String μεταβλητές για όλα τα στοιχεία του πίνακα CITATIONS της βάσης δεδομένων μας. Γι' αυτές τις μεταβλητές υπάρχουν getters και setters όπως και ένας constructor.

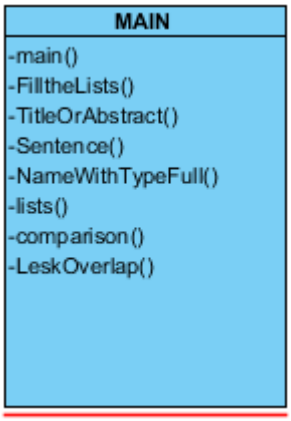


Εικόνα 15. Η κλάση Citations του project Lesk

Αντίστοιχα για τον πίνακα UMLSSense της βάσης δεδομένων υπάρχει η κλάση umlssense.java η οποία περιέχει όλα τα στοιχεία του πίνακα. Ακόμη, έχει και αυτή έναν constructor και getters-setters για όλες τις μεταβλητές.

Εικόνα 16. Η κλάση umlssense του project Lesk

Τέλος, η κλάση Main.java είναι η κύρια κλάση αφού με αυτή τρέχει το πρόγραμμα. Έχει 8 μεθόδους.



Εικόνα 17. Η κλάση Main του project Lesk

6.4.1 ΥΛΟΠΟΙΗΣΗ ΤΟΥ LESK ΜΕ ΔΙΠΛΟΤΥΠΑ

Αρχικά, δημιουργούμε 2 λίστες που θα αποθηκεύσουμε τα στοιχεία από τους πίνακες UMLSSense και CITATIONS της βάσης δεδομένων. Άλλους 2 που θα περιέχουν τις λέξεις που θα συγκρίνουμε και άλλη μία λίστα (stopwords) που θα περιέχει όλες τις κοινές λέξεις που βρίσκουμε σε κείμενα.

Η μέθοδος FilltheLists() συνδέεται με τη βάση και διαβάζει τους πίνακες UMLSSense και CITATION από τη βάση δεδομένων με δύο Queries. Έπειτα αποθηκεύει μία μία τις σειρές από τους πίνακες στις αντίστοιχες λίστες(arraylists).

```
String query = "SELECT * FROM it34.CITATIONS";
String query2 = "SELECT * FROM it34.UMLSSENSE";

 while (rs.next()) {
        Citations c = new Citations(rs.getString(1), rs.getString(2), rs.getString(3), rs.getString(4), rs.getString(5), rs.getString(6), rs.getString(7), rs.getString(8), rs.getString(9), rs.getString(10), "", "");

        citation.add(c);
}

 while (rs1.next()) {
        umlssense u = new umlssense(rs1.getString(1), rs1.getString(2), rs1.getString(3), rs1.getString(6), rs1.getString(4), rs1.getString(5));

        umlsse.add(u);
}
```

Η μέθοδος lists(String test, ArrayList l), η οποία παίρνει σαν ορίσματα μία μεταβλητή String και μία λίστα, χωρίζει την πρόταση που είναι αποθηκευμένη στην μεταβλητή String σε λέξεις και τις αποθηκεύει μία μία στη λίστα.

```
public static void lists(String test, ArrayList l) {
    StringTokenizer st = new StringTokenizer(test);
    while (st.hasMoreTokens()) {
      l.add(st.nextToken().toLowerCase());
    }
  }
```

Η μέθοδος comparison(ArrayList l1, ArrayList l2) δέχεται σαν ορίσματα δύο λίστες και τις συγκρίνει μεταξύ τους αφαιρώντας από τον πρώτο πίνακα τις κοινές τους λέξεις. Αυτή τη μέθοδο τη χρησιμοποιούμε για να αφαιρέσουμε από τη λίστα όλες τις συχνά εμφανιζόμενες λέξεις(stopwords).

```
public static void comparison(ArrayList l1, ArrayList l2) {
    l1.removeAll(l2);
}
```

Η μέθοδος LeskOverlap(ArrayList l1, ArrayList l2) δέχεται σαν ορίσματα δύο λίστες και είναι μέθοδος τύπου double. Αρχικοποιούμε ένα μετρητή και ελέγχουμε τις δύο αυτές λίστες για κοινές λέξεις. Σε περίπτωση που βρεθεί κοινή λέξη τότε αυξάνουμε τον μετρητή κατά ένα. Τέλος διαιρούμε τον τελικό αριθμό του μετρητή με το γινόμενο των μεγεθών των δύο λιστών και επιστρέφουμε το αποτέλεσμα.

```
public static double LeskOverlap(ArrayList l1, ArrayList l2) {
    double doub = 0;
    int sum = 0;
    for (int i = 0; i < l1.size(); i++) {
        for (int j = 0; j < l2.size(); j++) {
            if (l1.get(i).equals(l2.get(j))) {
                sum++;
            }
        }
    }
    doub = sum / (double) ((l1.size()) * (l2.size()));
    return doub;
}
```

6.4.2 ΥΛΟΠΟΙΗΣΗ ΤΟΥ LESK ΧΩΡΙΣ ΔΙΠΛΟΤΥΠΑ

Αρχικά, δημιουργούμε 2 λίστες που θα αποθηκεύσουμε τα στοιχεία από τους πίνακες UMLSSense και CITATIONS της βάσης δεδομένων όπως ακριβώς και στην προηγούμενη περίπτωση. Η αλλαγή εδώ είναι ότι αντί να δημιουργήσουμε άλλες δύο λίστες που θα περιέχουν τις λέξεις που θα συγκρίνουμε, δημιουργούμε 2 συλλογές (HashSet) που δεν επιτρέπουν την καταχώρηση ίδιων λέξεων ώστε να μην έχουμε διπλότυπα. Τέλος, δημιουργούμε άλλη μία λίστα (stopwords) που θα περιέχει όλες τις κοινές λέξεις που βρίσκουμε σε κείμενα.

Οι αλλαγές που υπάρχουν σε αυτή την υλοποίηση με βάση την προηγούμενη (6.4.1) είναι στις μεθόδους lists(), comparison() και LeskOverlap(). Στη μέθοδο lists() παίρνουμε σαν όρισμα μία μεταβλητή String και ένα HashSet.

```
public static void lists(String test, HashSet l) {
      StringTokenizer st = new StringTokenizer(test);
      while (st.hasMoreTokens()) {
         l.add(st.nextToken().toLowerCase());
      }
   }
```

Στη μέθοδο comparison() παίρνουμε σαν όρισμα μία συλλογή HashSet και μία λίστα ArrayList.

```
public static void comparison(HashSet l1, ArrayList l2) {
      l1.removeAll(l2);
   }
```

Και στη μέθοδο LeskOverlap() αντί να έχουμε για ορίσματα 2 ArrayLists έχουμε 2 HashSets. Επειδή, τα περιεχόμενα της συλλογής HashSet δεν είναι αριθμημένα δεν έχουμε την ίδια δομή επανάληψης όπως για μία ArrayList.

```
public static double LeskOverlap(HashSet<String> l1, HashSet<String> l2) {
      double doub = 0;
      int sum = 0;
      for (String x:l1) {
         for (String y:l2) {
            if (x.equals(y)) {
               sum++;
            }
         }
      }
      doub = sum / (double) ((l1.size()) * (l2.size()));
      return doub;
   }
```

6.4.3 ΥΛΟΠΟΙΗΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ LESK ΜΕ ΑΠΟΚΟΠΗ ΤΗΣ ΡΙΖΑΣ

Η αποκοπή της ρίζας από την λέξη (Stemming) είναι η διαδικασία για τη μείωση των κλιτών λέξεων και η διατήρηση μόνο της βάσης της. Η βάση δεν χρειάζεται να είναι ταυτόσημη με τη μορφολογική ρίζα της λέξης. Συνήθως αρκεί αν σχετικές λέξεις καθορίζουν τη βάση, ακόμη και αν αυτή η βάση δεν είναι από μόνη

της μία έγκυρη ρίζα. Από το 1968 έχουν κατασκευαστεί αλγόριθμοι που μελετούν την αποκοπή της ρίζας από τη λέξη. Πολλές μηχανές αναζήτησης θεωρούν λέξεις με την ίδια βάση ως συνώνυμα, μία διαδικασία που ονομάζεται conflation (ταύτιση). Στην περίπτωση μας θα χρησιμοποιούσαμε έναν αλγόριθμο αντιστοίχησης (Matching algorithm) ο οποίος χρησιμοποιεί μία βάση δεδομένων με ρίζες λέξεων. Αυτές οι ρίζες δεν είναι απαραίτητα οι ίδιες οι λέξεις. Βρίσκει τη ρίζα από κάθε λέξη που υπάρχει κοντά στην αμφίσημη λέξη και τη συγκρίνει με τον ορισμό της.

6.4.4 ΥΛΟΠΟΙΗΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ LESK ΜΕ ΤΑ ΒΑΡΗ ΤΩΝ ΛΕΞΕΩΝ

Το Tf–idf ή αλλιώς Term frequency–inverse document frequency (συχνότητα του όρου – αντίστροφη συχνότητα εγγράφου) είναι ένα αριθμητικό στατιστικό στοιχείο το οποίο μας δείχνει πόσο σημαντική είναι μία λέξη σε ένα κείμενο. Συχνά χρησιμοποιείται ως ένας συντελεστής βαρύτητας στην ανάκτηση πληροφοριών και κειμένων. Το βάρος κάθε λέξης αυξάνεται αναλογικά με τον αριθμό των φορών που μία λέξη εμφανίζεται στο κείμενο, αλλά αντισταθμίζεται από τη συχνότητα της λέξης, δηλαδή ότι μία λέξη είναι πιο κοινή από ότι οι άλλες. Η υλοποίηση μιας τέτοιας περίπτωσης είναι ενδιαφέρουσα ως προς τα αποτελέσματα που μπορεί να μας δώσει.

6.5 ΕΝΑΛΛΑΚΤΙΚΟ ΠΕΡΙΕΧΟΜΕΝΟ ΣΤΟ ΟΠΟΙΟ ΕΜΦΑΝΙΖΕΤΑΙ Ο ΑΣΑΦΗΣ ΟΡΟΣ

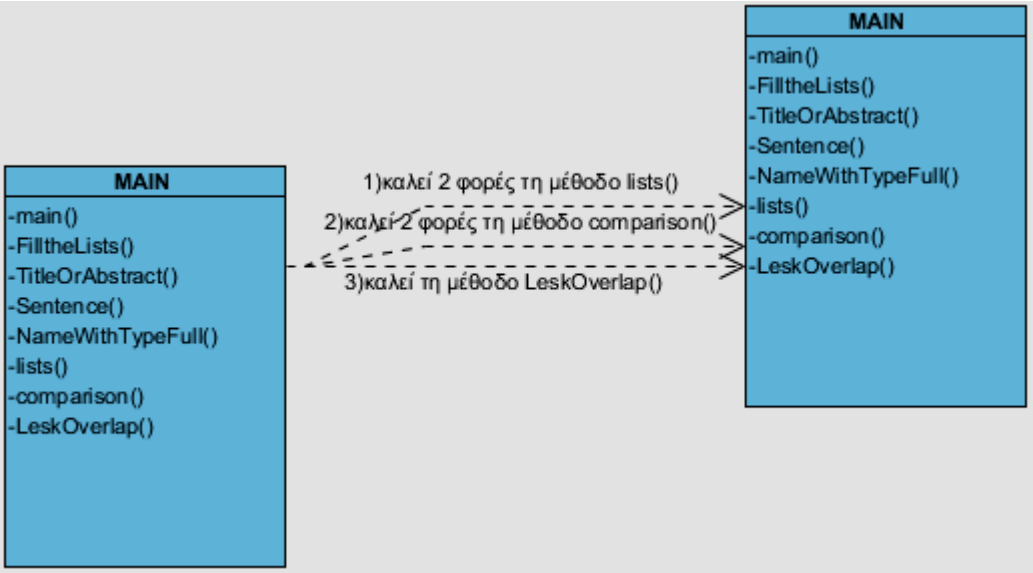
Υπάρχουν 4 τρόποι που μπορούμε να βγάλουμε αποτελέσματα ώστε να τα συγκρίνουμε με αυτά που έχουμε από τη μελέτη του UMLS. Αυτοί είναι οι εξής :

- Σύγκριση της στήλης UMLSDefinition του πίνακα UMLSSense με τη στήλη Title ή τη στήλη ABSTRACT του πίνακα CITATIONS.
- Σύγκριση της στήλης UMLSDefinition του πίνακα UMLSSense με τη στήλη Sentence του πίνακα CITATIONS.
- Σύγκριση του περιεχομένου των στηλών UMLSName και UMLSTypeFull του πίνακα UMLSSense με τη στήλη Sentence του πίνακα CITATIONS.
- Σύγκριση του λήμματος(ορισμού) της έννοιας με τις υπόλοιπες λέξεις της πρότασης ή ακόμα και του κειμένου.

Πρακτικά θα ασχοληθούμε μόνο με τις τρεις πρώτες περιπτώσεις, ενώ την τέταρτη θα την αναλύσουμε μόνο σε θεωρητική βάση.

6.5.1 ΣΥΓΚΡΙΣΗ ΤΟΥ DEFINITION ΜΕ ΤΟΝ ΤΙΤΛΟ 'Η ΤΟ ABSTRACT

Η μέθοδος TitleOrAbstract() συγκρίνει τον ορισμό της έννοιας, δηλαδή το UMLSDefinition του πίνακα UMLSSense, με το κείμενο που βρίσκεται ο διφορούμενος όρος. Αν βρίσκεται στον τίτλο τότε η σύγκριση θα γίνει με το Title, διαφορετικά αν βρίσκεται στην περίληψη η σύγκριση θα γίνει με το ABSTRACT. Το Title και το ABSTRACT είναι στήλες του πίνακα CITATIONS της βάσης δεδομένων που χρησιμοποιούμε. Σε αυτή τη μέθοδο με δύο δομές επανάληψης ελέγχουμε κάθε γραμμή από τις λίστες citation και umlsse (που αντιστοιχούν στα περιεχόμενα των κλάσεων Citations και umlssense). Σε περίπτωση που το όνομα του όρου είναι ίδιο (Term) καλούμε τη μέθοδο lists (που είδαμε παραπάνω) δύο φορές. Μία φορά για το περιεχόμενο της στήλης UMLSDefinition το οποίο θα μπει ως πρώτο όρισμα και ως δεύτερο μία κενή λίστα. Και μία φορά για το περιεχόμενο της στήλης του άλλου πίνακα που θα συγκρίνουμε, το οποίο είναι το πρώτο όρισμα. Δεύτερο όρισμα είναι μία κενή λίστα. Έπειτα, καλούμε δύο φορές τη μέθοδο comparison η οποία θα συγκρίνει τις δύο λίστες που πήραμε ως αποτέλεσμα από τις πάνω μεθόδους με τη λίστα που περιέχει τις συχνά επαναλαμβανόμενες λέξεις. Αφου, αφαιρέσει όλες τις κοινές λέξεις καλούμε τη μέθοδο LeskOverlap η οποία παίρνει σαν ορίσματα τις δύο παραπάνω λίστες και μας επιστρέφει έναν double αριθμό με το αποτέλεσμα της διαίρεσης που είδαμε παραπάνω. Με μία δομή επιλογής ελέγχουμε ποιός αριθμός είναι μεγαλύτερος και τον αποθηκεύουμε στο αντίστοιχο πεδίο της λίστας.

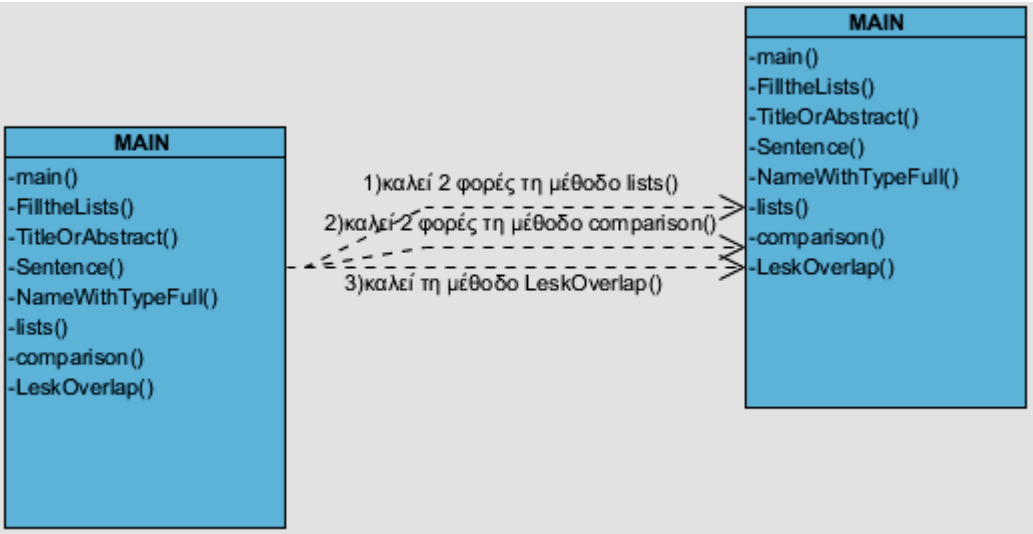


Εικόνα 18. Η μέθοδος TitleOrAbstract() της κλάσης Main και οι μέθοδοι που καλεί.

```
public static void TitleOrAbstract() {

        for (int j = 0; j < citation.size(); j++) {
            if (citation.get(j).foundin.equals("0")) {
                s = citation.get(j).Title;
            } else if (citation.get(j).foundin.equals("1")) {
                s = citation.get(j).Abstract;
            }
            int l = 0;
            double maxvalue = 0;

            for (int i = 0; i < umlsse.size(); i++) {
                list1.clear();
                list2.clear();

                if (citation.get(j).Term.toLowerCase().equals(umlsse.get(i).Term.toLowerCase())) {
                    lists(s, list1);
                    lists(umlsse.get(i).UmlsDefinition, list2);

                    comparison(list1, stopwords);
                    comparison(list2, stopwords);

                    double d = LeskOverlap(list1, list2);

                    if (d > maxvalue) {
                        maxvalue = d;

                        citation.get(j).SenceCodeUsingLesk = umlsse.get(i).SenseCode
                    }
                }
            }
        }
    }
```

6.5.2 ΣΥΓΚΡΙΣΗ ΤΟΥ DEFINITION ΜΕ ΤΗΝ ΠΡΟΤΑΣΗ ΠΟΥ ΒΡΙΣΚΕΤΑΙ Ο ΟΡΟΣ

Οι δύο επόμενοι μέθοδοι, οι Sentence() και NameWithTypeFull() έχουν ίδια δομή με την μέθοδο TitleOrAbstract() μόνο που αλλάζουν οι τιμές που δίνουμε για σύγκριση. Στη μέθοδο Sentence() συγκρίνουμε το περιεχόμενο της στήλης UMLSDefinition του πίνακα UMLSSense, δηλαδή τον ορισμό της έννοιας, με το περιεχόμενο της στήλης Sentence του πίνακα CITATIONS. Η στήλη Sentence περιέχει μόνο την πρόταση στην οποία βρίσκεται ο όρος.



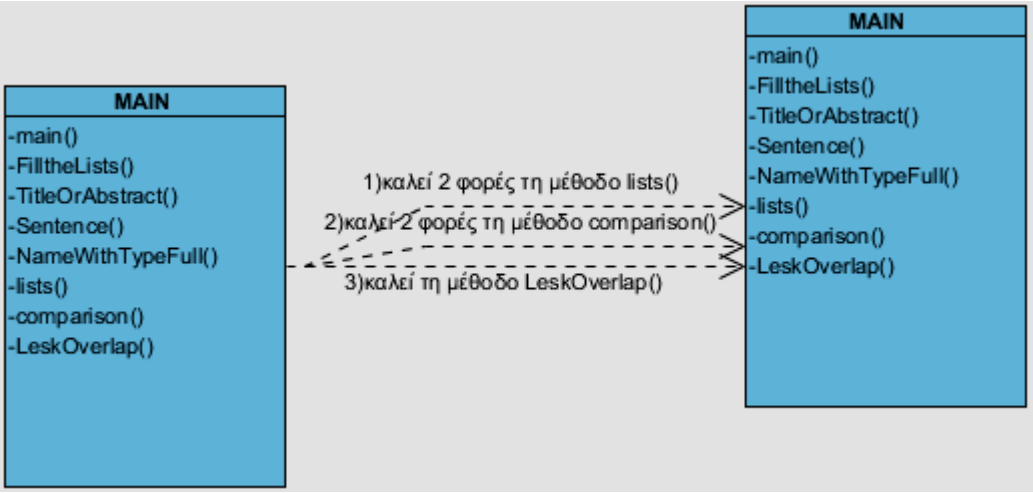
Εικόνα 19. Η μέθοδος Sentence() της κλάσης Main και οι μέθοδοι που καλεί.

```
public static void Sentence() {
      s = "";
      for (int j = 0; j < citation.size(); j++) {
         s = citation.get(j).Sentence;
         int l = 0;
         double maxvalue = 0;

         for (int i = 0; i < umlsse.size(); i++) {
            list1.clear();
            list2.clear();

            if (citation.get(j).Term.toLowerCase().equals(umlsse.get(i).Term.toLowerCase())) {
               lists(s, list1);
               lists(umlsse.get(i).UmlsDefinition, list2);

               comparison(list1, stopwords);
               comparison(list2, stopwords);

               double d = LeskOverlap(list1, list2);
               if (d > maxvalue) {
                  maxvalue = d;
                  citation.get(j).SenceCodeUsingLeskSentence = umlsse.get(i).SenseCode;
```

```
            }
        }
    }
}
}
```

6.5.3 ΣΥΓΚΡΙΣΗ ΤΟΥ ΟΝΟΜΑΤΟΣ ΚΑΙ ΤΟΥ ΤΥΠΟΥ ΤΟΥ ΟΡΟΥ ΜΕ ΤΗΝ ΠΡΟΤΑΣΗ ΣΤΗΝ ΟΠΟΙΑ ΒΡΙΣΚΕΤΑΙ Ο ΟΡΟΣ

Ενώ, στη μέθοδο NameWithTypeFull() συγκρίνουμε το περιεχόμενο της στήλης Sentence του πίνακα CITATIONS με τα περιεχόμενα των στηλών UMLSName και UMLSTypeFull του πίνακα UMLSSense.

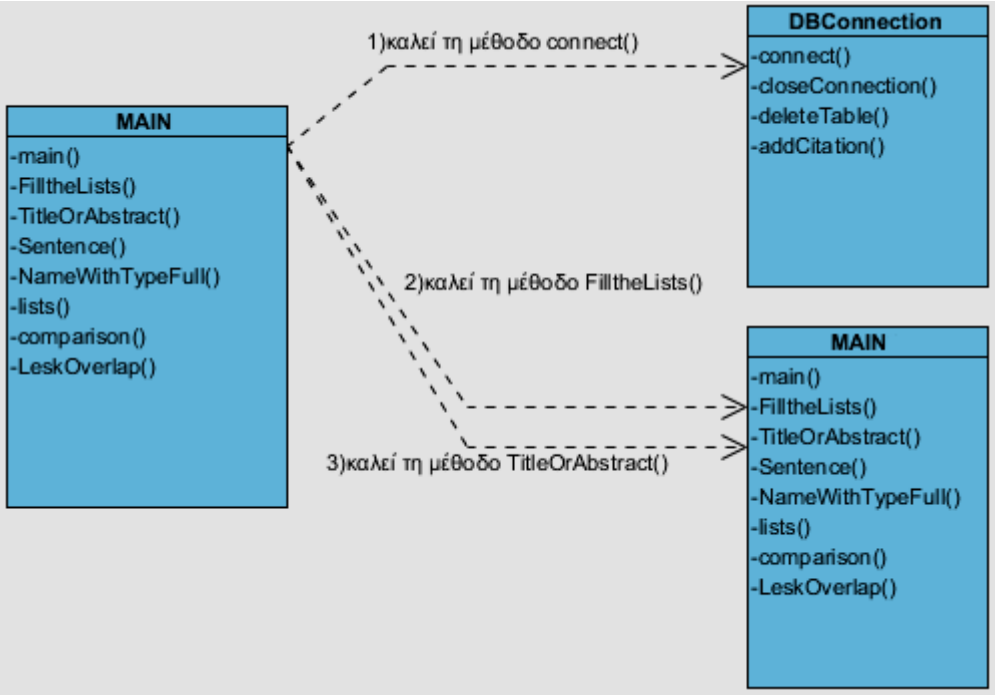


Εικόνα 20. Η μέθοδος NameWithTypeFull() της κλάσης Main και οι μέθοδοι που καλεί.

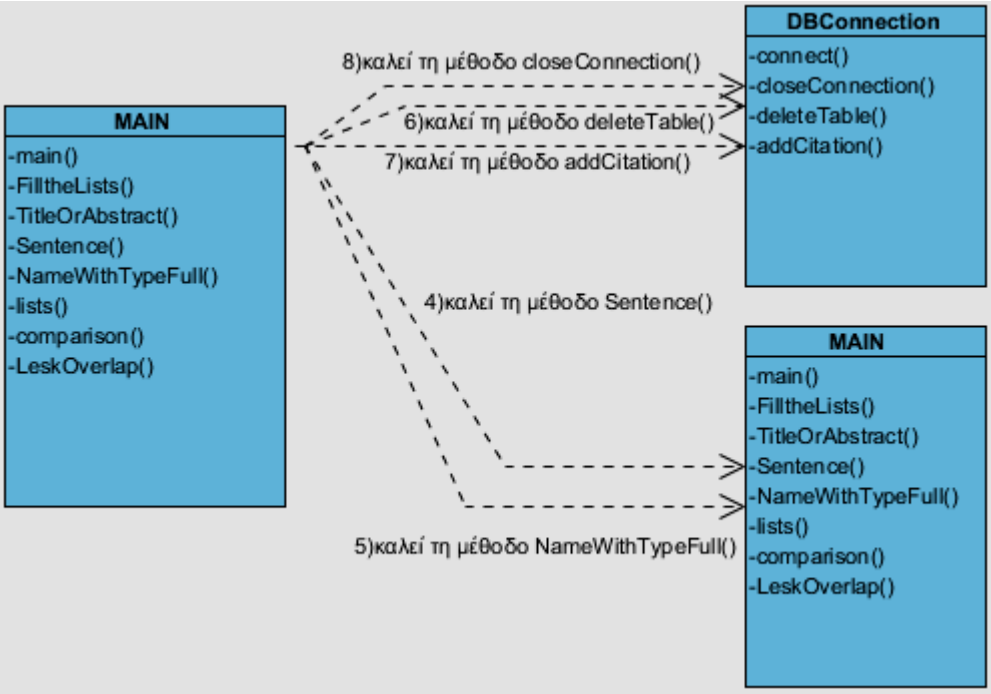
```
public static void NameWithTypeFull() {
      s = "";
      for (int j = 0; j < citation.size(); j++) {
         s = citation.get(j).Sentence;
         String type;
         int l = 0;
         double maxvalue = 0;

         for (int i = 0; i < umlsse.size(); i++) {
            list1.clear();
            list2.clear();

            if (citation.get(j).Term.toLowerCase().equals(umlsse.get(i).Term.toLowerCase())) {
               type = umlsse.get(i).UmlsName+" "+umlsse.get(i).UmlsTypeFull;
               lists(s, list1);
               lists(type, list2);

               comparison(list1, stopwords);
               comparison(list2, stopwords);
```

```
                double d = LeskOverlap(list1, list2);

                if (d > maxvalue) {
                   maxvalue = d;
                   citation.get(j).SenceCodeUsingLeskNameWithTypeFull = umlsse.get(i).SenseCode;
                }
             }
          }
       }
    }
```

Τέλος, έχουμε τη μέθοδο main() που κάνει τη σύνδεση με τη βάση καλώντας τη μέθοδο connect() της κλάσης DBConnection.java. Έπειτα, καλεί τη μέθοδο της ίδιας κλάσης FilltheLists() και γεμίζει την λίστα stopwords με τις συνιθισμένες λέξεις που παίρνει από το αρχείο "stopwords.txt". Μετά καλεί με τη σειρά τις μεθόδους TitleOrAbstract(),Sentence() και NameWithTypeFull(), γεμίζει τον πίνακα CITATIONS της ΒΔ με τα νέα στοιχεία και τέλος κλείνει τη σύνδεση με τη βάση δεδομένων καλώντας τη μέθοδο closeConnection() της κλάσης DBConnection.java.



Εικόνα 21. Η κύρια μέθοδος main() της κλάσης Main και οι μέθοδοι που καλεί (1)



Εικόνα 22. κύρια μέθοδος main() της κλάσης Main και οι μέθοδοι που καλεί (2)

```
public static void main(String[] args) throws FileNotFoundException, IOException, SQLException {
        DBConnection db = new DBConnection();
        db.connect("jdbc:mysql://galaxy.hua.gr:3306/it34", "username", "password");
        FilltheLists();

        BufferedReader in = null;
        in = new BufferedReader(new FileReader("stopwords.txt"));
        String str;
        while ((str = in.readLine()) != null) {
            stopwords.add(str);
        }

        TitleOrAbstract();

        Sentence();

        NameWithTypeFull();

        db.deleteTable();
        for (Citations c : citation) {
            db.addCitation(c);
        }
        db.closeConnection();
    }
```

6.5.4 ΣΥΓΚΡΙΣΗ ΤΟΥ ΛΗΜΜΑΤΟΣ ΤΗΣ ΕΝΝΟΙΑΣ ΜΕ ΤΙΣ ΥΠΟΛΟΙΠΕΣ ΛΕΞΕΙΣ ΤΗΣ ΠΡΟΤΑΣΗΣ

Ο αλγόριθμος Lesk αναθέτει μία έννοια σε έναν όρο που είναι σε ένα συγκεκριμένο πλαίσιο, συγκρίνοντας το λήμμα (ορισμό) κάθε πιθανής έννοιας με τα λήμματα όλων των άλλων λέξεων που βρίσκονται στο πλαίσιο. Η έννοια της λέξης που το λήμμα έχει περισσότερες κοινές λέξεις με τα λήμματα των γειτονικών λέξεων, επιλέγεται ως η καταλληλότερη έννοια. Για παράδειγμα, αν οι αγγλικές λέξεις car(αυτοκίνητο) και tire(λάστιχο) βρίσκονται στην ίδια πρόταση, τότε ελέγχοντας τα λήμματα τους θα δούμε ότι περιέχουν και τα δύο τη λέξη ‘wheel’.

- Car : four *wheel* motor vehicle usually propelled by an internal combustion engine.
- Tire: hoop that covers a *wheel*, usually made of rubber and filled with compressed air

Ο αρχικός αλγόριθμος Lesk συγκρίνει μόνο το λήμμα της αμφήσιμης λέξης με τις λέξεις που βρίσκονται κοντά της. Αυτό είναι ένας σημαντικός περιορισμός γιατί τα περισσότερα λήμματα είναι αρκετά σύντομα και δεν παρέχουν επαρκές λεξιλόγιο ώστε να γίνει καλύτερη αποσαφήνιση. Ενώ, με αυτή την εκτεταμένη περίπτωση του αλγορίθμου Lesk υπάρχουν περισσότερες πιθανότητες να επιλεγεί η σωστότερη έννοια.

7 ΑΠΟΤΕΛΕΣΜΑΤΑ – ΣΥΜΠΕΡΑΣΜΑΤΑ

7.1 ΑΠΟΤΕΛΕΣΜΑΤΑ

Στην παρούσα μελέτη οι τεχνικές επίλυσης της ασάφειας που χρησιμοποιήθηκαν, αξιολογήθηκαν με βάση συγκεκριμένα μέτρα απόδοσης. Τα μέτρα αυτά είναι η πιστότητα (accuracy), η κάλυψη (coverage) και η ακρίβεια (precision). Η πιστότητα (accuracy) ισούται με το πηλίκο των σωστών περιπτώσεων προς τις συνολικές περιπτώσεις. Η κάλυψη (coverage) ισούται με το πηλίκο των περιπτώσεων που έχει γίνει αποσαφήνιση προς τις συνολικές περιπτώσεις και τέλος η ακρίβεια (precision) ισούται με το πηλίκο των σωστών περιπτώσεων προς τις αποσαφηνισμένες περιπτώσεις.

Όλες οι περιπτώσεις επίλυσης αξιολογήθηκαν με τα ίδια μέτρα απόδοσης. Τα αποτελέσματα αναπαριστούνται σε έναν πίνακα. Ο πίνακας των αποτελεσμάτων παρέχει πληροφορίες όπως τα ποσοστά των μέτρων απόδοσης, τον τρόπο επίλυσης, το περιεχόμενο που γίνεται η έρευνα όπως και τα ακριβή αποτελέσματα που προέκυψαν. Από τις 5000 περιπτώσεις που έχουμε οι 4989 καταχωρήθηκαν στη βάση (οι άλλες 11 περιπτώσεις λόγω περιορισμένων πληροφοριών δεν καταχωρήθηκαν). Από αυτές τις 4989 περιπτώσεις οι ερευνητές του UMLS δώσανε αποτέλεσμα μόνο στις 3799 περιπτώσεις. Οι υπόλοιπες έχουν την τιμή ‘null’. Έτσι, σε αυτή τη μελέτη τα αποτελέσματα που έχουμε είναι με βάση τις συνολικά 3799 περιπτώσεις αποσαφήνισης.

	Περιεχόμενο	Accuracy %	Coverage %	Precision %	Σωστά	wsd	total
Επίλυση ασάφειας χωρίς χρήση συχνότερης έννοιας (με διπλότυπα)	Title/Abstract	45.7	79.5	57.3	1733	3023	3799
	Sentence	37.5	67.8	55.3	1427	2576	3799
	Name+TypeFull	19.6	49.3	39.8	747	1874	3799
Επίλυση ασάφειας με χρήση συχνότερης έννοιας(με διπλότυπα)	Title/Abstract	54.9	100	54.9	2089	3799	3799
	Sentence	52.5	100	52.5	1997	3799	3799
	Name+TypeFull	42.9	100	42.9	1630	3799	3799

Επίλυση ασάφειας χωρίς χρήση συχνότερης έννοιας(χωρίς διπλότυπα)	Title/Abstract	41.7	79.5	52.4	1587	3023	3799
	Sentence	36.3	67.8	53.5	1380	2576	3799
	Name+TypeFull	21.6	49.3	43.9	823	1874	3799
Επίλυση ασάφειας με χρήση συχνότερης έννοιας(χωρίς διπλότυπα)	Title/Abstract	51.1	100	51.1	1943	3799	3799
	Sentence	51.3	100	51.3	1950	3799	3799
	Name+TypeFull	44.9	100	44.9	1706	3799	3799
Επίλυση ασάφειας με χρήση μόνο της συχνότερης έννοιας (M1)	-	47.3	100	47.3	1800	3799	3799
Μέθοδος Aggire (MRREL) με το ίδιο σύνολο δεδομένων NLM-WSD	PageRank (ppr)	68.1	100	68.1	-	-	-

Πίνακας 4. Πίνακας αποτελεσμάτων

Ο πίνακας των αποτελεσμάτων περιέχει αρχικά τα αποτελέσματα από την επίλυση των αμφίσημων λέξεων με απλό τρόπο, δηλαδή χωρίς τη χρήση της συχνότερης έννοιας σε περιπτώσεις που δεν έχουμε αποτέλεσμα και με τη χρήση διπλότυπων. Σε αυτή την περίπτωση τα αποτελέσματα που έχουμε από την αυτόματη αποσαφήνιση χωρίζονται σε τρεις περιπτώσεις, που τις είδαμε σε παραπάνω κεφάλαιο. Στην πρώτη περίπτωση (Title/Abstract) που συγκρίνουμε τον τίτλο ή το κείμενο με τον ορισμό (ή λήμμα) του κάθε όρου έχουμε 1733 σωστά αποτελέσματα, δηλαδή το αποτέλεσμα που βγήκε από την αυτόματη αποσαφήνιση είναι ίδιο με το αποτέλεσμα που έδωσαν οι ερευνητές του UMLS. Οι αριθμός των περιπτώσεων που αποσαφηνίστηκαν είναι 3023. Αφού, γνωρίζουμε και τις συνολικές περιπτώσεις (3799) τότε μπορούμε να βγάλουμε και τα μέτρα απόδοσης, τα οποία τα βλέπουμε στον πίνακα. Η πιστότητα (accuracy) ισούται με 45.7 %, η κάλυψη (coverage) με 79.5 % και η ακρίβεια (precision) με 57.3 %. Παρατηρούμε ότι σε σύγκριση με τις άλλες δύο περιπτώσεις περιεχομένου (Sentence και Name+TypeFull) τα αποτελέσματα είναι καλύτερα. Στην περίπτωση με περιεχόμεο τα Name+TypeFull

ήταν αναμενόμενα τα χαμηλα ποσοστά στα μέτρα απόδοσης γιατί παίρνουμε ως μέτρο σύγκρισης τον όρο-λέξη και τον τύπο της έννοιας του, χωρίς να συμπεριλάβουμε τον ορισμό της έννοιας που παίρνουμε από τη βάση του UMLS. Στις άλλες δύο περιπτώσεις παρατηρούμε ότι στην περίπτωση που συγκρίνουμε μόνο την πρόταση με τον ορισμό της έννοιας έχουμε χαμηλότερα αποτελέσματα από αυτά που παίρνουμε στην πρώτη περίπτωση. Άρα συμπεραίνουμε ότι παίζουν σημαντικό ρόλο στα αποτελέσματα και λέξεις που δεν βρίσκονται στην ίδια πρόταση με την αμφίσημη λέξη.

Στην περίπτωση της επίλυσης της ασάφειας με χρήση της συχνότερης έννοιας, δηλαδή την αντικατάσταση των περιπτώσεων που δεν έχουν αποτέλεσμα ή έχουν την τιμή 'null' με την έννοια M1, παρατηρούμε μία αύξηση στα ποσοστά σε σύγκριση με την προηγούμενη επίλυση. Αρχικά βλέπουμε ότι το ποσοστό της κάλυψης έγινε 100% αφού συμπληρώθηκαν τα κενά πεδία με την τιμή M1. Άρα αυτομάτως η πιστότητα και η ακρίβεια έχουν την ίδια τιμή. Τα ποσοστά και των τριών περιεχομένων αυξήθηκαν χωρίς να έχουμε καμία μεγάλη αλλαγή. Έτσι, η περίπτωση με περιεχόμενο το Title/Abstract έχει το μεγαλύτερο ποσοστό (54.9).

Συνεχίζοντας να κάνουμε αλλαγές στο πρόγραμμα, δοκιμάσαμε την επίλυση των ασαφειών με περιεχόμενα που δεν περιέχουν διπλότυπα. Σε αυτήν την περίπτωση παρατηρούμε τα ίδια ποσοστά κάλυψης για όλα τα περιεχόμενα. Όμως, έχουμε μείωση στην πιστότητα και την ακρίβεια για τις δύο πρώτες περιπτώσεις με περιεχόμενα τα Title/Abstract και Sentence αντίστοιχα, που σημαίνει ότι τα διπλότυπα επηρεάζουν σε θετικό βαθμό τα αποτελέσματα όσον αφορά τα κείμενα. Αντιθέτως, στην περίπτωση με περιεχόμενο σύγκρισης το Name+TypeFull παρατηρούμε μία μικρή αύξηση στα ποσοστά, που σημαίνει ότι τα διπλότυπα επηρεάζουν αρνητικά τα αποτελέσματα όσον αφορά τις λέξεις.

Στην τελευταία αλλαγή στο πρόγραμμα που κάναμε αφαιρώντας τα διπλότυπα, προσθέσαμε την χρήση της συχνότερης έννοιας (M1) στις περιπτώσεις των πεδίων που είναι κενά ή περιέχουν την τιμή 'null'. Παρατηρώντας αυτόν τον τρόπο επίλυσης βλέπουμε ότι και πάλι το ποσοστό της κάλυψης είναι 100% και τα ποσοστά από τα άλλα δύο μέτρα απόδοσης έχουν μία μικρή αύξηση σε σχέση με την προηγούμενη περίπτωση.

Τέλος, προσθέσαμε στον πίνακα και την περίπτωση επίλυσης της ασάφειας μόνο με τη χρήση της συχνότερης έννοιας (M1), δηλαδή όλες οι περιπτώσεις να έχουν ως επιλεγόμενη έννοια τη M1. Σε αυτόν τον τρόπο επίλυσης παρατηρούμε ότι έχουμε πλήρη κάλυψη (100%), άρα η πιστότητα και η ακρίβεια είναι το ίδιο, και το ποσοστό της ακρίβειας είναι 47.3%. Εδώ πρέπει να τονίσουμε ότι κάποιες από τις μεθόδους έχουν χειρότερα αποτελέσματα από αυτό εδώ το ποσοστό. Αυτό το φαινόμενο είναι αρκετά συχνό σε περιπτώσεις αποσαφήνισης (wsd), δηλαδή πολύ απλοϊκες μέθοδοι να πηγαίνουν χειρότερα από τη μέθοδο χρησιμοποίησης της συχνότερης έννοιας. Είναι λογικό να πηγαίνει καλά η περίπτωση της συχνότερης έννοιας (Most frequent sense) μιας και βασίζεται σε γνώση που έχουμε από μεγάλα σύνολα κειμένων. Τοποθετήσαμε στον πίνακα και το μεγαλύτερο αποτέλεσμα από τη μελέτη του Aggire (2010)[7] ως μέτρο σύγκρισης, το οποίο είναι μεγαλύτερο από οτι τα δικά μας αποτελέσματα.

7.2 ΣΥΜΠΕΡΑΣΜΑΤΑ

Στην συγκεκριμένη πτυχιακή εργασία πραγματοποιήθηκε μία μελέτη στον τομέα της αποσαφήνισης βιοϊατρικών λέξεων που είχε σκοπό την ανάπτυξη ενός αλγορίθμου αυτόματης αποσαφήνισης και τη σύγκριση των αποτελεσμάτων του με αυτά που δίνονται από τους ερευνητές του UMLS. Πιο συγκεκριμένα υπήρχαν 50 αμφίσημες λέξεις με 100 περιπτώσεις για κάθε λέξη. Επεξεργάζοντας τα στοιχεία που μας δόθηκαν από τη βάση δεδομένων του UMLS και δημιουργώντας ένα αλγόριθμο αυτόματης αποσαφήνισης που βασίστηκε στον απλοποιημένο αλγόριθμο Lesk παράγαμε ορισμένα αποτελέσματα. Με βάση αυτά τα αποτελέσματα παρατηρούμε ότι το καλύτερο ποσοστό πιστότητας το είχαμε στην περίπτωση επίλυσης (54.9) που έχουμε διπλότυπα και χρησιμοποιούμε τη συχνότερη έννοια (M1) σε περιπτώσεις που δεν είχαμε αποτέλεσμα χρησιμοποιώντας τον αλγόριθμο. Σε αυτή την περίπτωση η κάλυψη της αποσαφήνισης είναι στο 100%. Το καλυτερο ποσοστό ακρίβειας το

[7] Agirre, E.; Soroa, A.; and Stevenson, M. (2010) “Graph-based Word Sense Disambiguation of biomedical documents”

είχαμε στην πρώτη περίπτωση που έχουμε διπλότυπα αλλά δεν χρησιμοποιούμε τη συχνότερη έννοια (M1). Για ένα δείγμα με κάλυψη 79.5% η ακρίβεια βρίσκεται στο 57.3%.

Έτσι, όπως φαίνεται στην παρούσα πτυχιακή εργασία μπορούν να βγούν συμπεράσματα μέσα από τα αποτελέσματα που έχουμε. Ακόμη, η εργασία αυτή έχει περιθώρια προέκτασης με άλλους τρόπους επίλυσης, όπως και αυτούς που αναφέραμε παραπάνω. Για παράδειγμα η περίπτωση που παίρνουμε τη ρίζα από κάθε λέξη και τις συγκρίνουμε είναι πολύ ενδιαφέρον ως προς τα αποτελέσματα που θα προκύψουν. Τέλος, να τονίσουμε ότι θα ήταν σημαντικό οι ερευνητές να ασχοληθούν με τη βελτίωση των περιεχομένων των βάσεων δεδομένων του UMLS ώστε να έχουμε μελλοντικά καλύτερα αποτελέσματα.

8 ΑΝΑΦΟΡΕΣ

Agirre, E.; Lopez de Lacalle, A.; Soroa, A. (2009) "Knowledge-based WSD on Specific Domains: Performing better than Generic Supervised WSD" *Proc. of IJCAI*

Gelbukh, A., Sidorov, G.. (2004) Automatic resolution of ambiguity of word senses in dictionary definitions (in Russian). J. Nauchno-Tehnicheskaya Informaciya (NTI), ISSN 0548-0027, ser. 2, N 3, 2004, pp. 10–15.

Bar-Hillel, Y. (1964). *Language and information*. Reading, MA: Addison-Wesley.

Agirre, E.; Soroa, A.; and Stevenson, M. (2010) “Graph-based Word Sense Disambiguation of biomedical documents”

Fellbaum, C. (1998). “WordNet an electronic lexical database”. MIT Press.

Fellbaum, C (1997) "Analysis of a handwriting task" *Proc. of ANLP-97 Workshop on Tagging Text with Lexical Semantics: Why, What, and How? Washington D.C., USA HLT Student Research Workshop* . Association for Computational Linguistics.

Vasilescu, F.; Langlais, P.; and Lapalme, G. (2004). Evaluating Variants of the Lesk Approach for Disambiguating Words. LREC, Portugal.

Humphreys, L. *et al.* (1998). The Unified Medical Language System: an Informatics

Kilgarriff and Rosenzweig, J. (2000). English SENSEVAL:Report and Results. In Proceedings of the 2nd International Conference on Language Resourcesand Evaluation, LREC, Athens, Greece.

Lenat, D. (2007) "Computers versus Common Sense". Retrieved 2012-09-27. (GoogleTachTalks on YouTube)

Lenat, D.; Guha, R. V. (1989). Building Large Knowledge-Based Systems, Addison-Wesley

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation, pages 24-26, New York, NY, USA. ACM.

Weeber, M. PhD; Mork, J, Msc; Aronson, A, PhD; (2001) "Developing a Test Collection for Biomedical Word Sense Disambiguation"

McInnes, B. (2008). An unsupervised vector approach to biomedical term disambiguation: integrating UMLS and Medline. In *Proceedings of the ACL-08: HLT Student Research Workshop* . Association for Computational Linguistics, Columbus, Ohio, pp. 49–54.

Navigli, R. (2006). Meaningful Clustering of Senses Helps Boost Word Sense Disambiguation Performance. Proc. of the 44th Annual Meeting of the Association for Computational Linguistics joint with the 21st International Conference on Computational Linguistics (COLING-ACL 2006), Sydney, Australia.

Navigli, R.; Crisafulli, G. (2010) Inducing Word Senses to Improve Web Search Result Clustering. Proc. of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010), MIT Stata Center, Massachusetts, USA.

Navigli, R.; Litkowski, K.; Hargraves, O. (2007). SemEval-2007 Task 07: Coarse-Grained English All-Words Task. Proc. of Semeval-2007 Workshop (SemEval), in the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), Prague, Czech Republic.

Navigli, R.; Lapata, M. (2010) An Experimental Study of Graph Connectivity for Unsupervised Word Sense Disambiguation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 32(4), IEEE Press, 2010.

Navigli, R.; Velardi, P. (2005). Structural Semantic Interconnections: a Knowledge-Based Approach to Word Sense Disambiguation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 27(7).

Palmer, M.; Babko-Malaya, O.; and Dang, H. T.. (2004). Different sense granularities for different applications. In Proceedings of the 2nd Workshop on Scalable Natural Language Understanding Systems in HLT/NAACL (Boston, MA).

Locke, W.N.; Booth, D.A., eds. (1955). "Translation". Machine Translation of Languages. Cambridge, Massachusetts: MIT Press. pp. 15–23. ISBN 0-8371-8434-7.

Snow, R.; Prakash, S.; Jurafsky, D.; Ng, A. (2007) Learning to Merge Word Senses, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL).

Snyder, B.; Palmer, M. (2004). The English all-words task. In Proc. of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3), Barcelona, Spain.

Weaver, W (1949). "Translation". In Locke, W.N.; Booth, A.D.. *Machine Translation of Languages: Fourteen Essays*. Cambridge, MA: MIT Press.

Wilks, Y.; Slator, B.; Guthrie, L. (1996). Electric Words: dictionaries, computers and meanings. Cambridge, MA: MIT Press.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In Proc. of the 33rd Annual Meeting of the Association for Computational Linguistics.

Fleiss, J.L. (1971), Measuring Nominal Scale Agreement Among Many Raters. Psychol. Bull., 76, 378-382.